

ISSAC 2008



Association for
Computing Machinery

Advancing Computing as a Science & Profession





**Association for
Computing Machinery**

Advancing Computing as a Science & Profession

**The Association for Computing Machinery
2 Penn Plaza, Suite 701
New York, New York 10121-0701**

Copyright © 2008 by the Association for Computing Machinery, Inc. (ACM). Permission to make digital or hard copies of portions of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyright for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permission to republish from: Publications Dept., ACM, Inc. Fax +1 (212) 869-0481 or <permissions@acm.org>.

For other copying of articles that carry a code at the bottom of the first or last page, copying is permitted provided that the per-copy fee indicated in the code is paid through the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923.

Notice to Past Authors of ACM-Published Articles

ACM intends to create a complete electronic archive of all articles and/or other material previously published by ACM. If you have written a work that has been previously published by ACM in any journal or conference proceedings prior to 1978, or any SIG Newsletter at any time, and you do NOT want this work to appear in the ACM Digital Library, please inform permissions@acm.org, stating the title of the work, the author(s), and where and when published.

ISBN: 978-1-59593-904-3

Printed in the USA

Foreword

This volume contains the proceedings of the 21st annual meeting of the International Symposium on Symbolic Computation, ISSAC 2008. The first ISSAC took place in 1988, but it inherited the legacies of a number of earlier meetings, SYMSAM, SYMSAC, EUROSAM, EUROCAL, stretching back to 1966. In this extended tradition, ISSAC 2008 is the 33rd meeting. The meeting took place from the 20th to 23rd of July 2008 in Hagenberg, a pretty village 20 km north-east of Linz in Austria.

The topics of the conference series include, but are not limited to:

- **Algorithmic Mathematics:** Algebraic, symbolic and symbolic-numeric algorithms. Simplification, function manipulation, summation, integration, linear algebra, number theory, polynomial/differential/difference equations, group and invariant theory, geometric computing.
- **Computer Science:** Theoretical and practical problems in symbolic computation. Systems, problem solving environments, user interfaces, software, libraries, concrete analysis, parallel/distributed computing and programming languages, benchmarking, theoretical and practical complexity, automatic differentiation, code generation, mathematical data structures and exchange protocols.
- **Applications:** Problem treatments using algebraic, symbolic or symbolic-numeric computation in an essential or a novel way. Engineering, economics and finance, physical and biological sciences, computer science, logic, mathematics and education.

The ISSAC 2008 program contained invited talks, contributed papers, tutorials, poster sessions and software exhibitions. ISSAC was also part of *RISC Summer 2008*, and a number of related conferences were held at the same location and nearby in time. This volume contains all contributed papers as well as abstracts of the invited talks and tutorials.

For ISSAC 2008, a total of 99 papers were accepted for review and each was distributed to members of the program committee and external reviewers. On average, 2.6 referee reports were obtained for each submission, and 40 papers were selected for presentation. They are representative of the many topics of research that make up the field of computer algebra. We wish to thank all the researchers who contributed papers for review, and to thank the many reviewers who generously gave their time to the selection process.

The success of ISSAC 2008 depends on many people, many of whom are listed in the following pages. To them, and to the rest of the community who contributed to ISSAC in some way, we extend our thanks. Finally, we thank the sponsors of conference, who are also listed later in these pages, and all to members of our host institution, the Research Institute for Symbolic Computation, RISC Linz, which celebrates 20 years this year.

Rafael Sendra

General Chair

Universidad de Alcalá
Spain

Laureano Gonzalez-Vega

Program Chair

Universidad de Cantabria
Spain

Franz Winkler

Local Arrangements Chair

Johannes Kepler Universität
Austria

David Jeffrey

Proceedings Editor

The University of Western Ontario
Canada

The cover, designed by Johannes Middeke, shows Castle Hagenberg and scenes of Linz.

Table of Contents

ISSAC 2008 Conference Organization	viii

Invited Talks

• Power Series and Linear Difference Equations	1
Sergei A. Abramov (<i>Computing Centre of Russian Academy of Sciences</i>)	
• Digital Atlases and Difference Forms	3
Elizabeth L. Mansfield (<i>University of Kent</i>)	
• Can We Create a Viable Free Open Source Alternative to Magma, Maple, Mathematica and Matlab?	5
William Stein (<i>University of Washington</i>)	

Contributed Papers

• Towards a Computer Algebraic Algorithm for Flat Output Determination	7
Felix Antritter (<i>Universität der Bundeswehr München</i>), Jean Lévine (<i>École des Mines de Paris</i>)	
• Regular Systems of Linear Functional Equations and Applications	15
Moulay A. Barkatou (<i>Université de Limoges</i>), Gary Broughton, Eckhard Pflügel (<i>Kingston University</i>)	
• Products of Ordinary Differential Operators by Evaluation and Interpolation	23
Alin Bostan, Frédéric Chyzak, Nicolas Le Roux (<i>INRIA Paris-Rocquencourt</i>)	
• Implicitization and Parametrization of Quadratic Surfaces with One Simple Base Point	31
Xuhui Wang, Falai Chen, Jiansong Deng (<i>University of Science and Technology of China</i>)	
• Solving Differential Equations in Terms of Bessel Functions	39
Ruben Debeerst (<i>Universität Kassel</i>), Mark van Hoeij (<i>Florida State University</i>), Wolfram Koepf (<i>Universität Kassel</i>)	
• On the Computation of the Topology of a Non-Reduced Implicit Space Curve	47
Diatta Niang Daouda (<i>University of Limoges and INRIA Sophia-Antipolis</i>), Bernard Mourrain (<i>Sophia-Antipolis</i>), Olivier Ruatta (<i>University of Limoges</i>)	
• Gröbner-free Normal Forms for Boolean Polynomials	55
Michael Brickenstein (<i>Mathematisches Forschungsinstitut Oberwolfach</i>), Alexander Dreyer (<i>Fraunhofer Institute for Industrial Mathematics</i>)	
• Q-adic Transform Revisited	63
Jean-Guillaume Dumas (<i>Université de Grenoble</i>)	
• Computing the Global Optimum of a Multivariate Polynomial over the Reals	71
Mohab Safey El Din (<i>INRIA Paris-Rocquencourt Center</i>)	
• Classification of the Perspective-Three-Point Problem, Discriminant Variety and Real Solving Polynomial Systems of Inequalities	79
Jean-Charles Faugère, Guillaume Moroz, Fabrice Rouillier, Mohab Safey El Din (<i>INRIA Paris-Rocquencourt Center</i>)	
• Complete Subdivision Algorithms, II: Isotopic Meshing of Singular Algebraic Curves	87
Michael Burr (<i>New York University</i>), Sung Woo Choi (<i>Duksung Women University</i>), Benjamin Galehouse, Chee K. Yap (<i>New York University</i>)	
• A Pommaret Division Algorithm for Computing Gröbner Bases in Boolean Rings	95
Vladimir P. Gerdt, Mikhail V. Zinin (<i>Joint Institute for Nuclear Research</i>)	
• On Lacunary Polynomial Perfect Powers	103
Mark Giesbrecht, Daniel S. Roche (<i>University of Waterloo</i>)	

• Plane Geometry and Convexity of Polynomial Stability Regions	111
Didier Henrion (<i>University of Toulouse</i>), Michael Šebek (<i>Czech Technical University in Prague</i>)	
• An Algorithm for Finding Symmetric Gröbner Bases in Infinite Dimensional Rings	117
Matthias Aschenbrenner (<i>University of California at Los Angeles</i>), Christopher J. Hillar (<i>Texas A&M University</i>)	
• Moment Matrices, Trace Matrices and the Radical of Ideals	125
Itzhit Janovitz-Freireich (<i>North Carolina State University</i>), Ágnes Szántó (<i>North Carolina State University</i>), Bernard Mourrain (<i>GALAAD, INRIA</i>), Lajos Rónyai (<i>MTA SZTAKI</i>)	
• Integration of Algebraic Functions: A Simple Heuristic for Finding the Logarithmic Part	133
Manuel Kauers (<i>Johannes Kepler Universität</i>)	
• Expressing a Fraction of Two Determinants as a Determinant	141
Erich Kaltofen (<i>North Carolina State University</i>), Pascal Koiran (<i>Université de Lyon</i>)	
• Symbolic Optimization of Algebraic Functions	147
Masaaki Kanno (<i>Japan Science and Technology Agency</i>), Kazuhiro Yokoyama (<i>Rikkyo University</i>), Hirokazu Anai (<i>Fujitsu Laboratories Ltd.</i>), Shinji Hara (<i>The University of Tokyo</i>)	
• Exact Certification of Global Optimality of Approximate Factorizations Via Rationalizing Sums-Of-Squares with Floating Point Scalars	155
Erich Kaltofen (<i>North Carolina State University</i>), Bin Li (<i>Key Laboratory of Mathematics Mechanization, AMSS</i>), Zhengfeng Yang (<i>North Carolina State University</i>), Lihong Zhi (<i>Key Laboratory of Mathematics Mechanization, AMSS</i>)	
• Numerical Primary Decomposition	165
Anton Leykin (<i>University of Minnesota</i>)	
• Computational D-module Theory with SINGULAR, Comparison with Other Systems and Two New Algorithms	173
Viktor Levandovskyy (<i>RWTH Aachen</i>), Jorge Martín Morales (<i>University of Zaragoza</i>)	
• Complex Brackets, Balanced Complex Differences, and Applications in Symbolic Geometric Computing	181
Hongbo Li, Lei Huang (<i>Chinese Academy of Sciences</i>)	
• The Complete Root Classification of a Parametric Polynomial on an Interval	189
Songxin Liang, David J. Jeffrey, Marc Moreno Maza (<i>The University of Western Ontario</i>)	
• Hilbert’s Nullstellensatz and an Algorithm for Proving Combinatorial Infeasibility	197
Jesús A. De Loera (<i>University of California at Davis</i>), J. Lee (<i>IBM T.J. Watson Research Center</i>), Peter N. Malkin, Susan Margulies (<i>University of California at Davis</i>)	
• When does $\langle T \rangle$ equal $\text{sat}(T)$?	207
François Lemaire (<i>Université de Lille</i>), Marc Moreno Maza, Wei Pan, Yuzhen Xie (<i>University of Western Ontario</i>)	
• Deciding Polynomial-Exponential Problems	215
Melanie Achatz (<i>University of Passau</i>), Scott McCallum (<i>Macquarie University</i>), Volker Weispfenning (<i>University of Passau</i>)	
• On Generalized LN-Surfaces in 4-Space	223
Martin Peternell, Boris Odehnal (<i>University of Technology, Vienna</i>)	
• A Rational Decomposition— Lemma for Systems of Linear Differential—Algebraic Equations	231
Eckhard Pflügel (<i>Kingston University</i>)	
• Good Reduction of Puiseux Series and Complexity of the Newton-Puiseux Algorithm over Finite Fields	239
Adrien Poteaux, Marc Rybowicz (<i>Université de Limoges</i>)	
• Multi-modular Algorithm for Computing the Splitting Field of a Polynomial	247
Guénaél Renault (<i>INRIA and University Paris 06</i>), Kazuhiro Yokoyama (<i>Rikkyo University</i>)	

• Linearizability of 1:-3 Resonant System with Homogeneous Cubic Nonlinearities	255
Zhibek Kadyrsizova (<i>al-Farabi Kazakh National University</i>), Valery G. Romanovski (<i>University of Maribor</i>)	
• Integro-Differential Polynomials and Operators	261
Markus Rosenkranz, Georg Regensburger (<i>Austrian Academy of Sciences</i>)	
• Power Series Composition and Change of Basis	269
Alin Bostan, Bruno Salvy (<i>INRIA Rocquencourt</i>), Éric Schost (<i>University of Western Ontario</i>)	
• Loewy Decomposition of Third-Order Linear PDE's in the Plane	277
Dima Grigoriev (<i>Université de Rennes</i>), Fritz Schwarz (<i>Fraunhofer Gesellschaft, Institut SCAI</i>)	
• The Nearest Polynomial with a Zero in a Given Domain from a Geometrical Viewpoint	287
Hiroshi Sekigawa (<i>Nippon Telegraph and Telephone Corporation</i>)	
• Moving Frames for Laplace Invariants	295
Ekaterina Shemyakova (<i>J. Kepler University</i>), Elizabeth L. Mansfield (<i>The University of Kent</i>)	
• Real Root Isolation for Exp-Log Functions	303
Adam Strzeboński (<i>Wolfram Research Inc.</i>)	
• Gröbner Bases for Public Key Cryptography	315
Massimo Caboara, Fabrizio Caruso, Carlo Traverso (<i>Università di Pisa</i>)	
• Computing the Multiplicity Structure from Geometric Involutive Form	325
Xiaoli Wu, Lihong Zhi (<i>Key Laboratory of Mathematics Mechanization, AMSS</i>)	
Tutorials	
• Exact Algorithms and Software in Optimization and Polyhedral Computation	333
Komei Fukuda (<i>ETH Zurich</i>)	
• Algorithmic Invariant Theory	335
Gregor Kemper (<i>Technische Universität München</i>)	
• Algebraic Algorithms and Coding Theory	337
Madhu Sudan (<i>Massachusetts Institute of Technology</i>)	
Author Index	338

ISSAC 2008 Conference Organization

General Chair	J. Rafael Sendra, Universidad de Alcalá, Spain
Program Committee Chair	Laureano Gonzalez-Vega, Universidad de Cantabria, Spain
Local Arrangements Chair	Franz Winkler, RISC, Universität Linz, Austria
Treasurer	Franz Lichtenberger, RISC, Universität Linz, Austria
Publicity Chair	Manfred Minimair, Seton Hall University, USA
Poster Committee Chair	Austin Lobo, Washington College, USA
Proceedings Editor	David Jeffrey, University of Western Ontario, Canada
Tutorial Chair	Ioannis Z. Emiris, Nat. Kapodistrian University of Athens, Greece
Software Exhibitions	Michael Monagan, Simon Fraser University, Canada
Webmaster	Wolfgang Windsteiger, RISC, Universität Linz, Austria
Program Committee	John Abbott, Università degli Studi di Genova, Italy Hirokazu Anai, Fujitsu Labs Ltd., Japan Laurent Buse, INRIA Sophia Antipolis, France Alicia Dickenstein, Universidad de Buenos Aires, Argentina Xiao-Shan Gao, Chinese Academy of Sciences, China Laureano Gonzalez-Vega, Universidad de Cantabria, Spain (Chair) Willem de Graaf, Università degli Studi di Trento, Italy Mark van Hoeij, Florida State University, USA Joris van der Hoeven, Université Paris-Sud, France Evelyne Hubert, INRIA Sophia Antipolis, France Alexander Hulpke, Colorado State University, USA M'hammed El Kahoui, Université Cadi Ayyad, Morocco Manuel Kauers, RISC, Universität Linz, Austria Jürgen Klüners, Heinrich-Heine-Universität Düsseldorf, Germany Viktor Levandovskyy, RWTH Aachen, Germany Steve Linton, University of St. Andrews, UK Greg Reid, ORCCA, University of Western Ontario, Canada Igor Shparlinski, Macquarie University, Australia Arne Storjohann, University of Waterloo, Canada Lihong Zhi, Chinese Academy of Sciences, China
Poster Committee	Maki Iwami, Osaka University of Economics and Law, Japan Wen-shin Lee, University of Antwerp, Belgium Austin Lobo, Washington College, USA; (Chair) Clement Pernet, University of Washington, USA Ekaterina Shemyakova, RISC, Universität Linz, Austria Min Wu, East China Normal University, China
Local Arrangements Committee	Ralf Hemmecke, RISC, Universität Linz, Austria Franz Lichtenberger, RISC, Universität Linz, Austria Wolfgang Schreiner, RISC, Universität Linz, Austria Wolfgang Windsteiger, RISC, Universität Linz, Austria Franz Winkler, RISC, Universität Linz, Austria; (Chair)
ISSAC Steering Committee	Jeremy Johnson, Drexel University, USA Xiao-Shan Gao, Chinese Academy of Sciences, China Andre Galligo, Universite Nice, France Gerhard Hiss, Universität Aachen, Germany Yosuke Sato, Tokyo University of Science, Japan Daniel Lichtblau, Wolfram Research, USA

ISSAC thanks the following external reviewers

John Abbott	Hirotschi Abo	Amod Agashe	Lionel Alberti
Juan Gerardo Alcazar	Amir Amiraslani	Ian Anderson	Phillip Barry
Saugata Basu	Christoph Benzmueller	Isabel Bermejo	Dan Bernstein
Anna Bigatti	Dario Bini	Paola Boito	Alin Bostan
Lyonell Boulton	Russell Bradford	Christopher Brown	Clemens Bruschek
Peter BuerGISser	Yann Bugeaud	Guillaume Cheze	Massimo Caboara
Emrah Cakcak	Andrea Caranti	Jacques Carette	Fabrizio Caruso
Francisco-Jesus Castro-Jimenez	Kai-Cheong Chan	Edgardo Cheb-Terrab	Shaoshi Chen
Guoting Chen	Jinsan Cheng	Richard Churchill	Frederic Chyzak
Carlos Cid	Thomas Cluzeau	David Cox	Teresa Crespo
Felipe Cucker	Juan Cuesta	Domenico D'Alessandro	Carlos D'Andrea
Xavier Dahan	James H. Davenport	Barry Dayton	Blandine Debraize
Gema M. Diaz-Toca	Claus Diem	Mohab Safey El Din	Tor Dokken
Leo Dorst	Tom Dowling	Raouf Dridi	Jean-Guillaume Dumas
Dominique Duval	Roy Dyckhoff	Mohamed Elkadi	Ioannis Emiris
Jan-Hendrik Evertse	Jean-Charles Faugere	Claus Fieker	Elisabetta Fortuna
Andre Galligo	Pierrick Gaudry	Luca Gemignani	Mark Giesbrecht
Marc Giusti	Ronald Goldman	Maria A. Gomez-Molleda	Laureano Gonzalez-Vega
Stef Graillat	Elena Gryazina	Li Guo	Kevin Hare
Andre Heck	Ralf Hemmecke	Didier Henrion	Willy Hereman
Mark Hickman	Jerome Hoffman	Dennis Hofheinz	Lei Hu
Evelyne Hubert	Thomas Hulin	Alexander Hulpke	David Jao
Abdul Salam Jarrah	Bert Juettler	Francoise Jung	Hiroshi Kai
Erich Kaltofen	Masaaki Kanno	Lena Kartashova	Andrei Kelarev
Michael Kerber	Wolfram Koepf	Irina Kogan	Alexander Kononov
Ilias Kotsireas	Werner Krandick	Rimvydas Krasauskas	Martin Kreuzer
Bernhard Kroen	Wolfgang Kuchlin	George Labahn	Reinhard Laubenbacher
Monique Laurent	Gregoire Lecerf	Wen-shin Lee	Viktor Levandovskyy
Robert Lewis	Anton Leykin	Ian Lisle	Daniel Loebenberger
Arne Lorenz	Frank Luebeck	Johan Lyhagen	Elizabeth Mansfield
Thomas Markwig	Guillermo Matera	Scott McCallum	Maurice Mignotte
Izumi Miyamoto	Michael Monagan	Teo Mora	Guillaume Moroz
Bernard Mourrain	Katsusuke Nabeshima	Kosaku Nagasaka	Max Neunhoffer
Anonymous Nobody	Masayuki Noro	Andrew Odlyzko	Francois Ollivier
Peter Olver	Fioravante Patrone	Roman Pearce	Ludovic Perret
Chris Peterson	Michel Petitot	Marko Petkovsek	Simon Plantinga
Renato Portugal	Victoria Powers	Alban Quadrat	Stefan Ratschan
Georg Regensburger	Nargol Rezvani	Dan Richardson	Daniel Robertz
Daniel Roche	J. Maurice Rojas	Lajos Ronyai	Markus Rosenkranz
Nicolas Le Roux	Olivier Ruatta	Yosuke Sato	Raffaele Scapellato
Arnold Schonhage	Hans Schoenemann	Eric Schost	Wolfgang Schreiner
Alexandre Sedoglavic	Werner Seiler	Hiroshi Sekigawa	Johan Simon Seland
Brigitte Servatius	David Sevilla	Fabien Seyfert	Douglas Shafer
Ekaterina Shemyakova	Kiyoshi Shirayanagi	Jamshid Shokrollahi	Michael Singer
Michael F. Singer	Gerald Sommer	Jonathan Sorenson	Doru Stefanescu
Damien Stehle	Adam Strzebonski	Thomas Sturm	Bernd Sturmfels
Seth Sullivan	Luis F. Tabera	Carlo Traverso	Philippe Trebuchet
Felix Ulmer	Bruno Vallet	Michael Vrahatis	Jacques-Arthur Weil
Uli Walther	Dingkang Wang	William Webb	Julian West
Oliver Wienand	Thomas Wolf	Wenyuan Wu	Jean-Claude Yakoubsohn
Zhenya Yan	Tjalling Ypma	Pei Yu	Chunming Yuan
Zhonggang Zeng	Wenhui Zhang	Eugene Zima	Paul Zimmermann

ISSAC 2008 is organized in cooperation with ACM-SIGSAM, and is sponsored by

Austrian Ministry of Science



Government of Upper Austria



Johannes Kepler University,
Linz, Austria



JOHANNES KEPLER
UNIVERSITY LINZ | JKU

Departamento de Matemáticas
(Universidad de Alcalá)



Maplesoft, Waterloo, Canada



Association of Computing Machinery



Special Interest Group in Symbolic
and Algebraic Manipulation



Advancing Computing as a Science & Profession

Power Series and Linear Difference Equations

[Invited Talk]

Sergei A. Abramov
Computing Centre
of Russian Academy of Sciences
Vavilova 40, 119991, Moscow GSP-1, Russia
sabramov@ccas.ru

Categories and Subject Descriptors

I.1.2 [Symbolic And Algebraic Manipulation]: Algorithms—Algebraic algorithms

General Terms

Algorithms, Theory

Keywords

Linear difference equation with polynomial coefficients, power series, sequential solution, subanalytic solution

Extended abstract

A doubly infinite complex number sequence

$$(c_n), \quad n \in \mathbb{Z}, \quad (1)$$

will be called a *sequential* solution of a difference equation of the form

$$a_d(z)y(z+d) + \dots + a_1(z)y(z+1) + a_0(z)y(z) = 0, \quad (2)$$

$a_1(z), a_2(z), \dots, a_{d-1}(z) \in \mathbb{C}[z]$, $a_0(z), a_d(z) \in \mathbb{C}[z] \setminus \{0\}$, if

$$a_d(n)c_{n+d} + \dots + a_1(n)c_{n+1} + a_0(n)c_n = 0$$

for all $n \in \mathbb{Z}$. A sequential solution (1) will be called *subanalytic* sequential (or just subanalytic, for short), if equation (2) has a solution in the form of a single-valued analytic function $f: \mathbb{C} \rightarrow \mathbb{C}$ such that $c_n = f(n)$ for all $n \in \mathbb{Z}$.

We discuss a way to compute the values of elements of a subanalytic solution of equation (2) at arbitrary integer points, in particular at the points where the polynomial $a_d(z-d)a_0(z)$ vanishes.

We show that the dimension of the $\mathbb{C}$ -linear space of all sequential solutions of (2) is $\geq d$, and for any integer $m \geq d$ there exists an equation of the form (2) such that this dimension is m . The dimension of the space of all subanalytic

solutions is equal to d , and, generally speaking, not all sequential solutions are subanalytic.

We also show that the application of the discrete Newton-Leibniz formula combined with known computer algebra algorithms for indefinite summation of elements of sequential solutions always gives the correct result in the case when a solution under consideration is subanalytic. However the result can be incorrect for a sequential solution that is not subanalytic. This “phenomenon” can be explained quite easily. If a summation algorithm like the Gosper algorithm ($d = 1$) or the Accurate summation algorithm (an arbitrary d) succeeds on (2) then the output of such a summation algorithm is not the sum of elements of a sequential solution but a *summing operator* R . For the difference operator L , which corresponds to (2), these algorithms try to construct R such that the operator $(E - 1) \circ R - 1$, where E is the shift operator, is right divisible by L . So

$$(E - 1) \circ R = 1 + M \circ L$$

for some operator M . However in the case when such a summation algorithm succeeds on L , the operators R, M have rational-function coefficients which can have integer poles.

So one faces the problem of the correct application of such an operator R to a given solution of $L(y) = 0$. Examples show that such an application is not always possible. What we prove, is in fact that if the solution of $L(y) = 0$ is subanalytic then such an application is possible.

Notice that the condition of subanalyticity is not a necessary condition for the correct applicability of the summing operator: there exist examples where the dimension of the space of “nice” sequential solutions is $> d$.

If a meromorphic solution $f(z)$ of equation (2) has some integer poles, then nevertheless it is possible to map $f(z)$ into a sequential solution (the *bottom* of $f(z)$) of (2) such that the summation algorithms mentioned above work correctly for it. The main idea of this mapping can be easily demonstrated for the simplest particular case, when all integer poles are of first order. In this case the bottom of $f(z)$ is the sequence (c_n) such that c_n is equal to zero if n is an ordinary point of $f(z)$, and c_n is equal to the residue of $f(z)$ at n otherwise. It can be shown that the bottom of a meromorphic solution is a subanalytic solution. (By a similar method it also can be proved that an arbitrary equation of the form (2) has an entire analytic solution.)

The invocation of formal power series enables us to prove some facts related to the case when an initial difference equation $L(y) = 0$ has coefficients in $K[z]$, where K is an arbitrary

trary field of zero characteristic. For example, let a summing operator $R \in K(z)[E]$ exist for L , then the equation $L(y) = 0$ has a sequential solution (c_n) , $c_n \in K$, $n \in \mathbb{Z}$, such that a correct application of R to (c_n) is possible.

If $d = 1$ then a sequential solution of (2) is a *hypergeometric* sequence. We also consider multidimensional hypergeometric sequences. An s -dimensional H -system is a system of equations for a single unknown function which has the form

$$\begin{aligned} & f_i(z_1, z_2, \dots, z_s) y(z_1, z_2, \dots, z_i + 1, \dots, z_s) \\ &= g_i(z_1, z_2, \dots, z_s) y(z_1, z_2, \dots, z_i, \dots, z_s), \end{aligned} \quad (3)$$

where f_i, g_i are relatively prime non-zero polynomials over $\mathbb{C}$ for $i = 1, 2, \dots, s$. (The prefix “ H ” refers to Jakob Horn and to the adjective “hypergeometric” as well.) An s -dimensional *hypergeometric* sequence is a sequential solution of some s -dimensional H -system (such sequence has to be defined for all $(n_1, n_2, \dots, n_s) \in \mathbb{Z}^s$).

Rational functions $F_1, F_2, \dots, F_s \in \mathbb{C}(z_1, z_2, \dots, z_s)$ are *compatible*, if

$$\begin{aligned} & F_i(z_1, z_2, \dots, z_j + 1, \dots, z_s) F_j(z_1, z_2, \dots, z_j, \dots, z_s) = \\ &= F_j(z_1, z_2, \dots, z_i + 1, \dots, z_s) F_i(z_1, z_2, \dots, z_i, \dots, z_s) \end{aligned}$$

for all $1 \leq i < j \leq s$. The H -system (3) is *consistent*, if the rational functions

$$F_i = \frac{g_i(z_1, z_2, \dots, z_s)}{f_i(z_1, z_2, \dots, z_s)}, \quad i = 1, 2, \dots, s,$$

are compatible. This consistency condition of (3) is similar to the condition of the commutation of differentiations by independent variables in the differential case.

The $\mathbb{C}$ -linear space of hypergeometric sequences that satisfy a given H -system $\mathcal{H}$ we will denote as $V(\mathcal{H})$. We prove that

- 1) $\dim V(\mathcal{H}) > 0$ for any consistent H -system $\mathcal{H}$;
- 2) if m, s are arbitrary natural numbers, then there exists an s -dimensional H -system $\mathcal{H}$ such that $\dim V(\mathcal{H}) = m$;
- 3) if $s = 1$ for an H -system $\mathcal{H}$, then $\dim V(\mathcal{H}) < \infty$, but for an arbitrary integer $s > 1$ there exists an s -dimensional H -system $\mathcal{H}$ such that $\dim V(\mathcal{H}) = \infty$.

We also give an appropriate corollary of the well-known Ore-Sato theorem on possible forms of s -dimensional hypergeometric sequences. Notice that, contrary to some interpretations found in the literature, the Ore-Sato theorem does *not* imply that every sequential solution of an H -system is of the form

$$R(n_1, \dots, n_s) \frac{\prod_{i=1}^p \Gamma(a_{i,1}n_1 + \dots + a_{i,s}n_s + \alpha_i)}{\prod_{j=1}^q \Gamma(b_{j,1}n_1 + \dots + b_{j,s}n_s + \beta_j)} u_1^{n_1} \dots u_s^{n_s},$$

where $R \in \mathbb{C}(n_1, n_2, \dots, n_s)$, $a_{ik}, b_{jk} \in \mathbb{Z}$, and $\alpha_i, \beta_j, u_k \in \mathbb{C}$. For example, $c_{n_1, n_2} = |n_1 - n_2|$, $(n_1, n_2) \in \mathbb{Z}^2$, is a 2-dimensional hypergeometric sequence which is not of this form.

Summation problems for s -dimensional hypergeometric sequences also are considered by computer algebra. For example, in the case $s = 2$ one of the integer variables n_1, n_2 can be the summation variable, while another one can be a parameter, which appears in the summation bounds. Zeilberger’s algorithm (“creative telescoping”) helps quite often to find a closed form of such sums. But if arbitrary 2-dimensional hypergeometric sequences are considered, then as in the one-dimensional case, the combination of creative telescoping with the discrete Newton-Leibniz formula can produce an incorrect result (examples can be given). However the notion of the subanalytic solution can be generalized for the multidimensional case. We show that the creative telescoping with the discrete Newton-Leibniz formula will give the correct result for any subanalytic solution of any such H -system, on which the creative telescoping succeeds.

Concerning the $\mathbb{C}$ -linear space of all subanalytic solutions of a consistent H -system we show that the dimension of this space is always ≤ 1 , and if $s > 1$, then the dimension of this space is 0 for some H -systems.

In addition we discuss some problems connected with searching for power series solutions of ordinary linear differential equations with polynomial coefficients, having in mind the power series whose coefficients can be expressed in closed form as functions of index. A key role in the process of constructing such power series solutions is played by an investigation of the family of difference equations, which are satisfied by coefficient sequences of power series solutions and which depend significantly on the expansion point.

The presented results were obtained by the author jointly with M. Barkatou, M. van Hoeij, and M. Petkovšek.

Digital Atlases and Difference Forms

Elizabeth L. Mansfield
Institute of Mathematics, Statistics and Actuarial Science
University of Kent
Canterbury CT2 7NF, United Kingdom
e.l.mansfield@kent.ac.uk

ABSTRACT

When integrating a differential equation numerically, it can be important for the solution method to reflect the geometric properties of the original model. These include conservation laws and first integrals, symmetries, and symplectic or variational structures. Thus there is an increasingly sophisticated subject of “geometric integration” concentrating mostly on local properties of the equation.

This talk is concerned with ways of ensuring that finite difference schemes accurately mirror *global* properties. To this end, lattice varieties are introduced on which finite difference schemes amongst others may be defined. There is no assumption of continuity, or that either the lattice variety or the difference systems have a continuum limit; our theory is more general than that of cubical complexes, and the proofs require a different foundation.

We show that the global structure of a lattice variety can be determined from its digital atlas. This is important for two reasons. First, if the digital atlas has the same “system of intersections” as that of the smooth model it approximates, you are guaranteed the same global information. Secondly, since our proofs are independent of any continuum limit, global information for inherently discrete models may be obtained. The techniques used are algebraic, specifically homological algebra, which amounts to linear algebra.

This talk has two meta-messages:

- Continuity is an illusion.
- If you want to capture analytic structures in discrete models successfully, *cherchez l’algèbre*.

No particular expertise is assumed for this talk, which is based on the paper, *Difference Forms* by Elizabeth L. Mansfield and Peter E. Hydon, to appear in Foundations of Computational Mathematics.

Categories and Subject Descriptors

G.2.0 [Mathematics of Computing]: DISCRETE MATHEMATICS—*General*

General Terms

Algorithms, Theory

Keywords

cohomology, difference chains, difference forms, lattice variety, local difference potentials, local exactness, symbolic numeric methods

1. EQUATIONS ON LATTICE VARIETIES

Global information about a space is important to numerical approximation because solutions to differential equations can and do depend on whether, say, the space is a torus (implicit if periodic boundary conditions are imposed) or a sphere (for example weather systems on the earth), whether there are missing points giving rise to monodromy, and so on. The global structure of a space is often encoded in its *cohomology* and software for calculating this for a variety of cohomology theories is now available [4–6].

Together with a representation of a global discrete space one needs to consider the kind of equations that may be defined on it. Considerable effort has been focused on analogues of differential forms, based on Whitney forms, for computational electromagnetism; see [1] for a comprehensive discussion of recent results. A related approach to the general finite element method uses *discrete differential forms* which are actually classical simplicial cochains, that is, linear maps from the vertices, edges, faces, and so on, to the coefficient field.

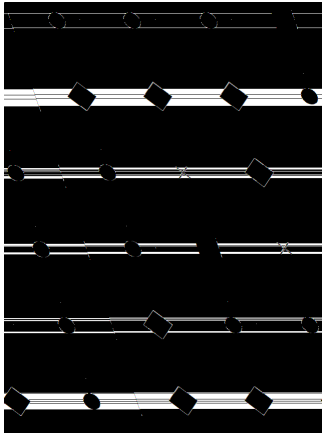
It is essential to realize that the term ‘*discrete differential form*’ does not imply that the underlying space is discrete. Indeed, the finite element method uses a simplicial approximation to a smooth manifold; the space remains continuous even though differentiability is lost on the edges of simplices.

By contrast, finite difference methods are defined in terms of mesh points, without there being a need for an underlying continuous space. The same is true for difference equations in general; where such equations are used to model an inherently discrete process, the imposition of a continuous structure can produce artifacts. A related philosophical problem was raised by Chard and Shapiro [3], who write of the loss of physical content by the standard process of taking a continuum limit of the discrete process and then re-discretizing for the purpose of computation. Further, there is interest in studying physical systems on discrete space-times, for example in quantum gravity.

To deal with difference equations in a theory that uses only concepts germane to difference equations themselves,

it is necessary to discard the continuous base space. At first sight this might seem to be disastrous, for most of the familiar and useful constructions are lost. These include continuity, the tangent bundle and the exterior derivative; indeed, difference operators are not derivations, that is, they have no Leibniz product rule. Moreover, loss of continuity means that one must abandon results for constructions that are based on an underlying continuum, such as cohomology theories based on cubical simplices. In fact, classical cubical chains can be mapped injectively but not bijectively to their difference analogues. Consequently, although some theorems from cubical simplex theories can be adapted to the context of difference equations, they must be reproved without reference to a continuous space.

Figure 1: Lattice with two points removed, showing which forward difference forms may be defined at which points.



Key: ● one form defined in direction indicated
 ■ zero form ◆ two form defined
 × point removed

There exist applications which are inherently discrete and systems for which there are either multiple or no continuum limits. Thus we prove results using only those tools and constructions which pertain to difference equations themselves, such as the shift operator. In finite difference methods, mesh points do not need to be evenly spaced but they are ordered in each direction by an integer label, at least locally. Instead of dealing with mesh points directly, it is helpful to regard the independent variables (coordinates) as being p -tuples of integers, where p is the dimension of the discretized problem. The usual ordering on $\mathbb{Z}$ provides sufficient structure, namely adjacency and orientation, to enable us to derive difference analogues of chains, exterior algebra, a coboundary operator, the de Rham complex and Stokes' Theorem; each analogue is adapted to a particular finite difference method. Further, we can provide the difference analogues of local potentials for curl-free and divergence-free fields in the smooth theory. In Figure 1, a simple example of a non-trivial lattice with forward difference forms is shown. While the theory is developed for forward difference equations, other schemes can be studied in the way we outline.

In the continuous case, one can regard a manifold as the result of gluing together coordinate patches. Similarly, it is

possible to glue together pieces of mesh, each of which has its own coordinate system, to form what we call a *lattice variety*. By combining the difference complex with a Čech complex, the difference cohomology may be calculated, following Weil's arguments [2] connecting de Rham and Čech cohomologies.

Finally, we give evidence for a conjecture that the dimension of the solution space of a globally defined linear recurrence relation equals the analogue of the Euler characteristic for the lattice variety. As the same quantity appears in the Morse Index Theorem, our conjecture might be thought to be (in some distant sense) an analogue of the Morse Index theorem for difference systems on lattice varieties. However, our difference index is not a result of a discrete Morse theory, such as the one constructed by Forman.

2. REFERENCES

- [1] D. Arnold, R. S. Falk and R. Winthur, Finite element exterior calculus, homological techniques and applications, *Acta Numerica* **15** (2006), 1–155.
- [2] R. Bott and L. W. Tu, *Differential Forms in Algebraic Topology*, Graduate Texts in Mathematics **82**, Springer Verlag, New York, 1982.
- [3] J. A. Chard and V. Shapiro, A multivector data structure for differential forms and equations, *Math. Comp. Sim.* **54** (2000), 33–64.
- [4] T. Kaczynski, K. Mischaikow and M. Mrozek, Computing Homology, *Homology, Homotopy and Applications* **5** (2003), 233–256.
- [5] M. Mrozek, P. Pilarczyk and N. Zelazna, Homology algorithm based on acyclic subspace, *Computers and Mathematics with Applications*, **55** (2008) 2395–2412.
- [6] S. Suuriniemi and L. Kettunen, Trade-off between information and complexity: a technique for automated topological computations, *COMPEL* **22** (2003), 481–494.

Can We Create a Viable Free Open Source Alternative to Magma, Maple, Mathematica and Matlab?

William Stein
Department of Mathematics
University of Washington
Seattle, Washington
USA
<http://wstein.org>
wstein@gmail.com

ABSTRACT

The goal of the Sage project (<http://sagemath.org>) is to create a truly viable free open source alternative to Magma, Maple, Mathematica and Matlab. Is this possible?

Categories and Subject Descriptors

D.2.8 [Software Engineering]: Miscellaneous

General Terms

Design

1. INTRODUCTION

For over a decade I have primarily done research in number theory that often involves computation, mainly using Magma. In 2004 I realized that it was stupid for me to continue building all of my work on Magma because Magma is proprietary, the development model is closed, Magma is expensive which is bad for students, and the language itself lacked many features (e.g., user defined classes) that I had requested repeatedly for over 5 years. Thus to continue to use *only* Magma unacceptably limited my potential in both research and education.

Having used Magma for many years, I simply could not switch to an existing open source system. The only serious free open source software for number theory is PARI, whose capabilities are far behind that of Magma in numerous critical areas of interest to me, including exact linear algebra, commutative algebra, and algebraic curves. And no other free system—GAP, Singular, Axiom, Maxima, Macaulay 2, etc.—even comes close in *all* these areas. In fact, after a decade of patiently waiting, I doubt they ever will.

Magma is the result of decades of hard work by extremely talented mathematicians and programmers such as John Cannon, Allan Steel, Claus Fieker, David Kohel, and many oth-

ers. I've worked with them and they are *simply amazing*, as is their software. The situation for me to find something similar but open source seemed hopeless. And all attempts to convince the Magma group to open source Magma failed.

In 2004, frustrated that there was no way to solve my problem, and driven by nothing but a naive compulsion, I started the Sage project as a free open source alternative to Magma, and spent a large amount of time working on it even though I was convinced that there was no hope of Sage ever succeeding. The first version of Sage consisted of the Python interpreter and a few scripts for doing number theory, with a design modeled on Magma.

My first real feedback from the computer algebra community came from Richard Fateman in December 2005 when he posted his opinion of the Sage project to sci.math.symbolic:

“By avoiding applications (say, to engineering design, finance, education, scientific visualization, etc etc) the activity [Sage] is essentially doomed. Why? Government funding for people or projects will be a small percentage of the funding for pure mathematics. That's not much. And the future is pretty grim.”

It is now nearly three years later and the Sage project currently has over 100 contributors and around 10,000 users. In November 2007, Sage won first place in the scientific category of the Trophées du Libre

<http://www.tropheesdulibre.org/>

a major international free software competition. Sage is funded by the US National Science Foundation, the US Department of Defense, the University of Washington, Microsoft Research, Google and private donations. Sage has new releases every two weeks, and typically 30–40 people contribute to each release. All new code contributions to Sage are peer reviewed, and every new function must be documented with tests that illustrate its usage. The documentation has over 50,000 lines of input examples.

2. WHAT IS SAGE?

1. a *huge distribution* of free open source mathematical software that is surprisingly easy to build from source,
2. a *set of interfaces* to most other mathematical software systems, and
3. a *new Python library* that fills in the numerous gaps in other open source math software included in Sage, unifies everything offering a smooth user experience, and provides a modern web-based graphical notebook interface with math typesetting and integrated 2D and 3D graphics.

3. WHAT MAKES SAGE UNIQUE?

3.1 Python and Cython

Sage is the first large general purpose mathematics software system that uses a mainstream programming language (Python) as the end user language. Python—easily one of the world’s top 10 programming languages—is a powerful and beautiful modern interpreted programming language with an organized and professional developer base and millions of users. Sage also makes extensive use of a Python-to-C compiler called Cython (see <http://www.cython.org>).

“It is our belief that this [Cython] is something missing in the world, and not for any good technical reasons. That it is eminently possible to have a language that gets down to the iron, runs at C speeds, and has no surprises in generated assembly, but at the same time guides you along to a clear, succinct and correct expression of complicated systems and algorithms.” – Dan Gindikin

By building on Python and Cython we see that Sage has a tremendous longterm advantage over every other general purpose computer algebra system. Sage gets excellent support for compiled code, thousands of third party Python libraries, object serialization, superb database support, and excellent mature numerical libraries (numpy and scipy).

3.2 Building the Car Instead of Reinventing the Wheel

Instead of reinventing the wheel, Sage combines many of the best existing open source systems that have been developed over the last 40 years (about *5 million lines of code*) with about *250,000 lines of new code*. Every single copy of Sage includes all of the following software (and much much more):

- Algebra and calculus: Maxima, SymPy
- High precision arithmetic: GMP, MPFR, MPFI, quad-double, Givaro
- Commutative algebra: Singular
- Number theory: PARI, NTL, mwrank, ECM, FLINTQS, GMP-ECM
- Exact linear algebra: LinBox, IML
- Group theory: GAP
- Scientific computation: GSL, SciPy, NumPy, cvxopt
- Statistical computation: R
- Graphics (2d and 3d): Matplotlib, Tachyon3d, Jmol

Sage is thus the first system to combine together such a wide range of libraries and programs in a meaningful way. This huge range of programs is tied together using Python’s excellent extensibility via C libraries and also in some cases using pseudo-tty’s. Sage has a highly developed unified collection of pseudo-tty based interfaces that make it is possible to make extensive use of Maple, Mathematica, Magma, Matlab, GAP, Maxima, Singular, PARI, and many other systems from anywhere within a single Sage program.

Curious? If you also *want* a viable open source alternative to Magma, Maple, Mathematica or Matlab, drop everything, try out Sage now and become a Sage developer.

<http://www.sagemath.org>

Towards a Computer Algebraic Algorithm for Flat Output Determination

Felix Antritter

Automatisierungs- und Regelungstechnik,
Universität der Bundeswehr München
Werner-Heisenberg-Weg 37
DE-85579 Neubiberg, Germany
felix.antritter@unibw.de

Jean Lévine

Centre Automatique et Systèmes
École des Mines de Paris
35, rue Saint-Honoré
77300 Fontainebleau, France
jean.levine@ensmp.fr

ABSTRACT

This contribution deals with nonlinear control systems. More precisely, we are interested in the formal computation of a so-called *flat output*, a particular generalized output whose property is, roughly speaking, that all the integral curves of the system may be expressed as smooth functions of the components of this flat output and their successive time derivatives up to a finite order (to be determined). Recently, a characterization of such flat output has been obtained in [14, 15], in the framework of manifolds of jets of infinite order (see e.g. [18, 9]), that yields an abstract algorithm for its computation. In this paper it is discussed how these conditions can be checked using computer algebra. All steps of the algorithm are discussed for the simple (but rich enough) example of a non holonomic car.

Categories and Subject Descriptors

I.1.4 [Computing Methodologies]: Symbolic and Algebraic Manipulation—*Applications*; G.4 [Mathematics of Computing]: Mathematical Software

General Terms

Theory

1. INTRODUCTION

We consider a nonlinear control system

$$\dot{x} = f(x, u) \quad (1)$$

where $x = (x_1, \dots, x_n)$ is the state vector, $u = (u_1, \dots, u_m)$ the control vector, $m \leq n$, and f is a meromorphic function of its arguments.

We say that this system is *differentially flat*, or shortly *flat* ([17, 7]), if and only if there exists a vector $y = (y_1, \dots, y_m)$ such that:

(i) y and its successive time derivatives $\dot{y}, \ddot{y}, \dots$ are functionally independent,

(ii) y is a function of x , u and a finite number of time derivatives of the components of u ,

(iii) x and u can be expressed as functions of the components of y and a finite number of their successive time derivatives:

$$x = \varphi(y, \dot{y}, \dots, y^{(\alpha)}), \quad u = \psi(y, \dot{y}, \dots, y^{(\alpha+1)})$$

for some multi-integer $\alpha = (\alpha_1, \dots, \alpha_m)$, and with the notation $y^{(\alpha)} = (\frac{d^{\alpha_1} y_1}{dt^{\alpha_1}}, \dots, \frac{d^{\alpha_m} y_m}{dt^{\alpha_m}})$.

A vector y having these properties is called a *flat output*.

This concept has inspired an important literature and a large number of practical and industrial applications (see e.g. [18] for a survey). Its main advantages rely on the simplicity to solve the *motion planning* and *stable tracking* problems.

Various formalisms have been introduced to study this remarkable class of systems: finite dimensional differential geometric approaches ([2, 10, 28, 29]), differential algebra and related approaches ([8, 1, 12]), infinite dimensional differential geometry of jets and prolongations ([9, 30, 22, 20, 24]). Among these contributions, the characterization of differential flatness takes a large part ([1, 2, 4, 10, 12, 19, 21, 23, 24, 25, 28, 29, 14, 15]).

We follow here the results of [14, 15] in the formalism of manifolds of jets of infinite order ([9, 13, 22, 31]). For the stated flatness conditions implicit systems are considered which are obtained from (1) by eliminating the input vector u . We recall the notions of Lie-Bäcklund equivalence and Lie-Bäcklund isomorphism in this context and the flatness necessary and sufficient conditions in terms of polynomial matrices and differential forms. Note that this approach may be seen as an extension to nonlinear systems of [16] and provide flatness conditions that are invariant by endogeneous dynamic feedback extension.

The derived conditions use differential operators which combine differential geometric concepts like exterior derivative and wedge product as well as algebraic concepts as operations on skew polynomials with coefficients that are meromorphic functions of the coordinates. Existing computer algebra systems offer lots of functionalities for each of the mentioned fields but their combination is not considered. In this paper we show how to implement such operators in Maple 11, and include them in an algorithm to check the flatness necessary and sufficient conditions. Note that this algorithm doesn't necessarily finish in a finite number of steps.

The paper is organized as follows: Section 2 is devoted to the basic description of implicit control systems on manifolds of jets of infinite order. The notions of Lie-Bäcklund

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

equivalence and Lie-Bäcklund isomorphism in this framework are recalled and the flatness property is defined precisely. In Section 3 the necessary and sufficient conditions for differential flatness derived in [14, 15] are stated and Section 4 discusses in more detail the introduced operators and conditions. General formulae for the operators are derived from its defining relations and it is explained how the operators and the algorithm developed in Section 3 can be implemented using a computer algebra system like *Maple 11*. Finally, in Section 5 the application of the implemented method is illustrated for the well-known nonholonomic car example. This example, though simple, may however call for all the resources of the algorithm. A first draft of this algorithm can be obtained by sending an e-mail to the first author.

2. IMPLICIT CONTROL SYSTEMS ON MANIFOLDS OF JETS OF INFINITE ORDER

Given an infinitely differentiable manifold X of dimension n , we denote its tangent space at $x \in X$ by $T_x X$, and its tangent bundle by TX . Let F be a meromorphic function from TX to $\mathbb{R}^{n-m}$. We consider an underdetermined implicit system of the form

$$F(x, \dot{x}) = 0 \quad (2)$$

regular in the sense that $\text{rk } \frac{\partial F}{\partial \dot{x}} = n - m$ in a suitable dense open subset of TX .

According to the implicit function theorem, any explicit system (1) with $x \in X$, $(x, f(x, u)) \in T_x X$ for every u in an open subset U of $\mathbb{R}^m$, and $\text{rk } \frac{\partial f}{\partial u} = m$ in a suitable open subset of $X \times U$, can be locally transformed into (2), and conversely.

A vector field f that depends, for every $x \in X$, on m independent variables $u \in \mathbb{R}^m$ in a meromorphic way with $\text{rk } \frac{\partial f}{\partial u} = m$ in a suitable open subset of $X \times \mathbb{R}^m$, satisfying $F(x, f(x, u)) = 0$ for every $u \in U$, is called *compatible* with (2).

Note that this elimination step, though easy for some classes of systems, e.g., affine with respect to u , may be non trivial in general. Remark also that the implicit representation (2), as opposed to (1), is invariant by endogeneous dynamic extension (see [9] for a precise definition).

In [9] (see also [22] where a similar approach has been developed independently), infinite systems of coordinates $(x, \bar{u}) = (x, u, \dot{u}, \dots)$ have been introduced to deal with prolonged vector fields

$$\bar{f}(x, \bar{u}) = \sum_{i=1}^n f_i(x, u) \frac{\partial}{\partial x_i} + \sum_{j=1}^m \sum_{k \geq 0} u_j^{(k+1)} \frac{\partial}{\partial u_j^{(k)}}$$

the original system being in explicit form (1).

Following [14, 15], we adopt an external description of the prolonged manifold containing the solutions of (2): we consider the infinite dimensional manifold $\mathfrak{X}$ defined by $\mathfrak{X} \stackrel{\text{def}}{=} X \times \mathbb{R}_\infty^n \stackrel{\text{def}}{=} X \times \mathbb{R}^n \times \mathbb{R}^n \times \dots$, made of an infinite (but countable) number of copies of $\mathbb{R}^n$, endowed with the product topology, and we assume that we are given the global infinite set of coordinates of $\mathfrak{X}$:

$$\bar{x} = (x, \dot{x}, \dots, x^{(k)}, \dots). \quad (3)$$

Recall that, in this topology, a function φ from $\mathfrak{X}$ to $\mathbb{R}$ is *continuous* (resp. *differentiable*) if φ depends only on a finite (but otherwise arbitrary) number of variables and is

continuous (resp. differentiable) with respect to these variables. C^∞ or analytic or meromorphic functions from $\mathfrak{X}$ to $\mathbb{R}$ are then defined as in the usual finite dimensional case since they only depend on a finite number of variables. We endow $\mathfrak{X}$ with the so-called trivial Cartan field ([13, 31])

$$\tau_{\mathfrak{X}} = \sum_{i=1}^n \sum_{j \geq 0} x_i^{(j+1)} \frac{\partial}{\partial x_i^{(j)}}. \quad (4)$$

We also denote by $L_{\tau_{\mathfrak{X}}} \varphi = \sum_{i=1}^n \sum_{j \geq 0} x_i^{(j+1)} \frac{\partial \varphi}{\partial x_i^{(j)}} = \frac{d\varphi}{dt}$ the Lie derivative of a differentiable function φ along $\tau_{\mathfrak{X}}$ and $L_{\tau_{\mathfrak{X}}}^k \varphi$ its k th iterate. Thus $x_i^{(k)} = \frac{d^k x_i}{dt^k} = L_{\tau_{\mathfrak{X}}}^k x_i$ for every $i = 1, \dots, n$ and $k \geq 1$, with the convention $x_i^{(0)} = x_i$.

Since $\frac{d}{dt} x_i^{(j)} \stackrel{\text{def}}{=} \dot{x}_i^{(j)} = x_i^{(j+1)}$, the Cartan field acts on coordinates as a shift to the right. $\mathfrak{X}$ is thus called *manifold of jets of infinite order*. From now on, $\bar{x}, \bar{y}, \dots$ stand for the sequences of jets of infinite order of $x, y, \dots$

A *regular implicit control system* is defined as a triple $(\mathfrak{X}, \tau_{\mathfrak{X}}, F)$ with $\mathfrak{X} = X \times \mathbb{R}_\infty^n$, $\tau_{\mathfrak{X}}$ its associated trivial Cartan field, and F meromorphic from TX to $\mathbb{R}^{n-m}$ satisfying $\text{rk } \frac{\partial F}{\partial \dot{x}} = n - m$ in a suitable open subset of TX .

2.1 Lie-Bäcklund equivalence for implicit systems

We recall from [14, 15] the following definition:

Let us consider two regular implicit control systems $(\mathfrak{X}, \tau_{\mathfrak{X}}, F)$, with $\mathfrak{X} = X \times \mathbb{R}_\infty^n$, $\dim X = n$ and $\text{rk } \frac{\partial F}{\partial \dot{x}} = n - m$, and $(\mathfrak{Y}, \tau_{\mathfrak{Y}}, G)$, with $\mathfrak{Y} = Y \times \mathbb{R}_\infty^p$, $\dim Y = p$, $\tau_{\mathfrak{Y}}$ its trivial Cartan field, and $\text{rk } \frac{\partial G}{\partial \dot{y}} = p - q$.

Set $\mathfrak{X}_0 = \{\bar{x} \in \mathfrak{X} | L_{\tau_{\mathfrak{X}}}^k F(\bar{x}) = 0, \forall k \geq 0\}$ and $\mathfrak{Y}_0 = \{\bar{y} \in \mathfrak{Y} | L_{\tau_{\mathfrak{Y}}}^k G(\bar{y}) = 0, \forall k \geq 0\}$. They are endowed with the topologies and differentiable structures induced by $\mathfrak{X}$ and $\mathfrak{Y}$ respectively.

DEFINITION 1. We say that the regular implicit control systems $(\mathfrak{X}, \tau_{\mathfrak{X}}, F)$ and $(\mathfrak{Y}, \tau_{\mathfrak{Y}}, G)$ are Lie-Bäcklund equivalent (or shortly L-B equivalent) at the pair of points $(\bar{x}_0, \bar{y}_0) \in \mathfrak{X}_0 \times \mathfrak{Y}_0$ if and only if

(i) there exist neighborhoods $\mathfrak{X}_0$ and $\mathfrak{Y}_0$ of $\bar{x}_0$ in $\mathfrak{X}_0$ and $\bar{y}_0$ in $\mathfrak{Y}_0$ respectively and a one-to-one meromorphic mapping $\Phi = (\varphi_0, \varphi_1, \dots)$ from $\mathfrak{Y}_0$ to $\mathfrak{X}_0$ satisfying $\Phi(\bar{y}_0) = \bar{x}_0$ and such that the trivial Cartan fields are Φ -related, namely $\Phi_* \tau_{\mathfrak{Y}} = \tau_{\mathfrak{X}}$;

(ii) there exists Ψ one-to-one and meromorphic from $\mathfrak{X}_0$ to $\mathfrak{Y}_0$, with $\Psi = (\psi_0, \psi_1, \dots)$, such that $\Psi(\bar{x}_0) = \bar{y}_0$ and $\Psi_* \tau_{\mathfrak{X}} = \tau_{\mathfrak{Y}}$.

The mappings Φ and Ψ are called mutually inverse Lie-Bäcklund isomorphisms at $(\bar{x}_0, \bar{y}_0)$.

The two systems $(\mathfrak{X}, \tau_{\mathfrak{X}}, F)$ and $(\mathfrak{Y}, \tau_{\mathfrak{Y}}, G)$ are called locally L-B equivalent if they are L-B equivalent at every pair $(\bar{x}, \Psi(\bar{x})) = (\Phi(\bar{y}), \bar{y})$ of an open dense subset $\mathcal{Z}$ of $\mathfrak{X}_0 \times \mathfrak{Y}_0$, with Φ and Ψ mutually inverse Lie-Bäcklund isomorphisms on $\mathcal{Z}$.

As a result, local L-B equivalence preserves equilibrium points, namely points $\bar{y}$ (resp. $\bar{x}$) such that $G(\bar{y}, 0) = 0$ (resp. $F(\bar{x}, 0) = 0$), and coranks ($m = q$).

2.2 Differential Forms

Let us introduce a basis of the tangent space $T_{\bar{x}} \mathfrak{X}$ of $\mathfrak{X}$ at a point $\bar{x} \in \mathfrak{X}$ consisting of the set of vectors

$$\left\{ \frac{\partial}{\partial x_i^{(j)}} | i = 1, \dots, n, j \geq 0 \right\}.$$

A basis of the cotangent space $T_{\bar{x}}^*\mathfrak{X}$ at $\bar{x}$ is given by $\{dx_i^{(j)} | i = 1, \dots, n, j \geq 0\}$ with $\langle dx_i^{(j)}, \frac{\partial}{\partial x_k^{(l)}} \rangle = \delta_{i,k} \delta_{j,l}$, $\delta_{i,k}$ being the Kronecker symbol.

The differential of F is thus given, in matrix notations, by

$$dF = \frac{\partial F}{\partial x} dx + \frac{\partial F}{\partial \dot{x}} d\dot{x}. \quad (5)$$

Note that the shift property of $\frac{d}{dt}$ on coordinates extends to differentials: $\frac{d}{dt} dx = d\dot{x} = d\frac{d}{dt}x$, i.e. $\frac{d}{dt}$ commutes with d .

Since a smooth function depends on a finite number of variables, its differential contains only a finite number of non zero terms. Accordingly, we define a 1-form on $\mathfrak{X}$ as a *finite* linear combination of the $dx_i^{(j)}$'s, with coefficients meromorphic from $\mathfrak{X}$ to $\mathbb{R}$ or, equivalently as a local meromorphic section of $T^*\mathfrak{X}$. The set of 1-forms is noted $\Lambda^1(\mathfrak{X})$. We also denote by $\Lambda^p(\mathfrak{X})$ the module of all the p -forms on $\mathfrak{X}$, by $(\Lambda^p(\mathfrak{X}))^m$ the space of all the m -dimensional vector p -forms on $\mathfrak{X}$, by $(\Lambda(\mathfrak{X}))^m$ the space of all the m -dimensional vector forms of arbitrary degree on $\mathfrak{X}$, and by $\mathcal{L}_q((\Lambda(\mathfrak{X}))^m) = \mathcal{L}((\Lambda^p(\mathfrak{X}))^m, (\Lambda^{p+q}(\mathfrak{X}))^m, p \geq 1)$, the space of all linear operators from $(\Lambda^p(\mathfrak{X}))^m$ to $(\Lambda^{p+q}(\mathfrak{X}))^m$ for all $p \geq 1$, where $\mathcal{L}(\mathcal{P}, \mathcal{Q})$ denotes the set of linear mappings from a given space $\mathcal{P}$ to a given space $\mathcal{Q}$.

Note that if Φ is a meromorphic mapping from $\mathfrak{Y}$ to $\mathfrak{X}$, the definition of the (backward) image by Φ of a 1-form is the same as in the finite dimensional context.

2.3 Flatness

First recall from [9] that a system in explicit form is flat if and only if it is L-B equivalent to a trivial system. The reader may easily check that this definition is just a concise restatement of the definition given in the Introduction. In our implicit context, it reads:

DEFINITION 2. *The implicit system $(\mathfrak{X}, \tau_{\mathfrak{X}}, F)$ is flat at $(\bar{x}_0, \bar{y}_0) \in \mathfrak{X}_0 \times \mathbb{R}_{\infty}^m$ if and only if it is L-B equivalent at $(\bar{x}_0, \bar{y}_0)$ to the trivial implicit system $(\mathbb{R}_{\infty}^m, \tau_{\mathbb{R}_{\infty}^m}, 0)$. In this case, the mutually inverse L-B isomorphisms Φ and Ψ are called inverse trivializations.*

The next result is proven in [15].

THEOREM 1. *The system $(\mathfrak{X}, \tau_{\mathfrak{X}}, F)$ is flat at $(\bar{x}_0, \bar{y}_0) \in \mathfrak{X}_0 \times \mathbb{R}_{\infty}^m$ if and only if there exists a local meromorphic invertible mapping Φ from $\mathbb{R}_{\infty}^m$ to $\mathfrak{X}_0$, with meromorphic inverse, satisfying $\Phi(\bar{y}_0) = \bar{x}_0$, and such that*

$$\Phi^* dF = 0. \quad (6)$$

3. NECESSARY AND SUFFICIENT CONDITIONS FOR DIFFERENTIAL FLATNESS

We now analyze condition (6) in more details: it characterizes the linear tangent mapping of Φ whose image entirely lies in the kernel of dF . The set of such mappings may be obtained in a systematic way in the framework of polynomial matrices by considering the following matrices polynomial with respect to the differential operator $\frac{d}{dt}$ (we use indifferently $\frac{d}{dt}$ for $L_{\tau_{\mathfrak{X}}}$ or $L_{\tau_{\mathbb{R}_{\infty}^m}}$, the context being unambiguous):

$$P(F) = \frac{\partial F}{\partial x} + \frac{\partial F}{\partial \dot{x}} \frac{d}{dt}, \quad P(\varphi_0) = \sum_{j \geq 0} \frac{\partial \varphi_0}{\partial y^{(j)}} \frac{d^j}{dt^j} \quad (7)$$

with $P(F)$ (resp. $P(\varphi_0)$) of size $(n-m) \times n$ (resp. $n \times m$). Equation (6) reads:

$$\Phi^* dF = P(F)P(\varphi_0)dy = 0. \quad (8)$$

Clearly, the entries of the matrices in (7) are polynomials of the differential operator $\frac{d}{dt}$ with meromorphic coefficients from $\mathfrak{X}$ to $\mathbb{R}$.

We denote by $\mathfrak{K}$ the field of meromorphic functions from $\mathfrak{X}$ to $\mathbb{R}$ and by $\mathfrak{K}[\frac{d}{dt}]$ the principal ideal ring of polynomials of $\frac{d}{dt}$ with coefficients in $\mathfrak{K}$. Note that $\mathfrak{K}[\frac{d}{dt}]$ is non commutative, even if $n = 1$: for every $a \in \mathfrak{K}$, $a \neq 0$, we have $(\frac{d}{dt} \cdot x - x \cdot \frac{d}{dt})(a) = \dot{x}a + x\dot{a} - x\dot{a} = \dot{x}a \neq 0$, or $\frac{d}{dt} \cdot x - x \cdot \frac{d}{dt} = \dot{x}$.

For $r, s \in \mathbb{N}$, let us denote by $\mathcal{M}_{r,s}[\frac{d}{dt}]$ the module of $r \times s$ matrices over $\mathfrak{K}[\frac{d}{dt}]$ (see e.g. [5]). Recall that, for any $r \in \mathbb{N}$, the inverse of a square invertible matrix of $\mathcal{M}_{r,r}[\frac{d}{dt}]$ is not in general in $\mathcal{M}_{r,r}[\frac{d}{dt}]$. Matrices whose inverse belong to $\mathcal{M}_{r,r}[\frac{d}{dt}]$ are called *unimodular matrices*. They form a multiplicative group denoted by $\mathcal{U}_r[\frac{d}{dt}]$. Every matrix in $\mathcal{M}_{r,s}[\frac{d}{dt}]$ admits a *Smith decomposition* (or diagonal reduction). Without loss of generality, we only state its definition for $P(F) \in \mathcal{M}_{n-m,n}[\frac{d}{dt}]$:

$$VP(F)U = (\Delta, 0_{n-m,m}) \quad (9)$$

with $0_{n-m,m}$ the $(n-m) \times m$ matrix whose entries are all zeros, $V \in \mathcal{U}_{n-m}[\frac{d}{dt}]$, $U \in \mathcal{U}_n[\frac{d}{dt}]$ and $\Delta \in \mathcal{M}_{n-m,n-m}[\frac{d}{dt}]$ a diagonal matrix whose entries $d_{i,i}$ divide $d_{j,j}$ for all $0 \leq i \leq j \leq n-m$. Moreover, the degrees of the $d_{i,i}$'s are uniquely defined (see [5]).

DEFINITION 3. *A matrix $M \in \mathcal{M}_{r,s}[\frac{d}{dt}]$ is said hyper-regular if and only if its Smith decomposition leads to either $(I_r, 0_{r,s-r})$ if $r < s$, or to I_r if $r = s$, or to $\begin{pmatrix} I_s \\ 0_{r-s,s} \end{pmatrix}$ if $r > s$.*

Note that a square matrix $M \in \mathcal{M}_{r,r}[\frac{d}{dt}]$ is hyper-regular if and only if it is unimodular.

According to the equivalence between flatness and controllability of the tangent linear system (see [9]) and controllability and freeness of the module associated to the tangent linear system (see [6]), it is proven in [15] that $P(F)$ is hyper-regular around every integral curve of the system $(\mathfrak{X}, \tau_{\mathfrak{X}}, F)$ if and only if its corresponding tangent module is free.

3.1 Algebraic characterization of the differential of a trivialization

From now on, we assume that $P(F)$ is hyper-regular in a neighborhood of $\bar{x}_0$. In other words, there exist V and U such that

$$VP(F)U = (I_{n-m}, 0_{n-m,m}). \quad (10)$$

U and V satisfying (10) are indeed non unique. We say that $U \in \mathcal{R} - \text{Smith}(P(F))$ and $V \in \mathcal{L} - \text{Smith}(P(F))$ if they are such that $VP(F)U = (I_m, 0)$.

Accordingly, if $M \in \mathcal{M}_{n,m}[\frac{d}{dt}]$ is hyper-regular with $m \leq n$, we say that $V \in \mathcal{L} - \text{Smith}(M)$ and $W \in \mathcal{R} - \text{Smith}(M)$ if $V \in \mathcal{U}_n[\frac{d}{dt}]$ and $W \in \mathcal{U}_m[\frac{d}{dt}]$ satisfy $VMW = \begin{pmatrix} I_m \\ 0 \end{pmatrix}$.

In place of (8), we first solve the matrix equation:

$$P(F)\Theta = 0 \quad (11)$$

where the entries of $\Theta \in \mathcal{M}_{n,m}[\frac{d}{dt}]$ are not supposed to be gradients of some function φ_0 .

LEMMA 1. *The set of hyper-regular matrices $\Theta \in \mathcal{M}_{n,m}[\frac{d}{dt}]$ satisfying (11) is nonempty and given by*

$$\Theta = U \begin{pmatrix} 0_{n-m,m} \\ I_m \end{pmatrix} W \quad (12)$$

with $U \in \mathbf{R} - \text{Smith}(P(F))$ and $W \in \mathcal{U}_m[\frac{d}{dt}]$ arbitrary.

LEMMA 2. For every $Q \in \mathbf{L} - \text{Smith}(\hat{U})$, with $\hat{U}$ given by

$$\hat{U} = U \begin{pmatrix} 0_{n-m,m} \\ I_m \end{pmatrix} \quad (13)$$

there exists $Z \in \mathcal{U}_m[\frac{d}{dt}]$ such that

$$Q\Theta = \begin{pmatrix} I_m \\ 0_{n-m,m} \end{pmatrix} Z. \quad (14)$$

Moreover, for every $Q \in \mathbf{L} - \text{Smith}(\hat{U})$, the submatrix $\hat{Q} = (0_{n-m,m}, I_{n-m})Q$ is equivalent to $P(F)$ ($\exists L \in \mathcal{U}_{n-m}[\frac{d}{dt}]$ such that $P(F) = L\hat{Q}$).

3.2 Integrability

Let us denote by $Q_{i,j} = \sum_{k \geq 0} Q_{i,j}^k \frac{d^k}{dt^k}$ the (i,j) -th polynomial entry of $Q \in \mathbf{L} - \text{Smith}(\hat{U})$ obtained from Lemma 2. We also denote by ω the m -dimensional vector 1-form defined by

$$\begin{aligned} \omega(\bar{x}) &= \begin{pmatrix} \omega_1(\bar{x}) \\ \vdots \\ \omega_m(\bar{x}) \end{pmatrix} = (I_m, 0_{m,n-m}) Q(\bar{x}) dx|_{x_0} \\ &= \begin{pmatrix} \sum_{j=1}^n \sum_{k \geq 0} Q_{1,j}^k(\bar{x}) dx_j^{(k)}|_{x_0} \\ \vdots \\ \sum_{j=1}^n \sum_{k \geq 0} Q_{m,j}^k(\bar{x}) dx_j^{(k)}|_{x_0} \end{pmatrix} \end{aligned} \quad (15)$$

the restriction to $\mathfrak{X}_0$ meaning that $\bar{x} \in \mathfrak{X}_0$ satisfies $L_{\tau_x}^k F = 0$ for all k and that the $dx_j^{(k)}$ are such that $dL_{\tau_x}^k F = 0$ in $\mathfrak{X}_0$ for all k . Since $\hat{Q}$ is hyper-regular, the forms $\omega_1, \dots, \omega_m$ are independent by construction.

Let us also recall that, if $\tau_1, \dots, \tau_m$ are given independent 1-forms in $\Lambda^1(\mathfrak{X}_0)$, the $\mathfrak{R}[\frac{d}{dt}]$ -ideal $\mathfrak{T}$ generated by $\tau_1, \dots, \tau_m$ is the set of all combinations with coefficients in $\mathfrak{R}[\frac{d}{dt}]$ of forms $\eta \wedge \tau_i$ with η an arbitrary form on $\mathfrak{X}_0$ of arbitrary degree and $i = 1, \dots, m$.

DEFINITION 4. We say that the $\mathfrak{R}[\frac{d}{dt}]$ -ideal $\mathfrak{T}$ generated by $\tau_1, \dots, \tau_m$ is strongly closed if and only if there exists a matrix $M \in \mathcal{U}_m[\frac{d}{dt}]$ such that $d(M\tau) = 0$.

This definition is indeed independent of the choice of generators.

THEOREM 2. A necessary and sufficient condition for system (2) to be flat at the point $(\bar{x}_0, \bar{y}_0)$ is that there exist $U \in \mathbf{R} - \text{Smith}(P(F))$ and $Q \in \mathbf{L} - \text{Smith}(\hat{U})$, with $\hat{U}$ given by (13), such that the $\mathfrak{R}[\frac{d}{dt}]$ -ideal Ω generated by the 1-forms $\omega_1, \dots, \omega_m$ defined by (15) is strongly closed in $\mathfrak{X}_0$.

In order to develop the expression $d(M\tau)$ for M polynomial matrix, we define the operator $\mathfrak{d}$ by:

$$\mathfrak{d}(H)\kappa = d(H\kappa) - Hd\kappa \quad (16)$$

for all m -dimensional vector p -form κ in $(\Lambda^p(\mathfrak{X}))^m$ and all $p \geq 1$. Note that (16) uniquely defines $\mathfrak{d}(H)$ as an element of $\mathcal{L}_1((\Lambda(\mathfrak{X}))^m)$.

We can prolong $\mathfrak{d}$ for all $\mu \in \mathcal{L}_q((\Lambda(\mathfrak{X}))^m)$ and for all $\kappa \in (\Lambda^p(\mathfrak{X}))^m$ and all $p \geq 1$ by the formula:

$$\mathfrak{d}(\mu)\kappa = d(\mu\kappa) - (-1)^q \mu d\kappa. \quad (17)$$

THEOREM 3. The $\mathfrak{R}[\frac{d}{dt}]$ -ideal Ω generated by the 1-forms $\omega_1, \dots, \omega_m$ defined by (15) is strongly closed in $\mathfrak{X}_0$ (or, equivalently, the system $(\mathfrak{X}, \tau_x, F)$ is flat) if and only if there exists $\mu \in \mathcal{L}_1((\Lambda(\mathfrak{X}))^m)$, and a matrix $M \in \mathcal{U}_m[\frac{d}{dt}]$ such that

$$d\omega = \mu \omega, \quad \mathfrak{d}(\mu) = \mu^2, \quad \mathfrak{d}(M) = -M\mu. \quad (18)$$

where we have noted $\mu^2 = \mu\mu$.

In addition, if (18) holds true, a flat output y is obtained by integration of $dy = M\omega$.

Note that conditions (18) may be seen as a generalization in the framework of manifolds of jets of infinite order of Cartan's well-known moving frame structure equations (see e.g. [3]).

3.3 A Theoretical Algorithm

From the necessary and sufficient conditions (18), we derive the following abstract algorithm:

1. We first compute a Smith decomposition¹ of $P(F)$ and then $\hat{U}$ as described in Lemmas 1 and 2. If $P(F)$ is not hyperregular, the system is non flat. Otherwise compute the vector 1-form ω defined by (15).
2. We compute the operator μ such that $d\omega = \mu\omega$ by componentwise identification. It is easy to prove that such μ always exists.
3. Among the possible μ 's, only those satisfying $\mathfrak{d}(\mu) = \mu^2$ are kept. If no μ satisfy this relation, the system is non flat.
4. We then compute M such that $\mathfrak{d}(M) = -M\mu$, still by componentwise identification.
5. Finally, only those matrices M which are unimodular are kept. If there are no such M , the system is non flat. In the opposite case, a flat output is obtained by integration of $dy = M\omega$, which is possible since $d(M\omega) = 0$.

4. COMPUTER ALGEBRA IMPLEMENTATION

Before discussing the computer algebra implementation, useful general structures and formulae are derived. Note that computational cost aspects are not considered here. We only aim at showing that Algorithm 3.3 can be implemented using a standard computer algebra system.

4.1 The structure of elements of $\mathcal{M}_{r,s}[\frac{d}{dt}]$

The elements of matrices $A \in \mathcal{M}_{r,s}[\frac{d}{dt}]$ have the structure

$$[A]_{ij} = \sum_{k \geq 0} a_{ijk} \frac{d^k}{dt^k}, \quad i = 1, 2, \dots, r; \quad j = 1, 2, \dots, s \quad (19)$$

where the a_{ijk} 's are smooth functions on $\mathfrak{X}$. Thus, for $\omega \in (\Lambda^p(\mathfrak{X}))^s$, $A\omega \in (\Lambda^p(\mathfrak{X}))^r$ is the vector whose i th component is

$$[A\omega]_i = \sum_{j=1}^s \sum_{k \geq 0} a_{ijk} L_{\tau_x}^k \omega_j, \quad i = 1, 2, \dots, r \quad (20)$$

¹Clearly, in this context, Smith-Decomposition is far from being optimal with respect to computational cost, but is only used to prove the existence of the resulting polynomial matrices. Improvements will be investigated in future work

The multiplication of two matrices $A \in \mathcal{M}_{r_1, r_2}[\frac{d}{dt}]$ and $B \in \mathcal{M}_{r_2, r_3}[\frac{d}{dt}]$ given by $[A]_{i,j} = \sum_{k \geq 0} a_{i,j,k} \frac{d^k}{dt^k}$, and $[B]_{i,j} = \sum_{k \geq 0} b_{i,j,k} \frac{d^k}{dt^k}$, is

$$\begin{aligned} [AB]_{i,j} &= \sum_{l=1}^{r_2} \sum_{k_1, k_2 \geq 0} a_{i,l,k_1} \frac{d^{k_1}}{dt^{k_1}} \left(b_{l,j,k_2} \frac{d^{k_2}}{dt^{k_2}} \right) \\ &= \sum_{l=1}^{r_2} \sum_{k_1, k_2 \geq 0} \sum_{k_3=0}^{k_1} \binom{k_1}{k_3} a_{i,l,k_1} \left(L_{\tau_{\mathcal{X}}}^{k_1-k_3} b_{l,j,k_2} \right) \frac{d^{k_2+k_3}}{dt^{k_2+k_3}} \end{aligned} \quad (21)$$

with $\binom{k_1}{k_3} = \frac{k_1!}{k_3!(k_1-k_3)!}$.

By explicitly indicating the dependence of the coordinates on the independent variable t it becomes possible to use the *mult* command (with *DDt* being used in our implementation to symbolize $\frac{d}{dt}$) of the *DETools* package of *Maple 11*. The function *mskew* (Input: $A \in \mathcal{M}_{r_1, r_2}[\frac{d}{dt}]$, $B \in \mathcal{M}_{r_2, r_3}[\frac{d}{dt}]$; Output: $C = AB \in \mathcal{M}_{r_1, r_3}[\frac{d}{dt}]$) provides a multiplication which corresponds to (21). Note furthermore that specifying the associated Cartan field to deal with functions of t only is not necessary (a discussion of this is done Section 4.4).

4.2 Smith-Decomposition of elements of $\mathcal{M}_{r,s}[\frac{d}{dt}]$

With the matrix multiplication over $\mathcal{R}[\frac{d}{dt}]$ using *mskew*, a Smith decomposition of matrices $A \in \mathcal{M}_{r,s}[\frac{d}{dt}]$ can be implemented by adapting, e.g., the algorithm given in [11] for polynomial matrices with constant coefficients to the non-commutative case by constructing suitable unimodular matrices for left and right actions (see [5] for more details). The resulting *Maple* procedure has been called *Smith_sa* (Input: $A \in \mathcal{M}_{r,s}[\frac{d}{dt}]$; Output: $U \in R - \text{Smith}(A)$ and $V \in L - \text{Smith}(A)$). The *DETools* package of *Maple 11* provides all necessary operations.

4.3 The structure of elements of $\mathcal{L}_q((\Lambda(\mathcal{X}))^m)$

The elements of an operator $\mu \in \mathcal{L}_q((\Lambda(\mathcal{X}))^m)$ have the structure

$$[\mu]_{ij} = \sum_{k \geq 0} \mu_{ijk} \wedge \frac{d^k}{dt^k}, \quad i, j = 1, 2, \dots, m \quad (22)$$

where μ_{ijk} is an arbitrary q -form, which means that, for every $\omega \in (\Lambda^p(\mathcal{X}))^m$, $\mu\omega \in (\Lambda^{p+q}(\mathcal{X}))^m$ is given by

$$[\mu\omega]_i = \sum_{j=1}^m \sum_{k \geq 0} \mu_{ijk} \wedge L_{\tau_{\mathcal{X}}}^k \omega_j, \quad i = 1, 2, \dots, m \quad (23)$$

To implement operators $\mu \in \mathcal{L}_q((\Lambda(\mathcal{X}))^m)$ in *Maple 11* we define the operator itself by specifying it of (*Maple*-) type *Matrix*, the components being differential forms on a previously specified *Frame*. A *Frame* fixes the coordinates of a jet space in the *DifferentialGeometry* package. The coefficients of the differential forms are then specified as polynomials in $\frac{d}{dt}$. Thus the evaluation of μ on a p -form ω , according to (23), uses both $\wedge$ and $\frac{d}{dt}$. Therefore, a special function, called *Dtwedge* (Input: $\omega \in (\Lambda^p(\mathcal{X}))^m$, $\omega \in (\Lambda^p(\mathcal{X}))^m$; Output: $\mu\omega \in (\Lambda^{p+q}(\mathcal{X}))^m$) is introduced. The core algorithm of this function, for fixed i and j , results to (assuming that μ has some finite degree l w.r.t $\frac{d}{dt}$)

- set $k=0$
- set $\gamma=0$
- while** $k \leq l$ **do**
 - extract the coefficient μ_{ijk} of μ_{ij} (using the command *DGmap* of the *DifferentialGeometry* package together with

the standard *Maple* command *coeff*)

- compute $\frac{d^k}{dt^k} \omega$
- set $\gamma = \gamma + \mu_{ijk} \wedge \frac{d^k}{dt^k} \omega$
- set $k=k+1$
- end while**
- set $\mu\omega = \gamma$

Note that, in this case, the time derivative of ω in step 2 has to be implemented using the Lie-Derivative with respect to a cartan field on the previously defined jet space of a suitable prolongation order (a more detailed discussion of the choice of the truncation order is given in Section 4.4). It becomes clear that, though the evaluation of (23) is a combination of standard operations, its implementation is not obvious in standard computer algebra systems.

4.3.1 Multiplication of two elements $\mu \in \mathcal{L}_{q_1}((\Lambda(\mathcal{X}))^m)$ and $\kappa \in \mathcal{L}_{q_2}((\Lambda(\mathcal{X}))^m)$

We consider $\mu = \sum_{k \geq 0} \mu_k \wedge \frac{d^k}{dt^k}$ where the μ_k 's are matrices whose entries are in $\Lambda^{q_1}(\mathcal{X})$, and $\kappa = \sum_{k \geq 0} \kappa_k \wedge \frac{d^k}{dt^k}$ where the κ_k 's are matrices whose entries are in $\Lambda^{q_2}(\mathcal{X})$. The product $\mu\kappa$ is evaluated by considering the product $\mu\kappa\omega$ for every $\omega \in (\Lambda(\mathcal{X}))^m$:

$$\begin{aligned} \mu\kappa\omega &= \mu(\kappa\omega) = \sum_{k_1 \geq 0} \mu_{k_1} \wedge L_{\tau_{\mathcal{X}}}^{k_1} \left(\sum_{k_2 \geq 0} \kappa_{k_2} \wedge L_{\tau_{\mathcal{X}}}^{k_2} \omega \right) \\ &= \sum_{k_1, k_2 \geq 0} \sum_{k_3=0}^{k_1} \binom{k_1}{k_3} \mu_{k_1} \wedge \left(L_{\tau_{\mathcal{X}}}^{k_1-k_3} \kappa_{k_2} \right) \wedge \left(L_{\tau_{\mathcal{X}}}^{k_2+k_3} \omega \right) \end{aligned}$$

Therefore:

$$\mu\kappa = \sum_{k_1, k_2 \geq 0} \sum_{k_3=0}^{k_1} \binom{k_1}{k_3} \mu_{k_1} \wedge \left(L_{\tau_{\mathcal{X}}}^{k_1-k_3} \kappa_{k_2} \right) \wedge \frac{d^{k_2+k_3}}{dt^{k_2+k_3}} \quad (24)$$

In particular, setting $\mu = \frac{d}{dt}$, we immediately deduce that:

$$\frac{d}{dt} \kappa = \sum_k \left(L_{\tau_{\mathcal{X}}} \kappa_k \wedge \frac{d^k}{dt^k} + \kappa_k \wedge \frac{d^{k+1}}{dt^{k+1}} \right) \quad (25)$$

The multiplication of two such operators is done with the function *Dtwedgeop*. Its input is: $\mu \in \mathcal{L}_{q_1}((\Lambda(\mathcal{X}))^m)$, $\kappa \in \mathcal{L}_{q_2}((\Lambda(\mathcal{X}))^m)$ and output: $\mu\kappa \in \mathcal{L}_{q_1+q_2}((\Lambda(\mathcal{X}))^m)$. Its core algorithm using *Maple* code results to (assuming again that μ has a finite degree l w.r.t. $\frac{d}{dt}$)

- set $j=0$
- compute $\gamma = \mu \wedge \kappa$
- while** $k \leq l$ **do**
 - compute $\kappa^{(j+1)} = \frac{d}{dt} \kappa^{(j)}$ according to (25)
 - set $\gamma = \gamma + \mu_{j+1} \kappa^{(j+1)}$
- end while**
- set $\mu\kappa = \gamma$

4.3.2 The operator $\mathfrak{d}$

At this point we investigate the definition (17) of the operator $\mathfrak{d} \in \mathcal{L}_1((\Lambda(\mathcal{X}))^m)$.

We first remark that, for $m=1$, if μ is a 0th-order polynomial w.r.t. $\frac{d}{dt}$, i.e. $\mu = \mu_0 \wedge$ with $\mu_0 \in \Lambda^q(\mathcal{X})$, (17) boils down to the usual anti-derivation property of the exterior derivative, i.e. for every $\omega \in \Lambda^p(\mathcal{X})$

$$d(\mu\omega) = \underbrace{d\mu_0 \wedge \omega}_{\mathfrak{d}(\mu)} + (-1)^q \mu_0 \wedge d\omega \quad (26)$$

Then, going back to the general case, with $\mu \in \mathcal{L}_q((\Lambda(\mathcal{X}))^m)$

and $\omega \in (\Lambda^p(\mathfrak{X}))^m$, we have, for $i = 1, 2, \dots, m$,

$$\begin{aligned} [d(\mu\omega)]_i &= d\left(\sum_{j=1}^m \sum_{k \geq 0} \mu_{ijk} \wedge L_{\tau_{\mathfrak{X}}}^k \omega_j\right) \\ &= \left(\sum_{j=1}^m \sum_{k \geq 0} d\mu_{ijk} \wedge L_{\tau_{\mathfrak{X}}}^k \omega_j\right. \\ &\quad \left.+ (-1)^q \sum_{j=1}^m \sum_{k \geq 0} \mu_{ijk} \wedge d(L_{\tau_{\mathfrak{X}}}^k \omega_j)\right) \quad (27) \\ &= \left(\sum_{j=1}^m \sum_{k \geq 0} d\mu_{ijk} \wedge L_{\tau_{\mathfrak{X}}}^k \omega_j\right. \\ &\quad \left.+ (-1)^q \sum_{j=1}^m \sum_{k \geq 0} \mu_{ijk} \wedge L_{\tau_{\mathfrak{X}}}^k (d\omega_j)\right) \end{aligned}$$

We then calculate $\mathfrak{d}(\mu)\omega$ by combining (17) with (27):

$$\begin{aligned} [\mathfrak{d}(\mu)\omega]_i &= [d(\mu\omega) - (-1)^q \mu d\omega]_i \\ &= \sum_{j=1}^m \sum_{k \geq 0} d\mu_{ijk} \wedge L_{\tau_{\mathfrak{X}}}^k \omega_j \quad (28) \end{aligned}$$

As a consequence, the entries of $\mathfrak{d}(\mu)$ are simply given by

$$[\mathfrak{d}(\mu)]_{ij} = \sum_{k \geq 0} d\mu_{ijk} \wedge \frac{d^k}{dt^k}, \quad i, j = 1, 2, \dots, m \quad (29)$$

Note that operators $\mu \in \mathcal{L}_q((\Lambda(\mathfrak{X}))^m)$ are specified for *Maple* as polynomials in DDt , the latter operator being a constant with respect to the Exterior Derivative on the Jet-Manifold $\mathfrak{X}$ with coordinates $(x, \dot{x}, \dots)$. Thus, for the chosen implementation in *Maple*, $\mathfrak{d}$ boils down to applying the *ExteriorDerivative* command of the *DifferentialGeometry* package to the chosen representation of μ , which readily gives the procedure *gdmu* (Input: $\mu_0 \in \Lambda^q(\mathfrak{X})$, Output: $\mathfrak{d}(\mu_0) \in \Lambda^{q+1}(\mathfrak{X})$).

4.4 Iterative increase of truncation order and degree

The Algorithm 3.3 includes a large number of degrees of freedom. There are many choices in the Smith decompositions yielding the vector 1-form ω , but any other choice of a basis of the ideal Ω is, at least algebraically, equivalent. Nevertheless, the set of operators μ such that $d\omega = \mu\omega$, which is always non empty, generally contains infinitely many elements. A lower bound of its degree w.r.t. $\frac{d}{dt}$ is easily computed but there is no a priori upper bound. The only restriction is that the equation $\mathfrak{d}(\mu) = \mu^2$ must be satisfied. If we note $\mu = \sum_{k \geq 0} \mu_k \wedge \frac{d^k}{dt^k}$, as a consequence of (29) and (24), the matrices μ_k must satisfy the infinite sequence of differential equations:

$$d\mu_k = \sum_{k_2=0}^k \sum_{k_1 \geq k-k_2} \binom{k_1}{k-k_2} \mu_{k_1} \wedge \left(L_{\tau_{\mathfrak{X}}}^{k_1+k_2-k} \mu_{k_2}\right) \quad (30)$$

for all $k \geq 0$. However, since the degree of μ w.r.t. $\frac{d}{dt}$ is finite, i.e. only finitely many μ_k 's are non zero, on the one hand, and since μ depends only on a finite number of coordinates of $\mathfrak{X}$ on the other hand, the number of non trivial equations in (30) is indeed finite. Moreover, (30) establishes a link between the number of coordinates that are active via the expression of $d\mu_k$, and the polynomial degree of μ . Therefore, for a given truncation order ² and a given degree which are compatible relatively to (30), solutions μ and M , if they exist, may be found using the above computer algebra program. If not, the truncation order and/or the degree can be increased. Unfortunately, there is no simple answer to the question “is this process ending?”.

²by truncation, we refer to the elimination of the infinitely many coordinates of which μ doesn't depend

Once μ is determined, according to step 4, a suitable unimodular matrix $M \in \mathcal{U}_p\left[\frac{d}{dt}\right]$ satisfying $\mathfrak{d}(M) = -M\mu$, has to be found. As a first guess an initial M can always be chosen as an upper triangular Matrix with ones on the main diagonal, the north-eastern entries of M being polynomials in $\frac{d}{dt}$ of suitable degree with coefficients depending on the truncated coordinates. If the provided free parameters in M are not sufficient to find a solution, more complicated unimodular matrices can be constructed by left and right multiplication with elementary left and right actions (see e.g. [5]).

If in step 4 no suitable matrix M can be found then it may be necessary to go back to step 3 and to increase the truncation order and/or the degree of μ to introduce additional degrees of freedom.

Using Maple 11 and the chosen implementation of the operators, construction methods of general operators $\mu \in$ and matrices $M \in$ for given truncation orders and degrees with respect to $\frac{d}{dt}$ could be implemented. This easily allows to iteratively increase the used truncation order and degrees which is essential for the application of this approach.

5. NON HOLONOMIC CAR

Consider the 3 dimensional system in the $x-y$ plane, representing a vehicle of length l , whose orientation is given by the angle θ , the coordinates (x, y) standing for the position of the middle of the rear axle, and controlled by the velocity modulus u and the angular position of the front wheels φ .

$$\begin{aligned} \dot{x} &= u \cos \theta \\ \dot{y} &= u \sin \theta \\ \dot{\theta} &= \frac{u}{l} \tan \varphi \end{aligned} \quad (31)$$

Since $n = 3$ and $m = 2$, $n - m = 1$ and (31) is equivalent to the single implicit equation obtained by eliminating the inputs u and φ :

$$F(x, y, \theta, \dot{x}, \dot{y}, \dot{\theta}) = \dot{x} \sin \theta - \dot{y} \cos \theta = 0 \quad (32)$$

We immediately have:

$$\begin{aligned} P(F) &= \left(\frac{\partial F}{\partial x} + \frac{\partial F}{\partial \dot{x}} \frac{d}{dt} \quad \frac{\partial F}{\partial y} + \frac{\partial F}{\partial \dot{y}} \frac{d}{dt} \quad \frac{\partial F}{\partial \theta} + \frac{\partial F}{\partial \dot{\theta}} \frac{d}{dt} \right) \\ &= \left(\sin \theta \frac{d}{dt} \quad -\cos \theta \frac{d}{dt} \quad \dot{x} \cos \theta + \dot{y} \sin \theta \right). \end{aligned} \quad (33)$$

In the following, all steps of Algorithm 3.3 are performed using the above presented Maple functions. Note that this well-known example has been chosen as it is at the same time challenging enough to illustrate most of the properties of the proposed approach and can at the same time be discussed in detail.

Step 1: Setting $E = \dot{x} \cos \theta + \dot{y} \sin \theta$, we apply the Smith decomposition algorithm (i.e. we apply the function *Smith_sa*) and we get $U \in \mathbb{R} - \text{Smith}(P(F))$ with

$$U = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ \frac{1}{E} & \frac{\cos \theta}{E} \frac{d}{dt} & -\frac{\sin \theta}{E} \frac{d}{dt} \end{pmatrix}.$$

Thus

$$\hat{U} = U \begin{pmatrix} 0_{1,2} \\ I_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ \frac{\cos \theta}{E} \frac{d}{dt} & -\frac{\sin \theta}{E} \frac{d}{dt} \end{pmatrix}$$

with I_2 the identity matrix of $\mathbb{R}^2$. Again, computing $Q \in \mathbf{L} - \mathbf{Smith}(\hat{U})$ yields

$$Q = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ \frac{\sin \theta}{E} \frac{d}{dt} & -\frac{\cos \theta}{E} \frac{d}{dt} & 1 \end{pmatrix}.$$

Multiplying Q by the vector $(dx, dy, d\theta)^T$, the last line reads $\frac{1}{E}(\sin \theta d\dot{x} - \cos \theta d\dot{y} + (\dot{x} \cos \theta + \dot{y} \sin \theta) d\theta) = \frac{1}{E}d(\dot{x} \sin \theta - \dot{y} \cos \theta)$ and, by (32), identically vanishes on $\mathfrak{X}_0$.

The remaining part of the system, namely

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} dx \\ dy \\ d\theta \end{pmatrix} = \begin{pmatrix} \omega_1 \\ \omega_2 \end{pmatrix}$$

is trivially strongly closed with $M = I_2$, which finally gives the flat output (which we denote here by y_f to avoid confusion with the coordinate y) $y_f = (y, x)^T$. We have thus recovered the flat output originally obtained in [27, 26], up to a permutation of the components of y .

Step 1 b: Other decompositions of $P(F)$, given by (33), may indeed be obtained. They are all equivalent in the sense that one decomposition may be deduced from another one by multiplication by a unimodular matrix. However, the resulting vector 1-form ω , contrarily to what happens in the previous example, may not be integrable. Our aim is here to show how the generalized moving frame structure equations (18) may be used to obtain an integrable $M\omega$. Such an example is provided by restarting the right-Smith decomposition of $P(F)$ by right-multiplying it by

$$\begin{pmatrix} \cos \theta & 0 & 0 \\ \sin \theta & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and using the formula $\sin \theta \frac{d}{dt}(\cos \theta) - \cos \theta \frac{d}{dt}(\sin \theta) = -\dot{\theta}$. The Smith decomposition algorithm yields

$$U = \begin{pmatrix} \cos \theta & -\frac{1}{\theta} \cos^2 \theta \frac{d}{dt} & \frac{1}{\theta} E \cos \theta \\ \sin \theta & 1 - \frac{1}{\theta} \sin \theta \cos \theta \frac{d}{dt} & \frac{1}{\theta} E \sin \theta \\ 0 & 0 & 1 \end{pmatrix},$$

i.e.

$$\hat{U} = \begin{pmatrix} -\frac{1}{\theta} \cos^2 \theta \frac{d}{dt} & \frac{1}{\theta} E \cos \theta \\ 1 - \frac{1}{\theta} \sin \theta \cos \theta \frac{d}{dt} & \frac{1}{\theta} E \sin \theta \\ 0 & 1 \end{pmatrix}$$

The Smith decomposition of $\hat{U}$ then yields $Q \in \mathbf{L} - \mathbf{Smith}(\hat{U})$ with

$$Q = \begin{pmatrix} -\tan \theta & 1 & 0 \\ 0 & 0 & 1 \\ -\frac{1}{\theta} \sin \theta \cos \theta \frac{d}{dt} & \frac{1}{\theta} \cos^2 \theta \frac{d}{dt} & -\frac{1}{\theta} E \cos \theta \end{pmatrix}.$$

Step 2: In this case ω is obtained as

$$\omega = (\omega_1, \omega_2)^T = \hat{Q} (dx, dy, d\theta)^T = (-\tan \theta dx + dy, d\theta)^T$$

Its exterior derivative is non zero:

$$d\omega = (d\omega_1, d\omega_2)^T = (-\frac{1}{\cos^2 \theta} d\theta \wedge dx, 0)^T$$

showing that ω is not closed. The simplest possible operator μ is of truncated order 0 and degree 0:

$$\mu = \begin{pmatrix} 0 & \mu_{120} \\ 0 & 0 \end{pmatrix} \wedge$$

with $\mu_{120} = \mu_{1210}(x, y, \theta)dx + \mu_{1220}(x, y, \theta)dy + \mu_{1230}(x, y, \theta)d\theta$. Evaluation of $\mu\omega$ using the *Dtwedge* function yields

$$\mu\omega = \begin{pmatrix} \mu_{120} \wedge d\theta \\ 0 \end{pmatrix}$$

with $\mu_{120} \wedge d\theta = \mu_{1210}dx \wedge d\theta + \mu_{1220}dy \wedge d\theta$. The comparison of $d\omega$ and $\mu\omega$ yields $\mu_{1210} = \frac{1}{\cos^2 \theta}$ and $\mu_{1220} = 0$, or

$$\mu = \begin{pmatrix} 0 & \frac{1}{\cos^2 \theta} dx + \mu_{1230}(x, y, \theta)d\theta \\ 0 & 0 \end{pmatrix} \wedge$$

Step 3: By direct computation (using *Dtwedgeop*):

$$\mu^2 = \begin{pmatrix} 0 & \mu_{120} \\ 0 & 0 \end{pmatrix} \wedge \begin{pmatrix} 0 & \mu_{120} \\ 0 & 0 \end{pmatrix} = 0$$

On the other hand, using *gdmu*, we have

$$\mathfrak{d}(\mu) = \begin{pmatrix} 0 & d\left(\frac{1}{\cos^2 \theta} dx + \mu_{1230}(x, y, \theta)d\theta\right) \\ 0 & 0 \end{pmatrix} \wedge$$

From $\mu^2 = \mathfrak{d}(\mu)$ we obtain the system of P.D.E.'s

$$\begin{aligned} \frac{\partial}{\partial x} \mu_{1230}(x, y, \theta) &= \frac{2 \sin(\theta)}{\cos^3(\theta)} \\ \frac{\partial}{\partial y} \mu_{1230}(x, y, \theta) &= 0 \end{aligned}$$

whose solution, using *pdsolve*, is

$$\mu_{1230}(x, y, \theta) = \frac{2 \sin(\theta)x}{\cos^3(\theta)} + C_1(\theta). \quad (34)$$

Thus

$$\mu = \begin{pmatrix} 0 & \frac{1}{\cos^2 \theta} dx + \left(\frac{2 \sin(\theta)x}{\cos^3(\theta)} + C_1(\theta)\right) d\theta \\ 0 & 0 \end{pmatrix}.$$

Step 4/5: The simplest unimodular matrix has truncation order 0 and degree 0:

$$M = \begin{pmatrix} 1 & m_{120}(x, y, \theta) \\ 0 & 1 \end{pmatrix}.$$

Computing $\mathfrak{d}(M) = -M\mu$, we get $\begin{pmatrix} 0 & dm_{120} \\ 0 & 0 \end{pmatrix} = \mu$, i.e.

$$\begin{aligned} \frac{\partial}{\partial x} m_{120}(x, y, \theta) &= -\frac{1}{\cos^2 \theta} \\ \frac{\partial}{\partial y} m_{120}(x, y, \theta) &= 0 \\ \frac{\partial}{\partial \theta} m_{120}(x, y, \theta) &= -\left(\frac{2 \sin(\theta)x}{\cos^3(\theta)} + C_1(\theta)\right) \end{aligned}$$

whose solution is (determined again with *pdsolve*)

$$m_{120} = -\frac{x}{\cos^2(\theta)} + C_2(\theta) \quad (35)$$

with $C_1(\theta)$ in (34) given by $C_1(\theta) = -\frac{d}{d\theta} C_2(\theta)$. It results that

$$M = \begin{pmatrix} 1 & -\frac{x}{\cos^2(\theta)} + C_2(\theta) \\ 0 & 1 \end{pmatrix}.$$

Thus, we get as differential of a flat output, which we denote here as above by y_f

$$dy_f = M\omega = \begin{pmatrix} -\tan(\theta)dx + dy + \left(-\frac{x}{\cos^2(\theta)} + C_2(\theta)\right)d\theta \\ d\theta \end{pmatrix}$$

This one-form is closed and, using *pdsolve* we obtain the flat output

$$y_f = (y - x \tan(\theta) + C_3(\theta), \theta)^T.$$

6. CONCLUSIONS

In this contribution the necessary and sufficient conditions for differential flatness of nonlinear control systems have been discussed with the aim of arriving at a computer algebra implementation. To this end general formulae for the used operators have been deduced from its defining relations. It could be shown that all used operators can be implemented using, e.g., the computer algebra system *Maple 11*. However, as operations from differential geometry as well as from algebra were needed, the operators could not be directly implemented but special functions had to be created to implement the action of the results on differential forms or other operators. We want to emphasize that this paper is a first step towards the formal computation of flat output where computational costs are voluntarily ignored. Obtaining more efficient algorithms will be the subject of future work.

7. REFERENCES

- [1] E. Aranda-Bricaire, C. Moog, and J.-B. Pomet. A linear algebraic framework for dynamic feedback linearization. *IEEE Trans. Automat. Contr.*, 40(1):127–132, 1995.
- [2] B. Charlet, J. Lévine, and R. Marino. Sufficient conditions for dynamic state feedback linearization. *SIAM J. Control Optimization*, 29(1):38–57, 1991.
- [3] S. Chern, W. Chen, and K. Lam. *Lectures on Differential Geometry*, volume 1 of *Series on University Mathematics*. World Scientific, 2000.
- [4] V. Chetverikov. New flatness conditions for control systems. In *Proceedings of NOLCOS'01, St. Petersburg*, pages 168–173, 2001.
- [5] P. Cohn. *Free Rings and Their Relations*. Academic Press, London, 1985.
- [6] M. Fliess. A remark on Willems' trajectory characterization of linear controllability. *Systems & Control Letters*, 19:43–45, 1992.
- [7] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Sur les systèmes non linéaires différentiellement plats. *C.R. Acad. Sci. Paris*, I-315:619–624, 1992.
- [8] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. Flatness and defect of nonlinear systems: introductory theory and examples. *Int. J. Control*, 61(6):1327–1361, 1995.
- [9] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. A Lie-Bäcklund approach to equivalence and flatness of nonlinear systems. *IEEE Trans. Automat. Control*, 44(5):922–937, 1999.
- [10] J. Franch. *Flatness, Tangent Systems and Flat Outputs*. PhD thesis, Universitat Politècnica de Catalunya Jordi Girona, 1999.
- [11] I. Gohberg, P. Lancaster, and L. Rodman. *Matrix Polynomials*. Academic Press, Berlin, 1982.
- [12] B. Jakubczyk. Invariants of dynamic feedback and free systems. In *Proc. ECC'93, Groningen*, pages 1510–1513, 1993.
- [13] I. S. Krasil'shchik, V. V. Lychagin, and A. M. Vinogradov. *Geometry of Jet Spaces and Nonlinear Partial Differential Equations*. Gordon and Breach, New York, 1986.
- [14] J. Lévine. On flatness necessary and sufficient conditions. In *Proc. of IFAC NOLCOS 2004 Conference*, Stuttgart, 2004.
- [15] J. Lévine. On necessary and sufficient conditions for differential flatness. *arXiv:math.OA/0605405v2*, pages 1–28, 2006.
- [16] J. Lévine and D. Nguyen. Flat output characterization for linear systems using polynomial matrices. *Systems & Control Letters*, 48:69–75, 2003.
- [17] P. Martin. *Contribution à l'étude des systèmes différentiellement plats*. PhD thesis, École des Mines de Paris, 1992.
- [18] P. Martin, R. Murray, and P. Rouchon. Flat systems. In G. Bastin and M. Gevers, editors, *Plenary Lectures and Minicourses, Proc. ECC 97, Brussels*, pages 211–264, 1997.
- [19] P. Martin and P. Rouchon. Systems without drift and flatness. In *Proc. MTNS 93, Regensburg, Germany*, August 1993.
- [20] P. Pereira da Silva and C. C. Filho. Relative flatness and flatness of implicit systems. *SIAM J. Control and Optimization*, 39(6):1929–1951, 2001.
- [21] P. S. Pereira da Silva. Flatness of nonlinear control systems : a Cartan-Kähler approach. In *Proc. Mathematical Theory of Networks and Systems (MTNS'2000)*, Perpignan, pages 1–10, 2000.
- [22] J.-B. Pomet. A differential geometric setting for dynamic equivalence and dynamic linearization. In B. Jakubczyk, W. Respondek, and T. Rzezuchowski, editors, *Geometry in Nonlinear Control and Differential Inclusions*, pages 319–339. Banach Center Publications, Warsaw, 1993.
- [23] J.-B. Pomet. On dynamic feedback linearization of four-dimensional affine control systems with two inputs. *ESAIM-COCV*, 1997. <http://www.emath.fr/Maths/Cocv/Articles/articleEng.html>.
- [24] M. Rathinam and R. Murray. Configuration flatness of Lagrangian systems underactuated by one control. *SIAM J. Control and Optimization*, 36(1):164–179, 1998.
- [25] P. Rouchon. Necessary condition and genericity of dynamic feedback linearization. *J. Math. Systems Estim. Control*, 4(2):257–260, 1994.
- [26] P. Rouchon, M. Fliess, J. Lévine, and P. Martin. Flatness and motion planning: the car with n-trailers. In *Proc. ECC'93, Groningen*, pages 1518–1522, 1993.
- [27] P. Rouchon, M. Fliess, J. Lévine, and P. Martin. Flatness, motion planning and trailer systems. In *Proc. IEEE Conf. Decision and Control, San Antonio*, December 1993.
- [28] W. Shadwick. Absolute equivalence and dynamic feedback linearization. *Systems Control Letters*, 15:35–39, 1990.
- [29] W. Sluis. A necessary condition for dynamic feedback linearization. *Systems Control Letters*, 21:277–283, 1993.
- [30] M. van Nieuwstadt, M. Rathinam, and R. Murray. Differential flatness and absolute equivalence of nonlinear control systems. *SIAM J. Control Optim.*, 36(4):1225–1239, 1998.
- [31] V. Zharinov. *Geometrical Aspect of Partial Differential Equations*. World Scientific, Singapore, 1992.

Regular Systems of Linear Functional Equations and Applications

Moulay A. Barkatou
Institut XLIM
UMR 6172, CNRS
Université de Limoges
123, Av. A. Thomas
87060 Limoges
France
moulay.barkatou@xlim.fr

Gary Broughton, Eckhard Pflügel
Faculty of CISM
Kingston University
Penrhyn Road
Kingston upon Thames, Surrey KT1 2EE
United Kingdom
G.Broughton@kingston.ac.uk,
E.Pfluegel@kingston.ac.uk

ABSTRACT

The algorithmic classification of singularities of linear differential systems via the computation of Moser- and super-irreducible forms as introduced in [21] and [16] respectively has been widely studied in Computer Algebra ([8, 12, 22, 6, 10]). Algorithms have subsequently been given for other forms of systems such as linear difference systems [4, 3] and the perturbed algebraic eigenvalue problem [18]. In this paper, we extend these concepts to the general class of systems of linear functional equations. We derive a definition of regularity for these type of equations, and an algorithm for recognizing regular systems. When specialised to q -difference systems, our results lead to new algorithms for computing polynomial solutions and regular formal solutions.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms

General Terms

Algorithms

Keywords

Computer Algebra, Systems of Linear Functional Equations, Moser-reduction, Super-reduction, Singularities.

1. INTRODUCTION

In this paper, we are concerned with the class of *systems of linear functional equations* which comprises common types of systems such as linear differential or linear (q -)difference systems.

The theoretical framework for these types of systems is *Pseudo-Linear Algebra*, which was introduced by Jacobson in [17].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC '08, July 20–23, 2008, Hagenberg, Austria.
Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Bronstein and Petkovšek in [14] review the basic objects of Pseudo-Linear Algebra in the context of Computer algebra. It is in this spirit that the first author of the current paper treats in [9] the problem of (global) factorization of linear functional systems. In this paper we are interested in the *local analysis* of these type of systems. For this we first introduce (see Section 3.1) a unifying framework that permits us to treat, simultaneously, all types of linear functional systems. This is done by using the language of pseudo-linear derivations over a field of discrete valuation [19]. Once this common framework is established, we give a definition of regularity for pseudo-linear systems (see Section 3.2) and develop a method for recognizing regular systems inspired by Moser's work on differential equations (see Section 3.3).

The rest of the paper is organized as follows. In Section 2 we present the concepts of *regularity* and *Moser-reducibility* of linear differential systems, and outline the principle of implementing the Moser-reduction. Our presentation aims to help the reader understand what motivates the way in which similar concepts are introduced in the following sections. Similarly, Section 3.4 deals with the concept of *super-reduction*, which is a generalization of the Moser-reduction. In Section 4 we give two algorithms for Moser- and super-reduction for linear functional equations. These algorithms are obtained from the algorithms as reviewed in Section 4.1 and Section 4.3. The originality of our approach is to prove that replacing, at each reduction step, the differential transformation (5) by the pseudo-linear transformation (10) results in valid reduction algorithms. Algorithms for the Moser- and super reduction of differential and difference systems have been given in the past, however to our knowledge, Moser- and super-irreducible form, have not been defined for q -difference systems. As an application we show in Section 5.1, that the approach developed in [7, 2] for computing polynomial solution (and more generally formal series solutions at infinity) for differential and difference systems is also applicable for q -difference systems. Our algorithms are implemented in the Maple package ISOLDE [11] and two examples of computation using this package are given in Section 6.

2. THE DIFFERENTIAL CASE

Let K be a subfield of $\mathbb{C}$, and denote by $K[[x]]$ the ring of formal power series in x over K and $K((x))$ its quotient field. If $a \in K((x))$, we denote the x -adic valuation of a by

$v(a)$ and if $A = (a_{i,j})$ is a matrix with coefficients in $K((x))$, we put $v(A) = \min v(a_{ij})$. Let ϑ denote the derivation $x \frac{d}{dx}$ of $K((x))$.

A linear differential system with a singularity at the origin is a system of the form

$$\vartheta Y = A(x)Y \quad (1)$$

where Y is an n -dimensional unknown vector and $A(x)$ is an $n \times n$ matrix with entries in $K((x))$. The matrix A can be expanded as:

$$A = x^{-r} \sum_{j=0}^{\infty} A_j x^j \quad (A_0 \neq 0) \quad (2)$$

where $A_j \in \mathcal{M}_n(K)$ and r is a non-negative integer called the *Poincaré-rank* of the system.

Performing the change of variable

$$Y = T(x)Z \quad (3)$$

where $T \in GL(n, K((x)))$ in (1) yields the following *equivalent* system

$$\vartheta Z = B(x)Z \quad (4)$$

where

$$B = T_{\vartheta}[A] := T^{-1}AT - T^{-1}\vartheta T. \quad (5)$$

A system (1) is called a *regular system* if it is equivalent to a system of Poincaré rank $r = 0$.

Given a system (1) with Poincaré rank $r > 0$, we wish to compute an equivalent system for which r is minimal. This is useful as it allows a regular system to be detected – in this case, the minimal Poincaré rank is $r = 0$.

This problem can be solved by using the concept of *Moser-reducibility* as defined in [21]. Following Moser we introduce the two following rational numbers

$$m(A) = \begin{cases} -v(A) + \frac{\text{rank}(A_0)}{n} & \text{if } v(A) < 0 \\ 0 & \text{if } v(A) \geq 0 \end{cases}$$

and

$$\mu(A) = \min_{T \in GL(n, K((x)))} (m(T_{\vartheta}[A])).$$

The system (1) is regular if and only if $\mu(A) = 0$. The matrix A is called *Moser-irreducible* if $m(A) = \mu(A)$ otherwise it is called *Moser-reducible*.

The reason for introducing $m(A)$ lies in the fact that it is possible to give a necessary and sufficient condition for reducibility which only involves A_0 and A_1 , as stated in [21]: If $r > 0$ then the matrix A is reducible if and only if the associated *Moser-polynomial*

$$\theta(\lambda) := x^{\text{rank } A_0} \det(x^{-1}A_0 + A_1 - \lambda I) \Big|_{x=0} \quad (6)$$

vanishes identically in λ .

We prove in [12] that if A is reducible, one can construct a matrix S of the form

$$S(x) = C \text{diag}(x^{d_1}, \dots, x^{d_n})$$

where C is an invertible constant matrix and where $d_i \in \{0, 1\}$, such that $m(S[A]) < m(A)$. Applying the above procedure, several times if necessary, $\mu(A)$ can be determined. Furthermore, a matrix polynomial T such that $m(T[A]) = \mu(A)$ is computed in this way.

Inspired by Moser's work, Dietrich [15] appears to be the first author to provide an effective direct algorithm for computing $\mu(A)$ without having to convert the system to a scalar equation. Apparently unaware of his approach, Hilali and Wazner [16] published a different method which they extended to compute the so-called *super-irreducible* forms of linear differential systems. The first author of this paper has given versions of these algorithms in [5, 8] for systems with rational function coefficients. Using a different approach, Levelt [20] has given an algorithm which can also be used to compute a system with minimal Poincaré-rank.

Inspired by all the previous works the first and third author of this paper have introduced new algorithms for the Moser- and super-reduction [12] (see Sections 4.1 and 3.4 for more details) which give alternatives to previous methods and require, in general, less reduction steps and are hence faster. The aim of the present paper is to show how these algorithms can be extended relatively easily to handling general systems of linear functional equations.

3. SYSTEMS OF PSEUDO-LINEAR EQUATIONS

3.1 Basic Definitions and Notation.

Discrete valuation fields – Basic Objects

In this section, and throughout the remainder of the paper, we let F be a commutative discrete valuation field of characteristic zero and denote by v the valuation of F . We introduce the basic objects we are concerned with in this paper, for more details we refer to classical algebra textbooks such as [19].

Recall that $v : F \longrightarrow \mathbb{Z} \cup \{+\infty\}$ is a map with the following properties : for $f, g \in F$ one has

1. $v(f) = +\infty \iff f = 0$,
2. $v(fg) = v(f) + v(g)$,
3. $v(f+g) \geq \min(v(f), v(g))$, and equality holds if $v(f) \neq v(g)$.

The *valuation ring* of F is $\mathcal{O} = \{f \in F : v(f) \geq 0\}$. The set $\mathfrak{M} = \{f \in F : v(f) > 0\}$ coincides with the set of non-invertible elements of $\mathcal{O}$. $\mathfrak{M}$ is the (unique) maximal ideal of $\mathcal{O}$, so $\bar{F} := \mathcal{O}/\mathfrak{M}$ is a field. It is called the residue field of F . Let π denote the canonical homomorphism from $\mathcal{O}$ onto $\bar{F}$. We denote by $\mathcal{U}$ the group of units of $\mathcal{O}$. One has $\mathcal{U} = \mathcal{O} \setminus \mathfrak{M}$.

EXAMPLE 3.1. $F = \mathbb{C}((x))$ the field of formal Laurent power series, equipped with the x -adic valuation. Its valuation ring is $\mathcal{O} = \mathbb{C}[[x]]$. The residue field may be identified with $\mathbb{C}$, and $\pi(f) = f(0)$ for $f \in \mathcal{O}$.

An element $t \in \mathcal{O}$ is said to be a uniformizing element (or a local parameter) if $v(t) = 1$. Recall that the valuation ring $\mathcal{O}$ is a principal ideal ring, that every proper ideal of $\mathcal{O}$ can be written as $t^m \mathcal{O}$ for some integer $m > 0$ and that the intersection of all proper ideals of $\mathcal{O}$ is the zero ideal. In particular, $\mathfrak{M} = t\mathcal{O}$. As a consequence, each nonzero element $f \in F$ can be uniquely written as $f = ut^{v(f)}$ for some unit $u \in \mathcal{U}$.

In the sequel we suppose that F is complete with respect to the valuation v and fix a local parameter t of F . Let R be a set of representatives of $\mathcal{O}$ in $\bar{F}$, that is a subset R of $\mathcal{O}$ such that $0 \in R$ and the map: $a \in R \mapsto \pi(a) \in \mathcal{O}/\mathfrak{M} = \bar{F}$ is bijective. Then every element $f \in F$ can be uniquely expanded as $f = \sum_{i=\nu}^{+\infty} f_i t^i$, where $\nu = v(f)$ and $f_i \in R$ with $f_\nu \neq 0$.

For $A = (a_{i,j}) \in \mathcal{M}_n(F)$ we define the *valuation* of A , by $v(A) = \min_{i,j} (v(a_{i,j}))$. Every nonzero matrix A with entries in F can be uniquely written as

$$A = t^{-r} \sum_{i=0}^{+\infty} A_i t^i \quad (7)$$

where $r = -v(A)$ and where the A_i are matrices with entries in R with $A_0 \neq 0$.

Pseudo-derivations on discrete valuation fields

Let ϕ be an isometry of F , that is an automorphism of F such that $v(\phi f) = v(f)$ for all $f \in F$. Let δ be a ϕ -derivation, i.e. a map $\delta : F \rightarrow F$ satisfying, for all $a, b \in F$

$$\delta(a+b) = \delta a + \delta b \quad \text{and} \quad \delta(ab) = \phi a \delta b + \delta a b \quad (\text{Leibniz rule})$$

We let $\mathcal{C}$ be the field of *constants* of F

$$\mathcal{C} = \{f \in F : \phi f = f \text{ and } \delta f = 0\}.$$

Note that when $\phi = 1_F$ the identity map of F then δ is simply a standard derivation. Otherwise, one can easily show that δ has to be of the form $\gamma(1_F - \phi)$ for some $\gamma \in F$ (see for example [14]).

Given δ fixed, we define its *degree* as

$$\omega(\delta) = \inf_{f \in F, f \neq 0} v(f^{-1} \delta(f)).$$

Note that δ is continuous, as a $\mathcal{C}$ -linear map from F into itself, if and only if $\omega(\delta) > -\infty$.

LEMMA 3.1. *For all integer k , one has*

$$v(t^{-k} \delta(t^k)) \geq v(t^{-1} \delta(t)).$$

Proof Let us notice first that it suffices to prove the lemma for $k \geq 1$ because $v(t^{-k} \delta(t^k)) = v(t^k \delta(t^{-k}))$. Indeed, for any nonzero $f \in F$ one has $f \delta(f^{-1}) = -\delta(f) \phi(f^{-1})$ and hence $v(f \delta(f^{-1})) = v(\phi(f^{-1}) \delta(f)) = v(f^{-1} \delta(f))$. Now using the Leibniz rule, one can prove, by induction on $k \geq 1$, that

$$t^{-k} \delta(t^k) = t^{-1} \delta(t) \sum_{i=0}^{k-1} u^i$$

where $u = \phi(t)t^{-1}$. The result follows by noticing that $v(u) = 0$. Finally, the lemma is true for $k = 0$. $\square$

LEMMA 3.2. *Suppose $\delta \neq 0$ and is continuous. Then*

$$\omega(\delta) = v(\tau^{-1} \delta(\tau))$$

for some local parameter τ .

Proof Since δ is continuous one has $\omega(\delta) = v(f^{-1} \delta(f))$ for some nonzero element $f \in F$. Let $m = v(f)$. If $m = 1$ then take $\tau = f$, otherwise put $z = t^{1-m} f$. The element z is a local parameter, for $v(z) = v(t) = 1$. Now compute

$$\delta(f) = \delta(t^{m-1} z) = \delta(t^{m-1}) \phi(z) + t^{m-1} \delta(z)$$

and

$$f^{-1} \delta(f) = t^{-(m-1)} \delta(t^{m-1}) z^{-1} \phi(z) + z^{-1} \delta(z).$$

It then follows that

$$\omega(\delta) = v(f^{-1} \delta(f)) \geq \inf(v(t^{-(m-1)} \delta(t^{m-1})), v(z^{-1} \delta(z)))$$

and by using Lemma 3.1

$$\omega(\delta) \geq \inf(v(t^{-1} \delta(t)), v(z^{-1} \delta(z)))$$

Now if $v(t^{-1} \delta(t)) \geq v(z^{-1} \delta(z))$ then take $\tau = z$, otherwise take $\tau = t$. $\square$

EXAMPLE 3.2.

1. $F = \mathbb{C}((x))$, $\phi = 1_F$ and $\delta = \frac{d}{dx}$. In this case $\omega(\delta) = -1 = v(x^{-1} \delta x)$ and $\omega(x \frac{d}{dx}) = 0$.
2. $F = \mathbb{C}((x^{-1}))$ equipped with the valuation: if $f = \sum_{i=i_0}^{+\infty} f_i x^{-i}$ with $f_{i_0} \neq 0$, $v(f) = i_0$. Here $t = x^{-1}$ is a uniformizing element of F . For $\phi = 1_F$ and $\delta = \frac{d}{dx}$ one has $\omega(\delta) = +1 = v(t^{-1} \delta(t)) = v(x \frac{d}{dx})$ while $\omega(x \frac{d}{dx}) = 0$.
3. $F = \mathbb{C}((x^{-1}))$, v as above and $t = x^{-1}$. Let ϕ be the $\mathbb{C}$ -automorphism defined by $\phi x = x - 1$ and $\delta = 1 - \phi$. One has $\omega(\delta) = 1 = v(t^{-1} \delta(t)) = v(x(\frac{1}{x} - \frac{1}{x-1}))$.
4. $F = \mathbb{C}((x^{-1}))$, v as above and $t = 1/x$. Let ϕ be the $\mathbb{C}$ -automorphism defined by $\phi x = qx$, $q \in \mathbb{C} \setminus \{0, 1\}$ and $\delta = 1 - \phi$. In this case $\omega(\delta) = 0 = v(t^{-1} \delta(t)) = v(x(\frac{1}{x} - \frac{1}{qx}))$.

3.2 Regular Pseudo-linear Systems

By a system of *pseudo-linear* equations over (F, ϕ, δ) we mean an equation of the form

$$\delta Y = A \phi Y \quad (8)$$

where A is an $n \times n$ matrix in F and where Y is an unknown n -dimensional column vector. We shall refer to the system (8) by $\delta, \phi[A]$. We note the following well-known examples of systems of pseudo-linear equations:

1. Differential equations: $F = \mathbb{C}((x))$, or $\mathbb{C}((x^{-1}))$, $\phi = 1_F$ and $\delta = \frac{d}{dx}$.
2. Difference equations: $F = \mathbb{C}((x^{-1}))$, $\delta = 1 - \phi$ where ϕ is the $\mathbb{C}$ -automorphism defined by $\phi x = x - 1$.
3. q -Difference equations: $F = \mathbb{C}((x))$ or $\mathbb{C}((x^{-1}))$, ϕ the $\mathbb{C}$ -automorphism defined by $\phi x = qx$, $q \in \mathbb{C} \setminus \{0, 1\}$ and $\delta = 1 - \phi$.

Consider a system of the form (8). The *gauge transformation* $Y = TZ$ with $T \in \text{GL}(n, F)$, leads to the *equivalent* system

$$\delta Z = B \phi Z \quad (9)$$

where B is given by

$$B = T^{-1} A \phi T - T^{-1} \delta T =: T_{\delta, \phi}[A]. \quad (10)$$

DEFINITION 3.1. A pseudo-linear system $\delta, \phi[A]$ is called *regular* (or (δ, ϕ) -regular) if there exists a gauge transformation $T \in \text{GL}(n, F)$ such that $v(T_{\delta, \phi}[A]) \geq \omega(\delta)$.

The first problem we want to solve is the following: Given a pseudo-linear system $_{\delta,\phi}[A]$, to decide whether $_{\delta,\phi}[A]$ is regular or not and in case $_{\delta,\phi}[A]$ is regular, to compute a gauge transformation T which takes the given system into an equivalent system $T_{\delta,\phi}[A]$ with $v(T_{\delta,\phi}[A]) \geq \omega(\delta)$. This is the subject of the following section.

3.3 Moser Reducibility for Pseudo-Linear Systems

We shall develop an algorithm to determine whether a given pseudo-linear system is regular. More generally this algorithm reduces a given system $_{\delta,\phi}[A]$ to an equivalent irreducible system $_{\delta,\phi}[B]$ with

$$v(B) = \min(\omega(\delta), \max\{v(P_{\delta,\phi}[A]) : P \in \text{GL}(n, F)\}).$$

The method followed is similar to the one used by Moser in the differential case [21].

A Reducibility Criterion

We associate with the system $_{\delta,\phi}[A]$ the following rational numbers:

$$m_{\delta,\phi}(A) = \begin{cases} \omega(\delta) - v(A) + \frac{\text{rank}(A_0)}{n} & \text{if } v(A) < \omega(\delta) \\ 0 & \text{if } v(A) \geq \omega(\delta) \end{cases}$$

and

$$\mu_{\delta,\phi}(A) = \min\{m_{\delta,\phi}(T_{\delta,\phi}[A]) \mid T \in \text{GL}(n, F)\}.$$

DEFINITION 3.2. *The matrix A is called irreducible w.r.t. (δ, ϕ) if $m_{\delta,\phi}(A) = \mu_{\delta,\phi}(A)$, otherwise it is called reducible.*

We remark that the system $_{\delta,\phi}[A]$ is regular if and only if $\mu_{\delta,\phi}(A) = 0$.

The following result is the analogue of Theorem 1 in [21] which gives a reducibility criterion in the differential case.

THEOREM 3.1. *Suppose that $r := -v(A) > -\omega(\delta)$ and let $n_0 = \text{rank}(A_0)$. Then the system $_{\delta,\phi}[A]$ is reducible if and only if the polynomial*

$$\theta(\lambda) := \pi(t^{n_0} \det(t^{r-1}A - \lambda I))$$

vanishes identically in λ .

REMARK 3.1. It is easy to see that the polynomial $\theta(\lambda)$ depends only on A_0, A_1 :

$$\theta(\lambda) = \pi(t^{n_0} \det(t^{-1}A_0 + A_1 - \lambda I))$$

This theorem can be proved in a quite similar way to that used for Theorem 1 in [21]. Due to lack of space we shall not detail the proof here, we just want to point out some useful facts which may help the reader to understand why the approach of Moser can be adapted to our situation

1. The first ingredient used in the proof by Moser (see Lemma 1 of [21]), namely the property that any gauge transformation T can be written in *Smith Normal Form*, remains valid in our situation: Any matrix $T \in \text{GL}(n, F)$ can be written as $T = Pt^\alpha Q$ where $P, Q \in \mathcal{M}_n(\mathcal{O})$ with $\det P, \det Q \in \mathcal{U}$,

$$t^\alpha = \text{diag}(t^{\alpha_1}, \dots, t^{\alpha_n})$$

α_i in $\mathbb{Z}$ with $\alpha_1 \leq \alpha_2 \leq \dots \leq \alpha_n$. This result follows from the fact that $\mathcal{O}$ is a principal ideal domain (the ideals of $\mathcal{O}$ are of the form $t^m \mathcal{O}$).

In the sequel we define the *span* of a transformation $T \in \text{GL}(n, F)$ as the nonnegative integer $\sigma(T) := -v(T) - v(T^{-1}) = \alpha_n - \alpha_1$.

2. The second ingredient used by Moser (see Lemma 2 in [21]) can be stated using our notation as:

LEMMA 3.3. *Let $A \in \mathcal{M}_n(\mathcal{O})$,*

$$A = A_0 + A_1 t + \dots, \text{ with } A_0 \neq 0.$$

A necessary and sufficient condition that there exists a matrix $T \in \text{GL}(n, F)$ such that the matrix

$$B := T^{-1}AT = B_0 + B_1 t + \dots$$

belongs to $\mathcal{M}_n(\mathcal{O})$ and satisfies

$$\text{rank}(B_0) < n_0 := \text{rank}(A_0)$$

is that the polynomial

$$\theta(\lambda) := \pi(t^{n_0} \det(t^{-1}A_0 + A_1 - \lambda I))$$

vanishes identically in λ . Moreover T can be chosen with $\sigma(T) = 1$.

It is important to notice here that in fact T can be chosen of the form $T = Ct^\alpha$ with C a “constant” matrix ($C \in \mathcal{M}_n(R)$ with $\det C \in \mathcal{U}$) and α as above with $\alpha_i \in \{0, 1\}$.

Suppose now that $\phi \neq 1_F$, write $\phi t = qt + O(t^2)$ with $0 \neq q \in R$ and keep the notation above. Then it can be proved that $T^{-1}\phi T = q^\alpha + O(t)$ and hence

$$T^{-1}A\phi T = (T^{-1}AT)T^{-1}\phi T = B_0 q^\alpha + O(t).$$

Now since $\text{rank}(B_0 q^\alpha) = \text{rank}(B_0)$ it follows that the above Lemma 3.3 remains valid if we define B as $B = T^{-1}A\phi T$.

3. Finally, the key for establishing Theorem 3.1 is (as in the proof by Moser) to remark that, using a transformation T as above, the reduction of the rank of A_0 is not affected by the term $T^{-1}\delta T$ in $T_{\delta,\phi}[A]$.

3.4 Super-Reduction for Pseudo-Linear Systems

In this section we extend the concept of the super-irreducible forms to pseudo-linear systems.

Consider again a pseudo-linear system $_{\delta,\phi}[A]$ of the form (8) where the matrix A is given by its t -adic expansion (7) and suppose that $r = -v(A) > -\omega(\delta)$.

Mimicking the differential case we define, for $1 \leq k \leq r + \omega(\delta)$, the rational number $m_{\delta,\phi}^k(A)$ by

$$m_{\delta,\phi}^k(A) = r + \omega(\delta) + \frac{n_0}{n} + \frac{n_1}{n^2} + \dots + \frac{n_{k-1}}{n^k}$$

where $n_i = n_i(A)$ is the number of rows of A with valuation $v(A) + i$.

Now define

$$\mu_{\delta,\phi}^k(A) = \min\{m_{\delta,\phi}^k(T_{\delta,\phi}[A]) \mid T \in \text{GL}(n, F)\}.$$

DEFINITION 3.3. *The matrix A or system (8) is said to be k -irreducible (w.r.t. (δ, ϕ)) if $m_{\delta,\phi}^k(A) = \mu_{\delta,\phi}^k(A)$. Otherwise A is called k -reducible. The matrix A or system (8) is said to be super-irreducible, if it is k -irreducible for every k , or equivalently if $m_{\delta,\phi}^{r+\omega(\delta)}(A) = \mu_{\delta,\phi}^{r+\omega(\delta)}(A)$.*

A criterion for k -reducibility is obtained in exactly the same way as in the differential case. One defines s_k and $\Theta_k(\lambda)$ as

$$s_k = s_k(A) := kn_0 + (k-1)n_1 + \dots + n_{k-1}$$

and

$$\Theta_k(\lambda) := t^{s_k} \det(t^{r-k}A - \lambda I_n)$$

and verifies that $\Theta_k(\lambda)$ belongs to $\mathcal{O}[\lambda]$. Then one can define the polynomial $\theta_k(\lambda) \in R[\lambda]$ as

$$\theta_k(\lambda) = \pi \left(t^{s_k} \det(t^{r-k} A - \lambda I_n) \right). \quad (11)$$

We then have the following

THEOREM 3.2. *The matrix A is k -irreducible, if and only if the polynomials $\theta_j(\lambda)$, ($j = 1, \dots, k$), do not vanish identically in λ .*

4. TWO REDUCTION ALGORITHMS

In this section we give two reduction algorithms for systems of linear functional equations. When specialising these algorithms to the differential case, they correspond to the Moser- and super-reduction algorithms as reviewed in Section 4.1 and Section 4.3. Our approach is to replace the differential equivalence (5) by the pseudo-linear transformation (10) and to show that this results in valid reduction algorithms. The main task for the first algorithm is to prove that Lemma 2.3 in [12] remains valid, for the second algorithm one has to show that the block-reduction algorithm, as introduced in [12, 13], can also be used when using pseudo-linear transformations.

4.1 Differential Reduction Algorithms

We consider a differential system $[A]$ of the form (1) with A given by its x -adic expansion (2). Here we are working with the field $F = K((x))$ equipped with $\phi = 1_F$, $\delta = \vartheta = xd/dx$ and we take $t = x$ as local parameter. We have $\omega(\vartheta) = 0$.

In [12], the first and third author of this paper have introduced a new method that implements the Moser-reduction for linear systems of differential equations. We shall now briefly outline this method, assuming that A_0 is nilpotent and in Jordan canonical form $A_0 = \text{diag}(J, 0_s)$ where J has d Jordan blocks of dimension $n_i \geq 2$ with $n_1 \geq \dots \geq n_d > n_{d+1} = \dots = n_{d+s} = 1$. Define l_i (c_i respectively) for $i = 1, \dots, d+s$, as the position of the i th zero row (column respectively) of A_0 .

The L -matrix $L(A, \lambda) \in k[\lambda]^{(d+s) \times (d+s)}$ is then defined by

$$L(A, \lambda) = ((\alpha_{i,j})) - \text{diag}(0_d, \lambda I_s)$$

where $((\alpha_{i,j})) \in k^{(d+s) \times (d+s)}$ is defined by setting $\alpha_{i,j}$ as the entry of A_1 of row l_i and column c_j . It was then shown in [12] that the system (1) is Moser-reducible if and only if $\det L(A, \lambda) \equiv 0$.

At each reduction step, this algorithm uses the following normalisation: construct a constant transformation C such that $C[A] = \tilde{A}$ has a corresponding L -matrix with the following special structure

$$\begin{pmatrix} L^{11} & L^{12} & 0 \\ L^{21} & L^{22} & 0 \\ L^{31} & L^{32} & L^{33} - \lambda \end{pmatrix} \quad (12)$$

where L^{11} , L^{22} and L^{33} are square matrices of dimension d , $s - q$ and q respectively with the additional condition that

$$\text{rank} \begin{pmatrix} L^{11} \\ L^{21} \end{pmatrix} + s - q = \text{rank} \begin{pmatrix} L^{11} & L^{12} \\ L^{21} & L^{22} \end{pmatrix} \quad (13)$$

and L_{33} is upper-triangular.

The diagonal transformation which carries out the reduction step is then of the form [12]

$$S = \text{diag}(I_{n_1-1}, x, \dots, I_{n_d-1}, x, xI_{s-q}, I_q). \quad (14)$$

4.2 The First Reduction Algorithm

Here again we are working with the field $F = K((x))$ equipped with the x -adic valuation. The residue field of F can be identified with K . We take $t = x$ as local parameter. We fix ϕ and δ and assume that $\phi \neq 1_F$, $\delta \neq 0$. Recall that δ is of the form $\delta = \gamma(1_F - \phi)$ for some nonzero element $\gamma \in F$.

Without any loss of generality we can assume that $\omega(\delta) = 0$ (to achieve this it suffices to replace δ by $\tilde{\delta} := x^{-\omega(\delta)}\delta$).

We write $\phi x = qx + O(x^2)$ for some nonzero element $q \in K$ and we obtain for $k \neq 0$

$$\phi(x^k) = q^k x^k + O(x^{k+1}). \quad (15)$$

We consider a pseudo-linear system $\delta, \phi[A]$ of the form (8) with A given by its x -adic expansion (2) and suppose that $r = -v(A) > 0 = -\omega(\delta)$.

The matrix A defines also a differential system $\vartheta, 1_F[A]$ of the form (1). Our assumption $\omega(\delta) = 0 = \omega(\vartheta)$ implies that $m_{\delta, \phi}(A) = m_{\vartheta, 1_F}(A)$. We shall refer to this quantity simply by $m(A)$.

Notice also that the two polynomials $\theta(\lambda)$ associated with the systems $\delta, \phi[A]$ and $\vartheta, 1_F[A]$ coincide. Therefore we have the following interesting property: *the matrix A is reducible w.r.t. (δ, ϕ) if and only if it is w.r.t. $(\vartheta, 1_F)$.*

Suppose now that A is reducible and let T be a transformation such that $m(T_{\vartheta, 1_F}[A]) < m(A)$. One question arises naturally: is it true that we have also $m(T_{\delta, \phi}[A]) < m(A)$? In general, the answer to this question is no. However, the answer is yes for transformations T of the form $T = CS$ where C is a constant matrix and $S = \text{diag}(x^{d_1}, \dots, x^{d_n})$ such that $\sigma(S) = \max d_i - \min d_i \leq 1$. This is precisely, as we have seen in the previous section, the type of transformation used in each reduction step in the algorithm for the Moser-reduction in the differential case [12].

We shall now prove this result. The constant transformation C which is used for the normalisation of the system can be used similarly as in the differential case since $C_{\phi, \delta}[A] = C^{-1}A\phi C - C^{-1}\delta C = C^{-1}AC = C_{\vartheta, 1_F}[A]$ for any constant transformation C , due to the fact that ϕ is a K -automorphism.

We will show that for the diagonal transformation S we have $m(S_{\phi, \delta}[A]) < m(A)$.

LEMMA 4.1. *The rank-reduction in the differential algorithm is achieved by using the similarity transformation $S^{-1}AS$.*

Proof Using the fact that S is a diagonal transformation $S = \text{diag}(x^{d_1}, \dots, x^{d_n}) =: x^D$, where $D = \text{diag}(d_1, \dots, d_n)$ with $d_i \in \mathbb{N}$, we compute $S^{-1} = x^{-D}$ and $\vartheta S = Dx^D$ hence $B = S_{\vartheta, 1_F}[A] = S^{-1}AS - D$. We can see that the matrix D does not affect the leading matrix B_0 of the transformed system since $r > 0$. Hence the reduction in rank will have to be achieved by $S^{-1}AS$ alone. $\square$

LEMMA 4.2. Let S and D be as in Lemma 4.1 and its proof, with $d_i \in \{0, 1\}$ ($i = 1 \dots n$). Then $\phi S = Sq^D + O(x^2)$.

Proof Using (15), we obtain

$$\phi(x^{d_i}) = \begin{cases} 1 & \text{if } d_i = 0 \\ q^{d_i} x^{d_i} + O(x^{d_i+1}) & \text{otherwise} \end{cases} \quad (16)$$

and, using $d_i \in \{0, 1\}$ we compute

$$\begin{aligned} \phi S &= \text{diag}(q^{d_1} x^{d_1}, \dots, q^{d_n} x^{d_n}) + O(x^2) \\ &= Sq^D + O(x^2). \end{aligned}$$

□

LEMMA 4.3. The operation $S^{-1}A\phi S$ with S as in the previous lemma achieves a rank-reduction.

Proof Using Lemma 4.2, we find

$$\begin{aligned} S^{-1}A\phi S &= S^{-1}A \left((Sq^D + O(x^2)) \right) \\ &= S^{-1}ASq^D + S^{-1}A \cdot O(x^2). \end{aligned} \quad (17)$$

Using Lemma 4.1, we know that $S^{-1}AS$ has a leading matrix with reduced rank, hence this is also true for

$$S^{-1}ASq^D =: B_0 x^{-r} + O(x^{-r+1}).$$

But it is clear that the second term in the last sum in (17) is $S^{-1}A \cdot O(x^2) = O(x^{-r+1})$ since $v(S^{-1}) = -1$, implying that this term does not affect B_0 . □

PROPOSITION 4.1. Suppose that A is Moser-reducible and the diagonal transformation $S = \text{diag}(x^{d_1}, \dots, x^{d_n})$ with $d_i \in \{0, 1\}$ satisfies $m(S_{\vartheta, 1F}[A]) < m(A)$. Then we also have $m(S_{\phi, \delta}[A]) < m(A)$.

Proof We have

$$\begin{aligned} S_{\phi, \delta}[A] &= S^{-1}A\phi(S) - S^{-1}\delta(S) \\ &= S^{-1}A\phi(S) - \tilde{C}x^{\omega(\delta)} + O(x^{1+\omega(\delta)}) \end{aligned}$$

where $\tilde{C}$ is a constant diagonal matrix. Using Lemma 4.3, we have $S^{-1}A\phi S = B = x^{-r}(B_0 + O(x))$, with $\text{rank } B_0 < \text{rank } A_0$. But we have assumed $\omega(\delta) = 0 > -r$ hence, the proposition follows. □

The first reduction algorithm is then

Pseudo.Linear.Reduction.1(A, ϕ, δ)

Input:

- $A \in \mathcal{M}_n(F)$,
- ϕ a K -automorphism,
- δ a Pseudo-derivative w.r.t. ϕ

Output:

- Transformation T such that $T_{\phi, \delta}[A]$ is Moser (δ, ϕ) -irreducible
- 1. $T := I_n$;
- 2. **while** $(r(A) > -\omega(\delta))$ **and** $(\theta(\lambda) \equiv 0)$ **do**
 - (a) Compute a constant transformation C to normalise A ;
 - (b) $A := C_{\phi, \delta}[A]$; $T := TC$;

(c) Compute a diagonal transformation S as in the differential case;

(d) $A := S_{\phi, \delta}[A]$; $T := TS$;

3. **return** T ;

4.3 The Second Reduction Algorithm

In [12] it was shown that the computation of a super-irreducible system can be reduced to the computation of several Moser-irreducible systems of smaller size, using a block-reduction algorithm.

In this section, we assume that the first reduction algorithm has been applied to the system (8). We will show that the *Direct Block Reduction Algorithm* as introduced in [13] can be used for systems of linear functional equations in order to obtain a second reduction algorithm which can be characterised as computing a new system of the form $\delta Y = B\phi Y$ where B is a block-triangular super-reduced matrix where each diagonal block is Moser-reduced.

As explained in [13], Section 3, the block-reduction is achieved by using *elementary gauge transformation* of the form

$$E_{i,j}(\alpha) = I_n + \alpha M_{i,j}$$

where $\alpha \in K[[x]]$ and where $M_{i,j}$ denotes the $n \times n$ matrix with all entries equal to 0, except the entry at position (i, j) which is equal to 1.

We recall that transforming a given differential system (1) with $E_{i,j}(\alpha)$ results in a new system whose coefficient matrix $\tilde{A}$ is obtained from A by adding to the j th column the i th column multiplied by α , then subtracting the j th row multiplied by α from the i th row, and adding $\vartheta(\alpha)$ to the entry in the (i, j) position.

The effect of using $E_{i,j}(\alpha)$ as a pseudo-linear transformation is very similar to the differential case, with the difference that the j th column of the transformed systems results from adding to the j th column the i th column multiplied by $\phi(\alpha)$ and also adding $\delta(\alpha)$ to the entry in the (i, j) position. The concept of *normalised Moser-irreducible forms* as introduced in [13] can be easily extended to the case of systems of linear functional equations as the normalisation is carried out using a constant transformation.

PROPOSITION 4.2. The direct Block-Reduction algorithm of [13] can be adapted for our use by replacing the elementary transformation $E_{ij}(\alpha)$ with $E_{ij}(q^{-h}\alpha)$ where $h = v(\alpha)$.

Proof By reviewing the process of eliminating terms in A it becomes apparent that the elimination is achieved using linear combinations of leading coefficients of elements in A , multiplied by the leading coefficients of α . Let $\alpha = ct^h + \dots$, it follows $\phi(\alpha) = q^h ct^h + \dots$, hence the leading coefficient of $\phi(q^{-h}\alpha)$ equals c . Hence using $E_{ij}(q^{-h}\alpha)$ for a pseudo-linear transformation carries out an identical elimination process on the corresponding leading terms. □

Remark 4.2. This does not imply that the final result of the block-reduction is identical in both of the differential and general pseudo-linear case. Crucially, each isolated step of the algorithm does indeed an identical elimination in both cases, but the transformations introduces also additional terms of higher order that are different.

5. APPLICATIONS FOR Q-DIFFERENCES

Let $q \in K$ with $q \neq 1$ and consider a linear q -difference system with coefficients in $K(x)$:

$$Y(qx) = M(x)Y(x), \quad M(x) \in \mathcal{M}_n(K(x)). \quad (18)$$

5.1 Polynomial Solutions

We are interested in this section by the problem of computing all the polynomial solutions of a system of the form (18). Algorithms for solving this problem in the differential and the difference cases have been proposed in [7, 2]. Here we shall show that the same approach remains valid for the q -difference case. Note that a different approach for finding polynomial solutions of systems of this type is given in [1]. For sake of brevity we shall consider here only the problem of computing a bound on the degree of polynomial solutions. Such a bound can be obtained from the so-called *indicial equation* (at $x = \infty$). Unfortunately the indicial equation is not immediately apparent for a given general system. The idea consists in reducing the given system to a *simple system* from which the *indicial equation* can be immediately obtained. We will show that such a simple form can be derived from a super-irreducible form in exactly the same way as in the difference and the differential cases [7, 2].

5.2 The indicial equation

A polynomial solution $Y \in K[x]^n$ of degree ν can be viewed as a local formal solution (at $x = \infty$) of the form

$$Y(x) = \sum_{i \geq 0} x^{-i+\nu} Y_i, \quad (19)$$

where $Y_i \in K^n$, $Y_0 \neq 0$ and $Y_i = 0$ for $i > \nu$.

The idea is to work with $F = K((x^{-1}))$, the completion of $K(x)$ w.r.t. to the t -adic valuation (here $t = x^{-1}$). Define ϕ and δ by $\phi(t) = qt$ and $\delta = 1_F - \phi$. Our system (18) can then be written as the pseudo-linear system:

$$\delta Y = A(t)\phi Y \quad (20)$$

where $A(t) = M(q^{-1}t^{-1}) - I_n \in \mathcal{M}_n(F)$.

Multiplying this system on the left by the diagonal matrix

$$D := \text{diag}(t^{\alpha_1}, \dots, t^{\alpha_n})$$

where $\alpha_i = -\min(v(A_{i,\cdot}), 0)$, $A_{i,\cdot}$ being the i th row of the matrix A , yields the equation

$$D\delta Y = C\phi Y, \quad (21)$$

where $C = DA$. By definition, one has $D, C \in \mathcal{M}_n(K[[t]])$. Put $C = \sum_{i \geq 0} C_i t^i$, $D = \sum_{i \geq 0} D_i t^i$.

We look for formal solutions of the form: $Y = \sum_{i=0}^{+\infty} t^{i+\nu} Y_i$ where $\nu \in K$, $Y_i \in K^n$ with $Y_0 \neq 0$. One has

$\phi Y = \sum_{i \geq 0} q^{i+\nu} t^{i+\nu} Y_i$ and $\delta Y = \sum_{i \geq 0} (1 - q^{i+\nu}) t^{i+\nu} Y_i$. Replacing D , C , ϕY and δY by their t -adic expansions in (21) and identifying coefficients of the same powers of t (after simplifying the factor t^ν) yields, in particular, the equation

$$(1 - q^\nu) D_0 Y_0 = q^\nu C_0 Y_0.$$

Thus in order that the system (21) admits a formal solution of the form (19), ν and Y_0 must satisfy the equation

$$(C_0 - (q^{-\nu} - 1)D_0) Y_0 = 0$$

which implies that $(q^{-\nu} - 1)$ must be a root of the polynomial

$$E(\lambda) := \det(C_0 - \lambda D_0).$$

As a consequence if $Y \in K[x]^n$ is a nonzero polynomial solution of (18) of degree ν then $E(q^{-\nu} - 1) = 0$. It follows that the degree of polynomial solution can be bounded by the biggest nonnegative integer ν such that $q^{-\nu} - 1$ is a root of $E(\lambda)$.

But, it may happen that the determinant $E(\lambda)$ vanishes identically in λ in which case it is quite useless to us. This motivates the following definition

DEFINITION 5.1. *The system (21) is said to be simple if $E(\lambda) := \det(C_0 - \lambda D_0) \neq 0$ (as a polynomial in λ). In this case the polynomial $E(\lambda)$ is called the indicial polynomial of (21).*

PROPOSITION 5.1. *Every q -difference system (20) can be reduced to an equivalent system (21) which is simple.*

Proof Since every q -difference system (20) is equivalent to a super-irreducible one, it suffices to prove that every super-irreducible system is simple.

Let $r = -v(A)$. If $r \leq 0$ then the system is simple. Suppose that $r > 0$ (notice that $\omega(\delta) = 0$) and let D and $C = DA$ be defined as above. In one hand, one has $\det(D) = t^{s_r}$ (see Section 3.4 for the definition of s_r) and hence $t^{s_r} \det(A - \lambda I_n) = \det(D) \det(A - \lambda I_n) = \det(DA - \lambda D)$. In the other hand

$\theta_r(\lambda) = t^{s_r} \det(A(t) - \lambda I_n)|_{t=0} = \det(C(t) - \lambda D(t))|_{t=0} = \det(C_0 - \lambda D_0)$. Now if (20) is super-irreducible then, by Theorem 3.2, the polynomial $\theta_r(\lambda)$ is not identically zero and (20) is simple. $\square$

6. EXAMPLES

Our reduction algorithms are implemented in the Maple package ISOLDE [11] and we shall now present two examples using this package. The function `super_reduce` takes as input a matrix A and as additional parameters the automorphism ϕ , pseudo-derivative δ and the quantity $\omega(\delta)$. The command `?super_reduce` displays additional user information.

As a first example, consider the difference system $\delta(Y) = A(x)\phi(Y)$ where $F = \mathbb{C}((x^{-1}))$, δ and ϕ are as in Example 2.2, third case, with $\omega(\delta) = 1$ and the matrix $A(x)$ is given by

$$A(x) = \begin{pmatrix} 5x^{-1} & x^2 & 0 \\ 0 & 0 & x^{-3} \\ x^{-3} & 0 & 3x^{-1} + 5x^{-2} \end{pmatrix}.$$

The valuation of A is $v(A) = -2 < \omega(\delta) = 1$ and hence, we calculate the number $m_{\delta, \phi}(A) = \omega(\delta) - v(A) + \frac{\text{rank } A_0}{n} = \frac{10}{3}$. We assign ϕ and δ in Maple:

```
> phi:= proc(a,x) return subs(x=x-1,a) end;
> delta:=proc(a,x) return a-phi(a,x) end;
```

Calling the super-reduction on this example gives a list of results, the first entry being the transformed system that is super-irreducible:

```
> tmp := super_reduce(A, x, 1/x, lambda,
```

phi, delta, 1, S, invS): tmp[1];

$$\begin{pmatrix} \frac{6x^2-9x+4}{(x-1)^3} & 0 & (x-1)^{-1} \\ \frac{x^2}{(x-1)^3} & 5x^{-1} & 0 \\ -3\frac{2x^3-6x^2+10x-5}{x(x-1)^3} & x^{-2} & \frac{5+x}{x(x-1)} \end{pmatrix}$$

This resulting system has $v(A) = 1 = \omega(\delta)$ hence, it is a regular system. As part of the third entry of the results, one obtains in this example the *indicial equation* of the system: `>tmp[3];`

$$[[0, \lambda^3 - 12\lambda^2 + 47\lambda - 60]]$$

As a second example, consider the q -difference system $\delta(Y) = A(x)\phi(Y)$ where, ϕ and δ are as in Example 2.2 fourth case, with $\omega(\delta) = 0$ and the matrix $A(x)$ is given by

$$A(x) = \begin{pmatrix} q^3x^4 & -x \\ x^3 & -1 \end{pmatrix}.$$

We keep $\delta = 1 - \phi$ and define ϕ by the following:

`> phi:= proc(a,x) return subs(x=q*x,a) end;`

This system is Moser-irreducible but not super-irreducible. We call the super-reduction on this example:

`> tmp := super_reduce(A, x, 1/x, lambda,`

`phi, delta, 1, S, invS): tmp[1];`

$$\begin{pmatrix} q^3x^4 - 1 & 0 \\ x^3 & -\frac{-1+q^3}{q^3} \end{pmatrix}$$

Additional output returns a list of polynomials:

`> tmp[3];`

$$[[0, 1 - q^3 - q^3\lambda], [1, 1], [2, 1], [3, 1], [4, \lambda - q^3]].$$

This system admits a polynomial solution $Y = \begin{pmatrix} 1 \\ x^3 \end{pmatrix}$ of

degree $\nu = 3$ and one indeed verifies that $E(q^{-3} - 1) = 0$ where $E(\lambda)$ is given by the first polynomial in the above output.

Acknowledgements

We would like to thank the anonymous referees for their comments, that helped to improve the final version of this paper.

7. REFERENCES

- [1] S. Abramov. Rational solutions of linear difference and q -difference equations with polynomial coefficients. In *Proceedings of ISSAC'95*, pages 303–308. ACM Press, 1995.
- [2] S. Abramov and M. A. Barkatou. Rational solutions of first order linear difference systems. In *Proceedings of ISSAC '98*, pages 124–131, Rostock, Germany, 1998. ACM Press.
- [3] M. A. Barkatou. *Contribution à l'étude des équations différentielles et de différences dans le champ complexe*. PhD thesis, INPG, 1989.
- [4] M. A. Barkatou. On the reduction of linear systems of difference equations. In *Proceedings of ISSAC '89*, pages 1–6, Portland, Oregon, 1989. ACM Press.
- [5] M. A. Barkatou. A rational version of Moser's Algorithm. In *Proceedings of ISSAC '95*, pages 297–302, Montreal, Canada, 1995. ACM Press.
- [6] M. A. Barkatou. An algorithm to compute the exponential part of a formal fundamental matrix solution of a linear differential system. *Journal of App. Alg. in Eng. Comm. and Comp.*, 8(1):1–23, 1997.
- [7] M. A. Barkatou. On rational solutions of systems of linear differential equations. *Journal of Symbolic Computation*, 28:547–567, 1999.
- [8] M. A. Barkatou. On super-irreducible forms of linear differential systems with rational function coefficients. *Journal of Computational and Applied Mathematics*, 162(1):1–15, 2004.
- [9] M. A. Barkatou. Factoring systems of linear functional equations using eigenrings. In I. S. Kotsireas and E. V. Zima, editors, *Latest Advances in Symbolic Algorithms*, pages 22–42. World Scientific, 2007.
- [10] M. A. Barkatou and E. Pflügel. An algorithm computing the regular formal solutions of a system of linear differential equations. *Journal of Symbolic Computation*, 28:569–588, 1999.
- [11] M. A. Barkatou and E. Pflügel. The ISOLDE package. A SourceForge Open Source project, <http://isolde.sourceforge.net>, 2006.
- [12] M. A. Barkatou and E. Pflügel. Computing super-irreducible forms of systems of linear differential equations via moser-reduction: a new approach. In *Proceedings of ISSAC '07*, pages 1–8, Waterloo, Canada, 2007. ACM Press.
- [13] M. A. Barkatou and E. Pflügel. On the Moser- and super-reduction algorithms of systems of linear differential equations and their complexity. *Submitted to JSC*, 2007.
- [14] M. Bronstein and M. Petkovsek. An Introduction to Pseudo-Linear Algebra. *Theoretical Computer Science*, 157(1):3–33, 1996.
- [15] V. Dietrich. Zur Reduktion von linearen Differentialgleichungssystemen. *Math. Ann.*, 237:79–95, 1978.
- [16] A. Hilali and A. Wazner. Formes super-irréductibles des systèmes différentiels linéaires. *Numer. Math.*, 50:429–449, 1987.
- [17] N. Jacobson. Pseudo-linear transformations. *Annals of Mathematics*, 33(2):484–507, 1937.
- [18] C.-P. Jeannerod and E. Pflügel. A reduction algorithm for matrices depending on a parameter. In *Proceedings of ISSAC '99*, pages 121–128, Vancouver, Canada, 1999. ACM Press.
- [19] S. Lang. *Algebra*. New York; London: Springer, c2002, 2002.
- [20] A. Levelt. Stabilizing differential operators: a method for computing invariants at irregular singularities. In M. Singer, editor, *Differential Equations and Computer Algebra*, pages 181–228. Academic Press, 1991.
- [21] J. Moser. The order of a singularity in Fuchs' theory. *Math. Z.*, pages 379–398, 1960.
- [22] E. Pflügel. Effective formal reduction of linear differential systems. *Appl. Alg. Eng. Comm. Comp.*, 10(2):153–187, 2000.

Products of Ordinary Differential Operators by Evaluation and Interpolation

Alin Bostan
Algorithms Project-Team,
INRIA Paris-Rocquencourt
78153 Le Chesnay (France)
Alin.Bostan@inria.fr

Frédéric Chyzak
Algorithms Project-Team,
INRIA Paris-Rocquencourt
78153 Le Chesnay (France)
Frederic.Chyzak@inria.fr

Nicolas Le Roux
Algorithms Project-Team,
INRIA Paris-Rocquencourt
78153 Le Chesnay (France)
Nicolas.Le_Roux@inria.fr

ABSTRACT

It is known that multiplication of linear differential operators over ground fields of characteristic zero can be reduced to a constant number of matrix products. We give a new algorithm by evaluation and interpolation which is faster than the previously-known one by a constant factor, and prove that in characteristic zero, multiplication of differential operators and of matrices are computationally equivalent problems. In positive characteristic, we show that differential operators can be multiplied in nearly optimal time. Theoretical results are validated by intensive experiments.

Categories and Subject Descriptors:

I.1.2 [Computing Methodologies]: Symbolic and Algebraic Manipulation – *Algebraic Algorithms*

General Terms: Algorithms, Theory

Keywords: Fast algorithms, differential operators.

1. INTRODUCTION

Multiplication in polynomial algebras $\mathbb{K}[X]$ and $\mathbb{K}[X, Y]$ over a field $\mathbb{K}$ has been intensively studied in the computer-algebra literature. Since the discovery of Karatsuba’s algorithm and the Fast Fourier Transform, hundreds of articles have been dedicated to theoretical and practical issues; see, e.g., [9, Ch. 8], [1], and the references therein. Not only are many other operations built upon multiplication, but often their complexity can be expressed in terms of the complexity of multiplication—whether as a constant number of multiplications or a logarithmic number of multiplications. In $\mathbb{K}[X]$, this is the case for Euclidean division, gcd and resultant computation, multipoint evaluation and interpolation, shifts, certain changes of bases, etc.

In the noncommutative setting of linear ordinary differential operators, the study is by far less advanced. The complexity of the product has been addressed only recently, by van der Hoeven in the short paper [11]: multiplication of operators over ground fields $\mathbb{K}$ of characteristic zero can be reduced by an evaluation-interpolation scheme to a constant

number C of matrix multiplications with elements in $\mathbb{K}$. Work in progress [3] suggests that linear algebra is again the bottleneck for computations of GCRDs and LCLMs.

This work aims at deepening the study started in [11] for characteristic 0. We improve van der Hoeven’s result along several directions: We make the constant factor C explicit in §3.2 and improve it in §4, and we prove in §3 that multiplication of matrices and of differential operators are equivalent computational problems—that is, they share the same exponent, thus answering the question left open in [11, §6, Remark 2]. As usual, those results hold for sufficiently large characteristic as well. We prolong the study to the case of (small) positive characteristic, by giving in §5 an algorithm for computing the product of two differential operators in softly quadratic complexity, that is, nearly optimally in the output size. This indicates that the equivalence result may fail to generalize to arbitrary fields.

In what follows, the field $\mathbb{K}$ has characteristic zero, unless stated otherwise. $\mathbb{K}[X]\langle\partial\rangle$ and $\mathbb{K}[X]\langle\theta\rangle$ respectively denote the associative algebras $\mathbb{K}\langle X, \partial; \partial X = X\partial + 1 \rangle$ and $\mathbb{K}\langle X, \theta; \theta X = X(\theta + 1) \rangle$.

	vdH _θ	lvdH _θ	vdH	lvdH	MulWeyl
Product by blocks	37	24	96	48	12
Zeros + Strassen	20	8	47	12	8

Table 1: Number of $n \times n$ matrix products for multiplication in $\mathbb{K}[X]\langle\theta\rangle$, resp. $\mathbb{K}[X]\langle\partial\rangle$, in bidegree (n, n) .

Table 1 encapsulates our improvements on the constant C . It displays the cost of linear algebra in van der Hoeven’s algorithms (vdH_θ, resp. vdH) and in the improved versions (lvdH_θ, resp. lvdH), which are described in §3.1, resp. §4.1, and in our algorithm (MulWeyl) in §4.2. The subscript θ refers to multiplication in $\mathbb{K}[X]\langle\theta\rangle$; its absence means a product in $\mathbb{K}[X]\langle\partial\rangle$. The first row provides bounds on the number of $n \times n$ matrix products used in each algorithm for multiplying operators in $\mathbb{K}[X]\langle\partial\rangle$, resp. $\mathbb{K}[X]\langle\theta\rangle$, of degree at most n in X and in ∂ , resp. θ , under the naive complexity estimate (1) below. This estimate reflects the choice of multiplying rectangular matrices by decomposing them into square blocks. The second row gives tighter bounds under the assumptions that: (i) any product by a zero block is discarded; (ii) when possible, a product of two 2×2 matrices of $n \times n$ blocks is computed as 7 block products, instead of 8, by using Strassen’s algorithm [14]; (iii) predicted non-trivial zero blocks in the output are not computed.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC’08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Canonical form and bidegree. In the algebra $\mathbb{K}[X]\langle\partial\rangle$, resp. $\mathbb{K}[X]\langle\theta\rangle$, the commutation rule allows one to rewrite any given element into a so-called *canonical form* with X on the left of monomials and ∂ , resp. θ , on the right, that is, as a linear combination of monomials $X^i\partial^j$, resp. $X^i\theta^j$, for uniquely-defined coefficients from $\mathbb{K}$. In either case, we speak of an element of bidegree (d, r) , resp. at most (d, r) , when the degree of its canonical form in X is d , resp. at most d , and that in ∂ , resp. θ , is r , resp. at most r . With natural notation, the bidegree (d_C, r_C) of a product $C = BA$ clearly satisfies $r_C = r_A + r_B$ and $d_C \leq d_A + d_B$.

The problem of computing the canonical form of the product of two elements of bidegree (d, r) from $\mathbb{K}[X]\langle\partial\rangle$, resp. from $\mathbb{K}[X]\langle\theta\rangle$, given in canonical form, is denoted $\langle d, r \rangle_\partial$, resp. $\langle d, r \rangle_\theta$.

Complexity measures. All complexity estimates are given in terms of arithmetical operations in $\mathbb{K}$, which we denote “ops.” We denote by $C_\theta, C_\partial : \mathbb{N} \rightarrow \mathbb{N}$ two functions such that Problems $\langle n, n \rangle_\partial$ and $\langle n, n \rangle_\theta$ can be solved in $C_\partial(n)$ and $C_\theta(n)$, respectively. We denote by $M : \mathbb{N} \rightarrow \mathbb{N}$ a function such that polynomials of degree at most n in $\mathbb{K}[X]$ can be multiplied in $M(n)$ ops. Using Fast Fourier Transform algorithms, $M(n)$ can be taken in $\mathcal{O}(n \log n)$ over fields with suitable roots of unity, and $\mathcal{O}(n \log n \log \log n)$ in the general case [13, 5]. We use the notation $f \in \tilde{\mathcal{O}}(g)$ for $f, g : \mathbb{N} \rightarrow \mathbb{N}$ if f is in $\mathcal{O}(g \log^m g)$ for some $m \geq 1$. For instance, $M(n)$ is in $\tilde{\mathcal{O}}(n)$. The problem of multiplying an $m \times n$ matrix by an $n \times p$ matrix is written $\langle m, n, p \rangle$. We let $MM : \mathbb{N}^3 \rightarrow \mathbb{N}$ be a function such that Problem $\langle m, n, p \rangle$ can be solved in $MM(m, n, p)$ ops. We use the abbreviation $MM(n)$ for $MM(n, n, n)$. The current tightest (strict) upper bound 2.376 for ω such that $MM(n) \in \mathcal{O}(n^\omega)$ is derived in [7]. For the time being, this estimate is only of theoretical relevance. Few practical algorithms with complexity better than cubic are currently known for matrix multiplication, among which Strassen’s algorithm [14] with exponent $\log_2 7 \approx 2.807$ and the Pan–Kaporin algorithm [12] with exponent 2.776. For rectangular matrix multiplication, we shall use the estimate

$$MM(an, bn, cn) \leq abc MM(n), \quad \text{for } a, b, c \in \mathbb{N}, \quad (1)$$

obtained by performing the naive product of $a \times b$ by $b \times c$ matrices whose coefficients are $n \times n$ blocks.

Furthermore, we assume that $M(n)$, $MM(n)$, $C_\partial(n)$, and $C_\theta(n)$ satisfy the usual super-linearity assumption of [9, §8.3, Eq. (9)] and also that, if $F(n)$ is any of these functions, then $F(cn)$ belongs to $\mathcal{O}(F(n))$, for all positive constants c .

Useful complexity results. Throughout, we shall freely use several classical results on the complexity of basic polynomial operations. They are encapsulated in Lemma 1. The corresponding algorithms are found in: [8, Algorithm E] for (a); [9, Chapter 10] for (b); [10, Th. 2.4 and 2.5] for (c); and [9, Cor. 8.29] for (d).

Lemma 1 *Let $\mathbb{K}$ be an arbitrary field. Let $a \in \mathbb{K}$, let $P(X) \in \mathbb{K}[X]$ be of degree less than n and $f, g \in \mathbb{K}[X, Y]$ of degree at most d in X and n in Y . One can perform: (a) the Taylor shift $Q(X) := P(X + a)$; (b) the multipoint evaluation and interpolation of P on $a, a+1, \dots, a+n$ if the characteristic of $\mathbb{K}$ is 0 or greater than n ; (c) the base change*

between the monomial and the falling factorial basis $(X)_k = X(X-1)\cdots(X-k+1)$ in $\mathcal{O}(M(n) \log n)$ ops. Moreover, one computes: (d) the product $h = fg$ in $\mathcal{O}(M(dn))$ ops.

2. NAIVE ALGORITHMS

In this section, we provide complexity estimates for several known algorithms for $\langle d, r \rangle_\partial$. We set

$$A = \sum_{i=0}^r \sum_{j=0}^d a_{i,j} X^j \partial^i, \quad B = \sum_{i=0}^r \sum_{j=0}^d b_{i,j} X^j \partial^i = \sum_{i=0}^r b_i(X) \partial^i.$$

For any $L = \sum_{i=0}^r l_i(X) \partial^i = \sum_{j=0}^d X^j l'_j(\partial)$, we define

$$\frac{dL}{dX} = \sum_{i=0}^r \frac{dl_i(X)}{dX} \partial^i, \quad \frac{dL}{d\partial} = \sum_{j=0}^d X^j \frac{dl'_j(\partial)}{d\partial}.$$

Naive expansion. The most naive calculation of BA is by expanding each $\partial^i X^l$ in the equality

$$BA = \sum_{i=0}^r \sum_{j=0}^d \sum_{k=0}^r \sum_{l=0}^d b_{i,j} a_{k,l} X^j \left(\partial^i X^l \right) \partial^k.$$

Using Leibniz’s formula $\partial^i X^l = \sum_{k=0}^{\min(i,l)} (l)_k \binom{i}{k} X^{l-k} \partial^{i-k}$ and the recurrences $(l)_{k+1} = (l)_k (l-k)$ and $\binom{i}{k+1} = \binom{i}{k} \frac{i-k}{k+1}$, the canonical form of $\partial^i X^l$ is computed in $\mathcal{O}(\min(i, l))$ ops. This induces a complexity $\mathcal{O}(d^2 r^2 \min(d, r))$ for computing BA . The estimate simplifies to $\mathcal{O}(n^5)$ if $d = r = n$.

Iterative schemes. Another calculation is by the formula

$$BA = \sum_{i=0}^r b_i(X) \left(\partial^i A \right) \quad (2)$$

and the observation that $\partial^i A$ has bidegree at most $(d, r+i)$ and is computed from $\partial^{i-1} A$ in $\mathcal{O}(dr)$ ops. by the identity

$$\partial T = T \partial + \frac{dT}{dX} \quad \text{for } T = T(X, \partial). \quad (3)$$

Therefore, the overall complexity is $\mathcal{O}(M(d)r^2 + dr^2) = \mathcal{O}(M(d)r^2)$. When $d = r = n$, this is $\mathcal{O}(M(n)n^2)$, and $\tilde{\mathcal{O}}(n^3)$ if FFT is used. Similar considerations based on

$$TX = XT + \frac{dT}{d\partial} \quad \text{for } T = T(X, \partial) \quad (4)$$

provide an algorithm in $\mathcal{O}(d^2 M(r))$, and one can always use the better algorithm by first comparing d and r .

Another formula, attributed to Takayama and used in several implementations (Takayama’s Kan system [15]; Maple’s *Ore_algebra* by Chyzak [6]), is given by the (finite) sum

$$BA = \sum_{k \geq 0} \frac{1}{k!} \left(\frac{d^k B}{d\partial^k} * \frac{d^k A}{dX^k} \right), \quad (5)$$

where the products $*$ are computed formally as commutative products between canonical forms, the resulting sum being viewed as a canonical form. Each of the derivatives has bidegree at most (d, r) and the derivative at order k can be computed in $\mathcal{O}(dr)$ ops. from the one at order $k-1$. The complexity is seen to be $\mathcal{O}(\min(d, r) M(dr))$ ops., by Lemma 1(d). When $d = r = n$, this is $\mathcal{O}(n M(n^2))$, or $\tilde{\mathcal{O}}(n^3)$ using FFT; the scheme (2) is just a bit better than (5).

3. EQUIVALENCE BETWEEN PRODUCTS OF MATRICES AND OPERATORS

Let $\mathbb{K}$ be a field of characteristic zero. In [11], van der Hoeven showed that $C_\theta(n)$ and $C_\theta(n)$ are in $\mathcal{O}(\text{MM}(n))$. When $\omega < 3$, this improves upon the algorithms in §2.

In this section, we explain and improve this result along two directions: we make the constant factor explicit in the estimate $C_\theta(n) \in \mathcal{O}(\text{MM}(n))$, and lessen it. Then, we prove that $\langle n, n, n \rangle$, $\langle n, n \rangle_\theta$, and $\langle n, n \rangle_\theta$ are equivalent computational problems, in a sense made clear below.

3.1 Product in $\mathbb{K}[X]\langle\theta\rangle$ reduces to matrix product: van der Hoeven’s algorithm revisited

A differential operator A in $\mathbb{K}[X]\langle\theta\rangle$ can be viewed as a $\mathbb{K}$ -endomorphism of $\mathbb{K}[X]$, mapping a polynomial f to $A(f)$. As such, it is represented, with respect to the canonical basis $(X^i)_{i \geq 0}$ of $\mathbb{K}[X]$, by an (infinite) matrix denoted M_∞^A . The submatrix of M_∞^A consisting of its first $r \geq 1$ rows and $c \geq 1$ columns is denoted $M_{r,c}^A$.

Van der Hoeven’s key observation is that an operator A of bidegree (d, r) is completely determined by the matrix $M^A := M_{d+r+1, r+1}^A$. Writing $A = \sum_{i=0}^d \sum_{j=0}^r a_{i,j} X^i \theta^j$ and using the relation $\theta^j(X^k) = k^j X^k$ yields

$$A(X^k) = \sum_{i,j} a_{i,j} k^j X^{i+k} = X^k \sum_{i=0}^d \tilde{A}_i(k) X^i,$$

where the polynomials $\tilde{A}_i$ are defined as $\tilde{A}_i(X) = \sum_{j=0}^r a_{i,j} X^j$ for all $0 \leq i \leq d$. Thus the matrix M^A has the following rectangular banded form:

$$M^A = \begin{bmatrix} \tilde{A}_0(0) & & & & \\ \tilde{A}_1(0) & \tilde{A}_0(1) & & & \\ & \vdots & \tilde{A}_1(1) & \ddots & \\ \tilde{A}_d(0) & \vdots & \ddots & \tilde{A}_0(r) & \\ & \tilde{A}_d(1) & & \tilde{A}_1(r) & \\ & & \ddots & \vdots & \\ & & & \tilde{A}_d(r) & \end{bmatrix}. \quad (6)$$

The knowledge of A is equivalent to that of all $d+1$ polynomials $\tilde{A}_i$. Each of the latter having degrees bounded by r , this is also equivalent to the data of the values $\tilde{A}_i(k)$, for $0 \leq k \leq r$ and $0 \leq i \leq d$. This is true by Lagrange interpolation. Thus, A is indeed completely determined by the $r+1$ polynomials $A(X^k)$, and also by the matrix M^A .

Now, let $A, B \in \mathbb{K}[X]\langle\theta\rangle$ and let C be BA . Then $M_\infty^C = M_\infty^B M_\infty^A$. If A, B , and C have bidegrees $(d_A, r_A), (d_B, r_B)$, and (d_C, r_C) , then the previous discussion implies the following “finite version” of this matrix equality:

$$M^C = M_{d_C+r_C+1, d_A+r_C+1}^B M_{d_A+r_C+1, r_C+1}^A, \quad (7)$$

which is the basis of the algorithm in [11], described below.

Putting all these considerations together leads to Algorithm Mul_θ in Fig. 1 and proves the following proposition.

Proposition 1 *Algorithm Mul_θ in Fig. 1 reduces the computation of the product $C = BA$ to the following tasks:*

(T1) $d_A + 1$ evaluations in degrees $\leq r_A$ on $0, 1, \dots, r_C$;

$\text{Mul}_\theta(B, A)$

Input: $A, B \in \mathbb{K}[X]\langle\theta\rangle$.
Output: their product $C = BA$.

1. Compute the $\tilde{A}_i$ ’s and $\tilde{B}_i$ ’s from A and B , then the matrices $M_{d_C+r_C+1, d_A+r_C+1}^B$ and $M_{d_A+r_C+1, r_C+1}^A$.
2. Compute M^C by Eq. (7).
3. Compute the $\tilde{C}_i$ ’s from M^C , then recover C .

Figure 1: Product of differential operators in θ .

(T2) $d_B + 1$ evaluations in degrees $\leq r_B$ on $0, 1, \dots, d_A + r_C$;

(T3) $d_C + 1$ interpolations in degrees $\leq r_C$ on $0, 1, \dots, r_C$;

(T4) an instance of $\langle d_C + r_C + 1, d_A + r_C + 1, r_C + 1 \rangle$.

PROOF. Eq. (6) shows that Step 1 in Algorithm Mul_θ is performed by the evaluation Tasks (T1–T2). Similarly, the interpolation Task (T3) performs Step 3. Finally, the product in Step 2 is computed by (T4). $\square$

We stress that the evaluation-interpolation scheme used in Algorithm Mul_θ requires that the interpolation points $0, 1, \dots, r_C$ be mutually distinct. Thus, this scheme would not have worked over a field of small characteristic, but would have remained valid in large enough characteristic.

In the original article [11], Tasks (T1–T3) are performed by matrix multiplications, as explained in the next lemma.

Lemma 2 *Let $d, r, s \in \mathbb{N}$ and let $a_0, \dots, a_s$ be distinct points in $\mathbb{K}$. Evaluating $d+1$ polynomials of degree r on the a_i ’s reduces to an instance of $\langle s+1, r+1, d+1 \rangle$ plus $\mathcal{O}(sr)$ ops. Interpolating $d+1$ polynomials of degree s on the a_i ’s amounts to an instance of $\langle s+1, s+1, d+1 \rangle$ plus $\mathcal{O}(s^2)$ ops.*

PROOF. The omitted proof is based on grouping multiplications by Vandermonde matrices into a single product. $\square$

Using Lemma 2, one immediately deduces the cost of van der Hoeven’s algorithm “à la lettre” (vdH_θ); the following enumeration displays only the dominating costs, quadratic estimates like $\mathcal{O}(r_C r_A)$ being intentionally neglected:

1. $\text{MM}(r_C + 1, r_A + 1, d_A + 1)$ for (T1);
2. $\text{MM}(d_A + r_C + 1, r_B + 1, d_B + 1)$ for (T2);
3. $\text{MM}(r_C + 1, r_C + 1, d_C + 1)$ for (T3);
4. $\text{MM}(d_C + r_C + 1, d_A + r_C + 1, r_C + 1)$ for (T4).

Notice that the last step dominates the cost.

For Problem $\langle n, n \rangle_\theta$ which is studied in [11], applying the estimate (1) leads to the number $2 + 3 + 2 \cdot 2 \cdot 2 + 4 \cdot 3 \cdot 2 = 37$ of $n \times n$ block multiplications given in column vdH_θ of Table 1. This estimate is however pessimistic and can be reduced to 20: Strassen’s formula reduces the 8 block products in Task (T3) to 7; the band structure of the matrices in Task (T4) reduces 24 to only 8 products of non-zero blocks.

A first improvement. Algorithm vdH_θ can be improved by making use of fast multipoint evaluation and interpolation of Lemma 1(b) to perform Steps 1 and 3 of Algorithm Mul_θ in Fig. 1. This remark will be crucial in our proof of equivalence in §3.2. We arrive at the following complexity estimates:

1. $\mathcal{O}(d_A M(r_C) \log r_C)$ for (T1);
2. $\mathcal{O}(d_B M(d_A + r_C) \log(d_A + r_C))$ for (T2);
3. $\mathcal{O}(d_C M(r_C) \log r_C)$ for (T3).

Assuming FFT is available for polynomial multiplication, the cumulated cost of Tasks (T1–T3) drops to

$$\tilde{\mathcal{O}}(d_{ArC} + d_B(d_A + r_C) + d_C r_C) \in \tilde{\mathcal{O}}(d_C r_C + d_A d_B).$$

This cost is nearly optimal, since it is almost linear in the number of non-zero elements of the matrices involved in Eq. (7). In the particular case of problem $\langle n, n \rangle_\theta$, we obtain the numbers 24 and 8 of column ldH_θ in Table 1.

3.2 Matrix multiplication reduces to product in $\mathbb{K}[X]\langle\partial\rangle$

In summary, the results of the previous section show that $C_\theta(n) \in \mathcal{O}(\text{MM}(n))$. Here we prove the converse statement, by proceeding in two steps. First, Lemma 3 shows that the multiplication, whose complexity is denoted $\text{T}(n)$, of two lower-triangular matrices of size $n \times n$ reduces to the product of two operators of bidegree at most (n, n) in (X, θ) . Secondly, Lemma 4 proves that multiplying two arbitrary matrices amounts to a constant number of products of lower-triangular matrices.

Lemma 3 $\text{T}(n) \in C_\theta(n) + \mathcal{O}(n M(n) \log n)$.

PROOF. Let L_1, L_2 be two $(n+1) \times (n+1)$ lower-triangular matrices. Denote $(t_{i,j})$ and $(s_{i,j})$ their coefficients, with $0 \leq i, j \leq n$. Let $\tilde{B}_\ell(X)$ and $\tilde{A}_\ell(X)$ be the (unique) polynomials in $\mathbb{K}[X]$ of degree at most $n - \ell$ that interpolate the elements of the ℓ th lower diagonal of L_1 , resp. L_2 , on the set $\{0, 1, \dots, n - \ell\}$. Specifically, for $0 \leq \ell, j \leq n$ with $\ell + j \leq n$, we have $t_{\ell+j,j} = \tilde{B}_\ell(j)$ and $s_{\ell+j,j} = \tilde{A}_\ell(j)$. Using fast interpolation, the computation of the polynomials $\tilde{B}_\ell(X)$ and $\tilde{A}_\ell(X)$, for $0 \leq \ell \leq n$, is done in $\mathcal{O}(n M(n) \log n)$ ops. Define $A = \sum_{\ell=0}^n \sum_{j=0}^{n-\ell} a_{\ell,j} X^\ell \theta^j$ and $B = \sum_{\ell=0}^n \sum_{j=0}^{n-\ell} b_{\ell,j} X^\ell \theta^j$ from the coefficients in $\tilde{A}_\ell(X) = \sum_{j=0}^{n-\ell} a_{\ell,j} X^j$ and $\tilde{B}_\ell(X) = \sum_{j=0}^{n-\ell} b_{\ell,j} X^j$. Let $C = BA$. Then, L_1 and L_2 are seen to be top-left blocks of M^B and M^A , and Eq. (7) with A replaced by C shows that the top-left $(n+1) \times (n+1)$ submatrix of M^C is the lower-triangular matrix $L_1 L_2$. This submatrix is computed starting from the coefficients of C using $\mathcal{O}(n M(n) \log n)$ ops., by fast multipoint evaluation. $\square$

Lemma 4 $\text{MM}(n) \in \mathcal{O}(\text{T}(n))$.

PROOF. Let M, N be $n \times n$ matrices. The identity

$$\begin{bmatrix} I_n & 0 & 0 \\ M & I_n & 0 \\ 0 & N & I_n \end{bmatrix}^2 = \begin{bmatrix} I_n & 0 & 0 \\ 2M & I_n & 0 \\ NM & 2N & I_n \end{bmatrix}$$

shows that $\text{MM}(\lceil n/3 \rceil) \leq \text{T}(n) \in \mathcal{O}(\text{T}(n))$ and the conclusion follows from the growth hypotheses on MM . $\square$

Lemmas 3 and 4 imply the main result of this section.

Theorem 1 *There exists a constant $K > 0$ such that*

$$\text{MM}(n) \leq K(C_\theta(n) + n M(n) \log n).$$

3.3 Equivalence between product in $\mathbb{K}[X]\langle\partial\rangle$ and in $\mathbb{K}[X]\langle\theta\rangle$

Relax $\mathbb{K}$ to be a field of arbitrary characteristic. Any operator $A = \sum_{i=0}^r \alpha_i \partial^i$ in $\mathbb{K}[X]\langle\theta\rangle$ with coefficients α_i of degree at most d can be expressed in the algebra $\mathbb{K}[X]\langle\partial\rangle$ as $A = \sum_{i=0}^r a_i \partial^i$, with coefficients a_i of degree at most $d + r$.

As indicated in the proof of [4, Cor. 2], performing the conversion from the representation in θ to the representation in ∂ amounts to multiplying a Stirling matrix S of size $r + 1$ by an $(r + 1) \times (d + 1)$ matrix containing the coefficients of the α_i 's. This matrix product can be decomposed into $d + 1$ matrix-vector products of the form $w = Sv$. The coefficients of the vector w represent the coefficients of the polynomial $\sum_i v_i X^i$ in the falling factorial basis $(X)_k$. As Lemma 1(c) holds for any characteristic, w can be computed using $\mathcal{O}(M(r) \log r)$ ops. To summarize, the coefficients a_i can be computed from the α_i 's in $\mathcal{O}(d M(r) \log r)$ ops.

Conversely, let $B = \sum_{i=0}^r b_i \partial^i$ be in $\mathbb{K}[X]\langle\partial\rangle$. It can be written in the algebra $\mathbb{K}[X, X^{-1}]\langle\theta\rangle$ of differential operators in θ with Laurent polynomial coefficients as follows: $B = \sum_{i=0}^r \beta_i \theta^i$. If the b_i 's have degrees bounded by d , then the β_i 's have degrees at most d and valuation at least $-r$ in X . A discussion similar to above shows that the computation of the coefficients β_i from the coefficients b_i amounts to multiplying the inverse of the Stirling matrix by an $(r + 1) \times (d + r + 1)$ matrix. This matrix product can be decomposed into $d + r + 1$ matrix-vector products by S^{-1} ; this amounts to expanding in the monomial basis $d + r + 1$ polynomials of degree at most r given in the falling factorial basis. Thus, the conversion can be done in $\mathcal{O}((d + r) M(r) \log r)$ ops.

We encapsulate this discussion into the following result, which proves that $\langle n, n \rangle_\partial$ and $\langle n, n \rangle_\theta$ are computationally equivalent, up to $\tilde{\mathcal{O}}(n^2)$ terms, in any characteristic.

Theorem 2 *There exist a constant $C > 0$ such that*

$$\begin{aligned} C_\theta(n) &\leq C(C_\partial(n) + n M(n) \log n), \\ C_\partial(n) &\leq C(C_\theta(n) + n M(n) \log n), \end{aligned}$$

over fields of any characteristic.

PROOF. Let A_1, A_2 be of bidegree (n, n) in $\mathbb{K}[X]\langle\theta\rangle$. By the previous discussion, converting them into $\mathbb{K}[X]\langle\partial\rangle$ has cost $\mathcal{O}(n M(n) \log n)$. Both A_1, A_2 have bidegrees at most $(2n, n)$ in (X, ∂) and can thus be multiplied using $C_\partial(2n) \in \mathcal{O}(C_\partial(n))$ ops. Converting the result back into $\mathbb{K}[X]\langle\theta\rangle$ costs $\mathcal{O}(n M(n) \log n)$ ops. This proves the first inequality.

Let now B_1 and B_2 be of bidegree (n, n) in $\mathbb{K}[X]\langle\partial\rangle$. Their conversion in $\mathbb{K}[X, X^{-1}]\langle\theta\rangle$ can be performed using $\mathcal{O}(n M(n) \log n)$ ops. and produces two operators C_1 and C_2 in $\mathbb{K}[X]\langle\theta\rangle$, of bidegrees at most $(2n, n)$ in (X, θ) such that $B_1 = X^{-n} C_1(X, \theta)$ and $B_2 = X^{-n} C_2(X, \theta)$. Using the commutation rule $C_2(X, \theta) X^{-n} = X^{-n} C_2(X, \theta - n)$, we deduce the equality $B_2 B_1 = X^{-2n} C_2(X, \theta - n) C_1(X, \theta)$. Writing $C_2(X, \theta) = \sum_{j=0}^n X^j c'_j(\theta)$ shows that computing the coefficients of $C_2(X, \theta - n)$ amounts to $n + 1$ polynomial shifts in $\mathbb{K}[\theta]$ in degree at most n . Each of these shifts can be computed in $\mathcal{O}(M(n) \log n)$ ops., using Lemma 1(a). Conversion of $B_2 B_1$ back into $\mathbb{K}[X]\langle\partial\rangle$ has the same cost. $\square$

4. BETTER CONSTANTS IN $\mathbb{K}[X]\langle\partial\rangle$

In §4.1, we revisit van der Hoeven's algorithm for $\langle n, n \rangle_\partial$ and exhibit the constant factor in its $\mathcal{O}(\text{MM}(n))$ cost. Then, we propose in §4.2 a new algorithm with a better constant.

4.1 Multiplication in $\mathbb{K}[X]\langle\partial\rangle$: van der Hoeven's algorithm revisited

Van der Hoeven's algorithm for computing products in $\mathbb{K}[X]\langle\partial\rangle$ is based on the fact that his algorithm for products in $\mathbb{K}[X]\langle\theta\rangle$ can be adapted to operators with Laurent polynomials coefficients. Indeed, to any $L \in \mathbb{K}[X, X^{-1}]\langle\theta\rangle$ of the form $L = \sum_{i=-v}^d \sum_{j=0}^r \ell_{i,j} X^i \theta^j$ is associated an infinite matrix representing the $\mathbb{K}$ -linear map of multiplication by L from $\mathbb{K}[X]$ to $X^{-v}\mathbb{K}[X]$. Its $(v+d+r+1) \times (r+1)$ -submatrix $M_{0,r}^L$ (defined shortly) is banded and it uniquely determines the operator L , as in the case of polynomial coefficients.

To be precise, for two integers $\alpha \leq \beta$ we denote by $M_{\alpha,\beta}^L$ the $(v+d+\beta-\alpha+1) \times (\beta-\alpha+1)$ matrix whose $(\gamma-\alpha+1)$ -th column, for $\alpha \leq \gamma \leq \beta$, contains the coefficients of $L(X^\gamma)$ on $X^{-v+\alpha}, \dots, X^{d+\beta}$. The matrix $M_{\alpha,\beta}^L$ has a banded form and contains on its diagonals the evaluations on the points $\alpha, \dots, \beta$ of the polynomials $\tilde{L}_{-v}, \dots, \tilde{L}_d$ defined by $\tilde{L}_i(X) = \sum_{j=0}^r \ell_{i,j} X^j$ for all $-v \leq i \leq d$.

Let A, B have valuations $-v_A, -v_B$ and degrees d_A, d_B with respect to X , and degrees r_A, r_B in θ . If $C = BA$ in $\mathbb{K}[X, X^{-1}]\langle\theta\rangle$, then the following equality, analogous to Eq. (7), holds:

$$M_{0,r_C}^C = M_{-v_A, d_A+r_C}^B M_{0,r_C}^A. \quad (8)$$

Likewise, the product of operators in $\mathbb{K}[X, X^{-1}]\langle\theta\rangle$ reduces to some evaluation and interpolation tasks (in order to convert between operators and matrices) and to the main matrix-multiplication task (8), which is an instance of $\langle v_C + d_C + r_C + 1, v_A + d_A + r_C + 1, r_C + 1 \rangle$.

The algorithm for multiplication in $\mathbb{K}[X]\langle\partial\rangle$ based on multiplication in $\mathbb{K}[X, X^{-1}]\langle\theta\rangle$ is described in Fig. 2 below.

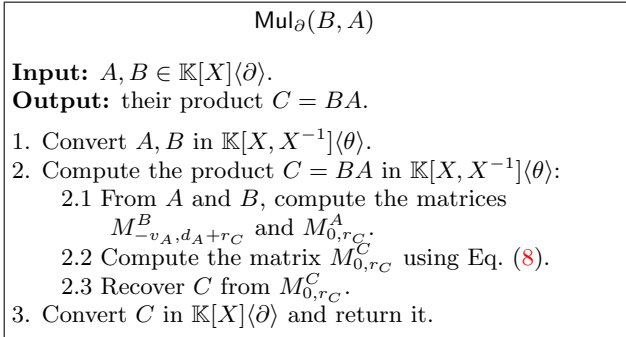


Figure 2: Product of differential operators in ∂ .

In what follows, we treat in more detail the main case of interest, $\langle n, n \rangle_\partial$, as solved by Algorithm Mul_∂ in Fig. 2. Van der Hoeven suggests to perform Steps 1 and 3 using matrix multiplications by Stirling matrices and their inverses [11, §5.1, Eqs. (12–13)] and Steps 2.1 and 2.3 using matrix multiplications by Vandermonde matrices and their inverses [11, §2 and §4]. The elements of all the needed Stirling and Vandermonde matrices (and their inverses) can be computed using $\mathcal{O}(n^2)$ ops. A careful inspection of the matrix sizes involved in Algorithm Mul_∂ shows that:

1. Step 1 reduces to 2 instances of $\langle 2n+1, n+1, n+1 \rangle$;
2. Step 3 reduces to an instance of $\langle 4n+1, 2n+1, 2n+1 \rangle$;
3. Step 2.1 reduces to an instance of $\langle 2n+1, n+1, 4n+1 \rangle$ and an instance of $\langle 2n+1, n+1, 2n+1 \rangle$;

4. Step 2.3 reduces to an instance of $\langle 4n+1, 2n+1, 2n+1 \rangle$;
5. Step 2.2 reduces to an instance of $\langle 6n+1, 4n+1, 2n+1 \rangle$.

This variant of the algorithm is what we call **vdH**. Using again the estimate (1) yields the constant 96 in Table 1.

Several Improvements. A first improvement on vdH is to use fast multipoint evaluation and interpolation for Steps 2.1 and 2.3. A second improvement concerns conversions back and forth between operators in $\mathbb{K}[X, X^{-1}]\langle\partial\rangle$ and in $\mathbb{K}[X, X^{-1}]\langle\theta\rangle$ (Steps 1 and 3). Instead of using matrix products by Stirling matrices and their inverses, one can apply Lemma 1(c), as explained in §3.3. Both improvements in conjunction with FFT lessen the cost of Steps 1, 2.1, 2.3, and 3 to a negligible $\tilde{\mathcal{O}}(n^2)$. We call this improved algorithm **lvdH**. Using (1) yields the constant 48 in column lvdH in Table 1. The constants 47 and 12 on the last row of the table are more technical and will be proved in [2]. They rely on observing that the output of lvdH requires partial calculation of (8), reducing to an instance of $\langle 4n+1, 3n+1, 2n+1 \rangle$.

4.2 A new, direct evaluation-interpolation algorithm

Let A and B be in $\mathbb{K}[X]\langle\partial\rangle$ with respective bidegrees (d_A, r_A) and (d_B, r_B) . We give here an evaluation-interpolation algorithm for computing $C = BA$ which essentially reduces to $\langle d_C + 1, d_A + r_C + 1, r_C + 1 \rangle$ for those bidegrees.

To achieve this, we interpret again a differential operator P in $\mathbb{K}[X]\langle\partial\rangle$ as a $\mathbb{K}$ -endomorphism of $\mathbb{K}[X]$, and represent it in the canonical basis $(X^i)_{i \geq 0}$ by an (infinite) matrix denoted $\tilde{M}_\infty^P$. The submatrix of $\tilde{M}_\infty^P$ consisting of its first $r+1 \geq 1$ rows and $c+1 \geq 1$ columns is denoted $\tilde{M}_{r,c}^P$.

Then, much like Algorithm Mul_θ in §3.1, our new algorithm MulWeyl in Fig. 4 relies on the key observation that an operator $P \in \mathbb{K}[X]\langle\partial\rangle$ of bidegree (d, r) is uniquely determined by the submatrix $\tilde{M}_{d,r}^P$ of $\tilde{M}_\infty^P$. This key fact is proved in Theorem 4 below. The principle of the algorithm is given in Fig. 3, where evaluation and interpolation are performed by truncated-series products. In the case of $\langle n, n \rangle_\partial$, the cor-

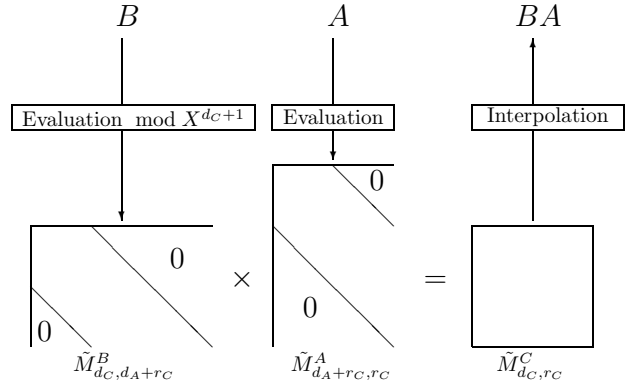


Figure 3: Evaluation-Interpolation w.r.t. ∂ .

responding matrices become $\tilde{M}_{2n,3n}^B$, $\tilde{M}_{3n,2n}^A$, and $\tilde{M}_{2n,2n}^C$.

Theorem 3 *Algorithm MulWeyl is correct and uses $\text{MM}(d_C + 1, d_A + r_C + 1, r_C + 1) + \tilde{\mathcal{O}}((d_C + r_C)^2)$ ops.*

MulWeyl(B, A)

Input: $A, B \in \mathbb{K}[X]\langle \partial \rangle$.

Output: their product $C = BA$.

1. Construct the matrices $\tilde{M}_{d_A+r_C, r_C}^A$ and $\tilde{M}_{d_C, d_A+r_C}^B$.
2. Compute the product $\tilde{M}_{d_C, d_A+r_C}^B \tilde{M}_{d_A+r_C, r_C}^A$.
3. Recover C from the product in Step 2.

Figure 4: Product of differential operators in ∂ .

PROOF. By the definition of the matrix $\tilde{M}_{r,c}^P$, the matrices constructed in Step 1 are associated to the linear map which sends $f \in \mathbb{K}[X]_{\leq r_C}$ to $A(f)$ in $\mathbb{K}[X]_{\leq d_A+r_C}$ and to the linear map which sends $f \in \mathbb{K}[X]_{\leq d_A+r_C}$ to $B(f) \bmod X^{d_C+1}$ in $\mathbb{K}[X]_{\leq d_C}$. Therefore, the product at Step 2 delivers the $(BA)(X^i) \bmod X^{d_C+1}$, $0 \leq i \leq r_C$. The $(d_C+1) \times (r_C+1)$ matrix computed is thus equal to $\tilde{M}_{d_C, r_C}^C$. This is summarized in the identity $\tilde{M}_{d_C, d_A+r_C}^B \tilde{M}_{d_A+r_C, r_C}^A = \tilde{M}_{d_C, r_C}^C$, in which the structure of zeros is given in Fig. 3. The interpolation of Step 3 relies on Theorem 4 below, which shows that $C = BA$ is fully and uniquely determined by $\tilde{M}_{d_C, r_C}^C$. This terminates the correctness proof. The claimed complexity derives immediately from Propositions 2 and 3 that are proved in the next subsections. $\square$

4.2.1 Interpolation theorem

We now state the main interpolation result, which we prove after recalling a useful filtration on $W = \mathbb{K}[X]\langle \partial \rangle$.

Theorem 4 For $d, r \in \mathbb{N}$, let $W_{d,r}$ denote its $\mathbb{K}$ -subspace

$$W_{d,r} = \{ P \in W : \deg_X(P) \leq d, \deg_\partial(P) \leq r \}.$$

Then, an isomorphism is given by the $\mathbb{K}$ -linear map

$$\begin{array}{ccc} \text{EvOp}_{d,r} : & W_{d,r} & \rightarrow \mathbb{K}^{(d+1) \times (r+1)} \\ & P & \mapsto \tilde{M}_{d,r}^P \end{array}$$

In order to prove Theorem 4, we use the filtration on W defined by the weights 1 on X and -1 on ∂ . The decomposition into homogeneous components of any $P \in W_{d,r}$ only involves weights between $-r$ and d . It actually admits a special form, to be exploited later, which is described now.

Lemma 5 The homogeneous decomposition of $P \in W_{d,r}$ is

$$P = \sum_{i=1}^r \ell_{-i}(X\partial)\partial^i + \sum_{i=0}^d X^i \ell_i(X\partial),$$

where the ℓ_i 's and ℓ_{-i} 's are polynomials of degree at most $\mu_i := \min(d-i, r)$ and $\mu_{-i} := \min(r-i, d)$, respectively.

PROOF. Let P be $\sum_{i,j} p_{i,j} X^j \partial^i$. Then P decomposes as the sum of $\sum_{i>j} p_{i,j} (X^j \partial^j) \partial^{i-j}$ and of $\sum_{i \leq j} X^{j-i} p_{i,j} (X^i \partial^i)$. Here $p_{i,j}$ is zero if $i > r$ or if $j > d$, therefore P is equal to

$$\sum_{s=1}^r \left(\sum_{j=0}^{\mu_{-s}} p_{s+j,j} X^j \partial^j \right) \partial^s + \sum_{t=0}^d X^t \left(\sum_{i=0}^{\mu_t} p_{t+i,i} X^i \partial^i \right). \quad (9)$$

Since any $X^i \partial^i$ can be written as a polynomial of degree i in $X\partial$, the conclusion follows by expressing each parenthesis in (9) as a polynomial in $X\partial$. $\square$

PROOF OF TH. 4. Since $\dim_{\mathbb{K}} W_{d,r}$ is $\dim_{\mathbb{K}} \mathbb{K}^{(d+1) \times (r+1)}$, it suffices to show that $\text{EvOp}_{d,r}$ is injective. Let P in $W_{d,r}$ be such that $\tilde{M}_{d,r}^P = 0$, or equivalently $P(X^k) \bmod X^{d+1} = 0$ for all $0 \leq k \leq r$. The decomposition of $P \in W_{d,r}$ in Lemma 5 enables one to evaluate it easily at X^k for $k \leq r$:

$$P(X^k) = \sum_{i=1}^k \frac{k!}{(k-i)!} \ell_{-i}(k-i) X^{k-i} + \sum_{i=0}^d \ell_i(k) X^{k+i}. \quad (10)$$

Since $P(X^k) \bmod X^{d+1} = 0$ for $k \leq r$, Eq. (10) implies:

- $\ell_i(k) = 0$ if $0 \leq i \leq d$, $0 \leq k \leq r$, and $k+i \leq d$,
- $\ell_{-i}(k-i) = 0$ if $1 \leq i \leq k$, $0 \leq k \leq r$, and $k-i \leq d$.

These equalities show that $\ell_i(0), \dots, \ell_i(\min(d-i, r))$ are zero for $0 \leq i \leq d$ and that $\ell_{-i}(0), \dots, \ell_{-i}(\min(r-i, d))$ are zero for $1 \leq i \leq r$. Finally, Lagrange interpolation and the degree bounds in Lemma 5 imply that all the polynomials ℓ_i and ℓ_{-i} are identically zero. Thus, P is 0. $\square$

A direct use of the ideas of this subsection would now end the proof of Theorem 3; the corresponding algorithm would first compute the polynomials ℓ_i and ℓ_{-i} , before evaluating them on $0, 1, \dots$. By the following next two subsections, we shall propose a better solution, avoiding a logarithmic factor and hiding a smaller constant in the $\tilde{\mathcal{O}}(\cdot)$ term.

4.2.2 Evaluation step

Here we focus on Step 1 of Algorithm MulWeyl, which is an instance of the task of computing the matrix $\tilde{M}_{m,n}^P$ for given $P = \sum_{i=0}^r \sum_{j=0}^d p_{i,j} X^j \partial^i$ in W and integers $m \geq d, n \geq r$. The announced better approach makes use of Algorithm Eval in Fig. 5, which is based on the following observation: Let $0 \leq k \leq n$. Then we have the identities

$$\begin{aligned} P(X^k) &= \sum_{i=0}^{\min(r,k)} \sum_{j=0}^d p_{i,j} \frac{k!}{(k-i)!} X^{k+j-i} \\ &= k! X^k \left(\sum_{\ell=-\min(r,k)}^d \left(\sum_{i=\max(0,-\ell)}^{\min(r,d-\ell,k)} \frac{p_{i,i+\ell}}{(k-i)!} \right) X^\ell \right). \end{aligned}$$

Therefore, for $-r \leq \ell \leq d$ and $0 \leq k \leq n$, the coefficient $(S_\ell)_k$ of X^k in the polynomial product

$$S_\ell = \left(\sum_{i=\max(0,-\ell)}^{\min(r,d-\ell)} p_{i,i+\ell} X^i \right) \left(\sum_{j=0}^n \frac{X^j}{j!} \right) \quad (11)$$

gives the coefficient of X^ℓ in $(k! X^k)^{-1} P(X^k)$. Thus the coefficients $(S_\ell)_k$ for $\max(0, -\ell) \leq k \leq \min(m-\ell, n)$ of S_ℓ are, up to factorials, the coefficients on a certain diagonal of the matrix $\tilde{M}_{m,n}^P$, the other diagonals of $\tilde{M}_{m,n}^P$ being zero.

Proposition 2 Algorithm Eval computes $\tilde{M}_{m,n}^P$ in $\mathcal{M}(mn) + \mathcal{O}(mn)$ ops.

PROOF. The series $\exp(X) \bmod X^{n+1}$ and the factorials $1, \dots, n!$ are computed by recurrence relations in $\mathcal{O}(n)$ ops. The computation of S_ℓ can be done in $\mathcal{M}(s_\ell)$ for the size s_ℓ of the corresponding diagonal of $\tilde{M}_{m,n}^P$. Summing over ℓ and appealing to properties of $\mathcal{M}$ leads to $\sum_\ell \mathcal{M}(s_\ell) \leq \mathcal{M}(\sum_\ell s_\ell) \leq \mathcal{M}(mn) + \mathcal{O}(mn)$, then to the announced complexity. $\square$

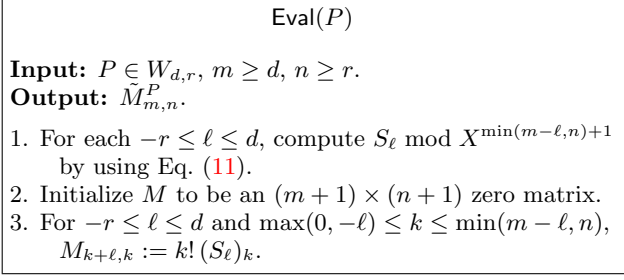


Figure 5: Evaluation in $\mathbb{K}[X]\langle\partial\rangle$.

4.2.3 Interpolation step

Given a $(d+1) \times (r+1)$ matrix M , Step 3 of Algorithm MulWeyl computes the only operator $P \in W_{d,r}$ satisfying $\tilde{M}_{d,r}^P = M$. This is done by inverting Eq. (11). The resulting algorithm is described in Fig. 6. A similar analysis to that of algorithm Eval leads to the estimate in Proposition 3.

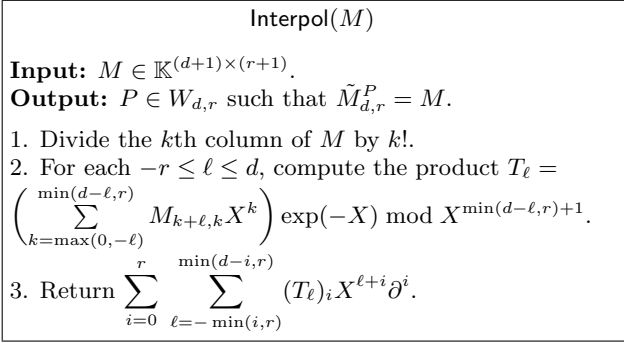


Figure 6: Interpolation in $\mathbb{K}[X]\langle\partial\rangle$.

Proposition 3 *Interpol computes P in $M(dr) + \mathcal{O}(dr)$ ops.*

4.3 Comparison of algorithms for $\langle n, n \rangle_\partial$

Algorithms Mul $_\partial$ in Fig. 2 and MulWeyl in Fig. 4 follow the same scheme: construction of evaluation matrices associated to A and B ; product of these matrices; reconstruction of C by interpolation from it. But they differ in the way to do this, and MulWeyl can be viewed as an improvement on Mul $_\partial$: the matrices computed by MulWeyl are submatrices of M^B and M^A in Algorithm Mul $_\partial$, as will be proved in [2]. Taking accurate sizes into account for $\langle n, n \rangle_\partial$, the dominant matrix-product problem drops from $\langle 6n+1, 4n+1, 2n+1 \rangle$ to $\langle 2n+1, 3n+1, 2n+1 \rangle$. Estimate (1) yields the number 12 in the last column of Table 1. Observing that the product at Step 2 of MulWeyl reduces to one instance of $\langle 2n+1, 2n+1, 2n+1 \rangle$ and one of $\langle n, n, n \rangle$, and appealing to Strassen's formula again, we obtain $7+1=8$ block products, as given on the last row of Table 1.

5. PRODUCT IN CHARACTERISTIC > 0

As already pointed out, the evaluation-interpolation algorithms of Sections 3 and 4 remain valid when the characteristic p of $\mathbb{K}$ is positive and sufficiently large, but they fail to work in small characteristic. For instance, MulWeyl solves Problem $\langle n, n \rangle_\partial$ for characteristic $p > 3n$.

In this section, we provide an algorithm of different nature which proves that, in characteristic p , the product of two operators of bidegree (n, n) either in $\mathbb{K}[X]\langle\theta\rangle$ or in $\mathbb{K}[X]\langle\partial\rangle$ can be computed in $\tilde{\mathcal{O}}(pn^2)$ ops. For small p , this result is nearly optimal, since it is softly linear in the output size.

Up to $\tilde{\mathcal{O}}(n^2)$ additional ops., multiplication in $\mathbb{K}[X]\langle\partial\rangle$ can be reduced to multiplication in $\mathbb{K}[X]\langle\theta\rangle$, as explained in §3.3. Thus, we focus on Problem $\langle n, n \rangle_\theta$.

Our algorithm Mul $_{\theta,p}$ for multiplication in $\mathbb{K}[X]\langle\theta\rangle$ is given in Fig. 7. It is based on the key fact that θ and X^p commute in characteristic p . This is used in Step 2, which reduces the product in $\mathbb{K}[X]\langle\theta\rangle$ to several products in the commutative polynomial ring $\mathbb{K}[X^p, \theta]$.

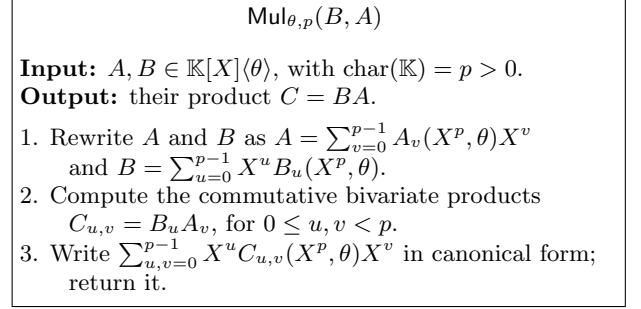


Figure 7: Product of differential operators in θ over a field of positive characteristic.

We now describe proper algorithmic choices that perform each step of Mul $_{\theta,p}$ in nearly optimal complexity.

Step 1 first rewrites A as $\sum_{v=0}^{p-1} X^v \tilde{A}_v(X^p, \theta)$ and B as $\sum_{u=0}^{p-1} X^u B_u(X^p, \theta)$, where $B_u, \tilde{A}_v$, $0 \leq u, v \leq p-1$ are polynomials in $\mathbb{K}[X^p, \theta]$ of bidegree at most $(\lfloor n/p \rfloor, n)$; this costs no ops. The commutation $\theta^j X^v = X^v (\theta + v)^j$ then enables one to rewrite A as $\sum_{v=0}^{p-1} A_v(X^p, \theta) X^v$, where $A_v(X^p, \theta)$ is $\tilde{A}_v(X^p, \theta - v)$. Thus, each A_v is obtained by computing $\lfloor n/p \rfloor + 1$ shifts of polynomials of degree at most n . By Lemma 1(a), this results in $\mathcal{O}(nM(n) \log n)$ ops. for Step 1.

Each product in Step 2 involves polynomials in $\mathbb{K}[X^p, \theta]$ of bidegree at most $(\lfloor n/p \rfloor, n)$. Thus using Lemma 1(d), Step 2 is performed in $\mathcal{O}(p^2 M(n^2/p)) \subseteq \mathcal{O}(pM(n^2))$ ops. Note that $C_{u,v}(X, Y)$ has bidegree at most $(2\lfloor n/p \rfloor, 2n)$.

To perform Step 3, each $C_{u,v}(X^p, \theta) X^v$ is first rewritten as $X^v \tilde{C}_{u,v}(X^p, \theta)$ by computing $2\lfloor n/p \rfloor + 1$ shifts of polynomials of degree at most $2n$. This can be done in $\mathcal{O}(pnM(n) \log n)$ ops. Finally, $\mathcal{O}(pn^2)$ ops. are sufficient to put $C = \sum_{u=0}^{p-1} X^u \sum_{v=0}^{p-1} X^v \tilde{C}_{u,v}(X^p, \theta)$ in canonical form.

Summarizing, we have just proved:

Theorem 5 *Let $\mathbb{K}$ be a field of characteristic p and let D be one of the operators ∂, θ . Then, two operators of bidegree (n, n) in $\mathbb{K}[X]\langle D \rangle$ can be multiplied in $\mathcal{O}(pM(n^2) + pnM(n) \log n)$ ops., thus in $\tilde{\mathcal{O}}(pn^2)$ ops. when FFT is used.*

6. EXPERIMENTS

Table 2 provides timings of calculations in magma by implementations of several algorithms and algorithmic variants. Each row corresponds to calculations on the same pair of randomly generated operators in bidegree (n, n) , for $n = 10 \cdot 2^k$. Coefficients are taken randomly from $\mathbb{Z}/p\mathbb{Z}$ when $p > 0$, the prime used being $p_1 = 65521$ (largest prime

to fit on 16 bits) and $p_2 = 4294967291$ (largest prime to fit on 32 bits). When $p = 0$, computations are performed over $\mathbb{Q}$, with random integer input coefficients on 16 bits.

p	k	S	B	BZ	vdH	Iter	Tak	Rec	Int	BZI	vdHI
p_1	3	0.25	0.26	0.25	0.39	0.32	1.23	0.01	0.64	5.22	59.8
p_1	4	0.95	0.97	0.95	1.68	4.13	12.09	0.03	4.37	35.0	418
p_1	5	4.08	4.11	4.34	8.10	37.2	123	0.20	30.2	240	2793
p_1	6	21.4	21.1	22.2	45.1	397	1407	1.56	209	1692	∞
p_1	7	107	105	104	275	∞	∞	13.3	1507	∞	∞
p_2	3	0.50	0.63	0.62	1.08	2.25	5.61	0.08	1.10	8.00	82.2
p_2	4	2.24	2.66	2.68	4.52	19.07	67.73	0.35	9.22	58.2	602
p_2	5	12.2	14.5	14.1	24.4	187	926	1.63	75.6	420	∞
p_2	6	88.1	111	114	172	2604	∞	9.40	770	3146	∞
p_2	7	1961	2452	2633	∞	∞	∞	59.1	∞	∞	∞
0	3	9.93	12.0	11.3	28.4	6.99	24.3	0.07	0.93	16.9	309
0	4	128	164	164	498	118	725	0.27	6.89	204	∞
0	5	2164	2737	2725	∞	2492	∞	4.37	51.4	3172	∞

Table 2: Timings on input of bidegree $(10 \cdot 2^k, 10 \cdot 2^k)$.

The calculations were performed on a Power Mac G5 with two CPUs at 2.7 GHz, 512 kB of L2 Cache per CPU, 2.5 GB of memory, and a bus of speed 1.35 GHz. The system used was Mac OS X 10.4.10, running Magma V2.13-15. Computations killed after one hour are marked ∞ .

We provide several variants of our algorithm (S, B, and BZ), as well as various others: **S**: direct call to magma’s matrix multiplication in order to compute $\tilde{M}_{2n,3n}^B \tilde{M}_{3n,2n}^A$; **B** and **BZ**: block decomposition into $n \times n$ matrices before calling magma’s matrix multiplication on, respectively, 11 block products (using Strassen’s algorithm) and by 8 block products (taking the nullity of 2 blocks into account as well); **vdH**: Van der Hoeven’s algorithm, as described in [11], and optimized as much as possible as the implementation S above; **Iter** and **Tak**: iterative formulas (2) and (5); **Rec**: magma’s multiplication of a $(2n+1) \times (3n+1)$ -matrix by a $(3n+1) \times (2n+1)$ -matrix, that is, essentially all the linear algebra performed in variant S (in practice, almost always in the cubic regime for the objects of interest); **Int**: fully interpreted implementation of Strassen’s product with cubic loop under a suitable threshold; **BZI** and **vdHI**: variants of the implementations BZ and vdH (with evaluation-interpolation steps improved) in which magma’s product of matrices has been replaced with Int.

p	k	p_1	p_1	p_2	p_2	p_2	0	0	0
		3	7	3	5	7	3	4	5
LA	$\mathcal{O}(\text{MM}(n))$	4%	13%	17%	16%	39%	36%	41%	52%
PP	$\mathcal{O}(n \text{M}(n))$	13%	25%	23%	23%	18%	36%	33%	24%
OM	$\mathcal{O}(n^2)$	38%	36%	30%	27%	11%	7%	6%	5%
IO	$\mathcal{O}(n^2)$	46%	27%	30%	33%	32%	21%	20%	19%

Table 3: Fraction of time spent in matrix product (LA), polynomial products (PP), other matrix operations (OM), and other interpreted operations (IO).

Comparing the columns Rec and, for instance, S, shows that linear algebra does not take the main part of the calculation time, although its theoretical complexity dominates. In this regard, we have been very cautious in our implementation to avoid any interpreted quadratic loops. Still, the result is that those quadratic tasks dominate the computation time. Details are given in Table 3. The conclusion is that having implemented the algorithms in an interpreted

language tends to parasitize the benchmarks. For comparison sake, we have also added timings for variants BZI and vdHI that use an interpreted matrix product. They both show the growth expected in theory, as well as the ratio from 8 to 96 announced in Table 1.

7. CONCLUSIONS, FUTURE WORK

Because of space limitation, various extensions could not be covered here. More results on the complexity of non-commutative multiplication of skew polynomials will be presented in an upcoming extended version [2]. Topics like multiplication of skew polynomials with unbalanced degrees and orders, or with sparse support, will be treated there. The case of rational (instead of polynomial) coefficients will also be considered. The methods of this article extend to multiplication of more general skew polynomials, in one or several variables, including for instance q -recurrences and partial differential operators.

The constants in Table 1 are all somewhat pessimistic. Tighter bounds can be obtained by, on the one hand, relaxing the naive assumption (1), on the other hand, taking advantage of the special shapes (banded, trapezoidal, etc) of the various matrices.

We also plan to provide a lower-level implementation. Hopefully, the timings would then reflect the theoretical results even better and will be close to those of naked matrix products.

Acknowledgments. This work was supported in part by the French National Agency for Research (ANR Project “Gecko”) and the Microsoft Research-INRIA Joint Centre. We thank the three referees for their valuable comments.

8. REFERENCES

- [1] D. J. Bernstein. Fast multiplication and its applications. To appear in Buhler-Stevenhagen *Algorithmic number theory*.
- [2] A. Bostan, F. Chyzak, and N. Le Roux. Skew-polynomial products by evaluation and interpolation. In preparation.
- [3] A. Bostan, F. Chyzak, Z. Li, and B. Salvy. Common multiples of linear ordinary differential and difference operators. In preparation.
- [4] A. Bostan and É. Schost. Polynomial evaluation and interpolation on special sets of points. *Journal of Complexity*, 21(4):420–446, August 2005.
- [5] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Inform.*, 28(7):693–701, 1991.
- [6] F. Chyzak. <http://algo.inria.fr/chyzak/mgfun.html>.
- [7] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, Mar. 1990.
- [8] J. von zur Gathen and J. Gerhard. Fast algorithms for Taylor shifts and certain difference equations. In *Proceedings of ISSAC’97*, pages 40–47, New York, 1997. ACM Press.
- [9] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, 1999.
- [10] J. Gerhard. Modular algorithms for polynomial basis conversion and greatest factorial factorization. In *RWCA’00*, pages 125–141, 2000.
- [11] J. van der Hoeven. FFT-like multiplication of linear differential operators. *Journal of Symbolic Computation*, 33(1):123–127, 2002.
- [12] I. Kaporin. The aggregation and cancellation techniques as a practical tool for faster matrix multiplication. *Theor. Comput. Sci.*, 315(2-3):469–510, 2004.
- [13] A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7:281–292, 1971.
- [14] V. Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13:354–356, 1969.
- [15] N. Takayama. <http://www.math.kobe-u.ac.jp/KAN/>.

Implicitization and Parametrization of Quadratic Surfaces with One Simple Base Point

Xuhui Wang Falai Chen Jiansong Deng

Department of Mathematics
University of Science and Technology of China
Hefei, Anhui, 230026, P.R. China
{chenfl,dengjs}@ustc.edu.cn

ABSTRACT

This paper discusses implicitization and parametrization of quadratic surfaces with one simple base point. The key point to fulfill the conversion between the implicit and the parametric form is to compute three linearly independent moving planes which we call the *weak μ -basis* of the quadratic surface. Beginning with the parametric form, it is easy to compute the weak μ -basis, and then to find its implicit equation. Inversion formulas can also be obtained easily from the weak μ -basis. For conversion from the implicit into the parametric form, we present a method based on the observation that there exists one self-intersection line on a quadratic surface with one base point. After computing the self-intersection line, we are able to derive the weak μ -basis, from which the parametric equation can be easily obtained. A method is also presented to compute the self-intersection line of a quadratic surface with one base point.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms—*Algebraic algorithms*; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—*Curve, surface, solid, and object representations*

General Terms

Algorithms

Keywords

Implicitization, moving plane, parametrization, weak μ -basis

1. INTRODUCTION

In Computer Aided Geometric Design and Geometric Modelling, there are two complimentary ways of representing geometric objects: implicitly and parametrically, each of which has different advantages and disadvantages. It is valuable to have both representations at the same time. Thus conversion

between parametric and implicit form has been an interesting research topic for the past twenty years. The process of converting a parametric form into an implicit form is called *implicitization*, whereas the reverse process is called *parametrization*. In this paper, we are dedicated to investigating the implicitization and parametrization of a quadratic surface with one simple base point, which will be referred to as QSOB in the rest of the paper for brevity.

There is a wealth literature focusing on implicitization of parametric surfaces in the past two decades, and various methods have been proposed to solve the implicitization problem. These methods can be mainly classified into three categories: the resultant based method, the Groebner bases method and the method of moving surfaces (see [3, 9, 16, 17] and the references therein). The resultant based method is generally efficient. However, it fails in many situations, for example, in the presence of base points. On the other hand, the method based on Groebner bases is computationally expensive. The method of moving surfaces proposed by Sederberg and the second author (of this paper) is generally much more efficient than the methods based on Groebner bases and resultants, and it even simplifies computation in the presence of base points. However, the method of moving surfaces is largely at an experimental stage, and there is no proof to show that the method of moving surfaces always works. A rigorous approach along this direction is a technique based on syzygies (or μ -basis). The concept of μ -basis was firstly introduced in [12] to provide a compact representation for the implicit equation of a rational curve. It was then generalized to rational ruled surfaces and general rational parametric surfaces [5, 6, 8]. Unfortunately, the μ -basis doesn't directly provide an implicit equation for a general rational surface. In this paper, we will show that the method of moving surfaces works well for quadratic surfaces with one simple base point. Specifically, we will show that a QSOB has exactly three linearly independent moving planes which we call the *weak μ -basis* of the QSOB. From the weak μ -basis, the implicit equation of the QSOB can be easily derived.

Since the implicit degree of a QSOB is $2^2 - 1 = 3$, the implicit representation of a QSOB is a cubic surface. There are many works discussing the parametrization of nonsingular cubic surfaces [1, 2, 18, 20, 21], and most of these approaches are based on the properties that a nonsingular cubic surface contains exactly 27 straight lines and 45 tri-tangent planes [14]. In [2], Berry and Patterson unified the parametrization and the implicitization of nonsingular cubic surfaces by using the Hilbert-Burch matrices, which is

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

essentially another representation of three moving planes. However, the cubic surface derived from a QSOB is a singular surface, which doesn't have the above properties. In section 4, we will show that the cubic surface derived from a QSOB has a self-intersection line. Based on this observation, an algorithm is presented to derive a weak μ -basis from the implicit equation of a QSOB, and thus a parametric equation is obtained.

The remainder of the paper is organized as follows. In Section 2, we recall some preliminary results about moving planes and cubic surfaces. In Section 3 and Section 4, we present techniques for the implicitization and the parametrization of a QSOB. Singularities of a QSOB are also analyzed in detail. We conclude the paper in Section 5.

2. PRELIMINARIES

Let $R[s, t]$ be the ring of bivariate polynomials in s, t over some infinite field F . A rational parametric surface in homogeneous form is defined as follows:

$$\mathbf{P}(s, t) = (a(s, t), b(s, t), c(s, t), d(s, t)), \quad (1)$$

where $a, b, c, d \in R[s, t]$ are polynomials with $\gcd(a, b, c, d) = 1$.

A *base point* of a rational surface $\mathbf{P}(s, t)$ is a parameter pair (s_0, t_0) such that $\mathbf{P}(s_0, t_0) = (0, 0, 0, 0)$. Here s_0 and t_0 lie in the algebraically closed extension field of F . A QSOB is a rational quadratic parametric surface with one simple base point. The total degree of a QSOB in s, t is 2 and its implicit degree is $2^2 - 1 = 3$. In this paper, we will provide an efficient and rigorous method for implicitization and parametrization of a QSOB using the method of moving planes.

A *moving plane* is a family of planes with parameter pair (s, t) :

$$L(\mathbf{X}; s, t) := A(s, t)x + B(s, t)y + C(s, t)z + D(s, t)w = 0, \quad (2)$$

where $\mathbf{X} = (x, y, z, w)$ and $A(s, t), B(s, t), C(s, t), D(s, t) \in R[s, t]$. Sometimes, we write it in a vector form: $\mathbf{L}(s, t) := (A(s, t), B(s, t), C(s, t), D(s, t))$. A moving plane $\mathbf{L}(s, t)$ is said to *follow* the rational surface $\mathbf{P}(s, t)$ if

$$\mathbf{L}(s, t) \cdot \mathbf{P}(s, t) = a(s, t)A(s, t) + b(s, t)B(s, t) + c(s, t)C(s, t) + d(s, t)D(s, t) \equiv 0. \quad (3)$$

For a QSOB, there are exactly three independent moving planes of degree 1 in s, t , and these three moving planes serve as the connection between the parametric form and the implicit form of the QSOB.

Since the implicit degree of a QSOB is three, the following result is useful for analyzing the singularities of QSOBs.

PROPOSITION 2.1. [15] *If a cubic surface has finitely many singular points, then the number of singular points is less than or equal to four. Moreover, the cubic surface containing precisely 4 singular points is unique up to linear isomorphism.*

3. IMPLICITIZATION

In this section, we temporarily switch the parameters s, t to homogeneous parameters s, t, u . Then the parametric surface $\mathbf{P}(s, t)$ becomes

$$\mathbf{P}(s, t, u) = (a(s, t, u), b(s, t, u), c(s, t, u), d(s, t, u)).$$

For a quadratic surface $\mathbf{P}(s, t, u)$ with one base point $(s_0, t_0, u_0) \neq (0, 0, 0)$, there exist a nonsingular matrix $B \in F^{3 \times 3}$ such that $(s_0, t_0, u_0) = (0, 1, 0) \cdot B$. Let $(s, t, u) = (\tilde{s}, \tilde{t}, \tilde{u}) \cdot B$, then we get a new parametric representation for the quadratic surface:

$$\tilde{\mathbf{P}}(\tilde{s}, \tilde{t}, \tilde{u}) = (\tilde{a}(\tilde{s}, \tilde{t}, \tilde{u}), \tilde{b}(\tilde{s}, \tilde{t}, \tilde{u}), \tilde{c}(\tilde{s}, \tilde{t}, \tilde{u}), \tilde{d}(\tilde{s}, \tilde{t}, \tilde{u}))$$

which has the same implicit representation as $\mathbf{P}(s, t, u)$.

Since $\tilde{\mathbf{P}}(\tilde{s}, \tilde{t}, \tilde{u})$ contains the base point $(0, 1, 0)$, it doesn't contain the term $\tilde{t}^2$. Therefore, without loss of generality, we may assume that, in non-homogeneous variables, the quadratic surface $\mathbf{P}(s, t)$ has the form

$$\mathbf{P}(s, t) = \mathbf{P}_1^T + \mathbf{P}_s^T s + \mathbf{P}_t^T t + \mathbf{P}_{s^2}^T s^2 + \mathbf{P}_{st}^T st, \quad (4)$$

where $\mathbf{P}_1, \mathbf{P}_s, \mathbf{P}_t, \mathbf{P}_{st}$ and $\mathbf{P}_{s^2}$ are all column vectors in F^4 .

Before going on, we give a simple method to detect if a given quadratic parametrization is a QSOB or not.

Let

$$\begin{cases} A_1x + B_1y + C_1z + D_1 = 0 \\ A_2x + B_2y + C_2z + D_2 = 0 \end{cases}$$

be an arbitrary line. The intersection of this line with the parametric equation (1) of a QSOB is determined by

$$\begin{cases} f(s, t) := A_1a(s, t) + B_1b(s, t) + C_1c(s, t) + D_1d(s, t) = 0 \\ g(s, t) := A_2a(s, t) + B_2b(s, t) + C_2c(s, t) + D_2d(s, t) = 0 \end{cases}$$

Let h be the resultant of f and g with respect to s (or t). If $\deg(h) = 3$, then the quadratic parametrization is a QSOB.

A major observation about QSOBs is the following theorem.

THEOREM 3.1. *For a QSOB, there are exactly three linearly independent moving planes with degree one in s, t that follow it.*

PROOF. Let

$$\mathbf{L}(s, t) = \mathbf{L}_s s + \mathbf{L}_t t + \mathbf{L}_1$$

be a moving plane with total degree 1 in s, t that follows $\mathbf{P}(s, t)$. Here $\mathbf{L}_1, \mathbf{L}_s$ and $\mathbf{L}_t$ are row vectors in F^4 .

From

$$\mathbf{L}(s, t) \cdot \mathbf{P}(s, t) \equiv 0,$$

one obtains

$$(\mathbf{L}_s, \mathbf{L}_t, \mathbf{L}_1) \cdot Q = 0, \quad (5)$$

where Q is the 12×9 matrix

$$Q = \begin{pmatrix} 0 & \mathbf{P}_1 & 0 & \mathbf{P}_s & \mathbf{P}_t & 0 & \mathbf{P}_{s^2} & \mathbf{P}_{st} & 0 \\ 0 & 0 & \mathbf{P}_1 & 0 & \mathbf{P}_s & \mathbf{P}_t & 0 & \mathbf{P}_{s^2} & \mathbf{P}_{st} \\ \mathbf{P}_1 & \mathbf{P}_s & \mathbf{P}_t & \mathbf{P}_{s^2} & \mathbf{P}_{st} & 0 & 0 & 0 & 0 \end{pmatrix}. \quad (6)$$

We first note that the rank of the set of vectors $\{\mathbf{P}_1, \mathbf{P}_s, \mathbf{P}_t, \mathbf{P}_{s^2}, \mathbf{P}_{st}\}$ must be four, otherwise the implicit representation of $\mathbf{P}(s, t)$ is a plane. Now we claim that $\text{rank}(Q) = 9$, which would imply that there are exactly three linearly independent moving planes of degree one in s, t following $\mathbf{P}(s, t)$.

Suppose on the contrary that the columns $\mathbf{C}_i$ of Q are linearly dependent. Then there exist constants $k_1, k_2, \dots, k_9$,

at least one of them are nonzero, such that

$$\sum_{i=1}^9 k_i \mathbf{C}_i = \mathbf{0}.$$

The above equation is equivalent to

$$(\mathbf{P}_1, \mathbf{P}_s, \mathbf{P}_t, \mathbf{P}_{s^2}, \mathbf{P}_{st}) \cdot K = \mathbf{0},$$

where

$$K = \begin{pmatrix} k_1 & k_2 & k_3 \\ k_2 & k_4 & k_5 \\ k_3 & k_5 & k_6 \\ k_4 & k_7 & k_8 \\ k_5 & k_8 & k_9 \end{pmatrix}.$$

Since $\text{rank}(\mathbf{P}_1, \mathbf{P}_s, \mathbf{P}_t, \mathbf{P}_{s^2}, \mathbf{P}_{st}) = 4$, we conclude that $\text{rank}(K) = 1$. We assume that at least one of k_1, k_2, k_3, k_4, k_5 is nonzero (the other cases can be treated similarly and hence are omitted), then there exist constants λ and μ such that

$$\begin{aligned} k_2 &= \lambda k_1, & k_4 &= \lambda k_2, & k_5 &= \lambda k_3, \\ k_7 &= \lambda k_4, & k_8 &= \lambda k_5, \\ k_3 &= \mu k_1, & k_5 &= \mu k_2, & k_6 &= \mu k_3, \\ k_8 &= \mu k_4, & k_9 &= \mu k_5. \end{aligned}$$

Hence

$$k_1 \neq 0, \quad k_2 = \lambda k_1, \quad k_3 = \mu k_1, \quad k_4 = \lambda^2 k_1, \quad k_5 = \lambda \mu k_1.$$

Therefore

$$\mathbf{P}_1 + \lambda \mathbf{P}_s + \mu \mathbf{P}_t + \lambda^2 \mathbf{P}_{s^2} + \lambda \mu \mathbf{P}_{st} = \mathbf{0},$$

which means that $\mathbf{P}(s, t)$ has another base point $(\lambda, \mu, 1)$ apart from $(0, 1, 0)$, a contradiction. Hence, $\text{rank}(Q) = 9$ and there exist three linearly independent moving planes following $\mathbf{P}(s, t)$. $\square$

REMARK 3.1. *The three moving planes in Theorem 3.1 are called a weak μ -basis of the QSOB. Since a weak μ -basis corresponds to a basis of the solution space of the linear system of equations (5), two different weak μ -bases differ by the multiplication of a 3×3 invertible matrix. Therefore, the weak μ -basis of a QSOB is unique in the sense that it is the same under an invertible linear transformation.*

The weak μ -basis provides a direct method to implicitize a QSOB.

THEOREM 3.2. *Let*

$$L_i(\mathbf{X}; s, t) = L_{i1}(\mathbf{X})s + L_{i2}(\mathbf{X})t + L_{i3}(\mathbf{X}),$$

$i = 1, 2, 3$ be a weak μ -basis of $\mathbf{P}(s, t)$. Here $L_{ij}(\mathbf{X})$ is a linear function in x, y, z, w , $i, j = 1, 2, 3$. Then the implicit equation of $\mathbf{P}(s, t)$ is given by $f(\mathbf{X}) := \det(M) = 0$. Here

$$M = \begin{pmatrix} L_{11}(\mathbf{X}) & L_{12}(\mathbf{X}) & L_{13}(\mathbf{X}) \\ L_{21}(\mathbf{X}) & L_{22}(\mathbf{X}) & L_{23}(\mathbf{X}) \\ L_{31}(\mathbf{X}) & L_{32}(\mathbf{X}) & L_{33}(\mathbf{X}) \end{pmatrix}. \quad (7)$$

PROOF. Firstly, we will prove that the three moving planes L_1, L_2 and L_3 are $R[x, y, z, w]$ -linearly independent, or equivalently, $f(\mathbf{X})$ is not identically zero.

We first point out several basic facts.

(i) Let $Q = (q_{ij})_{4 \times 4}$ be an invertible matrix, and let

$$\begin{aligned} \tilde{\mathbf{P}}(s, t) &:= (\tilde{a}(s, t), \tilde{b}(s, t), \tilde{c}(s, t), \tilde{d}(s, t))^T \\ &= Q(a(s, t), b(s, t), c(s, t), d(s, t))^T. \end{aligned}$$

Suppose that the implicit equation of $\tilde{\mathbf{P}}(s, t)$ is $\tilde{f}(\mathbf{X}) = 0$. Then the implicit equation of $\mathbf{P}(s, t)$ is $\tilde{f}(\mathbf{X} \cdot Q^T) = 0$.

(ii) $\mathbf{P}_1, \mathbf{P}_t, \mathbf{P}_{st}$ and $\mathbf{P}_{s^2}$ are all non-zero vectors. In fact, $\mathbf{P}_t = \mathbf{0}$ or $\mathbf{P}_{st} = \mathbf{0}$ indicates that $(0, 1, 0)$ is a double base point of $\mathbf{P}(s, t)$. $\mathbf{P}_1 = \mathbf{0}$ implies $(0, 0, 1)$ is a base point of $\mathbf{P}(s, t)$. Similarly, $\mathbf{P}_{s^2} = \mathbf{0}$ means $(1, 0, 0)$ is also a base point of $\mathbf{P}(s, t)$. But $\mathbf{P}(s, t)$ has only one base point $(0, 1, 0)$.

(iii) By elementary row operations, the 4×5 matrix $(\mathbf{P}_1, \mathbf{P}_s, \mathbf{P}_t, \mathbf{P}_{s^2}, \mathbf{P}_{st})$ can be reduced into

$$(\tilde{\mathbf{P}}_1, \tilde{\mathbf{P}}_s, \tilde{\mathbf{P}}_t, \tilde{\mathbf{P}}_{s^2}, \tilde{\mathbf{P}}_{st}) = \begin{pmatrix} 1 & e_1 & 0 & 0 & 0 \\ 0 & e_2 & 1 & 0 & 0 \\ 0 & e_3 & 0 & 1 & 0 \\ 0 & e_4 & 0 & 0 & 1 \end{pmatrix} \quad (8)$$

(iv) $e_1 e_3 - 1 = 0$ and $e_1 e_4 + e_2 = 0$ can't hold simultaneously. Here $e_i, i = 1, 2, 3, 4$ are elements in the matrix (8). Indeed, suppose towards contradiction that both equalities hold. Then the parametric surface

$$\tilde{\mathbf{P}}(s, t) = \tilde{\mathbf{P}}_1 + \tilde{\mathbf{P}}_s s + \tilde{\mathbf{P}}_t t + \tilde{\mathbf{P}}_{s^2} s^2 + \tilde{\mathbf{P}}_{st} st \quad (9)$$

has a base point $(-1, e_2, e_1)$, which contradicts the fact that $\tilde{\mathbf{P}}(s, t)$ has just one base point.

Now we may assume that $e_1 e_3 - 1 \neq 0$ (the case when $e_1 e_4 + e_2 \neq 0$ is similar and hence is omitted). One can easily compute that a weak μ -basis of $\tilde{\mathbf{P}}(s, t)$ is

$$\begin{aligned} L_1(\mathbf{X}) &:= \beta z - \alpha w + (-\gamma x + \alpha y)s, \\ L_2(\mathbf{X}) &:= -\alpha y + \beta e_1 z - \alpha e_1 w - \beta x s + \alpha x t, \\ L_3(\mathbf{X}) &:= e_4 z - e_3 w - w s + z t, \end{aligned}$$

where $\alpha = e_1 e_3 - 1, \beta = e_1 e_4 + e_2, \gamma = e_2 e_3 + e_4$. It can be easily seen that

$$f(\mathbf{X}) := \begin{vmatrix} -\gamma x + \alpha y & 0 & \beta z - \alpha w \\ -\beta x & \alpha x & -\alpha y + \beta e_1 z - \alpha e_1 w \\ -w & z & e_4 z - e_3 w \end{vmatrix}$$

contains the term $\alpha^2 y^2 z$. Thus $f(\mathbf{X}) \neq 0$.

Secondly, for any parameter values (s_0, t_0) , let $\mathbf{X}_0 = (a(s_0, t_0), b(s_0, t_0), c(s_0, t_0), d(s_0, t_0))$. We want to show that $f(\mathbf{X}_0) = 0$. From $L_i(\mathbf{X}_0, s_0, t_0) = 0, i = 1, 2, 3$, we have

$$\begin{pmatrix} L_{11}(\mathbf{X}_0) & L_{12}(\mathbf{X}_0) & L_{13}(\mathbf{X}_0) \\ L_{21}(\mathbf{X}_0) & L_{22}(\mathbf{X}_0) & L_{23}(\mathbf{X}_0) \\ L_{31}(\mathbf{X}_0) & L_{32}(\mathbf{X}_0) & L_{33}(\mathbf{X}_0) \end{pmatrix} \begin{pmatrix} s_0 \\ t_0 \\ 1 \end{pmatrix} = \mathbf{0}. \quad (10)$$

Therefore,

$$f(\mathbf{X}_0) = 0.$$

Since the degree of $f(\mathbf{X})$ is three, we can conclude that $f(\mathbf{X}) = 0$ is the implicit form of $\mathbf{P}(s, t)$. $\square$

REMARK 3.2. *In [4], the authors proved that the moving planes and moving quadrics method works for general rational surfaces in the presence of base points if the rational surfaces satisfy certain conditions (BP1-BP5). It can be shown*

that a QSOB satisfies the BP1-PB5 conditions. Thus Theorem 3.2 can be looked as a special case of Theorem 3.6 in the paper [4].

Based on the results of Theorem 3.1 and 3.2, a simple algorithm can be devised to implicitize a QSOB.

Algorithm IMPLICITIZATION-QSOB

Input: The parametric equation of a QSOB.

Output: The implicit equation of the QSOB.

Procedure:

1. Solve the linear system of equations (5) to obtain the weak μ -basis of the QSOB.
2. Compute the determinant of the matrix (7) formed by the weak μ -basis. The determinant gives the implicit equation of the QSOB.

REMARK 3.3. To improve the efficiency, one may first perform row reductions on the matrix $(\mathbf{P}_1, \mathbf{P}_s, \mathbf{P}_t, \mathbf{P}_{s^2}, \mathbf{P}_{st})$ and transform it into the simplest form, i.e., four of the column vectors are transformed into $(1, 0, 0, 0)^T$, $(0, 1, 0, 0)^T$, $(0, 0, 1, 0)^T$ and $(0, 0, 0, 1)^T$. Then we compute the weak μ -basis and the implicit equation of the corresponding surface $\tilde{\mathbf{P}}(s, t)$ after the transformation. The implicit equation of the surface $\mathbf{P}(s, t)$ is obtained by the same transformation of the variables from the implicit equation of $\tilde{\mathbf{P}}(s, t)$.

We illustrate the algorithm with an example.

Example 1. Let the quadratic parametric surface be

$$\mathbf{P}(s, t) = (s, t^2, st - 3, s - st)$$

with $(1, 0, 0)$ being a base point. A weak μ -basis of $\mathbf{P}(s, t)$ is

$$(L_1, L_2, L_3) = (s, t, 1) \begin{pmatrix} 0 & x - z - w & y \\ -x & -3x & -x + w \\ x - w & -3w & 0 \end{pmatrix}.$$

Then the implicit equation of $\mathbf{P}(s, t)$ is

$$\begin{aligned} f(\mathbf{X}) &:= \begin{vmatrix} 0 & x - z - w & y \\ -x & -3x & -x + w \\ x - w & -3w & 0 \end{vmatrix} \\ &= -x^3 + 3x^2y + x^2z + 3x^2w - 2xzw - 3xw^2 \\ &\quad + zw^2 + w^3 = 0. \end{aligned}$$

The weak μ -basis also provides a simple inversion formula for a QSOB. Suppose $\mathbf{X}_0$ is a point on a QSOB, the corresponding parameter values (s_0, t_0) can be solved from (10):

$$\begin{cases} s_0 = \frac{L_{12}(\mathbf{X}_0)L_{23}(\mathbf{X}_0) - L_{13}(\mathbf{X}_0)L_{22}(\mathbf{X}_0)}{L_{11}(\mathbf{X}_0)L_{22}(\mathbf{X}_0) - L_{12}(\mathbf{X}_0)L_{21}(\mathbf{X}_0)}, \\ t_0 = \frac{L_{13}(\mathbf{X}_0)L_{21}(\mathbf{X}_0) - L_{11}(\mathbf{X}_0)L_{23}(\mathbf{X}_0)}{L_{11}(\mathbf{X}_0)L_{22}(\mathbf{X}_0) - L_{12}(\mathbf{X}_0)L_{21}(\mathbf{X}_0)}. \end{cases} \quad (11)$$

For the QSOB in Example 1, the inversion formula is

$$s = \frac{3x}{x - z - w}, \quad t = \frac{x - w}{x}.$$

The weak μ -basis has some nice algebraic properties. For example, one has

THEOREM 3.3. Let $L_i(\mathbf{X}, s, t)$, $i = 1, 2, 3$ be a weak μ -basis of a QSOB $\mathbf{P}(s, t)$, and $f(\mathbf{X}) = 0$ be the implicit equation of $\mathbf{P}(s, t)$. Then

$$\langle L_1, L_2, L_3 \rangle \cap R[x, y, z, w] = \langle f \rangle.$$

Here $R[x, y, z, w]$ is the ring of polynomials in variables x, y, z, w over the field F .

PROOF. Let $(h_1, h_2, h_3) = (L_{12}, L_{22}, L_{32}) \times (L_{13}, L_{23}, L_{33})$. By Theorem 3.2, it is easy to show that $f = h_1L_1 + h_2L_2 + h_3L_3$. Hence $f \in \langle L_1, L_2, L_3 \rangle$.

On the other hand, for any $g(x, y, z, w) \in \langle L_1, L_2, L_3 \rangle \cap R[x, y, z, w]$, there exist polynomials h_i , $i = 1, 2, 3$ such that

$$g = h_1L_1 + h_2L_2 + h_3L_3.$$

Since L_i , $i = 1, 2, 3$ are moving planes following $\mathbf{P}(s, t)$, $g(a, b, c, d) \equiv 0$, i.e., g vanishes on $\mathbf{P}(s, t)$. Since f also vanishes on $\mathbf{P}(s, t)$ and it is irreducible, $f|g$. Hence $g \in \langle f \rangle$. This completes the proof. $\square$

4. PARAMETRIZATION

A QSOB represents a cubic surface in implicit form. Compared to non-singular cubic surfaces parameterized by cubic parametric surfaces with six generic base points, it has some special properties.

4.1 Singular locus of a QSOB

We first give two simple lemmas regarding the singular points of a cubic surface.

LEMMA 4.1. Let P_1 and P_2 be two singular points of a cubic surface. Then the line joining the two singular points is contained in the surface.

PROOF. Notice that any line intersects with a cubic surface at three points if the line is not contained in the cubic surface. Since the line joining P_1 and P_2 intersects with the cubic surface at four points (counting multiplicity), the line must be contained in the surface. $\square$

LEMMA 4.2. An irreducible cubic surface either has a self-intersection line, or has at most four singular points.

PROOF. If the cubic surface contains a non-planar space curve as its singular locus, then the line joining any two points on the space curve is contained in the surface by Lemma (4.1). Consequently some three-dimensional set is contained in the surface, an impossibility.

If the cubic surface contains a plane curve (not a line) as its locus, then the plane containing the plane curve is a part of the surface. This is impossible since the cubic surface is irreducible.

If the cubic surface contains a line l and a point P not on the line as its singular locus, then the plane determined by the point P and the line l is also contained in the surface, again a contradiction with the irreducibility of the cubic surface.

Finally, if the surface has only finite number of singular points, it has at most four singular points by Proposition 2.1. $\square$

THEOREM 4.3. [10] The cubic surface parametrized by a QSOB is a singular surface. The singular locus is a self-intersection line.

PROOF. Coffman et al. ([10]) provide a proof based on matrix algebra. In this paper, we give a simple proof by using the weak μ -basis of the QSOB.

Without loss of generality, we assume that the quadratic parametric surface has the simplified form (9). Let $\tilde{M}$ be the matrix formed by the weak μ -basis of $\tilde{\mathbf{P}}(s, t)$, i.e.,

$$\tilde{M} := \begin{pmatrix} -\gamma x + \alpha y & 0 & \beta z - \alpha w \\ -\beta x & \alpha x & -\alpha y + \beta e_1 z - \alpha e_1 w \\ -w & z & e_4 z - e_3 w \end{pmatrix}.$$

It is easy to show that $\text{rank}(\tilde{M}) = 1$ if and only if $\beta z - \alpha w = 0$ and $-\gamma x + \alpha y = 0$. That is, for any point (x, y, z, w) satisfying the two equations, it corresponds to at least two parameters, and thus corresponds to a singular point. So the intersection line of two planes $\beta z - \alpha w = 0$ and $-\gamma x + \alpha y = 0$ is a double line of $\mathbf{P}(s, t)$. By Lemma 4.2, there are no other singular points on $\mathbf{P}(s, t)$. $\square$

Based on the proof of Theorem 4.3, a simple method is derived to compute the self-intersection line of a QSOB parameterized by $\mathbf{P}(s, t)$.

Algorithm SELF-INTERSECTION-QSOB-I

Input: The parametric equation $\mathbf{P}(s, t)$ of a QSOB.

Output: The self-intersection line of the QSOB.

Procedure:

1. Compute the base point of $\mathbf{P}(s, t)$, and change the parametric equation into the form (4).
2. Compute a weak μ -basis $L_i(\mathbf{X}) = L_{i1}(\mathbf{X})s + L_{i2}(\mathbf{X})t + L_{i3}(\mathbf{X})$, $i = 1, 2, 3$ of $\mathbf{P}(s, t)$.
3. Perform row operations to the matrix M as defined in (7), such that one element of the second column is zero (say $L_{12}(\mathbf{X}) = 0$). This can happen because $L_{i2}(\mathbf{P}_t) = 0$ and $L_{i2}(\mathbf{P}_{st}) = 0$, $i = 1, 2, 3$. We still use the same notation for the matrix after row reduction.
4. The intersection of the two planes $L_{11}(\mathbf{X}) = 0$ and $L_{13}(\mathbf{X}) = 0$ is the self-intersection line of $\mathbf{P}(s, t)$.

For the quadratic parametric surface in Example 1, the self-intersection line is defined by $x - w = 0$ and $x = 0$, or equivalently $x = 0$ and $w = 0$.

For a QSOB defined by an implicit equation $f(x, y, z, w)$, how does one compute its self-intersection line? A standard way is to solve the system of equations:

$$\begin{aligned} f_x(x, y, z, w) &= 0, & f_y(x, y, z, w) &= 0, \\ f_z(x, y, z, w) &= 0, & f_w(x, y, z, w) &= 0. \end{aligned} \quad (12)$$

Or equivalently, to compute the intersection of four quadric surfaces. Computing the intersection of two quadric surfaces is not an easy task (see [22] and the references therein). In the following, we provide a method to detect if an irreducible cubic surface has a self-intersection line, and if it has, how to compute the self-intersection line.

We consider the intersection of the cubic surface with an arbitrary plane. First, we have the following lemma.

LEMMA 4.4. *Let P be a singular point of a cubic surface, and π be a plane passing through P . Then P is also a singular point of the curve which is the intersection of the cubic surface and the plane π .*

PROOF. Without loss of generality, we may assume that the plane π is defined by the equation $z = p_1x + p_2y + p_3w$. Then the intersection curve of the plane π and the cubic surface $f(x, y, z, w) = 0$ is defined by $g(x, y, w) := f(x, y, p_1x + p_2y + p_3w, w) = 0$. If P is a singular point of the cubic surface, then $f_x(P) = f_y(P) = f_z(P) = f_w(P) = 0$. So $g_x(P) = f_x(P) + f_z(P)p_1 = 0$. Similarly, $g_y(P) = f_y(P) + f_z(P)p_2 = 0$. Thus P is also a singular point of $g(x, y, w) = 0$. $\square$

Note that the converse of the above lemma is not true.

Now based on previous analysis, we can devise an algorithm to detect and compute the self-intersection line of a cubic surface.

Algorithm SELF-INTERSECTION-QSOB-II

Input: Implicit equation of an irreducible cubic surface.

Output: Detect if the cubic surface has a self-intersection line, and if yes, compute it.

Procedure:

1. Choose five random planes, and compute the intersection curves of the cubic surface with the five planes.
2. Compute the singular point (if any) of each intersection curve by the method in [7].
3. Verify if the singular points of the intersection curves are singular points of the cubic surface;
4. If we obtain 5 or more singular points of the cubic surface, then these singular points are all on a single line, and the line is the self-intersection line of the cubic surface. Otherwise, the cubic surface doesn't have a self-intersection line.

REMARK 4.1. *To detect if a cubic surface is a QSOB or not, one can simply find a random plane to intersect the surface, and check if the intersect curve contains a singular point or not.*

4.2 Parametrization method

Suppose we are given the implicit equation $f(x, y, z, w) = 0$ of a cubic surface parameterized by a QSOB. After we have computed the self-intersection line, a direct method to parameterize the surface is as follows. We find an arbitrary plane passing through the self-intersection line, and the plane intersects the cubic surface at a line l' besides the self-intersection line. The line l' gives a parametrization of the cubic surface. Unfortunately, such a parametrization is in general cubic instead of quadratic. To get a quadratic parametrization, we would like to seek the aid of a weak μ -basis.

To begin with, we choose two distinct planes $p_1 = 0, p_2 = 0$ passing through the self-intersection line l of the cubic surface. The plane $p_i = 0$ intersects with the cubic surface at l (a double line) and another line l_i , $i = 1, 2$. Thus, if we solve for one variable (say z) from $p_1 = 0$ to get $z =$

$z_1(x, y, w)$, and substitute it into $f(x, y, z, w)$, then one gets $f(x, y, z_1(x, y, w), w) = m^2 n_1$. Here m, n_1 are linear functions in x, y, w . Similarly, $f(x, y, z_2(x, y, w), w) = m^2 n_2$. Here $z_2(x, y, w)$ is solved from $p_2 = 0$ and n_2 is a linear function in x, y, w .

THEOREM 4.5. *Let the notation be as above. Then the implicit equation of the cubic surface parameterized by a QSOB can be written as $f(x, y, z, w) = \det(G)$, where*

$$G = \begin{pmatrix} p_1 & 0 & m \\ 0 & p_2 & m \\ -k_1 n_1 & -k_2 n_2 & h_0(x, y, z, w) \end{pmatrix}, \quad (13)$$

k_1, k_2 are undetermined coefficients, and $h_0(x, y, z, w)$ is a undetermined linear function.

PROOF. The proof is similar to the proof of Theorem 5 in [2]. We sketch it as follows.

Since $f(x, y, z, w)$ vanishes on the self-intersection line l and the other two lines l_1, l_2 , $f(x, y, z, w)$ must be contained in the radical ideal

$$J = \langle p_1, p_2 \rangle \cap \langle p_1, n_1 \rangle \cap \langle p_2, n_2 \rangle.$$

It is easy to show that

$$J = \langle p_1 p_2, p_1 n_2, p_2 n_1 \rangle.$$

So $f(x, y, z, w)$ can be expressed as

$$h_0 p_1 p_2 + h_1 p_1 n_2 + h_2 p_2 n_1,$$

where h_0, h_1, h_2 are homogeneous polynomials in $R[x, y, z, w]$.

Next from $f(x, y, z, w) \in \langle p_1, m \rangle \cap \langle p_1, n_1 \rangle$, one obtains that $m|h_2$. Similarly, we can get that $m|h_1$. Thus

$$f(x, y, z, w) = h_0 p_1 p_2 + k_1 p_2 m n_1 + k_2 p_1 m n_2,$$

where h_0 is a linear polynomial in x, y, z, w . Thus $f(x, y, z, w)$ can be expressed as the required form (13). $\square$

In Eq.(13), the unknown coefficients k_1, k_2 and the unknown linear function h_0 can be solved by undetermined coefficients. Thus the implicit equation of a QSOB can be written as the determinant of a three by three matrix in which the entries are linear with x, y, z, w . The three by three matrix provides us a method to parameterize the cubic surface.

Let

$$(L_1, L_2, L_3) := (s, t, 1) \cdot G, \quad (14)$$

then (L_1, L_2, L_3) can be rewritten as

$$(L_1, L_2, L_3) = (x, y, z, w) \cdot H, \quad (15)$$

where $H = (L_{ij}(s, t))_{4 \times 3}$ with $L_{ij}(s, t) \in R[s, t]$. L_1, L_2, L_3 will serve as the weak μ -basis of the parameterization of the cubic surface $f(x, y, z, w) = 0$.

THEOREM 4.6. *Let $L_{ij}(s, t)$ be given as above, and let $\mathbf{L}_i = (L_{1i}, L_{2i}, L_{3i}, L_{4i})^T$, $i = 1, 2, 3$. Then*

$$\mathbf{P}(s, t) := [\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3]$$

gives a quadratic parametrization of the cubic surface $f(x, y, z, w) = 0$. Here $[\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3]$ is the outer product of $\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3$,

i.e.,

$$[\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3] := \begin{pmatrix} \begin{vmatrix} L_{21} & L_{31} & L_{41} \\ L_{22} & L_{32} & L_{42} \\ L_{23} & L_{33} & L_{43} \end{vmatrix}, -\begin{vmatrix} L_{11} & L_{31} & L_{41} \\ L_{12} & L_{32} & L_{42} \\ L_{13} & L_{33} & L_{43} \end{vmatrix}, \\ \begin{vmatrix} L_{11} & L_{21} & L_{41} \\ L_{12} & L_{22} & L_{42} \\ L_{13} & L_{23} & L_{43} \end{vmatrix}, -\begin{vmatrix} L_{11} & L_{21} & L_{31} \\ L_{12} & L_{22} & L_{32} \\ L_{13} & L_{23} & L_{33} \end{vmatrix} \end{pmatrix}. \quad (16)$$

PROOF. Without loss of generality, we assume that the self-intersection line l coincides with x -axis, i.e., l is defined by $y = z = 0$.

We first prove that $\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3$ are $R[s, t]$ -linearly independent, i.e., the outer product of $\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3$ is not a zero vector.

Consider two planes $p_1 := y - \lambda_1 z = 0$ and $p_2 := y - \lambda_2 z = 0$ passing through x axis, here $\lambda_1 \neq \lambda_2$. Then $f(x, y, z, w) = \det(G)$ with

$$G = \begin{pmatrix} y - \lambda_1 z & 0 & z \\ 0 & y - \lambda_2 z & z \\ h_1 & h_2 & h_3 \end{pmatrix}.$$

where $h_1 = h_{11}x + h_{13}z + h_{14}w$, $h_2 = h_{21}x + h_{23}z + h_{24}w$, $h_3 = h_{31}x + h_{32}y + h_{33}z + h_{34}w$, and h_{ij} , $i = 1, 2, 3$, $j = 1, 2, 3, 4$ are constants. Furthermore, at least one of h_{11}, h_{21}, h_{31} is nonzero.

From (14) and (15), we get

$$H = \begin{pmatrix} h_{11} & h_{21} & h_{31} \\ s & t & h_{32} \\ -\lambda_1 s + h_{13} & -\lambda_2 t + h_{23} & s + t + h_{33} \\ h_{14} & h_{24} & h_{34} \end{pmatrix}.$$

Thus

$$\mathbf{L}_1 = (h_{11}, s, -\lambda_1 s + h_{13}, h_{14}),$$

$$\mathbf{L}_2 = (h_{21}, t, -\lambda_2 t + h_{23}, h_{24}),$$

$$\mathbf{L}_3 = (h_{31}, h_{32}, s + t + h_{33}, h_{34}).$$

One can verify directly that $[\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3]$ can't be identically the zero vector (otherwise $h_{11} = h_{21} = h_{31} = h_{14} = h_{24} = h_{34} = 0$, which implies that f involves only y, z , an impossibility), and it thus gives a parametrization of the cubic surface. Furthermore, it is easy to see that the total degree of the parametrization is two. This completes the proof. $\square$

Based on the above theorem, we can devise an algorithm to compute a quadratic parametrization of a cubic surface with a self-intersection line.

Algorithm PARAMETRIZATION-QSOB

Input: The implicit equation $f(x, y, z, w) = 0$ of a QSOB.

Output: The parametric equation of the QSOB.

Procedure:

1. Detect whether the cubic surface $f(x, y, z, w) = 0$ has a self-intersection line or not by the algorithm SELF-INTERSECTION-QSOB-II. If there exists a self-intersection line, compute it and denote it as l . Otherwise, Stop.
2. Choose any two distinct planes p_1, p_2 passing through the line l . Substituting the equation of each plane

into $f(x, y, z, w) = 0$, we will get a factorization as $m^2 n_1$ and $m^2 n_2$ respectively. Here m, n_1, n_2 are linear functions in x, y, z, w .

3. Solve for unknown coefficients $k_1, k_2, k_3, k_4, k_5, k_6$ from

$$f(x, y, z, w) = \begin{vmatrix} p_1 & 0 & m \\ 0 & p_2 & m \\ k_1 n_1 & k_2 n_2 & k_3 x + k_4 y + k_5 z + k_6 w \end{vmatrix}$$

4. Construct the weak μ -basis as in the form (15). Compute the outer product of the weak μ -basis. The result is a quadratic parametrization of the cubic surface $f(x, y, z, w) = 0$.

We give an example to illustrate the above algorithm.

Example 2. The implicit equation of $\mathbf{P}(s, t)$ in Example 1 is

$$f(x, y, z, w) := -x^3 + 3x^2 y + x^2 z + 3x^2 w - 2xzw - 3xw^2 + zw^2 + w^3 = 0.$$

and the singular locus of the cubic surface is $(x, y, z, w) = (0, y, z, 0)$. Choosing the planes $x - w = 0$, $x = 0$ passing through the self-intersection line, we can get the following factorization of $f(x, y, z, w)$:

$$f(x, y, z, w) = \begin{vmatrix} x - w & 0 & w \\ 0 & x & w \\ -3y & z + w & -x + 3y + z + 2w \end{vmatrix}.$$

The weak μ -basis is thus

$$(sx - sw - 3y, tx + z + w, (s + t)w - x + 3y + z + 2w),$$

or in vector form

$$\mathbf{L}_1 = (s, -3, 0, -s), \quad \mathbf{L}_2 = (t, 0, 1, 1), \\ \mathbf{L}_3 = (-1, 3, 1, s + t + 2).$$

Thus the outer product of $\mathbf{L}_1, \mathbf{L}_2, \mathbf{L}_3$ gives a parametrization of the cubic surface:

$$\mathbf{P}(s, t) = (-3t - 3, -s^2, 3t^2 - 3s + 6t + 3, 3s - 3t - 3).$$

Example 3. Consider the cubic surface

$$f(x, y, z, w) := xz^2 + y^2 z + 2y^2 w + yzw = 0.$$

The self-intersection line of $f = 0$ is $(x, y, z, w) = (x, 0, 0, w)$. Choosing two planes $y - z = 0$, $y = 0$ passing through the self-intersection line, we obtain

$$f(x, y, z, w) = \begin{vmatrix} y - z & 0 & z \\ 0 & y & z \\ -x - z - 3w & x & z + 2w \end{vmatrix}.$$

The weak μ -basis is

$$\mathbf{L}_1 = (-1, s, -s - 1, -3), \quad \mathbf{L}_2 = (1, t, 0, 0), \\ \mathbf{L}_3 = (0, 0, s + t + 1, 2).$$

Thus the parametrization of the cubic surface is given by

$$\mathbf{P}(s, t) = (-st - 3t^2 - t, s + 3t + 1, -2s - 2t, s^2 + 2st + t^2 + s + t).$$

5. CONCLUSION

In this paper, we have unified the implicitization and parametrization of a quadratic surface with one simple base point (QSOB) by using the weak μ -basis of the QSOB. Simple and efficient algorithms are derived to realize the conversion between implicit form and parametric form. We also provide a method to detect and compute the self-intersection line of a QSOB, either in parametric form or in implicit form. The singular locus of the QSOB is important for the computation of the weak μ -basis of a cubic surface derived from a QSOB. Examples are provided to illustrate the algorithms.

It would be interesting to generalize the moving plane method to handle the conversion between implicit form and parametric form of higher degree surfaces.

6. ACKNOWLEDGMENTS

This work is supported by A National Key Basic Research Project of China (No. 2004CB318000), NSF of China (No. 10671192, 60225002 and 60533060), One hundred Talent Project supported by CAS, Specialized Research Fund for the Doctoral Program of Higher Education (No. 20060358055), the 111 Project (No. b07033).

7. REFERENCES

- [1] C. Bajaj, R. Holt, and A. Netravali, Rational parametrizations of nonsingular real cubic surfaces, ACM Transactions on Graphics, Vol.17, 1–31, 1998.
- [2] T. G. Berry and R. R. Patterson, Implicitization and parametrization of nonsingular cubic surfaces, Computer Aided Geometric Design, Vol.18, 723–738, 2001.
- [3] B. Buchberger, Applications of Groebner bases in nonlinear computational geometry, In Kapur, D. and Mundy, J., (eds), Geometric Reasoning, Elsevier Science Publisher, MIT Press, 413–446, 1989.
- [4] L. Busé, D. Cox, and C. D'Andrea, Implicitization of surfaces in $\mathbb{P}^3$ in the presence of base points, Journal of Algebra and Its Applications, Vol.2, 189–214, 2003.
- [5] F. Chen, J. Zheng, and T. W. Sederberg, The μ -basis of a rational ruled surface, Computer Aided Geometric Design, Vol.18, 61–72, 2001.
- [6] F. Chen and W. Wang, Revisiting the μ -basis of a rational ruled surface, Journal of Symbolic Computation, Vol.36, No.5, 699–716, 2003.
- [7] F. Chen and W. Wang, Computing real inflection points of cubic algebraic curves, Computer Aided Geometric Design, Vol.20, 101–117, 2003.
- [8] F. Chen, D. Cox, and Y. Liu, The μ -basis and implicitization of a rational parametric surface, Journal of Symbolic Computation, Vol.39, 689–706, 2005.
- [9] E. Chionh and R. Goldman, Using multivariate resultants to find the implicit equation of a rational surface, The visual Computer: International Journal of Computer Graphics, Vol.8, 171–180, 1992.
- [10] A. Coffman, A.J. Schwartz, and C. Stanton, The algebra and geometry of Steiner and other quadratically parametrizable surfaces, Computer Aided Geometric Design, Vol.13, 257–286, 1996.

- [11] D. A. Cox, D. Little, and D. O'Shea, *Ideals, Varieties, and Algorithms — An Introduction to Computational Algebraic Geometry and Commutative Algebra*, 2nd ed., Springer, 2005.
- [12] D. A. Cox, T. W. Sederberg, and F. Chen, The moving line ideal basis of planar rational curves, *Computer Aided Geometric Design*, Vol.15, 803–827, 1998.
- [13] J. Deng, F. Chen, and L. Shen, Computing μ -bases of rational curves and surfaces using polynomial matrix factorization, *Proceedings of the ISSAC'2005*, Manuel Kauers ed., ACM Press, USA, 132–139, 2005.
- [14] B. Hunt, *The geometry of some special arithmetic quotients*, *Lecture Notes in Mathematics* 1637, Springer, 1996.
- [15] P. Irene, *Theory and history of geometric models*, Academic Press Europe, 41–42, 2007.
- [16] T. W. Sederberg, D. Anderson, and R. Goldman, Implicit representation of parametric curves and surfaces, *Computer Vision, Graphics, and Image Processing*, Vol.28, 72–84, 1984.
- [17] T. W. Sederberg and F. Chen, Implicitization using moving curves and surfaces, *Proceedings of Siggraph'1995*, 301–308, 1995.
- [18] T. W. Sederberg, Techniques for cubic algebraic surfaces, Part 2. *IEEE Computer Graphics and Applications*, Vol.10, No.5, 12–21, 1990.
- [19] J. Semple and L. Roth, *Introduction to Algebraic Geometry*, Clarendon Press, Oxford, Reprint of the 1949 original, 1985.
- [20] J. Schicho, Rational parametrization of real algebraic surfaces, *ISSAC'98: Proceedings of the 1998 international symposium on Symbolic and algebraic computation*, ACM Press, 302–308, 1998.
- [21] L. Shen, F. Chen, and J. Deng, Implicitization and parametrization of quadric and cubic surfaces by μ -bases, *Computing*, Vol.5, 131–142, 2006.
- [22] W. Wang, B. Joe, and R. Goldman, Computing quadric surface intersections based on an analysis of plane cubic curves, *Graphical Models*, Vol.64, 335–367, 2002.

Solving Differential Equations in Terms of Bessel Functions

Ruben Debeerst
Universität Kassel
Kassel, Germany
debeerst@mathematik.uni-kassel.de

Mark van Hoeij*
Florida State University
Tallahassee, FL, USA
hoeij@math.fsu.edu

Wolfram Koepf
Universität Kassel
Kassel, Germany
koepf@mathematik.uni-kassel.de

ABSTRACT

For differential operators of order 2, this paper presents a new method that combines generalized exponents to find those solutions that can be represented in terms of Bessel functions.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms—*Algebraic algorithms*; G.4 [Mathematical Software]: Algorithm design and analysis

General Terms

Algorithms

1. INTRODUCTION

Consider a differential operator $L = \sum_{i=0}^n a_i \partial^i$ with coefficients in some differential field K and $\partial = \frac{d}{dx}$. We search for solutions of $L(y) = 0$. In a first step, if $K = \mathbb{C}(x)$, then one can try to factor [2, 9, 10] the differential operator. From then on we will only consider irreducible $L \in K[\partial]$.

If B is a special function that satisfies a differential equation $L_B(B) = 0$, then the question if we can solve $L(y) = 0$ in terms of B is equivalent to the question whether there exist certain transformations that send L_B to L .

There are three types of transformations in $K[\partial]$ that preserve order two, namely: (i) change of variables¹ $x \rightarrow f(x)$, (ii) an exp-product $y \rightarrow \exp(\int r) \cdot y$, and (iii) a gauge² transformation $y \rightarrow r_0 y + r_1 y'$, where $'$ is $\frac{d}{dx}$. We take the parameters f, r, r_0, r_1 in K so that the result is again in $K[\partial]$.

Our algorithm can decide if an operator L can be obtained from the Bessel operator under these three transformations.

*Supported by NSF grants 0511544 and 0728853.

¹This always refers to a change of the *independent* variable.

²Terminology from [12, Definition 8].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Thus, given L , it can find all solutions of the form

$$\exp\left(\int r dx\right) (r_0 B_\nu(f(x)) + r_1 B_\nu(f(x))'), \quad (1)$$

where $r, r_0, r_1, f \in \mathbb{C}(x)$ and $B_\nu(x)$ is a Bessel function.

One could argue that this is only a minor addition to prior work. Prior algorithms treat more than just Bessel functions, and they also already treat two of these transformations; [3, 4, 5, 11, 15] treat (i)+(ii), and [1, 14] treat (ii)+(iii). However, we argue that unless all three transformations are covered simultaneously, the work can not be considered to be complete: take equations that the algorithm can solve, and then apply the non-treated transformation, and one encounters solvable equations that the algorithm does not solve.

In contrast, the set of equations solvable by our algorithm is closed under all three transformations. This closure property means that if L is an operator that the algorithm can solve, and if one applies the above “order preserving transformations”, then the result is again an operator that the algorithm can solve. This closure property is the key novelty in our algorithm. It requires (main task in this paper) solving a combinatorial problem introduced in Section 3.

Prior papers do not have this combinatorial problem; if one does not treat (iii) then there exists a *global* invariant [3, 4, 5, 15] that fully determines f and there is no need to combine *local* invariants such as (generalized) exponents. The papers [1, 14] treat (ii)+(iii) but not (i). Adding a restricted version of (i) to this (with f restricted to Möbius transformations $\frac{ax+b}{cx+d}$) requires only a little bit of extra code because such f is determined by 3 points that are easily obtained from L without combining (generalized) exponents. However, adding (i), for general f , to (ii)+(iii) is much more work; it is the above mentioned combinatorial problem and takes up most of our implementation.

In summary: Treating (i)+(ii)+(iii) simultaneously, for arbitrary $r, r_0, r_1, f \in \mathbb{C}(x)$, is the new result in this paper. Handling two of these transformations has already been done before (and: for more than just Bessel functions).

Our method can also be adapted to work for other special functions; our implementation in *Maple* can solve differential operator in terms of Bessel, Whittaker and Kummer functions. Due to page limitations we only treat the Bessel case here; the first author’s master thesis [7] explains the algorithm in more detail including the Whittaker case³. Two

³The master thesis and the *Maple* implementation can be obtained from <http://www.mathematik.uni-kassel.de/~debeerst/master/>.

important cases that are not yet treated are: the ${}_2F_1$ special function (for general f), and the Bessel case where f is not a rational function but the square root of a rational function. The combinatorial problem is more difficult for these cases.

2. PRELIMINARIES

2.1 Differential Operators

We denote by $K[\partial]$ the ring of differential operators with coefficients in K . Mostly we have $K = k(x)$ but sometimes we will also need power series coefficients $K = k((x))$. Here k will be $\mathbb{C}$ until Section 4.2 which will treat non-algebraically closed k .

A point $p \in \mathbb{C} \cup \{\infty\}$ is called a singular point of $L \in K[\partial]$ if p is a zero of the leading coefficient or a pole of one of the other coefficients. Otherwise, p is called regular.

By the solutions of L we mean the solutions of the differential equation $L(y) = 0$. If p is regular, we can express all solutions locally around p as power series $\sum_{i=0}^{\infty} b_i t_p^i$ where t_p denotes the local parameter which is $t_p = \frac{1}{x}$ if $p = \infty$ and $t_p = x - p$ otherwise.

2.2 Formal Solutions and Generalized Exponents

Definition 1. A universal extension U of $K = \mathbb{C}((x))$ is a minimal differential ring in which every operator $L \in K[\partial]$ has precisely $\deg(L)$ $\mathbb{C}$ -linear independent solutions.

We denote the solution space of L by $V(L) := \{y \in U \mid L(y) = 0\}$. There exists a universal extension U of $\mathbb{C}((x))$, so $V(L)$ has dimension $\deg(L)$ for every nonzero operator $L \in \mathbb{C}((x))[\partial]$. For the construction of U we refer to [13], Chapter 3.2, where U is denoted by $\text{UnivR}_{\widehat{K}}$.

Most importantly we know that the fundamental system of local solutions at $x = 0$ can be represented as

$$\exp\left(\sum_{i=1}^n c_i x^{-1/m}\right) x^\lambda S$$

for some $c_i, \lambda \in \mathbb{C}, n, m \in \mathbb{N}$ and where $S \in \mathbb{C}[[x^{1/m}]] \log(x)$ has a nonzero constant term. This can also be written as $\exp(\int e/x dx) S$ where $e \in E = \bigcup_{m \in \mathbb{N}} E_m$ and $E_m = \mathbb{C}[x^{-1/m}]$. If $e \in \mathbb{C}$, we get a solution $x^e S$ and e is called an *exponent*. For $e \in \mathbb{N}$ we get power series solutions. In general $e \in E$ is called *generalized exponent* and can depend on x . Solutions that involve a logarithm will be called *logarithmic solutions*.

This construction can be done at any point p , for which we just have to replace x by the local parameter t_p . Then a local solution at p has the representation

$$\exp\left(\int \frac{e}{t_p} dt_p\right) S, \quad e \in \mathbb{C}[t_p^{-\frac{1}{m}}], \quad S \in \mathbb{C}[[t_p^{\frac{1}{m}}]] \log(t_p) \quad (2)$$

If $m = 1$ (when no fractional powers of t_p occur in e and S) then e is called *unramified*.

At any point p there are n generalized exponents $e_1, \dots, e_n$ corresponding to a basis $\exp(\int e_i/t_p) S_i(p)$, $i = 1, \dots, n$ of $V(L)$. These generalized exponents can be computed [9].

2.3 Bessel Functions

The solutions of the operators $L_1 = x^2 \partial^2 + x \partial + (x^2 - \nu^2)$ and $L_2 = x^2 \partial^2 + x \partial - (x^2 + \nu^2)$ are called Bessel functions. The two linearly independent solutions $J_\nu(x)$ and $Y_\nu(x)$ of

L_1 are called Bessel functions of first and second kind, respectively. Similarly the solutions $I_\nu(x)$ and $K_\nu(x)$ of L_2 are called the modified Bessel functions of first and second kind.

The Bessel functions with parameter $\nu \in \frac{1}{2} + \mathbb{Z}$ are hyperexponential functions and in that case L_1 and L_2 are reducible. Since we only consider irreducible operators, we can assume $\nu \notin \frac{1}{2} + \mathbb{Z}$.

The transformation $x \rightarrow \sqrt{-1} \cdot x$ sends L_1 to L_2 and vice versa. Since we will allow such transformations later, we only have to deal with one of the two cases. We choose the modified Bessel case and we denote $L_B := L_2$.

For Bessel functions, the generalized exponents are unramified (i.e. $m = 1$ in the previous section) so no fractional exponents (Puiseux series) are needed. The operator L_B has generalized exponents $\pm \nu$ at 0 and $\pm t_\infty^{-1} + \frac{1}{2}$ at ∞ .

3. TRANSFORMATIONS

From now on we will restrict ourselves to irreducible operators of degree two (so the formula $r_0 y + r_1 y'$ in item (iii) below describes any K -linear combination of $y, y', y'', \dots$).

Definition 2. Let $K = k(x)$. A transformation between two differential operators $L_1, L_2 \in K[\partial]$ is a map from the solution space $V(L_1)$ onto the solution space $V(L_2)$. We will address the following transformations:

- (i) **change of variables** $y(x) \rightarrow y(f(x)), \quad f(x) \in K \setminus k$.
- (ii) **exp-product** $y \rightarrow \exp(\int r dx) \cdot y, \quad r \in K$.
- (iii) **gauge transformation** $y \rightarrow r_0 y + r_1 y', \quad r_0, r_1 \in K$.

For the resulting operator $L_2 \in K[\partial]$ we write $L_1 \xrightarrow{f} L_2$, $L_1 \xrightarrow{r} L_2$, and $L_1 \xrightarrow{r_0, r_1} L_2$, respectively. Furthermore, we write $L_1 \longrightarrow L_2$ if there exists a sequence of transformations that sends L_1 to L_2 .

The rational functions f, r, r_0 and r_1 will be called parameters of the transformation, and in case (ii) the function $\exp(\int r)$ is a hyperexponential function.

Lemma 1 *Let $L_1 \in K[\partial]$ be irreducible of degree two. If the parameters of the transformations above are given, we can always find $L_2 \in K[\partial]$ with $\deg(L_2) = 2$ such that $V(L_1)$ is mapped onto $V(L_2)$ by the given transformations.*

PROOF. If $y \in V(L_1)$ is mapped to $z \in V(L_2)$, then z, z' and z'' can be rewritten in terms of y and y' using L_1 . An ansatz for L_2 yields a system of two equations with three variables (see [7, Theorem 2.4] for details). $\square$

It is clear that if $L_1 \longrightarrow L_2$ and $L_1 = L_B$, the solutions of L_2 can be expressed by Bessel functions. Conversely, from examples studied so far, it appears that any operator $L \in K[\partial]$ that has Bessel solutions satisfies $L_B \longrightarrow L$ provided that we generalize (i) to allow f for which $f^2 \in K$. This converse statement remains to be proven/disproven (first the converse statement would need to be made precise by defining the phrase “has Bessel solutions”).

The algorithm presented in this paper finds solutions in all cases where $L_B \longrightarrow L$ is satisfied (with $f \in K$; the case $f \notin K, f^2 \in K$ is not handled in this paper).

Since we are now interested in finding those transformations for a given operator L we study them more precisely.

The relations $\rightarrow_E$ and $\rightarrow_G$ are equivalence relations (see [1] or [7]) but $\rightarrow_C$ is not (if f is not a Möbius transformation, then (i) is not invertible, unless we generalize to allow algebraic functions for f).

An important question when searching for transformations between two operators L_1 and L_2 is whether we can restrict our search to a specific order of the transformations $\rightarrow_C$, $\rightarrow_E$ and $\rightarrow_G$.

Lemma 2 *Let $L_1, L_2, L_3 \in K[\partial]$ be three differential operators such that $L_1 \rightarrow_G L_2 \rightarrow_E L_3$. Then there exists a differential operator $M \in K[\partial]$ such that $L_1 \rightarrow_E M \rightarrow_G L_3$. Similarly, if $L_1 \rightarrow_E L_2 \rightarrow_G L_3$ we find M such that $L_1 \rightarrow_G M \rightarrow_E L_3$.*

PROOF. Let $L_1 \xrightarrow{r_0, r_1}_G L_2 \xrightarrow{r}_E L_3$ and we denote $R = \exp(\int r)$. Then the solution space of L_3 is

$$\begin{aligned} V(L_3) &= \{R(r_0 y + r_1 y') \mid y \in V(L_1)\} \\ &= \{(r_0 - r_1 r)Ry + r_1(Ry)' \mid y \in V(L_1)\}. \end{aligned}$$

Hence, $L_1 \xrightarrow{r}_E M \xrightarrow{\bar{r}_0, \bar{r}_1}_G L_3$ for some $M \in K[\partial]$ with $\bar{r}_0 = r_0 - r_1 r$. The converse follows by the same computation. $\square$

Since the order of $\rightarrow_E$ and $\rightarrow_G$ can be switched we write $\rightarrow_{EG}$ for any sequence of those. The order concerning a change of variables can be changed as follows.

Lemma 3 *Let $L_1, L_2 \in K[\partial]$ such that $L_1 \rightarrow L_2$. Then there exists $M \in K[\partial]$ such that $L_1 \rightarrow_C M \rightarrow_{EG} L_2$.*

PROOF. As in the last proof one simply rewrites the solution space of $V(L_3)$ to show that

- (i) $L_1 \xrightarrow{r}_E L_2 \xrightarrow{f}_C L_3 \Rightarrow \exists M \in K[\partial]$ such that $L_1 \xrightarrow{f}_C M \xrightarrow{\bar{r}}_E L_3$, with $\bar{r} = R(f)'$ and $R(x) = \int r dx$.
- (ii) $L_1 \xrightarrow{r_0, r_1}_G L_2 \xrightarrow{f}_C L_3 \Rightarrow \exists M \in K[\partial]$ such that $L_1 \xrightarrow{f}_C M \xrightarrow{\bar{r}_0, \bar{r}_1}_G L_3$ with $\bar{r}_0 = r_0(f)$ and $\bar{r}_1 = r_1(f)/f'$.

The rest follows immediately. $\square$

We conclude: If $L_1 \rightarrow L_2$ for any sequence of transformations then $L_1 \xrightarrow{f}_C M \rightarrow_{EG} L_2$ for some $M \in K[\partial]$. Assume $L_2 = \partial^2 + a_1 \partial + a_0$ is given and we search for transformations where $L_1 = L_B$. If no gauge transformation occurred, then it is easy to recover f from L_2 by using the fact that $I := a_0 - a_1^2/4 - a_1'/2$ remains invariant under $\rightarrow_E$, and the fact that I completely determines f (see [3, 4, 5, 15] for details). But there is no simple formula for an invariant under $\rightarrow_{EG}$ that can determine f completely.

There are, however, local invariants for $\rightarrow_{EG}$, based on exponent differences introduced in the next section. Each local invariant only yields partial information about f , and our main task will be to:

Combinatorial Problem: recover f (and the Bessel parameter ν) by combining these partial pieces of data.

After f and ν are found, we can find the $M \rightarrow_{EG} L_2$ transformation using algorithms presented in [1].

3.1 The Exponent Difference

From here on, the word “exponent” refers to “unramified generalized exponent” (i.e. $m = 1$ in Section 2.2).

Lemma 4 *Let $L, M \in K[\partial]$ be two differential operators such that $M \xrightarrow{r}_E L$ and let e be an exponent of M at the point p . Furthermore, let r have the series representation*

$$r = \sum_{i=m}^{\infty} r_i t_p^i, \quad m \in \mathbb{Z}, m \leq -1.$$

Then $e + \sum_{i=m}^{-1} r_i t_p^{i+1}$ is an exponent of L at p .

If $M \rightarrow_G L$ and e is an exponent of M at the point p , then L has an exponent $\bar{e}$ with $\bar{e} \equiv e \pmod{\mathbb{Z}}$.

PROOF. Let t be the local parameter t_p . Since e is an exponent, M has a solution of the form

$$y = \exp\left(\int \frac{e}{t} dt\right) S,$$

for some $S \in \mathbb{C}[[t]][\ln(t)]$ with non-zero constant term. The exp-product converts this solution into

$$z = \exp\left(\int r dt\right) \exp\left(\int \frac{e}{t} dt\right) S.$$

In order to determine the exponent at p we have to rewrite this expression into the form (2). We have to handle the positive and negative powers of t in r separately. For the power series part $\bar{r} = \sum_{i=0}^{\infty} r_i t^i$ we get

$$\exp\left(\int \bar{r} dt\right) = \exp\left(\sum_{i=0}^{\infty} \frac{r_i}{i+1} t^{i+1}\right).$$

With $\exp(x) = \sum_{i=0}^{\infty} \frac{x^i}{i!}$ we can rewrite this as a power series in t such that $\exp(\int \bar{r} dt) = \sum_{i=0}^{\infty} a_i t^i$ with $a_0 = 1$.

The negative powers of t in the series expansion of r become a part of the exponent:

$$\exp\left(\int \sum_{i=m}^{-1} r_i t^i dt\right) = \exp\left(\int \frac{1}{t} \sum_{i=m}^{-1} r_i t^{i+1} dt\right).$$

Combining the two results we get

$$z = \exp\left(\int \frac{1}{t} \left(e + \sum_{i=m}^{-1} r_i t^{i+1}\right) dt\right) \bar{S},$$

where $\bar{S} \in \mathbb{C}[[t]][\ln(t)]$ has a non-zero constant term.

For a gauge transformation with parameters r_0 and r_1 , the result follows from the following facts: the exponents of r_0 and r_1 are integers, taking derivatives of unramified power series changes exponents only by integers, and finally, adding unramified power series can only change exponents by integers. Hence, gauge transformations only change exponents by integers (note: for ramified generalized exponents one would get $\bar{e} \equiv e \pmod{\frac{1}{m}\mathbb{Z}}$ with m as in Section 2.2). $\square$

If $\deg(L) = 2$, we have two exponents e_1, e_2 at a point p and we call $e_1 - e_2$ an exponent difference. The exponent difference is defined up to a $\pm$ sign and we denote $\Delta(L, p) = \pm(e_1 - e_2)$. It follows from the previous lemma that $\Delta \pmod{\mathbb{Z}}$ is invariant under exp-products and gauge transformations, i.e. $\Delta(L_1, p) \pmod{\mathbb{Z}} = \Delta(L_2, p) \pmod{\mathbb{Z}}$ if $L_1 \rightarrow_{EG} L_2$.

Since we will be interested in singularities of L which are logarithmic or whose exponent difference is not an integer, the following theorem will be important.

Theorem 1 *Let $L \in K[\partial]$ be a differential operator and let p be a point. If there exists an operator $M \in K[\partial]$ where p*

is regular such that $M \xrightarrow{EG} L$, then the solutions of L are not logarithmic and $\Delta(L, p) \in \mathbb{Z}$.

PROOF. Let M and p be as required. Then there exist rational functions $r_0, r_1, r_2 \in K$ and $\tilde{M} \in K[\partial]$ such that $M \xrightarrow{r_0} \tilde{M} \xrightarrow{r_1, r_2} L$. Furthermore, let p be a regular point of M . The generalized exponents at p are 0 and 1. Hence, $\Delta(M, p) \in \mathbb{Z}$ and from the previous lemma it follows that $\Delta(L, p) \in \mathbb{Z}$.

Since p is regular, the local solutions of M at p do not have logarithms. The local solutions of L at p can be derived from these by an exp-product and a gauge transformation. Neither of these transformations brings in logarithms. Hence, solutions of L at p are not logarithmic. $\square$

In the following we will use Δ to find the parameter f of the change of variables as well as the Bessel parameter ν that is involved. Assume that $K = \mathbb{C}(x)$; so we can factor every polynomial into linear factors. We will later treat $K = k(x)$ for finitely generated extensions k of $\mathbb{Q}$.

Theorem 2 Let $M \in K[\partial]$ such that $L_B \xrightarrow{f} M$, $f \in K$.

- (i) If p is a zero of f with multiplicity $m \in \mathbb{N}$, then p is a regular singularity of M and $\Delta(M, p) = \pm 2m\nu$.
- (ii) If p is a pole of f with multiplicity $m \in \mathbb{N}$ such that $f = \sum_{i=-m}^{\infty} f_i t_p^i$, then p is an irregular singularity of M and

$$\Delta(M, p) = \pm 2 \sum_{i=-m}^{-1} i f_i t_p^i. \quad (3)$$

PROOF. Let t be the local parameter t_p . To compute the generalized exponents of M at the point p , we start with a solution y of L_B , replace x by f to get a solution z of M and rewrite z into the form (2).

(a) Let p be a zero of f with multiplicity $m > 0$, then f has the representation $f = t^m \sum_{i=0}^{\infty} f_i t^i$ with $f_i \in k$ and $f_0 \neq 0$. Furthermore, let $y \in V(L_B)$ be a local solution at $x = 0$ of the form

$$y = x^\nu \sum_{i=0}^{\infty} a_i x^i, \quad a_i \in k, a_0 \neq 0.$$

If we now replace x by f , we get a local solution $z = f^\nu \sum_{i=0}^{\infty} a_i f^i$ of M . To compute the generalized exponent at p we rewrite z such that $z = \exp(\int e/t dt) \sum_{i=0}^{\infty} b_i t^i$ for some $e \in E, b_i \in k, b_0 \neq 0$. The fact that $f^i = t^{mi} \tilde{f}^i$, where the constant coefficient of $\tilde{f} \in k[[t]]$ is non-zero, simply yields $e_1 = m\nu$.

Similarly, for the second independent local solution of L_B at $x = 0$, which has exponent $-\nu$, we obtain the generalized exponent $e_2 = -m\nu$. Hence, the singularity p is regular and $\Delta(M, p) = \pm(e_1 - e_2) = \pm 2m\nu$.

If $\nu \in \mathbb{Z}$ the second independent solution contains a logarithm $\ln(x)$. However, we can still do the same computations. The solution z would then involve $\ln(t)$ and the result for the exponent is still true.

(b) A similar approach works in second case. Let p be a pole of f with multiplicity $m \in \mathbb{N}$. Then f can also be written as $f = t^{-m} \sum_{i=0}^{\infty} f_{i-m} t^i$ with $f_i \in k, f_{-m} \neq 0$.

We start with the local solution y of L_B at $x = \infty$ corresponding to the exponent $e := \frac{1}{t_\infty} + \frac{1}{2}$. There exists a series $S \in k[[t_\infty]]$ such that

$$y = \exp\left(\int \frac{e}{t_\infty} dt_\infty\right) S = \exp\left(-\frac{1}{t_\infty}\right) t_\infty^{1/2} S \quad (4)$$

is a solution of L_B . In order to get a solution z of M we have to replace x by f , i.e. $t_\infty = \frac{1}{x}$ by $\frac{1}{f}$. Hence, we do the following substitutions:

$$t_\infty \longrightarrow \frac{1}{f} = t^m \sum_{i=0}^{\infty} \tilde{f}_i t^i, \quad \tilde{f}_i \in k, \tilde{f}_0 \neq 0, \quad (5)$$

$$\frac{1}{t_\infty} \longrightarrow f,$$

$$\text{and } t_\infty^{1/2} \longrightarrow \frac{1}{f^{1/2}} = t^{m/2} \sum_{i=0}^{\infty} \bar{f}_i t^i, \quad \bar{f}_i \in k, \bar{f}_0 \neq 0.$$

We apply these substitutions to (4) and get a local solution z of M at $x = p$: $z = \exp(-f) t^{m/2} \tilde{S}$, $\tilde{S} \in k[[t]]$, where $\tilde{S}$ combines all the new series that we obtain from (5). As in the proof of Lemma 4 we can rewrite $\exp(\sum_{i=0}^{\infty} f_i t^i)$ as power series in t . The negative powers of t remain in the exponential part, which then becomes

$$\exp\left(-\sum_{i=-m}^{-1} f_i t^i\right) t^{m/2} = \exp\left(\int \frac{1}{t} \left(\sum_{i=-m}^{-1} -i f_i t^i + \frac{m}{2}\right) dt\right).$$

Thus, z has the generalized exponent $-(\sum_{i=-m}^{-1} i f_i t^i) + \frac{m}{2}$.

If we start with the second independent solution with generalized exponent $-\frac{1}{t_\infty} + \frac{1}{2}$ we similarly get $(\sum_{i=-m}^{-1} i f_i t^i) + \frac{m}{2}$. Hence, p is an irregular singularity of M and $\Delta(L, p) = \pm 2 \sum_{i=-m}^{-1} i f_i t^i$. $\square$

The last two theorems illustrate the following definitions.

Definition 3. A point p of $L \in K[\partial]$ for which $\Delta(L, p) \in \mathbb{Z}$ and L is not logarithmic at p is called an *exp-apparent* point. If p is not exp-apparent, p is called

- (i) *exp-regular* $\Leftrightarrow \Delta(L, p) \in \mathbb{C} \setminus \mathbb{Z}$ or L is logarithmic at p ,
- (ii) *exp-irregular* $\Leftrightarrow \Delta(L, p) \in \mathbb{C} \setminus [1/t_p] \setminus \mathbb{C}$.

We denote the set of singularities that are exp-regular by S_{reg} and those that are exp-irregular by S_{irr} .

Note that regular points are also exp-apparent and that every point which is not exp-apparent must be a singularity.

If we have an operator L such that $L_B \xrightarrow{f} M \xrightarrow{EG} L$, exp-apparent points of L are singularities of L which might have been introduced by exp-products and gauge transformations. So they are unimportant when searching for f . Exp-irregular singularities of L are also irregular singularities of M and correspond exactly to the poles of f . Finally, every exp-regular singularity is a zero of f (this is not a one-to-one correspondence; a zero of f need not be a singularity of L when $\nu \in \mathbb{Q}$). We combine these very important results in the following corollary.

Corollary 1 If $L_B \xrightarrow{f} M \xrightarrow{EG} L$, the following holds:

- (i) $p \in S_{irr} \Leftrightarrow p$ is a pole of f .
- (ii) $p \in S_{reg} \Rightarrow p$ is a zero of f .

PROOF. (i) If p is a pole of f , it follows from Theorem 2(ii) that $p \in S_{irr}$. If p is not a pole, $\Delta(M, p) \in \mathbb{C}$ and $p \notin S_{irr}$ by Theorem 1 and Theorem 2(i).

(ii) Using $S_{reg} \cap S_{irr} = \emptyset$ and (i), the only thing that remains to be proven is that $p \notin S_{reg}$ for any ordinary point p of f . So let p be neither a zero nor a pole of f . Then p is a regular point of M , i.e. $\Delta(M, p) \in \mathbb{Z}$ and the solutions of M at p are not logarithmic. From Theorem 1 it follows that this also holds for L . $\square$

The sets S_{reg} and S_{irr} can be computed easily. So we already know all poles of f . But since we have no equivalence in (ii), we might not see *all* zeros of f . However, from Theorem 2 we know more information about the poles of f : we can compute every polar part of f (up to a $\pm$ sign).

By the polar part of f at a point p we mean the negative power part of the series representation of f at p . So if $f = \sum_{i=m}^{\infty} f_i t_p^i$, $m \in \mathbb{Z}$, the polar part at p is $f = \sum_{i=m}^{-1} f_i t_p^i$. Hence, the polar part of f at $p \in \mathbb{C}$ is non-zero if and only if p is a pole of f . Considering the partial fraction decomposition, the polar parts at all $p \in \mathbb{C}$ determine f uniquely up to a polynomial in $k[x]$, e.g. $a_0 + g(x)$, $g(x) \in xk[x]$. Furthermore, the polynomial $g(x)$ is the polar part of f at $p = \infty$, which can be seen when representing f at $p = \infty$ (i.e. in terms of $t_{\infty} = \frac{1}{x}$). Thus, the sum of the polar parts determines f up to a constant a_0 .

Theorem 2(ii) shows how to compute those polar parts from the exponent differences $\Delta(L, p)$ at the exp-irregular points. Since the $\Delta(L, p)$ are defined up to $\pm$ signs, we obtain each polar part up to a sign as well. If S_{irr} has n elements, our algorithm checks all 2^n combinations⁴ of $\pm$ signs, yielding a set of candidates denoted by $\mathcal{F}$. One of these candidates will be f up to a constant.

If we know at least one zero of f we can use it to compute this constant. In this case we can find f by trying all candidates in $\mathcal{F}$. But if $S_{reg} = \emptyset$, then we will need an additional method (Section 4.1) to find f and the Bessel parameter ν .

3.2 The Parameter ν

An important property for Bessel functions is that the space $\mathbb{C}(x)B_{\nu}(x) + \mathbb{C}(x)B_{\nu}(x)'$, i.e. the space generated by all gauge transformations of a Bessel function $B_{\nu}(x)$, is invariant under $\nu \rightarrow \nu + 1$. In other words, it is sufficient to find ν modulo an integer. If we take a 'wrong' ν that is off by an integer, then this is caught by the gauge transformation, which is computed afterwards.

For the Bessel parameter ν we will have to consider several cases. We may assume $\nu \notin \frac{1}{2} + \mathbb{Z}$ (otherwise L_B is reducible). The following lemma handles the case $\nu \in \mathbb{Z}$.

Lemma 5 *Let $L \in K[\partial]$ be a differential operator. Assume $L_B \rightarrow L$. Then the following are equivalent:*

- (i) *The Bessel parameter is an integer, i.e. $\nu \in \mathbb{Z}$.*
- (ii) *There is an exp-regular singularity p of L such that L is logarithmic at p .*

PROOF. (i) $\Rightarrow$ (ii) If ν is an integer, the functions in the solution space $V(L_B)$ are gauge transformations of the functions in $V(L_B, \nu = 0)$. So it is sufficient to prove the case $\nu = 0$. Then L_B has a logarithmic solution at $x = 0$. This solution is transformed into a solution z of L . This works as in the proofs of Theorem 2 and Lemma 4. If p is a zero of the parameter f in the change of variables, the logarithm changes as follows:

$$\begin{aligned} \ln(x) \rightarrow \ln(f) &= \ln\left(t_p^m c \sum_{i=0}^{\infty} f_i t_p^i\right) \\ &= \ln(c) + m \ln(t_p) + \ln\left(1 + \sum_{i=1}^{\infty} f_i t_p^i\right), \end{aligned}$$

⁴Actually: 2^{n-1} combinations, see Example 1.

where $f_0 = 1$. So L has a logarithmic solution at the point p and since it was a zero of f , it is also exp-regular.

(ii) $\Rightarrow$ (i) If $\nu \notin \mathbb{Z}$ then the local solutions of L_B do not involve logarithms⁵ and hence the same is true for L . $\square$

We will use the fact from Theorem 2(i) that $\Delta(L, p)$ is $2m\nu \bmod \mathbb{Z}$ for exp-regular singularities $p \in S_{reg}$.

Lemma 6 *These statements are true for all $s \in S_{reg}$:*

- (i) *logarithmic case:* L logarithmic at $s \Leftrightarrow \nu \in \mathbb{Z}$
- (ii) *integer case:* $S_{reg} = \emptyset \Rightarrow \nu \in \mathbb{Q} \setminus \mathbb{Z}$
- (iii) *rational case:* $\Delta(L, s) \in \mathbb{Q} \setminus \mathbb{Z} \Rightarrow \nu \in \mathbb{Q} \setminus \mathbb{Z}$
- (iv) *base field case:* $\Delta(L, s) \in k \setminus \mathbb{Q} \Leftrightarrow \nu \in k \setminus \mathbb{Q}$
- (v) *irrational case:* $\Delta(L, s) \notin k \Leftrightarrow \nu \notin k$

Exactly one case applies (we assume $L_B \rightarrow L$). Case (v) only occurs if k is not algebraically closed (in Section 4.2).

PROOF. Case (i) has been proven in Lemma 5. By Theorem 2(i) $\Delta(L, s) = 2m_s\nu + z_s$ for all zeros s of the parameter f . Hereby, m_s is the multiplicity and $z_s \in \mathbb{Z}$. Since f has at least one zero (possibly at ∞) there is at least one such equation from which we can deduce cases (ii) to (v). $\square$

Recall that if $S_{reg} \neq \emptyset$, we can pick a zero $s \in S_{reg}$ of f . In that case we can compute candidates $\mathcal{F}$ for the parameter in the change of variables. For each candidate f we compute a set of candidates $\mathcal{N}$ for ν .

Definition 4. Let $s \in S_{reg} \neq \emptyset$ be a zero of the parameter $f \in K$. Let m_s be the multiplicity of s . We define

$$\begin{aligned} \mathcal{N}_s &:= \left\{ \frac{\Delta(L, s) + i}{2m_s} \mid 0 \leq i \leq 2m_s - 1 \right\} \\ \text{and } \mathcal{N} &:= \left\{ \pm \nu \bmod \mathbb{Z} \mid \forall s \in S_{reg} \exists z_s \in \mathbb{Z} : \right. \\ &\quad \left. \nu + z_s \in \mathcal{N}_s \text{ or } -\nu + z_s \in \mathcal{N}_s \right\}. \end{aligned}$$

Both sets are finite and it is easy to see:

Corollary 2 *If $L_B \rightarrow L$ and ν is the Bessel parameter in L_B then $\pm \nu \bmod \mathbb{Z} \in \mathcal{N}$.*

4. THE ALGORITHM

The input of our algorithm is a differential operator L_{in} and we want to know whether the solutions can be expressed in terms of Bessel functions. We assume that $L_B \rightarrow L_{in}$ for some transformations. If we find a solution to that problem, then we also find the solution space of L_{in} .

We will first assume $k = \mathbb{C}$ and will deal with more general fields k in the next section. Let L_{in} be a differential operator of degree two with coefficients in $K = \mathbb{C}(x)$.

Let's summarize the steps of the algorithm that we have deduced in the previous sections:

1. **(Singularities)** We can compute the singularities S of L_{in} by factoring the leading coefficient of L_{in} and the denominators of the other coefficients into linear factors.
2. **(Generalized exponents)** For each $s \in S$ we compute $d_s = \Delta(L_{in}, s)$, isolate exp-apparent points with $d_s \in \mathbb{Z}$, and differ between exp-regular singularities S_{reg} with $d_s \in \mathbb{C}$ and exp-irregular singularities S_{irr} with $d_s \in \mathbb{C}[t_s^{-1}] \setminus \mathbb{C}$.

⁵In general, if $\Delta(L, p) \notin \mathbb{Z}$ then no logarithm appears at p .

3. **(Polar parts)** We can use the exponent differences d_s for $s \in S_{irr}$ to compute candidates $\mathcal{F}$ for the parameter f up to a constant $c \in k$.
4. **(Constant term of f)** In all cases but the integer case we know at least one zero of f by picking some $s_0 \in S_{reg}$. So we can also compute the missing constant c for each $\tilde{f} \in \mathcal{F}$.
5. **(The set $\mathcal{N}$)** The set $\mathcal{N}$ is a set of candidates for ν . When not in the integer case, we compute this finite set as in Section 3.2; $\mathcal{N}$ might depend on f .
6. **(Compute M)** For each $f \in \mathcal{F}$ and each $\nu \in \mathcal{N}$ compute $M = M(\nu, f) \in k(x)[\partial]$ such that $L_B \xrightarrow{f} M$.
7. **(Exp-product and gauge transformation)** For each M decide (e.g. using [1]) whether $M \rightarrow_{EG} L_{in}$, and if so, compute the transformation.

The only case in which this algorithm does not yet work is when $S_{reg} = \emptyset$, which we will handle in the next section. Note that one can also use the case separation of Lemma 6 to reduce the number of candidates that we obtain from steps 4 and 5. More details can be found in [7].

Example 1: Let

$$L := 9(3x^2 - 14x + 7)(x-2)^2(x-5)^2(x-1)^4\partial^2 + \\ 9(3x^4 - 28x^3 + 82x^2 - 52x - 21)(x-2)(x-5)(x-1)^3\partial - \\ (3x^2 - 14x + 7)^3(9x^8 - 198x^7 + 1845x^6 - 9540x^5 + \\ 30060x^4 - 59328x^3 + 71860x^2 - 48968x + 14404).$$

We apply the algorithm above to L step by step:

Step 1: The zeros of the leading coefficient are 1, 2, 5 and $(7 \pm 2\sqrt{7})/3$. Furthermore, ∞ is also a singularity of L .

Step 2: The generalized exponents⁶ at the points $p = (7 \pm 2\sqrt{7})/3$ are 0 and 2; those at $p = 2$ are -2 and 2. No logarithm appears at these three points either, so they are exp-apparent and are not considered anymore. At the other points we compute the following exponent differences:

$$\Delta(M, 1) = 8/t_1 = 8/(x-1)$$

$$\Delta(M, 5) = 4/3$$

$$\Delta(M, \infty) = 6/t_\infty^3 - 40/t_\infty^2 + 64/t = 6x^3 - 40x^2 + 64x$$

Hence, $S_{irr} = \{1, \infty\}$ and $S_{reg} = \{5\}$.

Step 3: Using equation (3) in Theorem 2 we can compute the polar parts corresponding to the exp-irregular points:

$$f_1 = \frac{4}{x-1}, \quad f_\infty = x^3 - 10x^2 + 32x.$$

The set of candidates is $\mathcal{F} = \{f_1 + f_\infty, f_1 - f_\infty, -f_1 + f_\infty, -f_1 - f_\infty\}$. Since $L_B \xrightarrow{-x} L_B$ we can ignore the latter two candidates.

Step 4: The point 5 $\in S_{reg}$ must be a zero of f . Evaluating the candidates at this point yields the candidates

$$g_1 = f_1 + f_\infty - 36 \quad \text{and} \quad g_2 = f_1 - f_\infty + 34$$

for the parameter in the change of variables.

Step 5: To get candidates for ν we determine $\mathcal{N}$ which is equal to $\mathcal{N}_5$ because we just have one exp-regular point.

⁶E.g. using the command `gen_exp` in *Maple*.

Therefore we compute the multiplicity of the zero 5 of g_1 and g_2 ; it is 1 in both cases. Therefore:

$$\mathcal{N} = \mathcal{N}_5 = \left\{ \frac{2}{3}, \frac{7}{6} \right\}.$$

Step 6: For each pair f, ν compute M such that $L_B \xrightarrow{f} M$.

Step 7: Finally, for each M check if $M \rightarrow_{EG} L$.

In practice we combine steps 6 and 7. At the end we will get $L_B \xrightarrow{\nu=2/3} M$ for $f = (x-5)(x-2)^3/(x-1)$ and the solution space

$$C_1 I_{2/3} \left(\frac{(x-5)(x-2)^3}{x-1} \right) + C_2 K_{2/3} \left(\frac{(x-5)(x-2)^3}{x-1} \right).$$

One can solve Example 1 with prior algorithms [3, 4, 5, 15] because no gauge transformation is involved. An example not solved by prior algorithms will be given in Example 2.

4.1 Integer Case

Let $S_{reg} = \emptyset$, i.e. $\Delta(L_{in}, s) \in \mathbb{Z}$ for all $s \notin S_{irr}$. Then $\nu \in \mathbb{Q}$ and we define

$$\mathcal{N}(m) := \left\{ \frac{i}{2m}, i = 1, \dots, 2m-1 \right\} \setminus \left\{ \frac{1}{2} \right\}.$$

Then the following holds:

Lemma 7 (a) Let n be the degree of the numerator of f . Then there exists $p, \ell \in \mathbb{Z}$ such that $p \mid n$ and $\nu + \ell \in \mathcal{N}(p)$.

(b) For the degree of the numerator we know:

- (i) If $\infty \in S_{irr}$, then $\deg(\text{numer}(f)) = \deg(\text{numer}(f+c))$ for all $c \in \mathbb{C}$.
- (ii) If $\infty \notin S_{irr}$, then $p \mid \deg(\text{numer}(f)) \Leftrightarrow p \mid \deg(\text{denom}(f))$.

(c) Let $p \in \mathbb{N}$ and $p > 1$, $f \in k[x]$. With linear algebra we can decide whether there exists g such that $f = g^p$ and compute g explicitly.

We will use the lemma as follows: If we know the degree n of the numerator of f then by (a) we can compute candidates $\mathcal{N}(p)$ for ν for every $p \mid n$. By (b) it is enough to know candidates $\mathcal{F}$ for f modulo the constant term to get all $p \mid n$ we have to consider. In our case we still have an undetermined constant c . But if the numerator of f is a p -th power ($p > 1$), we can determine g by (c) such that $\text{numer}(f)$ can be equal to g^p . From $\text{numer}(f) - g^p = 0$ we can then compute candidates for c .

PROOF. (a) Let s be a zero of the parameter f of the change of variables, then $\Delta(L_{in}, s) = 2m\nu$ modulo $\mathbb{Z}$.⁷ Since $\nu \in \mathbb{Q}$ can be written as $\nu = z + \frac{\nu_1}{2p}$ with $z \in \mathbb{Z}, \nu_1, p \in \mathbb{N}$ and $0 < \nu_1 < 2p$, $\gcd(\nu_1, p) = 1$. Then $\nu - z \in \mathcal{N}(p)$ and $2m\nu = 2mz + \frac{m\nu_1}{p} \in \mathbb{Z}$. Since $z, m \in \mathbb{Z}$ and $\gcd(\nu_1, p) = 1$ this is equivalent to $p \mid m$. So p divides all multiplicities of the zeros of f . The degree of the numerator of f is equal to the sum of these multiplicities and we get $p \mid \deg(\text{numer}(f))$. Hence, p and $\ell = -z$ satisfy the statement.

(b) After using the exp-irregular points S_{irr} to find the polar parts, f has the form $f = \frac{f_1}{f_2} + c + f_3$, where $f_1, f_2, f_3 \in k[x]$ and $\deg(f_1) < \deg(f_2)$ or $f_1 = 0$. The polar parts for

⁷Since $\Delta(L_{in}, s) \in \mathbb{Z}$ in this section, we need no $\pm$ signs when working modulo $\mathbb{Z}$.

$s \in S_{irr} \setminus \{\infty\}$ are combined in $\frac{f_1}{f_2}$. The polynomial f_3 is the polar part of $\infty \in S_{irr}$.

(i) In this case $\infty \in S_{irr}$ and hence $f_3 \neq 0$. So c does not affect the degree of the numerator of f .

(ii) Since $\infty \notin S_{irr}$, $f_3 = 0$ and $f = \frac{f_1}{f_2} + c$ with $\deg(f_1) < \deg(f_2)$.

Case 1: If $c \neq 0$, then $\deg(\text{numer}(f)) = \deg(\text{denom}(f))$ and nothing remains to be proven.

Case 2: If $c = 0$, then ∞ is a zero of f . The multiplicity m must be a multiple of p . Otherwise $\Delta(L_{in}, \infty) \notin \mathbb{Z}$ since $\Delta(L_{in}, \infty) = 2m\nu + z$ for some $z \in \mathbb{Z}$ and $2\nu = \frac{\nu_1}{p}$. Hence, $m = kp$ for some $k \in \mathbb{N}$.

This multiplicity of the point ∞ is $m = \deg(\text{denom}(f)) - \deg(\text{numer}(f))$. This can be seen if $f(\frac{1}{x})$ is written as power series at the point 0. In total we get $\deg(\text{numer}(f)) = \deg(\text{denom}(f)) - kp$ for some $k \in \mathbb{N}$ and this proves (ii).

(c) Comparing the highest coefficients of g^p and f in an ansatz for g yields the result. $\square$

Concluding, the integer case works as follows. For each $f \in \mathcal{F}$ that is determined up to a constant term c we perform the following steps:

-
- 1 $n := \deg(\text{numer}(f))$ if $\infty \in S_{irr}$
and $n := \deg(\text{denom}(f))$ otherwise
 - 2 for each $p \mid n$, $p \neq 1$
 - 3 compute candidates $\mathcal{C}$ for the constant term c
 - 4 for each $c \in \mathcal{C}$ and each $\nu \in \mathcal{N}(p)$
 - 5 compute M such that $L_B \xrightarrow{f+c}_C M$
-

4.2 Solving Over a General Field k

Until now, we were working over the constant field $k = \mathbb{C}$ and we haven't thought of the speed of the algorithm yet. We started by computing all the singularities of L and did some computations with them. So what we actually did is factor the leading coefficient $l(x)$ of L into linear factors. This can be very expensive and can lead to a huge extension of $\mathbb{Q}$, in which all the other computations take place. In this section we will discuss how we can work over a much smaller extension of $\mathbb{Q}$.

We will use the following setting. Let k be a finitely generated extension of $\mathbb{Q}$ such that the input operator L has coefficients in $k[x]$ and let $K = k(x)$. For each irreducible factor $q(x)$ of $l(x)$ in $k[x]$ we pick one zero s . Furthermore, let $\sigma \in \text{Hom}_k(k(s), \bar{k})$ be an embedding of $k(s)$ in an algebraic closure $\bar{k}$ that keeps s fixed and we denote the *trace* of an element $a \in k(s)$:

$$\text{Tr}(a) = \sum_{\sigma \in \text{Hom}_k(k(s), \bar{k})} \sigma(a).$$

We will now focus on each of the seven steps of the algorithm and explain the changes that have to be made.

1. **(Singularities)** When we factor the coefficients of L in $k[x]$ we get irreducible factors whose degree can be greater than one. For each irreducible factor, we fix one zero. The finite singularities then are

$$S = \{ \sigma(s) \mid s \text{ zero of irred. factor, } \sigma \in \text{Hom}_k(k(s), \bar{k}) \}.$$

Now fix an irreducible factor $q(x)$ of $l(x)$ and let s be a zero of $q(x)$ and $\sigma \in \text{Hom}_k(k(s), \bar{k})$.

The strategy is to use the computations at the singularity s for other singularities $\sigma(s)$, which are zeros of the same irreducible factor.

2. **(Generalized exponents)** In the computation of the generalized exponent at the point $x = s$ the field $k(s)$ is taken as the field of constants. If e_1, e_2 are the exponents at s then the exponents at $\sigma(s)$ are $\sigma(e_1), \sigma(e_2)$. Note: e_1, e_2 may be defined over an extension field of $k(s)$, in which case the σ 's need to be extended as well.

Similarly, if y is a local solution at the point $x = s$, then $\sigma(y)$ is a local solution at $x = \sigma(s)$ because the operator cannot distinguish between the points s and $\sigma(s)$. Hence, $\Delta(L, \sigma(s)) = \sigma(\Delta(L, s))$.

Since all our results were based on generalized exponents and exponent differences we can now transfer results for s to $\sigma(s)$. So for each irreducible factor $q(x)$, only one of its zeros is needed in S_{reg} or in S_{irr} . In the terminology of [6], the singularities are computed *up to conjugation over k* (see Section 5.1 in [6] for more details).

3. **(Polar parts)** Let $s \in S_{irr}$. We can compute the polar part f_s corresponding to s . Assume for now that $f \in k(x)$ (we will explain below what to do if $f \in \bar{k}(x) \setminus k(x)$). Then the polar part corresponding to $\sigma(s)$ is $f_{\sigma(s)} = \sigma(f_s)$. So the trace of f_s is:

$$\text{Tr}(f_s) = \sum_{\sigma \in \text{Hom}_k(k(s), \bar{k})} \sigma(f_s) = \sum_{\sigma \in \text{Hom}_k(k(s), \bar{k})} f_{\sigma(s)}.$$

The result is the polar part of f corresponding to the irreducible polynomial $q(x)$. Hence, we computed the whole polar part corresponding to all zeros of $q(x)$ by just using one zero of $q(x)$.

4. **(Constant term of f)** Let $f = \tilde{f} + c$ for some $\tilde{f} = \tilde{f}(x) \in \mathcal{F}$ be a candidate for the parameter in the change of variables. If $S_{reg} \neq \emptyset$, then we know at least one zero of f . Assume $s \in S_{reg}$, then we compute c such that $f(s) = 0$. If $s \notin k$, we would get $c \notin k$ in general. However, the $\sigma(s)$ must be zeros of f as well. So $q(x)$ divides the numerator of f , which translates into a system of linear equations (in one unknown: c) defined over k .

The integer case needs no change.

5. **(The set $\mathcal{N}$)** For these computations we only used exp-regular points $s \in S_{reg}$ with $\Delta(L_{in}, s) = 2m_s\nu$. If $\nu \in k$ then $\Delta(L_{in}, \sigma(s)) = \sigma(\Delta(L_{in}, s)) = \Delta(L_{in}, s)$ for all $s \in S_{reg}$. If $\nu \notin k$ then we compute ν^2 instead, which will be in k . In either case, we need to use only one root s for each irreducible $q(x)$.

6./7. **(Compute M , exp-product and gauge transformation)** From here, everything works as before.

One problem remains when generalizing the algorithm as we did above. For computing the polar parts in step 3 we used that f is defined over k . However, we take k as the smallest field for which $L \in k(x)[\partial]$, in which case f need not be in $k(x)$.

For example $L_B \xrightarrow{f}_C L$ with $f = \sqrt{2}(x^2 - 2)$ and $L = (x^2 - 2)^2 x \partial^2 + (x^4 - 4) \partial - 4(2x^4 - 8x^2 + 8 + \nu^2)x^3$ is more complicated. Here $\sqrt{2}$ does not appear in L . More generally, if $L_B \xrightarrow{cx}_C L$ for a constant c , then just c^2 appears in L .

If $L_{in} \in k(x)[\partial]$ is given, we will thus restrict constant factors c of f to elements in quadratic extensions of k for which $c^2 \in k$. This is enough to combine all Bessel functions

in one algorithm. However, we still have to prove that this is really sufficient in all cases.

Let c be the constant factor we search for and let $s \in S_{irr}$ be a singularity. Since a constant factor of f is also a factor of its polar parts, c must be a factor of $\Delta(L_{in}, s)$. For each point s we have the constant fields $k \subseteq k(c) \subseteq k(c, s) =: k_s$. The exponent difference $\Delta(L_{in}, s)$ is defined over k_s and we can read off k_s from $\Delta(L_{in}, s)$. So we have to find algebraic extensions $\tilde{k}$ of k of degree two such that $\tilde{k} \subseteq k_s$ for all s . Then for each $\tilde{k}$, take a constant $c \in \tilde{k}$ satisfying $c^2 \in k$. This gives a finite list of candidates for the constant factor c . If $k(c, s) = k(s)$ for each s then add $c = 1$ to this list as well.

For each c in the list, we can divide all exponent differences (for $s \in S_{irr}$) by c and apply the algorithm as before. This effectively divides f by c , and if we picked the correct c , the quotient will be in $k(x)$ so then the algorithm will work correctly. If we try all candidates c , at some time we will get the right one and we will find a solution.

Example 2: Consider the differential equation

$$(x y''(x))'' - \left(\left(\frac{9}{x} + \frac{8}{M} x \right) y'(x) \right)' = \lambda^2 \left(\lambda^2 + \frac{8}{M} \right) x y(x)$$

for all $x \in (0, \infty)$, whereas M and λ are constant parameters. Factoring the corresponding differential operator (for example in *Maple*) gives two operators L_1 and L_2 of degree two. This example occurred in research of W. N. Everitt. Although it has been solved before (see [8]) by the second author⁸, our implementation is the first that can solve L_1 and L_2 completely automatically. One of them is

$$L_1 = \partial^2 + \frac{(\lambda^4 M^2 x^2 + 8 \lambda^2 M x^2 + 16 x^2 - 48 M)}{x(\lambda^4 M^2 x^2 + 8 \lambda^2 M x^2 + 16 x^2 - 16 M)} \partial + \frac{\lambda^2 (-4 \lambda^2 M^2 - 32 M + 16 x^2 + 8 \lambda^2 M x^2 + \lambda^4 M^2 x^2)}{\lambda^4 M^2 x^2 + 8 \lambda^2 M x^2 + 16 x^2 - 16 M}$$

and our implementation yields the solutions

$$\frac{C_1}{x} (-2\lambda M J_1(x\lambda) + (\lambda^2 M + 4) x J_0(x\lambda)) + \frac{C_2}{x} (-2\lambda M Y_1(x\lambda) + (\lambda^2 M + 4) x Y_0(x\lambda)).$$

Likewise our implementation solves L_2 as well.

Note: A referee alerted us that the soon to be released Maple 12 can solve L_1 and L_2 in terms of Heun functions. For completeness, we give here another equation that to our knowledge is not solved by other algorithms:

$$L = (4x^6 - 6x^4 - 5x^2 + 3)(x^4 + 1)^4 x \partial^2 - 3(4x^{10} - 2x^8 + 11x^6 - 13x^4 - 5x^2 + 1)(x^4 + 1)^3 \partial + (12x^{18} + 18x^{16} - 83x^{14} + 83x^{12} + 534x^{10} - 714x^8 - 299x^6 + 315x^4 + 224x^2 - 58)x^3.$$

5. CONCLUSION

We gave an algorithm to detect the Bessel type solutions of a second order differential operator with rational function coefficients and implemented it in *Maple*.

⁸Personal contribution at the International Conference on Difference Equations, Special Functions and Applications, Technical University Munich, Germany: July 2005.

The method can in principle be applied to the case of solutions in terms of the Gauß hypergeometric function. However, this case is significantly more difficult if we aim to solve it in general (if we restrict f to Möbius transformations then it is an easy addition to [1]). In particular, the important case where all exponent differences are in $\mathbb{Q}$ poses interesting but difficult problems that are as of yet not solved.

Another task for future work is to generalize our algorithm to allow f with $f^2 \in K$ (instead of $f \in K$ as in this paper) and then to prove completeness for that algorithm, i.e. for a definition of “Bessel type” that is both natural and general, prove that every Bessel type solution would always be found.

6. REFERENCES

- [1] BARKATOU, M. A., AND PFLÜGEL, E. On the Equivalence Problem of Linear Differential Systems and its Application for Factoring Completely Reducible Systems. In *ISSAC 1998*, 268–275.
- [2] BRONSTEIN, M. An improved algorithm for factoring linear ordinary differential operators. In *ISSAC 1994*, 336–340.
- [3] BRONSTEIN, M., AND LAFAILLE, S. Solutions of Linear Ordinary Differential Equations in Terms of Special Functions. In *ISSAC 2002*, 23–28.
- [4] CHAN, L., AND CHEB-TERRAB, E. S. Non-Liouvillean Solutions for Second Order Linear ODEs. In *ISSAC 2004*, 80–86.
- [5] CHEB-TERRAB, E. S. Solutions for the General, Confluent and Biconfluent Heun equations and their connection with Abel equations. *J. Phys. A: Math. Gen.* 37 (2004), 9923–9949.
- [6] CLUZEAU, T., AND VAN HOEIJ, M. A Modular Algorithm to Compute the Exponential Solutions of a Linear Differential Operator. *J. Symb. Comput.* 38 (2004), 1043–1076.
- [7] DEBEERST, R. *Solving Differential Equations in Terms of Bessel Functions*. Master’s thesis, Universität Kassel, 2007.
- [8] EVERITT, W. N., SMITH, D. J., AND VAN HOEIJ, M. The Fourth-Order Type Linear Ordinary Differential Equations. *arXiv:math/0603516* (2006).
- [9] VAN HOEIJ, M. *Factorization of Linear Differential Operators*. PhD thesis, Universiteit Nijmegen, 1996.
- [10] VAN DER HOEVEN, J. Around the Numeric-Symbolic Computation of Differential Galois Groups. *J. Symb. Comp.* 42 (2007), 236–264.
- [11] KAMRAN, N., OLVER, P. J. Equivalence of Differential Operators. *Siam J. Math. Anal.* 20 (1989), 1172–1185.
- [12] PERSON, A. C. *Solving Homogeneous Linear Differential Equations of Order 4 in Terms of Equations of Smaller Order*. PhD thesis, North Carolina State University, 2002.
- [13] VAN DER PUT, M., AND SINGER, M. F. *Galois Theory of Linear Differential Equations*, Springer, Berlin, 2003.
- [14] SHANIN, A. V., AND CRASTER, R. V. Removing False Singular Points as a Method of Solving Ordinary Differential Equations. *Eur. J. Appl. Math.* 13 (2002), 617–639.
- [15] WILLIS, B. L. An Extensible Differential Equation Solver. *SIGSAM Bulletin* 35 (2001), 3–7.

On the Computation of the Topology of a Non-Reduced Implicit Space Curve

Daouda Niang Diatta
University of Limoges, XLIM,
INRIA Sophia-Antipolis,
France.
dndiatta@sophia.inria.fr

Bernard Mourrain
INRIA Sophia-Antipolis,
France.
mourrain@sophia.inria.fr

Olivier Ruatta
University of Limoges, XLIM,
France.
olivier.ruatta@unilim.fr

ABSTRACT

An algorithm is presented for the computation of the topology of a non-reduced space curve defined as the intersection of two implicit algebraic surfaces. It computes a Piecewise Linear Structure (PLS) isotopic to the original space curve.

The algorithm is designed to provide the exact result for all inputs. It's a symbolic-numeric algorithm based on subresultant computation. Simple algebraic criteria are given to certify the output of the algorithm.

The algorithm uses only one projection of the non-reduced space curve augmented with adjacency information around some "particular points" of the space curve.

The algorithm is implemented with the Mathemagix Computer Algebra System (CAS) using the SYNAPS library as a backend.

Categories and Subject Descriptors

I.1.4 [Symbolic and Algebraic Manipulation]: Applications; I.3.5 [Computer Methodologies]: Computer Graphics, Computational Geometry and Object Modeling-Geometric Algorithms

General Terms

Algorithms

Keywords

Algebraic Curves, Subresultants Sequence, Generic Conditions, Topology Computation, Sturm-Habicht Sequence, Exact Geometric Computation

Introduction

The problem of computing the topological graph of algebraic curves plays an important role in many applications such as plotting [17] and sectioning in Computer Aided Geometric Design [19], [20]. A wide literature exists on the computation of the topology of plane curves [12, 16, 18, 7, 8, 9, 10,

1]. The problem of computing the topology of space curves has been less investigated. In [14], the case of intersections curves of parametric surfaces is considered, based on the analysis of planar curves in the parameter domains. In [2], Alcázar and Sendra give a symbolic-numeric algorithm for **reduced** space curves using subresultant and GCD computations of approximated polynomials. If their approach gives good practical results however it doesn't give a rigorous proof that a sufficient precision is selected for all inputs in the computation of GCD of approximated polynomials. In [15], Owen, Rockwood and Alyn give a numerical algorithm for **reduced** space curve using subdivision method. Their algorithm has a good complexity but the topology around the singularities of the space curve is not certified. We also mention the work in [11], where two projections of a **reduced** space curve are used, and where the connection algorithm is valid under genericity conditions. In [6], Elkoui gives a certified symbolic-numeric algorithm for space curve defined as the intersection of the vanishing sets of n trivariate polynomials. The main differences between our approach and his one is:

1. his algorithm applies to any number of polynomials but requires the computation of generators of the radical of the ideal, which involves Gröbner basis computation. Our algorithm involves only subresultant computations and applies directly to curves defined by 2 polynomials, which ideal is not necessarily radical.
2. his notion of genericity is more restrictive than ours.

We present a certified algorithm that computes the topology of **non-reduced** algebraic space curves without any Gröbner basis computation. We compute the topology of a plane projection of the space curve and then we lift the computed topology in the three dimensional space. The topology of the projected curve is computed using a classical sweeping algorithm (see [14], [12]). For the computation of the topology of a plane algebraic curve, we present an **efficient generic test** that certifies the output of the algorithm in [12].

For space curves, we introduce the notion of pseudo-generic position. A space curve is said to be in pseudo-generic position with respect to the (x, y) -plane if and only if almost every point of its projection on the (x, y) -plane has only one geometric inverse-image. A simple algebraic criterion is given to certify the pseudo-genericity of the position of a space curve. From a theoretical point of view, the use of the notion of curve in **pseudo-generic position gives us rational parametrizations of the space curve**. The use

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

of these rational parametrizations allows us to lift the topology computed after projection without any supplementary effort. From a practical point of view, the use of the rational parametrizations of the space curve makes the lifting faster, avoiding numerical problems.

We need to distinguish two kinds of singularities on the projected curve. A **certified** algorithm is given to do so.

Unlike previous approaches, our algorithm uses **only one projection** of the space curve and works for **non-reduced space curves**. We therefore avoid the cost of the second projection used by previous approaches.

In the next section we describe the fundamental algebraic tools that we use in this paper. In Section 2, we present our contribution to certify the algorithm for computing the topology of a plane algebraic curve. Our algorithm itself is introduced in Section 3. We report on our implementation and experiments in section 4.

1. SUBRESULTANTS

Let $P_1, P_2 \in \mathbb{Q}[X, Y, Z]$ and $\mathcal{C}_{\mathbb{R}} := \{(x, y, z) \in \mathbb{R}^3 \mid P_1(x, y, z) = P_2(x, y, z) = 0\}$ be the intersection of the vanishing sets of P_1 and P_2 . Our curve analysis needs to compute a plane projection of $\mathcal{C}_{\mathbb{R}}$. Subresultant sequences are a suitable tool to do it. For the reader's convenience, we recall their definition and relevant properties. For all the results of this section, we refer to [4], for proofs.

Let $\mathbb{A}$ be an integral domain. Let $P = \sum_{i=0}^p a_i X^i$ and $Q = \sum_{i=0}^q b_i X^i$ be two polynomials with coefficients in $\mathbb{A}$. We shall always assume $a_p \neq 0$, $b_q \neq 0$ and $p \geq q$.

Let $\mathbb{P}_r(\mathbb{A})$ be the set of polynomials in $\mathbb{A}[X]$ of degree not exceeding r , with the basis (as an $\mathbb{A}$ -module) $1, X, \dots, X^r$. If $r < 0$, we set $\mathbb{P}_r(\mathbb{A}) = \{0\}$ by convention. We will identify an element $S = s_0 + \dots + s_r X^r$ of $\mathbb{P}_r(\mathbb{A})$ with the row vector $(s_0, \dots, s_r)$.

Let k be an integer such that $0 \leq k \leq q$, and let Ψ_k :

$$\mathbb{P}_{q-k-1}(\mathbb{A}) \times \mathbb{P}_{p-k-1}(\mathbb{A}) \rightarrow \mathbb{P}_{p+q-k-1}(\mathbb{A})$$

be the $\mathbb{A}$ -linear map defined by $\Psi_k(U, V) = PU + QV$, with $M_k(P, Q)$ the $(p+q-k) \times (p+q-k)$ matrix of Ψ_k . As we write vectors as row vectors, we have

$$M_k(P, Q) = \begin{pmatrix} a_0 & \dots & a_p & & & \\ & \ddots & & \ddots & & \\ & & a_0 & \dots & a_p & \\ b_0 & \dots & b_q & & & \\ & \ddots & & \ddots & & \\ & & b_0 & \dots & b_q & \end{pmatrix}$$

That is $M_0(P, Q)$ is the classical Sylvester matrix associated to P, Q . To be coherent with the degree of polynomials, we will attach index $i-1$ to the i^{th} column of $M_k(P, Q)$, so the indices of the columns go from 0 to $p+q-k-1$.

Definition 1 For $j \leq p+q-k-1$ and $0 \leq k \leq q$, let $\text{sr}_{k,j}$ be the determinant of the submatrix of $M_k(P, Q)$ formed by the last $p+q-2k-1$ columns, the column of index j and all the $(p+q-2k)$ rows. The polynomial $\text{Sr}_k(P, Q) = \text{sr}_{k,0} + \dots + \text{sr}_{k,k} X^k$ is the k^{th} sub-GCD of P and Q , and its leading term $\text{sr}_{k,k}$ (also denoted sr_k) is the k^{th} subresultant of P and Q . So, it follows that $\text{Sr}_0(P, Q) = \text{sr}_0$ is the usual resultant of P and Q .

Remark 1

1. For $k < j \leq p+q-k-1$, we have $\text{sr}_{k,j} = 0$, because it is the determinant of a matrix with two equal columns.
2. If $q < p$, we have $\text{Sr}_q = (b_q)^{p-q-1}Q$ and $\text{sr}_q = (b_q)^{p-q}$.

The following proposition will justify the name of sub-GCD given to the polynomial Sr_k .

Proposition 1 Let d be the degree of the GCD of P and Q (d is defined because $\mathbb{A}$ is an integral domain, so we may compute the GCD over the quotient field of $\mathbb{A}$). Let k be an integer such that $k \leq d$.

1. The following assertions are equivalent:

- a) $k < d$;
- b) $\text{Sr}_k = 0$;
- c) $\text{sr}_k = 0$.

2. $\text{sr}_d \neq 0$ and Sr_d is the GCD of P and Q .

Theorem 1 Fundamental property of subresultants
The first polynomial Sr_k associated to P and Q with $\text{sr}_k \neq 0$ is the greatest common divisor of P and Q .

We will often call $(\text{Sr}_i)_i$ the subresultant sequence associated to P and Q and $(\text{sr}_{i,j})_{i,j}$ the sequence of their subresultants coefficients. We will denote by $\text{lcoef}_X(f)$ the leading coefficient of the polynomial f with respect to the variable X .

Theorem 2 Specialization property of subresultants
Let $P_1, P_2 \in \mathbb{A}[Y, Z]$ and $(\text{Sr}_i(Y, Z))_i$ be their subresultant sequence with respect to Z . Then for any $\alpha \in \mathbb{A}$ with:
 $\deg_Z(P(Y, Z)) = \deg_Z(P(\alpha, Z))$;
 $\deg_Z(Q(Y, Z)) = \deg_Z(Q(\alpha, Z))$;
 $(\text{Sr}_i(\alpha, Z))_i$ is the subresultant sequence of the polynomials $P(\alpha, Z)$ and $Q(\alpha, Z)$.

2. TOPOLOGY OF A PLANE ALGEBRAIC CURVE

Let $f \in \mathbb{Q}[X, Y]$ be a square free polynomial and

$$\mathcal{C}(f) := \{(\alpha, \beta) \in \mathbb{R}^2, f(\alpha, \beta) = 0\} \quad (1)$$

be the real algebraic curve associated to f . We want to compute the topology of $\mathcal{C}(f)$.

For curves in generic position, computing its critical fibers and one regular fiber between two critical ones is sufficient to obtain the topology using a sweeping algorithm (see [12]). But for a good computational behaviour, it is essential to certify the genericity of the position of the curve.

We propose an effective test allowing to certify the computation and connection, in a deterministic way. This is an important tool in order to address the case of space curves.

Now, let us introduce the definitions of generic position, critical, singular and regular points.

Definition 2 Let $f \in \mathbb{Q}[X, Y]$ be a square free polynomial and $\mathcal{C}(f) = \{(\alpha, \beta) \in \mathbb{R}^2 : f(\alpha, \beta) = 0\}$ be the curve defined by f . A point $(\alpha, \beta) \in \mathcal{C}(f)$ is called:

- a x -critical point if $\partial_Y f(\alpha, \beta) = 0$,

- a singular point if $\partial_X f(\alpha, \beta) = \partial_Y f(\alpha, \beta) = 0$,
- a regular point if $\partial_X f(\alpha, \beta) \neq 0$ or $\partial_Y f(\alpha, \beta) \neq 0$.

With these definitions we can describe the generic conditions required for plane curves.

Definition 3 Let $f \in \mathbb{Q}[X, Y]$ be a square free polynomial and $\mathcal{C}(f) = \{(\alpha, \beta) \in \mathbb{R}^2 : f(\alpha, \beta) = 0\}$ be the curve defined by f . Let $\mathcal{N}_x(\alpha) := \#\{\beta \in \mathbb{R}, \text{ such that } (\alpha, \beta) \text{ is a } x\text{-critical point of } \mathcal{C}(f)\}$. $\mathcal{C}(f)$ is in generic position for the x -direction, if:

1. $\forall \alpha \in \mathbb{C}, \mathcal{N}_x(\alpha) \leq 1$,
2. There is no asymptotic direction of $\mathcal{C}(f)$ parallel to the y -axis.

This notion of genericity also appears in [12] or [16]. In [12], the algorithm succeed if genericity conditions are satisfied. The authors give a numerical test that do **not guarantee to reject the curve** if it is not in generic position. So for some input curves the computed topology might not be exact.

A change of coordinates such that $\text{lcoef}_Y(f) \in \mathbb{Q}^*$ is sufficient to place $\mathcal{C}(f)$ in a position such that any asymptotic direction is not parallel to the y -axis. It remains to find an efficient way to verify the first condition. Using the next propositions, we give an algorithm to do so. We refer to [12], for proofs.

Proposition 2 Let $f \in \mathbb{Q}[X, Y]$ be a square free polynomial with $\text{lcoef}_Y(f) \in \mathbb{Q}^*$, $\text{Res}_Y(f, \partial_Y f)$ be the resultant with respect to Y of the polynomials f , $\partial_Y f$ and $\{\alpha_1, \dots, \alpha_l\}$ be the set of the roots of $\text{Res}_Y(f, \partial_Y f)$ in $\mathbb{C}$.

Then $\mathcal{C}(f)$ is in generic position if and only if $\forall i \in \{1, \dots, l\}$, $\gcd(f(\alpha_i, Y), \partial_Y f(\alpha_i, Y))$ has at most one root.

Let $f \in \mathbb{Q}[X, Y]$ be a square free polynomial with $\text{lcoef}_Y(f) \in \mathbb{Q}^*$ and $d := \deg_Y(f)$. We denote by $\text{Sr}_i(X, Y)$ the i^{th} subresultant polynomial of f and $\partial_Y f$ and $\text{sr}_{i,j}(X)$ the coefficient of Y^j in $\text{Sr}_i(X, Y)$. We define inductively the following polynomials:

$$\Phi_0(X) = \frac{\text{sr}_{0,0}(X)}{\gcd(\text{sr}_{0,0}(X), \text{sr}'_{0,0}(X))};$$

$$\forall i \in \{1, \dots, d-1\}, \Phi_i(X) = \gcd(\Phi_{i-1}(X), \text{sr}_{i,i}(X)) \text{ and } \Gamma_i(X) = \frac{\Phi_{i-1}(X)}{\Phi_i(X)}.$$

Proposition 3

1. $\Phi_0(X) = \prod_{i=1}^{d-1} \Gamma_i(X)$ and $\forall i, j \in \{1, \dots, d-1\}, i \neq j \implies \gcd(\Gamma_i(X), \Gamma_j(X)) = 1$;
2. Let $k \in \{1, \dots, d-1\}$, $\alpha \in \mathbb{C}$. $\Gamma_k(\alpha) = 0$ if and only if $\gcd(f(\alpha, Y), \partial_Y f(\alpha, Y)) = \text{Sr}_k(\alpha, Y)$;
3. $\{(\alpha, \beta) \in \mathbb{R}^2 : f(\alpha, \beta) = \partial_Y f(\alpha, \beta) = 0\} = \bigcup_{k=1}^{d-1} \{(\alpha, \beta) \in \mathbb{R}^2 : \Gamma_k(\alpha) = \text{Sr}_k(\alpha, \beta) = 0\}$.

In the following theorem, we give an effective and efficient algebraic test to certify the genericity of the position of a curve with respect to a given direction.

Theorem 3 Let $f \in \mathbb{Q}[X, Y]$ be a square free polynomial such that $\deg_Y(f) = d$, $\text{lcoef}_Y(f) \in \mathbb{Q}^*$. Then $\mathcal{C}(f)$ is in generic position for the projection on the x axis if and only if $\forall k \in \{1, \dots, d-1\}, \forall i \in \{0, \dots, k-1\}$, $k(k-i) \text{sr}_{k,i}(X) \text{sr}_{k,k}(X) - (i+1) \text{sr}_{k,k-1}(X) \text{sr}_{k,i+1}(X) = 0 \pmod{\Gamma_k(X)}$.

PROOF. Assume that $\mathcal{C}(f)$ is in generic position and let $\alpha \in \mathbb{C}$ be a root of $\Gamma_k(X)$. According to Proposition 3 (2) $\gcd(f(\alpha, Y), \partial_Y f(\alpha, Y)) = \text{Sr}_k(\alpha, Y) = \sum_{j=0}^k \text{sr}_{k,j}(\alpha) Y^j$. According to Proposition 2, $\text{Sr}_k(\alpha, Y)$ has only one root $\beta(\alpha) = -\frac{\text{sr}_{k,k-1}(\alpha)}{k \text{sr}_{k,k}(\alpha)}$, so $\text{Sr}_k(\alpha, Y) = \text{sr}_{k,k}(\alpha)(Y - \beta)^k$.

Binomial Newton formula gives

$$\text{Sr}_k(\alpha, Y) = \text{sr}_{k,k}(\alpha)(Y - \beta)^k = \text{sr}_{k,k}(\alpha) \sum_{i=0}^k \binom{k}{i} (-\beta)^{k-i} Y^i.$$

So by identification $\forall k \in \{1, \dots, d-1\}, \forall i \in \{0, \dots, k-1\}$ and $\forall \alpha \in \mathbb{C}$ such that $\Gamma_k(\alpha) = 0$,

$$k(k-i) \text{sr}_{k,i}(\alpha) \text{sr}_{k,k}(\alpha) - (i+1) \text{sr}_{k,k-1}(\alpha) \text{sr}_{k,i+1}(\alpha) = 0.$$

It is to say that $\forall k \in \{1, \dots, d-1\}, \forall i \in \{0, \dots, k-1\}$, $k(k-i) \text{sr}_{k,i}(X) \text{sr}_{k,k}(X) - (i+1) \text{sr}_{k,k-1}(X) \text{sr}_{k,i+1}(X) = 0 \pmod{\Gamma_k(X)}$.

Conversely, let α be a root of $\Gamma_k(X)$ such that

$$k(k-i) \text{sr}_{k,i}(\alpha) \text{sr}_{k,k}(\alpha) - (i+1) \text{sr}_{k,k-1}(\alpha) \text{sr}_{k,i+1}(\alpha) = 0.$$

With the same argument used in the first part of this proof we obtain

$$\begin{aligned} \gcd(f(\alpha, Y), \partial_Y f(\alpha, Y)) &= \text{Sr}_k(\alpha, Y) \\ &= \sum_{j=0}^k \text{sr}_{k,j}(\alpha) Y^j \\ &= \text{sr}_{k,k}(\alpha)(Y - \beta)^k \end{aligned} \quad (2)$$

with

$$\beta = -\frac{\text{sr}_{k,k-1}(\alpha)}{k \text{sr}_{k,k}(\alpha)}. \quad (3)$$

Then we conclude that $\gcd(f(\alpha, Y), \partial_Y f(\alpha, Y))$ has only one distinct root and, according to Proposition 2, $\mathcal{C}(f)$ is in generic position. $\square$

Remark 2 Theorem 3 shows that it is possible to check with certainty if a plane algebraic curve is in generic position or not. If not, we can put it in generic position by a basis change.

In fact, it is well known that there is only a finite number of bad changes of coordinates of the form $X := X + \lambda Y$, $Y := Y$, such that if $\mathcal{C}(f)$ is not in generic position then the transformed curve remains in a non-generic position (see eg. [12]).

3. TOPOLOGY OF IMPLICIT THREE DIMENSIONAL ALGEBRAIC CURVES

3.1 Description of the problem

Let $P_1, P_2 \in \mathbb{Q}[X, Y, Z]$ and

$$\mathcal{C}_{\mathbb{R}} := \{(x, y, z) \in \mathbb{R}^3 : P_1(x, y, z) = P_2(x, y, z) = 0\} \quad (4)$$

be the intersection of the surfaces defined by $P_1 = 0$ and $P_2 = 0$. We assume that $\gcd(P_1, P_2) = 1$ so that $\mathcal{C}_{\mathbb{R}}$ is a space curve. The ideal (P_1, P_2) is not necessary radical.

Definition 4 [Non-reduced space curve]

The space curve $\mathcal{C}_{\mathbb{R}}$ is **non-reduced** if the ideal generated by P_1 and P_2 is not radical.

Our goal is to analyze the geometry of $\mathcal{C}_R$ in the following sense: We want to compute a piecewise linear graph of $\mathbb{R}^3$ isotopic to the original space curve.

Our method allows to use a new sweeping algorithm using only one projection of the space curve.

To make the lifting possible using only one projection, a new definition of generic position for space curves and an algebraic characterization of it are given. We will also need to distinguish the "apparent singularities" and the "real singularities". A **certified** algorithm is given to **distinguish** these two kinds of **singularities**.

For the lifting phase, using the new notion of curve in **pseudo-generic position**, we give an algorithm that computes rational parametrizations of the space curve. The use of these rational parametrizations allows us to lift the topology of the projected curve without any supplementary computation.

3.2 Genericity conditions for space curves

Let $\Pi_z : (x, y, z) \in \mathbb{R}^3 \mapsto (x, y) \in \mathbb{R}^2$. We still denote $\Pi_z = \Pi_z|_{\mathcal{C}_R}$. Let $\mathcal{D} = \Pi_z(\mathcal{C}_R) \subset \mathbb{R}^2$ be the curve obtained by projection of $\mathcal{C}_R$.

We assume that $\deg_Z(P_1) = \deg(P_1)$ and $\deg_Z(P_2) = \deg(P_2)$ (by a basis change, these conditions are always satisfied). Let $h(X, Y)$ be the **squarefree** part of $\text{Res}_Z(P_1, P_2) \in \mathbb{Q}[X, Y]$.

Definition 5 [Pseudo-generic position]

Let $\mathcal{C}_\mathbb{C} := \{(x, y, z) \in \mathbb{C}^3 | P_1(x, y, z) = P_2(x, y, z) = 0\}$. The curve $\mathcal{C}_R$ is in pseudo-generic position with respect to the (x, y) -plane if and only if almost every point of $\Pi_z(\mathcal{C}_\mathbb{C})$ has only one geometric inverse-image, i.e. generically, if $(\alpha, \beta) \in \Pi_z(\mathcal{C}_\mathbb{C})$, then $\Pi_z^{-1}(\alpha, \beta)$ consists in one point possibly multiple.

Let m be the minimum of $\deg_Z(P_1)$ and $\deg_Z(P_2)$. The following theorems give us an effective way to test if a curve is in pseudo-generic position or not.

Theorem 4 Let $(\text{Sr}_j(X, Y, Z))_{j \in \{0, \dots, m\}}$ be the subresultant sequence and $(\text{sr}_j(X, Y))_{j \in \{0, \dots, m\}}$ be the principal subresultant coefficient sequence. Let $(\Delta_i(X, Y))_{i \in \{1, \dots, m\}}$ be the sequence of $\mathbb{Q}[X, Y]$ defined by the following relations

- $\Delta_0(X, Y) = 1; \Theta_0(X, Y) = h(X, Y);$
- For $i \in \{1, \dots, m\}$,
 $\Theta_i(X, Y) = \gcd(\Theta_{i-1}(X, Y), \text{sr}_i(X, Y)),$
 $\Delta_i(X, Y) = \frac{\Theta_{i-1}(X, Y)}{\Theta_i(X, Y)}.$

For $i \in \{1, \dots, m\}$, let $\mathcal{C}(\Delta_i) := \{(x, y) \in \mathbb{R}^2 | \Delta_i(x, y) = 0\}$ and $\mathcal{C}(h) := \{(x, y) \in \mathbb{R}^2 | h(x, y) = 0\}$ then

1. $h(X, Y) = \prod_{i=1}^m \Delta_i(X, Y),$
2. $\mathcal{C}(h) = \bigcup_{i=1}^m \mathcal{C}(\Delta_i),$
3. $\mathcal{C}_R$ is in pseudo-generic position with respect to the (x, y) -plane if and only if $\forall i \in \{1, \dots, m\}, \forall (x, y) \in \mathbb{C}^2$ such that $\text{sr}_{i,i}(x, y) \neq 0$ and $\Delta_i(x, y) = 0$, we have
 $\text{Sr}_i(x, y, Z) = \text{sr}_{i,i}(x, y) \left(Z + \frac{\text{sr}_{i,i-1}(x, y)}{i \text{sr}_{i,i}(x, y)} \right)^i.$

PROOF. 1. By definition, $\forall i \in \{1, \dots, m\}$,
 $\Delta_i(X, Y) = \frac{\Theta_{i-1}(X, Y)}{\Theta_i(X, Y)}.$ So by a trivial induction

$$\prod_{i=1}^m \Delta_i(X, Y) = \frac{\Theta_0(X, Y)}{\Theta_m(X, Y)}.$$

$\deg_Z(P_1) = \deg(P_1)$ and $\deg_Z(P_2) = \deg(P_2)$ imply $\text{sr}_m(X, Y) \in \mathbb{Q}^*$ (see Remark 1).

So $\Theta_m(X, Y) = \gcd(\Theta_{m-1}(X, Y), \text{sr}_m(X, Y)) = 1$, then $\prod_{i=1}^m \Delta_i(X, Y) = \Theta_0(X, Y) = h(X, Y).$

2. Knowing that $h(X, Y) = \prod_{i=1}^m \Delta_i(X, Y)$, it is clear that

$$\mathcal{C}(h) = \bigcup_{i=1}^m \mathcal{C}(\Delta_i).$$

3. Assume that $\mathcal{C}_R$ is in pseudo-generic position with respect to the (x, y) -plane. Let $i \in \{1, \dots, m\}$ and $(\alpha, \beta) \in \mathbb{C}^2$ such that $\text{sr}_i(\alpha, \beta) \neq 0$ and $\Delta_i(\alpha, \beta) = 0$. Then $\Delta_i(X, Y) = \frac{\Theta_{i-1}(X, Y)}{\Theta_i(X, Y)} \implies \Theta_{i-1}(\alpha, \beta) = 0$. Knowing that $\Theta_{i-1}(X, Y) = \gcd(\Theta_{i-2}(X, Y), \text{sr}_{i-1}(X, Y))$, so it exists $d_1, d_2 \in \mathbb{Q}[X, Y]$ such that $\Theta_{i-2}(X, Y) = d_1(X, Y)\Theta_{i-1}(X, Y)$ and $\text{sr}_{i-1}(X, Y) = d_2(X, Y)\Theta_{i-1}(X, Y)$. In this way, $\Theta_{i-1}(\alpha, \beta) = 0 \implies \Theta_{i-2}(\alpha, \beta) = 0$ and $\text{sr}_{i-1}(\alpha, \beta) = 0$. By using the same arguments, $\Theta_{i-2}(\alpha, \beta) = 0 \implies \Theta_{i-3}(\alpha, \beta) = 0$ and $\text{sr}_{i-2}(\alpha, \beta) = 0$. By repeating the same argument, we show $\text{sr}_{i-1}(\alpha, \beta) = \dots = \text{sr}_0(\alpha, \beta) = 0$. Because $\text{sr}_i(\alpha, \beta) \neq 0$, then the fundamental theorem of subresultant gives

$\gcd((P_1(\alpha, \beta, Z), P_2(\alpha, \beta, Z))) = \text{Sr}_i(\alpha, \beta, Z) = \sum_{j=0}^i \text{sr}_{i,i-j}(\alpha, \beta) Z^{i-j}$. Knowing that $\mathcal{C}_R$ is in pseudo-generic position with respect to the (x, y) -plane and $\Delta_i(\alpha, \beta) = 0$ then the polynomial $\text{Sr}_i(\alpha, \beta, Z)$ has only one distinct root which can be written $-\frac{\text{sr}_{i,i-1}(\alpha, \beta)}{i \text{sr}_{i,i-1}(\alpha, \beta)}$ depending on the relation between coefficients and roots of a polynomial. So $\text{Sr}_i(\alpha, \beta, Z) = \sum_{j=0}^m \text{sr}_{i,i-j}(\alpha, \beta) Z^{i-j} =$

$$\text{sr}_{i,i}(\alpha, \beta) \left(Z + \frac{\text{sr}_{i,i-1}(\alpha, \beta)}{i \text{sr}_{i,i-1}(\alpha, \beta)} \right)^i.$$

Conversely, assume that $\forall i \in \{1, \dots, m\}, \forall (x, y) \in \mathbb{C}^2$ such that $\text{sr}_i(x, y) \neq 0$ and $\Delta_i(x, y) = 0$, we have

$$\text{Sr}_i(x, y, Z) = \sum_{j=0}^m \text{sr}_{i,i-j}(x, y) Z^{i-j} =$$

$\text{sr}_{i,i}(x, y) \left(Z + \frac{\text{sr}_{i,i-1}(x, y)}{i \text{sr}_{i,i}(x, y)} \right)^i$. Let (α, β) be a point such that $\Delta_i(\alpha, \beta) = 0$ and $\text{sr}_i(\alpha, \beta) \neq 0$. Now if we define $\gamma := -\frac{\text{sr}_{i,i-1}(\alpha, \beta)}{i \text{sr}_{i,i}(\alpha, \beta)}$, then we obtain that $\text{Sr}_i(\alpha, \beta, \gamma) = 0$, and (α, β, γ) is the only point of $\mathcal{C}_\mathbb{C}$ with (α, β) as projection. Furthermore there are only finitely many points such that $\Delta_i(x, y) = 0$ and $\text{sr}_i(x, y) = 0$. So $\mathcal{C}_R$ is in pseudo-generic position with respect to the (x, y) -plane.

□

The following proposition is a corollary of the third result of the previous theorem. If $\mathcal{C}_R$ is in pseudo-generic position with respect to the (x, y) -plane, it gives a rational parametrization for the regular points of $\mathcal{C}_R$.

Proposition 4 Assume that $\mathcal{C}_R$ is in pseudo-generic position with respect to the (x, y) -plane and let $(\alpha, \beta, \gamma) \in \mathcal{C}_R$

such that $\text{sr}_i(\alpha, \beta) \neq 0$ and $\Delta_i(\alpha, \beta) = 0$. Then,

$$\gamma := -\frac{\text{sr}_{i,i-1}(\alpha, \beta)}{i \text{sr}_{i,i}(\alpha, \beta)}. \quad (5)$$

Remark 3 By construction, the parametrization given in Proposition 4 is valid when $\text{sr}_{i,i}(\alpha, \beta) \neq 0$. In pseudo-generic position, if $\text{sr}_{i,i}(\alpha, \beta) = 0$ then either $\Delta_j(\alpha, \beta) = 0$ for some $j > i$ or (α, β) is a x -critical point of $\mathcal{C}(\Delta_i)$ (see section 3.3).

The following theorem gives an algebraic test to certify the pseudo-genericity of the position of a space curve with respect to a given plane.

Theorem 5 Let $(\text{Sr}_j(X, Y, Z))_{j \in \{0, \dots, m\}}$ be the subresultant sequence associated to $P_1(X, Y, Z)$ and $P_2(X, Y, Z)$ and $(\Delta_i(X, Y))_{i \in \{1, \dots, m\}}$ be the sequence of $\mathbb{Q}[X, Y]$ previously defined. The curve $\mathcal{C}_{\mathbb{R}}$ is in pseudo-generic position with respect to the (x, y) -plane if and only if

$$\forall i \in \{1, \dots, m-1\}, \forall j \in \{0, \dots, i-1\},$$

$$i(i-j) \text{sr}_{i,j}(X, Y) \text{sr}_{i,i}(X, Y) - (j+1) \text{sr}_{i,i-1}(X, Y) \text{sr}_{i,j+1}(X, Y) = 0 \bmod \Delta_i(X, Y).$$

PROOF. Assume $\mathcal{C}_{\mathbb{R}}$ be in pseudo-generic position. Let $i \in \{1, \dots, m-1\}$, $j \in \{0, \dots, i-1\}$, $(\alpha, \beta) \in \mathbb{R}^2$ such that $\Delta_i(\alpha, \beta) = 0$.

If $\text{sr}_{i,i}(\alpha, \beta) = 0$, then, by Proposition 1, $\text{sr}_{i,i-1}(\alpha, \beta) = 0$, so $i(j+1) \text{sr}_{i,j+1}(\alpha, \beta) \text{sr}_{i,i}(\alpha, \beta) - (i-j) \text{sr}_{i,i-1}(\alpha, \beta) \text{sr}_{i,j}(\alpha, \beta) = 0$.

If $\text{sr}_{i,i}(\alpha, \beta) \neq 0$, then according to Theorem 4 (3)

$$\text{Sr}_i(\alpha, \beta, Z) = \text{sr}_{i,i}(\alpha, \beta) (Z - \gamma)^i \text{ where } \gamma := -\frac{\text{sr}_{i,i-1}(\alpha, \beta)}{i \text{sr}_{i,i}(\alpha, \beta)},$$

$$\text{then } \text{Sr}_i(\alpha, \beta, Z) = \sum_{j=0}^i \text{sr}_{i,i-j}(\alpha, \beta) Z^{i-j} = \text{sr}_{i,i}(\alpha, \beta) (Z - \gamma)^i.$$

Using the binomial Newton formula we obtain $\text{Sr}_i(\alpha, \beta, Z) = \text{sr}_{i,i}(\alpha, \beta) \sum_{j=0}^i \binom{i}{j} (-\gamma)^{i-j} Z^j$. So by identification, it comes that

$$\forall i \in \{1, \dots, m-1\}, \forall j \in \{0, \dots, i-1\}, \forall (\alpha, \beta), \text{st. } \Delta_i(\alpha, \beta) = 0$$

$$i(i-j) \text{sr}_{i,j}(\alpha, \beta) \text{sr}_{i,i}(\alpha, \beta) - (i+j) \text{sr}_{i,i-1}(\alpha, \beta) \text{sr}_{i,j+1}(\alpha, \beta) = 0. \text{ The reciprocal uses the same arguments. } \square$$

Remark 4 Theorem 5 shows that it is possible to check with certainty if a space algebraic curve is in pseudo-generic position or not. If it is not, we can put it in pseudo-generic position by a change of coordinates.

Let us introduce the definitions of generic position, critical, singular, regular points, apparent singularity and real singularity for a space algebraic curve.

Definition 6 Let $(g_1, \dots, g_s)$ be the radical ideal of the ideal (P_1, P_2) . Let $M(X, Y, Z)$ be the $s \times 3$ Jacobian matrix with $(\partial_X g_i, \partial_Y g_i, \partial_Z g_i)$ as its i th row.

1. A point $p \in \mathcal{C}_{\mathbb{R}}$ is regular (or smooth) if the rank of $M(p)$ is 2.
2. A point $p \in \mathcal{C}_{\mathbb{R}}$ which is not regular is called singular.
3. A point $p = (\alpha, \beta, \gamma) \in \mathcal{C}_{\mathbb{R}}$ is x -critical (or critical for the projection on the x -axis) if the curve $\mathcal{C}_{\mathbb{R}}$ is tangent at this point to a plane parallel to the (y, z) -plane. The corresponding α is called a x -critical value.

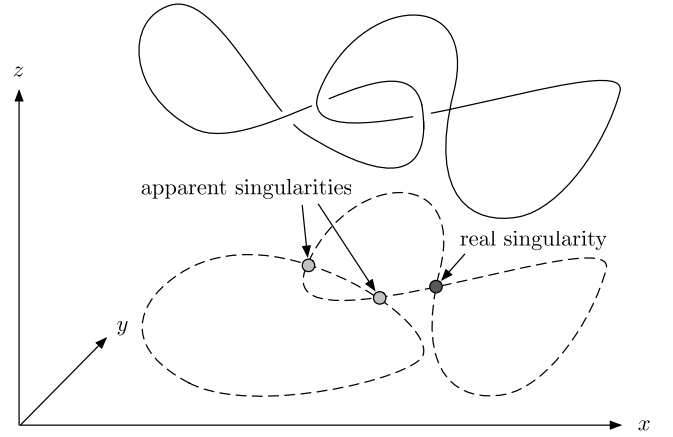


Figure 1: Apparent and real singularities.

Definition 7 [Apparent singularity, Real singularity]
We call:

1. Apparent singularities: the singularities of the projected curve $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ with at least two points as inverse-images (see figure 1).
2. Real singularities: the singularities of the projected curve $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ with exactly one point as inverse-image (see figure 1).

Definition 8 [Node]

We call a node an ordinary double point (both arcs have different tangential directions).

Definition 9 [Generic position]

The curve $\mathcal{C}_{\mathbb{R}}$ is in generic position with respect to the (x, y) -plane if and only if

1. $\mathcal{C}_{\mathbb{R}}$ is in pseudo-generic position with respect to the (x, y) -plane,
2. $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ is in generic position (as a plane algebraic curve) with respect to the x -direction,
3. any apparent singularity of $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ is a node.

This notion of genericity also appears in a slightly more restrictive form in [2] and [6].

The aim of the next section is to give an algorithm to certify the third point of the previous definition of generic position. We give also in this section an effective way to distinguish the real singularities from the apparent ones.

3.3 Distinguish real singularities and apparent singularities

In this section, we suppose that $\mathcal{C}_{\mathbb{R}}$ is in pseudo-generic position and $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ is in generic position as a plane algebraic curve.

Let $(\Gamma_j(X))_{j \in \{1, \dots, n\}}$ be the sequence of Γ polynomials associated to the plane curve $\mathcal{D}$ and $(\beta_j(X))_{j \in \{1, \dots, n\}}$ be the sequence of associated rational parametrization (see (3)). Let $(\text{Sr}_j(X, Y, Z))_{j \in \{0, \dots, m\}}$ be the subresultant sequence associated to $P_1, P_2 \in \mathbb{Q}[X, Y, Z]$. For any $(k, i) \in \{1, \dots, m\} \times \{0, \dots, k-1\}$ let, $R_{k,i}(X, Y)$ be the polynomial $k(k-i) \text{sr}_{k,i}(X, Y) \text{sr}_{k,k} - (i+1) \text{sr}_{k,k-1}(X, Y) \text{sr}_{k,i+1}(X, Y)$.

Lemma 1 Let $(a, b) \in \mathbb{R}^2$ such that $\text{sr}_{k,k}(a, b) \neq 0$, the polynomial $\text{Sr}_k(a, b, Z) = \sum_{i=0}^k \text{sr}_{k,i}(a, b) Z^i \in \mathbb{R}[Z]$ has one and only one root if and only if $\forall i \in \{0, \dots, k-1\} R_{k,i}(a, b) = 0$.

For any $j \in \{1, \dots, n\}$ we define the sequences

$$\begin{aligned} (u_{k,j}(X))_{k \in \{1, \dots, j\}} \text{ and } (v_{k,j}(X))_{k \in \{2, \dots, j\}} \text{ by} \\ u_{1,j}(X) &:= \gcd(\Gamma_j(X), \text{sr}_{1,1}(X, \beta_j(X))), \\ u_{k,j}(X) &:= \gcd(\text{sr}_{k,k}(X, \beta_j(X)), u_{k-1,j}(X)) \\ v_{k,j}(X) &:= \text{quo}(u_{k-1,j}(X), u_{k,j}(X)). \end{aligned}$$

For $k \in \{2, \dots, j\}$, $i \in \{0, k-1\}$, we define $(w_{k,i,j}(X))$ by

$$w_{k,0,j}(X) := v_{k,j}(X), \\ w_{k,i+1,j}(X) := \gcd(R_{k,i}(X, \beta_j(X)), w_{k,i,j}(X)).$$

More intuitively, for some j , the polynomials $v_{k,j}$ are exactly those with roots α such that the gcd of the projected plane curve and its derivative, localized at α , has degree j , and the gcd of the two surfaces, localized at $(\alpha, \beta_j(\alpha))$, has degree k .

Theorem 6 For any $j \in \{1, \dots, n\}$, let $(\Gamma_{j,k}(X))_{k \in \{1, \dots, j\}}$ and $(\chi_{j,k}(X))$ be the sequences defined by the following relations:

$$\begin{aligned} \Gamma_{j,1}(X) &= \text{quo}(\Gamma_j(X), u_{1,j}(X)) \text{ and } \Gamma_{j,k}(X) := w_{k,k,j}(X). \\ \chi_{j,k}(X) &:= \text{quo}(w_{k,0,j}(X), \Gamma_{j,k}(X)). \end{aligned}$$

1. For any root α of $\Gamma_{j,k}(X)$, the x -critical fiber $(\alpha, \beta_j(\alpha))$ contains only the point $(\alpha, \beta_j(\alpha), \gamma_j(\alpha))$ with $\gamma_j(\alpha) := -\frac{\text{sr}_{k,k-1}(\alpha, \beta_j(\alpha))}{k \text{sr}_{k,k}(\alpha, \beta_j(\alpha))}$, so $(\alpha, \beta_j(\alpha))$ is a real singularity.
2. For any root α of $\chi_{j,k}(X)$, $(\alpha, \beta_j(\alpha))$ is an apparent singularity.

PROOF. 1. Let α be a root of $\Gamma_{j,k}(X) := w_{k,k,j}(X) = \gcd(R_{k,k-1}(X, \beta_j(X)), w_{k,k-1,j}(X))$. Then $w_{k,k-1,j}(\alpha) = R_{k,k-1}(\alpha, \beta_j(\alpha)) = 0$.

$$w_{k,k-1,j}(X) := \gcd(R_{k,k-2}(X, \beta_j(X)), w_{k,k-2,j}(X)), \text{ so } \\ w_{k,k-2,j}(\alpha) = R_{k,k-2}(\alpha, \beta_j(\alpha)) = 0.$$

By induction, using the same argument, it comes that for i from 0 to $(k-1)$, $w_{k,i,j}(\alpha) = R_{k,i}(\alpha, \beta_j(\alpha)) = 0$.

$w_{k,0,j}(X) := v_{k,j}(X)$, so $v_{k,j}(\alpha) = 0$. Knowing that $v_{k,j}(X) := \text{quo}(u_{k-1,j}(X), u_{k,j}(X))$; $u_{k,j}$ and $u_{k-1,j}$ are square free, then $u_{k-1,j}(\alpha) = 0$ and $u_{k,j}(\alpha) \neq 0$. Knowing that $u_{k,j}(X) = \gcd(\text{sr}_{k,k}(X, \beta_j(X)), u_{k-1,j}(X))$, then $\text{sr}_{k,k}(\alpha, \beta_j(\alpha)) \neq 0$.

$$u_{k-1,j}(X) = \gcd(\text{sr}_{k-1,k-1}(X, \beta_j(X)), u_{k-2,j}(X)) \text{ and } \\ u_{k-1,j}(\alpha) = 0, \text{ so } \text{sr}_{k-1,k-1}(\alpha, \beta_j(\alpha)) = u_{k-2,j}(\alpha) = 0.$$

By induction, using the same argument, it comes that for i from 0 to $k-1$ $\text{sr}_{k,i}(\alpha, \beta_j(\alpha)) = 0$.

For i from 0 to $k-1$ $\text{sr}_{k,i}(\alpha, \beta_j(\alpha)) = 0$ and $\text{sr}_{k,k}(\alpha, \beta_j(\alpha)) \neq 0$, so by the fundamental theorem of subresultants,

$$\begin{aligned} \gcd(P_1(\alpha, \beta_j(\alpha), Z), P_2(\alpha, \beta_j(\alpha), Z)) &= \text{Sr}_k(\alpha, \beta_j(\alpha), Z) \\ &= \sum_{i=0}^k \text{sr}_{k,i}(\alpha, \beta_j(\alpha)) Z^i. \text{ Knowing that} \\ \gcd(P_1(\alpha, \beta_j(\alpha), Z), P_2(\alpha, \beta_j(\alpha), Z)) &= \text{Sr}_k(\alpha, \beta_j(\alpha), Z) \\ &= \sum_{i=0}^k \text{sr}_{k,i}(\alpha, \beta_j(\alpha)) Z^i \text{ and for } i \text{ from } 0 \text{ to } (k-1), \\ R_{k,i}(\alpha, \beta_j(\alpha)) &= 0 \text{ then by the previous lemma the poly-} \\ \text{nomial } \gcd(P_1(\alpha, \beta_j(\alpha), Z), P_2(\alpha, \beta_j(\alpha), Z)) &\text{ have only one} \\ \text{root } \gamma_j(\alpha) &:= -\frac{\text{sr}_{k,k-1}(\alpha, \beta_j(\alpha))}{k \times \text{sr}_{k,k}(\alpha, \beta_j(\alpha))}. \end{aligned}$$

2. Let α be a root of the polynomial

$$\begin{aligned} \chi_{j,k}(X) &:= \text{quo}(w_{k,0,j}(X), \Gamma_{j,k}(X)). \text{ Then } w_{k,0,j}(\alpha) = \\ &0 \text{ and } \Gamma_{j,k}(\alpha) = w_{k,k,j}(\alpha) \neq 0 \text{ because } w_{k,0,j}(X) \text{ and } \\ &\Gamma_{j,k}(X) \text{ are square free. For } i \text{ from } 0 \text{ to } k-1, \text{ know-} \\ &\text{ing that } w_{k,i+1,j}(X) := \gcd(R_{k,i}(X, \beta_j(X)), w_{k,i,j}(X)), \end{aligned}$$

$w_{k,0,j}(\alpha) = 0$ and $w_{k,k,j}(\alpha) \neq 0$, then it exist $i \in \{0, \dots, k-1\}$ such that $R_{k,i}(\alpha, \beta_j(\alpha)) \neq 0$. So by Lemma 1 the polynomial $\text{Sr}_k(\alpha, \beta_j(\alpha), Z) = \sum_{i=0}^k \text{sr}_{k,i}(\alpha, \beta_j(\alpha)) Z^i$ has at least two distinct roots.

It is clear that $\gcd(P_1(\alpha, \beta_j(\alpha), Z), P_2(\alpha, \beta_j(\alpha), Z)) = \text{Sr}_k(\alpha, \beta_j(\alpha), Z) = \sum_{i=0}^k \text{sr}_{k,i}(\alpha, \beta_j(\alpha)) Z^i$. $\gcd(P_1(\alpha, \beta_j(\alpha), Z), P_2(\alpha, \beta_j(\alpha), Z)) = \text{Sr}_k(\alpha, \beta_j(\alpha), Z)$ and $\text{Sr}_k(\alpha, \beta_j(\alpha), Z)$ has at least two distinct roots imply that $(\alpha, \beta_j(\alpha))$ is an apparent singularity. $\square$

Proposition 5 For $(j, k) \in \{1, \dots, n\} \times \{2, \dots, j\}$, let α be a root of $\chi_{j,k}(X)$. The apparent singularity $(\alpha, \beta_j(\alpha))$ is a node if and only if $(\partial_{XY}^2 h(\alpha, \beta_j(\alpha)))^2 - \partial_{X^2}^2 h(\alpha, \beta_j(\alpha)) \partial_{Y^2}^2 h(\alpha, \beta_j(\alpha)) \neq 0$.

PROOF. Let us write $h(X + \alpha, Y + \beta_j(\alpha)) = F_1 + F_2 + \dots + F_n$, where F_i is a form in $\mathbb{Q}[\alpha][X, Y]$ of degree i . The apparent singularity $(\alpha, \beta_j(\alpha))$ is a double point if and only if $F_2 \neq 0$. It is well known that $F_2 = 1/2(\partial_{XY}^2 h(\alpha, \beta_j(\alpha)) Y^2 + 2\partial_{XY}^2 h(\alpha, \beta_j(\alpha)) XY + \partial_{X^2}^2 h(\alpha, \beta_j(\alpha)) X^2)$ and its factorization gives the tangent lines of $\mathcal{D} := \mathcal{C}(h(X, Y))$ at $(\alpha, \beta_j(\alpha))$. So the apparent singularity $(\alpha, \beta_j(\alpha))$ is a node if and only the discriminant in Y of the polynomial $F_2(1, Y)$ is different to 0. $\square$

Theorem 7 $\mathcal{C}_{\mathbb{R}}$ is in generic position if and only if for any $(j, k) \in \{1, \dots, n\} \times \{2, \dots, j\}$ the polynomials $\chi_{j,k}(X)$ and $\partial_{XY}^2 h(X, \beta_j(X))^2 - \partial_{X^2}^2 h(X, \beta_j(X)) \partial_{Y^2}^2 h(X, \beta_j(X))$ are coprime.

PROOF. $\mathcal{C}_{\mathbb{R}}$ is in generic position if and only if any apparent singularity is a node. So the result comes clearly from the previous proposition. $\square$

3.4 Lifting and connection phase

In this section, we suppose that $\mathcal{C}_{\mathbb{R}}$ is in **generic position** that means that $\mathcal{C}_{\mathbb{R}}$ is in **pseudo-generic position**, $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ is in **generic position** as a plane algebraic curve and any **apparent singularity** of $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ is a **node**.

To compute the topology of $\mathcal{C}_{\mathbb{R}}$, we first compute the topology of its projection on the (x, y) -plane and in second we lift the computed topology.

As mentioned in section 2, to compute the topology of a plane algebraic curve in generic position, we need to compute its critical fibers and one regular fiber between two critical ones. So to obtain the topology of $\mathcal{C}_{\mathbb{R}}$, we just need to lift the critical and regular fibers of $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$.

Here after we explain how this lifting can be done without any supplementary computation for the regular fibers and the real critical fibers. And for the special case of the apparent singular fibers, we present a new approach for the lifting and the connections.

3.4.1 Lifting of the regular points of $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$

The lifting of the regular fibers of $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ is done by using the rational parametrizations given in Proposition 4.

3.4.2 Lifting of the real singularities of $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$

The lifting of the real singularities of $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$ is done by using the rational parametrizations given by 1. of Theorem 6.

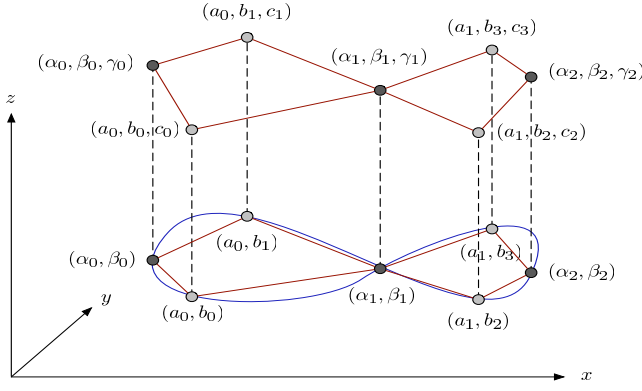


Figure 2: Connection between real singularities and regular points.

3.4.3 Connection between real singularities and regular points

For a space curve in pseudo-generic position, the connections between real singularities and regular points are exactly those obtained on the projected curve using Grandine's sweeping algorithm [12] (see figure 2).

3.4.4 Lifting of the apparent singularities

The lifting of the topology around an apparent singularity is a little more complex. Above an apparent singularity of $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}})$, we first have to compute the z -coordinates and secondly to decide which of the two branches passes over the other (see figure 3). We solve these problems by analyzing the situation at an apparent singularity.

According to Theorem 4 (2), $\mathcal{D} = \Pi_z(\mathcal{C}_{\mathbb{R}}) = \bigcup_{i=1}^m \mathcal{C}(\Delta_i)$, so an apparent singularity is a cross point of a branch of $\mathcal{C}(\Delta_i)$ and a branch of $\mathcal{C}(\Delta_j)$ with $i, j \in \{1, \dots, m\}$. So we have the following proposition.

Proposition 6 *If (α, β) is an apparent singularity of $\mathcal{D}$ such that $\Delta_i(\alpha, \beta) = \Delta_j(\alpha, \beta) = 0$, then the degree of the polynomial $\gcd(P_1(\alpha, \beta, Z), P_2(\alpha, \beta, Z)) \in \mathbb{R}[Z]$ will be $(i + j)$.*

Let (α, β) be an apparent singularity of $\mathcal{D}$ such that $\Delta_i(\alpha, \beta) = \Delta_j(\alpha, \beta) = 0$ and γ_1, γ_2 the corresponding z -coordinates. So by Proposition 6 and Proposition 1 $\text{sr}_{0,0}(\alpha, \beta) = \dots = \text{sr}_{i,i}(\alpha, \beta) = \dots = \text{sr}_{j,j}(\alpha, \beta) = \dots = \text{sr}_{i+j-1,i+j-1}(\alpha, \beta) = 0$. By Proposition 4, for any $(a, b, c) \in \mathcal{C}_{\mathbb{R}}$ such that $\Delta_i(a, b) = 0$ and $\text{sr}_{i,i}(a, b) \neq 0$ we have $c = -\frac{\text{sr}_{i,i-1}(a, b)}{i \text{sr}_{i,i}(a, b)}$. So the function $(x, y) \mapsto Z_i := -\frac{\text{sr}_{i,i-1}(x, y)}{i \text{sr}_{i,i}(x, y)}$ gives the z -coordinate of any $(a, b, c) \in \mathcal{C}_{\mathbb{R}}$ such that $\Delta_i(a, b) = 0$ and $\text{sr}_{i,i}(a, b) \neq 0$. $\Delta_i(\alpha, \beta) = 0$ but $\text{sr}_{i,i}(\alpha, \beta) = 0$, so the function Z_i is not defined on (α, β) . The solution comes from the fact that the function Z_i is continuously extensible on (α, β) . Let u_1 be the slope of the tangent line of $\mathcal{C}(\Delta_i)$ at (α, β) and $t \in \mathbb{R}^*$. Let $\gamma_i(t) := Z_i(\alpha, \beta + tu_1) = -\frac{\text{sr}_{i,i-1}(\alpha, \beta + tu_1)}{i \text{sr}_{i,i}(\alpha, \beta + tu_1)}$. Knowing that the algebraic curve $\mathcal{C}_{\mathbb{R}}$ hasn't any discontinuity, it comes $\lim_{t \rightarrow 0+} \gamma_i(t) = \lim_{t \rightarrow 0-} \gamma_i(t) = \gamma_1$. By the same arguments, if we denote u_2 the slope of the tangent line of $\mathcal{C}(\Delta_j)$ at (α, β) and $\gamma_j(t) := Z_j(\alpha, \beta + tu_2) =$

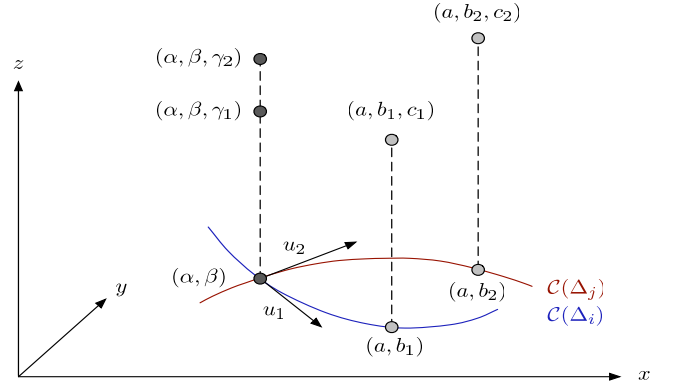


Figure 3: Lifting of an apparent singularity.

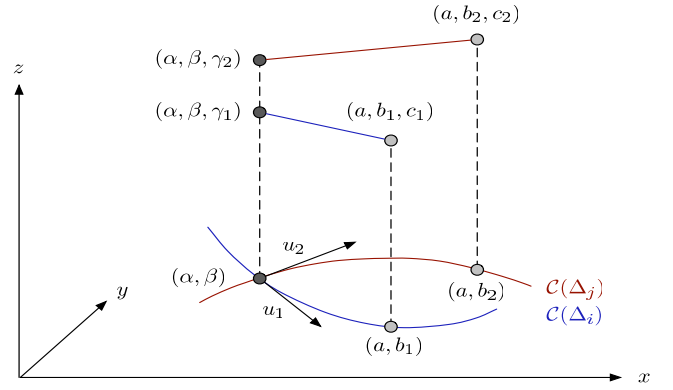


Figure 4: Connection above an apparent singularity.

$-\frac{\text{sr}_{j,j-1}(\alpha, \beta + tu_2)}{j \text{sr}_{j,j}(\alpha, \beta + tu_2)}$, then $\lim_{t \rightarrow 0+} \gamma_j(t) = \lim_{t \rightarrow 0-} \gamma_j(t) = \gamma_2$. The values u_1, u_2, γ_1 and γ_2 are computed using Taylor formulas and certified numerical approximations.

Now it remains to decide which of the two branches pass over the other. This problem is equivalent to the problem of deciding the connection around an apparent singularity. Let (a, b_1, c_1) and (a, b_2, c_2) the regular points that we have to connect to $(\alpha, \beta, \gamma_1)$ and $(\alpha, \beta, \gamma_2)$. The question is which of the points (a, b_1, c_1) and (a, b_2, c_2) will be connected to $(\alpha, \beta, \gamma_1)$ and the other to $(\alpha, \beta, \gamma_2)$ (see figure 3)? In [2] Alcázar and Sendra give a solution using a second projection of the space curve but it costs a computation of a Sturm Habicht sequence of P_1 and P_2 . Our solution does not use any supplementary computation. It comes from the fact that γ_1 is associated to u_1 and γ_2 to u_2 . Knowing that u_1 is the slope of the tangent line of $\mathcal{C}(\Delta_i)$ at (α, β) and u_2 the slope of the tangent line of $\mathcal{C}(\Delta_j)$ at (α, β) , so $(\alpha, \beta, \gamma_1)$ will be connected to (a, b_1, c_1) if (a, b_1) is on the branch associated to u_1 . If (a, b_1) is not on the branch associated to u_1 , then (a, b_1) is on the branch associated to u_2 , so $(\alpha, \beta, \gamma_2)$ will be connected to (a, b_1, c_1) (see figure 4).

Remark 5 For a curve in generic position any apparent singularity is a node, so the slopes at an apparent singularity are always distinct that is to say $u_1 \neq u_2$.

4. IMPLEMENTATION, EXPERIMENTS

A complete implementation of our method has been writ-

Curve	$P_1(x, y, z)$	$P_2(x, y, z)$	Time (s)
1	$x^2 + y^2 + z^2 - 1$	$x^2 - y^2 - z + 1$	0.032
2	$x^2 + y^2 + z^2 - 1$	$x^3 + 3x^2z + 3xz^2 + z^3 + y^3 - xyz - yz^2$	0.659
3	$(x - 2y + 2z)^2 + y^2 + z^2 - 1$	$z^3 - z - (x - 2y + 2z)^3 + 3(x - 2y + 2z)y^2$	2.125
4	$(x - 2y + 2z)^2 + y^2 + z^2 - 1$	$y^3 - (x - 2y + 2z)^3 - (x - 2y + 2z)yz$	1.031
5	$(x - y + z)^2 + y^2 + z^2 - 1$	$y^2 - (x - y + z)^2 - (x - y + z)z^2 - z^2((x - y + z)^2 + y^2)$	1.6963
6	$(x - y + z)^2 + y^2 + z^2 - 1$	$((x - y + z)^2 + y^2 + z^2)^2 - 4((x - y + z)^2 + y^2)$	2.228
7	$(x - y + z)^2 + y^2 - 2(x - y + z)$	$((x - y + z)^2 + y^2 + z^2)^2 - 4((x - y + z)^2 + y^2)$	2.875

Figure 5: Running time of experimentations.

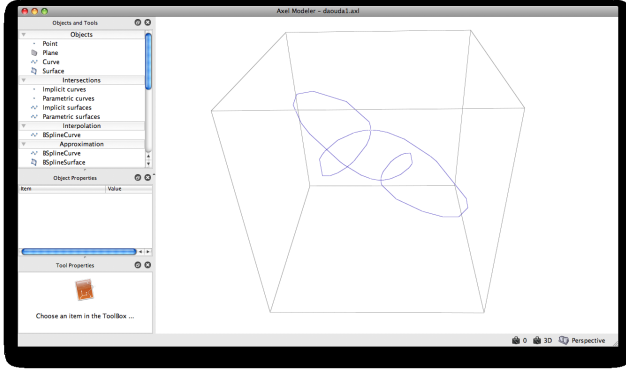


Figure 6: Computed topology of curve 2 of table 5.

ten using the Computer Algebra System Mathemagix¹. Results are visualized using the Axel² algebraic geometric modeler which allows the manipulation of geometric objects with algebraic representation such as implicit or parametric curves or surfaces.

Since existing methods have no publicly available implementations, table 5 only reports our experiments, performed on an Intel(R) Core machine clocked at 2GHz with 1GB RAM.

5. REFERENCES

- [1] Lionel Alberti and Bernard Mourrain. Visualisation of implicit algebraic curves. In Marc Alexa, Steven Gortler, and Tao Ju, editors, *Pacific Conference on Computer Graphics and Applications, IEEE Computer Society*, p. 303–312, 2007.
- [2] J.G. Alcazar, and J.R. Sendra. *Computation of the Topology of Algebraic Space Curves*. *J. Symbolic Comput.*, vol. 39, no. 6, 719–744, 2005.
- [3] S. Basu, R. Pollack and M.F. Roy. *Algorithms in real algebraic geometry*, Algorithms and Computation in Mathematics, vol. 10, second edition, Springer-Verlag, Berlin, 2006.
- [4] R. Benedetti and J.J. Risler. *Real algebraic and semi-algebraic sets*, Actualités Mathématiques. [Current Mathematical Topics], Hermann, Paris, 1990.
- [5] G.E. Collins. *Subresultants and reduced polynomial remainder sequences*. *J. ACM*, 14 :128-142, 1967.
- [6] M. El Kahoui. *Topology of real algebraic space curves*. *J. of Symbolic Computation*, 43: 235-258, 2008.
- [7] Arnborg and H.Feng. *Algebraic decomposition of regular curves*. *J. Symbolic Computation*, 5(1-2): 131-140, 1988.
- [8] Arnon and S. McCallum. *A polynomial-time algorithm for the topological type of a real algebraic curve*. *J. Symbolic Computation*, 5(1-2): 213-236, 1988.
- [9] M. Coste and M-F. Roy. *Thom’s lemma, the coding of real algebraic numbers and the computation of the topology of semi-algebraic sets*. *J. Symbolic Computation*, 5: 121-129, 1988.
- [10] T. Sakkalis. *The topological configuration of real algebraic curves*. *Bultin of the Australian Mathematical Society*, 43: 37-50, 19.
- [11] G. Gattellier, A. Labrouzy, B. Mourrain, and J.P. T  court. *Computing the topology of three-dimensional algebraic curves*. In *Computational methods for algebraic spline surfaces*, p. 27–43, Springer, Berlin, 2005.
- [12] L. Gonzalez-Vega, I. Necula. *Efficient topology determination of implicitly defined algebraic plane curves*. In *Comput. Aided Geom. Design*, vol. 19, no. 9, 719-743, 2002.
- [13] T.A Grandine. *Applications of contouring*. In *SIAM Rev.*, Vol. 42, no. 2, 297-316, 2000.
- [14] T.A Grandine, F.W Klein. *A new approach to the surface intersection problem*. In *Comput. Aided Geom. Design*, Vol. 14, no. 2, 111–134, 1997.
- [15] J.C. Owen, and A.P. Rockwood. *Intersection of general implicit surfaces*. In *Geometric modeling*, SIAM, 335–345, 1987.
- [16] A. Eigenwillig, M. Kerber and N. Wolpert. *Fast and Exact Geometric Analysis of Real Algebraic Plane Curves*. In *Proc. of the 2007 Int. Symp. on Symb. and Alg. Comp. (ISSAC 2007)*.
- [17] C. Mittermaier, W. Schreiner and F. Winkler. *Plotting Algebraic Space Curves by Cluter Computing*. In *Proc. of ASCM 2000*.pp. 49-58 .
- [18] H. Hong. *An Efficient Method for Analyzing The Topology of Plane Real Algebraic Curves*. *Math. and Comp. Sim.* 42 (1996) 541-582.
- [19] C. Bajaj, C.M. Hoffmann. *Tracing Surfaces Intersection*. 1988 *Comput. Aided. Geom. Design* 5, 285-307.
- [20] J. Keyser, T. Culver., D. Manocha, S. Krishnan *Efficient and Exact Manipulation of Algebraic Points and Curves* 2000 *Comput. Aided. Geom. Design* 32(11), 649-662.

¹<http://www.mathemagix.org/>

²<http://axel.inria.fr>

Gröbner-free Normal Forms for Boolean Polynomials

Michael Brickenstein
Mathematisches Forschungsinstitut Oberwolfach
Schwarzwaldstr. 9-11
77709 Oberwolfach-Walke, Germany
brickenstein@mfo.de

Alexander Dreyer
Fraunhofer Institute for Industrial Mathematics
Fraunhofer-Platz 1
67663 Kaiserslautern, Germany
alexander.dreyer@itwm.fraunhofer.de

ABSTRACT

This paper introduces a new method for interpolation of Boolean functions using Boolean polynomials. It was motivated by some problems arising from computational biology, for reverse engineering the structure of mechanisms in gene regulatory networks. For this purpose polynomial expressions have to be generated, which match known state combinations observed during experiments. Earlier approaches using Gröbner techniques have not been powerful enough to treat real-world applications.

The proposed method avoids expensive Gröbner basis computations completely by directly calculating reduced normal forms. The problem statement can be described by Boolean polynomials, i. e. polynomials with coefficients in $\{0, 1\}$ and a degree bound of one. Therefore, the reference implementations mentioned in this work are built on the top of the POLYBORI framework, which has been designed exclusively for the treatment of this special class of polynomials.

A series of randomly generated examples is used to demonstrate the performance of the direct method. It is also compared with other approaches, which incorporate Gröbner basis computations.

Categories and Subject Descriptors

I.1.2 [Computing Methodologies]: Symbolic and Algebraic Manipulation, Algorithms

General Terms

Algorithms, Theory

Keywords

Interpolation, Gröbner, normal forms, Boolean polynomials

1. INTRODUCTION

This work was originally motivated by some problems arising from computational biology, originally stated by Laubenbacher and Stigler in [10]. One important issue in the field of

system biology is to detect and model the causal behaviour of the mechanisms in gene regulatory networks [9].

Laubenbacher and Stigler propose a method for reverse engineering the structure of such networks from experimental data. They have presented an algebraic approach for the generation of Boolean networks, in which variables have only two possible states. For this purpose polynomial expressions were generated, which match known state combinations observed during experiments. In order to obtain compact formulations, they make use of Gröbner basis techniques for generating reduced normal forms.

Such systems can be described by Boolean polynomials, i. e. polynomials with coefficients in $\{0, 1\}$ and a degree bound of one. The POLYBORI framework introduced earlier [2] is capable of efficiently handling these special kind of polynomials. It forms the basis of the reference implementations mentioned in this work. Therefore, we start with a brief introduction about both subjects and some basics from computational algebra.

In the following, a new approach is presented for direct computation of reduced normal forms. The latter can be applied to the reverse engineering problems from computational biology. Note that the proposed method does not make use of expensive Gröbner basis computations. Indeed, the setup of a polynomial generating system is avoided completely.

2. BOOLEAN POLYNOMIALS

We are actually interested in modelling expressions from propositional logic as polynomials over the finite field with two elements. Allowing values from $\{0, 1\}$ only, the condition $x = x^2$ holds for all $x \in \mathbb{Z}_2$. Hence, we deal with elements of the polynomial ring $P = \mathbb{Z}_2[x_1, \dots, x_n]$ restricted by the *field equations*

$$x_1^2 = x_1, x_2^2 = x_2, \dots, x_n^2 = x_n. \quad (1)$$

This leads to a degree bound of one on all variables, and therefore we can restrict ourselves to a certain class of polynomials in the following.

Definition 1. (Boolean polynomial) Let $p \in \mathbb{Z}_2[x_1, \dots, x_n]$ be a polynomial, such that

$$p = a_1 \cdot x_1^{\nu_{11}} \cdot \dots \cdot x_n^{\nu_{1n}} + \dots + a_m \cdot x_1^{\nu_{m1}} \cdot \dots \cdot x_n^{\nu_{mn}} \quad (2)$$

with coefficients $a_i \in \{0, 1\}$ and $\nu_{ij} \in \{0, 1\}$. Then p is called a **Boolean polynomial**.

In our context it is enough to treat Boolean polynomials only, as they are exactly the canonical representatives of

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

residue classes in the quotient ring of $\mathbb{Z}_2[x_1, \dots, x_n]$ modulo the ideal of the field equations $\langle x_1^2 + x_1, \dots, x_n^2 + x_n \rangle$.

Any Boolean polynomial p is defined by the fact, whether each term $x_1^{\nu_{i1}} \dots x_n^{\nu_{in}}$ occurs in it. Analogously, the occurrences of the variables determine each term. One can assign a set $S_p = \{s_1, \dots, s_m\}$ to p consisting of different subsets s_k , $s_i \neq s_j$ for $i \neq j$, of the variable vector $\{x_1, \dots, x_n\}$. Then Equation (2) can be rewritten as

$$p = \sum_{s \in S_p} \left(\prod_{x_\nu \in s} x_\nu \right). \quad (3)$$

In fact, there is a one-to-one correspondence between the set of Boolean polynomials and the set of all subsets of the power set of the variable vector $\{x_1, \dots, x_n\}$ via the mapping defined by $S_p \mapsto \sum_{s \in S_p} \left(\prod_{x_\nu \in s} x_\nu \right) = p$.

For practical applications it is reasonable to assume that S_p is sparse, i. e. the set is only a small subset of the power set over the variable vector. Consequently, the strategies of the used algorithms have to be tuned to preserve sparsity.

2.1 Binary Decision Diagrams

In this section, we give a rough overview about binary decision diagrams (BDDs) and their application in the context of Boolean polynomials. A more exhaustive discussion can be found in [2]. BDDs are widely used in various areas as a unique representation of large sets, which could not be constructed efficiently by an enumerative approach.

Definition 2. (Binary Decision Diagram) A **binary decision diagram** (BDD) is a rooted, directed, and acyclic graph with two terminal nodes $\{0, 1\}$ and decision nodes. The latter have two ascending edges (high/low or then/else), each of which corresponding to the assignment of true or false, respectively, to a given Boolean variable.

A series of connected nodes of a BDD starting at the root and ending at the terminal node 1 is called a **path**. In case that the variable order is constant over all paths, we speak of an **ordered** BDD.

The BDD data structure is compact and easy to handle. For a more detailed treatment of the subject see [3] and [1].

Definition 3. Let b be a binary decision diagram.

- The decision variable associated with the root node of b is denoted by $\text{root}(b)$. If b is ordered, then x_i corresponds to the i -th variable, and one can uniquely identify a decision variable with its index. In the latter case, for $\text{root}(b) = x_i$, we set $\text{top}(b) = i$.
- Furthermore, $\text{then}(b)$ and $\text{else}(b)$ indicate the (sub-)diagrams, linked to then- and else-edge, respectively, of the root node of b .
- In addition, $\text{subset1}(b, x)$ and $\text{subset0}(b, x)$ denote the BDDs, where all nodes associated with x are replaced by their then- and else-branches, respectively. In particular, at root node $\text{subset1}(b, \text{root}(b)) = \text{then}(b)$ and $\text{subset0}(b, \text{root}(b)) = \text{else}(b)$ hold.
- For two BDDs b_1, b_0 , which do not depend on the decision variable x , the **if-then-else operator** $\text{ite}(x, b_1, b_0)$ denotes the BDD c , which is obtained by introducing a new node associated with the variable x , such that we have $\text{root}(c) = x$, $\text{then}(c) = b_1$, and $\text{else}(c) = b_0$.

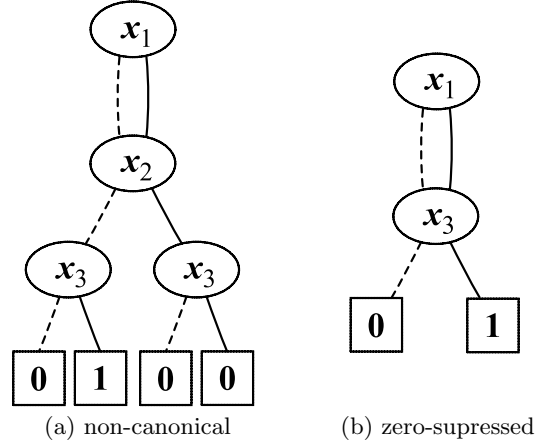


Figure 1: Different binary decision diagrams representing $x_1 x_3 + x_3$. Edges: then (—) and else (---).

The set of all valid paths of a binary decision diagram can be used to represent the terms of Boolean polynomials. Any Boolean polynomial p can be written as $x_i \cdot p_1 + p_0$, where x_i is the variable in p with the smallest index i , and p_1 and p_0 are Boolean polynomials not containing x_i . Hence p may be identified with a BDD $b = \text{ite}(x_i, b_1, b_0)$, where b_1 and b_0 are recursively defined as those diagrams corresponding to p_1 and p_0 , respectively.

For instance, Figure 1 shows some BDD variants for the polynomial $p = x_1 x_3 + x_3$. The first diagram can easily be derived from $p = x_1 \cdot p_1 + p_0$, with $p_1 = x_2 \cdot (x_3 \cdot 0 + 0) + x_3 \cdot 1 + 0$ and $p_0 = x_2 \cdot (x_3 \cdot 0 + 0) + x_3 \cdot 1 + 0$.

Unlike other BDD approaches, the diagram paths do not represent the valid solutions of the Boolean function behind p , but they form the sets $\{x_1, x_3\}$ and $\{x_3\}$ corresponding to polynomial terms directly.

For efficiency reasons it is useful to merge subdiagrams, i. e. if some edges point to equivalent subdiagrams, those are forced to point to the same diagram and share it. The following definition of the ZDD (sometimes also called ZBDD or ZOBDD) also utilises sparsity, which means that zero-terms are not stored.

Definition 4. (ZDD) Let z be an ordered binary decision diagram with equal subdiagrams merged. Then z is called a **zero-suppressed binary decision diagram** (ZDD), if and only if those nodes are eliminated, whose then-edges point to the 0-terminal.

Note, in this case elimination means that a node n is removed from the diagram and all edges pointing to it are linked to $\text{else}(n)$. Figure 1 illustrates this reduction step of a given BDD for the polynomial $x_1 x_3 + x_3$. In Figure 1(a) the then-edge of the right node with decision variable x_3 is pointing to the 0-terminal. Hence, it can be safely removed, without losing information. As a consequence, the then-edge of the x_2 -node is now connected to zero, and hence can also be eliminated. The effect of the complete zero-suppressed node reduction can be seen in Figure 1(b). Note, that the construction guarantees canonicity of resulting diagrams, see [6].

3. ALGEBRAIC BASICS

In this section, we recall some algebraic basics, including classical notions for the treatment of polynomial systems, as well as basic definitions and results from computational algebra. For a more detailed treatment of the subject see [7] and the references therein.

Let $P = \mathbb{Z}_2[x_1, \dots, x_n]$ be the polynomial ring over the field $\mathbb{Z}_2$. A **monomial ordering** on P , more precisely, on the set of monomials $\{x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n} \mid \alpha \in \mathbb{N}^n\}$, is a well ordering “ $>$ ” (i.e. each nonempty set has a smallest element with respect to “ $>$ ”) with the following additional property: $x^\alpha > x^\beta \Rightarrow x^{\alpha+\gamma} > x^{\beta+\gamma}$, for $\gamma \in \mathbb{N}^n$.

Let $f = \sum_\alpha c_\alpha \cdot x^\alpha$ ($c_\alpha \in \mathbb{Z}_2$) a polynomial. Then

$$\text{supp}(f) = \{x^\alpha \mid c_\alpha \neq 0\}$$

is called the **support** of f .

Furthermore, if $f \neq 0$ then $\text{lm}(f)$ denotes the **leading monomial** of f , the biggest monomial occurring in f with respect to “ $>$ ”. Moreover, we set $\text{tail}(f) = f - \text{lm}(f)$.

If $F \subset P$ is any subset, $L(F)$ denotes the **leading ideal** of F , i.e. the ideal in P generated by $\{\text{lm}(f) \mid f \in F \setminus \{0\}\}$. A monomial m is called a **standard monomial** for an ideal I , if $m \notin L(I)$.

The S -polynomial of $f, g \in P \setminus \{0\}$ with leading monomials $\text{lm}(f) = x^\alpha$ and $\text{lm}(g) = x^\beta$ is denoted by

$$\text{spoly}(f, g) = x^{\gamma-\alpha} f - x^{\gamma-\beta} g,$$

where $\gamma = (\max(\alpha_1, \beta_1), \dots, \max(\alpha_n, \beta_n))$. Also, we recall that $G \subset P$ is called a **Gröbner basis** of an ideal $I \subset P$, if $\{\text{lm}(g) \mid g \in G \setminus \{0\}\}$ generates $L(I)$ in the ring P and the inclusion $G \subset I$ holds.

Definition 5. Let $f, g_1, \dots, g_m \in P$, and let $h_1, \dots, h_m \in P$. Then

$$f = \sum_{i=1}^m h_i \cdot g_i \in K[x_1, \dots, x_n],$$

is called a **standard representation** of f with respect to $g_1, \dots, g_m$, if $h_i \cdot g_i = 0$ for all i , or $\text{lm}(h_i \cdot g_i) \leq \text{lm}(f)$ otherwise.

A polynomial f is said to be reduced against a generating system G , if and only if $\text{supp}(f) \cap L(G) = \emptyset$. A polynomial r is called the **reduced normal form** of f against G , if and only if it is reduced against G and $f - r$ has standard representation with respect to G . As the reduced normal form is unique, we may denote this by $r = \text{REDNF}(f, G)$. Moreover, the reduced normal does not depend on G itself, but only on the ideal spanned by G . In the following, we assume for every ideal I , that it contains the field equations. This has two direct consequences:

- Every ideal is radical,
- all zeros are contained in $\mathbb{Z}_2^n$.

Hence, there is a one-to-one correspondence between ideals and their varieties. Using this correspondence $\text{REDNF}(f, G)$ only depends on the common zeros of G , respectively $\langle G \rangle$.

Definition 6. Given a set of points P , we define its vanishing ideal $I(P)$ to be the set of all polynomials evaluating to zero on P :

$$I(P) = \{f \in \mathbb{Z}_2[x_1, \dots, x_n] \mid f(p) = 0 \forall p \in P\}$$

4. THE POLYBORI FRAMEWORK

With POLYBORI, we have implemented a framework for *Polynomials over Boolean Rings* based on Python and C++, which provides high-level data types for Boolean polynomials and monomials, exponent vectors, as well as for the underlying polynomial rings. The ring variables may be identified by their indices or by a custom string. Polynomial structures and monomials use ZDDs as internal storage type, but this is hidden from the user. The current implementation uses the decision-diagram management from CUDD [11]. Its functionality is included using interface classes, which allows an easy replacement of the underlying BDD system without extensive rewriting of crucial POLYBORI procedures.

In addition, basic polynomial operations – like addition and multiplication – have been implemented and associated to the corresponding operators. In order to enable efficient implementations, these operations were reformulated in terms of set operations, which are compatible with the ZDD approach. This also applies to other classical functionality like degree computation and leading-term computation.

4.1 Polynomial Operations

Boolean polynomial rings are motivated by the fact, that logical operations on bits can be reformulated in terms of addition and multiplication of $\mathbb{Z}_2$ -valued variables. Representing polynomials as ZDDs these operations may also be implemented as set operations. For instance, adding the polynomials $p = \sum_{s \in S_p} \left(\prod_{x_\nu \in s} x_\nu \right)$, $q = \sum_{s \in S_q} \left(\prod_{x_\nu \in s} x_\nu \right)$, with S_p and S_q as illustrated in Equation (3) is just

$$p + q = \sum_{s \in S_{p+q}} \left(\prod_{x_\nu \in s} x_\nu \right), \text{ where } S_{p+q} = S_p \oplus S_q. \quad (4)$$

Note that the *symmetric difference* on the right-hand side is defined as $A \oplus B = (A \cup B) \setminus (A \cap B)$ for sets A and B . Although each of these three operations is already available for ZDDs, it is more preferable to have them replaced by one specialised procedure. This avoids large intermediate sets (like $S_p \cup S_q$) and repeated iterations over both arguments. A suitable implementation of the Boolean polynomial addition is shown in Algorithm 1. Since the indices $\text{top}(p)$ and $\text{top}(q)$ are greater than i , the if-then-else operator $\text{ite}(x_i, p, q)$, which is equivalent to $x_i \cdot p + q$ here, can be implemented cheaply by linking then- and else-branches of the new root node for x_i to p and q , respectively.

The procedure also includes a cache lookup. It can be implemented cheaply, because polynomials have unique representations as ZDDs. Hence, previous computations of $f + g$ can be reused. The advantage of a recursive formulation is, that this also applies to those subpolynomials, which are generated by $\text{then}(f)$ and $\text{else}(f)$. Currently, the default settings of the underlying BDD library are used for cache-size and related parameters. Fine-tuning and optimisation in this area are subject of further research. In a similar manner a kind of Boolean multiplication can be computed.

The operations mentioned above are independent of the actual monomial ordering. It is a major feature of POLYBORI to make functions like the calculation of the **leading term** available for various nontrivial “ $<$ ”-relations. While the terms of a ZDD-stored polynomial can naturally be accessed in a lexicographical way, this does not apply to other orderings.

Algorithm 1 Recursive Boolean addition

Require: f, g Boolean polynomials**Ensure:** $h = f + g$

```

if  $f = 0$  then
   $h = g$ 
else if  $g = 0$  then
   $h = f$ 
else if  $f = g$  then
   $h = 0$ 
else
  if  $\text{isCached}(+, f, g)$  then
     $h = \text{cache}(+, f, g)$ 
  else
     $\text{set } \nu = \text{top}(f), \mu = \text{top}(g)$ 
    if  $\nu < \mu$  then
       $h = \text{ite}(x_\nu, \text{then}(f), \text{else}(f) + g)$ 
    else if  $\nu > \mu$  then
       $h = \text{ite}(x_\mu, \text{then}(g), f + \text{else}(g))$ 
    else
       $h = \text{ite}(x_\nu, \text{then}(f) + \text{then}(g), \text{else}(f) + \text{else}(g))$ 
       $\text{cache}(+, f, g) = h$ 
  return  $h$ 

```

In the following, we will always assume $x_1 > x_2 > \dots$ and that the variables with smaller index occur on top of the decision diagram. By taking advantage of cache and uniqueness properties POLYBORI implements degree and block orderings with a reasonable small computational overhead.

5. BOOLEAN INTERPOLATION

A partial Boolean function $f : \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2$ can be defined by two disjoint subsets Z, O of $\mathbb{Z}_2^n$, where $f(o) = 1$ and $f(z) = 0$ for each $z \in Z, o \in O$. Given O and Z we can also denote f by b_Z^O . The word *partial* implies, that $Z \cup O$ may be a proper subset of $\mathbb{Z}_2^n$.

Given two partial Boolean functions $b_{Z_1}^{O_1}, b_{Z_2}^{O_2} : \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2$. We define the sum $(b_{Z_1}^{O_1} + b_{Z_2}^{O_2}) : \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2$ by the mapping $x \mapsto b_{Z_1}^{O_1}(x) + b_{Z_2}^{O_2}(x)$ for all $x \in (Z_1 \cup O_1) \cap (Z_2 \cup O_2)$.

For this partial function definition we search for a Boolean polynomial p , where the associated function f_p is a specialisation of f , i.e. $f_p(x) = f(x)$ for each $x \in O \cup Z$.

Algorithms for calculating this interpolation can be compared (see Section 5.4) in performance and compactness of the result. Moreover, it would be worthwhile, if the output can be defined by a simple mathematical property.

In the following, we present a method with a very compact result, which is lexicographically reduced with respect to the vanishing ideal of $O \cup Z$. In this way, it can also be reinterpreted as fast normal form computation against this ideal.

A partial Boolean function can be represented as a pair of two ZDDs Z and O in the following way: A point v in $\mathbb{Z}_2^n$ can be identified in the same way like the monomial x^v with a subset of $1, \dots, n$ as $v \mapsto \{i | v_i = 1\}$. In the same way, we can identify a set of points with a subset of the power set of $\{1, \dots, n\}$ or a ZDD.

In this spirit, the addition of two partial Boolean functions could be expressed as follows in terms of ZDD operations:

$$D = (Z_1 \cup O_1) \cap (Z_2 \cup O_2)$$

$$(b_{Z_1}^{O_1} + b_{Z_2}^{O_2}) = b_{(Z_1 \cap Z_2) \cup (O_1 \cap O_2)}^{(O_1 \oplus O_2) \cap D}.$$

5.1 Encoding

A vector $v = (v_1, \dots, v_n) \in \mathbb{Z}_2^n$ can be encoded just by the set containing all variables x_i , where $v_i \neq 0$. Analogously to polynomials a subset of $\mathbb{Z}_2^n$ can be translated into a subset of the power set of $\{x_1, \dots, x_n\}$. Using this encoding, we obtain a compact data structure for these kinds of sets, which share some common parts of the vectors. The latter may be treated simultaneously by the a recursive method. Also, efficient caching of intermediate results for reuse is possible.

5.2 Zeros of a Boolean polynomial

In this section, we introduce an algorithm for computing the zeros of a given Boolean polynomial in a (possibly proper) subset of $\mathbb{Z}_2^n$. For illustration, a single vector $v = (v_1, \dots, v_n)$ in $\mathbb{Z}_2^n$ is considered. One may define a sequence by setting $p_0 = p$, and subsequently substituting all variables by its corresponding Boolean value in v as $p_i = p_{i-1}|_{x_i=v_i}$. This terminates ultimately in 0 or 1. Hence, it proves whether v is a zero of p or not. For a set of vectors, one could do this evaluation element-wise, but this naïve approach is not suitable for large sets. In contrast, using the encoding proposed above, some common parts of the vectors may be treated simultaneously. Again, the resulting subset may be encoded as a decision diagram “on the fly” as Algorithm 2 shows.

Algorithm 2 Recursive zeros: $\text{zeros}(p, S)$

Require: Boolean polynomial p , S set of points in $\mathbb{Z}_2^n$ represented as ZDDs**Ensure:** $\text{zeros}(p, S) = \{s \mid s \in S \text{ and } p(s) = 0\}$

```

if  $p = 0$  then
  return  $S$ 
if  $(p = 1)$  or  $(S = \emptyset)$  then
  return  $\emptyset$ 
if  $S = \text{ZDD}(\{\emptyset\})$  then
  if  $p$  has constant part then
    return  $\emptyset$ 
  else
    return  $S$ 
/* As from now  $p, S$  are certainly non-constant ZDDs. */
while  $\text{top}(p) < \text{top}(S)$  do
   $p = \text{else}(p)$ 
   $i = \min(\text{top}(p), \text{top}(S))$ 
   $p_0 = \text{subset0}(p, x_i)$ 
   $p_1 = \text{subset1}(p, x_i)$ 
   $S_0 = \text{subset0}(S, x_i)$ 
   $S_1 = \text{subset1}(S, x_i)$ 
   $Z_{00} = \text{zeros}(p_0, S_0)$ 
   $Z_{01} = \text{zeros}(p_0, S_1)$ 
   $Z_{11} = \text{zeros}(p_1, S_1)$ 
  return  $\text{ite}(x_i, S_1 \setminus (Z_{01} \oplus Z_{11}), Z_{00})$ 

```

Consequently, Algorithm 3 yields the remaining points. These are exactly those points, for which a given Boolean polynomial evaluates to one.

Algorithm 3 Ones: $\text{ones}(p, S)$

Require: Boolean polynomial p , S set of points in $\mathbb{Z}_2^n$ **Ensure:** $\text{ones}(p, S) = \{s \mid s \in S \text{ and } p(s) = 1\}$ **return** $S \setminus \text{zeros}(p, S)$

5.3 Normal forms against a variety

Normal forms are very hard to compute, in particular, if Gröbner bases calculations are in use. So, it is a quite natural question, whether the latter may be avoided by introducing specialised methods for calculating normal forms. The presented method works even without a generating system of the ideal, i.e. in case it is just given in form of its variety.

Since ideals in the ring of Boolean polynomials are in one-to-one correspondence to subset of $\mathbb{Z}_2^n$, this problem is quite general. Moreover, the Gröbner basis of the ideal can be a much bigger object than the corresponding variety.

Definition 7. Let $>$ be an arbitrary monomial ordering, then we can extend $>$ lexicographically to the set of Boolean polynomials (we make use of the fact, that all non-zero coefficients are one) by setting $p \geq q$ if and only if one of the following conditions holds:

- $q = 0$,
- $p \neq 0, q \neq 0$ and $\text{lm}(p) > \text{lm}(q)$,
- $p \neq 0, q \neq 0$, $\text{lm}(p) = \text{lm}(q)$ and $\text{tail}(p) \geq \text{tail}(q)$.

LEMMA 5.1. *Let $I \supset \langle x_1^2 + x_1, \dots, x_n^2 + x_n \rangle$ be an ideal, with $I \subset \mathbb{Z}_2[x_1, \dots, x_n]$, p a Boolean polynomial, and let G be a Gröbner basis of I . Then the following two statements are equivalent:*

1. p is the lexicographically smallest Boolean polynomial in $p + I$.
2. $\text{REDNF}(p, G) = p$ (i.e. p is reduced).

PROOF. A polynomial is reduced, if and only if all its terms are reduced (standard monomials).

Let us assume that a term t of p is not reduced: Then there exists an $f \in G$, such that t can be rewritten by f modulo I . Applying the field equations the result is again a Boolean polynomial. In this way, we have constructed a smaller Boolean polynomial in $p + I$.

On the other hand, let p be reduced. Assume, there exists a smaller polynomial q in the same residue class. We may assume that $\text{lm}(p) \neq \text{lm}(q)$. Then we know from general Gröbner basis theory, that $\text{REDNF}(p, G) = \text{REDNF}(q, G)$. Since p is reduced, it follows $p = \text{REDNF}(q, G)$ and $p - q$ has a standard representation (each summand has a leading term smaller or equal to $\text{lm}(p) = \text{lm}(p - q)$).

Hence, it follows, that one of the summands in this representation has $\text{lm}(p)$ as leading term and p is not reduced. $\square$

In order to show the basic principles of interpolating with Boolean polynomials, we provide a simple interpolation procedure given in Algorithm 4. If two sets Z and O of vectors are given, we can reduce the interpolation problem to the case, where the i -th index of all vectors is 0. For this purpose let Z_0, O_0 denote the sets of those vectors, whose i -th entry is zero (subset-0 points). Analogously, for $S \in \{Z, O\}$ let $S_1 = \{(v_1, \dots, v_{i-1}, 0, v_{i+1}, \dots, v_n) \mid v \in S \text{ and } v_i = 1\}$ denote the subset-1 points.

First, we calculate an interpolation h_e for $Z_0 \cup O_0$. In the second step, we may continue with an interpolation h of the subset-1 points in $Z_1 \cup O_1$. One could combine both recursively generated results to $x_i \cdot h + h_e$, but this yields wrong results for vectors in $O_1 \cap O_0$. This can easily be

Algorithm 4 $\text{interpolate\_simple}(b_Z^O)$

Require: b_Z^O a partial function definition.
Ensure: $\text{interpolate\_simple}(b_Z^O) = p$ with $f_p = b_Z^O$ on $Z \cup O$

```

if  $Z = \emptyset$  then
  return 1
if  $O = \emptyset$  then
  return 0
 $i = \min(\text{top}(O), \text{top}(Z))$ 
 $Z_1 = \text{subset1}(Z, x_i)$ 
 $Z_0 = \text{subset0}(Z, x_i)$ 
 $O_1 = \text{subset1}(O, x_i)$ 
 $O_0 = \text{subset0}(O, x_i)$ 
 $h_e = \text{interpolate\_simple}(b_{Z_0 \cup O_0}^{O_0})$ 
 $h_t = \text{interpolate\_simple}(b_{Z_1 \cup O_1}^{O_1}) + h_e$ 
return  $x_i \cdot h_t + h_e$ 

```

fixed by cancelling those terms already treated by h_e . In this case, it can be done just by adding h_e to h , because the terms with x_i do not influence the behaviour on $Z_0 \cup O_0$. For this reason h_e is computed first.

On the other hand, if we really want to have a minimal interpolation polynomial, then we should not adjust the bigger terms (those with x_i : $x_i \cdot h_t$) to the smaller terms (those without x_i), but do the opposite. So we would like to compute h_t first. The problem is, if we want to calculate the then-branch of the interpolation polynomial h_t , then we have to know the value h_e takes on the interpolation points specified for h_t . Moreover, we should not specify more interpolation points than needed for h_t , as for every point in $Z_1 \cup O_1$, which does not lie in $Z_0 \cup O_0$ (subset-0 points) the value of the function can be adjusted by h_e . Luckily the behaviour of h_e on $(Z_1 \cup O_1) \cap (Z_0 \cup O_0)$ is predictable and this gives exactly the set of interpolation points, which we will specify for h_t . This approach is incorporated in Algorithm 5.

Algorithm 5 $\text{interpolate\_smallest\_lex}(b_Z^O)$

Require: b_Z^O a partial function definition.
Ensure: $\text{interpolate\_smallest\_lex}(b_Z^O)$, smallest Boolean polynomial p with respect to lex. with $f_p = b_Z^O$ on $Z \cup O$

```

if  $Z = \emptyset$  then
  return 1
if  $O = \emptyset$  then
  return 0
 $i = \min(\text{top}(O), \text{top}(Z))$ 
 $Z_1 = \text{subset1}(Z, x_i)$ 
 $Z_0 = \text{subset0}(Z, x_i)$ 
 $O_1 = \text{subset1}(O, x_i)$ 
 $O_0 = \text{subset0}(O, x_i)$ 
 $C = (Z_1 \cup O_1) \cap (Z_0 \cup O_0)$  /*  $C$  forms the set of conflict points between 0 and 1-subsets */
 $f = b_{Z_1}^{O_1}$ 
 $g = b_{Z_0}^{O_0}$ 
 $h_t = \text{interpolate\_smallest\_lex}(f + g)$  /*  $f + g$  is only defined on  $C$  */
 $F = \text{ones}(h_t, ((Z_1 \cup O_1) \setminus C))$ 
/* non-conflict subset1 terms affected by else branch */
 $w = b_{((Z_1 \setminus C) \oplus F) \cup O_0}^{((Z_1 \setminus C) \oplus F) \cup Z_0}$ 
 $h_e = \text{interpolate\_smallest\_lex}(w)$ 
return  $x_i \cdot h_t + h_e$ 

```

EXAMPLE 5.2. We want to compute a minimal interpolation polynomial for b_Z^O using Algorithm 5, where

$$Z = \{(0, 0, 0), (1, 1, 0), (1, 0, 1)\}, \\ O = \{(1, 0, 0), (0, 1, 0), (0, 1, 1)\}.$$

In order to reduce the problem, we compute the cofactors with respect to the first variable and skip the first component, which is always zero. This yields

$$Z_1 = \{(1, 0), (0, 1)\}, O_1 = \{(0, 0)\}, \\ Z_0 = \{(0, 0)\}, O_0 = \{(1, 1), (1, 0)\}.$$

Hence $f + g = b_{\emptyset}^{\{(1,0),(0,0)\}}$, which implies $h_t = 1$. Here, we get a first impression of the reason why the algorithm actually returns the minimal interpolation polynomial: Since h_t is finally multiplied by x_0 , it is the lexicographically biggest part of our computation. We specify a minimal set of interpolation points. In this case, it is even possible to fulfil this partial function definition by the constant polynomial 1. For

$$w = b_{\{(0,0)\}}^{\{(1,1),(1,0),(0,1)\}}$$

we recursively obtain $h_e = \text{interpolate\_smallest\_lex}(w)$, which equals $x_1 \cdot x_2 + x_1 + x_2$. This gives us the result:

$$\begin{aligned} \text{interpolate\_smallest\_lex}(b_Z^O) &= x_0 \cdot h_t + h_e \\ &= x_0 + x_1 \cdot x_2 + x_1 + x_2 \end{aligned}$$

The following results show that the algorithm computes the desired interpolation.

LEMMA 5.3. Let S be a set of non-constant Boolean polynomials, v the smallest top variable occurring in S (biggest index: $\max\{\text{top } f \mid f \in S\}$), $S \ni p = v \cdot p_1 + p_0$ (v not occurring in p_1, p_0). Then p is the lexicographically smallest polynomial with respect to lexicographical monomial ordering if and only if p_1 is the minimal cofactor in S and for all $g \neq p$ with $g = v \cdot p_1 + g_0$ holds: $g_0 > p_0$.

PROOF. This lemma is a consequence of the fact, that if $p = v \cdot p_1 + p_0$ with v as top variable, all terms in which v occurs are bigger than those without v . So these are considered first in lexicographical comparison of Boolean polynomials. $\square$

THEOREM 5.4. Algorithm 5 returns an interpolation polynomial, which is the lexicographically smallest polynomial with respect to lexicographical monomial ordering under all polynomials interpolating the same function on P .

PROOF. We can assume that all recursive calls deliver a polynomial with the desired property.

It can be checked easily, that the result provides an interpolation of the specified function by doing case considerations.

To see that it is minimal, we use the previous lemma for minimality: First, we have to check, that h_t is chosen minimal. By our assumptions the recursion returns a minimal polynomial under the specified properties. Thus, we know that it is minimal under all possible choices for the then-branch h_t of our interpolation polynomial, fulfil the specified condition (which means that the recursive call chooses the minimal polynomial out of the full set of candidates). We just have to check, that all conditions are necessary for the then-branch of an interpolation polynomial.

In fact, we only prescribe the behaviour of h_t on all conflicting points, such originally prescribed points, which occur in both combinations: those, with index i set to 0 on the one hand (denoted by C_0), and index i set to 1 on the other (C_1). As the set C in the algorithm is already projected to 0 in the i -th component, we can denote this by

$$C_1 = \{c_{x_i=1} \mid c \in C\} \text{ and } C_0 = C.$$

For these points we have the following situation:

$$\forall c \in C_0 : h_e(c) = h_e(c_{x_i=1}), c_{x_i=1} \in C_1, (x_i \cdot h_t)(c) = 0.$$

No matter what h_t we choose, we get for every suitable h_e

$$h_e(c) = 1 \Leftrightarrow c \in O_0 \text{ for all } c \in C_0.$$

We conclude:

$$\begin{aligned} \forall c \in C_1 : h_t(c) = 1 \Leftrightarrow (c_{x_i=0} \in O_1 \text{ and } c_{x_i=0} \in Z_0) \text{ or} \\ (c_{x_i=0} \in O_0 \text{ and } c_{x_i=0} \in Z_1) \end{aligned}$$

These conditions on h_t are equivalent to the partial Boolean function definition $f + g$. Thus the specification $f + g$ for our then-branch contains only necessary interpolation points and recursively we get a minimal interpolation polynomial for this part. By this we fulfilled the first condition of our lemma and calculated a minimal h_t under all interpolation polynomials. Note, that if we branch at an index i , which is not the biggest possible for an interpolation, h_t will be just zero. Therefore, this is also compatible with the lemma.

Having the correct h_t , we have to check, that under all Boolean interpolation polynomials of the form $x_i \cdot h_t + h_e$, we choose a minimal h_e . The behaviour as function of h_e is uniquely determined on $Z \cup O$ by h_t and the prescribed interpolation values, and this is just, what our recursive call for calculating h_e does: passing these (possibly adjusted) values. Since all candidates for h_e need to fulfil this condition, we really get the full sets of candidates and by recursion a minimal else-branch. $\square$

The following lemma combines the well-known Propositions 4 and 8 from [5, Chapter 5, §3].

LEMMA 5.5. For each monomial ordering $>$ and each set of points P the set of standard monomials S with respect to $I(P)$ in $\mathbb{Z}_2[x_1, \dots, x_n]$ has the same cardinality as P :

$$|S| = |P|.$$

Hence, we have an obvious upper bound for the number of terms of reduced normal forms.

COROLLARY 5.6. Let $f \in \mathbb{Z}_2[x_1, \dots, x_n]$ be a Boolean polynomial, P a set of points in $\mathbb{Z}_2^n$. Then a reduced normal form $g = \text{REDNF}(f, I(P))$ has at most $|P|$ terms.

Algorithm 2 and Algorithm 5 can be combined to compute a reduced normal form without using Gröbner bases.

Algorithm 6 Reduced lexicographical normal form against a variety: `nf_by_interpolate( $f, P$ )`

Require: Boolean polynomial f , P set of points in $\mathbb{Z}_2^n$.

Ensure: `nf_by_interpolate( $f, P$ )` = $\text{NF}(f, I(P))$

$Z = \text{zeros}(f, P)$

return `interpolate_smallest_lex( $b_Z^{P \setminus Z}$ )`

THEOREM 5.7. *Algorithm 6 calculates a lexicographical normal form of f with respect to a Gröbner basis of the ideal*

$$I(P) = \{g \in \mathbb{Z}_2[x_1, \dots, x_n] \mid g(p) = 0 \text{ for all } p \in P\}.$$

PROOF. First note: A reduced normal form against a Gröbner basis G does not depend on the particular choice of G , but only on the ideal generated by G . Moreover, since Boolean polynomials are in one-to-one correspondence to Boolean functions, each Boolean polynomial representing the same function on P , lies in the same residue class modulo $I(P)$. A normal form $g = \text{NF}(f, G)$ of a polynomial against a Gröbner basis G can be characterised, to be the unique reduced polynomial, since $g = \text{NF}(g, G)$, where $g + \langle G \rangle = f + \langle G \rangle$. So we have to show for given f , and P :

1. $\text{nf\_by\_interpolate}(f, P)$ represents the same function restricted to P as f .
2. $\text{nf\_by\_interpolate}(f, P)$ is reduced.

The first claim holds by the correctness, as we have already proven, that nf_by_interpolate returns indeed an interpolating function. The reducedness follows from Lemma 5.1. $\square$

Next, Algorithm 7 computes the standard monomials of an ideal of a variety, which is given as a set of points.

Algorithm 7 Standard monomials of $I(P)$:
`standard_monomials_variety( $P$ )`

Require: P a set of points in $\mathbb{Z}_2^n$.

Ensure: $S = \{t \mid \exists \text{ reduced } p \in I(P) : t \text{ term of } p\}$

$S = \emptyset$

while $|P| \neq |S|$ **do**

$Z = \text{random\_subset}(P)$

$p = \text{interpolate\_smallest\_lex}(b_Z^{P \setminus Z})$

$S = S \cup \text{supp}(p)$

$S = \{t \text{ term} \mid \exists s \in S : t \text{ divides } s\}$

return S

The minimal elements of the remaining Boolean monomials are exactly the leading monomials of the minimal Gröbner basis. Hence, Algorithm 8 can be used to determine the size of the basis without explicitly computing it.

Algorithm 8 Leading monomials of a minimal Gröbner basis of $I(P)$

Require: P set of points in $\mathbb{Z}_2^n$.

Ensure: `leading_monomials_variety( $P$ ) =  $L(I(P))$`

$T = \{t \text{ Boolean term in } \mathbb{Z}_2[x_1, \dots, x_n]\}$

$R = T \setminus \text{standard\_monomials\_variety}(P)$

return `minimal_elements( $R$ )`

THEOREM 5.8. *Assume that the probability distribution of the choice of random subsets Z are independent equipartitions for each point $p \in P$ and form independent experiments. Then the probability p for the main loop of Algorithm 7 passing more than k iterations is less or equal to*

$$1 - (1 - \frac{1}{2^k})^{|P|}.$$

PROOF. For the analysis for the algorithm, we can ignore the step including term, which divide already found terms, as this only reduces the number of needed iterations. For a random subset $Z \subset P$, the probability that a standard monomial t occurs in the interpolation is $\frac{1}{2}$. The probability that a term t does not occur in k experiments is $\frac{1}{2^k}$. Since the probability distribution of independent equipartitions on P result in independence on the set of standard monomials, the probability that all $|P|$ standard monomial occur in k experiments is $(1 - \frac{1}{2^k})^{|P|}$. $\square$

All presented set operations – except the `random_subset` procedure – have been implemented in a decision diagram-style recursive approach. In particular, there is no problem to represent the set of terms T with 2^n elements in it, as the number of ZDD-nodes is just n (in this case).

Algorithm 9 Reduced lexicographical Gröbner basis of $I(P)$

Require: P set of points in $\mathbb{Z}_2^n$.

return $\{t + \text{nf\_by\_interpolate}(t, P) \mid t \in L(I(P))\}$

Algorithm 9 shows, that it is even possible to compute the complete Gröbner basis of $I(P)$ just having this normal form algorithm. However – of course, the actual value of this section is the computation of normal forms without generating the Gröbner basis.

5.4 Practical experiments

In this section, we consider randomly generated examples and show that we can still compute lexicographical normal forms, where the computation of the full Gröbner basis seems to be practically unfeasible. These generating systems are usually quite big. In fact for Table 1 with our experiments, the size of the Gröbner basis was computed by the function `leading_monomials_variety` of Algorithm 8, which returns the leading ideal. This is much cheaper than computing the Gröbner basis itself (at least using Algorithm 9 as you can see in the following). If you consider the size of the Gröbner basis s , the number of points $|P|$, the cost of a single normal form N . Then the cost of computing the leading ideal is approximately $\log(|P|) \cdot N$ and for the Gröbner basis $s \cdot N$ (if you like, you can assume, that s has the same dimension as $|P|$). This is of course no exact complexity analysis, but gives impression of the computation problem.

For computing the normal form against the variety, neither the Gröbner basis is needed, nor the leading ideal, but just a single call of `nf_by_interpolate`. This is much easier, in particular because the Gröbner basis in the biggest example in this listing has about 600000 elements. Due to the random nature of our examples, we expect the tail of each basis element quite dense in the set of standard monomials, which has size $|P|$, 500000 in this example.

Our timings have been done on an AMD Dual Opteron 2.2GHz (we have used only one CPU) with 16 GB RAM running Linux. We used random sets of points and random partial functions in 100 variables. Note that random data is supposed to be the worst case scenario for POLYBORI, as using caching techniques our algorithms work better on structured polynomials.

We were not able to run `interpolate_simple` on the bigger example, since the memory consumption was too high. On the other hand, this algorithm shows nice performance to simplicity in the small examples (mainly, as the expensive

# points	smallest lex.		interpolate.simple		# basis
	time/1 s	length	time/1 s	length	
100	0.01	42	0.00	12771	287
500	0.06	249	0.01	$\approx 2.9 \cdot 10^{10}$	1943
1000	0.29	508	0.01	$\approx 8.1 \cdot 10^{13}$	3393
5000	5.53	2552	1.47	$\approx 4.5 \cdot 10^{23}$	10319
10000	19.78	5020	37.18	$\approx 1.6 \cdot 10^{26}$	17868
50000	250.95	25012			82929
100000	897.85	50093			162024
200000	3488.61	99868			296697
500000	20336.02	249675			636542

Table 1: Interpolation vs. Gröbner approaches.

call to zeros is missing). In this sense, these results confirm, that our algorithm is able to compute normal forms without Gröbner basis up to a magnitude, in which it seems impossible to compute the Gröbner basis on nowadays machines in reasonable time.

6. CONCLUSIONS

Driven by an application from computational biology, we have introduced a new method for interpolation of Boolean functions using Boolean polynomials. Even without Gröbner basis methods, the approach introduced in this paper can be used to calculate reduced normal forms of polynomials. These are applicable for finding compact behavioural models during the reverse engineering of discrete systems, which arise in system biology. The latter were described by combinations of Boolean states resulting from experimental data.

Finally, a series of randomly generated examples was used to illustrate, that the proposed method is superior to those approaches, which incorporate Gröbner basis computations. This is due to the fact, that its Gröbner basis is too large to be treated by nowadays computers in reasonable time. In contrast, our method short-cuts directly to the calculation of reduced normal forms.

7. ACKNOWLEDGMENTS

This work has been partly financed by the *Deutsche Forschungsgemeinschaft* (DFG) under Grant No. GR 640/13-1, and it has been supported by the Rheinland-Pfalz cluster of excellence *Dependable Adaptive Systems and Mathematical Modelling* (DASMOD).

In addition, the authors thank Prof. Gert-Martin Greuel and Prof. Gerhard Pfister (both Department of Mathematics, University of Kaiserslautern, Germany) for their encouragement.

8. REFERENCES

- [1] B. Bérard, M. Bidoit, F. Laroussine, A. Petit, L. Petrucci, P. Schoenebelen, and P. McKenzie. *Systems and software verification: model-checking techniques and tools*. Springer-Verlag New York, Inc., New York, NY, USA, 1999.
- [2] M. Brickenstein and A. Dreyer. PolyBoRi: A framework for Gröbner basis computations with boolean polynomials. In *Electronic Proceedings of Effective Methods in Algebraic Geometry MEGA 2007*, June 2007.
- [3] R. E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, 35(8):677–691, 1986.
- [4] B. Buchberger. Ein Algorithmus zum Auffinden der Basiselemente des Restklassenrings nach einem nulldimensionalen Polynomideal. Dissertation, Universität Innsbruck, 1965.
- [5] D. Cox, J. Little, and D. O’Shea. *Ideals, Varieties, and Algorithms*. Springer, 3rd edition, 2007.
- [6] M. Ghasemzadeh. *A new algorithm for the quantified satisfiability problem, based on zero-suppressed binary decision diagrams and memoization*. PhD thesis, University of Potsdam, Potsdam, Germany, Nov. 2005.
- [7] G.-M. Greuel and G. Pfister. *A SINGULAR Introduction to Commutative Algebra*. Springer Verlag, 2002.
- [8] G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR 3.0. A Computer Algebra System for Polynomial Computations, Centre for Computer Algebra, University of Kaiserslautern, 2005.
- [9] H. Kitano. Systems biology: A brief overview. *Science*, 295:1662–1664, Mar. 2002.
- [10] R. Laubenbacher and B. Stigler. A Computational algebra approach to the reverse engineering of gene regulatory networks. *Journal of Theoretical Biology*, 229:523–537, 2004.
- [11] F. Somenzi. CUDD: CU decision diagram package. University of Colorado at Boulder, 2005. Release 2.4.1.

Q-adic Transform Revisited

Jean-Guillaume Dumas

Université de Grenoble, Laboratoire J. Kuntzmann, umr CNRS 5224.

BP 53X, 51, rue des Mathématiques. F38041 Grenoble, France.

Jean-Guillaume.Dumas@imag.fr

ABSTRACT

We present an algorithm to perform a simultaneous modular reduction of several residues. This enables to compress polynomials into integers and perform several modular operations with machine integer arithmetic. The idea is to convert the X -adic representation of modular polynomials, with X an indeterminate, to a q -adic representation where q is an integer larger than the field characteristic. With some control on the different involved sizes it is then possible to perform some of the q -adic arithmetic directly with machine integers or floating points. Depending also on the number of performed numerical operations one can then convert back to the q -adic or X -adic representation and eventually mod out high residues. In this note we present a new version of both conversions: more tabulations and a way to reduce the number of divisions involved in the process are presented. The polynomial multiplication is then applied to arithmetic and linear algebra in small finite field extensions.

Categories and Subject Descriptors:

G.4 [Mathematical Software]: Algorithm design and analysis; I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms; F.2.2 [Nonnumerical Algorithms and Problems]: Computations on discrete structures.

General Terms:

Algorithms, Performance.

Keywords:

Kronecker substitution ; Finite field ; Modular Polynomial Multiplication ; REDQ (simultaneous modular reduction) ; Small extension field ; DQT (Discrete Q-adic Transform) ; FQT (Fast Q-adic Transform).

1. INTRODUCTION

The FFLAS/FFPACK project has demonstrated the usefulness of wrapping cache-aware routines for efficient small finite field linear algebra [4, 5].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

A conversion between a modular representation of prime fields and e.g. floating points used exactly is natural. It uses the homomorphism to the integers. Now for extension fields (isomorphic to polynomials over a prime field) such a conversion is not direct. In [4] we proposed transforming the polynomials into a q -adic representation where q is an integer larger than the field characteristic. We call this transformation DQT for Discrete Q-adic Transform, it is a form of Kronecker substitution [7, §8.4]. With some care, in particular on the size of q , it is possible to map the operations in the extension field into the floating point arithmetic realization of this q -adic representation and convert back using an inverse DQT.

In this note we propose some implementation improvements: we propose to use a tabulated discrete logarithm for the DQT and we give a trick to reduce the number of machine divisions involved in the inverse. This then gives rise to an improved DQT which we thus call FQT (Fast Q-adic Transform). This FQT uses a simultaneous reduction of several residues, called REDQ, and some table lookup.

Therefore we recall in section 2 the previous conversion algorithm and discuss in section 3 about a floating point implementation of modular reduction. This implementation will be used throughout the paper to get fast reductions. We then present our new simultaneous reduction in section 4 and show in section 5 how a time-memory trade-off can make this reduction very fast. This fast reduction is then applied to modular polynomial multiplication with small prime fields in section 6. It is also applied to small extension field arithmetic and fast matrix multiplication over those fields in section 7.

2. Q-ADIC REPRESENTATION OF POLYNOMIALS

We follow here the presentation of [4] of the idea of [12]: polynomial arithmetic is performed a q -adic way, with q a sufficiently big prime or power of a single prime.

Suppose that $a = \sum_{i=0}^{k-1} \alpha_i X^i$ and $b = \sum_{i=0}^{k-1} \beta_i X^i$ are two polynomials in $\mathbb{Z}/p\mathbb{Z}[X]$. One can perform the polynomial multiplication ab via q -adic numbers. Indeed, by setting $\tilde{a} = \sum_{i=0}^{k-1} \alpha_i q^i$ and $\tilde{b} = \sum_{i=0}^{k-1} \beta_i q^i$, the product is computed in the following manner (we suppose that $\alpha_i = \beta_i = 0$ for $i > k-1$):

$$\tilde{a}\tilde{b} = \sum_{j=0}^{2k-2} \left(\sum_{i=0}^j \alpha_i \beta_{j-i} \right) q^j \quad (1)$$

Now if q is large enough, the coefficient of q^i will not ex-

ceed $q - 1$. In this case, it is possible to evaluate a and b as machine numbers (e.g. floating point or machine integers), compute the product of these evaluations, and convert back to polynomials by radix computations (see e.g. [7, Algorithm 9.14]). There just remains then to perform modulo p reductions on every coefficient as shown on example 1.

EXAMPLE 1. For instance, to multiply $a = X + 1$ by $b = X + 2$ in $\mathbb{Z}/3\mathbb{Z}[X]$ one can use the substitution $X = 100$: compute $101 \times 102 = 10302$, use radix conversion to write $10302 = q^2 + 3q + 2$ and reduce modulo 3 to get $a \times b = X^2 + 2$.

We call DQT the evaluation of polynomials modulo p at q and DQT inverse the radix conversion of a q -adic development followed by a modular reduction, as shown in algorithm 1.

Algorithm 1 Polynomial multiplication by DQT

Input Two polynomials v_1 and v_2 in $\mathbb{Z}/p\mathbb{Z}[X]$ of degree less than k .

Input a sufficiently large integer q .

Output $R \in \mathbb{Z}/p\mathbb{Z}[X]$, with $R = v_1.v_2$.

Polynomial to q -adic conversion

- 1: Set $\tilde{v}_1$ and $\tilde{v}_2$ to the floating point vectors of the evaluations at q of the elements of v_1 and v_2 . {Using e.g. Horner's formula}

One computation

- 2: Compute $\tilde{r} = \tilde{v}_1 \tilde{v}_2$

Building the solution

- 3: $\tilde{r} = \sum_{i=0}^{2k-2} \tilde{\mu}_i q^i$. {Using radix conversion, see e.g. [7, Algorithm 9.14]}
 - 4: For each i , set $\mu_i = \tilde{\mu}_i \bmod p$
 - 5: set $R = \sum_{i=0}^{2k-2} \mu_i X^i$
-

Depending on the size of q , the results can still remain exact and we obtain the following bounds generalizing that of [7, §8.4]:

THEOREM 1. [4] Let m be the number of available mantissa bits within the machine numbers and n_q be the number of polynomial products $v_1.v_2$ of degree k accumulated before the re-conversion. If

$$q > n_q k (p - 1)^2 \text{ and } (2k - 1) \log_2(q) < m, \quad (2)$$

then Algorithm 1 is correct.

Note that the integer q can be chosen to be a power of 2. Then the Horner like evaluation (line 1 of algorithm 1) of the polynomials at q is just a left shift. One can then compute this shift with exponent manipulations in floating point arithmetic and use native shift operator (e.g. the $<<$ operator in C) as soon as values are within the 32 (or 64 when available) bit range.

It is shown on [4, Figures 5 & 6] that this wrapping is already a pretty good way to obtain high speed linear algebra over some small extension fields. Indeed we were able to reach high peak performance, quite close to those obtained with prime fields, namely 420 Millions of finite operations per second (Mop/s) on a Pentium III, 735 MHz, and more than 500 Mop/s on a 64-bit DEC alpha 500 MHz. This was roughly 20 percent below the pure floating point performance and 15 percent below the prime field implementation.

3. EUCLIDEAN DIVISION BY FLOATING POINT ROUTINES

In the implementations of the proposed subsequent algorithms, we will make extensive use of Euclidean division in exact arithmetic. Unfortunately exact division is usually quite slow on modern computers. This division can thus be performed by floating point operations. Suppose we want to compute r/p where r and p are integers. Then their difference is representable by a floating point and, therefore, if r/p is computed by a floating point division with a rounding to nearest mode, [9, Theorem 1] assures that flooring the result gives the expected value. Now if a multiplication by a precomputed inverse of p is used (as is done e.g. in NTL [13]), proving the correctness for all r is more difficult, see [10] for more details. We therefore propose the following simple lemma which enables the use of the rounding upward mode to the cost of losing only one bit of precision:

LEMMA 1. For two positive integers p and r and $\epsilon > 0$, we have

$$\left\lfloor \frac{r}{p} \right\rfloor = \left\lfloor \left(r \left(\frac{1}{p} (1 + \epsilon) \right) \right) (1 + \epsilon) \right\rfloor \text{ as long as } r < \frac{1}{2\epsilon + \epsilon^2}.$$

PROOF. Consider $up \leq r < up + i$ with u, i positive integers and $i < p$. Then $\left\lfloor \frac{r}{p} \right\rfloor = u$ and $\frac{r}{p} (1 + \epsilon) (1 + \epsilon) = u + \frac{i}{p} + \frac{r}{p} (2\epsilon + \epsilon^2)$. The latter is maximal at $i = p - 1$. This proves that flooring is correct as long as $\frac{r}{p} (2\epsilon + \epsilon^2) < \frac{1}{p}$. $\square$

This proves that when rounding towards $+\infty$ it is possible to perform the division by a multiplication by the precomputed inverse of the prime number as long as r is not too large. Since our entries will be integers but stored in floating point format this is a potential significant speed-up.

4. REDQ: MODULAR REDUCTION IN THE DQT DOMAIN

The first improvement we propose to the DQT is to replace the costly modular reduction of the polynomial coefficients by a single division by p (or, better, by a multiplication by its inverse) followed by several shifts. In order to prove the correctness of this algorithm, we first need the following lemma:

LEMMA 2. For $r \in \mathbb{N}$ and $a, b \in \mathbb{N}^*$,

$$\left\lfloor \frac{\left\lfloor \frac{r}{b} \right\rfloor}{a} \right\rfloor = \left\lfloor \frac{r}{ab} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{r}{a} \right\rfloor}{b} \right\rfloor$$

PROOF. We proceed by splitting the possible values of r into intervals $kab \leq r < (k+1)ab$, where $k = \left\lfloor \frac{r}{ab} \right\rfloor$. Then $kb \leq \frac{r}{a} < (k+1)b$ and since kb is an integer we also have that $kb \leq \left\lfloor \frac{r}{a} \right\rfloor < (k+1)b$. Thus $k \leq \frac{\left\lfloor \frac{r}{a} \right\rfloor}{b} < k+1$ and $\left\lfloor \frac{\left\lfloor \frac{r}{a} \right\rfloor}{b} \right\rfloor = k$. Obviously the same is true for the left hand side which proves the lemma. $\square$

This idea is used in algorithm 2 to perform several remainderings with a single machine division (note that when q is a power of 2, and when elements are represented using an integral type, division by q^i and flooring are a single operation, a right shift).

THEOREM 2. Algorithm REDQ is correct.

Algorithm 2 REDQ

Input Two integers p and q satisfying the conditions (2).

Input $\tilde{r} = \sum_{i=0}^d \tilde{\mu}_i q^i \in \mathbb{Z}$.

Output $\rho \in \mathbb{Z}$, with $\rho = \sum_{i=0}^d \mu_i q^i$ where $\mu_i = \tilde{\mu}_i \bmod p$.

```
1:  $rop = \left\lfloor \frac{\tilde{r}}{p} \right\rfloor$ ;
2: for  $i = 0$  to  $d$  do
3:    $u_i = \left\lfloor \frac{\tilde{r}}{q^i} \right\rfloor - p \left\lfloor \frac{rop}{q^i} \right\rfloor$ ;
4: end for
5:  $\mu_d = u_d$ 
6: for  $i = 0$  to  $d-1$  do
7:    $\mu_i = u_i - qu_{i+1} \bmod p$ ;
8: end for
9: Return  $\rho = \sum_{i=0}^d \mu_i q^i$ ;
```

PROOF. First we need to prove that $0 \leq u_i < p$. By definition of the truncation, we have $\frac{\tilde{r}}{q^i} - 1 < \left\lfloor \frac{\tilde{r}}{q^i} \right\rfloor \leq \frac{\tilde{r}}{q^i}$ and $\frac{\tilde{r}}{pq^i} - 1 - \frac{1}{q^i} < \left\lfloor \frac{rop}{q^i} \right\rfloor \leq \frac{\tilde{r}}{pq^i}$. Thus $-1 < u_i < p + \frac{p}{q^i}$, which is $0 \leq u_i \leq p$ since u_i is an integer. We now consider the possible case $u_i = p$ and show that it does not happen. $u_i = p$ means that $\left\lfloor \frac{\tilde{r}}{q^i} \right\rfloor = p(1 + \left\lfloor \frac{rop}{q^i} \right\rfloor) = pg$. This means that $pgq^i \leq r < pgq^i + q^i$. So that in turns $gq^i \leq rop \leq \frac{\tilde{r}}{p} < gq^i + \frac{q^i}{p}$. Thus $g \leq \frac{rop}{q^i} < g + \frac{1}{p}$ so that $\left\lfloor \frac{rop}{q^i} \right\rfloor = g$. But then from the definition of g we have that $g = g - 1$ which is absurd. Therefore $0 \leq u_i \leq p-1$.

Second we show that $u_i = \sum_{j=i}^d \mu_j q^{j-i} \bmod p$. Line 3 of algorithm 2 defines $u_i = \left\lfloor \frac{\tilde{r}}{q^i} \right\rfloor - p \left\lfloor \frac{\left\lfloor \frac{\tilde{r}}{p} \right\rfloor}{q^i} \right\rfloor$ and thus lemma 2 gives that $u_i = \left\lfloor \frac{\tilde{r}}{q^i} \right\rfloor - p \left\lfloor \frac{\left\lfloor \frac{\tilde{r}}{p} \right\rfloor}{q^i} \right\rfloor$. The latter is $u_i = \left\lfloor \frac{\tilde{r}}{q^i} \right\rfloor \bmod p$. Now, since $\tilde{r} = \sum_{j=0}^d \tilde{\mu}_j q^j$, we have that $\left\lfloor \frac{\tilde{r}}{q^i} \right\rfloor = \sum_{j=i}^d \tilde{\mu}_j q^{j-i}$. Therefore, as $\mu_j = \tilde{\mu}_j \bmod p$, the equality is proven. $\square$

Note that the last steps are not needed when p divides q . Indeed in this case $q \equiv 0 \bmod p$. The trick works then simply as shown on example 2 below:

EXAMPLE 2. Let $a = X^2 + 2X + 3$ and $b = 4X^2 + 5X + 6$ unreduced modulo 5. Then $\tilde{a} \times \tilde{b} = 40013002800270018$, with $q = 10000$, for which we need to reduce five coefficients modulo 5. The trick is that we can recover all the residues at once. Line 1 produces $rop = [08002600560054003.6]$. It thus contains all the quotients $0;0002;0005;0005;0003$ and one has then just to multiply by p and subtract to get: $\tilde{a} \times \tilde{b} = 40013002800270018 - 2000500050003 \times 5 = 40003000300020003$ so that $a \times b = 4X^4 + 3X^3 + 3X^2 + 2X + 3$.

Now we can give a full example to show the last corrections required when p does not divide q . The first part of the algorithm, lines 1 to 3 is unchanged and is used to get small sizes for μ_i . The second part is then just a small correction modulo p to get the correct result.

EXAMPLE 3. Take the polynomial $R = 1234X^3 + 5678X^2 + 9123X + 4567$, the prime $p = 23$ and use $q = 10^6$. In this case, the division gives $rop = [1234005678009123004567/23] = 53652420783005348024$. Then the multiplication by the prime produces $rop \times 23 = 1234005678009123004552$ so that

$u_0 = 4567 - 4552 = 15$. We shift to get 1234005678009123 and $53652420783005 \times 23 = 1234005678009115$ which gives $u_1 = 9123 - 9115 = 8$. We shift and multiply twice to get $u_2 = 18$ and $u_3 = \mu_3 = 15$ just like in example 2. Now $-q = -10^6 \bmod 23 = 17$ which is non zero and thus we have to compute the corrections of lines 5 to 8 of algorithm 2. This can also be formalized as a matrix vector product:

$$\mu = \begin{bmatrix} 1 & 17 & 0 & 0 \\ 0 & 1 & 17 & 0 \\ 0 & 0 & 1 & 17 \\ 0 & 0 & 0 & 1 \end{bmatrix} u \bmod p$$

to get the final result, $R = 15X^3 + 20X^2 + 15X + 13 \bmod 23$.

The algorithm is efficient because one can precompute $1/p$, $1/q$, $1/q^2$ etc. and use multiplication to compute all of the mods. The computation of each u_i and μ_i can also be pipelined or vectorized since they are independent. As is, the benefit when compared to direct remaindering by p is that the corrections occur on smaller integers. Thus the remaindering by p can be faster. Actually, another major acceleration can be added: the fact that the μ_i are much smaller than the initial $\tilde{\mu}_i$ makes it possible to tabulate the corrections as shown next.

5. TIME-MEMORY TRADE-OFF IN REDQ

5.1 A Matrix version of the correction

Indeed, there is a bijection between the u_i and the μ_i . This can be viewed on the corrections of lines 5 to 8 of algorithm 2: view these corrections as a matrix-vector multiplication by a matrix Q_d as in example 3. Then we have that:

$$Q_d = \begin{bmatrix} 1 & -q & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & -q \\ 0 & \dots & \dots & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & q & q^2 & \dots & q^d \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & q^2 \\ \vdots & & \ddots & \ddots & q \\ 0 & \dots & \dots & 0 & 1 \end{bmatrix}^{-1}$$

5.2 Tabulations of the matrix-vector product and Time-Memory trade-off

Thus if the multiplication by Q_d is fully tabulated, it requires a table of size at least p^{d+1} . But, due to the nature of Q_d , we have the relations of figure 1.

$$Q_{2d} = \begin{bmatrix} \boxed{Q_d} & 0 \\ 0 & \boxed{Q_d} \end{bmatrix} \quad Q_{2d+1} = \begin{bmatrix} \boxed{Q_{d+1}} & 0 \\ 0 & \boxed{Q_d} \end{bmatrix}$$

Figure 1: Recurring relations on the Q_d matrices.

Therefore, it is very easy to tabulate with a table of size p^k only and perform $\left\lceil 1 + \frac{d+1-k}{k-1} \right\rceil = \left\lceil \frac{d}{k-1} \right\rceil$ table accesses as shown on example 4.

EXAMPLE 4. Let us compute the corrections for a degree 6 polynomial. One can tabulate the multiplication by Q_6 , a 7×7 matrix, with therefore p^7 entries each of size at least $7 \log_2(p)$. Or one can tabulate the multiplication by Q_2 , a 3×3 matrix. To compute $[\mu_0, \dots, \mu_6]^T = Q_6[u_0, \dots, u_6]^T$ one can instead use three multiplications by Q_2 and discard the last entry for the first two multiplications as shown on the following algorithm:

Algorithm 3 Q_6 with an extra memory of size p^3

Input $[u_0, \dots, u_6] \in \mathbb{Z}/p\mathbb{Z}^7$.

Input The table Q_2 of the associated 3×3 matrix-vector multiplication over $\mathbb{Z}/p\mathbb{Z}$.

Output $[\mu_0, \dots, \mu_6]^T = Q_6[u_0, \dots, u_6]^T$.

1: $a_0, a_1, a_2 = Q_2[u_0, u_1, u_2]$;

2: $b_0, b_1, b_2 = Q_2[u_2, u_3, u_4]$;

3: $c_0, c_1, c_2 = Q_2[u_4, u_5, u_6]$;

4: Return $[\mu_0, \dots, \mu_6]^T = [a_0, a_1, b_0, b_1, c_0, c_1, c_2]^T$;

When q is a power of 2, the computation of the u_i , in the first part of algorithm 2 requires 1 div & $(d+1)$ mul & $2d$ shifts. Now, the time memory trade-off enables to compute the second part at a choice of costs given on table 1.

Extra Memory	time
0	d (mul, add, mod)
p^2	d accesses
p^k	$\left\lceil \frac{d}{k-1} \right\rceil$ accesses
p^{d+1}	1 access

Table 1: Time-Memory trade-off in REDQ of degree d over $\mathbb{Z}/p\mathbb{Z}$

5.3 Indexing

In practice, indexing by a t -uple of integers mod p is made by evaluating at p , as $\sum u_i p^i$. If more memory is available, one can also directly index in the binary format using $\sum u_i \left(2^{\lceil \log_2(p) \rceil}\right)^i$. On the one hand all the multiplications by p are replaced by binary shifts. On the other hand, this makes the table grow a little bit, from p^k to $2^{\lceil \log_2(p) \rceil k}$.

6. COMPARISON WITH DELAYED REDUCTION FOR POLYNOMIAL MULTIPLICATION

The classical alternative to algorithm 1 to perform modular polynomial multiplication is to use delayed reductions e.g. as in [1]: the idea is to accumulate products of the form $\sum_i a_i b_{k-i}$, without reductions, while the sum does not overflow. Thus, if we use for instance a centered representation modulo p (integers from $\frac{1-p}{2}$ to $\frac{p-1}{2}$), it is possible to accumulate at least n_d products as long as

$$n_d(p-1)^2 < 2^{m+1} \quad (3)$$

The modular reduction can be made by many different ways (e.g. classical division, floating point multiplication by the inverse, Montgomery reduction, etc.), we just call the best one REDC here. It is at most equivalent to 1 machine division.

Now the idea of the FQT (Fast Q-adic Transform) is to represent modular polynomials of the form $P = \sum_{i=0}^N a_i X^i$ by $P = \sum P_i (X^{d+1})^i$ where the P_i are degree d polynomials stored in a single integer in the q -adic way.

Therefore, a product PQ has the form

$$\sum \left(\sum P_i Q_{t-i} \right) (X^{d+1})^t.$$

There, each multiplication $P_i Q_{t-i}$ is made by algorithm 1 on a single machine integer. The reduction is made by a tabulated REDQ and can also be delayed now as long as conditions (2) are guaranteed.

We want to compare these two strategies. We thus propose the following complexity model: we count only multiplications and additions in the field as an atomic operation and separate the machine divisions. We for instance approximate REDC by a machine division. We call REDQ $_k$ a simultaneous reduction of k residues. In our complexity model a REDQ $_k$ thus requires 1 division and $2k$ multiplications and additions. We also call d -FQT the use of a degree d q -adic substitution. Thus a multiplication $P_i Q_{t-i}$ in a d -FQT requires the reduction of $2d+1$ coefficients, i.e. a REDQ $_{2d+1}$.

Let P be a polynomial of degree N with indeterminate "X". If we use a d -FQT, it will then become a polynomial of degree D_q in the indeterminate $Y = X^{d+1}$. Thus,

$$D_q = \left\lceil \frac{N+1}{d+1} \right\rceil - 1.$$

Table 2 gives the respective complexities of both strategies.

	Mul & Add	Reductions
Delayed	$(2N+1)^2$	$(2N+1) \left\lceil \frac{2N+1}{n_d} \right\rceil$ REDC
d -FQT	$(2D_q+1)^2$	$(2D_q+1) \left\lceil \frac{2D_q+1}{n_q} \right\rceil$ REDQ $_{2d+1}$

Table 2: Modular polynomial multiplication complexities.

For instance, with $p = 3$, $N = 500$, if we choose a double floating point representation and a degree 4 DQT, the fully tabulated FQT boils down to 10^5 multiplications and additions and $4 \cdot 10^3$ divisions. For the same parameters, the classical polynomial multiplication algorithm requires 10^6 multiplications and additions and only 10^3 remaindering. This is roughly 9 times more operations as shown on figure 2.

Even by switching to a larger mantissa, say e.g. 128 bits, so that the DQT multiplications are roughly 4 times costlier than double floating point operations, this can still be useful: take $p = 1009$ and choose $d = 3$, still gives around 10^5 multiplications and additions over 128 bits and $4 \cdot 10^3$ divisions. This makes 8 times less operations. This should therefore still be faster than the delayed over 32 bits.

On figure 2, we compare also our two implementations with that of NTL [13]. We see that the FQT is faster than NTL as long as better algorithms are not used. Indeed the change of slope in NTL's curve reflects the use of Karatsuba's algorithm for polynomial multiplication. One should note that NTL also proposes a very optimized modulo 2 implementation which is an order of magnitude faster than our implementation on small primes. There is therefore room for more improvements on small fields. Our strategy is anyway very useful for small degrees and small primes. Furthermore,

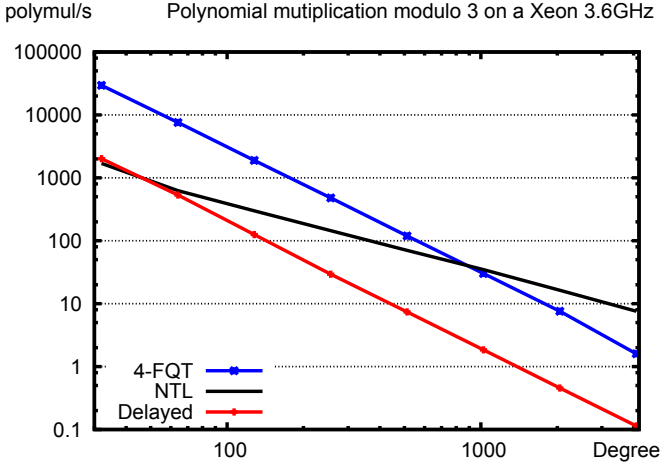


Figure 2: Polynomial multiplications modulo 3 per second on a Xeon 3.6 GHz

we have not implemented the FQT as the base case of faster recursive algorithms such as Karatsuba, Toom-Cook, etc. The figure shows that these recursive algorithms together with the FQT could be the fastest.

In particular, the FQT already improves the speed of small finite field extension's arithmetic as shown next.

7. APPLICATION TO SMALL FINITE FIELD EXTENSIONS

The isomorphism between finite fields of equal sizes gives us a canonical representation: any finite field extension is viewed as the set of polynomials modulo a prime p and modulo an irreducible polynomial $\mathcal{P}$ of degree k . Clearly we can thus convert any finite field element to its q -adic expansion; perform the FQT between two elements and then reduce the obtained polynomial modulo $\mathcal{P}$. Furthermore, it is possible to use floating point routines to perform exact linear algebra as demonstrated in [6].

Our strategy here, see algorithm 4, is thus to convert vectors over $\mathbf{GF}(p^k)$ to q -adic floating point, to call a fast numerical linear algebra routine (BLAS) and then to convert the floating point result back to the usual field representation. In this paper we propose to improve all the conversion steps of [4, algorithm 4.1] and thus approach the performance of the prime field wrapping also for small extension fields:

1. Replace the Horner evaluation of the polynomials, to form the q -adic expansion, by a single table lookup, recovering directly the floating point representation.
2. Replace the radix conversion and the costly modular reductions of each polynomial coefficient, by a single REDQ operation.
3. Replace the polynomial division by two table lookups and a single field operation.

Indeed, suppose the internal representation of the extension field is already by discrete logarithms and uses conversion tables from polynomial to index representations. See e.g. [1] for more details. Then we choose a time-memory trade-off for the REDQ operation of the same order of magnitude,

that is to say p^k . The overall memory required by these new tables only doubles and the REDQ requires only 2 accesses. Moreover, in the small extension, the polynomial multiplication must also be reduced by an irreducible polynomial, $\mathcal{P}$. We show next that this reduction can be precomputed in the REDQ table lookup and is therefore almost free.

Moreover, many things can be factorized if the field representation is by discrete logarithms. Indeed, the elements are represented by their discrete logarithm with respect to a generator of the field, instead of by polynomials. In this case there are already some table accesses for many arithmetic operations, see e.g. [1, §2.4] for more details.

More precisely, we here propose algorithm 4 for linear algebra over extension fields: line 1 is the table look-up of floating point values associated to elements of the field; line 2 is the numerical computation; line 3 to 7 is the first part of the REDQ reduction; line 8 and 9 are a time-memory trade-off with two table accesses for the corrections of REDQ, combined with a conversion from polynomials to discrete logarithm representation; the last line 10 combines the latter two results, inside the field.

Algorithm 4 Fast Dot product over Galois fields via FQT and FQT inverse

Input a field $\mathbf{GF}(p^k)$ with elements represented as exponents of a generator of the field.
Input Two vectors v_1 and v_2 of elements of $\mathbf{GF}(p^k)$.
Input a sufficiently large integer q .
Output $R \in \mathbf{GF}(p^k)$, with $R = v_1^T \cdot v_2$.

Tabulated q -adic conversion

{Use conversion tables from exponent to floating point evaluation}

- 1: Set $\tilde{v}_1$ and $\tilde{v}_2$ to the floating point vectors of the evaluations at q of the elements of v_1 and v_2 .

The floating point computation

- 2: Compute $\tilde{r} = \tilde{v}_1^T \tilde{v}_2$;

Computing a radix decomposition

- 3: $r = \lfloor \tilde{r} \rfloor$; { $r = \tilde{r}$ but we might need a conversion to an integral type}
- 4: $rop = \lfloor \frac{\tilde{r}}{p} \rfloor$;
- 5: **for** $i = 0$ to $2k - 2$ **do**
- 6: $u_i = \lfloor \frac{r}{q^i} \rfloor - p \lfloor \frac{rop}{q^i} \rfloor$;
- 7: **end for**

Tabulated radix conversion to exponents of the generator

{ μ_i is such that $\mu_i = \tilde{\mu}_i \bmod p$ for $\tilde{r} = \sum_{i=0}^{2k-2} \tilde{\mu}_i q^i$ }

- 8: Set $L = \text{representation}(\sum_{i=0}^{k-2} \mu_i X^i)$.
- 9: Set $H = \text{representation}(X^{k-1} \times \sum_{i=k-1}^{2k-2} \mu_i X^{i-k+1})$.

Reduction in the field

- 10: Return $R = H + L \in \mathbf{GF}(p^k)$;
-

A variant of REDQ is used in algorithm 4, but u_i still satisfies $u_i = \sum_{j=i}^{2k-2} \mu_j q^{j-i} \bmod p$ as shown in theorem 2. Therefore the representations of $\sum \mu_i X^j$ in the field can be precomputed and stored in two tables where the indexing will be made by $(u_0, \dots, u_{k-1})$ and $(u_{k-1}, \dots, u_{2k-2})$ and not by the μ_i 's as shown next.

THEOREM 3. *Algorithm 4 is correct.*

PROOF. There remains to prove that it is possible to compute L and H from the u_i . From the equality above, we see that $\mu_{2k-2} = u_{2k-2}$ and $\mu_i = u_i - qu_{i+1} \bmod p$, for $i = 0..(2k-3)$. Therefore a precomputed table of p^k entries, indexed by $(u_0, \dots, u_{k-1})$, can provide the representation of

$$L = \sum_{i=0}^{k-2} (u_i - qu_{i+1} \bmod p) X^i.$$

Another table with p^k entries, indexed by $(u_{k-1}, \dots, u_{2k-2})$, can provide the representation of

$$H = u_{2k-2} X^{2k-2} + \sum_{i=k-1}^{2k-3} (u_i - qu_{i+1} \bmod p) X^i.$$

Finally $R = X^{k-1} \times \sum_{i=k-1}^{2k-2} \mu_i X^{i-k+1} + \sum_{i=0}^{k-2} \mu_i X^i$ needs to be reduced modulo the irreducible polynomial used to build the field. But, if we are given the representations of H and L in the field, R is then equal to their sum inside the field, directly using the internal representations. $\square$

Table 3 recalls the respective complexities of conversion phase in the two presented algorithms.

Memory	Alg. 1 $3p^k$	Alg. 4 $6p^k$	Alg. 4 $4p^k + 2^{k \lceil \log_2 p \rceil + 1}$
Shift	$4k - 2$	$4k - 2$	$4k - 2$
Add	$4k - 4$	0	$2k - 1$
Axpy	0	$4k - 3$	$2k - 1$
Div	$2k - 1$	0	0
Table	0	3	3
Red	$\geq 5k$	4	4

Table 3: Complexity of the back and forth conversion between extension field and floating point numbers

Figure 4 shows only the speed of the conversion after the floating point operations. The log scales prove that for q ranging from 2^1 to 2^{26} (on a 32 bit Xeon) our new implementation is two to three times faster than the previous one.

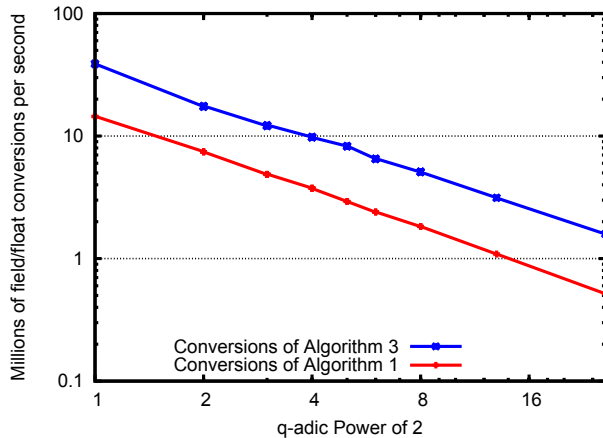


Figure 4: Small extension field conversion speed on a Xeon 3.6GHz

Furthermore, these improvements e.g. allow the extension field routines to reach the speed of 7800 millions of GF(9) operations per second (on a XEON, 3.6 GHz, using Goto BLAS-1.09 `dgemm` as the numerical routine [8] and FFLAS `fgemm` for the fast prime field matrix multiplication [6]) as shown on figure 3. The FFLAS routines are available within the LinBox 1.1.4 library [11] and the FQT is implemented in the `givgfqext.h` file of the Givaro 3.2.9 library [3].

With these new implementations, the obtained speed-up shown in figure 3 represents a reduction from the 15 percent overhead of the previous implementation to less than 4 percent now, when compared to GF(11).

8. CONCLUSION

We have proposed a new algorithm for simultaneous reduction of several residues stored in a single machine word. For this algorithm we also give a time-memory trade-off implementation enabling very fast running time if enough memory is available.

We have shown very effective applications of this trick for both modular polynomial multiplication, and extension fields conversion to floating point. The latter allows efficient linear algebra routines over small extension fields but also linear algebra over small prime fields as shown in [2].

Further improvements include comparison of running times between choices for q . Indeed our experiments were made with q a power of two and large table lookup. With q a multiple of p the table lookup is not needed but divisions by q^i will be more expensive.

It would also be interesting to see how does the trick extend in practice to larger precision implementations: on the one hand the basic arithmetic slows down, but on the other hand the trick enables a more compact packing of elements (e.g. if an odd number of field elements can be stored inside two machine words, etc.).

9. REFERENCES

- [1] Jean-Guillaume Dumas. Efficient dot product over finite fields. In Victor G. Ganzha, Ernst W. Mayr, and Evgenii V. Vorozhtsov, editors, *Proceedings of the seventh International Workshop on Computer Algebra in Scientific Computing, Yalta, Ukraine*, pages 139–154. Technische Universität München, Germany, July 2004.
- [2] Jean-Guillaume Dumas, Laurent Fousse, and Bruno Salvy. Compressed modular matrix multiplication. In *Proceedings of the Milestones in Computer Algebra 2008, Tobago*, May 2008.
- [3] Jean-Guillaume Dumas, Thierry Gautier, Pascal Giorgi, Clément Pernet, Jean-Louis Roch, and Gilles Villard. Givaro 3.2.9: C++ library for arithmetic and algebraic computations, 2007. ljk.imag.fr/CASYS/LOGICIELS/givaro.
- [4] Jean-Guillaume Dumas, Thierry Gautier, and Clément Pernet. Finite field linear algebra subroutines. In Teo Mora, editor, *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, Lille, France*, pages 63–74. ACM Press, New York, July 2002.
- [5] Jean-Guillaume Dumas, Pascal Giorgi, and Clément Pernet. FFPACK: Finite field linear algebra package. In Jaime Gutierrez, editor, *Proceedings of the 2004*

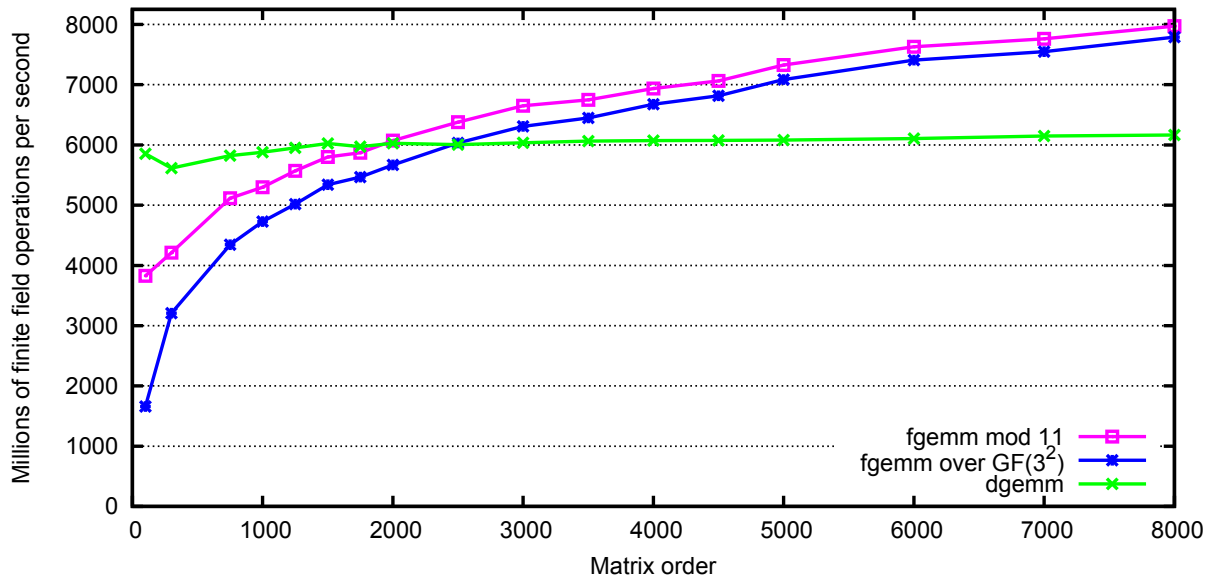


Figure 3: Speed of finite field Winograd matrix multiplication on a XEON, 3.6 GHz

- International Symposium on Symbolic and Algebraic Computation, Santander, Spain*, pages 119–126. ACM Press, New York, July 2004.
- [6] Jean-Guillaume Dumas, Pascal Giorgi, and Clément Pernet. Dense linear algebra over prime fields. *ACM Transactions on Mathematical Software*, 2009. to appear.
- [7] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, New York, NY, USA, 1999.
- [8] Kazushige Goto and Robert van de Geijn. On reducing TLB misses in matrix multiplication. Technical Report TR-2002-55, University of Texas, November 2002. FLAME working note #9.
- [9] Vincent Lefèvre. The Euclidean division implemented with a floating-point division and a floor. Technical report, INRIA Rhône-Alpes, 2005. <http://hal.inria.fr/inria-00000154>.
- [10] Vincent Lefèvre. The Euclidean division implemented with a floating-point multiplication and a floor. Technical report, INRIA Rhône-Alpes, 2005. <http://hal.inria.fr/inria-00000159>.
- [11] The LinBox Group. Linbox 1.1.4: Exact computational linear algebra, 2007. www.linalg.org.
- [12] B. David Saunders. Personal communication, 2001.
- [13] Victor Shoup. NTL 5.4.1: A library for doing number theory, 2007. www.shoup.net/ntl.

Computing the Global Optimum of a Multivariate Polynomial over the Reals

Mohab Safey El Din
INRIA, Paris-Rocquencourt Center, SALSA Project
UPMC, Univ Paris 06, LIP6
CNRS, UMR 7606, LIP6
UFR Ingénierie 919, LIP6 Passy-Kennedy,
Case 169, 4, Place Jussieu, F-75252 Paris Cedex
Mohab.Safey@lip6.fr

ABSTRACT

Let f be a polynomial in $\mathbb{Q}[X_1, \dots, X_n]$ of degree D . We provide an efficient algorithm in practice to compute the global supremum $\sup_{x \in \mathbb{R}^n} f(x)$ of f (or its infimum $\inf_{x \in \mathbb{R}^n} f(x)$). The complexity of our method is bounded by $D^{\mathcal{O}(n)}$. In a probabilistic model, a more precise result yields a complexity bounded by $\mathcal{O}(n^7 D^{4n})$ arithmetic operations in $\mathbb{Q}$. Our implementation is more efficient by several orders of magnitude than previous ones based on quantifier elimination. Sometimes, it can tackle problems that numerical techniques do not reach. Our algorithm is based on the computation of generalized critical values of the mapping $x \rightarrow f(x)$, i.e. the set of points $\{c \in \mathbb{C} \mid \exists (x_\ell)_{\ell \in \mathbb{N}} \subset \mathbb{C}^n \ f(x_\ell) \rightarrow c, \ ||x_\ell|| \ ||d_{x_\ell} f|| \rightarrow 0 \text{ when } \ell \rightarrow \infty\}$. We prove that the global optimum of f lies in its set of generalized critical values and provide an efficient way of deciding which value is the global optimum.

Categories and Subject Descriptors

I.1.2 [Computing Methodologies]: Symbolic and Algebraic Manipulation—*Algorithms*; G.1.6 [Mathematics of computing]: Numerical Analysis—*Optimization*; F.2.2 [Theory of Computation]: Analysis of algorithms and problem complexity—*Non numerical algorithms and problems: Geometrical problems and computation*

General Terms

Algorithms

Keywords

Global optimization, Polynomial system solving, real solutions, complexity

1. INTRODUCTION

Let $f \in \mathbb{Q}[X_1, \dots, X_n]$ be a multivariate polynomial of degree D . We consider below the problem of computing the global supremum of f , i.e. $\sup_{x \in \mathbb{R}^n} f(x)$.

Motivation and description of the problem. Global optimization of multivariate polynomials is a classical problem of scientific computing which appears in numerous applications from various fields (control theory, chemistry, economics, etc.). Hence, obtaining certified and efficient algorithms in practice solving this problem is a question of first importance.

Recently, numerical methods have been developed to tackle this problem. The algorithm of [32] is designed for unconstrained optimization and solves in fact an LMI (Linear Matrix Inequality) relaxation of the polynomial optimization problem. The algorithm of [26] is designed for polynomial constraint optimization and constructs also, in a different way, an LMI relaxation of the initial problem. The problem is that, according to [21, 13], a lot of LMI problems are ill-conditioned. In general, mixing numerical instability and ill-conditioning provides bad results in terms of quality of the output. As far as we know, in the context of global polynomial optimization, the conditioning of LMI problems and their numerical stability is a subject which has not been intensively investigated.

In this paper, we focus on symbolic methods to compute $\sup_{x \in \mathbb{R}^n} f(x)$ without assumption on f . Our aim is to design an efficient algorithm in practice which can be complementary to numerical methods when they fail. We focus on certified algorithms, but we also consider probabilistic ones, for complexity estimates.

The first way of computing $\sup_{x \in \mathbb{R}^n} f(x)$ by computer algebra techniques is to rewrite this problem as a quantifier elimination problem: $\exists e \in \mathbb{R} \ \forall x \in \mathbb{R}^n \ f(x) \leq e$ and $\forall \varepsilon \in \mathbb{R} \ \exists x \in \mathbb{R}^n \ \varepsilon > 0 \ f(x) > e - \varepsilon$. This problem can be solved using the cylindrical algebraic decomposition algorithm [9]. Nevertheless, it has a complexity which is doubly exponential in the number of variables and polynomial in the degree of f . The best implementations of this algorithm are limited to problems having 3 or 4 variables. Quantifier elimination problems can also be solved using the critical point method (see [6] and [7, Chapter 14]). The complexity of these methods is doubly exponential in the number of quantifiers. Thus, in the case of global polynomial optimization, this yields an algorithm (see [7, Chapter 14, Section 14.4]) whose complexity is $D^{\mathcal{O}(n)}$ since the number of quantifiers

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

is fixed. Nevertheless, the reduction of global optimization problems to quantifier elimination induces a growth of the complexity constant. Moreover, the algorithms in [7, Chapter 14] are exclusively designed to obtain deterministic complexity results. Algebraic manipulations and infinitesimals which are introduced to obtain complexity results spoil practical computations. This explains why no efficient implementations have been derived from these algorithms. Finally, note that obtaining algorithms for the computation of the global supremum of a polynomial without reducing this problem to a quantifier elimination one was an open problem in the scope of computer algebra.

Main results. The main result of this paper is an efficient algorithm computing the global supremum of a polynomial without reduction to quantifier elimination.

It is well known that computing $\sup_{x \in B} f(x)$ where B is compact can be tackled by computing the critical values of the mapping $x \rightarrow f(x)$ and studying the values taken by f over the boundary of B . These techniques are not sufficient to compute $\sup_{x \in \mathbb{R}^n} f(x)$. Indeed, consider the polynomial $f = X^2 + (XY - 1)^2$. It is always positive and considering its values at the sequence of points $(\frac{1}{\ell}, \ell)$, it is easy to see that $\sup_{x \in \mathbb{R}^n} (-f(x)) = 0$, while $\{x \in \mathbb{R}^n \mid f(x) = 0\}$ is obviously empty. The phenomenon which occurs here is that the polynomial f tends to a supremum “at infinity”, i.e. each sequence of points $(x_\ell)_{\ell \in \mathbb{N}} \subset \mathbb{R}^n$ such that $f(x_\ell) \rightarrow 0$ when ℓ tends to ∞ is such that $\|x_\ell\|$ tends to ∞ when ℓ tends to ∞ . On the other hand, it is well known that if, for such a sequence of points, $\|\text{grad}_{x_\ell} f\|$ does not tend to 0 when ℓ tends to ∞ , $f(x_\ell)$ can't have a finite limit. This leads to consider the notion of *generalized critical values* of a polynomial mapping $x \in \mathbb{C}^n \rightarrow f(x) \in \mathbb{C}$ introduced in [23]. The set of generalized critical values is defined as the set of complex numbers c such that there exists a sequence of points $(x_\ell)_{\ell \in \mathbb{N}} \subset \mathbb{C}^n$ satisfying $f(x_\ell) \rightarrow c$ and for all $(i, j) \in \{1, \dots, n\}^2$, $(X_i \frac{\partial f}{\partial X_j})(x_\ell) \rightarrow 0$ when ℓ tends to ∞ .

The first result (see Theorem 5 below) relates $\sup_{x \in \mathbb{R}^n} f(x)$ to the set of asymptotic critical values of the mapping $x \in \mathbb{R}^n \rightarrow f(x)$. More precisely, the statement of Theorem 5 is: *Let $f \in \mathbb{Q}[X_1, \dots, X_n]$ and $\mathcal{E} = \{e_1, \dots, e_\ell\}$ (with $e_1 < \dots < e_\ell$) be the set of real generalized critical values of the mapping $x \in \mathbb{R}^n \rightarrow f(x)$. Then $\sup_{x \in \mathbb{R}^n} f(x) < \infty$ if and only if there exists $1 \leq i_0 \leq \ell$ such that $\sup_{x \in \mathbb{R}^n} f(x) = e_{i_0}$.* It remains to show how to determine if the supremum of f over $\mathbb{R}^n$ is finite and, in this case, which of the generalized critical values is $\sup_{x \in \mathbb{R}^n} f(x)$. This is the aim of Theorem 6 whose statement is: *Let $\mathcal{E} = \{e_1, \dots, e_\ell\}$ be the set of real generalized critical values of the mapping $x \in \mathbb{R}^n \rightarrow f(x)$ such that $e_1 < \dots < e_\ell$. Consider $\{r_0, \dots, r_\ell\}$ a set of rationals such that $r_0 < e_1 < r_1 < \dots < e_\ell < r_\ell$. The supremum of f over $\mathbb{R}^n$ is finite if and only if there exists $i_0 \in \{1, \dots, \ell\}$ such that $\{x \in \mathbb{R}^n \mid f(x) = r_{i_0-1}\} \neq \emptyset$ and $\forall j \geq i_0 \quad \{x \in \mathbb{R}^n \mid f(x) = r_j\} = \emptyset$ if and only if $\sup_{x \in \mathbb{R}^n} f(x) = e_{i_0}$ (in this case $\sup_{x \in \mathbb{R}^n} f(x) = e_{i_0}$).*

Hence, the algorithm we obtain consists first in computing the set of generalized critical values of the mapping $x \in \mathbb{R}^n \rightarrow f(x)$. Then, it tests the emptiness of the real counterpart of smooth hypersurfaces defined by $f - r_j = 0$. This allows us to compute $\sup_{x \in \mathbb{R}^n} f(x)$ without any reduction to a quantifier elimination problem. The computations are based on algebraic elimination and finding the real so-

lutions of zero-dimensional polynomial systems. Our implementation uses Gröbner bases. At the end of the paper, we present some experiments showing the efficiency of this strategy compared to the ones reducing global optimization problems to a quantifier elimination one.

The complexity of the algorithm of [7, Chapter 14] reducing global optimization problems to quantifier elimination is $D^{O(n)}$. It is thus important to ensure that the algorithm which is designed here is in the same complexity class.

We make the distinction between algorithms whose probability of success depends on the entries (this is the case of some numerical algorithms running with fixed precision), algorithms whose probability of success depend on random choices done during the computation, and certified algorithms (the result is always correct but the complexity of such algorithms can depend on random choices done during the computation).

Substituting the computations of Gröbner bases by the geometric resolution algorithm designed in [30], one gets a probabilistic algorithm whose complexity is $O(n^7 D^{4n})$ arithmetic operations in $\mathbb{Q}$. As far as we know, such a complexity bound had never been obtained previously.

The running of our algorithms depend on some random choices of matrices and points which are valid outside a proper Zariski-closed subset. This is a first obstacle to get a complexity result in a deterministic framework. An other problem comes from the computation of critical values of a polynomial using Gröbner bases. We show how to compute the critical values using Gröbner bases with a complexity bounded by $D^{O(n)}$. The deformation technique we use is interesting in itself. Indeed, by generalizing the strategy developed in [39] (which had lead to practical improvements), it can be used to real solve general singular polynomial systems. Finally, we prove that, under the assumption that the first random choices are valid, the complexity of this algorithm is $D^{O(n)}$.

Related works. In the scope of numerical techniques, looking for the global supremum of a polynomial is tackled by looking for the smallest $\varepsilon > 0$ such that $\varepsilon - f > 0$. In general, the strategy consists in rewriting $\varepsilon - f$ as a sum of squares via LMI relaxations (see the works of Parrillo-Sturm-fels [32, 33] and the works of Lasserre [26] for methods based on moment theory), while it is well-known that a polynomial can be positive without being a sum of squares. More recently, Nie, Demmel and Sturm-fels [31] proposed a method based on computations over the ideal generated by the partial derivatives of the studied polynomial (when this ideal is zero-dimensional and radical). In particular, this technique seems to be more numerically stable than the previous ones. Lasserre developed an other approach for computing the real radical of zero-dimensional systems [27]. In any case, computing over the gradient ideal does not allow us to obtain the supremum of the considered polynomial. In [45], Schweighofer introduces the use of asymptotic critical values to obtain a numerical procedure computing the global supremum (or infimum) of a polynomial. Nevertheless, the conditioning and the numerical stability of this method is not studied.

The notion of generalized critical values we use in this paper was introduced in [23]. A first algorithm computing them is given. Efficient algorithms are given in [42, 40]. This computation is used in algorithms solving polynomial systems of inequalities in the RAGLIB maple package [41].

An other ingredient of this paper is the emptiness test of the real counterpart of a smooth hypersurface. Single exponential algorithms in the number of variables are given in [18, 19, 20, 35]. They are based on computations of critical points and have not lead to efficient implementations. Algorithms given in [43, 44] and some of their variants are implemented in RAGLIB [41]. A particular study leading to efficient algorithms dealing with the case of singular hypersurfaces is done in [39]. Algorithms relying on the critical point method and using evaluation and lifting techniques to encode polar varieties are given in [3, 2, 5, 4].

Conclusions and perspectives. We provide the first certified algorithm based on computer algebra techniques computing the supremum of polynomials over the reals without using quantifier elimination. This leads to an efficient implementation which allows us to deal with problems which are intractable by previous methods. The complexity of the method is singly exponential in the number of variables. We plan to extend these results in two ways. The first question which arises is to decide if the computed global supremum is reached and provide at least one point at which it is reached if it is. The second one is: given a singular point of a polynomial, decide if it is a local optimum. Finally, solving global optimization problems under some constraints is an area where the techniques we develop here can be used.

Plan of the paper. Section 2 provides the definition, useful properties and an algorithm for the computation of the set of generalized critical values of a polynomial mapping. In Section 3, we recall the basics of an efficient algorithm computing sampling points in the real counterpart of a smooth hypersurface which is used to test the emptiness of the considered real hypersurfaces. Then, in Section 4, we give the algorithm computing $\sup_{x \in \mathbb{R}^n} f(x)$ and its proof. Section 5 is devoted to complexity results. Section 6 presents some practical results we obtained with the algorithm designed here.

2. PROPERTIES AND COMPUTATION OF GENERALIZED CRITICAL VALUES

In this section, we recall the definitions and basic properties of generalized critical values which can be found in [23].

DEFINITION 1. [23] *A complex number $c \in \mathbb{C}$ is a critical value of the mapping $\tilde{f} : y \in \mathbb{C}^n \rightarrow f(y)$ if and only if there exists $z \in \mathbb{C}^n$ such that $f(z) = c$ and $\frac{\partial \tilde{f}}{\partial X_1}(z) = \dots = \frac{\partial \tilde{f}}{\partial X_n}(z) = 0$.*

A complex number $c \in \mathbb{C}$ is an asymptotic critical value of the mapping $\tilde{f} : y \in \mathbb{C}^n \rightarrow f(y)$ if and only if there exists a sequence of points $(z_\ell)_{\ell \in \mathbb{N}} \subset \mathbb{C}^n$ such that:

- a) $f(z_\ell)$ tends to c when ℓ tends to ∞ .
- b) $\|z_\ell\|$ tends to $+\infty$ when ℓ tends to ∞ .
- c) for all $(i, j) \in \{1, \dots, n\}^2$ $\|X_i(z_\ell)\| \cdot \|\frac{\partial \tilde{f}}{\partial X_j}(z_\ell)\|$ tends to 0 when ℓ tends to ∞ . The set of generalized critical values is the union of the sets of critical values and asymptotic critical values of $\tilde{f}$.

In the sequel, we denote by $K_0(f)$ the set of critical values of $\tilde{f}$, by $K_\infty(f)$ the set of asymptotic critical values of $\tilde{f}$, and by $K(f)$ the set of generalized critical values of $\tilde{f}$ (i.e. $K(f) = K_0(f) \cup K_\infty(f)$). In [23], the authors prove the following result which can be seen as a generalized Sard's theorem for generalized critical values. Bounds on the number of generalized critical values can be found also in [22].

THEOREM 1. *Let f be a polynomial in $\mathbb{Q}[X_1, \dots, X_n]$ of degree D . The set of generalized critical values $K(f)$ of the mapping $\tilde{f} : x \in \mathbb{C}^n \rightarrow f(x) \in \mathbb{C}$ is Zariski-closed in $\mathbb{C}$. Moreover, $D\#K_\infty(f) + \#K_0(f) \leq D^n - 1$*

Given two topological spaces V and W , a polynomial mapping $F : V \rightarrow W$, and a subset $\mathcal{W}$ of W , we say that F realizes a locally trivial fibration on $V \setminus F^{-1}(\mathcal{W})$ if for all connected open set $U \subset W \setminus \mathcal{W}$, for all $e \in U$, denoting by π the projection on the second member of the cartesian product $F^{-1}(e) \times U$, there exists a diffeomorphism φ such that the following diagram

$$\begin{array}{ccc} F^{-1}(e) \times U & \xrightarrow{\varphi} & F^{-1}(U) \\ & \pi \downarrow & \downarrow F \\ & & U \end{array}$$

is commutative. The main interest of the set of *generalized critical values* relies on its topological properties which are summarized below and proved in [23].

THEOREM 2. [23] *The mapping $f_{\mathbb{C}} : x \in \mathbb{C}^n \rightarrow f(x) \in \mathbb{C}$ realizes a locally trivial fibration in $\mathbb{C}^n \setminus f_{\mathbb{C}}^{-1}(K(f_{\mathbb{C}}))$. The mapping $f_{\mathbb{R}} : x \in \mathbb{R}^n \rightarrow f(x) \in \mathbb{R}$ realizes a locally trivial fibration in $\mathbb{R}^n \setminus f_{\mathbb{R}}^{-1}(K(f_{\mathbb{R}}))$.*

The set of critical values of f can be computed by the routine CRITICALVALUES given below. It takes as input $f \in \mathbb{Q}[X_1, \dots, X_n]$ and returns a univariate polynomial whose set of roots is the set of critical values of f .

CRITICALVALUES
Input: a polynomial $f \in \mathbb{Q}[X_1, \dots, X_n]$.
Output: a univariate polynomial whose set of roots is the set of critical values of the mapping $x \rightarrow f(x)$.
• Compute a Gröbner basis for a monomial ordering eliminating $[X_1, \dots, X_n]$ of the ideal generated by $f - T, \frac{\partial f}{\partial X_1}, \dots, \frac{\partial f}{\partial X_n}$ (where T is a new variable). • Let G be the Gröbner basis previously computed, return the element of G lying in $\mathbb{Q}[T]$.

We show now how to compute the set of asymptotic critical values of $\tilde{f}$. In the sequel, we consider maps between complex or real algebraic varieties. The notion of properness of such maps will be relative to the topologies induced by the metric topologies of $\mathbb{C}$ or $\mathbb{R}$. A map $\phi : V \rightarrow W$ of topological spaces is said to be proper at $w \in W$ if there exists a neighborhood B of w such that $\phi^{-1}(\bar{B})$ is compact (where $\bar{B}$ denotes the closure of B). The map ϕ is said to be proper if it is proper at all $w \in W$.

Notation 1. In order to describe the routine computing the set of asymptotic critical values of $\tilde{f}$ we introduce the following notations:

- Let $\mathbf{A} \in GL_n(\mathbb{Q})$ and $g \in \mathbb{Q}[X_1, \dots, X_n]$. We denote by $g^{\mathbf{A}}$ the polynomial $g(\mathbf{A} \cdot \mathbf{X})$ where $\mathbf{X} = [X_1, \dots, X_n]$
- DIMENSION is a procedure taking as input a finite set of polynomials in $\mathbb{Q}[X_1, \dots, X_n]$ and return the dimension of the ideal they generate.
- Consider an ideal, encoded by a Gröbner basis G , defining a curve C in $\mathbb{C}^n$ and a projection $\pi : C \rightarrow E \subset \mathbb{C}^n$ where E is a 1-dimensional affine space.

We denote by $\text{SETOFNONPROPERNESS}(G, \pi)$ a polynomial whose set of roots is the set of non-properness of π ; $\text{SETOFNONPROPERNESS}$ denoting a procedure computing it. Such a routine is given in [44, 28].

THEOREM 3. [42, 40] *Suppose that for all $1 \leq i \leq n$, $\deg(f, X_i) \geq 1$. If the determinant of the hessian matrix associated to f is identically null, the set of asymptotic critical values of the mapping $x \rightarrow f(x)$ is empty. Else, there exists a Zariski-closed subset $\mathcal{A} \subset GL_n(\mathbb{C})$ such that for all $\mathbf{A} \in GL_n(\mathbb{Q}) \setminus \mathcal{A}$ the Zariski-closure of the constructible set defined by*

$$f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \dots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}} = 0, \quad \frac{\partial f^{\mathbf{A}}}{\partial X_n} \neq 0$$

has dimension 1.

The routine given below and denoted by $\text{ASYMPTOTICCRITICALVALUES}$ is given in [40]. It improves the one of [42]. It computes a finite set of points containing the set of generalized critical values of f . The input is a polynomial f and the output is a univariate polynomial whose set of roots contain the set of asymptotic critical values of $\tilde{f}$.

ASYMPTOTICCRITICALVALUES	
Input: a polynomial $f \in \mathbb{Q}[X_1, \dots, X_n]$ such that $\deg(f, X_i) \geq 1$ for $i = 1, \dots, n$.	
Output: a univariate polynomial non identically null whose set of roots contains the set of asymptotic critical values of $x \rightarrow f(x)$.	
• If the determinant of the Hessian matrix of f is 0 return 1. • Choose $\mathbf{A} \in GL_n(\mathbb{Q})$ and compute a Gröbner basis $G^{\mathbf{A}}$ of $I^{\mathbf{A}} = \langle f^{\mathbf{A}} - T, \frac{\partial f^{\mathbf{A}}}{\partial X_1}, \dots, \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}}, L \frac{\partial f^{\mathbf{A}}}{\partial X_n} - 1 \rangle \cap \mathbb{Q}[X_1, \dots, X_n]$ • While $\text{DIMENSION}(I^{\mathbf{A}}) \neq 1$ do – Choose an other matrix $\mathbf{A} \in GL_n(\mathbb{Q})$ • Return $\text{SETOFNONPROPERNESS}(G^{\mathbf{A}}, \pi)$ where π is the projection $(x_1, \dots, x_n, t) \rightarrow t$.	

REMARK 1. *Note that if there exists $1 \leq i \leq n$, the asymptotic critical values of f can still be computed by considering f in $\mathbb{Q}[X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n]$. Note also that the curve defined as the Zariski-closure of the complex-solution set of $f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \dots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}} = 0, \frac{\partial f^{\mathbf{A}}}{\partial X_n} \neq 0$ has a degree bounded by $(D-1)^{n-1}$. Thus, the set of non-properness of the projection on T restricted to this curve has a degree bounded by $(D-1)^{n-1}$.*

In the sequel, for the sake of simplicity, we identify a polynomial $f \in \mathbb{Q}[X_1, \dots, X_n]$ with the mapping $f_{\mathbb{C}} : x \in \mathbb{C}^n \rightarrow f(x) \in \mathbb{C}$.

3. DECIDING THE EMPTINESS OF THE REAL COUNTERPART OF A SMOOTH HYPERSURFACE

We study now how to decide the emptiness of the real counterpart of a smooth hypersurface $\mathcal{H}$ defined by $f = 0$ (where $f \in \mathbb{Q}[X_1, \dots, X_n]$), i.e. $\{x \in \mathcal{H} \mid \frac{\partial f}{\partial X_1}(x) = \dots = \frac{\partial f}{\partial X_n}(x) = 0\}$ is empty. The routine we present is based on [43] since

it is, in practice, the most efficient. From the complexity viewpoint, it also provides slightly better complexity bounds than the ones obtained in [3, 2, 5, 4] in a probabilistic model. Denote by π_i the canonical projection $(x_1, \dots, x_n) \rightarrow (x_1, \dots, x_i)$, by $\mathbf{p}_n = (p_1, \dots, p_n)$ an arbitrarily chosen point in $\mathbb{Q}^n$ and by $\mathbf{p}_i$ the point $(p_1, \dots, p_i) \in \mathbb{Q}^i$. The geometric scheme of resolution is based on the following results. Up to a generic linear change of variables,

- the set of critical points of the restriction of π_1 to $\mathcal{H}$ is either zero-dimensional or empty;
- $\dim(\mathcal{H} \cap \pi_1^{-1}(\mathbf{p}_1)) = \dim(\mathcal{H}) - 1$;
- each connected component C of $\mathcal{H} \cap \mathbb{R}^n$ has a closed image by π_1 . Hence, either $\pi_1(C) = \mathbb{R}$ which implies that $C \cap (\mathcal{H} \cap \pi_1^{-1}(\mathbf{p}_1)) \neq \emptyset$ or C contains a critical point of the restriction of π_1 to $\mathcal{H}$. Computing at least one point in each connected component is then reduced to computing the critical points of the restriction of π_1 to $\mathcal{H}$ and performing a recursion on $\mathcal{H} \cap \pi_1^{-1}(\mathbf{p}_1)$ (whose dimension is less than the one of $\mathcal{H}$). More precisely, the following result is proved in [43]. We use in the following the notations introduced in the previous section. Additionally, given an algebraic variety $V \subset \mathbb{C}^n$ and a polynomial mapping $\varphi : \mathbb{C}^n \rightarrow \mathbb{C}^k$ (for some $k \in \{1, \dots, n\}$), $\mathfrak{C}(\varphi, V)$ denotes the critical locus of φ restricted to V .

THEOREM 4. [43] *Let $\mathcal{H} \subset \mathbb{C}^n$ be a smooth hypersurface defined by $f = 0$ (with $f \in \mathbb{Q}[X_1, \dots, X_n]$). There exists a proper Zariski-closed subset $\mathcal{A} \subset GL_n(\mathbb{C})$ such that for all $\mathbf{A} \in GL_n(\mathbb{Q}) \setminus \mathcal{A}$, the set of points*

$$\mathfrak{C}(\pi_1, \mathcal{H}^{\mathbf{A}}) \cup (\mathcal{H}^{\mathbf{A}} \cap \pi_{n-1}^{-1}(\mathbf{p}_{n-1})) \bigcup_{i=2}^n \mathfrak{C}(\pi_i, \mathcal{H}^{\mathbf{A}} \cap \pi_{i-1}^{-1}(\mathbf{p}_i))$$

has at most dimension 0 and a non-empty intersection with each connected component with $\mathcal{H}^{\mathbf{A}} \cap \mathbb{R}^n$.

REMARK 2. *In [43], the authors prove that $\mathbf{A}$ must be chosen such that the projection π_i restricted to $\mathfrak{C}(\pi_{i+1}, \mathcal{H}^{\mathbf{A}})$ is proper. An algorithm performing this test is given in [44] (see also [28]).*

The algorithm below is called ISEMPTY . It takes as input a square-free polynomial f whose complex set of solutions is a smooth hypersurface. It returns **false** if $f = 0$ has real solutions, else it returns **true**. This algorithm requires a routine ZERODIMSOLVE taking as input a polynomial system of equations defining a zero-dimensional variety and returning its real points (encoded by numerical approximations).

ISEMPTY	
Input: a polynomial $f \in \mathbb{Q}[X_1, \dots, X_n]$ defining a smooth hypersurface $\mathcal{H} \subset \mathbb{C}^n$	
Output: a boolean which equals true if and only if $\mathcal{H} \cap \mathbb{R}^n$ is empty.	
• Choose randomly $\mathbf{A} \in GL_n(\mathbb{Q})$, and $\mathbf{p} \in \mathbb{Q}^n$. • If $\text{ZERODIMSOLVE}(f^{\mathbf{A}}, \frac{\partial f^{\mathbf{A}}}{\partial X_2}, \dots, \frac{\partial f^{\mathbf{A}}}{\partial X_n}) \neq \emptyset$ return false • For $i \in \{1, \dots, n-2\}$ if $\text{ZERODIMSOLVE}(X_1 - p_1, \dots, X_i - p_i, f^{\mathbf{A}}, \frac{\partial f^{\mathbf{A}}}{\partial X_{i+2}}, \dots, \frac{\partial f^{\mathbf{A}}}{\partial X_n}) \neq \emptyset$ return false • If $\text{ZERODIMSOLVE}(f^{\mathbf{A}}, X_1 - p_1, \dots, X_{n-1} - p_{n-1}) \neq \emptyset$ return false else return true.	

4. THE ALGORITHM

The algorithm computing the supremum of a polynomial $f \in \mathbb{Q}[X_1, \dots, X_n]$ is based on computations of generalized critical values and computations testing the emptiness of the real counterpart of smooth hypersurfaces.

The result below shows that if a polynomial f has a finite optimum, this optimum is either a critical value or an asymptotic critical value of f .

THEOREM 5. *Let $f \in \mathbb{Q}[X_1, \dots, X_n]$ and $\mathcal{E} = \{e_1, \dots, e_\ell\}$ (with $e_1 < \dots < e_\ell$) be the set of real generalized critical values of the mapping $x \in \mathbb{R}^n \rightarrow f(x)$. Then $\sup_{x \in \mathbb{R}^n} f(x) < \infty$ if and only if there exists $1 \leq i_0 \leq \ell$ such that $\sup_{x \in \mathbb{R}^n} f(x) = e_{i_0}$.*

PROOF. Suppose that the supremum of f is finite and let $e = \sup_{x \in \mathbb{R}^n} f(x)$. This is equivalent to the following assertion: there exists $\alpha > 0$ small enough such that for all $\varepsilon \in]e, e + \alpha[$, the real counterpart of $f^{-1}(\varepsilon)$ is empty and for all $\varepsilon \in]e - \alpha, e[$ the real counterpart of $f^{-1}(\varepsilon)$ is not empty. Denote by I the interval $]e - \alpha, e + \alpha[$.

Suppose now that e is not a generalized critical value of f . Then, from Theorem 2, f realizes a locally trivial fibration over $f^{-1}(]e - \alpha, e + \alpha[)$. Hence, for all $\varepsilon \in I$, there exists a diffeomorphism φ such that the following diagram

$$\begin{array}{ccc} f^{-1}(\varepsilon) \times I & \xrightarrow{\varphi} & f^{-1}(I) \\ & \pi \searrow & \downarrow f \\ & & I \end{array}$$

commutes. Consider $\varepsilon \in]e - \alpha, e[$, $\varepsilon' \in]e, e + \alpha[$ (then $f^{-1}(\varepsilon') \cap \mathbb{R}^n = \emptyset$) and $x \in f^{-1}(\varepsilon) \cap \mathbb{R}^n$. From the above diagram, $f(\varphi(x, \varepsilon')) = \varepsilon'$ implies that $\varphi(x, \varepsilon') \in f^{-1}(\varepsilon') \cap \mathbb{R}^n$. This is a contradiction. $\square$

From Theorem 5, if the supremum of a polynomial mapping is finite, it is a generalized critical value. Determining which of these values the supremum is, can be tackled by solving the following quantifier elimination problem: $\exists e \in \mathcal{E} \forall \varepsilon > 0, \exists x \in \mathbb{R}^n f(x) \geq e - \varepsilon$ and $\nexists x \in \mathbb{R}^n f(x) > e$ where $\mathcal{E}$ is the set of generalized critical values of the mapping $x \in \mathbb{R}^n \rightarrow f(x)$. The following result shows how to determine if a generalized critical value given by an isolating interval is an optimum of the polynomial f without performing a quantifier elimination.

THEOREM 6. *Let $\mathcal{E} = \{e_1, \dots, e_\ell\}$ be the set of real generalized critical values of the mapping $x \in \mathbb{R}^n \rightarrow f(x)$ such that $e_1 < \dots < e_\ell$. Consider $\{r_0, \dots, r_\ell\}$ a set of rationals such that $r_0 < e_1 < r_1 < \dots < e_\ell < r_\ell$.*

The supremum of f over $\mathbb{R}^n$ is finite if and only if there exists $i_0 \in \{1, \dots, \ell\}$ such that $\{x \in \mathbb{R}^n \mid f(x) = r_{i_0-1}\} \neq \emptyset$ and $\forall j \geq i_0 \quad \{x \in \mathbb{R}^n \mid f(x) = r_j\} = \emptyset$ if and only if $\sup_{x \in \mathbb{R}^n} f(x) = e_{i_0}$ (in this case $\sup_{x \in \mathbb{R}^n} f(x) = e_{i_0}$).

PROOF. Suppose the supremum of f over $\mathbb{R}^n$ to be finite. From Theorem 5, there exists a generalized critical value e_{i_0} of f such that $\sup_{x \in \mathbb{R}^n} f(x) = e_{i_0}$. Equivalently, we have that for all $e > e_{i_0}$, $f^{-1}(x) \cap \mathbb{R}^n$ is empty and there exists $\alpha > 0$ small enough such that for all $\varepsilon \in]e_{i_0} - \alpha, e_{i_0}[$ $f^{-1}(\varepsilon) \cap \mathbb{R}^n$ is not empty.

By convention, if $i_0 = 1$, $e_{i_0-1} = -\infty$. Denote by I the interval $]e_{i_0-1}, e_{i_0}[$. Now, it remains to prove that for all

$\varepsilon \in I$, $f^{-1}(\varepsilon) \cap \mathbb{R}^n$ is not empty. From Theorem 2, given $\varepsilon \in]e_{i_0} - \alpha, e_{i_0}[$, there exists a diffeomorphism φ such that the following diagram

$$\begin{array}{ccc} f^{-1}(\varepsilon) \times I & \xrightarrow{\varphi} & f^{-1}(I) \\ & \pi \searrow & \downarrow f \\ & & I \end{array}$$

commutes. Supposing now that there exists $\varepsilon' \in I$ such that $f^{-1}(\varepsilon') \cap \mathbb{R}^n$ is empty and consider $x \in f^{-1}(\varepsilon)$. Since $f(\varphi(x, \varepsilon')) = \varepsilon'$, implies $\varphi(x, \varepsilon') \in f^{-1}(\varepsilon') \cap \mathbb{R}^n$, this yields a contradiction. $\square$

Finally, computing the supremum of a polynomial is reduced to test the emptiness of the real counterpart of smooth hypersurfaces. This problem can be tackled using the routine `ISEMPTY` described in Section 3.

Denoting by `FINDRATIONALS` a routine taking as input a univariate polynomial with coefficients in $\mathbb{Q}$ whose real roots are denoted by $e_1 < \dots < e_\ell$ and returning a list of rational numbers $r_0 < \dots < r_\ell$ such that $r_0 < e_1 < r_1 < e_2 < \dots < e_\ell < r_\ell$, we obtain the following algorithm.

OPTIMIZE
Input: a polynomial $f \in \mathbb{Q}[X_1, \dots, X_n]$ Output: if $\sup_{x \in \mathbb{R}^n} f(x)$ is infinite, it returns ∞ else it returns an interval isolating $\sup_{x \in \mathbb{R}^n} f(x)$ and a polynomial whose set of roots contains $\sup_{x \in \mathbb{R}^n} f(x)$.
• $P \leftarrow \text{ASYMPTOTICCRITICALVALUES}(f)$ • $Q \leftarrow \text{CRITICALVALUES}(f)$ • $L \leftarrow \text{FINDRATIONALS}(PQ)$, $N \leftarrow \#L$, $i \leftarrow N$ • while $i > 0$ and <code>ISEMPTY(f-L[i])</code> do $i \leftarrow i - 1$ • if $i = 0$ return ∞ else return an interval isolating the i-th root of PQ and the polynomial PQ.

5. COMPLEXITY RESULTS

Complexity estimates using the geometric resolution algorithm. One can substitute Gröbner bases computations in the routines `CRITICALVALUES`, `ASYMPTOTICCRITICALVALUES` and `ISEMPTY` by the geometric resolution algorithm [17, 30]. This algorithm is probabilistic. It is based on evaluation and lifting techniques taking advantage of the encoding of polynomials by straight-line programs. These techniques are introduced in [16, 14, 15]. This leads to complexity estimates for solving zero-dimensional polynomial systems which are polynomial in the complexity of evaluation of the input system, the number of variables and an intrinsic geometric degree bounded by D^n (where D bounds the degree of the polynomials given as input).

In [42, 40], we describe algorithms computing critical values and asymptotic critical values using the geometric resolution algorithm given in [30] having a complexity bounded by $\mathcal{O}(n^7 D^{4n})$ arithmetic operations in $\mathbb{Q}$. The algorithm of [43] has a complexity bounded by $\mathcal{O}(n^7 D^{3n})$. Since the number of generalized critical values is bounded by $\mathcal{O}(D^n)$, there are, in the worst case, at most $\mathcal{O}(D^n)$ hypersurfaces for which one has to test the emptiness of their real counterpart (see Remark 1). This leads to the following result.

THEOREM 7. *There exists a probabilistic algorithm computing the global supremum of a polynomial $f \in \mathbb{Q}[X_1, \dots,$*

$X_n]$ of degree D with a complexity within $\mathcal{O}(n^7 D^{4n})$ arithmetic operations in $\mathbb{Q}$.

The rest of this section is devoted to prove that there exists a certified algorithm computing the global supremum of f , without reduction to quantifier elimination, with a complexity within $D^{\mathcal{O}(n)}$ arithmetic operations in $\mathbb{Q}$ if some random choices done internally are generic enough. Note that proving this is equivalent to prove the same result for the computation of asymptotic critical values of f , which is the aim of the next paragraph. Indeed, the computation of real critical values of f can be done by computing the values taken by f at sampling points of the real counterpart of the algebraic variety defined by $\frac{\partial f}{\partial X_1} = \dots = \frac{\partial f}{\partial X_n} = 0$ with a complexity within $D^{\mathcal{O}(n)}$ arithmetic operations in $\mathbb{Q}$ using [7, Chapter 13]. The same algorithm can be used to test the emptiness of the real counterpart of smooth hypersurfaces. We give below another way to get a complexity bounded by $D^{\mathcal{O}(n)}$ arithmetic operations in $\mathbb{Q}$ to compute the critical values of f . The method we use is of interest for itself since it generalizes [39] to the case of singular polynomial systems and can lead to efficient implementations.

Computation of asymptotic critical values. Suppose the determinant of the hessian matrix of f to be not identically null (the cost of this computation is dominated by $D^{\mathcal{O}(n)}$). From [40], there exists $\mathcal{A} \subset GL_n(\mathbb{C})$ such that for all $\mathbf{A} \in GL_n(\mathbb{Q}) \setminus \mathcal{A}$ the ideal $I^{\mathbf{A}} = J^{\mathbf{A}} \cap \mathbb{Q}[X_1, \dots, X_n, T]$ where $J^{\mathbf{A}} = \langle f^{\mathbf{A}} - T, \frac{\partial f^{\mathbf{A}}}{\partial X_1}, \dots, \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}}, L \frac{\partial f^{\mathbf{A}}}{\partial X_n} - 1 \rangle$ is radical and has dimension 1. The degree of this ideal is bounded by $(D-1)^{n-1}$ (see also [40] for more accurate bounds on the degree of this curve depending on intrinsic quantities). Moreover, there exists a proper Zariski-closed subset $\mathcal{Z} \subset \mathbb{C}$ such that for all $\theta \in \mathbb{Q} \setminus \mathcal{Z}$, the ideal $I^{\mathbf{A}} + \langle T - \theta \rangle$ has dimension at most 0. Choosing $(D-1)^{n-1} + 1$ such points x and computing a rational parametrization with respect to a separating element u of these ideals has a cost which is dominated by $D^{\mathcal{O}(n)}$. If $\mathbf{A}$ is well-chosen there exists at least one θ such that $I^{\mathbf{A}} + \langle T - \theta \rangle$ is radical which can be decided looking at the rational parametrization of its set of solutions (see [37]). If the separating element is generic enough, it is a separating element for all valid choices of θ . Once these rational parametrizations are computed, using interpolation, one obtains a rational parametrisation encoding the curve defined by $I^{\mathbf{A}}$. If the first choices of matrix $\mathbf{A}$, the rationals θ and u are correct, the cost of the whole computation is dominated by $D^{\mathcal{O}(n)}$ arithmetic operations in $\mathbb{Q}$. Following [42, Section 4], one can retrieve the set of non-properness of the projection $(x_1, \dots, x_n, t) \rightarrow t$ restricted to $\mathfrak{C}^{\mathbf{A}}$ from such a rational parametrization, with a complexity bounded by $D^{\mathcal{O}(n)}$ arithmetic operations in $\mathbb{Q}$.

THEOREM 8. *If the first choices of $\mathbf{A}$, θ and u are correct, the procedure described above has a complexity within $D^{\mathcal{O}(n)}$ arithmetic operations in $\mathbb{Q}$ and returns the set of asymptotic critical values of the mapping $x \rightarrow f(x)$.*

Computation of critical values. If the singular locus of f is zero-dimension, [24, 25] and FGLM algorithm allow us to compute the critical values of f with a complexity within $D^{\mathcal{O}(n)}$ arithmetic operations in $\mathbb{Q}$. We show now how to modify CRITICALVALUES to obtain a complexity result without assumption on the dimension of the singular locus of f . To this end, we use deformation techniques. The method

is interesting in itself. It generalizes to the case of singular polynomial systems the approach developed in [39]. This had lead to significative practical improvement for finding the real solutions of singular hypersurfaces. The complexity of the method we present here is $D^{\mathcal{O}(n)}$.

Given $(q, i_0) \in \mathbb{N} \times \mathbb{Z}$, an infinitesimal ε , and a Puiseux series field $a = \sum_{i \geq i_0} a_i \varepsilon^{i/q}$ (with $a_{i_0} \neq 0$) in $\mathbb{C}(\varepsilon)$, $\lim_0(a)$ exists if $i_0 \geq 0$ and equals a_0 . Given $x = (x_1, \dots, x_n) \in \mathbb{C}(\varepsilon)$, if for all i , $\lim_0(x_i)$ exists, x is said to have a bounded limit $\lim_0(x) = (\lim_0(x_1), \dots, \lim_0(x_n))$, else x is said to have a non-bounded limit. If V is a subset of $\mathbb{C}(\varepsilon)^n$ (or $\mathbb{R}(\varepsilon)^n$), $\lim_0(V)$ denotes the set of bounded limits of the points in V . We consider in the sequel algebraic varieties defined in $\mathbb{C}(\varepsilon)^n$. Given $\mathcal{I} = \{i_1, \dots, i_\ell\} \subset \{1, \dots, n\}$ and $\sigma \in \{-1, 1\}^n$, denote by $V_{\varepsilon, \mathcal{I}}^{\mathbf{a}, \sigma}$ the algebraic variety defined by the system $S_{\varepsilon, \mathcal{I}}^{\mathbf{a}, \sigma} \frac{\partial f}{\partial X_{i_1}} - \sigma_{i_1} a_{i_1} \varepsilon = \dots = \frac{\partial f}{\partial X_{i_\ell}} - \sigma_{i_\ell} a_{i_\ell} \varepsilon = 0$ (where ε is an infinitesimal). Given a point $A \in \mathbb{Q}^n$, denote by $\mathfrak{W}_{\varepsilon, \mathcal{I}}^{A, \mathbf{a}, \sigma}$ the set of critical points of the square of the euclidean distance function from A restricted to $V_{\varepsilon, \mathcal{I}}^{\mathbf{a}, \sigma}$. If $y \in \mathbb{R}^n$ and $E \subset \mathbb{R}^n$, $\text{dist}(y, E)$ denotes the minimum distance from y to E . Finally, denote by $S \subset \mathbb{C}^n$ the algebraic variety defined by $\frac{\partial f}{\partial X_1} = \dots = \frac{\partial f}{\partial X_n} = 0$.

LEMMA 1. *There exists a Zariski-closed subset $\mathcal{A} \subsetneq \mathbb{C}^n$ such that for all $\mathbf{a} \in \mathbb{Q}^n \setminus \mathcal{A}$, $S_{\varepsilon, \mathcal{I}}^{\mathbf{a}, \sigma}$ generates a radical ideal and $V_{\varepsilon, \mathcal{I}}^{\mathbf{a}, \sigma}$ is smooth of dimension $n - \#\mathcal{I}$ if it is not empty. Given such a $\mathbf{a}$, there exists a Zariski-closed subset $\mathcal{A}_{\mathbf{a}} \subsetneq \mathbb{C}^n$ such that for all $A \in \mathbb{Q}^n \setminus \mathcal{A}_{\mathbf{a}}$, $\mathfrak{W}_{\varepsilon, \mathcal{I}}^{A, \mathbf{a}, \sigma}$ is zero-dimensional or empty.*

PROOF. This is a consequence of the algebraic Sard's theorem applied to the mapping $x \in \mathbb{C}^n \rightarrow (f_{i_1}(x), \dots, f_{i_\ell}(x))$ and of [1, Theorem 2.3] (see also [4]) applied to the system $f_{i_1} - \sigma_{i_1} a_{i_1} \varepsilon = \dots = f_{i_\ell} - \sigma_{i_\ell} a_{i_\ell} \varepsilon = 0$. $\square$

THEOREM 9. *Given $A = (a_1, \dots, a_n) \in \mathbb{Q}^n$ and C , a connected component of $S \cap \mathbb{R}^n$, there exists $\mathcal{I} \subset \{1, \dots, n\}$ such that $(\lim_0(\mathfrak{W}_{\varepsilon, \mathcal{I}}^{A, \mathbf{a}, \sigma}) \cap S) \cap C \neq \emptyset$.*

PROOF. Consider a point of C which is at minimal distance to A and suppose, without loss of generality, that in any neighbourhood of x , there exists a point at which all the partial derivatives of f are positive. Consider the maximal subset $\mathcal{I} \subset \{1, \dots, n\}$ (for the order inclusion) such that in each neighbourhood of x there exists x'_ε such that $f_i(x'_\varepsilon) - a_i \varepsilon = 0$ for all $i \in \mathcal{I}$. Remark that $\lim_0 x'_\varepsilon$ exists and equals x . We follow now *mutatis mutandis* the proof of [38, Lemma 3.7], to obtain the result.

Consider the set $\mathcal{M}$ of points of C at minimal distance from A . Let $r > 0$ be small enough so that the closed and bounded semi-algebraic set $T = \{x \in \mathbb{R}^n \mid \exists y \in \mathcal{M} \text{ dist}(y, \mathcal{M} \leq r)\}$ does not intersect $S \setminus C$. According to the above $\lim_0(x_\varepsilon) \in \mathcal{M}$. Denoting $T' = \{x \in \mathbb{R}^n \mid \exists y \in \mathcal{M} \text{ dist}(y, \mathcal{M} = r)\}$, remark that the points of $\lim_0(\mathfrak{W}_{\varepsilon, \mathcal{I}}^{A, \mathbf{a}, \sigma}) \cap S \cap T'$ are infinitesimally close of points of $S \cap T'$ which are not at minimal distance from A . So the minimal distance from A to $\mathfrak{W}_{\varepsilon, \mathcal{I}}^{A, \mathbf{a}, \sigma} \cap T$ is not obtained on T' . Thus, this minimal distance is obtained at a point which is a critical point of the square of the distance function to A on V and it is clear that the limits of these points lie in $\mathcal{M}$. $\square$

In order to compute rational parametrizations $\mathfrak{W}_{\varepsilon, \mathcal{I}}^{A, \mathbf{a}, \sigma}$, we follow a similar strategy than the one designed in the above paragraph, by specializing ε in rational values, using interpolation to obtain rational parametrizations with coefficients

in $\mathbb{Q}(\varepsilon)$ and [38] to compute $\lim_0(x)$ for each point x encoded by these rational parametrizations. We obtain a complexity within $D^{\mathcal{O}(n)}$ arithmetic operations in $\mathbb{Q}$. Finally, note that the number of $\mathcal{I}$ to consider is bounded by $D^{\mathcal{O}(n)}$.

THEOREM 10. *If the first choices of $\mathbf{a}$, $\mathbf{A}$, specialisation values of ε and a separating element are generic enough, one can compute at least one point in each connected component of $S \cap \mathbb{R}^n$ within a complexity bounded by $D^{\mathcal{O}(n)}$.*

6. IMPLEMENTATION AND PRACTICAL EXPERIMENTS

The algorithm we implemented uses the routines described in Sections 2, 3 since, up to the experiments we did, they are more efficient than the ones described in Section 5 to obtain complexity results. The implementation is done in Maple. Our implementation uses internal functions of the RAGLIB Maple package [41]. They are based on FGB [12] (implemented in C by J.-C Faugère) for the computations of Gröbner bases. They also use RS [36] (implemented in C by F. Rouillier) for finding the real solutions of zero-dimensional systems and the isolation of real roots of univariate polynomials with rational coefficients. The computations have been done on an Intel(R) Xeon(TM) CPU 3.20GHz (2048 KB of cache) with 6 GB of RAM.

The polynomials $\sum_{i=1}^n \prod_{j \neq i} (X_i - X_j)$ which are called in the sequel **LL_n** are used as a benchmark for numerical methods based on LMI-relaxations (see [45]). In the sequel we consider **LL5** (which has degree 4 and contains 4 variables), **LL7** (which has degree 6 and contains 6 variables) and **LL9** (which has degree 8 and contains 8 variables).

The following polynomial (denoted by **Vor1**) appears in [11]. The initial question was to decide if its discriminant (denoted by **Vor2**) with respect to the variable u is always positive. Answering to this question can be done by computing its infimum. The polynomial **Vor2** has 253 monomials and is of degree 18.

$$16a^2(a^2 + 1 + \beta^2)u^4 + 16a(-\alpha\beta a^2 + a\alpha\alpha + 2a\alpha^2 + 2a + 2a\beta^2 + a\gamma\beta - \alpha\beta)u^3 + ((24a^2 + 4a^4)\alpha^2 + (-24\beta a^3 - 8\gamma a^3 + 24\alpha a^2 - 8a\gamma)\alpha + 24a^2\beta^2 + 4\beta^2 - 8\beta\alpha a^3 + 4\gamma^2 a^2 - 24\gamma\beta a^2 - 8\alpha\beta\beta + 16a^2 + 4x^2 a^2)u^2 + (-4\alpha a^3 + 4\gamma a^2 - 4a\alpha - 8\alpha\alpha + 8\beta a^2 + 4\beta)(\beta - \alpha\alpha + \gamma - a\alpha)u + (a^2 + 1)(\beta - \alpha\alpha + \gamma - a\alpha)^2$$

The following polynomial appears in [10]. In this paper, one has to decide if this polynomial is always positive. We compute its infimum to answer this question.

$$1 - 8c^2d^2 - 196608e^3a^2d^2c + 1536eada^4c^2 + 21504e^2ad^2c - 4096e^2ac^3d^2 - 384eada^2 + 1024e^2ac + 16c^4d^4 - 72c^2d^4 + 1024c^2e^2 + 36864e^2a^2d^4 - 3456eada^4 + 262144e^4a^2c^2 - 32768e^3ac^2 + 256c^3d^2e - 576ced^2 + 81d^4 + 64ce - 18d^2$$

The following polynomial appears in [29].

$$4a^6c^2d^2 + 2a^2b^2d^6 - 6a^2b^2d^4 + a^4c^4 + 2a^4c^2d^6 - 6a^2b^2c^4 - 6a^4b^2c^4 + 4a^6b^2d^2 + a^8b^4 + 6b^4c^2d^2 - 2b^6c^4d^2 + a^8d^4 + 6a^2b^6d^2 - 8a^4b^4d^2 - 4a^4b^2d^6 - 6b^4c^4d^2 - 8a^4b^4c^2 + 6a^6b^2c^2 - 8a^2b^4c^2 + 6a^4b^4d^4 - 2b^4c^2d^4 - 4a^2b^4c^6 - 4a^6b^4c^2 - 6a^2b^4d^4 - 2a^4c^4d^2 + 10a^4b^2d^4 - 2a^2b^8c^2 - 6a^2b^6c^4 + a^4b^8 + 6a^2b^2d^2 + 6a^6b^4d^2 - 4a^4b^6d^2 + b^4d^4 + b^4c^8 + 10a^2b^4c^4 + 6a^2b^2c^2 + 4a^2b^6c^2 + a^4d^8 + 4b^6c^2d^2 + 6a^4b^6c^2 - 8a^4b^2d^2 + 4a^4b^2c^2 - 2a^8b^2d^2 + 6a^4c^2d^2 + 4a^2b^4d^2 - 6a^6b^2d^4 + 6a^4b^4c^4 - 2a^6c^2d^4 + 2b^4c^6d^2 + 2a^2b^2c^6 - 6a^4c^2d^4 + b^8c^4 + 2a^4b^2 - 4a^4d^2 + a^4 - 2b^6 - 2a^6 + a^8 + b^8 + b^4 + 2a^2b^4 + 2b^6c^6 - 2b^8c^2 - 6b^6c^4 + 2a^2b^4 - 2a^2b^2 - 2a^6b^6 + 2a^4b^6 - 2a^2b^8 - 6a^4b^2c^4d^2 + 2a^2b^4c^4d^2 + 2a^4b^2c^2d^4 - 6a^4b^2c^2d^4 - 6a^4b^2c^2d^2 - 6a^2b^4c^2d^2 + 4a^2b^2c^4d^2 + 2a^2b^2c^2d^6 - 2a^8b^2 - 2a^8b^2 + 2a^6b^2 + 6a^6b^2c^2d^2 - 10a^4b^4c^2d^2 - 4b^4c^6 + 6b^4c^4 + 6b^6c^2 - 2a^6c^2 + 2a^2b^2c^6d^2 + a^4c^4d^4 - 2a^4c^2 - 2b^6d^2 - 4a^4d^6 + 2a^6d^6 - 2a^8d^2 - 6a^6d^4 + 6a^6d^2 + b^4c^4d^4 - 4b^4c^2 + 6a^4d^4 - 2b^4d^2(a^2 - b^2)$$

In the sequel, we give the timings we obtained with an implementation of the algorithm we describe in this paper in the column **Algo**. Timings obtained by the numeric SOS Solver SOSTOOLS [34] are given in the column **SOS**. The

column **CAD** contains the timings we obtained to solve the quantifier elimination problem induced by the global optimization problem by using the QEPCAD software [8]. The symbol ∞ below means that no result has been obtained after 1 week of computations.

	Algo	SOS	CAD
LL5	67 sec.	1 sec.	∞
LL7	10 hours	12 sec.	∞
LL9	∞	∞	∞
Vor1	40 sec.	53 sec.	∞
Vor2	2 hours	∞	∞
IT	10 sec.	2 sec.	5 sec.

Comparatively to **CAD**, the algorithm we describe in this paper is clearly more efficient since it can tackle problems that are not reachable by QEPCAD. We observe that *on most of these examples* numerical methods are faster. Nevertheless, *on these examples*, numerical methods have not allowed us to tackle problems that are not reachable by our method. Note also that they can't solve **Vor2**. Nevertheless, we believe that these techniques may return on answer on problems that are not tractable by our method. Finally, we observed numerical instability or wrong results for the problems **LL5**, **LL7** and **Vor1**. This shows that obtaining global optimization solvers based on computer algebra techniques is important and can be complementary to numerical methods.

7. REFERENCES

- [1] P. Aubry, F. Rouillier, and M. Safey El Din. Real solving for positive dimensional systems. *Journal of Symbolic Computation*, 34(6):543–560, 2002.
- [2] B. Bank, M. Giusti, J. Heintz, and G.-M. Mbakop. Polar varieties and efficient real equation solving: the hypersurface case. *Journal of Complexity*, 13(1):5–27, 1997.
- [3] B. Bank, M. Giusti, J. Heintz, and G.-M. Mbakop. Polar varieties and efficient real elimination. *Mathematische Zeitschrift*, 238(1):115–144, 2001.
- [4] B. Bank, M. Giusti, J. Heintz, and L.-M. Pardo. Generalized polar varieties and efficient real elimination procedure. *Kybernetika*, 40(5):519–550, 2004.
- [5] B. Bank, M. Giusti, J. Heintz, and L.-M. Pardo. Generalized polar varieties: Geometry and algorithms. *Journal of complexity*, 2005.
- [6] S. Basu, R. Pollack, and M.-F. Roy. On the combinatorial and algebraic complexity of quantifier elimination. *Journal of ACM*, 43(6):1002–1045, 1996.
- [7] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in real algebraic geometry*. Springer-Verlag, 2003.
- [8] C. Brown and H. et al. Hong. Qepcad - quantifier elimination by partial cylindrical algebraic decomposition. available at <http://www.cs.usna.edu/qepcad/B/QEPCAD.html>.
- [9] G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. *Lecture notes in computer science*, 33:515–532, 1975.
- [10] I. Emiris and E. Tsigaridas. Real algebraic numbers and polynomial systems of small degree. submitted to *Journal of Symbolic Computation*, 2008.
- [11] H. Everett, D. Lazard, S. Lazard, and M. Safey El Din. The voronoi diagram of three lines in r^3 . In

SoCG '07: Proceedings of the 23-rd annual symposium on computational geometry, pages 255–264, 6 2007.

- [12] J.-C. Faugère. Gb/FGb. available at <http://fgbrs.lip6.fr>.
- [13] R.M. Freund, F. Odonez, and K.C. Toh. Behavioral measures and their correlation with ipm iteration counts on semidefinite programming problems. *Mathematical programming*, 109(2-3):445–475, 2007.
- [14] M. Giusti, K. Hägele, J. Heintz, J.-E. Morais, J.-L. Montaña, and L.-M. Pardo. Lower bounds for Diophantine approximation. In *Proceedings of MEGA '96*, number 117, 118 in *Journal of Pure and Applied Algebra*, pages 277–317, 1997.
- [15] M. Giusti, J. Heintz, J.-E. Morais, J. Morgenstern, and L.-M. Pardo. Straight-line programs in geometric elimination theory. *Journal of Pure and Applied Algebra*, 124:101–146, 1998.
- [16] M. Giusti, J. Heintz, J.-E. Morais, and L.-M. Pardo. When polynomial equation systems can be solved fast? In *Proceedings of AAECC-11*, volume 948 of *LNCS*, pages 205–231. Springer, 1995.
- [17] M. Giusti, G. Lecerf, and B. Salvy. A Gröbner free alternative for polynomial system solving. *Journal of Complexity*, 17(1):154–211, 2001.
- [18] D. Grigoriev and N. Vorobjov. Solving systems of polynomials inequalities in subexponential time. *Journal of Symbolic Computation*, 5:37–64, 1988.
- [19] J. Heintz, M.-F. Roy, and P. Solernò. On the complexity of semi-algebraic sets. In *Proceedings IFIP'89 San Francisco, North-Holland*, 1989.
- [20] J. Heintz, M.-F. Roy, and P. Solernò. On the theoretical and practical complexity of the existential theory of the reals. *The Computer Journal*, 36(5):427–431, 1993.
- [21] D. Henrion. *Polynômes et optimisation convexe en commande robuste*. PhD thesis, LAAS, Toulouse, 2007. Habilitation à diriger des recherches.
- [22] Z. Jelonek and K. Kurdyka. On asymptotic critical values of a complex polynomial. *J. Reine Angew. Math.*, 565:1–11, 2003.
- [23] K. Kurdyka, P. Orro, and S. Simon. Semialgebraic sard theorem for generalized critical value. *Journal of differential geometry*, 56(1):67–92, 2000.
- [24] Y. N. Lakshmann. A single exponential bound of the complexity of computing Gröbner bases of zero-dimensional ideals. In C. Traverso T. Mora, editor, *Proc. Effective Methods in Algebraic Geometry, MEGA '90*, volume 94 of *Progress in Mathematics*, pages 227–234. Birkhäuser, 1991.
- [25] Y.N. Lakshmann and D. Lazard. On the complexity of zero-dimensional algebraic systems. In *MEGA*, volume 94 of *Progress in Mathematics*, pages 217–225. Birkhäuser, 1991.
- [26] J.-B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.
- [27] J. B. Lasserre, M. Laurent, and P. Rostalski. Semidefinite characterization and computation of real radical ideals, 2006.
- [28] D. Lazard and F. Rouillier. Solving parametric polynomial systems. *J. Symb. Comput.*, 42:636–667, 2007.
- [29] Daniel Lazard. Quantifier elimination: optimal solution for two classical examples. *J. Symb. Comput.*, 5(1-2):261–266, 1988.
- [30] G. Lecerf. Computing the equidimensional decomposition of an algebraic closed set by means of lifting fibers. *Journal of Complexity*, 19(4):564–596, 2003.
- [31] Jiawang Nie, James Demmel, and Bernd Sturmfels. Minimizing polynomials via sum of squares over the gradient ideal. *Math. Program.*, 106(3):587–606, 2006.
- [32] P. A. Parrilo. Semi-definite relaxations for semi-algebraic problems. *Mathematical Programming*, 92(2):293–320, 2003.
- [33] P. Parrilo and B. Sturmfels. Minimizing polynomial functions. *Dimacs Series in Discrete Mathematics and Theoretical Computer Science*, 60:83, 2003.
- [34] S. Prajna, A. Papachristodoulou, P. Parrilo, and P. Seiler. Sostools. available at <http://www.cds.caltech.edu/sostools/>.
- [35] J. Renegar. On the computational complexity and geometry of the first order theory of the reals. *Journal of Symbolic Computation*, 13(3):255–352, 1992.
- [36] F. Rouillier. RS, RealSolving. available at <http://fgbrs.lip6.fr>.
- [37] F. Rouillier. Solving zero-dimensional systems through the Rational Univariate Representation. *AAECC Journal*, 9(5):433–461, 1999.
- [38] F. Rouillier, M.-F. Roy, and M. Safey El Din. Finding at least one point in each connected component of a real algebraic set defined by a single equation. *Journal of Complexity*, 16:716–750, 2000.
- [39] M. Safey El Din. Finding sampling points on real hypersurfaces in easier in singular situations. In *MEGA (Effective Methods in Algebraic Geometry) Electronic proceedings*, 2005.
- [40] M. Safey El Din. Practical and theoretical issues for the computation of generalized critical values of a polynomial mapping and its applications. In *Proceedings of Asian Symposium on Computer Mathematics 2007*, 2007. to appear.
- [41] M. Safey El Din. RAGLib (Real Algebraic Geometry Library), Maple package. available at <http://www-spiral.lip6.fr/~safey/RAGLib>, 2007.
- [42] M. Safey El Din. Testing sign conditions on a multivariate polynomial and applications. *Mathematics in Computer Science*, 1(1):177–207, December 2007.
- [43] M. Safey El Din and É. Schost. Polar varieties and computation of one point in each connected component of a smooth real algebraic set. In *Proceedings of ISSAC 2003*, pages 224–231. ACM Press, 2003.
- [44] M. Safey El Din and É. Schost. Properness defects of projections and computation of one point in each connected component of a real algebraic set. *Journal of Discrete and Computational Geometry*, 2004.
- [45] M. Schweighofer. Global optimization of polynomials using gradient tentacles and sums of squares. *SIAM Journal on Optimization*, 17(3):920–942, 2006.

Classification of the Perspective-Three-Point Problem, Discriminant Variety and Real Solving Polynomial Systems of Inequalities

Jean-Charles Faugère, Guillaume Moroz, Fabrice Rouillier, and Mohab Safey El Din
INRIA, Paris-Rocquencourt Center, SALSA Project
UPMC, Univ Paris 06, LIP6
CNRS, UMR 7606, LIP6
UFR Ingénierie 919, LIP6 Passy-Kennedy
Case 169, 4, Place Jussieu, F-75252 Paris
{Jean-Charles.Faugere,Guillaume.Moroz,Fabrice.Rouillier,Mohab.Safey}@lip6.fr

ABSTRACT

Classifying the Perspective-Three-Point problem (abbreviated by P3P in the sequel) consists in determining the number of possible positions of a camera with respect to the apparent position of three points. In the case where the three points form an isosceles triangle, we give a full classification of the P3P. This leads to consider a polynomial system of polynomial equations and inequalities with 4 parameters which is generically zero-dimensional. In the present situation, the parameters represent the apparent position of the three points so that solving the problem means determining all the possible numbers of real solutions with respect to the parameters' values and give a sample point for each of these possible numbers. One way for solving such systems consists first in computing a *discriminant variety*. Then, one has to compute at least one point in each connected component of its real complementary in the parameter's space. The last step consists in specializing the parameters appearing in the initial system by these sample points. Many computational tools may be used for implementing such a general method, starting with the well known Cylindrical Algebraic Decomposition (CAD in short), which provides more information than required. In a first stage, we propose a full algorithm based on the straightforward use of some sophisticated software such as FGb (Gröbner bases computations) RS (real roots of zero-dimensional systems), DV (Discriminant varieties) and RAGlib (Critical point methods for semi-algebraic systems). We then improve the global algorithm by refining the required computable mathematical objects and related algorithms and finally provide the classification. Three full days of computation were necessary to get this classification which is obtained from more than 40000 points in the parameter's space.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Categories and Subject Descriptors

G.4 [Mathematics of computing]: Mathematical Software; F.2.2 [Theory of Computation]: Analysis of algorithms and problem complexity—*Non numerical algorithms and problems: Geometrical problems and computation*; I.4.1 [Computing Methodologies]: Image Processing and Computer Vision—*Digitization and Image Capture*

General Terms

Algorithms

Keywords

Polynomial system solving, Perspective-Three-Point problem, real solutions, complexity, computer vision

1. INTRODUCTION

The perspective- n -points problem has been widely studied during the past decades [7, 13, 30, 9]. The goal is to determine the position of a camera given the apparent position of n points. This problem has many applications in fields such as computer vision [16] or automated cartography [7] for example. It was summarized in [7] for the general case of n points as follows:

“Given the relative spatial location of n control points, and given the angle to every pair of control points from an additional point called the Center of Perspective (CP), find the lengths of the line segments joining CP to each of the control points.”

It was proved in 1984 [8] that for $n \geq 6$, the position of the Center of Perspective is uniquely determined by the angles from CP to the pairs of control points. Different algorithms were designed for the direct computation. Harlick et al. reviewed the major direct solutions before 1991 in [13]. More recently, this direct problem has been revisited in [21], [1], [22] and references therein.

However, in [7] Fischler and Bolles observed that the P3P problem could have from 0 up to 4 solutions, according to the triangle lengths and the angles observed from the perspective point of view. Moreover, in [15] and references therein, Z.Y. Wu and F.C. Hu show that the P5P problem can have 2 solutions, and the P4P problem up to 5 solutions. But for

each number of solutions, the corresponding set of parameters was not given.

In [10], X.-S. Gao and J. Tang have finally proved that for $n \geq 4$, the set of parameters where the PnP problem admits more than one solution has a zero volume.

In [30] and [9] Yang, Gao and al. exhibit respectively a partial and an exhaustive set of polynomial conditions to decide the number of solutions of the P3P problem. These conditions were obtained by combining a triangular decomposition, resultant computations and a careful use of the Descartes' rule of sign and of the Sylvester-Habicht sequences. However, these conditions are rather complex and do not give usable informations on the geometry of these cells. In particular, testing the satisfiability of such conditions is challenging since it is not feasible with current CAD and generic CAD software.

Recently in [31], [29], the authors give some geometrical condition for the P3P problem to have 4 solutions, and provide a guide to arrange control points in real applications.

In this article, we present an efficient and certified method to obtain a more intuitive classification of the parameters of a system with respect to its number of solutions. We then apply this method to the P3P problem in the case where the three points form an isosceles triangle. Our output will be a so-called *open classification*. In particular, *for each possible number of solutions of the perspective three point problem we provide at least one point in each component of the corresponding set of parameters*. Our classification is said open because we ensure that the components we compute have a non-null volume and are thus reachable in practice. In particular we omit components of null volume, since they have no physical interest.

Main result and related work. The complete resolution of the P3P problem is given for an isosceles triangle. It required 3 days of computations. The number of computed points is 60086.

A first key tool which allowed us to solve this problem is the *discriminant variety*. This notion is related to the implicit function theorem and has many variants, as those presented in [19, 11] for example. We use the definition of [17], where a Discriminant Variety of a parametric system with rational or real coefficients is an algebraic variety of the parameter's space such that, among other remarkable properties, the number of real solutions of the initial system is invariant on each connected component of its complementary over the reals. For its computation, we used the algorithm provided in [17] and implemented in the DV package for the so-called *well-behaved systems*.

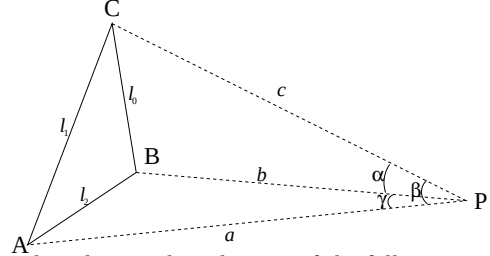
CAD implementations could not compute at least one point in each connected component of the complementary of the discriminant variety over the reals, due to the prohibitive number of cells (and timing) induced by the doubly exponential behaviour of the algorithms. Thus we developed new algorithms to tackle this problem. These algorithms are based on the critical point method. These methods are known to have a better worst-case complexity than CAD algorithm (see [3, Chapter 13] and references therein). Nevertheless, up to now, they used infinitesimals [2, 12, 14] which makes heavier the arithmetic with which computations are performed and, finally, spoil the practical behavior of this method. The first algorithm we developed, generalizes the strategy developed in [26] and computes *a priori* specialization values for the infinitesimal. This algorithm allowed

us to solve the problem but required 3 weeks of computations. Then, we developed a second algorithm whose practical behavior is much better, since this one allows us to solve completely the problem in 3 days.

In the first two parts, we will recall the P3P problem and present our general classification strategy. Then we will detail the two main algorithms of our method. Finally, we will show its application on the P3P problem.

2. DESCRIPTION OF THE PERSPECTIVE THREE POINT PROBLEM

Let A, B and C be the three control points, P be the perspective point and α, β, γ be respectively the three angles $\widehat{BPC}, \widehat{APC}$ and $\widehat{APB}$. Furthermore, let $a = PA$, $b = PB$, $c = PC$, $l_0 = AB$, $l_1 = BC$ and $l_2 = AC$.



The lengths a, b, c are the solutions of the following equations given in [7]:

$$\begin{cases} l_0^2 &= a^2 + b^2 - 2ab \cos(\alpha) \\ l_1^2 &= b^2 + c^2 - 2bc \cos(\beta) \\ l_2^2 &= a^2 + c^2 - 2ac \cos(\gamma) \end{cases}$$

We denote by u, v and w the expressions $2 \cos(\alpha)$, $2 \cos(\beta)$ and $2 \cos(\gamma)$. Moreover, as in [9], we express all the lengths of our problem relatively to l_0 . Thus we introduce $A = \frac{a}{l_0}$, $B = \frac{b}{l_0}$, $C = \frac{c}{l_0}$. And for the length of the triangle, we use $x = \frac{l_2^2}{l_0^2}$ and $t = \frac{l_1^2}{l_0^2}$. Thus we get the following simplified system:

$$\begin{cases} 1 &= A^2 + B^2 - ABu \\ t &= B^2 + C^2 - BCv \\ x &= A^2 + C^2 - ACw \end{cases}$$

with the following constraints:

$$x > 0, t > 0, -2 < u < 2, -2 < v < 2, -2 < w < 2$$

where:

- A, B, C are the *unknowns*
- x, t, u, v, w are the *parameters*

We will present a general method to classify the parameters of such a system. Given a number k , this method allows us to say if there exists an open set of the parameters where the system admit exactly k solutions.

We will show the application of this method for the classification of the parameters of the P3P problem, in the case where the triangle is isosceles.

3. CLASSIFICATION METHOD - DISCRIMINANT VARIETY

Goal. Let $S_{\mathbf{T}}(\mathbf{X})$ be a parametric system of polynomial equalities and inequalities in $\mathbb{Q}[\mathbf{T}][\mathbf{X}]$, where $\mathbf{T} = T_1, \dots, T_s$

are the parameters and $\mathbf{X} = X_1, \dots, X_n$ the unknowns. We want to be able to answer to the following question:
“Given a parametric system $S_{\mathbf{T}}$ and an integer i , does there exist an open set $\mathcal{O}$ in the parameters’ space such that for all $p_0 \in \mathcal{O}$, the number of solutions of S_{p_0} is i ? If yes, give explicitly a point $a \in \mathcal{O}$ ”.

For this purpose, we present a method to classify the parametric values p_0 of a dense open set of $\mathcal{P}$ according to the number of real solutions of S_{p_0} . In the following, $\mathcal{P}$ will denote the real parameters’ space. The method we describe in this article computes exactly an *open classification* of $\mathcal{P}$ with relation to $S_{\mathbf{T}}$ according to the following definition:

DEFINITION 1. (Open classification) *Let $S_{\mathbf{T}}(\mathbf{X})$ be a parametric system. Let $k \in \mathbb{N}$ and $\mathcal{O}_0, \dots, \mathcal{O}_k$ be open sets (for the euclidean topology) in the parameters’ space such that:*

$$\begin{cases} \forall p_0 \in \mathcal{O}_i, S_{p_0} \text{ has } i \text{ real solutions} \\ \bigcup_{i=0}^k \mathcal{O}_i \text{ is dense in the parameters’ space} \end{cases}$$

We call the family $(\mathcal{O}_i)_{0 \leq i \leq k}$ an *open classification* of $\mathcal{P}$ with relation to $S_{\mathbf{T}}$.

As announced in the introduction, the proposed methods will be based on the discriminant variety introduced in [17].

DEFINITION 2. (Discriminant variety) *Given a constructible set $\mathcal{C}$, a discriminant variety of $\mathcal{C}$ is an algebraic set in the parameter’s space such that a restriction of the trivial projection from $\mathcal{C}$ onto the complementary of the discriminant variety in the parameters’ space defines an analytic cover.*

In addition, a discriminant variety is the parameters’ space itself if and only if each of the (complex) fibers is infinite.

DEFINITION 3. (Minimal discriminant variety) *The minimal discriminant variety is the intersection of all the discriminant varieties (and is thus a discriminant variety).*

REMARK 1. *In particular, the complementary of a discriminant variety defines an open classification of $\mathcal{P}$ with relation to $S_{\mathbf{T}}$.*

Computing an open classification. Given a parametric system $S_{\mathbf{T}}$, we show that an *open classification* of $S_{\mathbf{T}}$ can be represented by (q, F, ϕ) , which are defined as follows:

- q is a polynomial and a discriminant variety of $S_{\mathbf{T}}$;
- F a set of rational points in each connected component of $q \neq 0$;
- ϕ is a table which associates to each point p_0 of F the number of solutions of the 0-dimensional systems S_{p_0} .

In this representation, each $\mathcal{O}_i$ is represented by q and the subset of points $\phi^{-1}(i) \subset F$ such that:

$$\mathcal{O}_i = \{ x \in \mathcal{P} \mid \text{there exists } p \in \phi^{-1}(i) \text{ and a continuous path from } p \text{ to } x \text{ included in } q \neq 0 \}$$

To compute this representation, our algorithm is naturally decomposed in three steps:

Input: a parametric system $S_{\mathbf{T}}$, the set of parameters $\mathbf{T}$, and the set of unknowns $\mathbf{X}$.

Output: the 3-tuple (q, F, ϕ)

Main algorithm:

Step a: The discriminant variety q . For the first step, we compute q as a polynomial vanishing at the discriminant variety of $S_{\mathbf{T}}$. The full algorithm may be found in [17] and the main ideas of its computation are recalled in the appendix. It is implemented in the maple DVLIB package and will be directly available in the next release of Maple [18].

Step b: The sampling points F . The critical point method allows to compute at least one point in each connected component of a semi-algebraic set defined by strict inequalities. An algorithm using these methods is given in [26]. We show in section 4 an improvement of this algorithm. In this step, F is a finite set of point in each connected component of the semi-algebraic set defined by $q \neq 0$. This function is implemented in the maple RAGLIB package.

Step c: The table ϕ Finally, we compute a table where each point p_0 of F is associated to the number of real solutions of the system S_{p_0} . For this step, we use the Rational Univariate Representation presented in [23] and implemented in the RS software which gives a list of non overlapping boxes with rational bounds, containing the real solutions of a zero-dimensional system. $\square$

Theoretically, the first step has the largest complexity upper bound. However, in practice the behavior of the three steps does not follow the same scheme. In particular, the first step is not often slower than the other steps.

4. SOLVING SYSTEMS OF POLYNOMIAL INEQUALITIES

As described above, once a discriminant variety V is computed, one has to compute at least one point in each connected component of $\mathbb{R}^n \setminus V$. RAGLIB provides routines allowing us to tackle this computation. This section is devoted to present the algorithms we implemented in RAGLIB and give some sketch of proofs. The techniques we use are based on computations of critical points or critical values of polynomial mappings. In the whole section, we consider a polynomial family $(f_1, \dots, f_s)$ in $\mathbb{Q}[X_1, \dots, X_n]$ of degree bounded by D . We denote by $\mathcal{S}$ the semi-algebraic set defined by $f_1 > 0, \dots, f_s > 0$ which is supposed to be bounded. Denote by $\mathbb{Q}_+$ the set of positive rationals. Given $\mathbf{a} = (a_1, \dots, a_s) \in \mathbb{Q}_+^s$, $\mathcal{I} = \{i_1, \dots, i_\ell\} \subset \{1, \dots, s\}$, and $e \in \mathbb{R}$ (resp. E a new variable) we denote by $S_{\mathcal{I},e}^{(\mathbf{a})}$ (resp. $S_{\mathcal{I},E}^{(\mathbf{a})}$) the polynomial system $f_{i_1} - a_{i_1}e = \dots = f_{i_\ell} - a_{i_\ell}e = 0$ (resp. $f_{i_1} - a_{i_1}E = \dots = f_{i_\ell} - a_{i_\ell}E = 0$) and by $V_{\mathcal{I},e}^{(\mathbf{a})} \subset \mathbb{C}^n$ (resp. $V_{\mathcal{I},E}^{(\mathbf{a})} \subset \mathbb{C}^{n+1}$) the algebraic variety it defines. In the sequel, we use the following notations:

- Π denotes the canonical projection $(x_1, \dots, x_n, e) \in \mathbb{C}^{n+1} \rightarrow (x_1, \dots, x_n) \in \mathbb{C}^n$;
- The canonical projection $(x_1, \dots, x_n) \in \mathbb{C}^n \rightarrow x_i \in \mathbb{C}$ (resp. $(x_1, \dots, x_n, e) \in \mathbb{C}^{n+1} \rightarrow e \in \mathbb{C}$) is denoted by π_i (resp. π_E).
- Given a polynomial mapping $\varphi : \mathbb{C}^n \rightarrow \mathbb{C}^k$ and an algebraic variety $V \subset \mathbb{C}^n$, $\mathfrak{C}(\varphi, V)$ denotes the critical locus of the restriction of φ to V ;
- Given a polynomial mapping $\varphi : \mathbb{C}^n \rightarrow \mathbb{C}^k$ and an algebraic variety $V \subset \mathbb{C}^n$, $\mathcal{K}(\varphi, V)$ denotes the critical values of the restriction of φ to V ;
- Given $\mathcal{I}$ and $\mathbf{a}$, $\mathfrak{C}_{\mathcal{I}}^{\mathbf{a}}$ denotes the Zariski-closure of

$$\bigcup_{e \in \mathbb{C} \setminus \mathcal{K}(\pi_1, V_{\mathcal{I},E}^{(\mathbf{a})})} \{(x, e) \in \mathbb{C}^{n+1} \mid x \in \mathfrak{C}(\pi_1, V_{\mathcal{I},E}^{(\mathbf{a})})\}$$

- Given $\mathcal{I}$ and $\mathbf{a}$, $e_{\mathcal{I}}^{\mathbf{a}}$ denotes the smallest positive value of $\mathcal{K}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})}) \cup_{j \in \{1, \dots, s\} \setminus \mathcal{I}} \pi_E(\mathfrak{C}_{\mathcal{I}}^{\mathbf{a}} \cap \{x \in \mathbb{R}^n \mid f_j(x) = 0\})$.

The following result shows that under some assumptions of genericity on $\mathbf{a}$ and e , $V_{\mathcal{I},e}^{(\mathbf{a})}$ is smooth and equi-dimensional, and the ideal generated by $S_{\mathcal{I},e}^{(\mathbf{a})}$ is radical. These properties are useful to compute critical points of polynomial mappings restricted to $V_{\mathcal{I},e}^{(\mathbf{a})}$ by intersecting $V_{\mathcal{I},e}^{(\mathbf{a})}$ with the vanishing set of some minors of the Jacobian matrix associated to $S_{\mathcal{I},e}^{(\mathbf{a})}$.

LEMMA 1. *Given $\mathcal{I} \subset \{1, \dots, s\}$, there exists a proper Zariski-closed subset $\mathcal{A} \times \mathcal{E} \subsetneq \mathbb{C}^s \times \mathbb{C}$ such that for all $\mathbf{a} \in \mathbb{Q}^s \setminus \mathcal{A}$ and $e \in \mathbb{Q} \setminus \mathcal{E}$, $S_{\mathcal{I},e}^{(\mathbf{a})}$ generates a radical and equi-dimensional ideal and $V_{\mathcal{I},e}^{(\mathbf{a})}$ is smooth of dimension $n - \#\mathcal{I}$ or empty.*

PROOF. Consider the polynomial mapping $\varphi : x \in \mathbb{C}^n \rightarrow (f_{i_1}(x), \dots, f_{i_\ell}(x)) \in \mathbb{C}^\ell$. From the algebraic Sard's theorem [3], the set of critical values of φ is contained in a proper Zariski-closed subset $\mathcal{A} \subsetneq \mathbb{C}^\ell$. This implies that for all $(a_{i_1}, \dots, a_{i_\ell}) \in \mathbb{Q}^\ell \setminus \mathcal{A}$, the system $f_{i_1} - a_{i_1} = \dots = f_{i_\ell} - a_{i_\ell} = 0$ generates a radical and equidimensional ideal and defines a smooth variety of dimension $n - \ell$. Considering now the restriction of the mapping $(x, e) \in \mathbb{C}^n \times \mathbb{C} \rightarrow e$ to the variety defined by $f_{i_1} - a_{i_1}e = \dots = f_{i_\ell} - a_{i_\ell}e = 0$ and using as previously, Sard's theorem ends the proof. $\square$

First algorithm. This paragraph contains a description of a first algorithm which generalizes to the case of polynomial systems of inequalities the strategy developed in [26]. From the proposition below, one can reduce the computation of at least one point in each connected component of $\mathcal{S}$ to the computation of at least one point in each connected component of real algebraic sets $V_{\mathcal{I},e}^{(\mathbf{a})} \cap \mathbb{R}^n$ if e is small enough.

PROPOSITION 1. *Let C be a connected component of $\mathcal{S}$. There exists $\mathcal{I} \subset \{1, \dots, s\}$ and $e_0 > 0$ such that for all $e \in]0, e_0[$, there exists a connected component of $V_{\mathcal{I},e}^{(\mathbf{a})} \cap \mathbb{R}^n$ contained in C .*

PROOF. The result is an immediate application of the transfer principle and [3, Chapter 13, Proposition 13.2], remarking that $\mathcal{S}$ is also defined by $\frac{f_1}{a_1} > 0, \dots, \frac{f_s}{a_s} > 0$. $\square$
Given $\mathbf{a} \in \mathbb{Q}^s$, a connected component C of $\mathcal{S}$, and $\mathcal{I}$ such that the first item of Proposition 1 above applies, we show below how to compute e_0 .

THEOREM 1. *Let C be a connected component of $\mathcal{S}$. There exists $\mathcal{I} \subset \{1, \dots, s\}$ such that for all $e \in]0, e_{\mathcal{I}}^{\mathbf{a}}[$ there exists a point of $\mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$ lying in C ;*

PROOF. Let $\mathbf{a} = (a_1, \dots, a_s) \in \mathbb{Q}^s$. Consider a maximal (for the order inclusion) subset $\mathcal{I} = \{i_1, \dots, i_\ell\} \subset \{1, \dots, s\}$ such that Proposition 1 applies. Then, there exists $e_0 > 0$ such that for $e \in]0, e_0[$ there exists a compact connected component C_e of $V_{\mathcal{I},e}^{(\mathbf{a})} \cap \mathbb{R}^n$ included in C . Since C_e is compact, it contains a point x_e of $\mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})}) \subset \mathfrak{C}_{\mathcal{I}}^{\mathbf{a}}$. Denote by B a ball containing C . Since $e_{\mathcal{I}}^{\mathbf{a}}$ is less than or inferior to the minimum of the positive real numbers of $\mathcal{K}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})})$, the Ehresmann's fibration theorem implies that Π_E realizes a locally trivial fibration over $\{(x, e) \in B \times \mathbb{R}\} \cap V_{\mathcal{I},E}^{(\mathbf{a})} \cap \pi_E^{-1}(]0, e_{\mathcal{I}}^{\mathbf{a}}[)$.

This implies that C_e varies continuously and remains non-empty and compact when e varies in $]0, e_{\mathcal{I}}^{\mathbf{a}}[$ and then has a non-empty intersection with $\mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$. Note at last that by definition of $e_{\mathcal{I}}^{\mathbf{a}}$, for all $j \in \{1, \dots, s\} \setminus \mathcal{I}$, f_j is positive at each point of this intersection. $\square$

Given $\mathbf{A} \in GL_n(\mathbb{Q})$ and $f \in \mathbb{Q}[X_1, \dots, X_n]$, we denote by $f^{\mathbf{A}}$ the polynomial $f(\mathbf{A}\mathbf{X})$ where $\mathbf{X} = [X_1, \dots, X_n]$. If $V \subset \mathbb{C}^n$ is an algebraic variety defined by $f_1 = \dots = f_s = 0$ (with $\{f_1, \dots, f_s\} \subset \mathbb{Q}[X_1, \dots, X_n]$, $V^{\mathbf{A}}$ denotes the variety defined by $f_1^{\mathbf{A}} = \dots = f_s^{\mathbf{A}} = 0$.

PROPOSITION 2. *Let $e \in \mathbb{C} \setminus \mathcal{K}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})})$. There exists a Zariski-closed subset $\mathcal{A} \subsetneq GL_n(\mathbb{C})$ such that for all $\mathbf{A} \in GL_n(\mathbb{Q}) \setminus \mathcal{A}$, $\mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})\mathbf{A}})$ is either empty or zero-dimensional.*

PROOF. Since $e \notin \mathcal{K}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})})$, $V_{\mathcal{I},e}^{(\mathbf{a})}$ is smooth. Now, the result is an immediate consequence of [27, Theorem 2]. $\square$
Finally, the algorithm consists in considering all the polynomial systems $S_{\mathcal{I},e}^{(\mathbf{a})}$, compute a univariate polynomial whose set of roots is $\mathcal{K}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})})$, find a rational number $e > 0$ less than the smallest positive root of this polynomial and compute a rational parameterization of $\mathcal{K}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$ up to a generic enough linear change of variables. All these computations are done using algebraic elimination routines. In practice RAGLIB uses Gröbner bases FGB engine [5, 6].

REMARK 2. *In order to generalize this strategy to the general case of non-bounded semi-algebraic sets, it is sufficient to compute the set of generalized critical values of π_E restricted to $V_{\mathcal{I},E}^{(\mathbf{a})}$ (see [26, 25] for algorithms performing such computations), find a rational number $e > 0$ less than the smallest generalized critical value, and to compute at least one point in each connected component of $V_{\mathcal{I},e}^{(\mathbf{a})}$ (see [27] for an efficient algorithm performing such a task).*

Second algorithm. We will see in the next section that the results (in terms of computation timings) obtained by the above algorithm are not satisfactory even if it has allowed us to solve completely the classification problem we consider here. This paragraph is devoted to design an other algorithm, based on similar techniques than those developed above, which is more efficient. It avoids the computations of $\mathcal{K}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})})$ which are expensive, in particular when the critical locus $\mathfrak{C}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})})$ has not dimension 0. This is the case in the application we consider. Given a connected component C of $\mathcal{S}$, there exists $\mathcal{I} \subset \{1, \dots, s\}$ from Proposition 1 such that the real algebraic set $V_{\mathcal{I},e}^{(\mathbf{a})}$ has a connected component contained in C for e small enough. The idea we develop here is the following. Instead of computing a specialization value for e (see Theorem 1 above), we focus on the informations one can get when e tends to 0. From Proposition 2, there exists a Zariski-closed subset $\mathcal{E} \subsetneq \mathbb{C}$ such that for all $e \in \mathbb{R} \setminus \mathcal{E}$, up to a generic linear change of coordinates, $\mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$ is either empty or zero-dimensional. In the sequel, we denote by $\mathfrak{C}_{\mathcal{I}}^{\mathcal{E}}$ the Zariski-closure of $\cup_{e \in \mathbb{C} \setminus \mathcal{E}} \mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$. From Hardt's semi-algebraic triviality theorem, there exists $e_0 \in \mathbb{R}$ such that $\mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})}) \times]0, e_0[$ is homeomorphic $\cup_{e \in]0, e_0[} \mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$. Thus, one can define by continuity the set $\lim_0 \mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$ of finite limits of $\mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$ when e tends to 0. In particular, we look at $\lim_0 \mathfrak{C}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$ and $\lim_0 \mathcal{K}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})})$.

THEOREM 2. Consider C be a connected component of the bounded semi-algebraic set $\mathcal{S}$. There exists $\mathcal{I} \subset \{1, \dots, s\}$ such that

- a) given $\{\xi_1, \dots, \xi_k\} = \lim_0(\mathcal{K}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})}))$ (with $\xi_1 < \dots < \xi_k$),
 - b) and $r_1, \dots, r_{k-1}$ be rationals such that $\xi_1 < r_1 < \dots < r_{k-1} < \xi_k$,
- there exists $i \in \{1, \dots, k-1\}$ for which $C \cap \pi_1^{-1}(r_i) \neq \emptyset$.

PROOF. From Proposition 1, there exists $\mathcal{I} \subset \{1, \dots, s\}$ a maximal set (for the order inclusion) and $e_0 > 0$ such that for $e \in]0, e_0[$, $V_{\mathcal{I},e}^{(\mathbf{a})} \cap \mathbb{R}^n$ has a connected component C_e included in C .

Since C is bounded, C_e is compact. Thus, its image by the projection π_1 is a closed interval $[a_e, b_e]$. We prove now that when e tends to 0, a_e (resp. b_e) has a finite limit in $\mathbb{R}$ denoted by a_0 (resp. b_0). Since $\mathcal{S}$ is supposed to be bounded, there exists a ball $B \subset \mathbb{R}^n$ such that $C \subset B$. Suppose now that a_e has not a finite limit in $\mathbb{R}$ when e tends to 0. This implies that there exists e small enough such that C_e is not contained in B while it is still contained in C . This is a contradiction. Consider now $r \in]a_0, b_0[$. There obviously exists $e > 0$ such that $a_e < r < b_e$. Thus $C \cap \pi_1^{-1}(r) \neq \emptyset$. $\square$ Denote by $\text{Jac}(f_{i_1}, \dots, f_{i_\ell})$ the Jacobian matrix associated to $f_{i_1}, \dots, f_{i_\ell}$.

$$\begin{bmatrix} \frac{\partial f_{i_1}}{\partial X_1} & \dots & \frac{\partial f_{i_1}}{\partial X_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_{i_\ell}}{\partial X_1} & \dots & \frac{\partial f_{i_\ell}}{\partial X_n} \end{bmatrix}$$

Denote by $\Sigma_{\mathcal{I}}$ the set of all $(n-\ell, n-\ell)$ minors of $\text{Jac}(f_{i_1}, \dots, f_{i_\ell})$. We consider in the sequel $\Delta_{\mathcal{I}}$ the set of $(n-\ell, n-\ell)$ minors of the matrix obtained after removing the first column of $\text{Jac}(f_{i_1}, \dots, f_{i_\ell})$ and $\delta_{\mathcal{I}} = \Sigma_{\mathcal{I}} \setminus \Delta_{\mathcal{I}}$. In the sequel, $\delta_{\mathcal{I}}[i]$ denotes the i -th element of $\delta_{\mathcal{I}}$ and $\delta_{\mathcal{I}}^{(i)}$ denotes the first i elements of $\delta_{\mathcal{I}}$. Given $\mathcal{I} = \{i_1, \dots, i_\ell\} \subset \{1, \dots, s\}$ and $\mathbf{a} = (a_1, \dots, a_s) \in \mathbb{Q}^s$, we consider polynomial systems $\mathcal{S}_{\mathcal{I}}^{\mathbf{a}}$ $a_{i_1}f_{i_2} - a_{i_2}f_{i_1} = \dots = a_{i_1}f_{i_\ell} - a_{i_\ell}f_{i_1} = 0$ and the ideals defined by:

$$I_i = \langle \mathcal{S}_{\mathcal{I}}^{\mathbf{a}}, L\delta_{\mathcal{I}}[i+1] - 1, \delta_{\mathcal{I}}^{(i)}, \Delta_{\mathcal{I}} \rangle \cap \mathbb{Q}[X_1, \dots, X_n]$$

The following result shows how to compute $\lim_0(\mathcal{K}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})}))$ under assumptions of genericity on $\mathbf{a}$. Its proof is based on similar techniques than the ones used in [24].

THEOREM 3. There exists a Zariski-closed subset $\mathcal{A} \times \mathcal{E} \subset \mathbb{C}^n \times \mathbb{C}$ such that for all $\mathbf{a} \in \mathbb{Q}^s \setminus \mathcal{A}$, $e \in \mathbb{R} \setminus \mathcal{E}$ and for all $\mathcal{I} \subset \{1, \dots, s\}$, $\mathcal{S}_{\mathcal{I},e}^{(\mathbf{a})}$ generates a radical and equidimensional ideal and $V_{\mathcal{I},e}^{(\mathbf{a})}$ is smooth of dimension $n - \#\mathcal{I}$ if it is not empty. Then, $\lim_0(\mathcal{K}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})}))$ belongs to the union of the sets of solutions of $(I_i + \langle f_{i_1} \rangle) \cap \mathbb{Q}[X_1]$ for $i = 0$ to $\#\delta_{\mathcal{I}} - 1$.

Thus, the algorithm consists, for all $\mathcal{I} \subset \{1, \dots, s\}$, in computing $\lim_0(\mathcal{K}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})}))$ which is represented by a univariate polynomial. Then, it isolates the real roots of these polynomials and find one rational number r between each successive isolated real root. Then, it performs recursively, substituting X_1 by r in the input polynomials.

Note the difference with CAD algorithm: here we obtain boundary points of the projections of the connected components of the studied semi-algebraic set by computing directly limits of critical points. This allows to avoid the growth

of degree and the appearance of superfluous values induced by the recursive projection step of CAD. Note also that we never compute the critical loci $\mathcal{C}(\pi_E, V_{\mathcal{I},e}^{(\mathbf{a})})$ considered in the algorithm described in the previous paragraph.

REMARK 3. If $V_{\mathcal{I},0}^{\mathbf{a}}$ is smooth and $\mathcal{S}_{\mathcal{I},0}^{\mathbf{a}}$ generates a radical equidimensional ideal, $\lim_0(\mathcal{K}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})}))$ is contained in $\mathcal{K}(\pi_1, V_{\mathcal{I},0}^{\mathbf{a}})$.

Given $\mathcal{I} \subset \{1, \dots, s\}$, note that the computations performed by the algorithm we present in this paragraph can be seen as computing $\mathfrak{C}_{\mathcal{I}}^{\mathbf{a}} \cap \{(x, 0) \in \mathbb{C}^{n+1}\}$. The following result shows that the degree of the curve $\mathfrak{C}_{\mathcal{I}}^{\mathbf{a}}$ is well-controlled and is related to the positive dimensional components of $\mathfrak{C}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})})$.

PROPOSITION 3. The degree of $\mathfrak{C}_{\mathcal{I}}^{\mathbf{a}}$ is bounded by $D^\ell(D-1)^{n-\ell} \binom{n-1}{\#\mathcal{I}} - \delta_{\text{sing}}$ where δ_{sing} is the sum of the degrees of the positive irreducible components of $\mathfrak{C}(\pi_E, V_{\mathcal{I},E}^{(\mathbf{a})})$.

Its proof uses a similar reasoning than those in [24] and bi-homogeneous bounds computed from a Lagrangian formulation of critical points (see [28]).

REMARK 4. When $\mathcal{S}$ is not supposed to be bounded, the above algorithm has to be modified to compute sampling points of $\mathcal{S}$ by substituting the computation of $\lim_0(\mathcal{K}(\pi_1, V_{\mathcal{I},e}^{(\mathbf{a})}))$ by the computation of limits of generalized critical values of π_1 restricted to $V_{\mathcal{I},e}^{(\mathbf{a})}$ when e tends to 0.

5. COMPUTATIONS AND RESULTS

We show here the results of the computations we obtained solving the P3P problem. We do the computation by restriction to the case where the triangle we observe is isosceles, that is: $l_0 = l_1$. The system we consider is:

$$\begin{cases} 1 &= A^2 + B^2 - ABu \\ 1 &= B^2 + C^2 - BCv \\ x &= A^2 + C^2 - ACw \end{cases}$$

It has 4 parameters u, v, w, x and 3 unknowns A, B, C .

All the computations have been performed on a PC Intel(R) Xeon(TM) CPU 3.20GHz with 6Gb of RAM.

The minimal discriminant variety. We first compute the minimal discriminant variety with the *DV* software in about 1 minute. The result is the polynomial D given in appendix. It is the minimal discriminant variety of the P3P parametric system when the triangle is isosceles. We can notice that D has 7 factors of respective degrees 1, 1, 1, 2, 2, 3, 13, and whose number of terms is at most 153. Along with the constraints on the parameters, the discriminant variety allows us to define the following semi-algebraic set:

$$D \neq 0, x > 0 - 2 < u < 2, -2 < v < 2, -2 < w < 2$$

The parametric system has a constant number of solutions on each connected component of this semi-algebraic set.

REMARK 5. The above semi-algebraic set is not bounded in the variable x , which is needed to apply the methods presented in section 4.

Thus we split this set into $x < 1$ and $x > 1$. Using the variable $y = \frac{1}{x}$, this leads to the study of two bounded semi-algebraic set:

$$\mathcal{H}_x \begin{cases} D \neq 0 \\ 0 < x < 1 \\ -2 < u < 2 \\ -2 < v < 2 \\ -2 < w < 2 \end{cases} \quad \text{and} \quad \mathcal{H}_y \begin{cases} D_y \neq 0 \\ 0 < y < 1 \\ -2 < u < 2 \\ -2 < v < 2 \\ -2 < w < 2 \end{cases}$$

where D_y denotes the polynomial obtained by the substitution of x by $\frac{1}{y}$ in $y^5 D$.

Solving polynomial systems of inequalities. We consider now the two semi-algebraic sets $\mathcal{H}_x$ and $\mathcal{H}_y$. Thanks to the property of the discriminant variety D , we know that on each connected component of these semi-algebraic sets, the parametric system has a constant number of solutions.

To get a *open classification* we first tried to compute a Cylindrical Algebraic Decomposition. However, after one month of computation, we could only complete the projection phase, but not the lifting phase neither with *Maple* nor with *Magma* software. Finally, we implemented the algorithms computing sampling points described in Section 4 in semi-algebraic sets defined by the systems $\mathcal{H}_x$ and $\mathcal{H}_y$. The first algorithm returned a result after 3 weeks of computations, and the second after 3 days. As explained above, this is mainly due to the fact that the discriminant variety contains singularities of high dimension. More generally, we observed that the computation of critical values of the projection π_E considered in Section 5 were particularly difficult. The critical loci of this projection restricted to the varieties considered in Subsection 5.1 have a big dimension. Most of the time spent by the first algorithm described in Subsection 5.1 is spent in these computations. The second algorithm described in Subsection 5.2 avoids the computations of the singularities which appear during the running of the first algorithm. Moreover, as explained in Subsection 5.2, its complexity depends on the *real* geometry of the considered semi-algebraic set. This probably explains why it is so efficient in our case.

These implementations will be soon available in the next release of the RAGLIB Maple package. We successfully got one point in each connected components of $\mathcal{H}_x$ and $\mathcal{H}_y$. As a result we get 13612 points distributed in every connected cell of $\mathcal{H}_x$ and 46474 points in $\mathcal{H}_y$. These points can be downloaded at

<http://www-spiral.lip6.fr/~moroz/P3P.html>

Note that contrarily to polynomials generated randomly, the minimal discriminant variety contains singularities of high dimensions which makes them more difficult to study. Moreover, since D is a minimal discriminant variety, this also ensures us that all conditions on the parameters discriminating the parameters' space according to the number of solutions of the system would contain such singularities.

As we can see on the figures 1 and 2, some connected cells seem very small and almost intractable with random approximations. The drawings show the graph of $\mathcal{H}_x$ around the point p_0 , one of the points returned by our computation:

$$\begin{aligned} p_0 &:= (x_0, u_0, v_0, w_0) \\ &= (452735729, 3371082457, 2763844376, 26504177576) \\ &\quad (9148876946, 1706654848, 1399264123, 13260182015) \\ &\simeq (0.0494853, 1.97525, 1.97521, 1.99877) \end{aligned}$$

On each figure, we present 2 slices centered on p_0 . The first figure shows a global view of $\mathcal{H}_x$ and p_0 , while the second figure shows a much closer neighborhood of p_0 . According

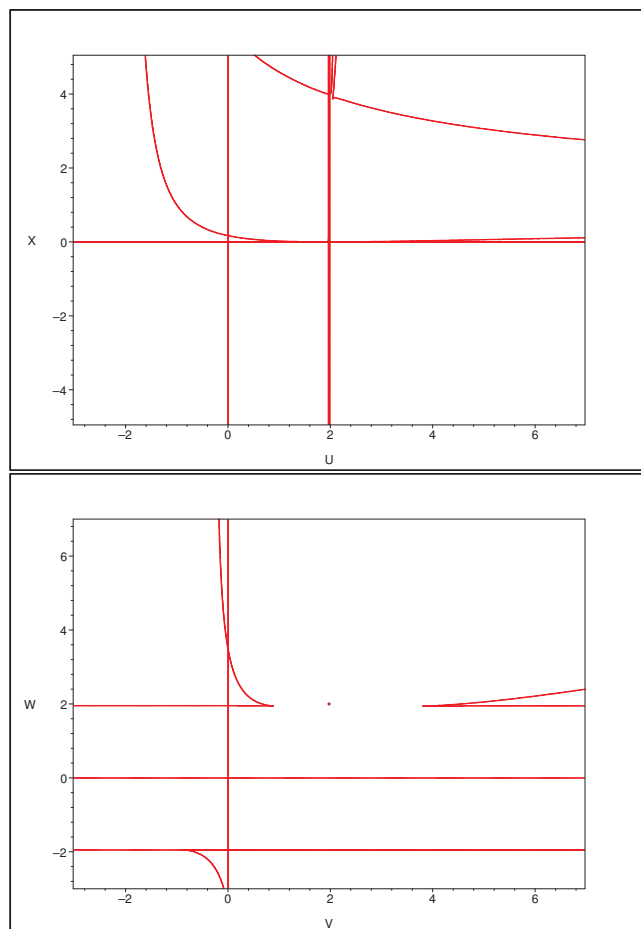


Figure 1: Two slices of $\mathcal{H}$ partly specialized in p_0 : - in the first, the variables v and w are specialized - in the second, the variables u and x are specialized

to the slices, we can see that we have detected here a very small connected cell of $\mathcal{H}_x$.

More generally, the set of points we computed intersects each connected component of $\mathcal{H}_x$ and $\mathcal{H}_y$, and we now need to compute the number of solutions of the parametric system specialized in each point to achieve our classification.

Zero-dimensional system solving. In this step, we compute the number of real solutions satisfying the constraint of the problem for 60086 parameters' values. The mean time to solve each corresponding 0-dimensional system is about 0.05 second.

Finally, we can recover the fact that the parametric system of our problem may have exactly 0, 1, 2, 3 or 4 solutions satisfying the inequalities' constraints. We present in table 1 a sample point in the parameters' space where the system has i solutions for i from 0 to 4.

Moreover, even if we do not have a complete CAD of the discriminant variety, we can have a geometric view of each connected cell of the parameters' space associated to a given number of solutions by drawing the neighborhood of each computed point. As we saw in the previous section, this allowed us for example to exhibit a very small cell, and to compute the number of distinct solutions of the system restricted to this cell, which is exactly 4.

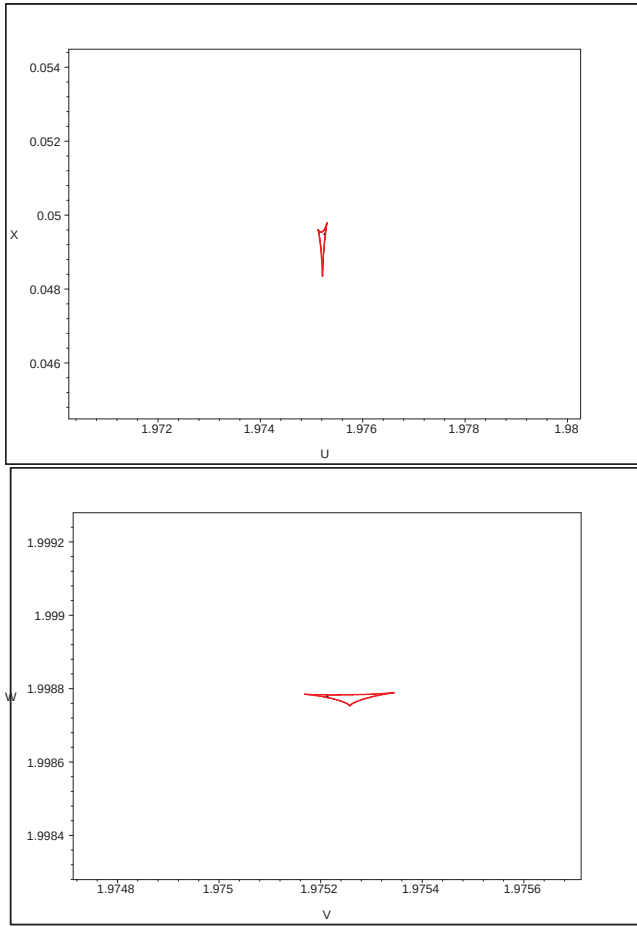


Figure 2: Two closer slices of $\mathcal{H}$ partly specialized in p_0

6. CONCLUSION

We give in this article a full algorithm to compute an intuitive classification of the parameters of a parametric system given by polynomial equations and inequations. Our method is validated by the description of the parameters' space of the P3P problem when the three control points form an isosceles triangle. We also think that optimizations our implementations and algorithms are possible to make the study of a more general classification.

APPENDIX

Elements of discriminant variety theory

The discriminant variety is presented in [17]. We recall here how to compute it for a *well-behaved* parametric system.

DEFINITION 4. A parametric system $S_{\mathbf{T}}$ is said *well-behaved* if and only if:

- i) The number of equations equals the number of unknowns
- ii) For all p_0 outside a Zariski closed set, S_{p_0} is radical and zero-dimensional.

The P3P problem and most of the problems coming from applications are modeled by well-behaved systems.

Given a *well-behaved* parametric system $S_{\mathbf{T}}$, let g denote the product of the polynomial inequations of $S_{\mathbf{T}}$. and π the projection map from the solutions of $S_{\mathbf{T}}$ to the parameters' space. If $\mathcal{F}$ is a subset of the parameters' space, then $S_{\mathcal{F}}$

Number of solutions	x	u	v	w
0	452735729	-1087810617	-2322378129	4610994663
	9148876946	4897634788	10447926511	2334015862
1	452735729	-1087810617	-2322378129	-10016606887
	9148876946	4897634788	10447926511	5135366188
2	452735729	-1087810617	2322378129	10016606887
	9148876946	4897634788	10447926511	5135366188
3	452735729	-1087810617	1270625905	2776826855
	9148876946	4897634788	5709068079	1423637843
4	1415953531	4824522087	2413516911	11184766673
	12404789665	13860411335	4607583958	5921669493

Table 1: Sample parametric points corresponding to a wanted number of solutions

denotes the restriction of the parametric system $S_{\mathbf{T}}$ to $\mathcal{F}$. The discriminant variety can be decomposed in four algebraic components:

- i) V_{ineq} is the projection of the zeros of the polynomial equations and g
- ii) V_{sing} is the Zariski closure of the projection of the singular locus of π
- iii) V_c is the closure of the critical values of π
- iv) V_{∞} is the set of parameters' values p_0 such that for all neighborhood $\mathcal{U}_0$ of p_0 , the real solutions of $S_{\mathcal{U}_0}$ are not bounded.

The components V_{ineq} , V_{sing} , V_c may be computed by saturation and elimination of variables, which may be handled with Gröbner bases computations (see [4] for example). The component V_{∞} may be obtained by extracting some coefficients of a gröbner basis with relation to a block ordering satisfying $\mathbf{X} \gg \mathbf{T}$. More details on these computation may be found in [17]. Beside, complexity results of this method are given in [20].

Discriminant variety for the isosceles P3P problem

$$\begin{aligned}
D := & x(-x+2+w)(x-2+w) \\
& (-x+u^2)(-x+v^2)(-uvw+w^2-4+v^2+u^2) \\
& (-2x^2u^3v^5w^3-72xuv^5w-8u^3v^3w^3-96x^4u^3vw+ \\
& 6x^2uv^5w^3+4x^3u^4v^4w^2-8x^2u^3v^3w^3+1248xu^2v^2- \\
& 24x^3u^4v^2-4x^3u^6w^2-4x^3v^6w^2-24x^3u^2v^4- \\
& 96xu^2v^4-128x^5v^2-18x^3u^2v^4w^2-384xuv^3w- \\
& 18x^3u^4v^2w^2-12uv^5w^3-96xu^4v^2+24x^3v^4w^2- \\
& 240xu^4+576xu^4+x^5u^4v^4-768x^2u^2+ \\
& 64x^5u^2v^2+576x^2v^4-768x^2v^2+64x^4u^4- \\
& 416x^3v^4+64x^3v^6-96x^4uv^3w+256x^4uvw+ \\
& 48x^2u^3v^3w^3+8x^2uv^5w+12x^2u^6v^2+168xv^4w^2- \\
& 2x^2u^2v^6w^2+12x^2v^6w^2+12x^2u^2v^6-40x^2u^4v^4+ \\
& 168xu^4w^2+12x^2u^6w^2-8xu^4v^4+16x^5v^4+ \\
& xu^6v^2w^2-768x^3uvw+32x^4u^3v^3w-4xu^6v^2- \\
& 12xu^6w^2-12xv^6w^2+96xu^3v^3w^3+16x^2u^4v^4w^2+ \\
& 8x^4u^5v^3w+48x^2u^3v^3w^3+96x^3u^3vw-2x^2u^6v^2w^2+ \\
& 96x^3uv^3w+60xu^4v^2w^2+96xu^3v^3w^3+60xu^2v^4w^2+ \\
& 6xu^2v^2w^4+8x^4uv^5w-336xu^2v^2w^2-384xuv^3w- \\
& 2xu^4vw^2-4xu^2v^6-1152x^2u^2v^2-27xu^4w^4- \\
& 16x^4u^6-96x^2v^6+64x^3u^6+64xu^6+ \\
& 64xv^6-240xu^4-128x^5u^2-1024x^4+ \\
& 1024x^3+768x^2uvw+xu^2v^6w^2+8xu^5v^3w+ \\
& 6xu^4v^2w^4+48u^5vw+8x^3u^5vw+16x^5u^4- \\
& 27xv^4w^4+24x^3u^4w^2+4x^4u^6v+192x^2v^2w^2- \\
& 128x^4u^2v^2+4v^6w^2-4x^4u^4v^4+64x^4v^4+ \\
& 6xu^2v^4w^4+8xu^3v^5w-2xu^5v^3w^3+192x^2u^3vw+ \\
& 6xu^5vw^3+96u^3v^3w+192x^2uvw+32x^3u^3v^3w- \\
& 2xu^3v^5w^3+4u^6w^2+256x^3u^2v^2+96x^3v^2w^2+ \\
& 48xu^5w-36u^4v^2w^2+256x^4u^2-8x^5u^4v^2- \\
& 416x^3u^4+256x^4v^2+384x^3v^2-96x^2u^6- \\
& 72xu^5vw-76xu^3v^3w^3+12u^4v^2w^4+4x^4u^2v^6- \\
& 12x^3u^6v^2+96x^3u^2w^2-256x^3w^2-8x^5u^2v^4- \\
& 16v^6-16x^4v^6+384x^3u^2-16u^6+ \\
& 8x^3uv^5w+xu^4v^4w^4+6xuv^5w^3-36u^2v^4w^2+ \\
& 12u^2v^4w^4-4u^3v^3w^5-2x^4u^3v^5w+x^3u^6v^2w^2- \\
& 6x^3u^5v^3w-192x^2v^4w^2-48u^2v^4-12x^3u^2v^6- \\
& 192x^2u^4w^2+8x^4u^5vw+192x^2u^2w^2-48u^4v^2+ \\
& 144xu^3v^3w-2x^4u^5v^3w-192x^2uvw^3+176x^2uv^2+ \\
& 24x^3u^4v^4+176x^2u^2v^4-160x^2u^3v^3w-6x^3u^3v^5w+ \\
& 6x^2u^5v^3w-12u^5vw^3+x^3u^2v^6w^2-2x^2u^5v^3w^3+ \\
& 256x^5)
\end{aligned}$$

A. REFERENCES

- [1] M.-A. Ameller, B. Triggs, and L. Quan. Camera pose revisited - new linear algorithms, December 13 2000.
- [2] S. Basu, R. Pollack, and M.-F. Roy. A new algorithm to find a point in every cell defined by a family of polynomials. In *Quantifier elimination and cylindrical algebraic decomposition*. Springer-Verlag, 1998.
- [3] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in real algebraic geometry*. Springer-Verlag, 2003.
- [4] E. Becker and V. Weipsfenning. *Gr bner bases: a computational approach to commutative algebra*. Graduate Texts in Mathematics: readings in Mathematics. Springer-Verlag, 1993.
- [5] J.-C. Faug re. A new efficient algorithm for computing Gr bner bases (F4).-. *Journal of Pure and Applied Algebra*, 139(1-3):61-88, 1999.
- [6] J.-C. Faug re. A new efficient algorithm for computing Gr bner without reduction to zero (F5). In *Proceedings of ISSAC 2002*, pages 75 - 83. ACM Press, 2002.
- [7] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381-395, 1981.
- [8] S. Ganapathy. Decomposition of transformation matrices for robot vision. In *CRA84*, pages 130-139, 1984.
- [9] X.-S. Gao, X. Hou, J. Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(8):930-943, 2003.
- [10] X. S. Gao and J. Tang. On the probability of the number of solutions for the P4P problem. *Journal of Mathematical Imaging and Vision*, 25(1):79-86, July 2006.
- [11] I. M. Gelfand, M. M. Kapranov, and A. V. Zelevinsky. *Discriminants, resultants, and multidimensional determinants*. Mathematics: Theory & Applications. Birkh user Boston Inc., Boston, MA, 1994.
- [12] D. Grigoriev and N. Vorobjov. Solving systems of polynomials inequalities in subexponential time. *Journal of Symbolic Computation*, 5:37-64, 1988.
- [13] R. M. Haralick, K. Ottenberg, and M. Nolle. Analysis and solutions of the three point perseperspective pose estimation problem. In *Proceedings Computer Vision and Pattern Recognition '91*, pages 592-598, Lahaina, Maui, June 1991.
- [14] J. Heintz, M.-F. Roy, and P. Solern . On the theoretical and practical complexity of the existential theory of the reals. *The Computer Journal*, 36(5):427-431, 1993.
- [15] Z.Y. Hu and F.C. Wu. A note on the number of solutions of the noncoplanar p4p problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):550-555, 2002.
- [16] B. Lacolle, O. Leboulleux, B. Conio, and R. Horaud. An analytic solution for the perspective 4-point problem. *CVGIP: Image Understanding*, 48(2):277-278, November 1989.
- [17] D. Lazard and F. Rouillier. Solving parametric polynomial systems. *Journal of Symbolic Computation*, 42:636-667, 2007.
- [18] S. Liang, J. Gerhard, and D. Jeffrey. A new maple package for solving parametric polynomial systems. In *MACIS*, 2007.
- [19] J. Milnor. *Morse Theory*, volume 51 of *Annals of mathematics studies*. Princeton University Press, Princeton, 1963.
- [20] G. Moroz. Complexity of the resolution of parametric systems of polynomial equations and inequations. In Jean-Guillaume Dumas, editor, *International Symposium on Symbolic and Algebraic Computation*, pages 246-253. ACM Press, 2006. isbn: 1-59593-276-3.
- [21] L. Quan and Z.-D. Lan. Linear N-point camera pose determination. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(8):774-780, 1999.
- [22] G. Reid, J. Tang, and L. Zhi. A complete symbolic-numeric linear method for camera pose determination. In J. Rafael Senda, editor, *ISSAC 2003*, pages 215-223, pub-ACM:adr, 2003. ACM Press.
- [23] F. Rouillier. Solving zero-dimensional systems through the Rational Univariate Representation. *AAECC Journal*, 9(5):433-461, 1999.
- [24] M. Safey El Din. Finding sampling points on real hypersurfaces in easier in singular situations. In *MEGA (Effective Methods in Algebraic Geometry) Electronic proceedings*, 2005.
- [25] M. Safey El Din. Practical and theoretical issues for the computation of generalized critical values of a polynomial mapping and its applications. In D. Kapur, editor, *Proceedings of Asian Symposium on Computer Mathematics 2007*, volume 5081 of LNAI. Springer-Verlag, 2008.
- [26] M. Safey El Din. Testing sign conditions on a multivariate polynomial and applications. *Mathematics in Computer Science*, 1(1):177-207, December 2007.
- [27] M. Safey El Din and  . Schost. Polar varieties and computation of one point in each connected component of a smooth real algebraic set. In *Proceedings of ISSAC 2003*, pages 224-231. ACM Press, 2003.
- [28] M. Safey El Din and P. Tr buchet. Strong bi-homogeneous B zout theorem and its use in effective real algebraic geometry. *INRIA Research Report*, 2006.
- [29] Y. H. Wu and Z. Y. Hu. PnP problem revisited. *Journal of Mathematical Imaging and Vision*, 24(1):131-141, January 2006.
- [30] L. Yang. A simplified algorithm for solution classification of the perspective-three-point problem, December 1998.
- [31] C.-X. Zhang and Z.-Y. Hu. A general sufficient condition of four positive solutions of the p3p problem. *J. Comput. Sci. Technol.*, 20(6):836-842, 2005.

Complete Subdivision Algorithms, II: Isotopic Meshing of Singular Algebraic Curves

[Extended Abstract]

Michael Burr^{*}

Sung Woo Choi[†]

Benjamin Galehouse^{*}

Chee K. Yap^{*}

ABSTRACT

Given a real function $f(X, Y)$, a box region B and $\varepsilon > 0$, we want to compute an ε -isotopic polygonal approximation to the curve $C : f(X, Y) = 0$ within B . We focus on subdivision algorithms because of their adaptive complexity. Plantinga & Vegter (2004) gave a numerical subdivision algorithm that is exact when the curve C is non-singular. They used a computational model that relies only on function evaluation and interval arithmetic. We generalize their algorithm to any (possibly non-simply connected) region B that does not contain singularities of C . With this generalization as subroutine, we provide a method to detect isolated algebraic singularities and their branching degree. This appears to be the first complete *numerical* method to treat implicit algebraic curves with isolated singularities.

Categories and Subject Descriptors

F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems— *Geometrical Problems and Computations*

General Terms

Algorithms, Theory

Keywords

Meshing, Singularity, Root bound, Evaluation bound, Im-

PLICIT algebraic curve, Complete numerical algorithm, Subdivision algorithm.

1. INTRODUCTION

Given $\varepsilon > 0$, a box region B and a real function $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, we want to compute a polygonal approximation P to the implicit curve $C : f = 0$ restricted to B . The approximation P must be (1) “topologically correct” and (2) “ ε -close” to $C \cap B$. We use the standard interpretation of requirement (2), that $d(P, C \cap B) \leq \varepsilon$ where $d(\cdot, \cdot)$ is the Hausdorff distance on compact sets. In recent years, it has become accepted [2] to interpret requirement (1) to mean P is isotopic to $C \cap B$, which we denote by $P \approx C \cap B$. This means we not only require that P and $C \cap B$ to be homeomorphic, but they must be embedded in $\mathbb{R}^2$ “in the same way”. E.g., if $C \cap B$ consists of two ovals, these can be embedded in $\mathbb{R}^2$ as two ovals exterior to each other, or as two nested ovals. Isotopy, but not homeomorphism, requires P to respect this distinction. In this paper, we mainly focus on topological correctness, since achieving ε -closeness is not an issue for our subdivision approach (but cf. [2, pp. 213-4]).

We may call the preceding the **2-D implicit meshing problem**. The term “meshing” comes from the corresponding problem in 3-D: given $\varepsilon > 0$ and an implicit surface $S : f(X, Y, Z) = 0$, we want to construct a triangular mesh M such that $d(M, S) \leq \varepsilon$ and $M \approx S$. It is interesting to identify the 1-D meshing with the well-known problem of real root isolation and refinement for a real function $f(X)$.

Among the approaches to most computational problems on curves and surfaces, the algebraic approaches and geometric/numerical approaches constitute two extremes of a spectrum. Algebraic methods can clearly solve most problems in this area, e.g., by an application of the general theory of cylindrical algebraic decomposition (CAD) [1]. Purely algebraic methods are generally not considered practical, even in the plane (e.g., [13, 21]). But efficient solutions have been achieved for special cases such as intersecting quadrics in 3-D [20]. At the other end of the spectrum, we have the geometric/numerical approaches that emphasize numerical approximation and iteration. An important class of such algorithms is the class of **subdivision algorithms** which can be viewed as generalized binary search. Such algorithms are practical in two senses: they are easy to implement and their complexity is more adaptive [26]. Another key feature

^{*}Courant Institute, NYU, 251 Mercer Street, NY, NY 10012. Emails: {burr@cims, galehouse@cims, yap@cs}.nyu.edu. Yap’s work is supported by NSF Grant CCF-0728977. Burr’s work is partially supported by NSF Grant DMS-0701578. Galehouse’s work is partially supported by the DoE under contract DE-FG02-00ER25053.

[†]Department of Mathematics, Duksung Women’s University, Seoul, Korea. Email: swchoi@duksung.ac.kr

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC’08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

of subdivision algorithms is “locality”, meaning that we can restrict our computation to some region of interest.

Unfortunately, geometric/numerical methods seldom have global correctness guarantees. The most famous example is the Marching Cube algorithm (1987) of Lorensen & Cline. Many authors have tried to improve the correctness of subdivision algorithms (e.g., Stander & Hart [24]). So far, such efforts have succeeded under one of the following situations:

- (A0) Requiring niceness assumptions such as being non-singular or Morse.
- (A1) Invoking algebraic techniques such as resultant computation or manipulation of algebraic numbers.

It is clear that (A0) should be avoided. Generally, we call a method “complete” if the method is correct without any (A0) type restrictions. But many incomplete algorithms (e.g., Marching cube) are quite useful in practice. We want to avoid (A1) because algebraic manipulations are harder to implement and such techniques are relatively expensive and non-adaptive [26]. For instance, the subdivision meshing algorithm of Plantinga & Vegter [19, 18] requires an (A0) assumption, the non-singularity of surfaces. The subdivision algorithm of Seidel & Wolpert [21] requires¹ an (A1) technique, namely, the computation of resultants. We thus classify [21] as a hybrid approach that combines geometric/numerical with algebraic techniques. Prior to our work, we are not aware of any meshing algorithm that can handle singularities without resorting to resultant computation. In general, hybrid methods offer considerable promise (e.g., Hong [13]). This is part of a growing trend to employ numerical techniques for speeding up algebraic computation.

The recent collection [2, Chapter 5] reviews the current literature in meshing in 2- and 3-D: the subdivision algorithms of Snyder [23, 22] and also Plantinga & Vegter; the sampling approach of Boissonnat & Oudot [4] and Cheng, Dey, Ramos and Ray [8]; the Morse Theory approaches of Stander & Hart [24] and Boissonnat, Cohen-Steiner & Vegter [3]; and an algebraic sweepline approach of Mourrain & T  court [17]. The subdivision algorithm of Plantinga & Vegter is remarkable in the following sense: even though it is globally isotopic, it does not guarantee isotopy of the curve within each cell of the subdivision. In contrast, Snyder’s subdivision approach [23, 22] computes correct topology in each cell (the algorithm is currently incomplete [2, p. 195]).

The basic idea of sampling approaches is to reduce meshing of a surface S to computing the Delaunay triangulation of a sufficiently dense set of sample points on S [2, p. 201–213]. Cheng, Dey, Ramos and Ray [8] need the primitive operation of solving a system of equations involving f and its derivatives. Boissonnat and Oudot [4] need a primitive for intersecting the surface with a Voronoi edge. These primitives yield sample points on the surface. These points are algebraic, so implementing the primitives exactly would require strong algebraic techniques. For restrictions and open problems in sampling approaches, see [2, p. 227–229]. In sharp contrast, the computational primitives needed by the Plantinga & Vegter approach works directly with bigfloats, with modest requirements on f .

The complete removal of (A0) type restrictions is the major open problem faced by purely numerical approaches to

¹Their paper subtitle “Exploiting a little more Geometry and a little less Algebra” speaks to our concerns with (A1).

meshing. Thus, Boissonnat et al [2, p. 187] states that “*meshing in the vicinity of singularities is a difficult open problem and an active area of research*”. Most of the techniques described in their survey are unable to handle singularities. It should be evident that this open problem has an implicit requirement to avoid the use of (A1) techniques.

The present paper presents a purely numerical subdivision method for meshing algebraic curves with isolated singularities. In a certain sense, this is the most general geometric situation since reduced algebraic curves have only isolated singularities by Proposition 1. Our starting point is the algorithm of Plantinga & Vegter [19, 18] for implicit meshing of curves. It is important to understand the computational model of Plantinga & Vegter which is also used in this paper. Two capabilities are assumed with regards to $f(X, Y)$:

- (i) Sign evaluation of $f(p)$ at dyadic points p .
- (ii) f is C^1 and we can evaluate the interval analogues of f , $\frac{\partial f}{\partial X}$, $\frac{\partial f}{\partial Y}$ on dyadic intervals.

Note that Marching Cube only requires capability (i). Let the **class** PV denote the set of all real functions $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ for which capabilities (i) and (ii) are available. Many common functions of analysis (e.g., hypergeometric functions [10]) belong to PV . So the approach of Plantinga & Vegter admits a more general setting than algebraic curves.

Some of our recent work that addresses the above (A0)/(A1) concerns include [26] (Bezier curve intersection), [7] (solving triangular systems), [6] (numerical root isolation for multiple zeros) and [5] (integral analysis of real root isolation). The last two papers study the 1-D analogue of the Plantinga & Vegter Algorithm. The philosophy behind all these papers is the design and analysis of complete numerical methods based on approximations, iteration and adaptive methods. Topological exactness is achieved using suitable algebraic bounds, ranging from classic root separation bounds to evaluation bounds and geometric separation bounds. We emphasize that the worst-case complexity of adaptive algorithms (e.g., as determined by the worst case root bounds) ought not to be the chief criterion for evaluating the usefulness of these algorithms: for the majority of inputs, these algorithms terminate fast. Note that the zero bounds are only used as stopping criteria for iteration in the algorithms, and simple estimates can easily be computed. Such computations does not mean we compute resultants, even though their justification depend on resultant theory. The present paper continues this line of investigation.

Overview of Paper.

- In Section 3, we extend the Plantinga & Vegter algorithm to compute an isotopic approximation of the curve $C : f = 0$ restricted to a “nice region” that need not be simply connected. C may have singularities outside R and we only need $f \in PV$.
- In Section 4, we provide the algebraic evaluation bounds necessary for meshing singular curves.
- In Section 5, we provide a subdivision method to isolate all the singularities of a square-free integer polynomial $f(X, Y)$.
- In Section 6, given a box B containing an isolated singularity p , we provide a method to compute the branching degree of p .

- In Section 7, we finally present the overall algorithm to compute the isotopic polygonal approximation.
- We conclude in Section 8.

Basic Terminology. Let $\mathbb{F} := \mathbb{Z}[\frac{1}{2}] = \{m2^n : m, n \in \mathbb{Z}\}$ be the set of **dyadic numbers** or **bigfloats**. All our numerical computation are performed in $\mathbb{F}$. For $S \in \mathbb{R}$, let $\Box S$ be the set of closed intervals $[a, b]$ with endpoints in S , $\{a, b\} \subseteq S$. We write $\Box S^n$ for $(\Box S)^n$. In particular, $\Box \mathbb{F}$ is the set of dyadic intervals, and $\Box \mathbb{R}^n$ is the set of n -boxes. The boundary of a set $S \subseteq \mathbb{R}$ is denoted ∂S . If $f : \mathbb{R}^n \rightarrow \mathbb{R}$, and $S \subseteq \mathbb{R}$, then $f(S) := \{f(x) : x \in S\}$. A function $\Box f : \Box \mathbb{F}^2 \rightarrow \Box \mathbb{F}$ is a **box function** for f provided (i) $f(B) \subseteq \Box f(B)$ and (ii) if $B_0 \supseteq B_1 \supseteq \dots$ and $\lim_i B_i \rightarrow p$ then $\lim_i \Box f(B_i) \rightarrow f(p)$. We regard the limit of intervals in terms of the limits of their endpoints. We say $f \in PV$ if $f \in C^1$ (has continuous derivatives $\frac{\partial f}{\partial X}, \frac{\partial f}{\partial Y}$), there is an algorithm to determine $\text{sign}(f(p))$ for $p \in \mathbb{F}^2$ and $\Box f, \Box \frac{\partial f}{\partial X}, \Box \frac{\partial f}{\partial Y}$ are computable.

The **size** of a box is the maximum length of one of its sides (all of our boxes will be square). We **split** a box B by subdividing it into 4 subboxes of equal size. These subboxes are the **children** of B , of half the size of B . Two boxes are **neighbors** if one box has an edge that overlaps an edge of the other box (the two boxes may have different sizes). Starting with B_0 , the child-parent relationships obtained by an arbitrary sequence of splits yields a **quadtree** rooted at B_0 . The set of leaves in such a quadtree constitute a partition of B_0 . We only consider boxes of the form $B = I \times J$ where I, J are dyadic intervals. For simplicity, we assume B is square although it is possible to extend our algorithms to boxes with aspect ratio at most 2.

Basic Algebraic Facts. Let $\mathbb{D}$ be a UFD and $f, g \in \mathbb{D}[\mathbf{X}] = \mathbb{D}[X_1, \dots, X_n]$ where $\mathbf{X} = (X_1, \dots, X_n)$. We say f, g are **similar** if there exists $a, b \in \mathbb{D} \setminus \{0\}$ such that $af = bg$, and write $f \sim g$. Otherwise, f and g are **dissimilar**. The **square-free part** of f is defined as

$$\text{SqFree}(f) := \frac{f}{\text{GCD}(f, \partial_1 f, \dots, \partial_n f)} \quad (1)$$

where $\partial_{X_i} = \partial_i$ indicates differentiation with respect to X_i . f is said to be **square-free** if $\text{SqFree}(f) = f$. From (1) we see that computing $\text{SqFree}(f)$ from f involves only rational operations of $\mathbb{D}$. As the gradient of f is $\nabla f = (\partial_1 f, \dots, \partial_n f)$, we may also write $\text{GCD}(f, \nabla f)$ for $\text{GCD}(f, \partial_1 f, \dots, \partial_n f)$. See [25, Chap. 2] for standard conventions concerning GCD .

Let k be an algebraically closed field. For $S \subseteq k[\mathbf{X}] = k[X_1, \dots, X_n]$ and $B \subseteq k^n$, let $\text{Zero}_B(S) := \{p \in B : f(p) = 0 \text{ for all } f \in S\}$ denote the **zero set of S relative to B** . If $B = k^n$, then we simply write $\text{Zero}(S)$.

In 1-dimension, a square-free polynomial $f \in \mathbb{Z}[X]$ has no singularities (i.e., multiple zeros). We now recall two generalizations of this result that will be necessary in the remainder of the paper. See [12, 9, 11] for similar results.

PROPOSITION 1 ([11, Ex.14.3]). *The singular points of any variety form a proper subvariety.*

The singular points of $\text{Zero}(f)$ are defined to be the points where $\nabla \text{SqFree}(f) = 0$. The above result is critical in our paper, because it implies that if $f \in \mathbb{R}[X, Y]$ is square-free, then the singular points are a proper subvariety of union of curves and hence must be a finite set of points.

PROPOSITION 2 (Sard's Theorem [11, Prop.14.4]). *Let $f : X \rightarrow Y$ be any surjective map of varieties defined over a field k of characteristic 0. Then there exists a nonempty open subset $U \subseteq Y$ such that for any smooth point $p \in f^{-1}(U) \cap X_{sm}$ in the inverse image of U , the differential df_p is surjective.*

Note that X_{sm} denotes the set of smooth points of variety X . The condition that the differential df_p is surjective is equivalent to insisting that $\nabla f(p) \neq 0$. The most important example that we consider is $f : \mathbb{R}^2 \rightarrow \mathbb{R}$. Every point in $\mathbb{R}^2 = X$ is smooth and $\mathbb{R} \setminus U$ is only a finite set. Hence, there are only a finite number of level sets parameterized by h where $\text{Zero}(f(X, Y) - h)$ has a singular point.

2. ALGORITHM OF Plantinga & Vegter

First we recall the Plantinga & Vegter algorithm. Given $\varepsilon > 0$, a box $B_0 \in \Box \mathbb{F}^2$ and $f : \mathbb{R}^2 \rightarrow \mathbb{R}$, we want to compute a polygonal ε -approximation P of the restriction of the curve $C : f = 0$ to B_0 : $d(P, C \cap B_0) \leq \varepsilon$ and $P \approx C \cap B_0$. For simplicity, we focus on topological correctness: $P \approx C \cap B_0$, since it is easy to refine the subdivision to achieve $d(P, C \cap B_0) \leq \varepsilon$. The Plantinga & Vegter algorithm is based on two simple predicates on boxes B :

- Predicate $C_0(B)$ holds if $0 \notin \Box f(B)$.
- Predicate $C_1(B)$ holds if $0 \notin (\Box \frac{\partial f}{\partial X}(B))^2 + (\Box \frac{\partial f}{\partial Y}(B))^2$.

These predicates are easily implemented if $f \in PV$, using interval arithmetic. Moreover, if $C_0(B)$ holds, then the curve C does not intersect B . Note that if B satisfies C_1 , then any child of B also satisfies C_1 .

The input box B_0 is a dyadic square, and output is an undirected graph $G = (V, E)$ where each vertex $v \in V$ is a dyadic point, $v \in \mathbb{F}^2$. G represents a polygonal approximation P of $C \cap B_0$.

The algorithm has 3 phases, where Phase i ($i = 1, 2, 3$) is associated with a queue Q_i containing boxes. Initially, $Q_1 = \{B_0\}$, and $Q_2 = Q_3 = \emptyset$. When Q_i is empty, we proceed to the Phase $i + 1$.

- **PHASE 1: SUBDIVISION.** While Q_1 is non-empty, remove some B from Q_1 , and perform the following: If $C_0(B)$ holds, B is discarded. If $C_1(B)$ holds, insert B into Q_2 . Otherwise, split B into four subboxes which are inserted into Q_1 .
- **PHASE 2: BALANCING.** This phase “balances” the subdivision; a subdivision is **balanced** the size of two neighboring boxes differ by at most a factor of 2. Queue Q_2 is a min-priority queue, where the size of a box serves as its priority. While Q_2 is non-empty, remove the min-element B from Q_2 , and perform the following: For each B -neighbor B' with size more than twice the size of B , remove B' from Q_2 and split B' . Insert each child B'' of B' into Q_2 provided $C_0(B'')$ does not hold. B'' might be a new neighbor of B and B'' might be split subsequently. When, finally, every neighbor of B is at most twice the size of B , we insert B into Q_3 .
- **PHASE 3: CONSTRUCTION.** This phase constructs the graph $G = (V, E)$. Initially, the boxes in Q_3 are unmarked. While Q_3 is non-empty, remove any B from Q_3 and mark it. Now construct a set $V(B)$ of vertices.

For each B -neighbor B' , if B' is unmarked, evaluate the sign of $f(p)f(q)$ where p, q are endpoints of the segment $B \cap B'$. If $f(p)f(q) < 0$, create a vertex $v = (p + q)/2$ for the graph G . Also put v into $V(B)$. NOTE: if $f(p) = 0$ for any endpoint p , arbitrarily perturb p so that $f(p) \neq 0$. If B' is marked, retrieve the vertex v (if any) on the edge $B \cap B'$, and put v into $V(B)$. It can be shown that $|V(B)| \in \{0, 2, 4\}$. If $|V(B)| = 2$, put the edge (p, q) into G to connect the vertices in $V(B)$. If $|V(B)| = 4$, there is a simple rule to insert two non-crossing edges into G (see [19, 18]).

The output graph $G = (V, E)$ is a collection $P = P(G)$ of closed polygons or polygonal lines with endpoints in ∂B_0 .

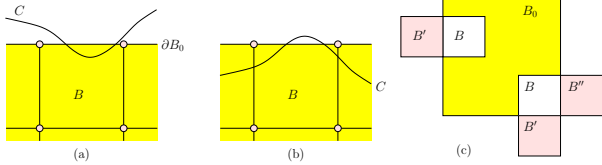


Figure 1: (a) incursion, (b) excursion, (c) boundary boxes and their complements

In what sense is P the correct output? Intuitively, P should be isotopic to $\{f = 0\} \cap B_0$. We certainly cannot handle the curve C having tangential but non-crossing intersection [26] with ∂B_0 . Assuming only transversal intersections, we still face two problems: if the curve C (locally) enters and exits ∂B_0 by visiting only one box $B \subseteq B_0$, the above algorithm would fail to detect this small component. See Figure 1(a). Conversely, the curve C might escape undetected from B_0 locally at a box B (Figure 1(b)). If we choose B_0 large enough, such errors cannot arise; but this is wasteful if we are only interested in a local region. If C has singularities, this is not even an option.

In this paper, we avoid any “largeness” assumption on B_0 . We next extend the Plantinga & Vegter algorithm to arbitrary B_0 so that a suitable correctness statement can be made about the output polygonal approximation P . In fact, B_0 need neither be a box nor be simply-connected.

3. EXTENSION OF Plantinga & Vegter

A major limitation Plantinga & Vegter that we seek to address is that small incursions and excursions, as displayed in Figure 1 might not be represented. The problem arises because Plantinga & Vegter constructs an isotopy to pull any B -excursion into B , or B -incursion into the neighboring B' ; but this can change the desired topology when B is a boundary box. Hence, we could eliminate this problem by ensuring that the curve passes through each boundary edge at most once. A test is for this can be done by ensuring that $0 \notin \Box f_x(H)$ for horizontal boundary edges H , and similarly $0 \notin \Box f_y(V)$ for vertical boundary edges V . This clearly yields a polygonal approximation P that satisfies $P \approx C \cap B_0$. This approach requires knowing the exact topology on the boundary of B_0 and resembles Snyder’s approach [23]; in higher dimensions, we need to recursively solve the problem in lower dimensions (on ∂B_0). This recursive solution can become expensive in higher dimensions.

In the spirit of the Plantinga & Vegter algorithm, we now provide an alternative solution that avoids exact boundary

topology. The idea is to slightly enlarge B_0 so that incursions/excursions can be removed by isotopy. This leads to a weaker correctness statement (Theorem 3). Compared to the exact (recursive) approach, we may split less often.

The basic idea is that, in addition to subdividing B_0 we find a slightly larger region B_0^+ which includes a collar of squares around B_0 . We ensure that at least one of predicates C_0, C_1 holds on each of these squares. Such a collar rules out excessive excursions. We then do some additional checks to ensure that any incursion is detected.

Call a box $B \subseteq B_0$ a **boundary box** if ∂B intersects ∂B_0 . Let B be a such a box. If B is not a corner box, it has a unique **complementary box** B' such that $\partial B \cap \partial B_0 = \partial B' \cap \partial B_0$ and the interiors of B' and B_0 are disjoint. Say B, B' are **partners** of each other. If B is a corner box, it determines two complementary boxes B', B'' . See Figure 1(c). Among complementary boxes that satisfy C_1 but not C_0 , we classify them into **transient** and **non-transient**, according to the sign pattern of f at their vertices, Figure 2. Intuitively, transient boxes are inconclusive for detecting incursions and need to be split. Eventually the split boxes are discarded or non-transient.

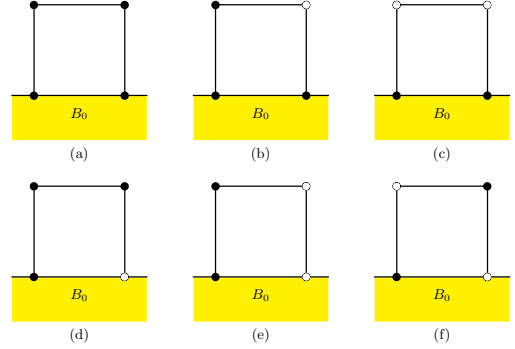


Figure 2: Classification of complementary boxes according to the sign of its vertices (up to reflection): (b), (c) are transient, but (a), (d), (e) are non-transient. Case (f) is excluded by C_1 . NOTE: The white and black vertices have opposite signs: if white is $+$ then black is $-$, and vice-versa.

We now present the Extended Plantinga & Vegter algorithm. It has 3 Phases that parallel the algorithm in Section 2. Phase i (for $i = 1, 2, 3$) works off queues Q_i and Q'_i , transferring boxes into Q_{i+1} and Q'_{i+1} .

- **PHASE 1: SUBDIVISION.** While Q_1 is non-empty, remove some B from Q_1 , and perform the following: If $C_0(B)$ holds B is discarded. If $C_1(B)$ holds, and also $C_1(B')$ or $C_0(B')$ holds for every complementary box B' of B , put B into Q_2 and place its complementary boxes B' into Q'_2 . Otherwise, split B into four subboxes which are inserted into Q_1 .

We define B_0^+ to be the union of B_0 and all of the boxes B' which were placed into Q'_2 . Now, any excursion from boundary box B through ∂B_0 must remain within some complementary $B' \subseteq B_0^+$. So excursions are accounted for using deformations within B_0^+ . But we must detect incursions.

- **PHASE 2: BALANCING with INCURSION CHECK.** The balancing of boxes in Q_2 is done as in Phase 2 of Section 2. Next, we perform an analogous while-loop on Q'_2 : While Q'_2 is non-empty, remove any B' from Q'_2 . If the partner of B' had been split during balancing, we **half-split** B' (this means we split it into its four children and put the two children that intersect ∂B_0 into Q'_2). Otherwise, we place B' into Q'_3 .

For each B' that we put into Q'_3 , if B' is transient, we do an additional incursion check. Note that B' is transient means that the two endpoints of $B' \cap \partial B_0$ have a common sign $\sigma(B') \in \{+1, -1\}$. An incursion is evidenced by discovering any point in $B' \cap \partial B_0$ whose sign is different from $\sigma(B')$. If there is an incursion, we will place another copy of B' into an incursion queue Q_I . Correctness of this process (detailed next) will be demonstrated in full paper.

Let $Q_{B'}$ be a working queue, initialized to store B' . While $Q_{B'}$ is non-empty, remove some B'' from $Q_{B'}$. If an endpoint of $B'' \cap \partial B_0$ has sign different than $\sigma(B')$, insert B' into Q_I and terminate this while-loop. If B'' is C_0 or non-transient, discard B'' . Otherwise, half-split B'' and put its two children into $Q_{B'}$. Termination of this loop is assured.

- **PHASE 3: CONSTRUCTION.** First, perform the Phase 3 of Section 2 which constructs a graph $G = (V, E)$. Next we augment this graph by adding a small incursion from each B' in Q_I into B_0 . More precisely, if B is the partner of B' , we insert two points u, v from the interior of the edge $B' \cap \partial B_0$ into the vertex set V . Also insert the edge (u, v) into the edge set E . This edge will be homotopic to a suitably defined incursion component.

Recall B_0^+ is B_0 augmented by a set of complementary boxes. The graph $G = (V, E)$ constructed by our algorithm represents a polygonal approximation $P \subseteq B_0$ comprising of polygonal paths and closed polygons.

THEOREM 3 (Weak Correctness.). *Let $C = \{p \in \mathbb{R}^2 \mid f(p) = 0\}$ be non-singular in the (original) box B_0 . Let P be the polygonal approximation from the Extended Plantinga & Vegter Algorithm. If C only meets ∂B_0 transversally, then:*

- (1) *The above procedure always halts.*
- (2) *There exists a region $\mathcal{B}$ isotopic to B_0 , which satisfies $B_0 \subset \mathcal{B} \subset B_0^+$, such that $P \approx C \cap \mathcal{B}$.*
- (3) *Every component of P contained in ∂B_0 corresponds to at least one incursion.*

Under condition (2) of this theorem, we call P a **weak isotopic approximation** to C within the region B_0 .

3.1 Extension to Nice Regions

It is essential in our applications later to extend the above refinements to non-simply connected regions. For this purpose, we define a **nice region** R_0 (relative to a square B_0) is the union of any collection of leaves taken from a quadtree rooted at B_0 . Thus, $R_0 \subseteq B_0$. To extend Theorem 3 to nice regions, we note two simple modifications:

- (a) A complementary box B' of a boundary box $B \subseteq R_0$ may intersect the interior of R_0 or other complimentary boxes.

Thus, Phase 1 must split such boundary boxes B sufficiently. (b) The region R_0 can have concave corners. We can classify the complementary boxes at concave corners into transient and non-transient ones as shown in Figure 3.

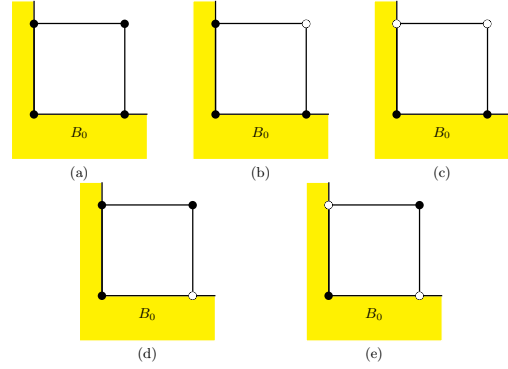


Figure 3: Classification of complementary boxes (up to reflection) at a concave corner: the case (b) is transient, and (a), (c), (d) are non-transient. Case (e) is excluded by the C_1 assumption.

4. EVALUATION BOUNDS

For any function f , define its **evaluation bound** to be

$$\text{EV}(f) := \inf\{|f(p)| : f(p) \neq 0, \nabla f(p) = 0\} \quad (2)$$

Such bounds were used in [7, 5]. From Proposition 2, we see that $\{f(p) : p \in \text{ZERO}(f), \nabla f(p) = 0\}$ is a finite set and therefore $\text{EV}(f) > 0$. However there is no explicit bound readily available. We provide such a bound:

LEMMA 4. *If $f \in \mathbb{Z}[X, Y]$ has degree d and $\|f\| < 2^L$ then $-\lg \text{EV}(f) = O(d^2(L + d))$. More precisely,*

$$\text{EV}(f)^{-1} \leq \max \left\{ \left[d^{d+5} 2^{L+2d+10} \right]^{d^2-1}, \left[d^{2d+6} 2^{3L+5d} \right]^d \right\}$$

Let f_x, f_y denote the derivatives of f . We may write

$$\text{ZERO}(f_x, f_y) = \bigcup_i U_i \cup \bigcup_j V_j$$

where U_i are 1-dim and V_j are 0-dim irreducible components. On each component U_i or V_j , one can show that the function f is constant. E.g. $f = (xy + 1)^2 - 1$, $f_x = 2(xy + 1)y$ and $f_y = 2(xy + 1)x$. Then $U_1 = \{xy + 1 = 0\}$ and $V_1 = \{(0, 0)\}$. The function f is equal to 1 on U_1 and equal to 0 on V_1 .

Let $g := \text{GCD}(f_x, f_y)$, and also

$$g_x := f_x/g, \quad g_y := f_y/g.$$

Clearly, we have

$$\text{ZERO}(f_x, f_y) = \text{ZERO}(g) \cup \text{ZERO}(g_x, g_y).$$

Since $\text{GCD}(g_x, g_y) = 1$, we conclude that $\text{ZERO}(g_x, g_y)$ has no 1-dimensional component. Conversely, the hyper-surface $\text{ZERO}(g)$ has no 0-dimensional component. This proves:

LEMMA 5.

$$\text{ZERO}(g) = \bigcup_i U_i, \quad \text{ZERO}(g_x, g_y) = \bigcup_j V_j.$$

We now provide some bounds. Let $\|f\|_k$ will denote the k -norm of f , where we use $k = 1, 2, \infty$. We just write $\|f\|$ for $\|f\|_\infty$, denoting the height of f . As parameters, we use d and L where $\deg f \leq d$ and $\|f\| < 2^L$.

We now view the ring $\mathbb{Z}[X, Y] \simeq \mathbb{Z}[X][Y] \simeq \mathbb{Z}[Y][X]$ in three alternative ways. A bivariate polynomial f in X and Y can be written as $f = f(X, Y)$, $f = f(X; Y)$ or $f = f(Y; X)$ to indicate these three views. As a member of $\mathbb{Z}[X, Y]$, the coefficients of $f(X, Y)$ are elements of $\mathbb{Z}$. But $f = f(X; Y)$ is a member of $\mathbb{Z}[Y][X]$ whose coefficients are polynomials in $\mathbb{Z}[Y]$. Similarly for $f = f(Y; X)$. The leading coefficient and degree of f are likewise affected by these views: $lc(f(X; Y)) \in \mathbb{Z}[X]$ but $lc(f(X, Y)) \in \mathbb{Z}$, $d = \deg(f(X, Y))$ is the total degree of f while $\deg(f(X; Y))$ is the largest power of Y occurring in f .

We use Mahler's basic inequality ([25, p. 351]) that if $g \in \mathbb{Z}[X, Y]$ and $g|f$ then

$$\|g(X, Y)\|_1 \leq 2^D \|f(X, Y)\|_1 \quad (3)$$

where $D = \deg(f(X; Y)) + \deg(f(Y; X))$. This implies:

$$\|g(X, Y)\|_1 \leq 4^{d-1} d^2 2^L, \quad \|g_x(X, Y)\|_1 \leq 4^{d-1} d^2 2^L. \quad (4)$$

It suffices to show the bound for $\|g\|_1$: note that $g|f_x$ and $\|f_x\|_1 \leq d^2 2^L$, $\deg(f_x(X; Y)) + \deg(f_x(Y; X)) \leq 2d - 2$. The bound then follows from (3).

Let $h(X)$ be the leading coefficient of $g(X; Y)$. Since $h(X)$ has degree $\leq d - 1$, there is an integer $x_0 \in \{0, 1, \dots, d - 1\}$ such that $h(x_0) \neq 0$. Intersect $\text{ZERO}(g)$ with the line $X = x_0$. CLAIM: This line cuts each non-vertical component U_i in a finite but non-zero number of points. In proof, let $g = \prod_i g_i$ where $\text{ZERO}(g_i) = U_i$. Setting $d_i := \deg g_i(X; Y)$, we see that the vertical components correspond to $d_i = 0$. Then $lc(g(X; Y)) = \prod_i lc(g_i(X; Y))$ and $lc(g(x_0; Y)) \neq 0$ iff for all i , $lc(g_i(x_0; Y)) \neq 0$. But $g_i(x_0; Y)$ is a polynomial of degree d_i in $\mathbb{Z}[Y]$, and has exactly d_i solutions in $\mathbb{C}$.

Write $f_0(Y) := f(x_0, Y)$ and $g_0(Y) = g(x_0, Y)$. From (4):

$$\|g_0\|_1 \leq d^d \|g(X, Y)\|_1 \leq 4^{d-1} d^{d+2} 2^L. \quad (5)$$

It is also easy to see that

$$\|f_0\| \leq d^{d+1} 2^L. \quad (6)$$

Suppose $\beta \in \text{ZERO}(g_0) \setminus \text{ZERO}(f_0)$. We want a lower bound on $|f_0(\beta)|$. For this purpose, we use an evaluation bound from [5, Theorem 13(b)]:

PROPOSITION 6 (Evaluation Bound [5]). *Let $\phi(x), \eta(x) \in \mathbb{C}[x]$ be complex polynomials of degrees m and n . Let $\beta_1, \dots, \beta_n$ be all the zeros of $\eta(x)$. Suppose there exists relatively prime $F, H \in \mathbb{Z}[x]$ such that $F = \phi\bar{\phi}, H = \eta\bar{\eta}$ for some $\bar{\phi}, \bar{\eta} \in \mathbb{C}[x]$. If the degrees of $\bar{\phi}$ and $\bar{\eta}$ are $\bar{m}$ and $\bar{n}$, then*

$$\prod_{i=1}^n |\phi(\beta_i)| \geq \frac{1}{lc(\bar{\eta})^m \cdot ((m+1)\|\phi\|)^{\bar{n}} M(\bar{\eta})^m \cdot ((\bar{m}+1)\|\bar{\phi}\|)^{n+\bar{n}} M(H)^{\bar{m}}}. \quad (7)$$

We shall choose the variables in Proposition 6 as follows:

$$\phi := f_0, \quad H = \frac{g_0}{\text{GCD}(f_0, g_0)}.$$

Moreover, let $\bar{\phi} := 1$, $\eta(X) := X - \beta$ and $\bar{\eta} := H/\eta$. Hence

$$m \leq d, \quad n \leq d, \quad \bar{m} = 0, \quad \bar{n} \leq d - 1.$$

Also

$$lc(\bar{\eta}) = lc(H) = lc(g_0) \leq \|g_0\| \leq \|g_0\|_1. \quad (8)$$

Further,

$$M(\bar{\eta}) \leq M(H) \leq \|H\|_1 \leq 2^d \cdot \|g_0\|_1.$$

Finally, an application of Proposition 6 gives

$$\begin{aligned} |f_0(\beta)|^{-1} &\leq lc(\bar{\eta})^d \cdot ((d+1)\|f_0\|)^{d-1} \cdot M(\bar{\eta})^d \\ &< [lc(\bar{\eta}) \cdot (d+1)\|f_0\| \cdot M(\bar{\eta})]^d \\ &\leq [\|g_0\|_1 \cdot (d+1)2^L \cdot 2^d \|g_0\|_1]^d \\ &\leq [d^{2d+6} 2^{3L+5d}]^d. \end{aligned} \quad (9)$$

(9) is a lower bound on $|f(p)|$ where p lies in a non-vertical component U_i . By considering $g(Y; X)$, the same bound applies for $|f(p)|$ when p lies in a vertical component U_i .

We obtain a lower bound for $f(p)$ with $p \in \text{ZERO}(g_x, g_y)$. Consider the system $\Sigma \subseteq \mathbb{Z}[X, Y]$ where

$$\Sigma = \{Z - f(X, Y), g_x(X, Y), g_y(X, Y)\}$$

The zeros $(\xi_1, \xi_2, \xi_3) = (\xi_1, y, f(x, y)) \in \mathbb{C}^3$ of Σ satisfy $\xi_3 = f(\xi_1, \xi_2)$. Since Σ is a zero dimensional system, we may apply the multivariate zero bound in [25, p. 350]. This bound says that

$$|\xi_3|^{-1} < (2^{3/2} N K)^D 2^{8(d-1)} \quad (10)$$

where $N = \binom{1+2(d-1)}{3}$, $D = d^2 - 1$ and

$$K = \max\{\sqrt{3}, \|g_x\|_2, \|g_y\|_2, \|Z - f(X, Y)\|_2\}.$$

We have $\|Z - f(X, Y)\|_2 \leq 1 + (d+1)2^L$. From (4), we see that $K \leq 4^{d-1} d^{d+2} 2^L$. Using the bound $N < 2d^3$, we obtain

$$|\xi_3|^{-1} < [4^{d+5} d^{d+5} 2^L]^{d^2-1}. \quad (11)$$

Now Lemma 4 easily follows from (9) and (11).

5. ISOLATING SINGULAR POINTS

In the rest of this paper, we assume that $f \in \mathbb{Z}[X, Y]$, and the curve $C : f = 0$ intersects ∂B_0 tangentially. We would like to use the Extended Plantinga & Vegter algorithm to compute an isotopic approximation to $\text{Zero}(f)$ when f has only isolated singularities. Since the Plantinga & Vegter algorithm does not terminate near singular points, it is necessary to isolate the singular points from the rest of B_0 .

We use the auxiliary function $F = f^2 + f_X^2 + f_Y^2$. Finding the singular points of $f = 0$ amounts to locating and isolating the zeros of this non-negative function. We use a simple mountain pass theorem [14] adapted to B_0 to ensure our algorithm isolates the zeros.

THEOREM 7. *Suppose that $F \geq 0$ on B_0 , and that $F > 0$ on ∂B_0 . Then there is path $\gamma : [0, 1] \rightarrow B_0$ connecting the two distinct roots of F which minimizes $M_\gamma = \max_{x \in [0, 1]} F(\gamma(x))$ and it either contains a point y where $\nabla F(\gamma(y)) = 0$ or a point y where $\gamma(y) \in \partial B$.*

This can be proved using path deformation and the compactness of B_0 , or it can be seen as a simple application of the topological mountain pass theorem presented in [14]. Because of this theorem, distinct zeros of F within B_0 are separated by barriers of height $\epsilon = \min(\text{EV}(F), \min F(\partial B_0))$. This leads us to the following multistep process to localize these zeros. The goal is to find a small rectangle with diameter less than some δ around each zero.

STEP 0: DETERMINING ϵ . Push B_0 into a queue of squares Q_1 . While there is an S in Q_1 remove it and evaluate $\square F(S)$. If $\square F(S) > 0$ we push S into the queue Q_∂ . If $0 \in \square F(S)$, subdivide S and push the children of S which intersect ∂B_0 into Q_1 and the others into Q_{int} . Once this terminates, $Q_1 = \emptyset$ and we have a collection of final squares Q_∂ , which contains all of ∂B_0 . For each of these S we actually can find an ϵ_S with $\square F(S) > \epsilon_S > 0$. We take ϵ to be the minimum of all these ϵ_S and $\text{EV}(F)$.

STEP 1: INITIAL SUBDIVISION. Initialize queue Q_2 with the union of Q_∂ and Q_{int} . Initialize Q_3 to be empty. Reusing the initial subdivision is only an optimization. While there is an S in Q_2 remove it and evaluate $\square F(S)$. If $\square F(S) > \epsilon/3$, discard S . Else if $F(S) < 2\epsilon/3$, place S into Q_3 . Else subdivide S and push its children into Q_2 .

Once Q_2 is empty, group the elements of Q_3 into connected regions A_i ($i \in I$). Each A_i contains at most one root, since otherwise, there would be a path connecting the roots within A_i . The value of F along this path would be bounded above by $2\epsilon/3$ contradicting the mountain pass theorem. For later reference, let C be the region $B_0 \setminus \bigcup_i A_i$. F is greater than $\epsilon/3$ on C and that $\partial B_0 \subset C$ by Step 0.

STEP 2: REFINEMENT. For each A_i ($i \in I$), initialize queue $Q_{4,i}$ with all squares $S \in A_i$. So long as neither terminating condition 1 nor 2 (below) hold, we perform the following: For each S in $Q_{4,i}$, if $0 \in \square F(S)$, subdivide S and push its children into $Q_{4,i}$. If $0 \notin \square F(S)$, discard S . We terminate when either of the following two conditions are met:

1. $Q_{4,i}$ is empty, in which case there isn't a zero in A_i .
2. A'_i , the contents of $Q_{4,i}$ satisfy all of the following:
 - (a) $\square F(S) < \epsilon/3$ for some $S \in A'_i$
 - (b) R_i , the smallest rectangle containing A'_i , lies within the region covered by the original A_i .
 - (c) The diameter of R_i is less than δ .

We claim that each R_i contains exactly one root. In Step 1, we showed that A_i contains at most one root. To see that R_i contains a root, take a point of A'_i where $F < \epsilon/3$, then follow the path of steepest descent to reach a zero of F . Because F is less than $\epsilon/3$ on this curve, the curve cannot pass through the region C to reach any other R_j or to leave B_0 . Therefore there must be a zero within A_i . It is in R_i because our conditions ensure that F is positive on $A_i \setminus R_i$.

6. DETERMINING THE DEGREE OF SINGULAR POINTS

The following standard result from [15, 16] shows that the global structure of zero sets:

PROPOSITION 8 (Zero Structure). *Let f be real analytic. Then $\text{Zero}(f)$ can be decomposed into a finite union of pieces homeomorphic to $(0, 1)$, pieces homeomorphic to S^1 , and singular points.*

Viewing $\text{Zero}(f)$ as a multigraph G , the **degree** of a singular point is its degree as a vertex of G . We now determine such degrees. Let δ_3 be a separation bound between singular points, so if p and q are two distinct singular points of $\text{Zero}(f)$, then the distance between p and q is at least δ_3 . Let δ_4 be a separation bound so that if r is a point on $\text{Zero}(f)$

such that $\nabla f(r)$ is in the same direction as the line between p and r , then the distance between p and r is at least δ_4 . If s is on $\text{Zero}(f)$ so that s is closer to the singular point p than either δ_3 or δ_4 , then by following the paths $\text{Zero}(f)$ away from s , one of the paths strictly monotonically approaches p until it reaches p and the other path locally strictly monotonically recedes from p . See [26] for explicit bounds on δ_3 as a function of degree and height of $f(X, Y)$. We can similarly derive explicit bounds on δ_4 .

To find the degree of a singular point, assume that we have two boxes $B_1 \supsetneq B_2$ where the diameter of B_1 is less than both δ_3 and δ_4 , B_2 contains a singular point of f and there is some radius $r > 0$ such that a circle of radius r centered at any point p inside B_2 must lie entirely within the annulus $B_1 \setminus B_2$. See Figure 4. Furthermore, to apply our extended Plantinga & Vegter algorithm of Section 4, we can ensure that $B_1 \setminus B_2$ is a nice region.

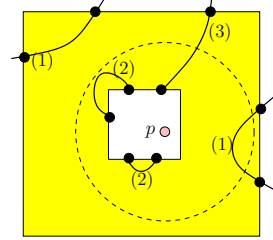


Figure 4: Annular region $B_1 \setminus B_2$ with singularity p and the three types (1), (2), (3) of components.

Now, there are 3 types of components in $\text{Zero}(f) \cap (B_1 \setminus \text{int}(B_2))$: (1) images of $[0, 1]$ both of whose endpoints are on ∂B_1 , (2) images of $[0, 1]$ both of whose endpoints are on ∂B_2 , and (3) images of $[0, 1]$ with one endpoint on each of ∂B_1 and ∂B_2 . These three types are illustrated in Figure 4. Let s be a point on any of these components, then traveling along $\text{Zero}(f)$ in one direction must lead to the singular point and the other direction must leave the neighborhood (be further than $\min\{\delta_3, \delta_4\}$) of the singular point.

LEMMA 9. *The degree of the singular point in B_2 is the number of components of type 3.*

Any component accumulating on a singular point exits the neighborhood of the singular point and the only way to leave the neighborhood is by way of a type 3 component.

7. OVERALL ALGORITHM

We now put all the above elements together to find a weak isotopic approximation to the algebraic curve $C : f = 0$ within a nice region R_0 where $f(X, Y) \in \mathbb{Z}[X, Y]$ has only isolated singularities. For simplicity, we assume that ∂R_0 intersects the curve $C : f = 0$ transversally. We first find the singularities of the curve $C : f = 0$ in R_0 . Using the technique of Section 5, we can isolate the singularities p_i ($i = 1, 2, \dots$) into pairwise disjoint boxes B_i . We may assume the size of the B_i 's is at most $\min\{\delta_3, \delta_4\}/6$. Let B'_i be the box of size 5 times the size of B_i , and concentric with B_i ; we may further assume $B'_i \subseteq R_0$. Now we proceed to run the extended Plantinga & Vegter algorithm on the nice region $R^* := R_0 \setminus \bigcup_i B_i$, yielding a polygonal approximation P . We directly incorporate the technique of Section 6 into

the following argument. If p_i is the singular point in B_i , then the degree of p_i is equal to the number of type 3 components in $P \cap (B'_i \setminus B_i)$. We directly connect these components directly to p_i , and discard any type 2 components. This produces the desired isotopic approximation.

Remarks: (1) We could avoid the assumption that C and ∂R_0 intersect transversally provided R_0 is a nice region relative to a box B_0 whose corners have integer or algebraic coordinates. Using the geometric separation bounds in [26] we can detect an actual transversal intersection.

(2) We have not discussed ε -approximation because this is relatively easy to achieve in the Plantinga & Vegter approach. We only have to make sure that each subdivision box that contains a portion of the polygonal approximation P has size at most $\varepsilon/4$.

8. CONCLUSION

This paper presents the first complete numerical subdivision algorithm for meshing an implicit algebraic curve that has only isolated singularities. This solves an open problem in the exact numerical approaches to meshing in 2-D [2, p. 187]. We pose three challenges:

- (a) An worst case complexity bound for our procedure is possible. But this may not be the best way to measure adaptive algorithms. We would like to provide are adaptive bounds, similar to the integral analysis in [5].
- (b) In 3-D, a square-free integer polynomial $f(X, Y, Z)$ could have a 1-dimensional singularities. We pose the problem of designing a purely numerical subdivision algorithm to handle 1-dimensional singularities.
- (c) The practical implementation of an adaptive algorithm handling singularities, even based on our outline, must handle many important details. Computational experience is invaluable for future research into singularity computation.

9. REFERENCES

- [1] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*. Algorithms and Computation in Mathematics. Springer, 2003.
- [2] J.-D. Boissonnat, D. Cohen-Steiner, B. Mourrain, G. Rote, and G. Vegter. Meshing of surfaces. In J.-D. Boissonnat and M. Teillaud, editors, *Effective Computational Geometry for Curves and Surfaces*. Springer, 2006. Chapter 5.
- [3] J.-D. Boissonnat, D. Cohen-Steiner, and G. Vegter. Isotopic implicit surfaces meshing. In *ACM Symp. on Theory of Computing*, pages 301–309, 2004.
- [4] J.-D. Boissonnat and S. Oudot. Provably good sampling and meshing of Lipschitz surfaces. In *Proc. 22nd ACM Symp. on Comp. Geometry*, pages 337–346, 2006. Sedona, Arizona.
- [5] M. Burr, F. Krahmer, and C. Yap. Integral analysis of evaluation-based real root isolation, Mar. 2008. Submitted, FOCS 2008.
- [6] M. Burr, V. Sharma, and C. Yap. Evaluation-based root isolation, Nov. 2007. In preparation.
- [7] J.-S. Cheng, X.-S. Gao, and C. K. Yap. Complete numerical isolation of real zeros in general triangular systems. In *Proc. ISSAC'07*, pages 92–99, 2007.
- [8] S.-W. Cheng, T. Dey, E. Ramos, and T. Ray. Sampling and meshing a surface with guaranteed topology and geometry. In *Proc. 20th ACM Symp. on Comp. Geometry*, pages 280–289, 2004.
- [9] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer-Verlag, New York, 1992.
- [10] Z. Du and C. Yap. Uniform complexity of approximating hypergeometric functions with absolute error. In S. il Pae and H. Park, editors, *Proc. 7th Asian Symp. on Computer Math. (ASCM 2005)*, pages 246–249, 2006.
- [11] J. Harris. *Algebraic Geometry*. Springer-Verlag, 1992.
- [12] R. Hartshorne. *Algebraic Geometry*. Springer-Verlag, New York, 1977.
- [13] H. Hong. An efficient method for analyzing the topology of plane real algebraic curves. *Mathematics and Computers in Simulation*, 42:571–582, 1996.
- [14] Y. Jabri'. *The Mountain Pass Theorem: Variants, Generalizations and Some Applications*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2003.
- [15] S. G. Krantz and H. R. Parks. *A primer of real analytic functions*. Birkhäuser Verlag, Basel, 1992.
- [16] S. Łojasiewicz. *Introduction to complex analytic geometry*. Birkhäuser Verlag, Basel, 1991. Translated from the Polish by Maciej Klimek.
- [17] B. Mourrain and J.-P. Tékourt. Isotopic meshing of a real algebraic surface. Technical Report RR-5508, INRIA, Sophia-Antipolis, France, Feb. 2005. Also, electronic proceedings, MEGA 2005.
- [18] S. Plantinga. *Certified Algorithms for Implicit Surfaces*. Ph.D. thesis, Groningen University, Institute for Mathematics and Computing Science, Groningen, Netherlands, Dec. 2006.
- [19] S. Plantinga and G. Vegter. Isotopic approximation of implicit curves and surfaces. In *Proc. Eurographics Symposium on Geometry Processing*, pages 245–254, New York, 2004. ACM Press.
- [20] E. Schoemer and N. Wolpert. An exact and efficient approach for computing a cell in an arrangement of quadrics. *Comput. Geometry: Theory and Appl.*, 33:65–97, 2006.
- [21] R. Seidel and N. Wolpert. On the exact computation of the topology of real algebraic curves. In *Proc. 21st ACM Symp. on Comp. Geometry*, pages 107–116, 2005. Pisa, Italy.
- [22] J. M. Snyder. *Generative Modeling for Computer Graphics and CAD: Symbolic Shape Design using Interval Analysis*. Academic Press, 1992.
- [23] J. M. Snyder. Interval analysis for computer graphics. *SIGGRAPH Comput. Graphics*, 26(2):121–130, 1992.
- [24] B. T. Stander and J. C. Hart. Guaranteeing the topology of an implicit surface polygonalization for interactive meshing. In *Proc. 24th Computer Graphics and Interactive Techniques*, pages 279–286, 1997.
- [25] C. K. Yap. *Fundamental Problems of Algorithmic Algebra*. Oxford University Press, 2000.
- [26] C. K. Yap. Complete subdivision algorithms, I: Intersection of Bezier curves. In *22nd ACM Symp. on Comp. Geometry*, pages 217–226, July 2006.

A Pommaret Division Algorithm for Computing Gröbner Bases in Boolean Rings

Vladimir P. Gerdt, Mikhail V. Zinin

Laboratory of Information Technologies, Joint Institute for Nuclear Research
141980 Dubna, Russia
gerdt@jinr.ru, mzinin@gmail.com

ABSTRACT

In this paper an involutive algorithm for construction of Gröbner bases in Boolean rings is presented. The algorithm exploits the Pommaret monomial division as an involutive division. In distinction to other approaches and due to special properties of Pommaret division the algorithm allows to perform the Gröbner basis computation directly in a Boolean ring which can be defined as the quotient ring $\mathbb{F}_2[x_1, \dots, x_n] / \langle x_1^2 + x_1, \dots, x_n^2 + x_n \rangle$. Some related cardinality bounds for Pommaret and Gröbner bases are derived. Efficiency of our first implementation of the algorithm is illustrated by a number of serial benchmarks.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms

General Terms

Algorithms

Keywords

Boolean ring, Gröbner basis, Involutive algorithm, Pommaret division

1. INTRODUCTION

In specific cases Gröbner bases as an algorithmic tool for Boolean polynomials, where the only coefficients and exponents are 0 and 1, have revealed their practicality for solving real-world problems. Among the most impressive applications there is breaking the first HFE (Hidden Fields Equations) challenge in the public key cryptography done at first [1] on the basis of algorithm F_5 [2] and then [3] on the basis of algorithm F_4 [4]. To apply the technique of Boolean Gröbner bases to a greater degree one has to adopt the most efficient algorithms for constructing Gröbner bases in commutative algebra to multilinear structure of Boolean

polynomials. A striking example of such adaptation is C++ library **PolyBoRi** [5] implementing a specialized version of *slimgb*, a modified Buchberger algorithm, together with new data structures for efficient computation of Boolean Gröbner bases. Recent experimentation with **PolyBoRi** presented in [6] demonstrates its competitiveness with the best modern SAT (Boolean Satisfiability)-solvers. In combination with preprocessing (transformation and simplification) of CNF (Conjunctive Normal Form) - this becomes applicable to problems of industrial category [7].

One more promising application of Boolean Gröbner bases may be simulation of quantum computation on a classical computer. As shown in [8], the unitary matrix (i.e. quantum computation) generated by a quantum circuit constructed of the Toffoli and Hadamard gates, which form a universal gate basis, can be computed by counting the number of common roots in F_2 for the system of Boolean polynomials uniquely determined by the circuit. In [9] a **Mathematica** package was designed that allows a user to input a quantum circuit and generate automatically the circuit Boolean polynomials. The circuit $2^n \times 2^n$ unitary matrix for n qubits can be straightforwardly computed by the standard linear algebra methods. In practice, however, modern quantum simulators may deal with at most 20–25 qubits because of computer memory limitation. The system of Boolean polynomials for a n -qubit quantum circuit contains $n + 1$ quadratic and cubic polynomials whose number of variables is equal to the number of Hadamard gates in the circuit. Single-exponential complexity of the Gröbner basis computation in Boolean rings [10] and ability to compute Boolean Gröbner bases for polynomial systems with 80 [1, 3] and even with more than 100 variables [6] gives a real hope that Gröbner bases, at least in special cases, may provide a leap forward in simulation of quantum computation. Moreover, quite moderate experimental memory consumption with **PolyBoRi** demonstrated by benchmarks in [6] together with the statement made in [11] that Gröbner basis computation in Boolean rings is in the PSPACE complexity class strengthens this hope.

In the given paper we present a specialization to systems of Boolean polynomials of the involutive division algorithm designed in [12] and further improved in [13, 14]. The algorithm exploits the particular involutive division called in [12] after Pommaret, and due to its peculiarities avoids the use of field polynomials as elements in the generating set (cf. Theorem 63 in [6]). This makes possible, among other things, to decrease number of intermediate prolongations and reductions and to use the maximal possible vectorization of the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

monomial data structures. We compare our very first implementation of the algorithm in C++ with **Singular**, **Fgb** and **PolyBoRi** for the degree-reverse-lexicographical order and with **Singular** and **PolyBoRi** for the pure lexicographical order by using some serial benchmarks.

2. PRELIMINARIES

2.1 Boolean Ring

Let $\mathbf{x} := \{x_1, \dots, x_n\}$ be a set of variables, $\mathbb{F}_2[\mathbf{x}]$ be the ring of commutative polynomials over the finite field $\mathbb{F}_2$ of two elements $\{0, 1\}$ and $\mathcal{I} \subset \mathbb{F}_2[\mathbf{x}]$ an ideal. When polynomial set $P = \{p_1, \dots, p_k\}$ is a basis of $\mathcal{I}$ we shall write $\mathcal{I} = \langle P \rangle$. A *Boolean ring* is a ring of *Boolean functions* of n variables, i.e. mappings from $\{0, 1\}^n$ to $\{0, 1\}$. This ring is a commutative ring which can be regarded as the quotient ring

$$\mathbb{B}[\mathbf{x}] := \mathbb{F}_2[\mathbf{x}] / \langle x_1^2 + x_1, \dots, x_n^2 + x_n \rangle. \quad (1)$$

Multiplication in $\mathbb{B}[\mathbf{x}]$ is *idempotent* and addition is *nilpotent*

$$\forall b \in \mathbb{B}[\mathbf{x}] : b^2 = b, b + b = 0. \quad (2)$$

Elements in $\mathbb{B}[\mathbf{x}]$ are *Boolean polynomials* and can be represented as finite sums

$$\sum_j \prod_{x \in \Omega_j \subseteq \mathbf{x}} x \quad (3)$$

of *Boolean monomials*. Each monomial is a conjunction. If set Ω is empty, then the corresponding monomial is the unity Boolean function 1. Polynomials in $\mathbb{F}_2[\mathbf{x}]$ of the form (3) we shall also call *Boolean*. The binomials $\varphi_i := x_i^2 + x_i \in \mathbb{F}_2[\mathbf{x}]$ ($1 \leq i \leq n$) are called *field polynomials* [6, 10] since equation $\varphi_i = 0$ provides the restriction $x_i \in \mathbb{F}_2$. Hereafter, we shall use the canonical ring homomorphism induced by (1)

$$\phi : \mathbb{F}_2[\mathbf{x}] \rightarrow \mathbb{B}[\mathbf{x}]. \quad (4)$$

2.2 Pommaret Division and Pommaret Basis

Let $\succ$ be an admissible monomial order such that

$$x_1 \succ x_2 \cdots \succ x_n.$$

The following definitions [12] and notations are related to both rings $\mathbb{F}_2[\mathbf{x}]$ and $\mathbb{B}[\mathbf{x}]$. We shall denote the leading monomial of p by $\text{lm}(p)$ and for a set P of polynomials its leading monomial set will be denoted by $\text{lm}(P)$. The degree in x_i of a monomial m will be denoted by $\deg_i(m)$.

Definition 1. For a polynomial p such that

$$\text{lm}(p) = x_1^{d_1} \cdots x_{k-1}^{d_{k-1}} x_k^{d_k}, \quad 1 \leq k \leq n, \quad d_k > 0$$

the variables in $N\mathcal{M}_{\mathcal{P}}(p) := \{x_1, \dots, x_{k-1}\}$ are *Pommaret* or $\mathcal{P}$ -*nonmultiplicative*, and the remaining variables in set $\mathbf{x}$ are $\mathcal{P}$ -*multiplicative*. For $p = 1$ all the variables are $(\mathcal{P})$ -multiplicative. A monomial v is *Pommaret* or $\mathcal{P}$ -*divisor* of a monomial u (denotation: $v \mid_{\mathcal{P}} u$), and respectively u is $\mathcal{P}$ -*multiple* of v , if $u = v \cdot w$ and all variables occurring in w in nonzero degree are $\mathcal{P}$ -multiplicative for v .

Pommaret division is *involution* [12] and satisfies

$$\forall u, v, w : u \mid_{\mathcal{P}} w \wedge v \mid_{\mathcal{P}} w \implies u \mid_{\mathcal{P}} v \vee v \mid_{\mathcal{P}} u. \quad (5)$$

Given two polynomials $p, g \in \mathbb{F}_2[\mathbf{x}]$ (or $p, g \in \mathbb{B}[\mathbf{x}]$) such that p has a monomial u satisfying $\text{lm}(g) \mid_{\mathcal{P}} u$, polynomial p is $\mathcal{P}$ -*reducible modulo* g and admits the *elementary $\mathcal{P}$ -reduction*

$$p \xrightarrow[g]{} p + g \frac{u}{\text{lm}(g)}. \quad (6)$$

Otherwise, p is $\mathcal{P}$ -*irreducible modulo* g . If F is a polynomial set, and a polynomial p is $\mathcal{P}$ -irreducible modulo each $f \in F$, then p is in the $\mathcal{P}$ -*normal form modulo* F (denotation: $p = NF_{\mathcal{P}}(p, F)$). The reduction (6) will be called *Boolean* if both polynomials p and q are Boolean.

A polynomial set F is $\mathcal{P}$ -*autoreduced* if

$$\forall f \in F : f = NF_{\mathcal{P}}(f, F \setminus \{f\}).$$

From condition (5) it follows that if a polynomial set F is $\mathcal{P}$ -autoreduced, then a monomial u may have at most one $\mathcal{P}$ -divisor in $\text{lm}(F)$. This implies uniqueness of the chain of elementary $\mathcal{P}$ -reductions (6) and thus uniqueness of $NF_{\mathcal{P}}(p, F)$ for any polynomial p .

A $\mathcal{P}$ -autoreduced set P is a *Pommaret basis* of $\langle P \rangle$ if

$$\forall p \in P, \forall x \in N\mathcal{M}_{\mathcal{P}}(p) : NF_{\mathcal{P}}(p \cdot x, P) = 0. \quad (7)$$

The product $p \cdot x$ in (7) is called $(\mathcal{P})$ -*nonmultiplicative prolongation* of p with respect to x .

The next proposition shows that mapping (4) preserves the $\mathcal{P}$ -divisibility relation $\mid_{\mathcal{P}}$. This invariance property of Pommaret division singles it out of all other known involutive divisions (for example, those studied in [12, 14]) and makes it applicable to the direct computation of Gröbner bases in Boolean rings as we show in the next section.

PROPOSITION 1. *The monomial separation of variables into $\mathcal{P}$ -multiplicative and nonmultiplicative and thus their $\mathcal{P}$ -divisibility is preserved by homomorphism (4).*

PROOF. Follows immediately from the obvious fact that both monomials $u \in \mathbb{F}_2[\mathbf{x}]$ and $\phi(u) \in \mathbb{B}[\mathbf{x}]$ have the same index of the smallest (w.r.t. $\succ$) variable entering in a positive degree. $\square$

3. THE ALGORITHM

3.1 Computation in $\mathbb{F}_2[\mathbf{x}]$

Let $B \subset \mathbb{F}_2[\mathbf{x}] \setminus \{0\}$ be a finite set of Boolean polynomials of the form (3), $\Phi := \cup_{i=1}^n \{x_i^2 + x_i\}$ and $F := B \cup \Phi$.

The below algorithm **PommaretBasis** computes a Pommaret basis of the ideal $\langle F \rangle \subset \mathbb{F}_2[\mathbf{x}]$. It is the simplest version of the general algorithm **InvolutionBasis** [13, 14] adjusted to Pommaret division.

Correctness. First we note that $\langle F \rangle = \langle P, Q \rangle$ is invariant of the main loop 3-16 where P contains a part of the intermediate basis whereas Q contains all $\mathcal{P}$ -nonmultiplicative prolongations of elements in P and (possibly) the remaining part of intermediate basis. The subalgorithm **HeadNF** invoked at step 7 computes the *head $\mathcal{P}$ -normal form* of q modulo P by doing elementary $\mathcal{P}$ -reductions (6) of the leading monomial in q modulo elements in P . The subalgorithm **AutoreduceTails** of step 17 performs the *tail $\mathcal{P}$ -autoreduction* of polynomials in P when reductions (6) are applied to all tail monomials of every polynomial in P modulo the rest polynomials in P . The loop 3-16 terminates when $Q = \emptyset$. At the initialization step 2 set P is trivially $\mathcal{P}$ -autoreduced. Polynomial h computed at step

7 is $\mathcal{P}$ -head reduced modulo P . The selection strategy of steps 1 and 6 together with the **for**-loop apparently yields that $\text{lm}(P)$ remains $\mathcal{P}$ -reduced after extension P with h at step 14. Thereby, after the tail autoreduction of step 17 the conditions (7) are fulfilled for P , and the output polynomial set P is a Pommaret basis of $\langle F \rangle$.

Algorithm: PommaretBasis ($F, \succ$)

```

1: choose  $f \in F$  with lowest  $\text{lm}(f)$  w.r.t.  $\succ$ 
2:  $P := \{f\}$ ;  $Q := F \setminus \{f\} \cup \{f \cdot x \mid x \in NM_{\mathcal{P}}(f)\}$ 
3: do
4:    $h := 0$ 
5:   while  $Q \neq \emptyset$  and  $h = 0$  do
6:     choose  $q \in Q$  with lowest  $\text{lm}(q)$  w.r.t.  $\succ$ 
7:      $Q := Q \setminus \{q\}$ ;  $h := \text{HeadNF}_{\mathcal{P}}(q, P)$ 
8:   od
9:   if  $h \neq 0$  then
10:    for all  $\{p \in P \mid \text{lm}(h) \mid_{\mathcal{P}} \text{lm}(p)\}$  do
11:       $P := P \setminus \{p\}$ ;
12:       $Q := Q \cup \{p\} \setminus \{p \cdot x \mid x \in NM_{\mathcal{P}}(p)\}$ 
13:    od
14:     $P := P \cup \{h\}$ ;  $Q := Q \cup \{h \cdot x \mid x \in NM_{\mathcal{P}}(h)\}$ 
15:  fi
16: od while  $Q \neq \emptyset$ 
17:  $P := \text{AutoreduceTails}_{\mathcal{P}}(P)$ 
18: return  $P$ 

```

Termination. Due to the presence of field polynomials in the input set F ideal $\langle F \rangle$ is zero-dimensional. However the Pommaret division is not Noetherian [12], and though any zero-dimensional polynomial ideal has a finite Pommaret basis [15], even in this case the Pommaret division algorithms may not terminate for a degree incompatible monomial order (see Example 8.2 in the first paper of reference [12]). Hence, one has to prove termination of the above algorithm for the input polynomial set $B \cup \Phi \subset \mathbb{F}_2[\mathbf{x}]$.

Subalgorithms **HeadNF** and **AutoreduceTails** terminate since they perform $\mathcal{P}$ -reductions which form a subset of the conventional Gröbner reductions [16, 17]. The **while**- and **for**-loops are obviously finite. If $\text{lm}(h) \neq \text{lm}(q)$, i.e. $\text{lm}(h) \prec \text{lm}(q)$, holds after step 7 and $h \neq 0$, then before insertion of h into P the condition $\text{lm}(h) \notin \langle \text{lm}(P) \rangle$ must hold. Otherwise there would be an element $p \in P$ such that $\text{lm}(p)$ is a proper divisor of $\text{lm}(h)$ and $\text{lm}(h) = \text{lm}(p) \cdot v \cdot w$ where monomial $v \neq 1$ contains variables from $NM_{\mathcal{P}}(p)$ whereas w does not contain them. In this case $p \cdot v$ must have been inserted in P due to the selection strategy of step 6 that makes $\text{lm}(h)$ $\mathcal{P}$ -reducible modulo p , i.e. $\text{lm}(p) \cdot v \mid_{\mathcal{P}} \text{lm}(h)$, in contradiction to computation of step 7. Dickson's lemma [17] ensures that extension of P with elements whose leading monomials are not in $\langle \text{lm}(P) \rangle$ can be performed only finitely many times. This implies finiteness of displacements of elements from P to Q done at step 12.

It remains to show that the main loop 3-16 also cannot run infinitely many times. If this would be the case the intermediate basis P could not be bounded. Recall that the input set F consists of multilinear Boolean polynomials and quadratic field polynomials. In this case the following lemma is applicable.

LEMMA 1. *Let $q \in Q$ be the nonmultiplicative prolongation $p \cdot x_i$ where p is a Boolean polynomial and $\text{lm}(q)$*

*is quadratic in x_i . If q is selected at step 6 of algorithm **PommaretBasis** applied to set $B \cup \Phi$, then a nonzero head $\mathcal{P}$ -normal form h of q modulo P computed at step 7 of the algorithm has Boolean $\text{lm}(h)$.*

PROOF. It is clear that q has the following structure

$$\text{lm}(q) = u \cdot x_i^2 \cdot v \wedge \begin{cases} i = 1 & : u = 1 \\ i > 1 & : \deg_j(u) \leq 1 \ (1 \leq j \leq i-1) \end{cases}.$$

Then, by the selection strategy of step 6, the corresponding field polynomial $\varphi_i = x_i^2 + x_i$ had to be processed in the **while**-loop before q . This implies one of two alternatives:

(i) there is a subset $S \subset P$ containing φ_i and its $\mathcal{P}$ -nonmultiplicative prolongations such that

$$\exists s \in S : s = u \cdot \varphi_i;$$

(ii) $\varphi_i \notin P \wedge \exists f \in P : f = u \cdot x_i$.

In case (ii) it is clear, however, that $\text{lm}(p)$ is $\mathcal{P}$ -multiple of $u \cdot x_i$ what is impossible since $\text{lm}(P)$ is $\mathcal{P}$ -autoreduced. Thus the case (i) takes place, and the (unique) elementary reduction step (6) of $\text{lm}(q)$ modulo s decreases its degree in x_i to 1. If the reduction leads to the new leading monomial quadratic in x_i , then this monomials is again (uniquely) reducible to the monomial linear in x_i , etc. $\square$

COROLLARY 1. *The output Pommaret basis of algorithm **PommaretBasis** applied to $B \cup \Phi$ may include only Boolean polynomials and prolongations of field polynomials.*

PROOF. Subalgorithm **AutoreduceTails** applied at step 17 to the polynomial set whose leading monomials either are Boolean or nonmultiplicative prolongations of x^2 ($x \in \mathbf{x}$), by the same arguments as used in the proof of Lemma 1, reduces the tail of every polynomial with the Boolean leading monomial to the Boolean form. $\square$

By Lemma 1, $h \neq 0$ computed at step 7 for a $\mathcal{P}$ -nonmultiplicative prolongation of a Boolean polynomial has a Boolean leading monomial. As to the field polynomials φ_k ($1 \leq k \leq n$) themselves and their $\mathcal{P}$ -nonmultiplicative prolongations, x_k is $\mathcal{P}$ -multiplicative for every ϕ_k and its prolongations. Therefore, polynomials of degree 3 in x_k never occur in Q and hence in P . This completes the proof of termination because the number of elements in P is bounded.

PROPOSITION 2. *A Pommaret basis P_{Φ} of the polynomial ideal $\langle \Phi \rangle \in \mathbb{F}_2[\mathbf{x}]$ generated by the field polynomials is given by*

$$P_{\Phi} = \cup_{i=1}^n \cup_{\Omega \subseteq \mathbf{x} \setminus \{x_i, \dots, x_n\}} \left\{ \varphi_i \prod_{x \in \Omega} x \right\}. \quad (8)$$

PROOF. A Pommaret basis is uniquely defined by an ideal and monomial order, and for $\langle F \rangle \subset \mathbb{F}_2[\mathbf{x}]$ the Pommaret basis is a superset of the reduced Gröbner basis of $\langle F \rangle$ [12]. Since elements (field polynomials) in Φ have co-prime leading monomials, the Gröbner basis is Φ in accordance to the Buchberger criterion [16]. Thereby, the Pommaret basis must contain set Φ and all mutually $\mathcal{P}$ -irreducible nonmultiplicative prolongations of its elements which are multilinear in the prolonged variables. Apparently, the prolongations in (8) related to different field polynomials φ_i and φ_j ($i \neq j$) are mutually $\mathcal{P}$ -irreducible. It is easy to see that mutual irreducibility holds also for the elements in (8) which are prolongations of the same field

polynomial. In addition, set (8) contains all such prolongations. Furthermore, the repeated nonmultiplicative prolongation with respect to $x \in \mathbf{x}$ of a polynomial in (8) linear (and nonconstant) in x is obviously $\mathcal{P}$ -reducible modulo set (8) to the polynomial itself. $\square$

COLORRARY 2. *Cardinality of a Pommaret basis for an ideal in $\mathbb{F}_2[\mathbf{x}]$ generated by a set of Boolean polynomials in n variables extended with the set Φ of field polynomials is bounded by $2^n - 1$.*

PROOF. First we show that cardinality of the set P_Φ in (8) is exactly $2^n - 1$. The field polynomial φ_1 enters in P_Φ without prolongations, and the number of elements in the subset of P_Φ related to the field polynomial φ_i ($i > 1$) is the total number of Boolean monomials in variables $\{x_1, \dots, x_{i-1}\}$, i.e. 2^{i-1} . Hence, $|P_\Phi| = \sum_{i=1}^n 2^{i-1} = 2^n - 1$.

The set $\text{lm}(P_\Phi)$ of leading monomials in P_Φ reads

$$\{x_1^{i_1} \cdots x_{k-1}^{i_{k-1}} x_k^2 \mid k = 1, \dots, n; i_j = 0, 1; j = 1, \dots, k-1\}.$$

Now, to prove that this set has the maximal possible cardinality among the Pommaret bases of initial ideals generated by $B \cup \Phi$ where B is a set of Boolean polynomials, it suffices to show that insertion of arbitrary $\mathcal{P}$ -autoreduced set of Boolean monomials into $\text{lm}(P_\Phi)$ followed by $\mathcal{P}$ -autoreduction never causes an increase of the cardinality. Let U be such a set of Boolean monomials. If $U = \{1\}$, then the Pommaret basis of $U \cup \text{lm}(P_\Phi)$ is also $\{1\}$. Otherwise, every monomial in $u \in U$ $\mathcal{P}$ -reduces to zero¹ at least one element in $p \in \text{lm}(P_\Phi)$, namely, that whose image under the ring homomorphism (4) is u . As this takes place, the number of $\mathcal{P}$ -irreducible Boolean monomials cannot exceed $2^n - 1$. $\square$

Below we shall keep denotation P_Φ for the Pommaret basis of $\langle B \rangle$ and use the following definition (cf. [6]).

Definition 2. Let $B \subset \mathbb{F}_2[\mathbf{x}]$ be a finite set of Boolean polynomials, G be a Gröbner basis of $\langle B, \Phi \rangle \subset \mathbb{F}_2[\mathbf{x}]$, and $G_\mathbb{B}$ be the subset $G \setminus P_\Phi$ of Boolean polynomials in G . Then the set $\phi(G_\mathbb{B}) \subset \mathbb{B}[\mathbf{x}]$ will be called a (*Boolean*) *Gröbner basis* of $\langle B \rangle \subset \mathbb{B}[\mathbf{x}]$. Respectively, given a Pommaret basis P of $\langle B, \Phi \rangle \subset \mathbb{F}_2[\mathbf{x}]$, the image of set $P \setminus P_\Phi$ under the canonical homomorphism (4) will be called a (*Boolean*) *Pommaret basis* of $\langle B \rangle \subset \mathbb{B}[\mathbf{x}]$.

THEOREM 1. *Given a set B of Boolean polynomials, ideal $\langle B \rangle$ in the Boolean ring $\mathbb{B}[\mathbf{x}]$ has unique Pommaret basis.*

PROOF. Follows immediately from Definition 2 and uniqueness of Pommaret bases [12] for ideals $\langle B, \Phi \rangle$ and $\langle \Phi \rangle$. $\square$

3.2 Computation in $\mathbb{B}[\mathbf{x}]$

Now we are going to show that the image of a Boolean Pommaret basis P (as it defined above) under the canonical isomorphism (4) is exactly the output of algorithm **PommaretBasis** when it takes the Boolean polynomial set $B \subset \mathbb{B}[\mathbf{x}]$ as an input and performs operations directly in the Boolean ring. Accordingly, its subset $G \subseteq P$ obtained by the conventional autoreduction of P is exactly the image of the reduced Gröbner basis of $\langle B \rangle$ as defined above. Thereby,

¹Actually u provides $\mathcal{P}$ -reduction to zero of the whole polynomial in P_Φ .

Table 1: Computation of Pommaret basis for $B = \{xy + y + 1\}$, $x \succ y$

Steps of algorithm	Sets P and Q		
	elements in P	NMP	Q
initialization	$xy + y + 1$	x	$\{x\}$
iteration	x	—	$\{xy + y + 1\}$
	x	—	
	$y + 1$	x	$\{xy + x\}$
	x	—	
	$y + 1$	x	$\{\}$

unicity of Pommaret bases, by Theorem 1, is closely interconnected with unicity of reduced Boolean Gröbner bases.

Consider algorithm **PommaretBasis** applied to a set $F \subset \mathbb{B}[\mathbf{x}]$ of polynomials in the Boolean ring where addition and multiplication satisfy (2) and prove it correctness and termination.

Correctness. In accordance to Proposition 1, the set of $\mathcal{P}$ -nonmultiplicative variables for a Boolean polynomial $p \in \mathbb{F}_2[\mathbf{x}]$ is the same as that for its canonical image $\phi(p) \in \mathbb{B}[\mathbf{x}]$. Apparently, set P remains head $\mathcal{P}$ -autoreduced after every run of the main loop 3-16. If the loop is terminated, then the output basis is $\mathcal{P}$ -autoreduced. We have to show that, given an input set $B \in \mathbb{B}[\mathbf{x}]$, the output P of the algorithm is exactly the Boolean subset in the output of the same algorithm for $B \cup \Phi \in \mathbb{F}_2[\mathbf{x}]$. By Lemma 1 and Corollary 1, the head and tail $\mathcal{P}$ -reduction of a nonmultiplicative prolongation p of a Boolean polynomial, when it is processed, effectively applies the Boolean rule $x^2 \rightarrow x$ if *reductor* g in (6) is an element in P_Φ . This shows that reductions modulo elements in P_Φ serve only to simulate Boolean multiplication in $\mathbb{F}_2[\mathbf{x}]$ and do not contribute to detection of those elements in the ideal $\langle B, \Phi \rangle$ whose leading monomials are not in $\text{lm}(P)$. All other $\mathcal{P}$ -reductions which are performed in the course of algorithm **PommaretBasis** applied to the input set $B \cup \Phi \in \mathbb{F}_2[\mathbf{x}]$ are Boolean. Moreover, they are in one-to-one correspondence with the reductions which are performed in the Boolean ring $\mathbb{B}[\mathbf{x}]$ when the algorithm is applied to B . In so doing, the result of every elementary Boolean reduction in $\mathbb{B}[\mathbf{x}]$ is the homomorphic image under (4) of the reduction result in $\mathbb{F}_2[\mathbf{x}]$.

Termination. Follows by the observation that operations in $\mathbb{B}[\mathbf{x}]$ mirror a subset of operations in $\mathbb{F}_2[\mathbf{x}]$.

We illustrate the operation of algorithm **PommaretBasis** by a very simple bivariate example $B := \{xy + y + 1\} \subset \mathbb{B}[x, y]$ for a monomial order satisfying $x \succ y$. The intermediate polynomials in set P , their nonmultiplicative variables and the set Q of the corresponding prolongations are shown, respectively, in columns 2, 3 and 4 of Table 1. At the initialization step (line 6 in the algorithm) $P = \{xy + y + 1\}$ and the set Q is given by $\{x \cdot (xy + y + 1) = x\}$. Then the values of P and Q after every run of the main loop 3-16 are shown in the next rows of the table.

The output Pommaret basis $\{x, y + 1\}$ for this example coincides with the reduced Gröbner basis. Generally, a reduced Gröbner basis G is a subset of Pommaret basis P such that every element in G has no proper ancestors defined as follows (cf. [14]).

Definition 3. Given a Pommaret basis P , an *ancestor* of a Boolean polynomial $p \in P$ is a polynomial $q \in P$ with

$\text{lm}(q)$ of the smallest total degree among those satisfying $\text{lm}(p) = \text{lm}(q) \cdot u$ where u is either 1 or a Boolean monomial containing $\mathcal{P}$ -nonmultiplicative variables for q . If $u \neq 1$ the ancestor q of p is called *proper*.

Thereby, algorithm **PommaretBasis** can also be used to output the reduced Gröbner basis from the computed Pommaret basis by extracting elements which do not have proper ancestors. After a suitable modification of the algorithm this can be done without extra reductions [14].

THEOREM 2. *Cardinality of a reduced Gröbner basis G in the Boolean ring $\mathbb{B}[\mathbf{x}]$ is bounded by*

$$|G| \leq \binom{n}{\lfloor \frac{n}{2} \rfloor} = \mathcal{O}\left(\frac{2^n}{\sqrt{n}}\right). \quad (9)$$

PROOF. The binomial coefficient $\binom{n}{\lfloor \frac{n}{2} \rfloor}$ where $\lfloor z \rfloor$ is the floor function specifies the maximal cardinality of subsets in a n -element set, none of which contains another, in accordance to the classical Sperner's theorem in algebraic combinatorics [18]. This is just the case for reduced Boolean Gröbner bases. The asymptotic estimation of the binomial coefficient reads

$$\binom{n}{\lfloor \frac{n}{2} \rfloor} \approx 2^n \sqrt{\frac{2}{\pi n}}.$$

□

For even n the monomial Boolean Gröbner basis of maximal cardinality $\binom{n}{n/2}$ consists of all possible distinct elements of degree $n/2$ whereas for odd n there are two different monomial Boolean Gröbner bases of maximal cardinality which have degree of their elements $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$, respectively. Here $\lceil z \rceil$ is the ceil function.

It is easy to see that the indicated Gröbner bases of maximal cardinality are also Pommaret bases. However, Boolean Pommaret bases may have cardinality larger than the bound (9). For example, the Pommaret basis in $\mathbb{B}[\mathbf{x}]$ for ideal $\langle x_n \rangle$ has 2^{n-1} elements

$$\{x_1^{i_1} \cdots x_{n-1}^{i_{n-1}} x_n \mid i_k = 0, 1; k = 1, \dots, n-1\}.$$

4. EFFICIENCY FEATURES

4.1 Avoidance of Some Useless Critical Pairs

Algorithm **PommaretBasis** admits only $\mathcal{P}$ -reductions. By this reason reduction is applied only to critical pairs (S -polynomials) of the form

$$S(f, g) = f + g \cdot u \quad f \in Q, g \in P \subset \mathbb{B}[\mathbf{x}]$$

Here monomial u may contain only $\mathcal{P}$ -multiplicative variables for g . Polynomial f is either element in the input set B or a nonmultiplicative prolongation of $p \in P$. Therefore, the algorithm treats a certain subset of all possible Gröbner critical pairs in the intermediate basis. Namely, for doing $\mathcal{P}$ -reduction only those Gröbner S -polynomials are processed which are given by

$$S(p, g) = p \cdot v + g \cdot u, \quad v = \prod_{x \in \Omega_v} x, \quad u = \prod_{x \in \Omega_u} x. \quad (10)$$

Here subset $\Omega_v \subseteq \mathbf{x}$ is either empty or contains variables which are nonmultiplicative for p . Respectively, the variables that occur in $\Omega_u \subseteq \mathbf{x}$ are multiplicative for g . Those S -polynomials which do not have the structure (10) are useless and automatically ignored in the course of algorithm **PommaretBasis**.

4.2 Using Janet Tree

To provide a fast search for the unique $\mathcal{P}$ -reductor in P where P is an intermediate set of Boolean polynomials in algorithm **PommaretBasis**, we use a special rooted binary tree proposed in [13] and called *Janet tree*. Janet tree is an appropriate data structure to search for a *Janet divisor* in a given monomial set. The last set is the set of leading monomials of the intermediate basis P in the algorithm. As well as Pommaret division in Definition 1 *Janet division* is defined by a separation of variables into multiplicative and nonmultiplicative given as follows [12, 14].

Definition 4. For each finite monomial set U and $0 \leq i \leq n$ partition U into groups $[d_0, d_1, \dots, d_i]$ labeled by $d_0, \dots, d_i \in \mathbb{N}_{\geq 0}$ ($U = [0]$) and composed of

$$\{u \in U \mid d_0 = 0, d_1 = \deg_{x_1}(u), \dots, d_i = \deg_{x_i}(u)\}.$$

Given $u \in [d_0, \dots, d_{i-1}]$, variable x_i is *Janet multiplicative* for u if

$$\deg_i(u) = \max\{\deg_i(v) \mid v \in [d_0, \dots, d_{i-1}]\}. \quad (11)$$

If equality (11) does not hold, then variable x_i is *Janet non-multiplicative*.

Applicability of Janet tree for searching a Pommaret divisor is based on the following proposition [12].

PROPOSITION 3. *Let U be a $\mathcal{P}$ -autoreduced monomial set, and w be a monomial (not necessarily in U). Then if $u \in U$ is a Pommaret divisor of w , then u is also its Janet divisor. The converse is generally not true.*

This proposition implies that to search for a Pommaret divisor one can first search for a Janet divisor and if the last one is found then to check whether it is a Pommaret divisor. This check is readily done and practically has no impact on computing time.

We illustrate the use of Janet tree in searching a $\mathcal{P}$ -divisor by a simple example. Let P be a monomial set

$$P := \{xy, xz, yz\} \quad (x \succ y \succ z).$$

Janet tree for P has the structure shown in Figure 1. The elements in P are located in the leaves of the tree whereas its edges are assigned to certain variables. A solid arrow shows transition in the tree from a parent node to the left child when degree of the current variable is increased from 0 to 1. A dashed arrow shows transition to the right child when the current variable is changed to the next one with respect to the order chosen.

Let us consider a search in the tree of Figure 1 for a $\mathcal{P}$ -divisor of monomial $u := xyz$. The search is started from the root, and because of $\deg_x(u) = 1$ the variable x is selected as a current variable. Then transition is done to the left child, i.e. the node whose degree index in x is 1. Since in the Boolean case the degree in a variable cannot exceed 1, the next transition is to be done to the second variable y shown by the dashed arrow. The arrow points at the right

tory of computation [14]. Though the experimental study of effectiveness of involutive criteria in $\mathbb{Q}[\mathbf{x}]$ shows [14] that the role of criteria is not so crucial for involutive algorithms as for the Buchberger algorithm, the criteria importance in Boolean rings requires a thorough investigation.

We compared the timings of BPB in computing the reduced Boolean Gröbner bases with those of **Singular** 3.0.4 [20] and **PolyBoRi** 0.3.1 [5, 6] for the lexicographical order. For the degree-reverse-lexicographical order we run also the **FGb** library [21] which is used with **Maple**.

All the timings (in seconds) collected in Tables 2 and 3 were done on 2×Opteron-242 (1.6 Ghz) with 6Gb of RAM, running under Gentoo Linux 2006.1 and compiled with gcc-4.1.1 (gmp-4.2.1). As this takes place, our code did not utilize the two-processor facility of our machine. The running times were confined within 2 hours. Computation of benchmarks *life* 14 and *redcyc* 14-16 with **FGb** is interrupted because of the internal limitations in this library.

As benchmarks we took some of the serial examples from the standard collections [22] widely used for experimentation with Gröbner bases software. Only those of the serial benchmarks which do not contain variables in degree higher than 1 were used. They are *cyclic*, *eco*, *redcyc(lic)*, *redco* and *katsura*. In addition to these famous series, a new one that has Boolean structure was taken from paper [23]. We call the last serial benchmarks *life*, since they were derived from analysis of the famous Game of Life by J.Conway. Every example of this new series consists of a single polynomial in variables $x_0, \dots, x_i$ of the form:

$$x_i + x_{i-1}(\sigma_{i-2} + \sigma_{i-3} + \sigma_3 + \sigma_2) + \sigma_{i-2} + \sigma_3.$$

Here σ_k is the k -th symmetric polynomial in $x_0, \dots, x_{i-2}$.

The timings in Tables 2 and 3 show that our implementation is comparable in speed with the other systems and packages.

6. CONCLUSION

We shown that the involutive algorithm based on Pommet division can be applied to computation of Gröbner bases in Boolean rings and even in its simplest, far from being optimal, form reveals rather good computational efficiency. As an improvement of the algorithm, we plan in particular to implement involutive criteria for detection of superfluous reductions for S -polynomials of form (10). To implement the criteria one needs to modify the algorithm to track history of computation as done for the general involutive algorithm in [14]. As other possible improvements we shall look for heuristically better selection strategies for nonmultiplicative prolongations than that used in the above described algorithm and for the most appropriate data structures for Boolean monomials and polynomials. Besides, we should find out why our implementation reveals irregular (non-monotonic) behavior of running time for some example in Tables 2 and 3.

The improved version, after its intensive benchmarking for optimization of running time and memory consumption, is planned to be included as a module to the **GINV** (Gröbner INVolutive) [24] open source software.

Somewhat surprisingly, but among known involutive divisions (see [14], references therein and more recent paper [25]) only Pommet division allows to compute Gröbner bases directly in Boolean rings without field polynomials. Thus, Janet division being the most frequently used in the litera-

Table 3: Lex order

Bench- mark	n	Singular 3.0.4	PolyBoRi 0.2	BPB
cyclic	8	0.02	0.70	0.00
	10	>2h	1.00	0.00
	12	76.97	3.86	5.84
	14	820.18	54.75	0.38
	16	>2h	>2h	>2h
eco	10	0.03	0.52	0.00
	11	0.03	0.56	0.00
	12	0.18	0.56	0.00
	13	0.20	0.56	0.00
	14	1.66	0.58	0.00
	15	1.66	0.66	0.02
	16	12.54	0.58	1.59
	17	12.53	0.59	0.04
	18	129.23	0.60	0.02
	19	126.08	0.69	0.63
	20	1567.20	0.69	2784.70
katsura	10	0.12	0.61	0.09
	11	0.31	0.69	0.48
	12	1.05	0.79	2.24
	13	3.24	0.92	53.65
	14	10.28	1.15	673.90
	15	30.81	2.17	4399.48
life	8	0.08	0.52	0.01
	9	1.72	0.62	0.16
	10	3.13	0.92	0.36
	11	241.26	2.27	12.85
	12	1872.54	3.72	160.34
	13	>2h	8.61	2439.67
redcyc	12	3.94	1.15	0.01
	14	75.33	3.46	0.11
	16	3866.93	42.79	1.64
	18	>2h	1395.34	295.87
redco	10	0.04	0.51	0.00
	11	0.10	0.56	0.00
	12	0.28	0.55	0.00
	13	0.78	0.54	0.00
	14	2.81	0.66	0.08
	15	7.02	0.66	0.06
	16	20.61	0.68	0.62
	17	63.76	0.64	0.00
	18	243.58	0.87	248.61
	19	703.74	0.93	146.36
	20	1622.36	1.03	0.00

ture, is not suitable for Boolean rings. To show this, consider again the example of Section 3.2. The Boolean polynomial $p := xy + y + 1$ for any admissible monomial order such that $x \succ y$ has no Janet nonmultiplicative variables: both x and y are Janet multiplicative by Definition 4. Therefore, p is a Janet basis of $\langle p \rangle \subset \mathbb{B}[x, y]$ but not a Gröbner basis. If add the field polynomial

$$\phi_x := x^2 + x, \quad \phi_y := y^2 + y$$

and consider the ideal $\langle p, \phi_x, \phi_y \rangle$ in $\mathbb{F}_2[x, y]$, then variable x becomes (Janet)nonmultiplicative for p . The Janet division algorithm [13, 14] applied to $\{p, \phi_x, \phi_y\}$ outputs the reduced Boolean Gröbner basis of Table 1.

It should be noted that Pommaret division is also singled out in study of some combinatorial problems in commutative algebra and in algebras of solvable type [26].

7. ACKNOWLEDGMENTS

The research presented in the paper was supported in part by grant 07-01-00660 from the Russian Foundation for Basic Research and by grant 1027.2008.2 from the Ministry of Education and Science of the Russian Federation. The authors want to thank Yurii Blinkov for helpful discussions and the anonymous referees for their useful critical remarks.

8. REFERENCES

- [1] J.-C.Faugère and A.Joux. Algebraic Cryptanalysis of Hidden Field Equations (HFE) Using Gröbner Bases. *LNCS 2729*, Springer-Verlag, 2003, pp.44–60.
- [2] J.-C.Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). *Proceedings of ISSAC 2002*, ACM Press, New York, 2002, pp.75–83.
- [3] <http://magma.maths.usyd.edu.au/users/allan/gb/>
- [4] J.-C.Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure and Applied Algebra*, 139(1-3): 61–68, 1999.
- [5] M.Brickenstein and A.Dreyer. PolyBoRi: A framework for Gröbner basis computations with Boolean polynomials. *Electronic Proceedings of the MEGA 2007*. <http://www.ricam.oeaw.ac.at/mega2007/>
- [6] M.Brickenstein, A.Dreyer, G.-M.Greuel and O.Wienand. *New developments in the theory of Gröbner bases and applications to formal verification*. arXiv:math.AC/0801.1177
- [7] C.Condrat and P.Kalla. A Gröbner Basis Approach to CNF-Formulae Preprocessing. Tools and Algorithms for the Construction and Analysis of Systems. *LNCS 4424*, Springer-Verlag, 2007, pp.618–631.
- [8] C.M.Dawson, H.L.Haselgrove, A.P.Hines, D.Mortimer, M.A.Nielsen and T.J.Osborne. *Quantum computing and polynomial equations over the finite field $\mathbb{Z}_2$* . arXiv:quant-ph/0408129
- [9] V.P.Gerdt, R.Kragler and A.N.Prokopenya. On Computer Algebra Application to Simulation of Quantum Computation. *Models and Methods in Few-and Many-Body Systems*, S.A.Sofianos (ed.). University of South Africa, Pretoria, 2007, pp.219–232.
- [10] M. Bardet, J.-C.Faugère and B.Salvy. *Complexity of Gröbner Basis computation for Semi-regular Overdetermined sequences over $\mathbb{F}_2$ with solutions in $\mathbb{F}_2$* . INRIA report RR-5049, 2003.
- [11] Q.-N.Tran and M.Y.Vardi. Gröbner Bases Computation in Boolean Rings for Symbolic Model Checking. *Modelling and Simulation*, R.Wamkeue (ed.), ACTA Press, 2007, pp.440–445.
- [12] V.P.Gerdt and Yu.A.Blinkov. Involutive Bases of Polynomial Ideals. *Mathematics and Computers in Simulation*, 45:519–542, 1998. arXiv:math.AC/9912027; Minimal Involutive Bases. *ibid.*, 543–560, arXiv:math.AC/9912029
- [13] V.P.Gerdt, Yu.A.Blinkov and D.A.Yanovich. Construction of Janet bases. I. Monomial bases. *Computer Algebra in Scientific Computing / CASC 2001*, Springer-Verlag, Berlin, 2001, pp.233–247; II. Polynomial bases, *ibid.*, pp.249–263.
- [14] V.P.Gerdt. Involutive Algorithms for Computing Gröbner Bases. *Computational Commutative and Non-Commutative Algebraic Geometry*, S.Cojocaru, G.Pfister and V.Ufnarovski (eds.), NATO Science Series, IOS Press, 2005, pp. 199–225. arXiv:math.AC/0501111
- [15] J.Apel. A Gröbner Approach to Involutive Bases. *Journal of Symbolic Computation*, 19(5): 441–458, 1995.
- [16] B.Buchberger. Gröbner Bases: an Algorithmic Method in Polynomial Ideal Theory. *Recent Trends in Multidimensional System Theory*, N.K. Bose (ed.), Reidel, Dordrecht, 1985, pp.184–232.
- [17] T.Becker, V.Weispfenning and H.Kredel. Gröbner Bases. A Computational Approach to Commutative Algebra. *Graduate Texts in Mathematics 141*, Springer-Verlag, New York, 1993.
- [18] K.Engel. Sperner Theory. *Encyclopedia of Mathematics and its Applications 65*, Cambridge University Press, 1997.
- [19] J.Apel and R.Hemmecke. Detecting unnecessary reductions in an involutive basis computation. *Journal of Symbolic Computation*, 40(4-5): 1131–1149, 2005.
- [20] G.-M.Greuel, G.Pfister and H.Schönemann. *Singular 3.0.4. A Computer Algebra System for Polynomial Computations*. Centre for Computer Algebra, University of Kaiserslautern, 2007. <http://www.singular.uni-kl.de>
- [21] <http://fgbrs.lip6.fr/salsa/Software/>
- [22] <http://www-sop.inria.fr/saga/POL>
<http://www.math.uic.edu/~jan/demo.html>
- [23] V.V.Korniyak. On Compatibility of Discrete Relations. *LNCS 3718*, Springer-Verlag, 2005, pp.272–284. arXiv:math-ph/0504048
- [24] <http://invo.jinr.ru>
- [25] A.Semenov. On Connection Between Constructive Involutive Divisions and Monomial Orderings. *LNCS 4194*, Springer-Verlag, 2006, pp.261–278.
- [26] W.M.Seiler. *A Combinatorial Approach to Involution and Delta-Regularity I: Involutive Bases in Polynomial Algebras of Solvable Type; II: Structure Analysis of Polynomial Modules with Pommaret Bases*, Preprints, Universität Kassel, 2007.

On Lacunary Polynomial Perfect Powers

Mark Giesbrecht^{*}
 Symbolic Computation Group
 Cheriton School of Computer Science
 University of Waterloo
 Waterloo, Ontario, Canada
 mwg@uwaterloo.ca
 www.cs.uwaterloo.ca/~mwg/

Daniel S. Roche
 Symbolic Computation Group
 Cheriton School of Computer Science
 University of Waterloo
 Waterloo, Ontario, Canada
 droche@cs.uwaterloo.ca
 www.cs.uwaterloo.ca/~droche/

ABSTRACT

We consider the problem of determining whether a t -sparse or lacunary polynomial f is a perfect power, that is, $f = h^r$ for some other polynomial h and $r \in \mathbb{N}$, and of finding h and r should they exist. We show how to determine if f is a perfect power in time polynomial in t and $\log \deg f$, i.e., polynomial in the size of the lacunary representation. The algorithm works over $\mathbb{F}_q[x]$ (at least for large characteristic) and over $\mathbb{Z}[x]$, where the cost is also polynomial in $\log \|f\|_\infty$. Subject to a conjecture, we show how to find h if it exists via a kind of sparse Newton iteration, again in time polynomial in the size of the sparse representation. Finally, we demonstrate an implementation using the C++ library NTL.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms—*Algebraic algorithms, Analysis of algorithms*; F.2.1 [Analysis of Algorithms and Problem Complexity]: Numerical Algorithms and Problems—*Computations on polynomials, Number-theoretic computations*

General Terms

Algorithms

Keywords

Lacunary polynomial, black box polynomial, sparse polynomial, perfect power

1. INTRODUCTION

Computational work on *lacunary* polynomials has proceeded apace for the past three decades. From the dramatic initial intractability results of [16, 17], through progress in algorithms (e.g., [3, 22, 12]) and complexity (e.g., [13, 18,

9]), to recent breakthroughs in root finding and factorization [7, 11, 14], these works have important and practical consequences.

By a lacunary or *supersparse* polynomial f , we mean with

$$f = \sum_{1 \leq i \leq t} c_i \bar{x}^{\bar{e}_i} \in \mathbb{F}[x_1, \dots, x_\ell], \quad (1.1)$$

where $\mathbb{F}$ is a field, $c_0, \dots, c_t \in \mathbb{F} \setminus \{0\}$, $e_1, \dots, e_t \in \mathbb{N}^\ell$ are distinct exponent tuples with $0 \leq \|e_1\|_1 \leq \dots \leq \|e_t\|_1 = \deg f$, and by $\bar{x}_i^{\bar{e}_i}$ we mean the monomial $x_1^{e_{i1}} x_2^{e_{i2}} \dots x_\ell^{e_{i\ell}}$ of degree $\|\bar{e}_i\|_1 = \sum_{1 \leq j \leq \ell} e_{ij}$. We say f is t -sparse and write $\tau(f) = t$. We will largely consider the univariate case

$$f = \sum_{1 \leq i \leq t} c_i x^{e_i} \in \mathbb{F}[x], \quad (1.2)$$

where $0 \leq e_1 < e_2 < \dots < e_t = \deg f$.

In this paper, we examine an important operation: detecting whether a lacunary polynomial f is a nontrivial perfect power of another (not necessarily lacunary) polynomial h , and if so producing the power r and possibly the h such that $f = h^r$.

We will always assume that $\tau(f) \geq 2$; otherwise, $f = x^n$, and determining whether f is a perfect power is equivalent to determining if n is not prime, for which there are well-established methods.

The defining methodology of our and previous work in this area is the sensitivity of the cost to the sparse representation. That is, we want algorithms which require a number of bit operations that is polynomial in t and $\log \deg f$. When $f \in \mathbb{Z}[x]$, we furthermore want algorithms for which the number of bit operations is polynomial in $\log \|f\|_\infty$, where $\|f\|_\infty = \max_{1 \leq i \leq t} |c_i|$ (for $f \in \mathbb{Q}[x]$, we simply work with $\bar{f} = cf \in \mathbb{Z}[x]$, for the smallest $c \in \mathbb{Z} \setminus \{0\}$). This size reflects that of the typical linked representation of polynomials in modern computer algebra systems like Maple and Mathematica.

1.1 Related work and methods

Two well-known techniques can be applied to the problem of testing for perfect powers, and both are very efficient when $f = h^r$ is dense. We can compute the squarefree decomposition of f as in [24], and determine whether f is a perfect power by checking whether the GCD of the exponents of all nontrivial factors in the squarefree decomposition is at least 2. An even faster method (in theory and practice) to find h given $f = h^r$ is by a Newton iteration. This technique has also proven to be efficient in computing perfect roots of (dense) multi-precision integers [2, 4]. In summary however,

^{*}Supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

we note that both these methods require approximately linear time in the *degree* of f , which may be exponential in the lacunary size.

Newton iteration has also been applied to finding perfect polynomial roots of lacunary (or other) polynomials given by straight-line programs. Kaltofen [10] shows how to compute a straight-line program for h , given a straight-line program for $f = h^r$ and the value of r . This method has complexity polynomial in the size of the straight-line program for f , and in the degree of h , and in particular is effective for large r . We do not address the powerful generality of straight-line programs, but do avoid the dependence on the degree of h .

Closest to this current work, Shparlinski [22] shows how to recognize whether $f = h^2$ for a lacunary polynomial f . Shparlinski uses random evaluations and tests for quadratic residues. How to determine whether a lacunary polynomial is *any* perfect power is posed as an open question.

1.2 Our contributions

Given a lacunary polynomial $f \in \mathbb{Z}[x]$, we present an algorithm to compute an $r \in \mathbb{Z}_{>1}$ such that $f = h^r$ for some $h \in \mathbb{Z}[x]$, or determine that no such r exists. Our algorithm requires polynomial time in the sparse input size and in fact is quite efficient, requiring about $O(t \log^2 \|f\|_\infty \log^2 n)$ machine operations (for convenience here we use soft-Oh notation: for functions σ and φ we say $\sigma \in O^-(\varphi)$ if $\sigma \in O(\varphi \log^c \varphi)$ for some constant $c > 0$). Our algorithms are probabilistic of the Monte Carlo type. That is, they have the ability to generate random bits at unit cost and, for any input, on any execution have a probability of getting an incorrect answer of less than $1/2$ (this possibility of error can be made arbitrarily small with a few repeated executions).

We also answer Shparlinski's open question on perfect powers of lacunary polynomials over finite fields, at least for the case of large characteristic. That is, when the characteristic q of the finite field is greater than $\deg f$, we provide a Monte Carlo algorithm that determines if there exists an $h \in \mathbb{F}_q[x]$ and r such that $f = h^r$, and finds r if it exists.

An implementation of our algorithm in NTL indicates excellent performance on sparse inputs when compared to a fast implementation based on previous technology (a variable-precision Newton iteration to find a power-series r th root of f , followed by a Monte Carlo correctness check).

Actually computing h such that $f = h^r$ is a somewhat trickier problem. Conjectures of Schinzel [20] suggest that again provided F has zero or sufficiently large characteristic, h may well be lacunary as well. In fact, we show that if $f \in \mathbb{Z}[x]$, then the number of terms in h is bounded by $\|f\|_\infty$. Conditional on the truth of a (we believe) reasonable conjecture, and with the knowledge that f is a perfect r th power, we can explicitly compute $h \in \mathbb{F}[x]$, again in time polynomial in $\log n$, t , and $\log \|f\|_\infty$.

The remainder of the paper is arranged as follows. In Section 2 we present the main theoretical tool for our algorithms and then show how to employ it for polynomials over finite fields and the integers. We also show that if a lacunary polynomial is a perfect power of some h , then it cannot be a high power, and show how to identify it. We also show how to reduce the multivariate problem to the univariate one. In Section 3 we show how to compute h such that $f = h^r$ (given that such h and r exist), subject to a conjecture we posit quite reasonable. Finally, in Section 4, we present an experimental implementation of our algorithm in NTL.

2. TESTING FOR PERFECT POWERS

In this section we describe a method to determine if a lacunary polynomial $f \in \mathbb{Z}[x]$ is a perfect power. That is, do there exist $h \in \mathbb{Z}[x]$ and $r > 1$ such that $f = h^r$? The polynomial h need not be lacunary, though some conjectures suggest it may well have to be.

We first describe algorithms to test if f is an r th power of some polynomial h , where f and r are both given and r is assumed to be prime. We present and analyze variants that work over finite fields $\mathbb{F}_q$ and over $\mathbb{Z}$. In fact, these algorithms for fixed r are for *black-box* polynomials: they only need to evaluate f at a small number of points. That this evaluation can be done quickly is a property of lacunary and other classes of polynomials.

For lacunary f we then show that, in fact, if h exists at all then r must be small unless $f = x^n$. And if f is a perfect power, then there certainly exists a prime r such that f is an r th power. So in fact the restrictions that r is small and prime are sufficient to cover all nontrivial cases, and our method is complete.

2.1 Detecting given r th powers

Our main tool in this work is the following theorem which says that, with reasonable probability, a polynomial is an r th power if and only if the modular image of an evaluation in a specially constructed finite field is an r th power.

THEOREM 2.1. *Let $q \in \mathbb{Z}$ be a prime power and $r \in \mathbb{N}$ a prime dividing $q - 1$. Suppose that $f \in \mathbb{F}_q[x]$ has degree $n \leq 1 + \sqrt{q}/2$ and is not a perfect r th power in $\mathbb{F}_q[x]$. Then*

$$R_f^{(r)} = \#\{c \in \mathbb{F}_q : f(c) \in \mathbb{F}_q \text{ is an } r\text{th power}\} \leq \frac{3q}{4}.$$

PROOF. The r th powers in $\mathbb{F}_q$ form a subgroup H of $\mathbb{F}_q^*$ of index r and size $(q-1)/r$ in $\mathbb{F}_q^*$. Also, $a \in \mathbb{F}_q^*$ is an r th power if and only if $a^{(q-1)/r} = 1$. We use the method of “completing the sum” from the theory of character sums. We refer to [15], Chapter 5, for an excellent discussion of character sums. By a multiplicative character we mean a homomorphism $\chi : \mathbb{F}_q^* \rightarrow \mathbb{C}$ which necessarily maps $\mathbb{F}_q$ onto the unit circle. As usual we extend our multiplicative characters χ so that $\chi(0) = 0$, and define the trivial character $\chi_0(a)$ to be 0 when $a = 0$ and 1 otherwise.

For any $a \in \mathbb{F}_q^*$,

$$\frac{1}{r} \sum_{\chi^r = \chi_0} \chi(a) = \begin{cases} 1 & \text{if } a \in H, \\ 0 & \text{if } a \notin H, \end{cases}$$

where χ ranges over all the multiplicative characters of order r on $\mathbb{F}_q^*$ — that is, all characters that are isomorphic to the trivial character on the subgroup H . Thus

$$\begin{aligned} R_f^{(r)} &= \sum_{a \in \mathbb{F}_q^*} \left(\frac{1}{r} \sum_{\chi^r = \chi_0} \chi(f(a)) \right) = \frac{1}{r} \sum_{\chi^r = \chi_0} \sum_{a \in \mathbb{F}_q^*} \chi(f(a)) \\ &\leq \frac{q}{r} + \frac{1}{r} \sum_{\substack{\chi^r = \chi_0 \\ \chi \neq \chi_0}} \left| \sum_{a \in \mathbb{F}_q} \chi(f(a)) \right|. \end{aligned}$$

Here we use the obvious fact that

$$\sum_{a \in \mathbb{F}_q^*} \chi_0(f(a)) \leq \sum_{a \in \mathbb{F}_q} \chi_0(f(a)) = q - d \leq q,$$

where d is the number of distinct roots of f in $\mathbb{F}_\varrho$. We next employ the powerful theorem of Weil [23] on character sums with polynomial arguments (see Theorem 5.41 of [15]), which shows that if f is *not* a perfect r th power of another polynomial, and χ has order $r > 1$, then

$$\left| \sum_{a \in \mathbb{F}_\varrho} \chi(f(a)) \right| \leq (n-1)\varrho^{1/2} \leq \frac{\varrho}{2},$$

using the fact that we insisted $n \leq 1 + \sqrt{\varrho}/2$. Summing over the $r-1$ non-trivial characters of order r , we deduce that

$$R_f^{(r)} \leq \frac{\varrho}{r} + \frac{r-1}{r} \cdot \frac{\varrho}{2} \leq \frac{3\varrho}{4}. \quad \square$$

2.2 Certifying specified powers over $\mathbb{F}_q[x]$

Theorem 2.1 allows us to detect when a polynomial $f \in \mathbb{F}_\varrho[x]$ is a perfect r th power, for known r dividing $\varrho-1$: choose random $\alpha \in \mathbb{F}_\varrho$ and evaluate $\xi = f(\alpha)^{(e-1)/r} \in \mathbb{F}_\varrho$. Recall that $\xi = 1$ if and only if $f(\alpha)$ is an r th power.

- If f is an r th power, then clearly $f(\alpha)$ is an r th power and we always have $\xi = 1$.
- If f is not an r th power, Theorem 2.1 demonstrates that for at least $1/4$ of the elements of $\mathbb{F}_\varrho$, $f(\alpha)$ is not an r th power. Thus, for α chosen randomly from $\mathbb{F}_\varrho$ we would expect $\xi \neq 1$ with probability at least $1/4$.

For a polynomial $f \in \mathbb{F}_q[x]$ over an arbitrary finite field $\mathbb{F}_q$, where q is a prime power such that $q-1$ is not divisible by r , we proceed by constructing an extension field $\mathbb{F}_{q^{r-1}}$ over $\mathbb{F}_q$. From Fermat's Little Theorem and the fact that $r \nmid q$, we know $r \mid (q^{r-1} - 1)$, and we can proceed as above. We now present and analyze this more formally.

Algorithm IsPerfectRthPowerGF

Input: A prime power q , $f \in \mathbb{F}_q[x]$ of degree $n \leq 1 + \sqrt{q}/2$, $r \in \mathbb{N}$ a prime dividing n , and $\epsilon \in \mathbb{R}_{>0}$

Output: True if f is the r th power of a polynomial in $\mathbb{F}_\varrho[x]$; False otherwise.

- 1: Find an irreducible $\Gamma \in \mathbb{F}_q[z]$ of degree $r-1$, successful with probability at least $\epsilon/2$
 - 2: $\varrho \leftarrow q^{r-1}$
 - 3: Define $\mathbb{F}_\varrho = \mathbb{F}_q[z]/(\Gamma)$
 - 4: $m \leftarrow 2.5(1 + \lceil \log_2(1/\epsilon) \rceil)$
 - 5: **for** i from 1 to m **do**
 - 6: Choose random $\alpha \in \mathbb{F}_\varrho$
 - 7: $\xi \leftarrow f(\alpha)^{(e-1)/r} \in \mathbb{F}_\varrho$
 - 8: **if** $\xi \neq 1$ **then**
 - 9: **return** False
 - 10: **return** True
-

Notes on IsPerfectRthPowerGF.

To accomplish Step 1, a number of fast probabilistic methods are available to find irreducible polynomials. We employ the algorithm of Shoup [21]. This algorithm requires $O((r^2 \log r + r \log q) \log r \log \log r)$ operations in $\mathbb{F}_q$. It is probabilistic of the Las Vegas type, and we assume that it always stops within the number of operations specified, and returns the correct answer with probability at least $1/2$ and “Fail” otherwise (it never returns an incorrect answer). The algorithm is actually presented in [21] as *always* finding an irreducible polynomial, but requiring *expected* time

as above; by not iterating indefinitely our restatement allows for a Monte Carlo analysis in what follows. To obtain an irreducible Γ with failure probability at most $\epsilon/2$ we run (our modified) Shoup's algorithm $1 + \lceil \log_2(1/\epsilon) \rceil$ times.

The restriction that $n \leq 1 + \sqrt{2}$ (or alternatively $q \geq 4(n-1)^2$) is not problematic. If this condition is not met, simply extend $\mathbb{F}_q$ with an extension of degree $\nu = \lceil \log_q(4(n-1)^2) \rceil$ and perform the algorithm over $\mathbb{F}_{q^\nu}$. At worst, each operation in $\mathbb{F}_{q^\nu}$ requires $O(M(\log n))$ operations in $\mathbb{F}_q$.

Here we define $M(r)$ as a number of operations in $\mathbb{F}$ to multiply two polynomials of degree $\leq r$ over $\mathbb{F}$, for any field $\mathbb{F}$, or the number of bit operations to multiply two integers with at most r bits. Using classical arithmetic $M(r)$ is $O(r^2)$, while using the fast algorithm of [5] we may assume $M(r)$ is $O(r \log r \log \log r)$.

THEOREM 2.2. *Let q be a prime power, $f \in \mathbb{F}_q[x]$, $r \in \mathbb{N}$ a prime dividing $\deg f$ and $\epsilon > 0$. If f is a perfect r th power the algorithm **IsPerfectRthPowerGF** always reports this. If f is not a perfect r th power then, on any invocation, this is reported correctly with probability at least $1 - \epsilon$.*

PROOF. That the algorithm always works when f is perfect power is clear from the above discussion. When f is not a perfect power, each iteration of the loop will obtain $\xi \neq 1$ (and hence a correct output) with probability at least $1/4$. By iterating the loop m times we ensure that the probability of failure is at most $\epsilon/2$. Adding this to the probability that Shoup's algorithm for Step 1 will fail yields a total probability of failure of at most ϵ . $\square$

THEOREM 2.3. *On inputs as specified, the algorithm **IsPerfectRthPowerGF** requires $O((rM(r) \log r \log q) \cdot \log(1/\epsilon))$ operations in $\mathbb{F}_q$ plus the cost to evaluate $\alpha \mapsto f(\alpha)$ at $O(\log(1/\epsilon))$ points $\alpha \in \mathbb{F}_{q^{r-1}}$.*

PROOF. As noted above, Shoup's [21] algorithm requires $O((r^2 \log r + r \log q) \log r \log \log r)$ field operations per iteration, which is within the time specified. The main cost of the loop in Steps 4–8 is computing $f(\alpha)^{(e-1)/r}$, which requires $O(\log \varrho)$ or $O(r \log q)$ operations in $\mathbb{F}_\varrho$ using repeated squaring, plus one evaluation of f at a point in $\mathbb{F}_\varrho$. Each operation in $\mathbb{F}_\varrho$ requires $O(M(r))$ operations in $\mathbb{F}_q$, and we repeat the loop $O(\log(1/\epsilon))$ times. $\square$

COROLLARY 2.4. *Given $f \in \mathbb{F}_q[x]$ of degree n with $\tau(f) = t$, and $r \in \mathbb{N}$ a prime dividing n , we can determine if f is an r th power with*

$$O((rM(r) \log r \log q + tM(r) \log n) \cdot \log(1/\epsilon))$$

operations in $\mathbb{F}_q$. When f is an r th power, the output is always correct, while if f is not an r th power, the output is correct with probability at least $1 - \epsilon$.

2.3 Certifying specified powers over $\mathbb{Z}[x]$

For an integer polynomial $f \in \mathbb{Z}[x]$, we proceed by working in the homomorphic image of $\mathbb{Z}$ in $\mathbb{F}_p$ (and then in an extension of that field). We must ensure that the homomorphism preserves the perfect power property we are interested in with high probability. Let $\text{disc}(f) = \text{res}(f, f') \in \mathbb{Z}$ be the discriminant of f . The proof of the following is left to the reader.

LEMMA 2.5. *Let $f \in \mathbb{Z}[x]$ and p a prime such that $p \nmid \text{disc}(f)$. Then f is a perfect power in $\mathbb{Z}[x]$ if and only if $\bar{f} = f \bmod p$ is a perfect power in $\mathbb{F}_p[x]$.*

Using the Hadamard Inequality, it is easily shown that $|\text{disc}(f)| \leq n^n \|f\|_2^{2n-1}$, which has at most

$$\mu = \lceil \lceil \log_2(n^n \|f\|_2^{2n-1}) \rceil / \lceil \log_2(4(n-1)^2) \rceil \rceil$$

prime factors greater than $4(n-1)^2$ (we require the lower bound $4(n-1)^2$ to employ Theorem 2.1 without resorting to field extensions). Here $\|f\|_2$ is the coefficient 2-norm of f : if f is as in (1.2) then $\|f\|_2 = (\sum_{1 \leq i \leq t} |c_i|^2)^{1/2}$. Choose a $\gamma \geq 4(n-1)^2$ such that the number of primes $\pi(2\gamma) - \pi(\gamma)$ between γ and 2γ is at least $4\mu + 1$. By [19], $\pi(2\gamma) - \pi(\gamma) \geq 2\gamma/(5 \ln \gamma)$ for $\gamma \geq 59$. Thus if $\gamma \geq \max\{14\mu \ln(14\mu), 100\}$, then a random prime not equal to r in the range $\gamma \dots 2\gamma$ divides $\text{disc}(f)$ with probability at most $1/4$. Primes p of this size have only $\log_2 p \in O(\log n + \log \log \|f\|_\infty)$ bits.

Algorithm IsPerfectRthPowerZ

Input: $f \in \mathbb{Z}[x]$ of degree n ; $r \in \mathbb{N}$ a prime dividing n ; $\epsilon \in \mathbb{R}_{>0}$;

Output: True if f is the r th power of a polynomial in $\mathbb{Z}[x]$; False otherwise

```

1:  $\mu \leftarrow \lceil \lceil \log_2(n^n \|f\|_2^{2n-1}) \rceil / \lceil \log_2(4(n-1)^2) \rceil \rceil$ 
2:  $\gamma \leftarrow \max\{14\mu \ln(14\mu), 4(n-1)^2, 100\}$ 
3: for  $i$  from 1 to  $\dots \lceil \log_2(1/\epsilon) \rceil$  do
4:    $p \leftarrow$  random prime in the range  $\gamma \dots 2\gamma$ 
5:   if NOT IsPerfectRthPowerGF( $p, f \bmod p, r, 1/4$ ) then
6:     return False
7: return True
```

THEOREM 2.6. *Let $f \in \mathbb{Z}[x]$ of degree n , $r \in \mathbb{N}$ dividing n and $\epsilon \in \mathbb{R}_{>0}$. If f is a perfect r th power, the algorithm **IsPerfectRthPowerZ** always reports this. If f is not a perfect r th power, on any invocation of the algorithm, this is reported correctly with probability at least $1 - \epsilon$.*

PROOF. If f is an r th power then so is $f \bmod p$ for any prime p , and so is any $f(\alpha) \in \mathbb{F}_p$. Thus, the algorithm always reports that f is an r th power. Now suppose f is not an r th power. If $p \mid \text{disc}(f)$ it may happen that $f \bmod p$ is an r th power. This happens with probability at most $1/4$ and we will assume that the worst happens in this case. When $p \nmid \text{disc}(f)$, the probability that **IsPerfectRthPowerGF** incorrectly reports that f is an r th power is also at most $1/4$, by our choice of parameter ϵ . Thus, on any iteration of steps 4–6, the probability of finding that f is an r th power is at most $1/2$. The probability of this happening $\lceil \log_2(1/\epsilon) \rceil$ times is clearly at most ϵ . $\square$

THEOREM 2.7. *On inputs as specified, the algorithm **IsPerfectRthPowerZ** requires*

$$O\left(rM(r) \log r \cdot M(\log n + \log \log \|f\|_\infty) \cdot (\log n + \log \log \|f\|_\infty) \cdot \log(1/\epsilon)\right),$$

or $O^*(r^2(\log n + \log \log \|f\|_\infty)^2 \cdot \log(1/\epsilon))$ bit operations, plus the cost to evaluate $(\alpha, p) \mapsto f(\alpha) \bmod p$ at $O(\log(1/\epsilon))$ points $\alpha \in \mathbb{F}_p$ for primes p with $\log p \in O(\log n + \log \log \|f\|_\infty)$.

PROOF. The number of operations required by each iteration is dominated by Step 5, for which $O(rM(r) \log r \log p)$ operations in $\mathbb{F}_p$ is sufficient by Theorem 2.3. Since $\log p \in O(\log n + \log \log \|f\|_\infty)$ we obtain the final complexity as stated. $\square$

Again, we obtain the following corollary for t -sparse polynomials in $\mathbb{Z}[x]$. This follows since the cost of evaluating a t -sparse polynomial $f \in \mathbb{Z}[x]$ modulo a prime p is $O(t \log \|f\|_\infty \log p + t \log n M(\log p))$ bit operations.

COROLLARY 2.8. *Given $f \in \mathbb{Z}[x]$ of degree n , with $\tau(f) = t$, and $r \in \mathbb{N}$ a prime dividing n , we can determine if f is an r th power with*

$$O^*((r^2 \log^2 n + t \log^2 n + t \log \|f\|_\infty \log n) \cdot \log(1/\epsilon))$$

bit operations. When f is an r th power, the output is always correct, while if f is not an r th power, the output is correct with probability at least $1 - \epsilon$.

2.4 An upper bound on r .

In this subsection we show that if $f = h^r$ and $f \neq x^n$ then r must be small. Over $\mathbb{Z}[x]$ we show that $\|h\|_2$ is small as well. A sufficiently strong result over many fields is demonstrated in [20], Theorem 1, where it is shown that if f has sparsity $t \geq 2$ then $t \geq r + 1$ (in fact a stronger result is shown involving the sparsity of h as well). This holds when either the characteristic of the ground field of f is zero or greater than $\deg f$.

Here we give a (much) simpler result for polynomials in $\mathbb{Z}[x]$, which bounds $\|h\|_2$ and is stronger at least in its dependency on t though it also depends upon the coefficients of f .

THEOREM 2.9. *Suppose $f \in \mathbb{Z}[x]$ with $\deg f = n$ and $\tau(f) = t$, and $f = h^r$ for some $h \in \mathbb{Z}[x]$ of degree s and $r \geq 2$. Then $\|h\|_2 \leq \|f\|_1^{1/r}$.*

PROOF. Let $p > n$ be prime and $\zeta \in \mathbb{C}$ a p th primitive root of unity. Then

$$\|h\|_2^2 = \sum_{0 \leq i \leq s} |h_i|^2 = \frac{1}{p} \sum_{0 \leq i < p} |h(\zeta^i)|^2.$$

(this follows from the fact that the Discrete Fourier Transform (DFT) matrix is orthogonal). In other words, the average value of $|h(\zeta^i)|^2$ for $i = 0 \dots p-1$ is $\|h\|_2^2$, and so there exists a $k \in \{0, \dots, p-1\}$ with $|h(\zeta^k)|^2 \geq \|h\|_2^2$. Let $\theta = \zeta^k$. Then clearly $|h(\theta)| \geq \|h\|_2$. We also note that $f(\theta) = h(\theta)^r$ and $|f(\theta)| \leq \|f\|_1$, since $|\theta| = 1$. Thus,

$$\|h\|_2 \leq |h(\theta)| = |f(\theta)|^{1/r} \leq \|f\|_1^{1/r}. \quad \square$$

The following corollary is particularly useful.

COROLLARY 2.10. *If $f \in \mathbb{Z}[x]$ is not of the form x^n , and $f = h^r$ for some $h \in \mathbb{Z}[x]$, then*

$$(i) \quad r \leq 2 \log_2 \|f\|_1.$$

$$(ii) \quad \tau(h) \leq \|f\|_1^{2/r}$$

PROOF. Part (i) follows since $\|h\|_2 \geq \sqrt{2}$. Part (ii) follows because $\|h\|_2 \geq \sqrt{\tau(h)}$. $\square$

These bounds relate to the sparsity of f since $\|f\|_1 \leq \tau(f) \|f\|_\infty$.

2.5 Perfect Power Detection Algorithm

We can now complete the perfect power detection algorithm, when we are given only the t -sparse polynomial f (and not r).

Algorithm IsPerfectPowerZ

Input: $f \in \mathbb{Z}[x]$ of degree n and sparsity $t \geq 2$, $\epsilon \in \mathbb{R}_{>0}$ **Output:** True and r if $f = h^r$ for some $h \in \mathbb{Z}[x]$

False otherwise.

```

1:  $\mathcal{P} \leftarrow \{\text{primes } r \mid n \text{ and } r \leq 2 \log_2(t \|f\|_\infty)\}$ 
2: for  $r \in \mathcal{P}$  do
3:   if IsPerfectRthPowerZ( $f, r, \epsilon/\#\mathcal{P}$ ) then
4:     return True and  $r$ 
5: return False

```

THEOREM 2.11. *If $f \in \mathbb{Z}[x] = h^r$ for some $h \in \mathbb{Z}[x]$, the algorithm **IsPerfectPowerZ** always returns “True” and returns r correctly with probability at least $1 - \epsilon$. Otherwise, it returns “False” with probability at least $1 - \epsilon$. The algorithm requires $O(t \log^2 \|f\|_\infty \log^2 n \log(1/\epsilon))$ bit operations.*

PROOF. From the preceding discussions, we can see that if f is a perfect power, then it must be a perfect r th power for some $r \in \mathcal{P}$. So the algorithm must return true on some iteration of the loop. However, it may incorrectly return true *too early* for an r such that f is not actually an r th power; the probability of this occurring is the probability of error when f is not a perfect power, and is less than $\epsilon/\#\mathcal{P}$ at each iteration. So the probability of error on any iteration is at most ϵ , which is what we wanted.

The complexity result follows from the fact that each $r \in \mathcal{P}$ has $r \leq t - 1$ and obtain the following algorithm. $\square$

For polynomials in $\mathbb{F}_q[x]$ we use Schinzel’s bound that $r \leq t - 1$ and obtain the following algorithm.

Algorithm IsPerfectPowerGF

Input: $f \in \mathbb{F}_q[x]$ of degree n and sparsity t , where the characteristic of $\mathbb{F}_q$ is greater than n , and $\epsilon \in \mathbb{R}_{>0}$ **Output:** True and r if $f = h^r$ for some $h \in \mathbb{F}_q[x]$;

False otherwise.

```

1:  $\mathcal{P} \leftarrow \{\text{primes } r \mid n \text{ and } r \leq t\}$ 
2: for  $p \in \mathcal{P}$  do
3:   if IsPerfectRthPowerGF( $f, r, \epsilon/\#\mathcal{P}$ ) then
4:     return True and  $r$ ;

```

THEOREM 2.12. *If $f = h^r$ for $h \in \mathbb{F}_q[x]$, the algorithm **IsPerfectPowerGF** always returns “True” and returns r correctly with probability at least $1 - \epsilon$. Otherwise, it returns “False” with probability at least $1 - \epsilon$. The algorithm requires $O(t^3(\log q + \log n))$ operations in $\mathbb{F}_q$.*

PROOF. The proof is equivalent to that of Theorem 2.11, using the complexity bounds in Corollary 2.4. $\square$

2.6 Detecting multivariate perfect powers

In this subsection we examine the problem of detecting multivariate perfect powers. That is, given a lacunary $f \in \mathbb{F}[x_1, \dots, x_\ell]$ as in (1.1), how do we determine if $f = h^r$ for some $h \in \mathbb{F}[x_1, \dots, x_\ell]$ and $r \in \mathbb{N}$. This is done simply as a reduction to the univariate case.

The proof of the following is left to the reader.

LEMMA 2.13. *Given $f \in \mathbb{F}[x_1, \dots, x_\ell]$ of total degree $n > 0$ and such that $\deg_{x_1} f > 0$. Let*

$$\Delta = \text{disc}_{x_1}(f) = \text{res}_{x_1}(f, \partial f / \partial x_1) \in \mathbb{F}[x_2, \dots, x_\ell].$$

Assume that $a_2, \dots, a_\ell \in \mathbb{F}$ with $\Delta(a_2, \dots, a_\ell) \neq 0$. Then $f(x_1, \dots, x_\ell)$ is a perfect power if and only if $f(x_1, a_2, \dots, a_\ell) \in \mathbb{F}[x_1]$ is a perfect power.

It is easy to see that the total degree of Δ is less than $2n^2$. Thus, for randomly chosen $a_2, \dots, a_\ell$ from a set $\mathcal{S} \subseteq \mathbb{F}$ of size at least $8n^2$ we have $\Delta(a_2, \dots, a_\ell) = 0$ with probability less than $1/4$. This can be made arbitrarily small by increasing the set size and/or repetition. We then run the appropriate univariate algorithm over $\mathbb{F}[x_1]$ (depending upon the field) to identify whether or not f is a perfect power, and if so, to find r .

3. COMPUTING PERFECT ROOTS

Once we have determined that $f \in \mathbb{F}[x]$ is equal to h^r for some $h \in \mathbb{F}[x]$, an obvious question to ask is how to actually compute h . Here we give an algorithm to accomplish this task, subject to a conjecture.

3.1 Sparsity bounds

The conjecture we rely on relates to some questions first raised by Erdős almost 60 years ago [8] on the number of terms of a square of a polynomial. Schinzel later answered these questions and in fact generalized to the case of perfect powers. For any polynomial $h \in \mathbb{F}[x]$, Schinzel proved that $\tau(h^r)$ tends to infinity as $\tau(h)$ tends to infinity [20]. He also gave an explicit upper bound on $\tau(h)$ in terms of $\tau(h^r)$, which unfortunately is an exponential function and therefore not useful for us to prove polynomial-time complexity.

However, our own (limited) investigations, along with more extensive ones by Coppersmith & Davenport [6], and later Abbott [1], suggest that, for any $h \in \mathbb{F}[x]$, where the characteristic of $\mathbb{F}$ is not too small, $\tau(h) \in O(\tau(h^r) + r)$.

We make use of the following slightly stronger conjecture, which suffices to prove our algorithm runs in polynomial time.

CONJECTURE 3.1. *For $r, s \in \mathbb{N}$, if the characteristic of $\mathbb{F}$ is zero or greater than rs , and $h \in \mathbb{F}[x]$ with $\deg h = s$, then*

$$\tau(h^i \bmod x^{2s}) < \tau(h^r \bmod x^{2s}) + r, \quad i = 1, 2, \dots, r - 1.$$

This corresponds to intuition and experience, as the system is still overly constrained with only s degrees of freedom. A weaker conjecture would suffice to prove polynomial time, but we use the stated bounds as we believe these give more accurate complexity measures.

3.2 Perfect root computation algorithm

Our algorithm is essentially a Newton iteration, with special care taken to preserve sparsity. We start with the image of h modulo x , using the fact that $f(0) = h(0)^r$, and at Step $i = 1, 2, \dots, \lceil \log_2(\deg h + 1) \rceil$, we compute the image of h modulo x^i .

Here, and for the remainder of this section, we will assume that $f, h \in \mathbb{F}[x]$ with degrees n and s respectively such that $f = h^r$ for $r \in \mathbb{N}$ at least 2, and that the characteristic of $\mathbb{F}$ is either zero or greater than n . As usual, we define $t = \tau(f)$. We require the following simple lemma.

LEMMA 3.2.¹ *Let $k, \ell \in \mathbb{N}$ such that $\ell \leq k$ and $k + \ell \leq s$, and suppose $h_1 \in \mathbb{F}[x]$ is the unique polynomial with degree less than k satisfying $h_1^r \equiv f \bmod x^k$. Then*

$$\tau(h_1^{r+1} \bmod x^{k+\ell}) \leq 2t(t + r).$$

¹Lemma subject to the validity of Conjecture 3.1.

PROOF. Let $h_2 \in \mathbb{F}[x]$ be the unique polynomial of degree less than ℓ satisfying $h_1 + h_2 x^k \equiv h \pmod{x^{k+\ell}}$. Since $h^r = f$,

$$f \equiv h_1^r + r h_1^{r-1} h_2 x^k \pmod{x^{k+\ell}}.$$

Multiplying by h_1 and rearranging gives

$$h_1^{r+1} \equiv h_1 f - r f h_2 x^k \pmod{x^{k+\ell}}.$$

Because $h_1 \pmod{x^k}$ and $h_2 \pmod{x^\ell}$ each have at most $\tau(h)$ terms, which by Conjecture 3.1 is less than $t - r$, the total number of terms in $h_1^{r+1} \pmod{x^{k+\ell}}$ is less than $2t(t - r)$. $\square$

This essentially tells us that the “error” introduced by examining higher-order terms of h_1^r is not too dense. It leads to the following algorithm for computing h .

Algorithm ComputePolyRoot

Input: $f \in \mathbb{F}[x]$, $r \in \mathbb{N}$ such that f is a perfect r th power

Output: $h \in \mathbb{F}[x]$ such that $f = h^r$

1: $u \leftarrow$ highest power of x dividing f
2: $f_u \leftarrow$ coefficient of x^u in f
3: $g \leftarrow f / (f_u x^u)$
4: $h \leftarrow 1$, $k \leftarrow 1$
5: **while** $kr \leq \deg g$ **do**
6: $\ell \leftarrow \min\{k, (\deg g)/r + 1 - k\}$

7: $a \leftarrow \frac{hg - h^{r+1} \pmod{x^{k+\ell}}}{rx^k}$

8: $h \leftarrow h + (a/g \pmod{x^\ell})x^k$

9: $k \leftarrow k + \ell$

10: $b \leftarrow$ any r th root of f_u in $\mathbb{F}$

11: **return** $bhx^{u/r}$

THEOREM 3.3. *If $f \in \mathbb{F}[x]$ is a perfect r th power, then **ComputePolyRoot** returns an $h \in \mathbb{F}[x]$ such that $h^r = f$.*

PROOF. Let u, f_u, g be as defined in Steps 1–4. Thus $f = f_u g x^u$. Now let $\hat{h}$ be some r th root of f , which we assume exists. If we similarly write $\hat{h} = \hat{h}_v \hat{g} x^v$, with $\hat{g}(0) = 1$, then $\hat{h}^r = \hat{h}_v^r \hat{g}^r x^{vr}$. Therefore f_u must be a perfect r th power in $\mathbb{F}$, $r|u$, and g is a perfect r th power in $\mathbb{F}[x]$ of some polynomial with constant coefficient equal to 1.

Denote by h_i the value of h at the beginning of the i th iteration of the while loop. So $h_1 = 1$. We claim that at each iteration through Step 6, $h_i^r \equiv g \pmod{x^k}$. From the discussion above, this holds for $i = 1$. Assuming the claim holds for all $i = 1, 2, \dots, j$, we prove it also holds for $i = j+1$.

From Step 8, $h_{j+1} = h_j + (a/g \pmod{x^\ell})x^k$, where a is as defined on the j th iteration of Step 7. We observe that

$$h_j h_j^r \equiv h_j^{r+1} + r h_j^r (a/g \pmod{x^\ell})x^k \pmod{x^{k+\ell}}.$$

From our assumption, $h_j^r \equiv f \pmod{x^k}$, and $l \leq k$, so we have

$$h_j h_j^{r+1} \equiv h_j^{r+1} + r a x^k \equiv h_j^{r+1} + h_j f - h_j^{r+1} \equiv h_j f \pmod{x^{k+\ell}}$$

Therefore $h_{j+1}^r \equiv f \pmod{x^{k+\ell}}$, and so by induction the claim holds at each step. Since the algorithm terminates when $kr > \deg g$, we can see that the final value of h is an r th root of g . Finally, $(bhx^{u/r})^r = f_u g x^u = f$, so the theorem holds. $\square$

THEOREM 3.4.² *If $f \in \mathbb{F}[x]$ has degree n and t nonzero terms, then **ComputePolyRoot** uses $O((t+r)^4 \log r \log n)$ operations in $\mathbb{F}$ and an additional $O((t+r)^4 \log r \log^2 n)$ bit operations, not counting the cost of Step 10.*

PROOF. First consider the cost of computing h^{r+1} in Step 7. This will be accomplished by repeatedly squaring and multiplying by h , for a total of at most $2\lfloor \log_2(r+1) \rfloor$ multiplications. As well, each intermediate product will have at most $\tau(f) + r < (t+r)^2$ terms, by Conjecture 3.1. The number of field operations required, at each iteration, is $O((t+r)^4 \log r)$, for a total cost of $O((t+r)^4 \log r \log n)$.

Furthermore, since $k + \ell \leq 2^i$ at the i th step, for $1 \leq i < \log_2 n$, the total cost in bit operations is less than

$$\sum_{1 \leq i < \log_2 n} (t+r)^4 \log_2 r i \in O((t+r)^4 \log r \log^2 n).$$

In fact, this is the most costly step. The initialization in Steps 1–4 uses only $O(t)$ operations in $\mathbb{F}$ and on integers at most n . And the cost of computing the quotient on Step 8 is proportional to the cost of multiplying the quotient and dividend, which is at most $O(t(t+r))$. $\square$

The method used for Step 10 depends on the field $\mathbb{F}$. For $\mathbb{F} = \mathbb{Q}$, we just need to find two integer perfect roots, which can be done in “nearly linear” time [4]. Otherwise, we can compute a root of $x^r - f_u$ using $O(r^{O(1)})$ operations in $\mathbb{F}$.

When $\mathbb{F} = \mathbb{Q}$, we must account for coefficient growth. We use the normal notion of the size of a rational number: For $\alpha \in \mathbb{Q}$, write $\alpha = a/b$ for a, b relatively prime integers. Then define $\mathcal{H}(\alpha) = \max\{|a|, |b|\}$. And for $f \in \mathbb{Q}[x]$ with coefficients $c_1, \dots, c_t \in \mathbb{Q}$, write $\mathcal{H}(f) = \max \mathcal{H}(c_i)$.

Thus, the size of the lacunary representation of $f \in \mathbb{Q}[x]$ is proportional to $\tau(f)$, $\deg f$, and $\log \mathcal{H}(f)$. Now we prove the bit complexity of our algorithm is polynomial in these values, when $\mathbb{F} = \mathbb{Q}$.

THEOREM 3.5.² *Suppose $f \in \mathbb{Q}[x]$ has degree n and t nonzero terms, and is a perfect r th power. **ComputePolyRoot** computes an r th root of f using $O^-(t(t+r)^4 \cdot \log n \cdot \log \mathcal{H}(f))$ bit operations.*

PROOF. Let $h \in \mathbb{Q}[x]$ such that $h^r = f$, and let $c \in \mathbb{Z}_{>0}$ be minimal such that $ch \in \mathbb{Z}[x]$. Gauß’s Lemma tells us that c^r must be the least positive integer such that $c^r f \in \mathbb{Z}[x]$ as well. Then, using Theorem 2.9, we have:

$$\mathcal{H}(h) \leq \|ch\|_\infty \leq \|ch\|_2 \leq (t\|c^r f\|_\infty)^{1/r} \leq t^{1/r} \mathcal{H}(f)^{(t+1)/r}.$$

(The last inequality comes from the fact that the lcm of the denominators of f is at most $\mathcal{H}(f)^t$.)

Hence $\log \mathcal{H}(h) \in O((t \log \mathcal{H}(f))/r)$. Clearly the most costly step in the algorithm will still be the computation of h_i^{r+1} at each iteration through Step 7. For simplicity in our analysis, we can just treat h_i (the value of h at the i th iteration of the while loop in our algorithm) as equal to h (the actual root of f), since we know $\tau(h_i) \leq \tau(h)$ and $\mathcal{H}(h_i) \leq \mathcal{H}(h)$.

Lemma 3.2 and Conjecture 3.1 tell us that $\tau(h^i) \leq 2(t+r)^2$ for $i = 1, 2, \dots, r$. To compute h^{r+1} , we will actually compute $(ch)^{r+1} \in \mathbb{Z}[x]$ by repeatedly squaring and multiplying by ch , and then divide out c^{r+1} . This requires at most $\lfloor \log_2 r + 1 \rfloor$ squares and products.

Note that $\|(ch)^{2i}\|_\infty \leq (t+r)^2 \|(ch)^i\|_\infty^2$ and $\|(ch)^{i+1}\|_\infty \leq (t+r)^2 \|(ch)^i\|_\infty \|ch\|_\infty$. Therefore

$$\|(ch)^i\|_\infty \leq (t+r)^{2r} \|ch\|_\infty^r, \quad i = 1, 2, \dots, r,$$

²Theorem subject to the validity of Conjecture 3.1.

and thus $\log \|(ch)^i\|_\infty \in O(r(t+r) + t \log \mathcal{H}(f))$, for each intermediate power $(ch)^i$.

Thus each of the $O((t+r)^4 \log r)$ field operations at each iteration costs $O(M(t \log \mathcal{H}(f) + \log r(t+r)))$ bit operations, which then gives the stated result. $\square$

3.3 Further comments on the algorithm

We have shown that, subject to the truth of Conjecture 3.1, and using the bounds on r from Corollary 2.10 and [20], our algorithm runs in polynomial time in lacunary representation size of the input. We hope to understand more about the size of a perfect r th root so that our algorithm can be made unconditional, as it seems to perform well in practice.

Note that `ComputePolyRoot` and the theorems that follow require that f is actually a perfect r th power. However, with the concrete bounds on the size of the r th root we have proven and conjectured, it would be easy to terminate the algorithm immediately whenever the partial computation of the result h is “too big” (either in sparsity or height). Finally, using the ideas of sparse interpolation from [12], we can certify that the h computed is actually a perfect r th root with $O(t)$ evaluations in f and in h . These modifications will not affect the asymptotic complexity of the algorithm.

Our algorithm could also be used to compute a right decomposition factor of f of degree n/r (that is, $h \in \mathbb{F}[x]$ such that $f = g \circ h$ for some $g \in \mathbb{F}[x]$) when one exists. This is because the high-order terms of f are the same as those from h^r . However, even less is known about the size of h in this case, so proving a polynomial-time complexity would be more difficult (or rely on more shaky conjectures).

Another approach might be to construct a black box for evaluating h from f and r (and hence avoid difficult conjectures on the sparsity of h). We could then choose to reconstruct h via sparse interpolation. The techniques used in [10] to compute polynomial roots of straight-line programs might be useful here, though it is unclear to us how to avoid the dependence on the degree of h .

4. IMPLEMENTATION

To investigate the practicality of our algorithms, we implemented `IsPerfectPowerZ` using Victor Shoup’s NTL. This is a high-performance C++ for fast dense univariate polynomial computations over $\mathbb{Z}[x]$ or $\mathbb{F}_q[x]$.

NTL does not natively support a lacunary polynomial representation, so we wrote our own using vectors of coefficients and of exponents. In fact, since `IsPerfectPowerZ` is a black-box algorithm, the only sparse polynomial arithmetic we needed to implement was for evaluation at a given point.

The only significant diversion between our implementation and the algorithm specified in Section 2 is our choice of the ground field. Rather than working in a degree- $(r-1)$ extension of $\mathbb{F}_p$, we simply find a random p in the same range such that $(r-1) \mid p$. It is more difficult to prove that we can find such a p quickly (using e.g. the best known bounds on Linnik’s Constant), but in practice this approach is very fast because it avoids computing in field extensions.

As a point of comparison, we also implemented the Newton iteration approach to computing perfect polynomial roots, which appears to be the fastest known method for dense polynomials. This is not too dissimilar from the techniques from the previous section on computing a lacunary r th root, but without paying special attention to sparsity. We work modulo a randomly chosen prime p to compute an r th per-

fect root h , and then use random evaluations of h and the original input polynomial f to certify correctness. This yields a Monte Carlo algorithm with the same success probability as ours, and so provides a suitable and fair comparison.

We ran two sets of tests comparing these algorithms. The first set, depicted in Figure 1, does not take advantage of sparsity at all; that is, the polynomials are dense and have close to the maximal number of terms. It appears that the worst-case running time of our algorithm is actually a bit better than the Newton iteration method on dense input, but on the average they perform roughly the same. The lower triangular shape comes from the fact that both algorithms can (and often do) terminate early. The visual gap in the timings for the sparse algorithm comes from the fact that exactly half of the input polynomials were perfect powers. It appears our algorithm terminates more quickly when the polynomial is not a perfect power, but usually takes close to the full amount of time otherwise.

The second set of tests, depicted in Figure 2, held the number of terms of the perfect power, $\tau(f)$, roughly fixed, letting the degree n grow linearly. Here we can see that, for sufficiently sparse f , our algorithm performs significantly and consistently better than the Newton iteration. In fact, we can see that, with some notable but rare exceptions, it appears that the running time of our algorithm is largely independent of the degree when the number of terms remains fixed. The outliers we see probably come from inputs that were unluckily dense (it is not trivial to produce examples of h^r with a given fixed number of nonzero terms, so the sparsity did vary to some extent).

Perhaps most surprisingly, although the choices of parameters for these two algorithms only guaranteed a probability of success of at least $1/2$, in fact over literally millions of tests performed with both algorithms and a wide range of input polynomials, not a single failure was recorded. This is of course due to the loose bounds employed in our analysis, indicating a lack of understanding at some level, but it also hints at the possibility of a deterministic algorithm, or at least one which is probabilistic of the Las Vegas type.

Both implementations are available as C++ code downloadable from the second author’s website.

5. REFERENCES

- [1] J. Abbott. Sparse squares of polynomials. *Math. Comp.*, 71(237):407–413 (electronic), 2002.
- [2] E. Bach and J. Sorenson. Sieve algorithms for perfect power testing. *Algorithmica*, 9(4):313–328, 1993.
- [3] M. Ben-Or and P. Tiwari. A deterministic algorithm for sparse multivariate polynomial interpolation. In *Proc. STOC 1988*, pages 301–309, New York, N.Y., 1988. ACM Press.
- [4] D. J. Bernstein. Detecting perfect powers in essentially linear time. *Mathematics of Computation*, 67(223):1253–1283, 1998.
- [5] D. Cantor and E. Kaltofen. Fast multiplication of polynomials over arbitrary algebras. *Acta Informatica*, 28:693–701, 1991.
- [6] D. Coppersmith and J. Davenport. Polynomials whose powers are sparse. *Acta Arith.*, 58(1):79–87, 1991.
- [7] F. Cucker, P. Koiran, and S. Smale. A polynomial time algorithm for Diophantine equations in one variable. *J. Symbolic Comput.*, 27(1):21–29, 1999.

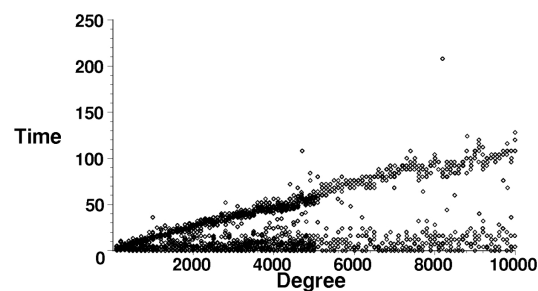
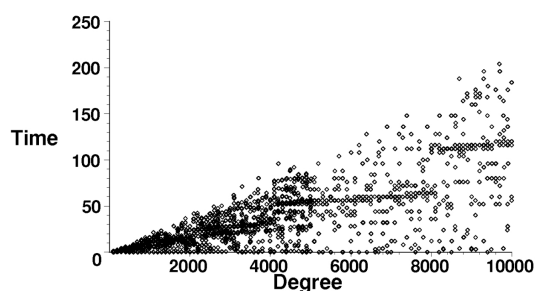


Figure 1: Comparison of Newton Iteration (left) vs. our `IsPerfectPowerZ` (right). Inputs are dense.

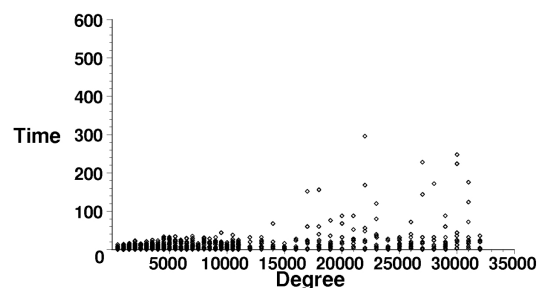
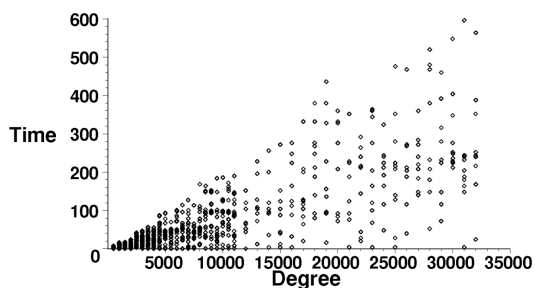


Figure 2: Comparison of Newton Iteration (left) vs our `IsPerfectPowerZ` (right). Inputs are sparse, with sparsity fixed around 500.

- [8] P. Erdős. On the number of terms of the square of a polynomial. *Nieuw Arch. Wiskunde* (2), 23:63–65, 1949.
- [9] J. von zur Gathen, M. Karpinski, and I. Shparlinski. Counting curves and their projections. In *ACM Symposium on Theory of Computing*, pages 805–812, 1993.
- [10] E. Kaltofen. Single-factor hensel lifting and its application to the straight-line complexity of certain polynomials. In *STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 443–452, New York, NY, USA, 1987. ACM.
- [11] E. Kaltofen and P. Koiran. Finding small degree factors of multivariate supersparse (lacunary) polynomials over algebraic number fields. In *ISSAC '06: Proceedings of the 2006 international symposium on Symbolic and algebraic computation*, pages 162–168. ACM Press, New York, NY, USA, 2006.
- [12] E. Kaltofen and W. s. Lee. Early termination in sparse interpolation algorithms. *J. Symbolic Comput.*, 36(3-4):365–400, 2003. International Symposium on Symbolic and Algebraic Computation (ISSAC'2002) (Lille).
- [13] M. Karpinski and I. Shparlinski. On the computational hardness of testing square-freeness of sparse polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 6(027), 1999.
- [14] H. W. Lenstra, Jr. Finding small degree factors of lacunary polynomials. In *Number theory in progress, Vol. 1 (Zakopane-Kościelisko, 1997)*, pages 267–276. de Gruyter, Berlin, 1999.
- [15] R. Lidl and H. Niederreiter. *Finite Fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Addison-Wesley, Reading MA, 1983.
- [16] D. A. Plaisted. Sparse complex polynomials and polynomial reducibility. *J. Comp. and System Sciences*, 14:210–221, 1977.
- [17] D. A. Plaisted. New NP-hard and NP-complete polynomial and integer divisibility problems. *Theor. Computer Science*, 31:125–138, 1984.
- [18] A. Quick. Some gcd and divisibility problems for sparse polynomials. Technical Report 191/86, University of Toronto, 1986.
- [19] J. B. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Ill. J. Math.*, 6:64–94, 1962.
- [20] A. Schinzel. On the number of terms of a power of a polynomial. *Acta Arith.*, 49(1):55–70, 1987.
- [21] V. Shoup. Fast construction of irreducible polynomials over finite fields. *J. Symbolic Comput.*, 17(5):371–391, 1994.
- [22] I. Shparlinski. Computing Jacobi symbols modulo sparse integers and polynomials and some applications. *J. Algorithms*, 36(2):241–252, 2000.
- [23] A. Weil. On some exponential sums. *Proc Nat. Acad. Sci. U.S.A.*, 34:204–207, 1948.
- [24] David Y.Y. Yun. On square-free decomposition algorithms. In *SYMSAC '76: Proceedings of the third ACM symposium on Symbolic and algebraic computation*, pages 26–35, New York, NY, USA, 1976. ACM.

Plane Geometry and Convexity of Polynomial Stability Regions

Didier Henrion
LAAS-CNRS
University of Toulouse
France
henrion@laas.fr

Michael Šebek
Faculty of Electrical Engineering
Czech Technical University in Prague
Czech Republic
m.sebek@c-a-k.cz

ABSTRACT

The set of controllers stabilizing a linear system is generally non-convex in the parameter space. In the case of two-parameter controller design (e.g. PI control or static output feedback with one input and two outputs), we observe however that quite often for benchmark problem instances, the set of stabilizing controllers seems to be convex. In this note we use elementary techniques from real algebraic geometry (resultants and Bézoutian matrices) to explain this phenomenon. As a byproduct, we derive a convex linear matrix inequality (LMI) formulation of two-parameter fixed-order controller design problem, when possible.

Categories and Subject Descriptors

G.1.5 [Numerical analysis]: Roots of nonlinear equations—*Polynomials, methods for*; G.1.6 [Numerical analysis]: Optimization—*Convex programming*; J.7 [Computer Applications]: Computers in other systems—*Command and control*

General Terms

Algorithms, Theory.

Keywords

Control Theory, Convexity, Resultants.

1. INTRODUCTION

Despite its elementary formulation, the problem of fixed-order controller design for linear time-invariant systems remains mostly open. Especially scarce are numerically efficient computer-aided control system design algorithms in the fixed-order case, sharply contrasting with the large number of tools available to solve static state feedback design or dynamical output feedback design with controllers of the same order as the plant. Mathematically, fixed-order controller design can be formulated as a non-convex non-smooth optimization problem in the parameter space. To the best of our knowledge, randomized algorithms are amongst the most efficient numerical methods to cope with this class of difficult problems. See [4, 13] for computer experiments supporting this claim,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

thanks to public-domain Matlab packages (HIFOO and the Randomized Control System Toolbox).

This note was motivated by the observation, made by the first author during a workshop at AIM in August 2005 [8], that 6 out of the 7 two-dimensional instances of static output feedback (SOF) design problems found in the database COMPluib [10] seem to be convex. Further motivation was provided by the excellent historical survey [7] on D-decomposition techniques, previously studied in deep detail in [12] and [1]. In [7] the authors describe the intricate geometry of two-dimensional stability regions with the help of illustrative examples. Quite often, the stability regions represented in these references seem to be convex.

In this note, we use basic results from real algebraic geometry to detect convexity of the stability region in the two-parameter case (including PI controllers, PID with constant gain, SOF design with one input two outputs or two inputs one output). We also derive, when possible, a linear matrix inequality (LMI) formulation of the stability region.

2. PROBLEM STATEMENT

We consider a parametrized polynomial

$$p(s, k) = p_0(s) + k_1 p_1(s) + k_2 p_2(s) \quad (1)$$

where the $p_i(s) \in \mathbb{R}[s]$ are given polynomials of $s \in \mathbb{C}$ and the $k_i \in \mathbb{R}$ are parameters. We assume, without loss of generality, that the ratio $p_1(s)/p_2(s)$ is not a constant.

Define the stability region

$$S = \{k \in \mathbb{R}^2 : p(s, k) \text{ stable}\}$$

where stability is meant in the continuous-time sense, i.e. all the roots of $p(s, k)$ must lie in the open left half-plane¹.

We are interested in the following problems:

- Is stability region S convex ?
- If it is convex, give an LMI representation

$$S = \{k \in \mathbb{R}^2 : F_0 + F_1 k_1 + F_2 k_2 \succ 0\}$$

when possible, where the F_i are real symmetric matrices to be found, and $\succ 0$ means positive definite.

2.1 Example: PI controller design

Let $b(s)/a(s)$ with $a(s), b(s) \in \mathbb{R}[s]$ denote the transfer function of an open loop plant, and consider a proportional integral (PI) controller $k_1/s + k_2$ in a standard negative feedback configuration. The

¹Similar results can be derived for discrete-time unit disk stability or other semialgebraic stability domains of the complex plane, but this is not covered here.

closed-loop characteristic polynomial (1) is then $p(s, k) = sa(s) + (k_1 + k_2s)b(s)$ hence $p_0(s) = sa(s)$, $p_1(s) = b(s)$ and $p_2(s) = sb(s)$.

2.2 Example: static output feedback

Given matrices $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$, $C \in \mathbb{R}^{p \times n}$, we want to find a matrix $K \in \mathbb{R}^{m \times p}$ such that the closed-loop matrix $A + BKC$ is stable. When $mp = 2$, the characteristic polynomial (1) writes $p(s, k) = \det(sI_n - A - BKC)$ hence $p_0(s) = \det(sI_n - A)$, $p_1(s) = \det(sI_n - B[k_1, 0]C)$ and $p_2(s) = \det(sI_n - B[0, k_2]C)$.

3. HERMITE MATRIX

The Routh-Hurwitz criterion for stability of polynomials has a symmetric version called the Hermite criterion. A polynomial is stable if and only if its Hermite matrix, quadratic in the polynomial coefficients, is positive definite. In control systems terminology, the Hermite matrix is a particular choice of a Lyapunov matrix certifying stability [11]. Algebraically, the Hermite matrix can be defined via the Bézoutian, a symmetric form of the resultant [6, Section 5.1.2].

Let $a(u)$, $b(u)$ be two polynomials of degree n of the indeterminate u . Define the Bézoutian matrix $B_u(a, b)$ as the symmetric matrix of size n with entries b_{ij} satisfying the linear equations

$$\frac{a(u)b(v) - a(v)b(u)}{v - u} = \sum_{i=1}^n \sum_{j=1}^n b_{ij} u^{i-1} v^{j-1}.$$

The polynomial $r_u(a, b) = \det B_u(a, b)$ is the resultant of $a(u)$ and $b(u)$ with respect to u . It is obtained by eliminating u from the system of equations $a(u) = b(u) = 0$.

The Hermite matrix of $p(s)$ is defined as the Bézoutian matrix of the real part and the imaginary part of $p(j\omega)$:

$$\begin{aligned} p_R(\omega^2) &= \operatorname{Re} p(j\omega) \\ \omega p_I(\omega^2) &= \operatorname{Im} p(j\omega) \end{aligned}$$

that is, $H(p) = B_\omega(p_R(\omega^2), \omega p_I(\omega^2))$. Let us assume that $p(s)$ is monic, with unit leading coefficient. The Hermite stability criterion can be formulated as follows.

LEMMA 1. *Polynomial $p(s)$ is stable if and only if $H(p) \succ 0$.*

Proof: The proof of this result can be found in [11] for example. It can also be proved via Cauchy indices and Hermite quadratic forms for counting real roots of polynomials, see [3, Section 9.3] $\square$.

By construction, the Hermite matrix of parametrized polynomial (1)

$$H(p(s, k)) = H(k) = \sum_{i_1, i_2=0}^2 H_{i_1 i_2} k_1^{i_1} k_2^{i_2}$$

is quadratic in k . Therefore, the Hermite criterion yields a quadratic matrix inequality formulation of the stability region:

$$S = \{k \in \mathbb{R}^2 : H(k) \succ 0\}.$$

Quadratic matrix inequalities, a generalization of bilinear matrix inequalities, typically generate non-convex regions. For example, the scalar quadratic inequality $k_1^2 - 1 > 0$ models a disconnected, hence non-convex set.

Surprisingly, it turns out that S , even though modeled by a quadratic matrix inequality, is often a convex set for practical problem instances. Here are some examples.

3.1 Examples: static output feedback

Consider the 7 two-parameter SOF problems found in the database COMpleib [10], labelled AC4, AC7, AC17, NN1, NN5, NN17 and HE1. Stability regions are represented as shaded gray areas on Figures 1 to 7. Visual inspection reveals that 6 out of 7 stability regions seem to be convex. The only apparently nonconvex example is HE1.

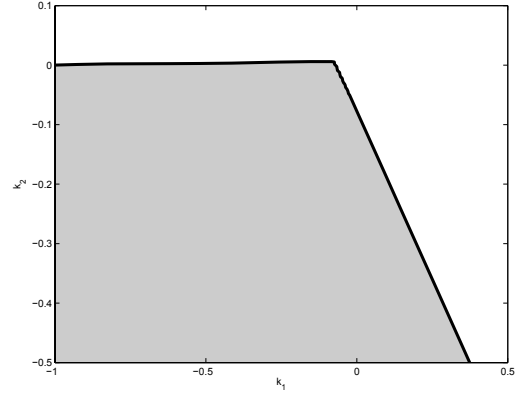


Figure 1: AC4.

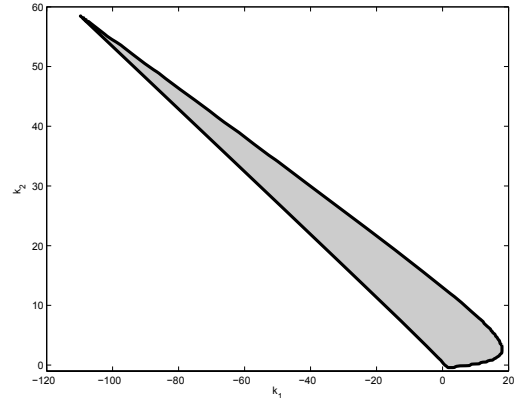


Figure 2: AC7.

In the remainder of the paper we will explain why such planar stability regions are likely to be convex, and how we can constructively derive their LMI formulations when possible.

4. RATIONAL BOUNDARY OF THE STABILITY REGION

Define the curve

$$C = \{k \in \mathbb{R}^2 : p(j\omega, k) = 0, \omega \in \mathbb{R}\}$$

which is the set of parameters k for which polynomial $p(s, k)$ has a root along the boundary of the stability region, namely the imaginary axis. Studying this curve is the key idea behind the D-decomposition approach [7]. The curve partitions the plane (k_1, k_2) into regions in which the number of stable roots of $p(s, k)$ remains constant. The union of regions for which this number is equal to the degree of $p(s, k)$ is the stability region S . Hence the boundary of S is included in curve C .

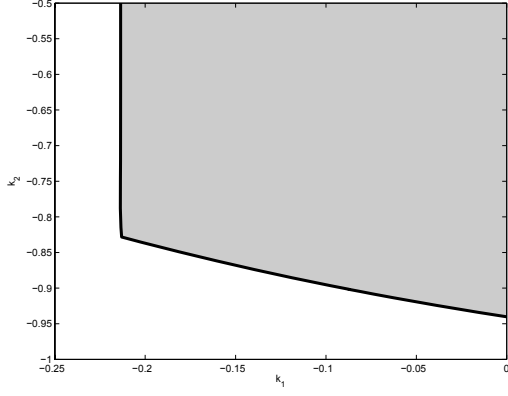


Figure 3: AC17.

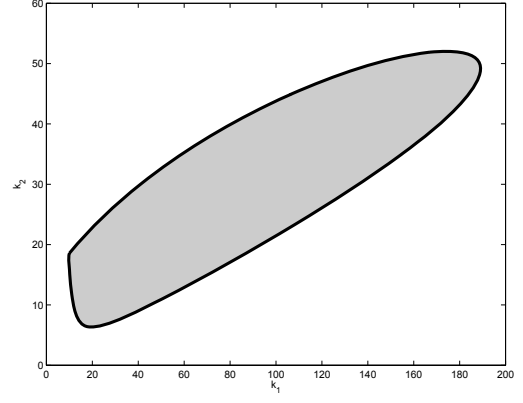


Figure 5: NN5.

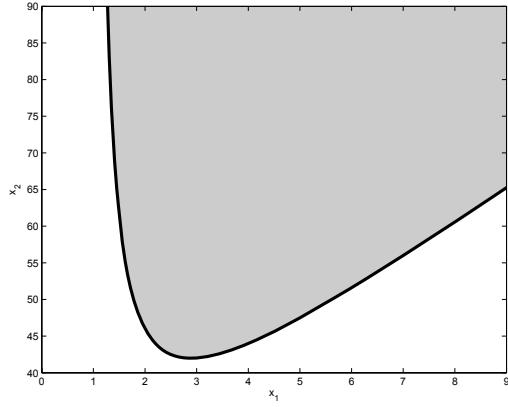


Figure 4: NN1.

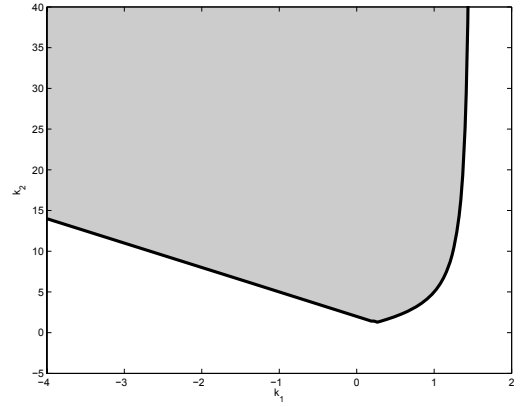


Figure 6: NN17.

Note that $p(j\omega, k) = 0$ for some $w \in \mathbb{R}$ if and only if

$$\begin{aligned} p_R(\omega^2, k) &= p_{0R}(\omega^2) + k_1 p_{1R}(\omega^2) + k_2 p_{2R}(\omega^2) = 0 \\ \omega p_I(\omega^2, k) &= \omega p_{0I}(\omega) + k_1 \omega p_{1I}(\omega) + k_2 \omega p_{2I}(\omega) = 0 \end{aligned}$$

Recall that we denote by $r_\omega(q_1, q_2)$ the resultant of polynomials $q_1(\omega), q_2(\omega)$ obtained by eliminating the scalar indeterminate ω . From the definition of the Hermite matrix, it holds

$$h(k) = r_\omega(p_R(\omega^2, k), \omega p_I(\omega^2, k)) = \det H(k) \quad (2)$$

from which the implicit algebraic description

$$C = \{k : h(k) = 0\}$$

follows.

LEMMA 2. *The determinant of the Hermite matrix can be factored as*

$$h(k) = l(k)g(k)^2$$

where $l(k)$ is affine, and $g(k)$ is a generically irreducible polynomial.

Proof: The result follows from basic properties of resultants:

$$\begin{aligned} h(k) &= r_\omega(p_R(\omega^2, k), \omega p_I(\omega^2, k)) \\ &= r_\omega(p_R(\omega^2, k), \omega^2) r_\omega(p_R(\omega^2, k), p_I(\omega^2, k)) \\ &= r_\omega(p_R(\omega, k), \omega) r_\omega(p_R(\omega, k), p_I(\omega, k))^2. \end{aligned}$$

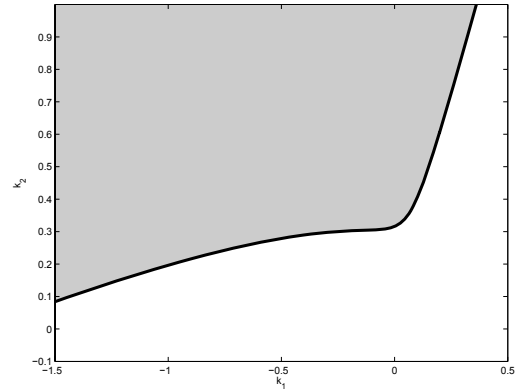


Figure 7: HE1.

Take $g(k) = r_\omega(p_R(\omega, k), p_I(\omega, k))$. Since $p_R(\omega, k)$ is affine in k it follows that $l(k) = r_\omega(p_R(\omega, k), \omega)$ is affine in k . $\square$

The curve can therefore be decomposed as the union of a line and a simpler algebraic curve

$$C = L \cup G = \{k : l(k) = 0\} \cup \{k : g(k) = 0\}.$$

The equation of line L was already given in the proof of Lemma 2,

namely

$$l(k) = r_\omega(p_R(\omega), \omega) = p_R(0, k) = p_{0R}(0) + k_1 p_{1R}(0) + k_2 p_{2R}(0).$$

The defining polynomial of the other curve component G can be obtained via the formula

$$g(k) = r_\omega(p_R(\omega, k), p_I(\omega, k)).$$

From the relations

$$\begin{bmatrix} p_{1R}(\omega^2) & p_{2R}(\omega^2) \\ \omega p_{1I}(\omega^2) & \omega p_{2I}(\omega^2) \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = - \begin{bmatrix} p_{0R}(\omega^2) \\ \omega p_{0I}(\omega^2) \end{bmatrix}$$

we derive a rational parametrization of G :

$$\begin{aligned} \begin{bmatrix} k_1(\omega^2) \\ k_2(\omega^2) \end{bmatrix} &= \frac{\begin{bmatrix} p_{2I}(\omega^2) & -p_{2R}(\omega^2) \\ -p_{1I}(\omega^2) & p_{1R}(\omega^2) \end{bmatrix}}{p_{1I}(\omega^2)p_{2R}(\omega^2) - p_{1R}(\omega^2)p_{2I}(\omega^2)} \begin{bmatrix} p_{0R}(\omega^2) \\ p_{0I}(\omega^2) \end{bmatrix} \\ &= \begin{bmatrix} \frac{q_1(\omega^2)}{q_0(\omega^2)} \\ \frac{q_2(\omega^2)}{q_0(\omega^2)} \end{bmatrix} \end{aligned} \quad (3)$$

which is well-defined since by assumption $p_1(s)/p_2(s)$ is not a constant. From this parametrization we can derive a symmetric linear determinantal form of the implicit equation of this curve.

LEMMA 3. *The symmetric affine pencil*

$$G(k) = B_\omega(q_1, q_2) + k_1 B_\omega(q_2, q_0) + k_2 B_\omega(q_1, q_0).$$

is such that $G = \{k : \det G(k) = 0\}$.

Proof: Rewrite the system of equations (3) as

$$\begin{aligned} a(\omega^2, k) &= q_1(\omega^2) - k_1 q_0(\omega^2) = 0 \\ b(\omega^2, k) &= q_2(\omega^2) - k_2 q_0(\omega^2) = 0 \end{aligned}$$

and use the Bézoutian resultant to eliminate indeterminate ω and obtain conditions for a point (k_1, k_2) to belong to the curve. The Bézoutian matrix is $B_\omega(a, b) = B_\omega(q_1 - k_1 q_0, q_2 - k_2 q_0) = B_\omega(q_1, q_2) + k_1 B_\omega(q_2, q_0) + k_2 B_\omega(q_1, q_0)$. Linearity in k follows from bilinearity of the Bézoutian and the common factor q_0 . $\square$

Finally, let $C(k) = \text{diag}\{l(k), G(k)\}$ so that curve C can be described as a determinantal locus

$$C = \{k : \det C(k) = 0\}.$$

5. LMI FORMULATION

Curve C partitions the plane into several connected components, that we denote by S_i for $i = 1, \dots, N$.

LEMMA 4. *If $C(k) \succ 0$ for some point k in the interior of S_i for some i then $S_i = \{k : C(k) \succeq 0\}$ is a convex LMI region.*

Proof: Follows readily from the affine dependence of $C(k)$ on k and from the fact that the boundary of S_i is included in C . $\square$

Convex sets which admit an LMI representation are called rigidly convex in [9]. Rigid convexity is stronger than convexity. It may happen that S_i is convex for some i , yet $C(k)$ is not positive definite for points k within S_i .

LEMMA 5. *Stability region S is the union of sets S_i containing points k such that $H(k) \succ 0$.*

Proof: Follows readily from Lemma 1. $\square$

Note that it may happen that S_i is convex LMI for some i , yet $H(k)$ is not positive definite for points k within S_i .

COROLLARY 1. *If $H(k) \succ 0$ and $C(k) \succ 0$ for some point k in the interior of S_i , then S_i is an LMI region included in the stability region.*

Proof: Combine Lemmas 4 and 5. $\square$

Quite often, on practical instances, we observe that $S = S_i$ for some i is a convex LMI region.

Practically speaking, once curve C is expressed as a determinantal locus, the search of points k such that $C(k) \succ 0$ can be formulated as an eigenvalue problem, but this is out of the scope of this paper.

6. EXAMPLES

6.1 Example 1

As mentioned in [7], Vishnegradsky in 1876 considered the polynomial $p(s, k) = s^3 + k_1 s^2 + k_2 s + 1$ and concluded that its stability region $S = \{k : k_1 > 0, k_1 k_2 > 1\}$ is convex hyperbolic.

The Hermite matrix of $p(s, k)$ is given by

$$H(k) = \begin{bmatrix} k_2 & 0 & 1 \\ 0 & k_1 k_2 - 1 & 0 \\ 1 & 0 & k_1 \end{bmatrix}$$

and hence after a row and column permutation the quadratic matrix inequality formulation

$$S = \{k : \begin{bmatrix} k_2 & 1 & 0 \\ 1 & k_1 & 0 \\ 0 & 0 & k_1 k_2 - 1 \end{bmatrix} \succ 0\}$$

explains why the region is convex. Indeed, the determinant of the 2-by-2 upper matrix, affine in k , is equal to the remaining diagonal entry, which is here redundant. The stability region can therefore be modeled as the LMI

$$S = \{k : \begin{bmatrix} k_1 & 1 \\ 1 & k_2 \end{bmatrix} \succ 0\}$$

see Figure 8.

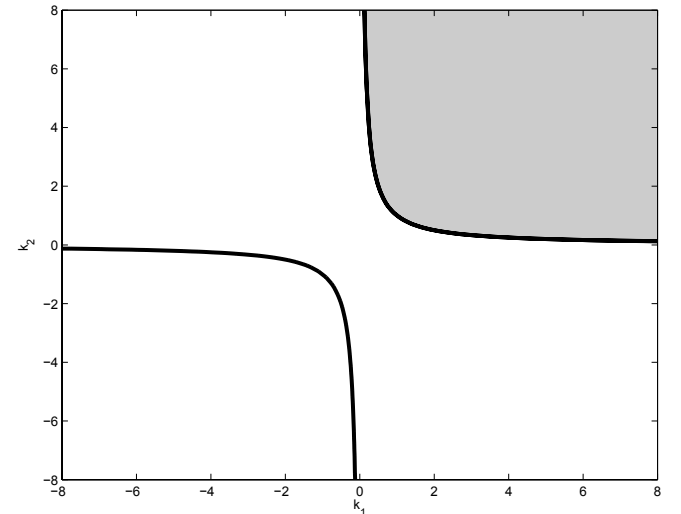


Figure 8: Vishnegradsky's degree 3 polynomial. Rational curve C with convex LMI stability region S (in gray)

6.2 Example 2

Consider problem NN1 from [10], for which $p_0(s) = s(s^2 - 13)$, $p_1(s) = s(s - 5)$, $p_2(s) = s + 1$ in (1). We have $p_R(\omega^2) = -k_1\omega^2 + k_2$ and $p_I(\omega^2) = -\omega^2 - 13 - 5k_1 + k_2$ and

$$H(k) = \begin{bmatrix} k_2(-13 - 5k_1 + k_2) & 0 & -k_2 \\ 0 & k_1(-13 - 5k_1 + k_2) - k_2 & 0 \\ -k_2 & 0 & k_1 \end{bmatrix}.$$

Hence $h(k) = \det H(k) = k_2(-13k_1 - k_2 - 5k_1^2 + k_1k_2)^2$ and then $l(k) = k_2$, $g(k) = -13k_1 - k_2 - 5k_1^2 + k_1k_2$. A rational parametrization of the curve $G = \{k : g(k) = 0\}$ is given by

$$\begin{aligned} k_1(\omega^2) &= (\omega^2 + 13)/(\omega^2 - 5) \\ k_2(\omega^2) &= \omega^2(\omega^2 + 13)/(\omega^2 - 5) \end{aligned}$$

from which we derive the symmetric affine determinantal representation $G = \{k : \det G(k) = 0\}$ with

$$G(k) = \begin{bmatrix} 169 + 65k_1 - 18k_2 & 13 + 5k_1 \\ 13 + 5k_1 & 1 - k_1 \end{bmatrix}.$$

The pencil representing C is therefore

$$C(k) = \begin{bmatrix} k_2 & 0 & 0 \\ 0 & 169 + 65k_1 - 18k_2 & 13 + 5k_1 \\ 0 & 13 + 5k_1 & 1 - k_1 \end{bmatrix}.$$

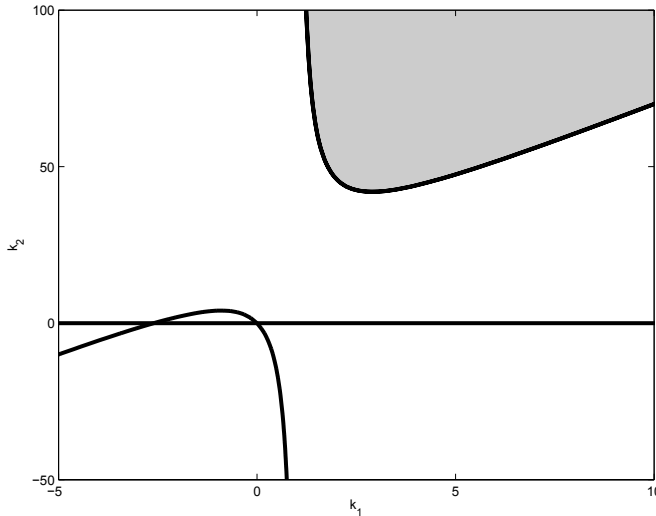


Figure 9: NN1 SOF example. Rational curve C with convex LMI stability region S (in gray)

We can check that $S = \{k : C(k) \succ 0\}$ is a convex LMI formulation of the stability region represented on Figure 4. Compare with Figure 9 where we represent also the curve $\tilde{C} = \{k : \det C(k) = 0\}$.

6.3 Example 3

This example, originally from Francis (1987), is also described in [7]. A SISO plant $(s - 1)(s - 2)/(s + 1)(s^2 + s + 1)$ must be stabilized with a PI controller $k_1 + k_2/s$. Equivalently, $p_0(s) = s(s + 1)(s^2 + s + 1)$, $p_1(s) = s(s - 1)(s - 2)$ and $p_2(s) = (s - 1)(s - 2)$ in (1).

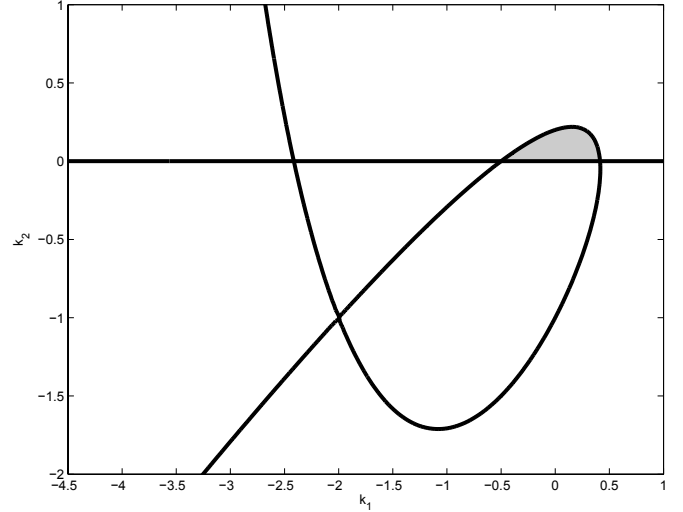


Figure 10: Francis' example. Rational curve C with convex LMI stability region S (in gray)

For these values we obtain $C(k) = \text{diag}\{l(k), G(k)\}$ with $l(k) = 2k_2$ and

$$G(k) = \begin{bmatrix} 14 + 28k_1 - 54k_2 & -20 - 40k_1 + 18k_2 & 2 + 4k_1 \\ -20 - 40k_1 + 18k_2 & 77 - 53k_1 + 36k_2 & -11 + 5k_1 \\ 2 + 4k_1 & -11 + 5k_1 & 5 + k_1 \end{bmatrix}$$

and the LMI stability region $S = \{k : C(k) \succ 0\}$ represented on Figure 10 together with the quartic curve $C = \{k : \det C(k) = 0\}$.

6.4 Example 4

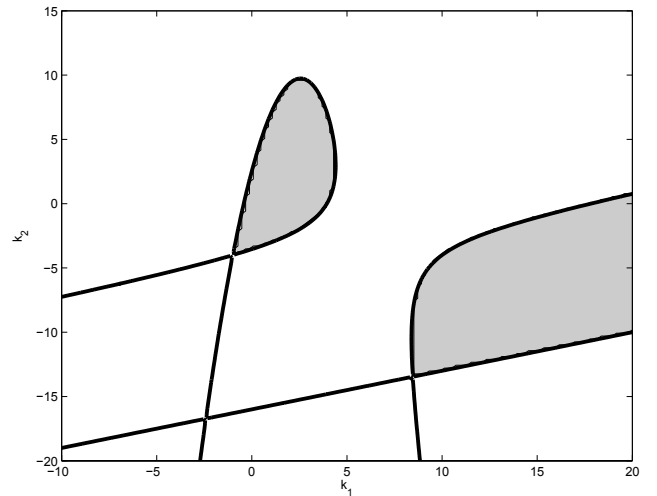


Figure 11: Ackermann's example with $a = 1$. Rational curve C and stability region S consisting of two disconnected regions (in gray), the one including the origin being LMI.

Consider [1, Example 14.4] for which $p_0(s) = s^4 + 2s^3 + 10s^2 + 10s + 14 + 2a$, $p_1(s) = 2s^3 + 2s - 3/10$, $p_2(s) = 2s + 1$, with $a \in \mathbb{R}$ a parameter.

We obtain $C(k) = \text{diag} \{I(k), G(k)\}$ with

$$\begin{aligned} I(k) &= 140 + 20a - 3k_1 + 10k_2 \\ G_{1,1}(k) &= 7920 + 4860a + 400a^2 + (-1609 - 60a)k_1 \\ &\quad + (-270 + 200a)k_2 \\ G_{2,1}(k) &= -8350 - 2000a + 1430k_1 + 130k_2 \\ G_{2,2}(k) &= 8370 - 1230k_1 - 100k_2 \\ G_{3,1}(k) &= 900 + 200a - 130k_1 \\ G_{3,2}(k) &= -900 + 100k_1 \\ G_{3,3}(k) &= 100 \end{aligned}$$

with $G_{i,j}(k)$ denoting the (i, j) entry in $G(k)$, unspecified entries being zero or deduced by symmetry.

When $a = 1$, the stability region consists of two disconnected components. The one containing the origin $k_1 = k_2 = 0$ is the LMI region $\{k : C(k) \succ 0\}$, see Figure 11.

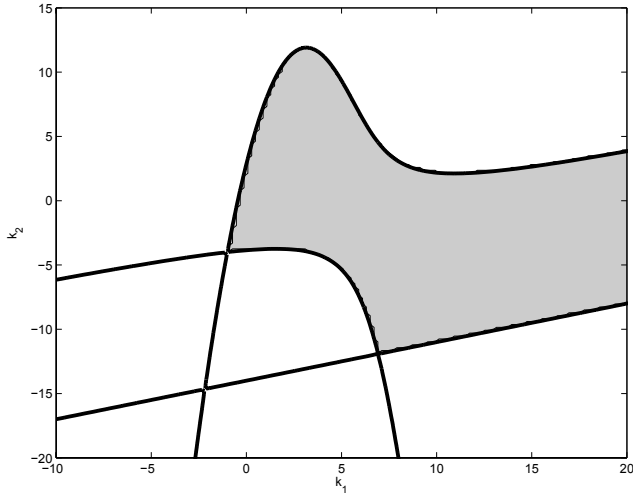


Figure 12: Ackermann's example with $a = 0$. Rational curve C and non-convex stability region S (in gray).

When $a = 0$, the stability region S is the non-convex region represented on Figure 12. The LMI region $\{k : C(k) \succ 0\}$ is not included in S in this case.

7. CONCLUSION

In this paper we have explained why the planar stability region of a polynomial may be convex with an explicit LMI representation. This is an instance of hidden convexity of a set which is otherwise described by intersecting (generally non-convex) Routh-Hurwitz minors sublevel sets or by enforcing positive definiteness of a (generally non-convex) quadratic Hermite matrix.

Practically speaking, optimizing a closed-loop performance criterion over an LMI formulation of the stability region is much simpler than optimizing over the non-linear formulation stemming from the Routh-Hurwitz minors or the Hermite quadratic matrix inequality.

Convexity in the parameter space was already exploited in [5, 2] in the context of PID controller design. It was shown that when the proportional gain is fixed, the set of integral and derivative gains is a union of a finite number of polytopes.

Extension of these ideas to the case of more than 2 parameters seems to be difficult. The problem of finding a symmetric affine determinantal representation of rationally parametrized surfaces or

hypersurfaces is not yet well understood, to the best of our knowledge. For example, in the simplest third degree case $p(s, k) = s^3 + k_1 s^2 + k_2 s + k_3$, how could we find four symmetric real matrices A_0, A_1, A_2, A_3 satisfying $\det(A_0 + A_1 k_1 + A_2 k_2 + A_3 k_3) = k_1 k_3 - k_2$?

8. ACKNOWLEDGMENTS

This work was partly supported by project No. MSM6840770038 of the Ministry of Education of the Czech Republic.

9. REFERENCES

- [1] J. Ackermann et al. *Robust control: the parameter space approach*. Springer, 1993.
- [2] J. Ackermann, D. Kaesbauer. Stable polyhedra in parameter space. *Automatica*, 39:937–943, 2003.
- [3] S. Basu, R. Pollack, M.-F. Roy. *Algorithms in real algebraic geometry*. Springer, 2003.
- [4] J. V. Burke, D. Henrion, A. S. Lewis, M. L. Overton. Stabilization via nonsmooth, nonconvex optimization. *IEEE Trans. Autom. Control*, 51(11):1760–1769, 2006.
- [5] A. Datta, A. Ho, S. P. Bhatthacharyya. *Structure and synthesis of PID controllers*. Springer, 2000.
- [6] M. Elkadi, B. Mourrain. *Introduction à la résolution des systèmes polynomiaux*. Springer, 2007.
- [7] E. N. Gryazina, B. T. Polyak. Stability regions in the parameter space: D-decomposition revisited. *Automatica*, 42(1):13–26, 2006.
- [8] J. W. Helton, P. A. Parrilo, M. Putinar. *Theory and algorithms of linear matrix inequalities - questions and discussions of the literature*. Compiled by S. Prajna. The American Institute of Mathematics, Palo Alto, USA, March 2006.
- [9] J. W. Helton, V. Vinnikov. Linear matrix inequality representation of sets. *Comm. Pure Applied Math.* 60(5):654–674, 2007.
- [10] F. Leibfritz. *COMpleib: constraint matrix optimization problem library - a collection of test examples for nonlinear semidefinite programs, control system design and related problems*. Technical Report, Univ. Trier, Germany, 2004.
- [11] P. C. Parks, V. Hahn. *Stability theory*. Prentice Hall, 1993. Original version published in German in 1981.
- [12] D. D. Šiljak. Parameter space methods for robust control design: a guided tour. *IEEE Trans. Autom. Control*, 34(7):674–688, 1989.
- [13] R. Tempo, G. Calafiore, F. Dabbene. *Randomized algorithms for analysis and control of uncertain systems*. Springer, 2005.

An Algorithm for Finding Symmetric Gröbner Bases in Infinite Dimensional Rings

[Extended Abstract]

Matthias Aschenbrenner
Department of Mathematics
University of California
Los Angeles, CA 90095
matthias@math.ucla.edu

Christopher J. Hillar
Department of Mathematics
Texas A&M University
College Station, TX 77843
chillar@math.tamu.edu

ABSTRACT

A symmetric ideal $I \subseteq R = K[x_1, x_2, \dots]$ is an ideal that is invariant under the natural action of the infinite symmetric group. We give an explicit algorithm to find Gröbner bases for symmetric ideals in the infinite dimensional polynomial ring R . This allows for symbolic computation in a new class of rings. In particular, we solve the ideal membership problem for symmetric ideals of R .

Categories and Subject Descriptors

G.4 [Mathematical Software]: Discrete Mathematics—algorithm design and analysis; G.2.m [Discrete Mathematics]: [miscellaneous]; J.2 [Physical Sciences and Engineering]: [chemistry, mathematics and statistics]

General Terms

Algorithms, Theory

Keywords

Invariant ideal, partial ordering, symmetric group, Gröbner basis, polynomial reduction, algorithm

1. INTRODUCTION

In computational algebra, one encounters the following general problem.

PROBLEM 1.1. *Let I be an ideal of a ring R and let $f \in R$. Determine whether $f \in I$.*

When $R = K[x_1, \dots, x_n]$ is a polynomial ring in n indeterminates over a field K , this problem has a complete solution due to Buchberger [2] (for a nice exposition, see [4, 3]).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

THEOREM 1.2 (BUCHBERGER). *Let $I = \langle f_1, \dots, f_m \rangle_R$ be an ideal of $R = K[x_1, \dots, x_n]$. Then, there is a computable, finite set of polynomials G such that for every polynomial f , we have $f \in I$ if and only if the polynomial reduction of f with G is 0.*

One remarkable feature of this result is that once such a Gröbner basis G for I is found, any new instance of the question “Is $f \in I$?” can be solved very quickly in principle (of course, in practice, there are many issues involving the coefficient heights of the polynomials involved). It is difficult not to stress the importance of Theorem 1.2; it forms the backbone of the field of computational algebraic geometry and has many applications, too numerous to list here. We should mention that there have been various improvements to Buchberger’s algorithm. Currently, the algorithm of Faugere [6] is generally regarded as the fastest.

We shall consider a different but related membership problem; one that at first glance would not seem to be solvable as completely as Buchberger had done with $K[x_1, \dots, x_n]$. Let $X = \{x_1, x_2, \dots\}$ be an infinite collection of indeterminates, indexed by the positive integers, and let $\mathfrak{S}_\infty$ be the group of permutations of X . For a positive integer N , we will also let $\mathfrak{S}_N$ denote the set of permutations of $\{1, \dots, N\}$. Fix a field K and let $R = K[X]$ be the polynomial ring in the indeterminates X . The group $\mathfrak{S}_\infty$ acts naturally on R : if $\sigma \in \mathfrak{S}_\infty$ and $f \in K[x_1, \dots, x_n]$, then

$$\sigma f(x_1, \dots, x_n) = f(x_{\sigma 1}, \dots, x_{\sigma n}) \in R. \quad (1)$$

We motivate our discussion with the following concrete problem. Questions of this nature arise in applications to chemistry [8, 9, 10] and algebraic statistics [5].

PROBLEM 1.3. *Let $f_1 = x_1^3 x_3 + x_1^2 x_2^3$ and $f_2 = x_2^2 x_3^2 - x_2^2 x_1 + x_1 x_3^2$ and consider the ideal of $R = K[X]$ generated by all permutations of f_1 and f_2 :*

$$I = \langle \mathfrak{S}_\infty f_1, \mathfrak{S}_\infty f_2 \rangle_R.$$

Is the following polynomial with 10 indeterminates in I ?

$$\begin{aligned} f = & -x_{10}^2x_9x_5^6 - 2x_{10}^2x_9x_8^3x_5^5 - x_{10}^2x_8^6x_5^4 + 3x_{10}^2x_8^2 + 3x_{10}^2x_7 \\ & + 3x_{10}x_9x_7x_4^3x_3^2x_2^2x_1 + 3x_{10}x_9x_7x_4^3x_3^2x_1^2 \\ & - 3x_{10}x_9x_7x_4^3x_2^2x_1^2 - x_9^2x_8^7x_7x_6x_5^6 - 2x_9x_8^{10}x_7x_6x_5^5 \\ & + x_9x_5^3x_3x_2x_1^3 + x_9x_5^3x_2^4x_1^2 + x_9x_3x_2^3x_1^4 + x_9x_2^6x_1^3 \\ & - x_8^{13}x_7x_6x_5^4 - 3x_8^2x_7 + x_7^2x_6x_3^3x_2^7 + x_7^2x_6x_3^3x_2^5x_1 \\ & - x_7^2x_6x_3x_2^7x_1 + x_5x_4^2 - 3x_5x_3^2 + 2x_5x_1^2 + x_4^2x_3^2 - 2x_3^2x_1^2 \\ & + 5x_3x_1^5 + 5x_2^3x_1^4. \end{aligned}$$

More generally, given $f \in R$, how can we determine if $f \in I$?

Naively, one could solve this problem using Buchberger's algorithm with truncated polynomial rings $R_n = K[x_1, \dots, x_n]$. Namely, for each $n \geq 10$, compute a Gröbner basis G_n for the ideal $I_n = \langle \mathfrak{S}_n f_1, \mathfrak{S}_n f_2 \rangle_{R_n}$, and reduce f by G_n .

There are several problems with this approach. For one, this method requires computation of many Gröbner bases (the bottleneck in any symbolic computation), the number of which depends on the number of indeterminates appearing in f . Additionally, it lacks the ability to solve new membership problems quickly, a powerful feature of Buchberger's technique. One might hope to at least restrict the number of Gröbner basis computations in terms of the number of indeterminates appearing in f , however, the following simple example should temper one's optimism a little.

EXAMPLE 1.4. Let I be the ideal generated by all permutations of $x_1 + x_2$. Then, $I = \langle x_1, x_2, \dots \rangle_R$, but

$$x_1 \notin \langle x_1 + x_2 \rangle_{K[x_1, x_2]}.$$

Our main result in this paper is an effective algorithm that solves the general membership problem for symmetric ideals (such as those appearing in Problem 1.3) and has all of the important features of Buchberger's method. It is the first algorithm of its kind that we are aware of (although it is similar in spirit to Buchberger's original algorithm). Before we state our theorem explicitly (Theorem 1.6), we develop some notation. In general, we first give the main ideas in the text informally and clarify the notions later.

Let $R[\mathfrak{S}_\infty]$ denote the (left) group ring of $\mathfrak{S}_\infty$ over R with multiplication given by $f\sigma \cdot g\tau = fg(\sigma\tau)$ for $f, g \in R$ and $\sigma, \tau \in \mathfrak{S}_\infty$, and extended by linearity. The action (1) naturally gives R the structure of a (left) module over the ring $R[\mathfrak{S}_\infty]$. For instance, we have

$$[x_1(12) + x_2(23)] \cdot (x_1x_3 + x_2) = x_1x_2x_3 + x_1^2 + x_1x_2^2 + x_2x_3.$$

An ideal $I \subseteq R$ is called *symmetric* if

$$\mathfrak{S}_\infty I := \{\sigma f : \sigma \in \mathfrak{S}_\infty, f \in I\} \subseteq I.$$

Symmetric ideals are then simply the $R[\mathfrak{S}_\infty]$ -submodules of R .

Also, for the purposes of this work, we will use the following notation. Let B be a ring and let G be a subset of a B -module M . Then $\langle f : f \in G \rangle_B$ will denote the B -submodule of M generated by the elements of G . This notation greatly simplifies expressing symmetric ideals in terms of their generators.

EXAMPLE 1.5. $I = \langle x_1, x_2, \dots \rangle_R$ is an invariant ideal of R . Written as a module over the group ring $R[\mathfrak{S}_\infty]$, it has the compact presentation $I = \langle x_1 \rangle_{R[\mathfrak{S}_\infty]}$.

We may now state our main theorem.

THEOREM 1.6. Let $I = \langle f_1, \dots, f_m \rangle_{R[\mathfrak{S}_\infty]}$ be a symmetric ideal of R . Then, there is a computable, finite set of polynomials G such that for every polynomial f , we have $f \in I$ if and only if the polynomial reduction of f with G is 0.

We should remark here that the polynomial reduction appearing in Theorem 1.6 is only a slight modification of the reduction in the context of normal (finite dimensional) polynomial rings. We will also call the sets G appearing above *Gröbner bases* for reasons which will be evident in the section that follows.

EXAMPLE 1.7. The ideal $I = \langle x_1^3x_3 + x_1^2x_2^3, x_2^2x_3^2 - x_2^2x_1 + x_1x_3^2 \rangle_{R[\mathfrak{S}_\infty]}$ from Problem 1.3 has a Gröbner basis given by:

$$G = \mathfrak{S}_3 \cdot \{x_3x_2x_1^2, x_3^2x_1 + x_2^4x_1 - x_2^2x_1, x_3x_1^3, x_2x_1^4, x_2^2x_1^2\}.$$

Once G is found, testing whether a polynomial f is in I can be done using the reduction algorithm found in Section 4; for instance, one finds that $f \in I$ for the polynomial encountered in Problem 1.3. $\square$

In Section 2, we discuss the history of this problem and state some of the foundational results that are ingredients in the proof of Theorem 1.6. In particular, we discuss there an important partial order on monomials that respects the action of the symmetric group. Section 3 briefly reviews the notion of reduction that occurs in our more general context, and finally, in Section 4, we describe our algorithm. To keep the paper as expository as possible, we have left out many of the (technical) proofs that will appear in a much longer version of this paper.

2. GRÖBNER BASES FOR SYMMETRIC IDEALS

The following was proved recently in [1]. It says that while ideals of $R = K[X]$ are too big in general, those with extra structure have finite presentations.

THEOREM 2.1. Every symmetric ideal of R is finitely generated as an $R[\mathfrak{S}_\infty]$ -module. In other words, R is a Noetherian $R[\mathfrak{S}_\infty]$ -module.

REMARK 2.2. Symmetric ideals can be arbitrarily complex in the following sense. For each n , there are symmetric ideals of R that cannot have fewer than n $R[\mathfrak{S}_\infty]$ -module generators [7]. Moreover, such ideals are not always monomial.

Theorem 2.1 was motivated by finiteness questions in chemistry [8, 9, 10] and algebraic statistics [5] involving chains of symmetric ideals I_k ($k = 1, 2, \dots$) contained in finite dimensional polynomial rings R_k . We refer the reader to [1] for more details.

In the course of proving Theorem 2.1, it was shown that, in a certain sense, a symmetric ideal I has a finite minimal Gröbner basis (see below for a review of these concepts). Moreover, the existence of such a set of generators solves the ideal membership problem in R .

THEOREM 2.3. Let G be a Gröbner basis for a symmetric ideal I . Then $f \in I$ if and only if f has normal form 0 with respect to G .

The normal form reduction we are talking about here is a modification of the standard notion in polynomial theory and Gröbner bases; we describe it in more detail below. Unfortunately, the techniques used to prove finiteness in [1] are nonconstructive and therefore do not give methods for computing Gröbner bases in R . Our main result is an algorithm for finding these bases.

THEOREM 2.4. *Let $I = \langle f_1, \dots, f_m \rangle_{R[\mathfrak{S}_\infty]}$ be a symmetric ideal of R . There exists an effective algorithm to compute a finite minimal Gröbner basis for I .*

COROLLARY 2.5. *There exists an effective algorithm to solve the ideal membership problem for symmetric ideals in the infinite dimensional ring $K[x_1, x_2, \dots]$.*

The following is a brief review of the Gröbner basis theory for symmetric ideals (see [1] for more details). Let us first note that an infinite permutation acting on a polynomial may be replaced with a finite one.

LEMMA 2.6. *Let $\sigma \in \mathfrak{S}_\infty$ and $f \in R$. Then there exists a positive integer N and $\tau \in \mathfrak{S}_N$ such that $\tau f = \sigma f$.*

Let Ω be the set of monomials in indeterminates $x_1, x_2, \dots$, including the constant monomial 1. Order the variables $x_1 < x_2 < \dots$, and let $\leq$ be the induced lexicographic (total) well-ordering of monomials. Given a polynomial $f \in R$, we set $\text{lm}(f)$ to be the leading monomial of f with respect to $\leq$ and $\text{lt}(f)$ to be its leading term. The following partial ordering on Ω respects the action of $\mathfrak{S}_\infty$ and refines the division partial order on Ω .

DEFINITION 2.7. *(The symmetric cancellation partial ordering)*

$$v \preceq w \quad :\Longleftrightarrow \quad \begin{cases} v \leq w \text{ and there exist } \sigma \in \mathfrak{S}_\infty \\ \text{such that } \sigma v | w \text{ and} \\ \sigma u \leq \sigma v \text{ for all } u \leq v. \end{cases}$$

REMARK 2.8. *A permutation σ in the definition need not be unique. Also, we say that such a permutation witnesses $v \preceq w$. We will give a more computationally useful description of this partial order in Theorem 2.21 below.*

EXAMPLE 2.9. *As an example of this relation, consider the following chain,*

$$x_1^3 \preceq x_1^2 x_2^3 \preceq x_1 x_2^2 x_3^3.$$

To verify the first inequality, notice that $x_1^2 x_2^3 = x_1^2 \sigma(x_1^3)$, in which σ is the transposition (12). If $u = x_1^{u_1} \dots x_n^{u_n} \leq x_1^3$, then it follows that $n = 1$ and $u_1 \leq 3$. In particular, $\sigma u = x_2^{u_1} \leq x_2^3 = \sigma x_1^3$. Verification of the other inequality is similar.

Alternatively, one may use Lemmas 2.12, 2.13, and 2.14 to produce these and many other examples of such relations. $\square$

Although this partial order appears technical, it can be reconstructed from the following two properties. The first one says that the leading monomial of σf is the same as $\sigma \text{lm}(f)$ whenever σ is a witness to a relation involving $\text{lm}(f)$, while the latter can be viewed as a kind of “ S -pair” leading term cancellation.

LEMMA 2.10. *Let f be a nonzero polynomial and $w \in \Omega$. Suppose that $\sigma \in \mathfrak{S}_\infty$ witnesses $\text{lm}(f) \preceq w$, and let $u \in \Omega$ with $u \sigma \text{lm}(f) = w$. Then $\text{lm}(u \sigma f) = u \sigma \text{lm}(f)$.*

LEMMA 2.11. *Suppose that $m_1 \preceq m_2$ and f_1, f_2 are two polynomials with lexicographic leading monomials m_1 and m_2 , respectively. Then there exists a permutation σ and $0 \neq c \in K$ such that*

$$f_2 - c \frac{m_2}{\sigma m_1} \sigma f_1$$

consists of monomials (lexicographically) smaller than m_2 .

The following two lemmas allow us to generate many relations, including the ones in the above example. Proofs can also be found in [1].

LEMMA 2.12. *Suppose that $x_1^{a_1} \dots x_n^{a_n} \preceq x_1^{b_1} \dots x_n^{b_n}$ where $a_i, b_j \in \mathbb{N}$, $b_n > 0$. Then for any $c \in \mathbb{N}$, we have $x_1^{a_1} \dots x_n^{a_n} \preceq x_1^c x_2^{b_1} \dots x_{n+1}^{b_n}$.*

LEMMA 2.13. *Suppose that $x_1^{a_1} \dots x_n^{a_n} \preceq x_1^{b_1} \dots x_n^{b_n}$, where $a_i, b_j \in \mathbb{N}$, $b_n > 0$. Then for any $a, b \in \mathbb{N}$ such that $a \leq b$, we have $x_1^a x_2^{a_1} \dots x_{n+1}^{a_n} \preceq x_1^b x_2^{b_1} \dots x_{n+1}^{b_n}$.*

The next fact is essentially a consequence of [1, Lemma 2.14].

LEMMA 2.14. *Let $u, v \in \Omega$ and set n to be the largest index of indeterminates appearing in v . If $u \preceq v$, then there is a witness $\sigma \in \mathfrak{S}_n$, and if $a, b \in \mathbb{N}$ are such that $a \leq b$, then $u x_{n+1}^a \preceq v x_{n+1}^b$.*

In this setting, we need a notion of leading monomials of a set of polynomials that interacts with the symmetric group action. For a set of polynomials I , we define

$$\text{lm}(I) = \langle w \in \Omega : \text{there exists } 0 \neq f \in I \text{ with } \text{lm}(f) \preceq w \rangle_K,$$

the span of all monomials which are $\preceq$ larger than leading monomials in I . If I happens to be a symmetric ideal, then it follows from Lemma 2.10 that

$$\text{lm}(I) = \langle \text{lm}(f) : f \in I \rangle_K$$

corresponds to a more familiar set of monomials. With these preliminaries in place, we state the following definition from [1].

DEFINITION 2.15. *We say that a subset B of a symmetric ideal $I \subseteq R$ is a Gröbner basis for I if $\text{lm}(B) = \text{lm}(I)$.*

Additionally, a Gröbner basis is called *minimal* if no leading monomial of an element in B is $\preceq$ smaller than any other leading monomial of an element in B . In analogy to the classical case, a Gröbner basis B generates the ideal I :

$$I = \langle B \rangle_{R[\mathfrak{S}_\infty]}.$$

The authors of [1] prove the following finiteness result for symmetric ideals; it is an analog to the corresponding statement for finite dimensional polynomial rings. As a corollary, they obtain Theorem 2.1.

THEOREM 2.16. *A symmetric ideal of R has a finite Gröbner basis.*

Although much of the intuition involving Gröbner bases from the finite dimensional case transfers over faithfully to the ring R , one needs to be somewhat careful in general. For example, monomial generators do not automatically form a

Gröbner basis for a symmetric ideal I (see Example 2.24 below). However, we do have a description of minimal Gröbner bases for monomial ideals, and this is the content of Theorem 2.22 below. To state it, we need to introduce a special class of permutations to give a more workable description of the symmetric cancellation partial order. This description will be used in our algorithm that finds symmetric Gröbner bases.

Fix a monomial $g = \mathbf{x}^{\mathbf{a}} = x_1^{a_1} \cdots x_n^{a_n}$. A *downward elementary shift* (resp. *upward elementary shift*) of g is a permutation σ which acts on $\mathbf{a}$ as transposition of two consecutive coordinates, the smaller (resp. larger) of which is zero. A *downward shift* (resp. *upward shift*) of g is a product of downward elementary shifts (resp. upward elementary shifts) that begin with g . A *shift permutation* of g is either a downward shift or an upward shift of g . If $g, h \in \Omega$ and σ is an upward shift of g with $h = \sigma g$, then we write $g \sim_{\sigma} h$. For example, $\sigma = (341)$ is an upward elementary shift of $g = x_2^3 x_3 x_5^2$ and $\tau = (32)(56)(341)$ is an upward shift of g ; in this case, $g \sim_{\tau} h$ for $h = x_3^3 x_4 x_6^2$.

The following fact should be clear.

LEMMA 2.17. *If $g \sim_{\sigma} h$ and $h \sim_{\tau} k$, then $g \sim_{\tau\sigma} k$.*

A more concrete description of these permutations is given by the following straightforward lemma, which follows directly from the definitions.

LEMMA 2.18. *Let g be a monomial, and let $i_1 < \cdots < i_n$ be those indices appearing in the indeterminates dividing g . Then σ is an upward shift permutation of g if and only if*

$$\sigma i_1 < \sigma i_2 < \cdots < \sigma i_n \text{ and } \sigma i_k \geq i_k, \quad k = 1, \dots, n.$$

The following fact gives a relationship between shift permutations and the symmetric cancellation partial order.

LEMMA 2.19. *Let g and h be monomials with $g \sim_{\sigma} h$ for some $\sigma \in \mathfrak{S}_{\infty}$. Then $g \preceq h$. Moreover, we have $h \sim_{\sigma^{-1}} g$.*

PROOF. By Lemma 2.17, we may suppose that σ as in the statement of the lemma acts on g by transposing x_i and x_{i+1} . Write $g = x_1^{a_1} \cdots x_i^{a_i} x_{i+2}^{a_{i+2}} \cdots x_n^{a_n}$ with $a_n > 0$; we must verify that

$$x_1^{a_1} \cdots x_i^{a_i} x_{i+2}^{a_{i+2}} \cdots x_n^{a_n} \preceq x_1^{a_1} \cdots x_{i-1}^{a_{i-1}} x_{i+1}^{a_i} x_{i+2}^{a_{i+2}} \cdots x_n^{a_n}.$$

This is proved by induction on n . When $n = 1$, we have $i = 1$, and the claim reduces to Lemma 2.12. In general, we have two cases to consider. If $i = n > 1$, then the claim follows from Lemma 2.13 and induction. Alternatively, if $i < n$ and $n > 1$, then we may apply Lemma 2.14 and induction. The second claim is clear from the definitions. $\square$

REMARK 2.20. *A word of caution is in order. Suppose that g and h are monomials with $g \sim_{\sigma} h$ for some $\sigma \in \mathfrak{S}_{\infty}$. Then it can happen that σ is not a witness for the (valid) relation $g \preceq h$. For example, if $\sigma = (14)(23)$, $g = x_2$, and $h = x_3$, then $g \sim_{\sigma} h$. However, the relation $x_1 \leq x_2$ does not imply $\sigma x_1 \leq \sigma x_2$ as one can easily check.*

We now state a new characterization of the symmetric cancellation partial order.

THEOREM 2.21. *Two monomials v and w satisfy $v \preceq w$ if and only if there is an upward shift $\sigma \in \mathfrak{S}_N$ of v such that $\sigma v \mid w$, where N is the largest index of indeterminates appearing in w .*

PROOF. We prove the only-if direction ($\Rightarrow$); the converse is clear from Lemma 2.19 and Definition 2.7. Let N be the largest index of indeterminates appearing in w . If $v \preceq w$, then there is a monomial m and a witness $\sigma \in \mathfrak{S}_N$ such that $w = m\sigma v$ by Lemma 2.14. For the rest of the argument, we fix this permutation σ . We will prove that σ is an upward shift of v using the characterization found in Lemma 2.18.

Write $v = x_{i_1}^{v_{i_1}} \cdots x_{i_n}^{v_{i_n}}$, in which $i_1 < \cdots < i_n$ are all the indices appearing in v . We prove the following claim by induction on the number of indeterminates n appearing in v :

$$(u \leq v \Rightarrow \sigma u \leq \sigma v \text{ for all } u \in \Omega) \\ \Rightarrow (\sigma i_1 < \cdots < \sigma i_n \text{ and } i_k \leq \sigma i_k \text{ for all } k \leq n). \quad (2)$$

The result in the theorem is then implied by Lemma 2.18. We take for our base case of induction $n = 0$ (so that $v = 1$), as the statement is vacuously true. Also, if $n = 1$ and $i_1 = 1$, then the statement is clear, so we suppose from now on that $i_n > 1$.

Fix a monomial v with $n + 1$ indeterminates; we must show that (2) holds. Therefore, assume that σ is such that $u \leq v \Rightarrow \sigma u \leq \sigma v$ for all $u \in \Omega$. For a positive integer c , consider the monomial $u_c = (x_1 \cdots x_{i_{n+1}-1})^c \leq v$. Since $u_c \leq v$, we have by assumption that

$$\sigma u = (x_{\sigma 1} \cdots x_{\sigma(i_{n+1}-1)})^c \leq x_{\sigma i_1}^{v_{i_1}} \cdots x_{\sigma i_{n+1}}^{v_{i_{n+1}}} = \sigma v.$$

If $\sigma i_{n+1} \leq \sigma i_j$ for some $j < n + 1$, then by choosing c sufficiently large (say, larger than the degree of v), the above inequality is impossible. Therefore, it follows that $\sigma i_j < \sigma i_{n+1}$ for all $j < n + 1$. Next, we show that $i_{n+1} \leq \sigma i_{n+1}$. Suppose by way of contradiction that $\sigma i_{n+1} < i_{n+1}$. Then, $\sigma i_j < i_{n+1}$ for all $j < n + 1$. In particular, $\sigma v < v$, and thus $\sigma^s v \leq \sigma v < v$ for all positive integers s . Hence, $v = \sigma^{N!} v < v$, a contradiction.

Our final step is to invoke the induction hypothesis and prove the other inequalities on the right-hand side of (2). Suppose that $u = x_1^{u_1} \cdots x_{i_n}^{u_{i_n}} \leq x_{i_1}^{v_{i_1}} \cdots x_{i_n}^{v_{i_n}}$ so that we have $u x_{i_{n+1}}^{v_{i_{n+1}}} \leq v$. By assumption,

$$\sigma(u x_{i_{n+1}}^{v_{i_{n+1}}}) = (\sigma u) x_{\sigma i_{n+1}}^{v_{i_{n+1}}} \leq x_{\sigma i_k}^{v_{i_k}} \cdots x_{\sigma i_{n+1}}^{v_{i_{n+1}}} = \sigma v,$$

and thus (since we are using the lexicographic ordering),

$$\sigma u \leq x_{\sigma i_1}^{v_{i_1}} \cdots x_{\sigma i_n}^{v_{i_n}}.$$

It follows from induction applied to the monomial $x_{i_1}^{v_{i_1}} \cdots x_{i_n}^{v_{i_n}}$ in n indeterminates that $\sigma i_1 < \cdots < \sigma i_n$ and $i_k \leq \sigma i_k$ for all $k \leq n$. This proves the claim and completes the proof of the theorem. $\square$

The main result of this section is the following.

THEOREM 2.22. *Let G be a finite set of monomials of the same degree, and let N be the largest index of indeterminates appearing in any monomial in G . Then $H = \mathfrak{S}_N G$ is a (finite) Gröbner basis for $I = \langle G \rangle_{R[\mathfrak{S}_{\infty}]}$. Moreover, if we let S be the set,*

$\{h \in H : \text{there exists } g \in H \setminus \{h\} \text{ and } \sigma \in \mathfrak{S}_N \text{ with } g \sim_{\sigma} h\}$
then $H \setminus S$ is a minimal Gröbner basis for I .

PROOF. Let G, H, S, N , and I be as in the statement of the theorem; we first show that H is a Gröbner basis for I . The inclusion $\text{lm}(H) \subseteq \text{lm}(I)$ is clear from the definition. So

suppose that $w \in \text{lm}(I)$ is a monomial; we must show that $h \preceq w$ for some $h \in H$. Set $w = u\sigma g$ for some monomial u , witness $\sigma \in \mathfrak{S}_\infty$, and $g \in G$. Since $\sigma g \preceq u\sigma g = w$, it suffices to show that $h \preceq \sigma g$ for some $h \in H$. Let τ be a downward shift that takes σg to a monomial h with indices at most N . Then h has the same type (its unordered vector of exponents) as g , and therefore there is a permutation $\gamma \in \mathfrak{S}_N$ such that $h = \gamma g$. It follows that $h \in H$ and $h \sim_{\tau^{-1}} \sigma g$ so that $h \preceq \sigma g$ by Lemma 2.19.

Next, we observe that $H \setminus S$ is still a Gröbner basis since $g \sim_\sigma h$ implies that $g \preceq h$. Therefore, it remains to prove that $H \setminus S$ is minimal. If $h, g \in H$ are related by $g \preceq h$, then $h = m\sigma g$ for a witness σ and a monomial m . Since each element of H has the same degree, we have $m = 1$. By Theorem 2.21, it follows that we may choose $\sigma \in \mathfrak{S}_N$ such that $g \sim_\sigma h$. Therefore, we are only removing unnecessary elements from the Gröbner basis H when we discard the monomials in S . This completes the proof. $\square$

COROLLARY 2.23. *Let G be a finite set of monomials, and let N be the largest index of indeterminates appearing in any monomial in G . Then $\mathfrak{S}_N G$ is a (not necessarily minimal) Gröbner basis for $I = \langle G \rangle_{R[\mathfrak{S}_\infty]}$.*

EXAMPLE 2.24. *The ideal $I = \langle x_1^2 x_3 \rangle_{R[\mathfrak{S}_\infty]}$ has a Gröbner basis,*

$$H = \{x_1 x_2^2, x_1 x_3^2, x_1^2 x_2, x_2 x_3^2, x_1^2 x_3, x_2^2 x_3\}.$$

However, it is not minimal. Removing those elements that are the result of upward shifts, we are left with the following minimal Gröbner basis for I : $\{x_1 x_2^2, x_1^2 x_2\}$. $\square$

3. REDUCTION OF POLYNOMIALS

Before describing our Gröbner basis algorithm, we must recall the ideas of reduction from [1]. Let $f \in R$, $f \neq 0$, and let B be a set of nonzero polynomials in R . We say that f is *reducible by B* if there exists $g \in B$ such that we have $\text{lm}(g) \preceq \text{lm}(f)$, witnessed by some $\sigma \in \mathfrak{S}_\infty$ and

$$\text{lt}(f) = aw\sigma \text{lt}(g)$$

for some nonzero $a \in K$ and a monomial $w \in \Omega$ such that $w\sigma \text{lm}(g) = \text{lm}(f)$. In this case we write $f \xrightarrow[B]{} h$, where

$$h = f - (aw\sigma g),$$

and we say that f *reduces to h by B* . We say that f is *reduced* with respect to B if f is not reducible by B . By convention, the zero polynomial is reduced with respect to B . Trivially, every element of B reduces to 0.

The smallest quasi-ordering on R extending the relation $\xrightarrow[B]{}^*$ is denoted by $\xrightarrow[B]{}^*$. If $f, h \neq 0$ and $f \xrightarrow[B]{}^* h$, then $\text{lm}(h) < \text{lm}(f)$, by Lemma 2.11. In particular, every chain

$$h_0 \xrightarrow[B]{} h_1 \xrightarrow[B]{} h_2 \xrightarrow[B]{} \dots$$

with all $h_i \in R \setminus \{0\}$ is finite. (Since the term ordering $\leq$ is well-founded.) Hence there exists $r \in R$ such that $f \xrightarrow[B]{}^* r$ and r is reduced with respect to B ; we call such an r a *normal form* of f with respect to B .

LEMMA 3.1. *Suppose that $f \xrightarrow[B]{}^* r$. Then there exist polynomials $g_1, \dots, g_n \in B$, $\sigma_1, \dots, \sigma_n \in \mathfrak{S}_\infty$ and $h_1, \dots, h_n \in$*

R such that

$$f = r + \sum_{i=1}^n h_i \sigma_i g_i \quad \text{and} \quad \text{lm}(f) \geq \max_{1 \leq i \leq n} \text{lm}(h_i \sigma_i g_i).$$

(In particular, $f - r \in \langle B \rangle_{R[\mathfrak{S}_\infty]}$.)

LEMMA 3.2. *Let I be a symmetric ideal of R and B be a set of nonzero elements of I . The following are equivalent:*

1. *B is a Gröbner basis for I .*
2. *Every nonzero $f \in I$ is reducible by B .*
3. *Every $f \in I$ has normal form 0. (In particular, $I = \langle B \rangle_{R[\mathfrak{S}_\infty]}$.)*
4. *Every $f \in I$ has unique normal form 0.*

PROOF. The implications $(1) \Rightarrow (2) \Rightarrow (3) \Rightarrow (4)$ are either obvious or follow from the remarks preceding the lemma. Suppose that (4) holds. Every $f \in I \setminus \{0\}$ with $\text{lt}(f) \notin \text{lt}(B)$ is reduced with respect to B , hence has two distinct normal forms (0 and f), a contradiction. Thus $\text{lt}(I) = \text{lt}(B)$. $\square$

4. DESCRIPTION OF THE ALGORITHM

We begin by describing a method that checks when two monomials are $\preceq$ comparable, returning a permutation (if it exists) witnessing the relation. This is accomplished using the characterization given by Theorem 2.21. In this regard, it will be useful to view monomials in R as vectors of integers $v = (v_1, v_2, \dots)$ with finite support in $\mathbb{N}^\infty$.

ALGORITHM 4.1. *(Comparing monomials in the symmetric cancellation order)*

Input: Two monomials v and w with largest indeterminate in w having index N .

*Output: A permutation $\sigma \in \mathfrak{S}_N$ if $v \preceq w$; otherwise, **false**.*

1. *Set $t := 1$, $\text{match} := \{\}$;*
2. *For $i = 1$ to N :*

For $j = t$ to N :

If $v_i \neq 0$ and $v_i \leq w_j$, then

$t := j + 1$;

$\text{match} := \text{match} \cup \{(i, j)\}$;

Break inner loop;

$t := \max\{i + 1, t\}$;

3. *If match contains fewer elements than the support of v , return **false**;*
4. *For $j = N$ down to 1:*

Set $i :=$ largest integer $\leq N$ not appearing as a first coordinate in match ;

If j is not a second coordinate in match , then $\text{match} := \text{match} \cup (i, j)$;

5. *Return the permutation that match represents;*

EXAMPLE 4.2. Consider the vectors $v = (1, 2, 0, 2)$ and $w = (0, 3, 4, 1)$ representing monomials $x_1^2 x_2^2 x_1$ and $x_4 x_3^4 x_2^3$ respectively. Then, Algorithm 4.1 will return false since $\text{match} = \{(1, 2), (2, 3)\}$ contains less than three elements after Step (2).

On the other hand, running the algorithm on inputs $v = (3, 2, 0, 0, 5)$ and $w = (5, 1, 4, 6, 9)$ will produce an output of $\{(1, 1), (2, 3), (3, 2), (4, 4), (5, 5)\}$, which correctly gives the witness $\sigma = (23)$ to the relation $x_1^3 x_2^2 x_5^5 \preceq x_1^5 x_2 x_3^4 x_4^6 x_5^9$.

We also need to know how to compute a reduction of a polynomial f by another polynomial g (assuming that f is reducible by g). Given a witness σ , however, this is calculated in Lemma 2.10. Specifically, we set

$$SG_\sigma(f, g) = f - \frac{\text{lt}(f)}{\sigma \text{lt}(g)} \sigma g. \quad (3)$$

Notice that when $\sigma = (1)$, the polynomial $SG_\sigma(f, g)$ resembles the normal S -pair from standard Gröbner basis theory.

The general case of reducing a polynomial f by a set B is performed as follows; it is a modification of ordinary polynomial division in the setting of finite dimensional polynomial rings.

ALGORITHM 4.3. (Reducing a polynomial f by an ordered set of polynomials B)

Input: Polynomial f and an ordered set $B = (b_1, \dots, b_s) \in R^s$.

Output: A normal form (remainder) of f with respect to B .

1. Set $p := f$, $r := 0$, $\text{divoccured} := 0$;

2. While $p \neq 0$:

$i := 1$;

$\text{divoccured} := 0$;

While $i \leq s$;

$g := b_i$;

If there exists witness σ to $\text{lm}(g) \preceq \text{lm}(p)$, then

$p := SG_\sigma(p, g)$;

$\text{divoccured} := 1$;

Break inner loop;

Else, $i := i + 1$;

If $\text{divoccured} = 0$, then

$r := r + \text{lt}(p)$;

$p := p - \text{lt}(p)$;

3. Return r ;

EXAMPLE 4.4. Let $f = x_3^2 x_2^2 + x_2 x_1$ and $B = (x_3 x_1 + x_2 x_1)$. Reducing f by B is the same as reducing f by $x_3 x_1 + x_2 x_1$ twice as one can check. The resulting polynomial is $x_3^3 x_1 + x_2 x_1$.

Before coming to our main result, we describe a truncated version of it.

ALGORITHM 4.5. (Constructing a truncated Gröbner basis for a symmetric ideal)

Input: An integer N and polynomials $F = \{f_1, \dots, f_n\} \subset K[x_1, \dots, x_N]$.

Output: A truncated Gröbner basis for $I = \langle f_1, \dots, f_n \rangle_{R[\mathfrak{S}_\infty]}$.

1. Set $F' := F$;

2. For each pair (f_i, f_j) :

For each pair (σ, τ) of permutations in $\mathfrak{S}_N$:

$h :=$ the S -polynomial of σf_i and τf_j ;

Set r to be the reduction of h by $\mathfrak{S}_N B'$;

If $r \neq 0$, then $B' := B' \cup \{r\}$;

3. Return B' ;

REMARK 4.6. As we have seen, it is not enough to choose N to be the largest indeterminate appearing in F (c.f. Remark 1.4).

We call the input N the order of a truncated basis for F .

ALGORITHM 4.7. (Constructing a Gröbner basis for a symmetric ideal)

Input: Polynomials $F = \{f_1, \dots, f_n\} \subset K[x_1, \dots, x_N]$.

Output: A Gröbner basis for $I = \langle f_1, \dots, f_n \rangle_{R[\mathfrak{S}_\infty]}$.

1. Set $F' := F$, $i := N$;

2. While true:

Set F' to be a truncated Gröbner basis of F' of order i ;

If every element of F' reduces to 0 by $\mathfrak{S}_N F$, then return F ;

$F := F'$;

$i := i + 1$;

EXAMPLE 4.8. Consider $F = \{x_1 + x_2, x_1 x_2\}$ from the introduction. One iteration of Algorithm 4.7 with $i = 2$ gives $F' = \{x_1 + x_2, x_1^2\}$. The next two iterations produce $\{x_1\}$ and thus the algorithm returns with this as its answer.

5. REFERENCES

- [1] M. Aschenbrenner and C. J. Hillar. Finite generation of symmetric ideals. *Trans. Amer. Math. Soc.*, 359(11):5171–5192, 2007.
- [2] B. Buchberger. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. *Aequationes Math.*, 4:374–383, 1970.
- [3] D. Cox, J. Little, and D. O’Shea. *Ideals, varieties, and algorithms*. Undergraduate Texts in Mathematics. Springer, New York, third edition, 2007. An introduction to computational algebraic geometry and commutative algebra.
- [4] D. A. Cox, J. Little, and D. O’Shea. *Using algebraic geometry*, volume 185 of *Graduate Texts in Mathematics*. Springer, New York, second edition, 2005.

- [5] M. Drton, B. Sturmfels, and S. Sullivant. Algebraic factor analysis: tetrads, pentads and beyond. *Probab. Theory Related Fields*, 138(3-4):463–493, 2007.
- [6] J.-C. Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, pages 75–83 (electronic), New York, 2002. ACM.
- [7] C. J. Hillar and T. Windfeldt. Minimal generators for symmetric ideals. *Proc. Amer. Math. Soc.*, to appear.
- [8] A. Mead, E. Ruch, and A. Schönhofer. Theory of chirality functions, generalized for molecules with chiral ligands. *Theor. Chim. Acta*, 29:269–304, 1973.
- [9] E. Ruch and A. Schönhofer. Theorie der chiralitätsfunktionen. *Theor. Chim. Acta*, 19:225–287, 1970.
- [10] E. Ruch, A. Schönhofer, and I. Ugi. Die vandermondesche determinante als näherungsansatz für eine chiralitätsbeobachtung, ihre verwendung in der stereochemie und zur berechnung der optischen aktivität. *Theor. Chim. Acta*, 7:420–432, 1967.

Moment Matrices, Trace Matrices and the Radical of Ideals

Itanit Janovitz-Freireich
and Ágnes Szántó

*

North Carolina State
University
Campus Box 8205
Raleigh, NC, 27695
USA

ijanovi2@ncsu.edu,
aszanto@ncsu.edu

Bernard Mourrain

†

GALAAD
INRIA,
Sophia Antipolis,
France

mourrain@sophia.inria.fr

Lajos Rónyai

‡

MTA SZTAKI
1111 Budapest,
Lágymányosi u. 11
Hungary

lajos@csillag.ilab.sztaki.hu

ABSTRACT

Let $f_1, \dots, f_s \in \mathbb{K}[x_1, \dots, x_m]$ be a system of polynomials generating a zero-dimensional ideal $\mathcal{I}$, where $\mathbb{K}$ is an arbitrary algebraically closed field. Assume that the factor algebra $\mathcal{A} = \mathbb{K}[x_1, \dots, x_m]/\mathcal{I}$ is Gorenstein and that we have a bound $\delta > 0$ such that a basis for $\mathcal{A}$ can be computed from multiples of $f_1, \dots, f_s$ of degrees at most δ . We propose a method using Sylvester or Macaulay type resultant matrices of $f_1, \dots, f_s$ and J , where J is a polynomial of degree δ generalizing the Jacobian, to compute moment matrices, and in particular matrices of traces for $\mathcal{A}$. These matrices of traces in turn allow us to compute a system of multiplication matrices $\{M_{x_i} | i = 1, \dots, m\}$ of the radical $\sqrt{\mathcal{I}}$, following the approach in the previous work by Janovitz-Freireich, Rónyai and Szántó. Additionally, we give bounds for δ for the case when $\mathcal{I}$ has finitely many projective roots in $\mathbb{P}_{\mathbb{K}}^m$.

Categories and Subject Descriptors

I.1.2 [Computing Methodologies]: Algebraic Manipulations—*Algebraic Algorithms*

General Terms

Algorithms, Theory

*North Carolina State University, Department of Mathematics. Research supported by NSF grant CCR-0347506.

†GALAAD, INRIA, Sophia Antipolis, France. Research partially supported by the ANR GECKO.

‡Computer and Automation Institute of the Hungarian Academy of Sciences, and Budapest University of Technology and Economics. Research supported in part by OTKA grant NK63066.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC '08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Keywords

Moment Matrices, Matrices of Traces, Radical Ideal, Solving polynomial systems.

1. INTRODUCTION

This paper is a continuation of our previous investigation in [22, 23] to compute the approximate radical of a zero dimensional ideal which has zero clusters. The computation of the radical of a zero dimensional ideal is a very important problem in computer algebra since a lot of the algorithms for solving polynomial systems with finitely many solutions need to start with a radical ideal. This is also the case in many numerical approaches, where Newton-like methods are used. From a symbolic-numeric perspective, when we are dealing with approximate polynomials, the zero-clusters create great numerical instability, which can be eliminated by computing the approximate radical.

The theoretical basis of the symbolic-numeric algorithm presented in [22, 23] was Dickson's lemma [14], which, in the exact case, reduces the problem of computing the radical of a zero dimensional ideal to the computation of the nullspace of the so called matrices of traces (see Definition 14): in [22, 23] we studied numerical properties of the matrix of traces when the roots are not multiple roots, but form small clusters. Among other things we showed that the direct computation of the matrix of traces (without the computation of the multiplication matrices) is preferable since the matrix of traces is continuous with respect to root perturbations around multiplicities while multiplication matrices are generally not.

It turns out that the computationally most expensive part of the method in [22, 23] is the computation of the matrix of traces. We address this problem in the present paper, and give a simple algorithm using only Sylvester or Macaulay type resultant matrices and elementary linear algebra to compute matrices of traces of zero dimensional ideals satisfying certain conditions.

More precisely, we need the following assumptions: let $\mathbf{f} = [f_1, \dots, f_s]$ be a system of polynomials of degrees $d_1 \geq \dots \geq d_s$ in $\mathbb{K}[\mathbf{x}]$, with $\mathbf{x} = [x_1, \dots, x_m]$, generating an ideal $\mathcal{I}$ in $\mathbb{K}[\mathbf{x}]$, where $\mathbb{K}$ is an arbitrary algebraically closed field. We assume that the algebra $\mathcal{A} := \mathbb{K}[\mathbf{x}]/\mathcal{I}$ is finite dimensional over $\mathbb{K}$ and that we have a bound $\delta > 0$ such that a basis

$S = [b_1, \dots, b_N]$ of $\mathcal{A}$ can be obtained by taking a linear basis of the space of polynomials of degree at most δ factored by the subspace generated by the multiples of $f_1, \dots, f_s$ of degrees at most δ . By slight abuse of notation we denote the elements of the basis S which are in $\mathcal{A}$ and some fixed preimages of them in $\mathbb{K}[\mathbf{x}]$ both by $b_1, \dots, b_N$. Thus we can assume that the basis S consists of monomials of degrees at most δ . Note that we can prove bounds $\delta = \sum_{i=1}^{m+1} d_i - m$ (or $\delta = \sum_{i=1}^m d_i - m$ if $s = m$) if $\mathcal{I}$ has only finitely many projective common roots in $\mathbb{P}_{\mathbb{K}}^m$ and have no common roots at infinity, using a result of Lazard [33] (see Theorem 3).

Furthermore, we also assume that $\mathcal{A}$ is Gorenstein over $\mathbb{K}$ (see Definition 1). Note that in practice we can easily detect if $\mathcal{A}$ is not Gorenstein (see Remark 10). Also, a random change of projective variables can eliminate roots at infinity with high probability when they are in finite number, but we will address the necessity of this assumption in an upcoming paper.

The main ingredient of our method is a Macaulay type resultant matrix $\text{Mac}_{\Delta}(\mathbf{f})$, which is defined to be a maximal row-independent submatrix of the transpose matrix of the degree Δ Sylvester map $(g_1, \dots, g_s) \mapsto \sum_{i=1}^s f_i g_i \in \mathbb{K}[\mathbf{x}]_{\Delta}$ for $\Delta \leq 2\delta + 1$ (see Definition 5). Using our assumptions on $\mathcal{A}$, we can compute a basis S of $\mathcal{A}$ using $\text{Mac}_{\Delta}(\mathbf{f})$, and we also prove that a random element $\mathbf{y}$ of the nullspace of $\text{Mac}_{\Delta}(\mathbf{f})$ provides a non-singular $N \times N$ moment matrix $\mathfrak{M}_S(\mathbf{y})$ with high probability (similarly as in [31]). This moment matrix allows us to compute the other main ingredient of our algorithm, a polynomial J of degree at most δ , such that J is the generalization of the Jacobian of $f_1, \dots, f_s$ in the case when $s = m$. The main result of the paper now can be formulated as follows:

THEOREM *Let $S = [b_1, \dots, b_N]$ be a basis of $\mathcal{A}$ with $\deg(b_i) \leq \delta$. With J as above, let $\text{Syl}_S(J)$ be the transpose matrix of the map $\sum_{i=1}^N c_i b_i \mapsto J \cdot \sum_{i=1}^N c_i b_i \in \mathbb{K}[\mathbf{x}]_{\Delta}$ for $c_i \in \mathbb{K}$. Then*

$$[\text{Tr}(b_i b_j)]_{i,j=1}^N = \text{Syl}_S(J) \cdot X,$$

where X is the unique extension of the matrix $\mathfrak{M}_S(\mathbf{y})$ such that $\text{Mac}_{\Delta}(\mathbf{f}) \cdot X = 0$.

Once we compute the matrix of traces $R := [\text{Tr}(b_i b_j)]_{i,j=1}^N$ and the matrices $R_{x_k} := [\text{Tr}(x_k b_i b_j)]_{i,j=1}^N = \text{Syl}_S(x_k J) \cdot X$ for $k = 1, \dots, m$, we can use the results of [22, 23] to compute a system of multiplication matrices for the (approximate) radical of $\mathcal{I}$ as follows: if $\tilde{R}$ is a (numerical) maximal non-singular submatrix of R and $\tilde{R}_{x_k}$ is the submatrix of R_{x_k} with the same row and column indices as in $\tilde{R}$, then the solution M_{x_k} of the linear matrix equation

$$\tilde{R} M_{x_k} = \tilde{R}_{x_k}$$

is an (approximate) multiplication matrix of x_k for the (approximate) radical of $\mathcal{I}$. See [23] for the definition of (approximate) multiplication matrices. Note that a generating set for the radical $\sqrt{\mathcal{I}}$ can be obtained directly from the definition of multiplication matrices, in particular, it corresponds to the rows of the matrices $M_{x_1}, \dots, M_{x_m}$.

We also point out that in the $s = m$ case these multiplication matrices M_{x_k} can be obtained even more simply using the nullspace of $\text{Mac}_{\Delta}(\mathbf{f})$ and the Jacobian J of $\mathbf{f}$, without computing the matrices of traces.

We also note here that in a follow up paper we will consider an extension of our present results which works also in the non-Gorenstein case to compute the matrices of traces. Furthermore, that paper will also extend our results to the affine complete intersection case using Bezout matrices.

2. RELATED WORK

The motivation for this work was the papers [31, 32] where they use moment matrices to compute the radical of real and complex ideals. They present two versions of the method for the complex case: first, in [32] they double up the machinery for the real case to obtain the radical of the complex ideal. However, in [31] they significantly simplify their method and show how to use moment matrices of maximal rank to compute the multiplication matrices of an ideal between $\mathcal{I}$ and its radical $\sqrt{\mathcal{I}}$. In particular, in the Gorenstein case they can compute the multiplication matrices of $\mathcal{I}$. In fact, in [31] they cite our previous work [22] to compute the multiplication matrices of $\sqrt{\mathcal{I}}$ from the multiplication matrices of $\mathcal{I}$, but the method proposed in the present paper is much simpler and more direct.

Note that one can also obtain the multiplication matrices of $\mathcal{I}$ with respect to the basis $S = [b_1, \dots, b_N]$ by simply eliminating the terms not in S from $x_k b_i$ using $\text{Mac}_{\delta+1}(\mathbf{f})$. The advantage of computing multiplication matrices of the radical $\sqrt{\mathcal{I}}$ is that it returns matrices which are always simultaneously diagonalizable, and possibly smaller than the multiplication matrices of $\mathcal{I}$, hence easier to work with. Moreover, if S contains the monomials $1, x_1, \dots, x_m$, one eigenvector computation yields directly the coordinates of the roots.

Computation of the radical of zero dimensional complex ideals is very well studied in the literature: methods most related to ours include [18, 5] where matrices of traces are used in order to find generators of the radical, and the matrices of traces are computed using Gröbner Bases; also, in [1] they use the traces to give a bound for the degree of the generators of the radical and use linear solving methods from there; in [19] they describe the computation of the radical using symmetric functions which are related to traces. One of the most commonly quoted method to compute radicals is to compute the projections $\mathcal{I} \cap \mathbb{K}[x_i]$ for each $i = 1, \dots, m$ and then use univariate squarefree factorization (see for example [17, 26, 10, 20]). The advantage of the latter is that it can be generalized for higher dimensional ideals (see for example [25]). We note here that an advantage of the method using matrices of traces is that it behaves stably under perturbation of the roots of the input system, as was proved in [23]. Other methods to compute the radical of zero dimensional ideals include [24, 16, 28, 29, 30, 39]. Applications of computing the radical include [21], where they show how to compute the multiplicity structure of the roots of $\mathcal{I}$ once the radical is computed.

Methods for computing the matrix of traces directly from the generating polynomials of $\mathcal{I}$, without using multiplication matrices, include [13, 6] where they use Newton Sums, [7, 8, 9] where they use residues and [12] using resultants. Besides computing the radical of an ideal, matrices of traces have numerous applications mainly in real algebraic geometry [2, 35, 4], or in [36] where trace matrices are applied to find separating linear forms deterministically.

3. MOMENT MATRICES AND MATRICES OF TRACES

Let $\mathbf{f} = [f_1, \dots, f_s]$ be a system of polynomials of degrees $d_1 \geq \dots \geq d_s$ in $\mathbb{K}[\mathbf{x}]$, where $\mathbf{x} = [x_1, \dots, x_m]$ and $\mathbb{K}$ is an arbitrary algebraically closed field. Let $\mathcal{I}$ be the ideal generated by $f_1, \dots, f_s$ in $\mathbb{K}[\mathbf{x}]$ and define $\mathcal{A} := \mathbb{K}[\mathbf{x}]/\mathcal{I}$. We assume throughout the paper that $\mathcal{A}$ is a finite dimensional vector space over $\mathbb{K}$ and let $\mathcal{A}^*$ denote the dual space of $\mathcal{A}$.

Let us first recall the definition of a Gorenstein algebra (c.f. [27, 37, 15, 31]). Note that these algebras are also referred to as Frobenius in the literature, see for example [3].

DEFINITION 1. *A finite dimensional $\mathbb{K}$ -algebra $\mathcal{A}$ is Gorenstein (over $\mathbb{K}$) if there exists a nondegenerate $\mathbb{K}$ -bilinear form $B(x, y)$ on $\mathcal{A}$ such that*

$$B(ab, c) = B(a, bc) \text{ for every } a, b, c \in \mathcal{A}.$$

Note that this is equivalent to the fact that $\mathcal{A}$ and $\mathcal{A}^*$ are isomorphic as $\mathcal{A}$ modules. It is also equivalent to the existence of a $\mathbb{K}$ -linear function $\Lambda : \mathcal{A} \rightarrow \mathbb{K}$ such that the bilinear form $B(a, b) := \Lambda(ab)$ is nondegenerate on $\mathcal{A}$.

ASSUMPTION 2. *Throughout the paper we assume that $\mathcal{A}$ is Gorenstein. Furthermore, we also assume that we have a bound $\delta > 0$ such that*

$$N := \dim_{\mathbb{K}} \mathbb{K}[\mathbf{x}]_{\delta} / \langle f_1, \dots, f_s \rangle_{\delta} = \dim_{\mathbb{K}} \mathbb{K}[\mathbf{x}]_d / \langle f_1, \dots, f_s \rangle_d \quad (1)$$

for all $d \geq \delta$ and that

$$N = \dim \mathcal{A}. \quad (2)$$

Here

$$\langle f_1, \dots, f_s \rangle_d := \left\{ \sum_i f_i p_i : \deg(p_i) \leq d - d_i \right\}. \quad (3)$$

We fix $S = [b_1, \dots, b_N]$ a monomial basis for $\mathcal{A}$ such that $\deg(b_i) \leq \delta$ for all $i = 1, \dots, N$. Let D be the maximum degree of the monomials in S . Thus $D \leq \delta$.

We have the following theorem giving bounds for δ in the case when $\mathbf{f}$ has finitely many projective roots.

THEOREM 3. *Let $\mathbf{f} = [f_1, \dots, f_s]$ be a system of polynomials of degrees $d_1 \geq \dots \geq d_s$ in $\mathbb{K}[\mathbf{x}]$. Assume that $f_1, \dots, f_s$ has finitely many projective common roots in $\mathbb{P}_{\mathbb{K}}^m$. Assume further that $f_1, f_2, \dots, f_s$ have no common roots at infinity. Then:*

1. *If $s = m$ then for $\delta := \sum_{i=1}^m (d_i - 1)$ conditions (1) and (2) are satisfied. Furthermore, in this case $\mathcal{A}$ is always Gorenstein.*
2. *If $s > m$ then for $\delta := \sum_{i=1}^{m+1} d_i - m$ conditions (1) and (2) are satisfied.*

PROOF. For the first assertion let $\mathbf{f}^h$ be the homogenization of $\mathbf{f}$ using a new variable x_{m+1} . Using our assumption that $\mathbf{f}^h$ has finitely many roots in $\mathbb{P}_{\mathbb{K}}^m$ and $s = m$, one can see that $(\mathbf{f}^h)$ is a regular sequence in $R := \mathbb{K}[x_1, \dots, x_m, x_{m+1}]$.

Define the graded ring $B := R/\langle \mathbf{f}^h \rangle$. Following the approach and notation in [38], we can now calculate the Hilbert series of B , defined by $H(B, \lambda) = \sum_d \mathcal{H}_B(d) \lambda^d$, where $\mathcal{H}_B$ is the Hilbert function of B . We have

$$H(R, \lambda) = \frac{H(B, \lambda)}{(1 - \lambda^{d_1}) \dots (1 - \lambda^{d_m})},$$

and using the simple fact that

$$H(R, \lambda) = \frac{1}{(1 - \lambda)^{m+1}}$$

we obtain that

$$\begin{aligned} H(B, \lambda) &= \frac{(1 + \lambda + \dots + \lambda^{d_1-1}) \dots (1 + \lambda + \dots + \lambda^{d_m-1})}{(1 - \lambda)} \\ &= g(\lambda)(1 + \lambda + \dots), \end{aligned}$$

where

$$g(\lambda) = (1 + \lambda + \dots + \lambda^{d_1-1}) \dots (1 + \lambda + \dots + \lambda^{d_m-1}).$$

This implies that the Hilbert function

$$\mathcal{H}_B(\delta) = \mathcal{H}_B(\delta + 1) = \mathcal{H}_B(\delta + 2) = \dots$$

Note that dehomogenization induces a linear isomorphism $B_d \rightarrow \mathbb{K}[\mathbf{x}]_d / \langle f_1, \dots, f_s \rangle_d$, where B_d stands for the degree d homogeneous part of B . From this, using that there are no common roots at infinity, we infer that for $d \geq \delta$ $\dim_{\mathbb{K}} \mathbb{K}[\mathbf{x}]_d / \langle f_1, \dots, f_s \rangle_d = \dim_{\mathbb{K}} \mathcal{A} = N$, which implies (1) and (2).

Note that the common value $N = \mathcal{H}_B(\delta)$ is the sum of the coefficients of g , which is

$$g(1) = \prod_{i=1}^m d_i.$$

To prove that $\mathcal{A}$ is Gorenstein, we cite [15, Proposition 8.25, p. 221] where it is proved that if $f_1, \dots, f_m$ is an affine complete intersection then the Bezoutian $B_{1, f_1, \dots, f_m}$ defines an isomorphism between $\mathcal{A}^*$ and $\mathcal{A}$.

To prove the second assertion we note that [33, Theorem 3.3] implies that

$$\dim_{\mathbb{K}} B_{\delta} = \dim_{\mathbb{K}} B_{\delta+1} = \dots$$

From here we obtain (1) and (2) as in the Case 1.

□

REMARK 4. *Note that in general $\mathcal{I}_d \neq \langle f_1, \dots, f_s \rangle_d$, where $\mathcal{I}_d$ is the set of elements of $\mathcal{I}$ with degree at most d and $\langle f_1, \dots, f_s \rangle_d$ was defined in (3). This can happen when the system has a root at infinity, for example, if $f_1 = x+1$, $f_2 = x$ then $\mathcal{I}_0 = \text{span}_{\mathbb{K}}(1)$ but $\langle f_1, f_2 \rangle_0 = \{0\}$. However, using the homogenization $f_1^h, \dots, f_s^h$, the degree d part of the homogenized ideal is always equal to the space spanned by the multiples of $f_1^h, \dots, f_s^h$ of degree d . The above example also demonstrates that $\dim \mathcal{A}$ is not always the same as $\dim \mathbb{K}[\mathbf{x}]_d / \langle f_1, \dots, f_s \rangle_d$ for large enough d , because above $\dim \mathcal{A} = 0$ but $\dim \mathbb{K}[x, y]_d / \langle f_1, f_2 \rangle_d = 1$ for all $d \geq 0$.*

Next we will define Sylvester and Macaulay type resultant matrices for $f_1, \dots, f_s$.

DEFINITION 5. *Define*

$$\Delta := \max(\delta, 2D + 1)$$

where δ and D are defined in Assumption 2.

Let $\text{Syl}_{\Delta}(\mathbf{f})$ be the transpose matrix of the linear map

$$\begin{aligned} \bigoplus_i \mathbb{K}[\mathbf{x}]_{\Delta-d_i} &\longrightarrow \mathbb{K}[\mathbf{x}]_{\Delta} \\ (g_1, \dots, g_s) &\mapsto \sum_{i=1}^s f_i g_i \end{aligned} \quad (4)$$

written in the monomial bases. So, in our notation, $\text{Syl}_\Delta(\mathbf{f})$ will have rows which correspond to all polynomials $f_i x^\alpha$ of degree at most Δ .

Let $\text{Mac}_\Delta(\mathbf{f})$ be a row submatrix of $\text{Syl}_\Delta(\mathbf{f})$ of maximal size with linearly independent rows.

REMARK 6. In the case where $s = m$, for generic $\mathbf{f}$ we can directly construct $\text{Mac}_\Delta(\mathbf{f})$ by taking the restriction of the map (4) to

$$\bigoplus_{i=1}^m \mathcal{S}_i(\Delta) \longrightarrow \mathbb{K}[\mathbf{x}]_\Delta$$

where $\mathcal{S}_i(\Delta) = \text{span}\{\mathbf{x}^\alpha : |\alpha| \leq \Delta - d_i, \forall j < i, \alpha_j < d_j\}$.

Here $\text{Mac}_\Delta(\mathbf{f})$ is a submatrix of the classical Macaulay matrix of the homogenization of $\mathbf{f}$ and some f_{m+1}^h , where f_{m+1}^h is any homogeneous polynomial of degree $\Delta - \delta$: we only take the rows corresponding to the polynomials in $\mathbf{f}$. Since the Macaulay matrix is generically non-singular, $\text{Mac}_\Delta(\mathbf{f})$ will also be generically full rank.

Note that with our assumption that $f_1, \dots, f_m$ has finitely many projective roots, we have that $\text{Mac}_\Delta(\mathbf{f})$ has column corank $N := \prod_{i=1}^m d_i$.

Since $\Delta \geq \delta$, by Assumption 2 the corank of $\text{Mac}_\Delta(\mathbf{f}) = N$, where N is the dimension of $\mathcal{A}$. Also, we can assume that the elements of the basis S of $\mathcal{A}$ are monomials of degree at most δ , and that the first columns of $\text{Mac}_\Delta(\mathbf{f})$ correspond to the basis S of $\mathcal{A}$.

Fix an element

$$\mathbf{y} = [y_\alpha : \alpha \in \mathbb{N}^m, |\alpha| \leq \Delta]^T$$

of the nullspace $\text{Null}(\text{Mac}_\Delta(\mathbf{f}))$, i.e. $\text{Mac}_\Delta(\mathbf{f}) \cdot \mathbf{y} = 0$.

DEFINITION 7. Let S be the basis of $\mathcal{A}$ as above, consisting of monomials of degree at most D . Using $\mathbf{y}$ we can define $\Lambda_{\mathbf{y}} \in \mathcal{A}^*$ by $\Lambda_{\mathbf{y}}(g) := \sum_{\mathbf{x}^\alpha \in S} y_\alpha g_\alpha$, where $g = \sum_{\mathbf{x}^\alpha \in S} g_\alpha \mathbf{x}^\alpha \in \mathcal{A}$. Note that every $\Lambda \in \mathcal{A}^*$ can be defined as $\Lambda_{\mathbf{y}}$ for some $\mathbf{y} \in \text{Null}(\text{Mac}_\Delta(\mathbf{f}))$ or more generally with an element of $\mathbb{K}[\mathbf{x}]^*$ which vanishes on the ideal $\mathcal{I}$.

Define the moment matrix $\mathfrak{M}_S(\mathbf{y})$ to be the $N \times N$ matrix given by

$$\mathfrak{M}_S(\mathbf{y}) = [y_{\alpha+\beta}]_{\alpha, \beta},$$

where α and β run through the exponents of the monomials in S . Note that $\mathfrak{M}_S$ is only a submatrix of the usual notion of moment matrices in the literature, see for example [11].

For $p \in \mathcal{A}$, we define the linear function $p \cdot \Lambda \in \mathcal{A}^*$ as $p \cdot \Lambda(g) := \Lambda(pg)$ for all $g \in \mathcal{A}$.

REMARK 8. If one considers a linear function Λ on $\mathcal{A}$, such that the bilinear form $(x, y) \mapsto \Lambda(xy)$ is nondegenerate on $\mathcal{A}$, then the moment matrix corresponding to this Λ will be the one whose (i, j) -th entry is just $\Lambda(b_i b_j)$. Moreover, for $g, h \in \mathcal{A}$

$$\Lambda_{\mathbf{y}}(gh) = \text{coeff}_S(g)^T \cdot \mathfrak{M}_S(\mathbf{y}) \cdot \text{coeff}_S(h)$$

where $\text{coeff}_S(p)$ denotes the vector of coefficients of $p \in \mathcal{A}$ in the basis S .

The following proposition is a simple corollary of [31, Prop 3.3 and Cor. 3.1].

PROPOSITION 9. Let $\mathbf{y}$ be a random element of the vector space $\text{Null}(\text{Mac}_\Delta(\mathbf{f}))$. With high probability, $\mathfrak{M}_S(\mathbf{y})$ is non-singular.

REMARK 10. Using the above proposition, one can detect whether the algebra $\mathcal{A}$ is not Gorenstein with high probability by simply computing the rank of $\mathfrak{M}_S(\mathbf{y})$ for (perhaps several) random elements $\mathbf{y}$ in $\text{Null}(\text{Mac}_\Delta(\mathbf{f}))$.

REMARK 11. By [31, Theorem 2.6 and Lemma 3.2] one can extend $\mathbf{y}$ to $\tilde{\mathbf{y}} \in \mathbb{K}^{\mathbb{N}^m}$ such that the infinite moment matrix $\mathfrak{M}(\tilde{\mathbf{y}}) := [\tilde{y}_{\alpha+\beta}]_{\alpha, \beta \in \mathbb{N}^m}$ has the same rank as $\mathfrak{M}_S(\mathbf{y})$ and the columns of $\mathfrak{M}(\tilde{\mathbf{y}})$ vanish on all the elements of the ideal $\mathcal{I}$.

Next we define a basis dual to $S = [b_1, \dots, b_N]$ with respect to the moment matrix $\mathfrak{M}_S(\mathbf{y})$. Using this dual basis we also define a polynomial J which is in some sense a generalization of the Jacobian of a well-constrained polynomial system.

DEFINITION 12. From now on we fix $\mathbf{y} \in \text{Null}(\text{Mac}_\Delta(\mathbf{f}))$ such that $\mathfrak{M}_S(\mathbf{y})$ is invertible and we will denote by Λ the corresponding element $\Lambda_{\mathbf{y}} \in \mathcal{A}^*$. We define

$$\mathfrak{M}_S^{-1}(\mathbf{y}) =: [c_{ij}]_{i,j=1}^N.$$

Let $b_i^* := \sum_{j=1}^N c_{ji} b_j$. Then $[b_1^*, \dots, b_N^*]$ corresponds to the columns of the inverse matrix $\mathfrak{M}_S^{-1}(\mathbf{y})$ and they also form a basis for $\mathcal{A}$. Note that we have $\Lambda(b_i b_j^*) = 1$, if $i = j$, and 0 otherwise.

Define the generalized Jacobian by

$$J := \sum_{i=1}^N b_i b_i^* \mod \mathcal{I} \quad (5)$$

expressed in the basis $S = [b_1, \dots, b_N]$ of $\mathcal{A}$.

REMARK 13. Note that since $\sum_{i=1}^N b_i b_i^*$ has degree at most $2D$, and $\Delta > 2D$, we can use $\text{Mac}_\Delta(\mathbf{f})$ to find its reduced form, which is J . Because of this reduction, we have that $\deg(J) \leq D \leq \delta$.

Also note that the notion of generalized Jacobian was also introduced in [3]. Its name come from the fact that if $s = m$ and if Λ is the so called residue (c.f. [15]), then $\sum_{i=1}^N b_i b_i^* = J$ is the Jacobian of $f_1, \dots, f_m$.

We now recall the definition of the multiplication matrices and the matrix of traces as presented in [23].

DEFINITION 14. Let $p \in \mathcal{A}$. The multiplication matrix M_p is the transpose of the matrix of the multiplication map

$$\begin{aligned} M_p : \mathcal{A} &\longrightarrow \mathcal{A} \\ g &\mapsto pg \end{aligned}$$

written in the basis S .

The matrix of traces is the $N \times N$ symmetric matrix:

$$R = [\text{Tr}(b_i b_j)]_{i,j=1}^N$$

where $\text{Tr}(pq) := \text{Tr}(M_{pq})$, M_{pq} is the multiplication matrix of pq as an element in $\mathcal{A}$ in terms of the basis $S = [b_1, \dots, b_N]$ and Tr indicates the trace of a matrix.

The next results relate the multiplication by J matrix to the matrix of traces R .

PROPOSITION 15. Let M_J be the multiplication matrix of J with respect to the basis S . We then have that

$$M_J = [\text{Tr}(b_i b_j^*)]_{i,j=1}^N.$$

PROOF. Let $\Lambda \in \mathcal{A}^*$ be as in Definition 12. For any $h \in \mathcal{A}$ we have that

$$\begin{aligned} h &= \sum_{j=1}^N \Lambda(hb_j)b_j^* = \sum_{j=1}^N \Lambda(hb_j^*)b_j \\ \Rightarrow hb_i &= \sum_{j=1}^N \Lambda(hb_j^*b_i)b_j \Rightarrow M_h[i, j] = \Lambda(hb_j^*b_i) \\ \Rightarrow \text{Tr}(h) &= \sum_{i=1}^N \Lambda(hb_i^*b_i) = \Lambda(h \sum_{i=1}^N b_i^*b_i). \end{aligned}$$

Since $J = \sum_{i=1}^N b_i^*b_i$ in $\mathcal{A}$, we have $\text{Tr}(h) = \Lambda(hJ)$. Therefore

$$M_J[i, j] = \Lambda(Jb_j^*b_i) = \text{Tr}(b_j^*b_i) = \text{Tr}(b_i b_j^*)$$

□

COROLLARY 16.

$$M_J \cdot \mathfrak{M}_S(\mathbf{y}) = [\text{Tr}(b_i b_j)]_{i,j=1}^N = R,$$

or equivalently $J \cdot \Lambda = \text{Tr}$ in $\mathcal{A}^*$.

PROOF. The coefficients of b_i^* in the basis $S = [b_1, \dots, b_N]$ are the columns of $\mathfrak{M}_S^{-1}(\mathbf{y})$, which implies that

$$M_J = [\text{Tr}(b_i b_j^*)]_{i,j=1}^N = [\text{Tr}(b_i b_j)]_{i,j=1}^N \cdot \mathfrak{M}_S^{-1}(\mathbf{y}).$$

Therefore we have that $M_J \cdot \mathfrak{M}_S(\mathbf{y}) = [\text{Tr}(b_i b_j)]_{i,j=1}^N$. □

Finally, we prove that the matrix of traces R can be computed directly from the Sylvester matrix of $f_1, \dots, f_s$ and J , without using the multiplication matrix M_J . First we need a lemma.

LEMMA 17. *There exists a unique matrix $\mathfrak{R}_S(\mathbf{y})$ of size $|\text{Mon}_{\leq}(\Delta) - S| \times |S|$ such that*

$$\text{Mac}_{\Delta}(\mathbf{f}) \cdot \begin{bmatrix} \mathfrak{M}_S(\mathbf{y}) \\ \mathfrak{R}_S(\mathbf{y}) \end{bmatrix} = 0$$

PROOF. By our assumption that the first columns of $\text{Mac}_{\Delta}(\mathbf{f})$ correspond to S we have

$$\text{Mac}_{\Delta}(\mathbf{f}) = \begin{bmatrix} B & A \end{bmatrix},$$

where the columns of B are indexed by the monomials in S . Note here that by Assumption 2 the rows of $\text{Mac}_{\Delta}(\mathbf{f})$ span $\mathcal{I}_{\Delta}$, and the monomials in S span the factor space $\mathbb{K}[\mathbf{x}]_{\Delta}/\mathcal{I}_{\Delta}$. These together imply that the (square) submatrix A is invertible.

Then

$$\begin{bmatrix} B & A \end{bmatrix} \cdot \begin{bmatrix} Id_{N \times N} \\ -A^{-1}B \end{bmatrix} = 0$$

which implies that

$$\text{Mac}_{\Delta}(\mathbf{f}) \cdot \begin{bmatrix} \mathfrak{M}_S(\mathbf{y}) \\ \mathfrak{R}_S(\mathbf{y}) \end{bmatrix} = 0,$$

where $\mathfrak{R}_S(\mathbf{y}) = -A^{-1}B \cdot \mathfrak{M}_S(\mathbf{y})$.

□

By construction, the column of $\mathfrak{M}_S(\mathbf{y})$ indexed by $b_j \in S$ corresponds to the values of $b_j \cdot \Lambda \in \mathcal{A}^*$ on $b_1, \dots, b_N$. The same column in $\mathfrak{R}_S(\mathbf{y})$ corresponds to the values of $b_j \cdot \Lambda$ on the complementary set of monomials of $\text{Mon}_{\leq}(\Delta)$. The column in the stacked matrix corresponds to the value of $b_j \cdot \Lambda$ on all the monomials in $\text{Mon}_{\leq}(\Delta)$. To evaluate $b_j \cdot \Lambda(p)$ for a polynomial p of degree $\leq \Delta$, we simply compute the inner product of the coefficient vector of p with this column.

DEFINITION 18. *Let $S = [b_1, \dots, b_N]$ be the basis of $\mathcal{A}$ as above, and let $P \in \mathbb{K}[\mathbf{x}]$ be a polynomial of degree at most $D + 1$.*

Define $\text{Syl}_S(P)$ to be the matrix with rows corresponding to the coefficients of the polynomials $(b_1 P), \dots, (b_N P)$ in the monomial basis $\text{Mon}_{\leq}(\Delta)$ (we use here that $\deg(b_i) \leq D$, thus $\deg(b_i P) \leq 2D + 1 \leq \Delta$).

Furthermore, we assume that the monomials corresponding to the columns of $\text{Syl}_S(P)$ are in the same order as the monomials corresponding to the columns of $\text{Mac}_{\Delta}(\mathbf{f})$.

THEOREM 19.

$$\text{Syl}_S(J) \cdot \begin{bmatrix} \mathfrak{M}_S(\mathbf{y}) \\ \mathfrak{R}_S(\mathbf{y}) \end{bmatrix} = [\text{Tr}(b_i b_j)]_{i,j=1}^N$$

PROOF. Since the j -th column of the matrix

$$\begin{bmatrix} \mathfrak{M}_S(\mathbf{y}) \\ \mathfrak{R}_S(\mathbf{y}) \end{bmatrix}$$

represents the values of $b_j \cdot \Lambda$ on all the monomials of degree less than or equal to Δ , and the i -th row of $\text{Syl}_S(J)$ is the coefficient vector of $b_i J$, we have

$$\begin{aligned} \text{Syl}_S(J) \cdot \begin{bmatrix} \mathfrak{M}_S(\mathbf{y}) \\ \mathfrak{R}_S(\mathbf{y}) \end{bmatrix} &= [(b_j \cdot \Lambda)(b_i J)]_{i,j=1}^N \\ &= [\Lambda(Jb_i b_j)]_{i,j=1}^N \\ &= [\text{Tr}(b_i b_j)]_{i,j=1}^N. \end{aligned}$$

□

We can now describe the algorithm to compute a set of multiplication matrices M_{x_i} , $i = 1, \dots, m$ of the radical $\sqrt{\mathcal{I}}$ of $\mathcal{I}$ with respect to a basis of $\mathbb{K}[\mathbf{x}]/\sqrt{\mathcal{I}}$. To prove that the algorithm below is correct we need the following result from [23, Proposition 8.3] which is the consequence of the fact that the kernel of the matrix of traces corresponds to the radical of $\mathcal{A}$:

PROPOSITION 20. *Let $\tilde{R}$ be a maximal non-singular submatrix of the matrix of traces R . Let r be the rank of $\tilde{R}$, and $T := [b_{i_1}, \dots, b_{i_r}]$ be the monomials corresponding to*

the columns of $\tilde{R}$. Then T is a basis of the algebra $\mathbb{K}[\mathbf{x}]/\sqrt{\mathcal{I}}$ and for each $k = 1, \dots, m$, the solution M_{x_k} of the linear matrix equation

$$\tilde{R}M_{x_k} = \tilde{R}_{x_k}$$

is the multiplication matrix of x_k for $\sqrt{\mathcal{I}}$ with respect to T . Here $\tilde{R}_{x_k}$ is the $r \times r$ submatrix of $[Tr(x_k b_i b_j)]_{i,j=1}^N$ with the same row and column indices as in $\tilde{R}$.

ALGORITHM 21. INPUT: $\mathbf{f} = [f_1, \dots, f_s] \in \mathbb{K}[\mathbf{x}]$ of degrees $d_1, \dots, d_s$ generating an ideal $\mathcal{I}$ and $\delta > 0$ such that they satisfy the conditions in Assumption 2. An optional input is $D \leq \delta$, which by default is set to be δ .

OUTPUT: A basis T for the factor algebra $\mathbb{K}[\mathbf{x}]/\sqrt{\mathcal{I}}$ and a set of multiplication matrices $\{M_{x_i} | i = 1, \dots, m\}$ of $\sqrt{\mathcal{I}}$ with respect to the basis T .

1. Compute $\text{Mac}_\Delta(\mathbf{f})$ for $\Delta := \max(2D + 1, \delta)$
2. Compute a basis S of $\mathbb{K}[\mathbf{x}]_\Delta / (\mathbf{f})_\Delta$ such that the polynomials in S have degrees at most D . Let $S = [b_1, \dots, b_N]$.
3. Compute a random combination $\mathbf{y}$ of the elements of a basis of $\text{Null}(\text{Mac}_\Delta(\mathbf{f}))$.
4. Compute the moment matrix $\mathfrak{M}_S(\mathbf{y})$ defined in Definition 7 and $\mathfrak{R}_S(\mathbf{y})$ defined in Lemma 17.
5. Compute $\mathfrak{M}_S^{-1}(\mathbf{y})$ and the basis $[b_1^*, \dots, b_N^*]$ defined in Definition 12.
6. Compute $J = \sum_{i=1}^N b_i b_i^* \bmod \mathcal{I}$ using $\text{Mac}_\Delta(\mathbf{f})$.
7. Compute $\text{Syl}_S(J)$ and $\text{Syl}_S(x_k J)$ for $k = 1, \dots, m$ defined in Definition 18.
8. Compute

$$R = [Tr(b_i b_j)]_{i,j=1}^N = \begin{bmatrix} \text{Syl}_S(J) \end{bmatrix} \cdot \begin{bmatrix} \mathfrak{M}_S(\mathbf{y}) \\ \mathfrak{R}_S(\mathbf{y}) \end{bmatrix}$$

and

$$R_{x_k} := [Tr(x_k b_i b_j)]_{i,j=1}^N = \begin{bmatrix} \text{Syl}_S(x_k J) \end{bmatrix} \cdot \begin{bmatrix} \mathfrak{M}_S(\mathbf{y}) \\ \mathfrak{R}_S(\mathbf{y}) \end{bmatrix}$$

for $k = 1, \dots, m$.

9. Compute $\tilde{R}$, a maximal non-singular submatrix of R . Let r be the rank of $\tilde{R}$, and $T := [b_{i_1}, \dots, b_{i_r}]$ be the monomials corresponding to the columns of $\tilde{R}$.
10. For each $k = 1, \dots, m$ solve the linear matrix equation $\tilde{R}M_{x_k} = \tilde{R}_{x_k}$, where $\tilde{R}_{x_k}$ is the submatrix of R_{x_k} with the same row and column indices as in $\tilde{R}$.

REMARK 22. Since the bound given in Theorem 3 might be too high, it seems reasonable to design the algorithm in an iterative fashion, similarly to the algorithms in [31, 32, 40], in order to avoid nullspace computations for large matrices. The bottleneck of our algorithm is doing computations with $\text{Mac}_\Delta(\mathbf{f})$, since its size exponentially increases as Δ increases.

REMARK 23. Note that if $s = m$ then we can use the conventional Jacobian of $f_1, \dots, f_m$ in the place of J , and any $|\text{Mon}_\leq(\Delta)| \times |S|$ matrix X such that it has full rank and $\text{Mac}_\Delta(\mathbf{f}) \cdot X = \mathbf{0}$ in the place of

$$\begin{bmatrix} \mathfrak{M}_S(\mathbf{y}) \\ \mathfrak{R}_S(\mathbf{y}) \end{bmatrix}.$$

Even though this way we will not get matrices of traces, a system of multiplication matrices of the radical $\sqrt{\mathcal{I}}$ can still be recovered: if $\tilde{Q}$ denotes a maximal non-singular submatrix of $\text{Syl}_S(J) \cdot X$, and $\tilde{Q}_{x_k}$ is the submatrix of $\text{Syl}_S(x_k J) \cdot X$ with the same row and column indices as in $\tilde{Q}$, then the solution M_{x_k} of the linear matrix equation $\tilde{Q}M_{x_k} = \tilde{Q}_{x_k}$ gives the same multiplication matrix of $\sqrt{\mathcal{I}}$ w.r.t. the same basis T as the above Algorithm.

REMARK 24. As M_{x_k} is the matrix of multiplication by x_k modulo the radical ideal $\sqrt{\mathcal{I}}$, its eigenvectors are (up to a non-zero scalar) the interpolation polynomials at the roots of $\mathcal{I}$. Similarly the eigenvectors of the transposed matrix $M_{x_k}^t$ are (up to a non-zero scalar) the evaluation at the roots ζ of $\mathcal{I}$ (see [34, 15] for more details). The vector which represents this evaluation at ζ in the dual space $\mathcal{A}^*$ is the vector of values of $[b_1, \dots, b_N]$ at ζ . To obtain these vectors, we solve the generalized eigenvalue problem $(\tilde{R}_{x_k}^t - z\tilde{R}^t)w = 0$ and compute $v = \tilde{R}^t w$. The vectors v will be of the form $[b_1(\zeta), \dots, b_N(\zeta)]$ for ζ a root of $\mathcal{I}$. If $b_1 = 1, b_2 = x_1, \dots, b_{m+1} = x_m$, we can read directly the coordinates of ζ from this vector.

4. EXAMPLES

In this section we present three examples. Each of them has three polynomials in two variables. The first one is a system which has roots with multiplicities, the second one is a system which has clusters of roots, and the third one is a system obtained by perturbing the coefficients of the first one. For each of them we compute the Macaulay matrix $\text{Mac}_\Delta(\mathbf{f})$, the vector $\mathbf{y}$ in its nullspace, the moment matrix $\mathfrak{M}_S(\mathbf{y})$, the polynomial J , the matrix of traces R and the (approximate) multiplication matrices of the (approximate) radical, following Algorithm 21.

The exact system:

$$\mathbf{f} = \begin{cases} 3x_1^2 + 18x_1x_2 - 48x_1 + 21x_2^2 - 114x_2 + 156 \\ x_1^3 - \frac{259}{4}x_1^2x_2 + \frac{493}{4}x_1x_2^2 - \frac{611}{4}x_1x_2 + \frac{2423}{4}x_1x_2 - \frac{1175}{2}x_1 \\ - 5x_2^3 + 6x_2^2 + x_2 + 5 \\ x_1^3 + \frac{81}{4}x_1^2x_2 - \frac{163}{4}x_1x_2^2 + \frac{21}{4}x_1x_2 + \frac{87}{4}x_1x_2 - \frac{151}{2}x_1 - x_2^3 \\ + 4x_2^2 + 2x_2 + 3 \end{cases}$$

$\mathbf{f}$ has common roots $(-1, 3)$ of multiplicity 3 and $(2, 2)$ of multiplicity 2.

The system with clusters:

$$\bar{\mathbf{f}} = \begin{cases} 3x_1^2 + 17.4x_1x_2 - 46.5x_1 + 23.855x_2^2 - 127.977x_2 + 171.933 \\ x_1^3 - 72.943x_1^2x_2 + 139.617x_1x_2^2 - 8.417x_1x_2^2 - 124.161x_1x_2 \\ + 295.0283x_1 - 5x_2^3 + 6x_2^2 + x_2 + 5 \\ x_1^3 + 21.853x_1^2x_2 - 43.658x_1x_2^2 - 27.011x_1x_2^2 + 185.548x_1x_2 \\ - 274.649x_1 - x_2^3 + 4x_2^2 + 2x_2 + 3 \end{cases}$$

$\bar{\mathbf{f}}$ has two clusters: $(-1, 3)$, $(-0.9, 3)$, $(-1.01, 3.1)$ and $(2, 2)$, $(1.9, 2)$ each of radius 10^{-1} .

The perturbed system:

$$\hat{\mathbf{f}} = \begin{cases} 3x_1^2 + 18x_1x_2 - 48x_1 + 21.001x_2^2 - 113.999x_2 + 156.001 \\ 1.001x_1^3 - 64.751x_1^2x_2 + 123.250x_1^2 - 152.750x_1x_2^2 \\ + 605.751x_1x_2 - 587.500x_1 - 4.999x_2^3 + 6.0001x_2^2 + x_2 + 5 \\ x_1^3 + 20.249x_1^2x_2 - 40.750x_1^2 + 5.249x_1x_2^2 + 21.749x_1x_2 \\ - 75.5x_1 - 1.001x_2^3 + 4x_2^2 + 2x_2 + 3 \end{cases}$$

is obtained from $\mathbf{f}$ by a random perturbation of size 10^{-3} . This system has no common roots.

We set $\delta = 6$, $D = 2$ and $\Delta = 6$. The Sylvester matrices in all three cases were size 28×28 and in the first two cases they had rank 23 while in the last case it was full rank. In the first two cases the fact that the corank is 5 indicates that there are 5 solutions, counting multiplicities. For these cases we computed a basis $S := [1, x_1, x_2, x_1x_2, x_1^2]$ for the factor algebra by taking maximum rank submatrices of the Macaulay matrices. In the third case, we simply erased the columns of the Macaulay matrix corresponding to the monomials in S . From here, we chose random elements in the nullspaces of the (cropped) Macaulay matrices to compute the moment matrices:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{6}{7} & -\frac{34}{7} \\ 0 & 0 & -\frac{52}{7} & \frac{10}{7} & -\frac{6}{7} \\ 0 & -\frac{6}{7} & \frac{10}{7} & -\frac{40}{7} & -\frac{36}{7} \\ 0 & -\frac{34}{7} & -\frac{6}{7} & -\frac{36}{7} & -\frac{276}{7} \end{bmatrix},$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.787 & -20.059 \\ 0 & 0 & -7.207 & 1.219 & 1.787 \\ 0 & 1.787 & 1.219 & 7.702 & -43.499 \\ 0 & -20.059 & 1.787 & -43.499 & -43.644 \end{bmatrix} \text{ and } \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.858 & -4.848 \\ 0 & 0 & -7.428 & 1.428 & -0.858 \\ 0 & -0.858 & 1.428 & -5.719 & -5.125 \\ 0 & -4.848 & -0.858 & -5.125 & -39.404 \end{bmatrix}.$$

The polynomials J , computed from the moment matrices are:

$$J = 5 - \frac{3}{10}x_1 - \frac{26}{15}x_2 - \frac{1}{30}x_1x_2 - \frac{1}{5}x_1^2$$

$$\bar{J} = 5 + 0.916x_1 - 1.952x_2 - 0.636x_1x_2 - 0.106x_1^2$$

$$\hat{J} = 4.999 - 0.306x_1 - 1.733x_2 - 0.030x_1x_2 - 0.200x_1^2.$$

After computing the matrices $\text{Syl}_S(J)$ and $\mathfrak{R}_S(\mathbf{y})$, we obtain the matrices of traces:

$$\begin{bmatrix} 5 & 1 & 13 & -1 & 11 \\ 1 & 11 & -1 & 25 & 13 \\ 13 & -1 & 35 & -11 & 25 \\ -1 & 25 & -11 & 59 & 23 \\ 11 & 13 & 25 & 23 & 35 \end{bmatrix},$$

$$\begin{bmatrix} 4.999 & 0.990 & 13.100 & -1.031 & 10.440 \\ 0.990 & 10.440 & -1.031 & 23.812 & 12.100 \\ 13.100 & -1.031 & 35.610 & -11.206 & 23.812 \\ -1.031 & 23.812 & -11.206 & 56.533 & 21.337 \\ 10.440 & 12.100 & 23.812 & 21.337 & 31.729 \end{bmatrix} \text{ and } \begin{bmatrix} 5 & 0.995 & 13.002 & -1.017 & 11.003 \\ 0.995 & 10.999 & -1.015 & 25.019 & 12.913 \\ 13.002 & -1.017 & 35.013 & -11.064 & 25.029 \\ -1.017 & 25.0256 & -11.061 & 59.129 & 22.770 \\ 11.003 & 12.870 & 25.0519 & 22.644 & 34.968 \end{bmatrix}.$$

The first matrix R has rank 2, while $\bar{R}$ and $\hat{R}$ have rank 5. In the first case we follow steps 9 and 10 of Algorithm 21 to obtain the multiplication matrices of the radical with respect to its basis $T = [1, x_1]$:

$$\begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} \frac{7}{3} & -\frac{1}{3} \\ \frac{3}{2} & \frac{1}{2} \end{bmatrix},$$

with respective eigenvalues $[2, -1]$ and $[2, 3]$.

For the second case we use the method described in [22, 23] to compute the approximate multiplication matrices of

the approximate radical of the clusters. Using Gaussian Elimination with complete pivoting, we found that the almost vanishing pivot elements were of the order of magnitude of 10^{-1} which clearly indicated the numerical rank. Using the submatrices obtained from the complete pivoting algorithm we got the following approximate multiplication matrices of the approximate radical with respect to the basis $T = [x_1x_2, x_2]$:

$$\begin{bmatrix} 0.976 & 1 \\ 1.895 & -4.623 \times 10^{-7} \end{bmatrix} \text{ and } \begin{bmatrix} 2.346 & -0.354 \\ -0.671 & 2.691 \end{bmatrix}.$$

The norm of the commutator of these matrices is 0.002 and their eigenvalues are respectively $[1.949, -0.972]$ and $[2.001, 3.036]$. Note that the corresponding roots $[1.949, 2.001]$ and $[-0.972, 3.036]$ are within 10^{-2} distance from the centers of gravity of the clusters, as was shown in [22, 23] (recall that the radius of the clusters was 10^{-1}).

In the third case, the numerical rank was not easy to determine using either SVD or complete pivoting. However, when we assume that the numerical rank of R is 2, and we cut the matrix R using the output of the complete pivoting algorithm, then we obtain the multiplication matrices with respect to the basis $T = [x_1x_2, x_2]$:

$$\begin{bmatrix} 1.005 & 0.992 \\ 1.992 & 0.005 \end{bmatrix} \text{ and } \begin{bmatrix} 2.327 & -0.330 \\ -0.663 & 2.664 \end{bmatrix}.$$

The norm of the commutator of these matrices is 0.010 and their eigenvalues are respectively $[1.997, -0.987]$ and $[1.999, 2.993]$ (recall that the perturbation of the polynomials was of size 10^{-3}).

5. CONCLUSION

In this paper we gave an algorithm to compute matrices of traces and the radical of an ideal $\mathcal{I}$ which has finitely many projective common roots, none of them at infinity and its factor algebra is Gorenstein. A follow-up paper will consider an extension of the above algorithm which also works in the non-Gorenstein case and for systems which have roots at infinity, as well as an alternative method using Bezout matrices for the affine complete intersection case to compute the radical $\sqrt{\mathcal{I}}$.

6. REFERENCES

- [1] I. Armendáriz and P. Solernó. On the computation of the radical of polynomial complete intersection ideals. In *AAECC-11: Proceedings of the 11th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, pages 106–119, 1995.
- [2] E. Becker. Sums of squares and quadratic forms in real algebraic geometry. In *De la géométrie algébrique réelle (Paris, 1990)*, volume 1 of *Cahiers Sém. Hist. Math. Sér. 2*, pages 41–57. 1991.
- [3] E. Becker, J. P. Cardinal, M.-F. Roy, and Z. Szafraniec. Multivariate Bezoutians, Kronecker symbol and Eisenbud-Levine formula. In *Algorithms in algebraic geometry and applications (Santander, 1994)*, volume 143 of *Progr. Math.*, pages 79–104.
- [4] E. Becker and T. Wörmann. On the trace formula for quadratic forms. In *Recent advances in real algebraic geometry and quadratic forms*, volume 155 of *Contemp. Math.*, pages 271–291. 1994.
- [5] E. Becker and T. Wörmann. Radical computations of zero-dimensional ideals and real root counting. In *Selected papers presented at the international IMACS symposium on Symbolic computation, new trends and developments*, pages 561–569, 1996.
- [6] E. Briand and L. Gonzalez-Vega. Multivariate Newton sums: Identities and generating functions. *Communications in Algebra*, 30(9):4527–4547, 2001.

- [7] J. Cardinal and B. Mourrain. Algebraic approach of residues and applications. In J. Reneger, M. Shub, and S. Smale, editors, *Proceedings of AMS-Siam Summer Seminar on Math. of Numerical Analysis (Park City, Utah, 1995)*, volume 32 of *Lectures in Applied Mathematics*, pages 189–219, 1996.
- [8] E. Cattani, A. Dickenstein, and B. Sturmfels. Computing multidimensional residues. In *Algorithms in algebraic geometry and applications (Santander, 1994)*, volume 143 of *Progr. Math.*, pages 135–164, 1996.
- [9] E. Cattani, A. Dickenstein, and B. Sturmfels. Residues and resultants. *J. Math. Sci. Univ. Tokyo*, 5(1):119–148, 1998.
- [10] D. A. Cox, J. B. Little, and D. O’Shea. *Using Algebraic Geometry*, volume 185 of *Graduate Texts in Mathematics*. Springer-Verlag, NY, 1998. 499 pages.
- [11] R. E. Curto and L. A. Fialkow. Solution of the truncated complex moment problem for flat data. *Mem. Amer. Math. Soc.*, 119(568):x+52, 1996.
- [12] C. D’Andrea and G. Jeronimo. Rational formulas for traces in zero-dimensional algebras. <http://arxiv.org/abs/math.AC/0503721>, 2005.
- [13] G. M. Díaz-Toca and L. González-Vega. An explicit description for the triangular decomposition of a zero-dimensional ideal through trace computations. In *Symbolic computation: solving equations in algebra, geometry, and engineering (South Hadley, MA, 2000)*, volume 286 of *Contemp. Math.*, pages 21–35, 2001.
- [14] L. Dickson. *Algebras and Their Arithmetics*. University of Chicago Press, 1923.
- [15] M. Elkadi and B. Mourrain. *Introduction à la résolution des systèmes polynomiaux*, volume 59 of *Mathématiques et Applications*. 2007.
- [16] P. Gianni and T. Mora. Algebraic solution of systems of polynomial equations using Gröbner bases. In *Applied algebra, algebraic algorithms and error-correcting codes (Menorca, 1987)*, volume 356 of *Lecture Notes in Comput. Sci.*, pages 247–257, 1989.
- [17] P. Gianni, B. Trager, and G. Zacharias. Gröbner bases and primary decomposition of polynomial ideals. *J. Symbolic Comput.*, 6(2-3):149–167, 1988. Computational aspects of commutative algebra.
- [18] L. González-Vega. The computation of the radical for a zero dimensional ideal in a polynomial ring through the determination of the trace for its quotient algebra. *Preprint*, 1994.
- [19] L. González-Vega and G. Trujillo. Using symmetric functions to describe the solution set of a zero-dimensional ideal. In *Applied algebra, algebraic algorithms and error-correcting codes (Paris, 1995)*, volume 948 of *Lecture Notes in Comput. Sci.*, pages 232–247.
- [20] G.-M. Greuel and G. Pfister. *A Singular introduction to commutative algebra*. 2002. With contributions by Olaf Bachmann, Christoph Lossen and Hans Schönemann, With 1 CD-ROM (Windows, Macintosh, and UNIX).
- [21] W. Heiß, U. Oberst, and F. Pauer. On inverse systems and squarefree decomposition of zero-dimensional polynomial ideals. *J. Symbolic Comput.*, 41(3-4):261–284, 2006.
- [22] I. Janovitz-Freireich, L. Rónyai, and Ágnes Szántó. Approximate radical of ideals with clusters of roots. In *ISSAC ’06: Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, pages 146–153, 2006.
- [23] I. Janovitz-Freireich, L. Rónyai, and Ágnes Szántó. Approximate radical for clusters: a global approach using gaussian elimination or svd. *Mathematics in Computer Science*, 1(2):393–425, 2007.
- [24] H. Kobayashi, S. Moritsugu, and R. W. Hogan. On radical zero-dimensional ideals. *J. Symbolic Comput.*, 8(6):545–552, 1989.
- [25] T. Krick and A. Logar. An algorithm for the computation of the radical of an ideal in the ring of polynomials. In *Applied algebra, algebraic algorithms and error-correcting codes (New Orleans, LA, 1991)*, volume 539 of *Lecture Notes in Comput. Sci.*, pages 195–205.
- [26] T. Krick and A. Logar. Membership problem, representation problem and the computation of the radical for one-dimensional ideals. In *Effective methods in algebraic geometry (Castiglione, 1990)*, volume 94 of *Progr. Math.*, pages 203–216, 1991.
- [27] E. Kunz. *Kähler differentials*. Advanced lectures in Mathematics. Friedr. Vieweg and Sohn, 1986.
- [28] Y. N. Lakshman. On the complexity of computing a Gröbner basis for the radical of a zero-dimensional ideal. In *In Proceedings of the Twenty Second Symposium on Theory of Computing*, pages 555–563, 1990.
- [29] Y. N. Lakshman. A single exponential bound on the complexity of computing Gröbner bases of zero-dimensional ideals. In *Effective methods in algebraic geometry (Castiglione, 1990)*, volume 94 of *Progr. Math.*, pages 227–234, 1991.
- [30] Y. N. Lakshman and D. Lazard. On the complexity of zero-dimensional algebraic systems. In *Effective methods in algebraic geometry (Castiglione, 1990)*, volume 94 of *Progr. Math.*, pages 217–225, 1991.
- [31] J. B. Lasserre, M. Laurent, and P. Rostalski. A unified approach to computing real and complex zeros of zero-dimensional ideals. *preprint*, 2007.
- [32] J. B. Lasserre, M. Laurent, and P. Rostalski. Semidefinite characterization and computation of zero-dimensional real radical ideals. *to appear in Foundations of Computational Mathematics*, 2007.
- [33] D. Lazard. Résolution des systèmes d’équations algébriques. *Theoret. Comput. Sci.*, 15(1):77–110, 1981.
- [34] B. Mourrain. Computing isolated polynomial roots by matrix methods. *J. of Symbolic Computation, Special Issue on Symbolic-Numeric Algebra for Polynomials*, 26(6):715–738, Dec. 1998.
- [35] P. Pedersen, M.-F. Roy, and A. Szpirglas. Counting real zeros in the multivariate case. In *Computational algebraic geometry (Nice, 1992)*, volume 109 of *Progr. Math.*, pages 203–224. Boston, MA, 1993.
- [36] F. Rouiller. Solving zero-dimensional systems through the rational univariate representation. In *AAECC: Applicable Algebra in Engineering, Communication and Computing*, volume 9, pages 433–461, 1999.
- [37] G. Scheja and U. Storch. Über Spurfunktionen bei vollständigen Durchschnitt. *J. Reine Angew Mathematik*, 278:174–190, 1975.
- [38] R. P. Stanley. *Combinatorics and commutative algebra*, volume 41 of *Progress in Mathematics*. Birkhäuser, 1996.
- [39] K. Yokoyama, M. Noro, and T. Takeshima. Solutions of systems of algebraic equations and linear maps on residue class rings. *J. Symbolic Comput.*, 14(4):399–417, 1992.
- [40] L. Zhi and G. Reid. Solving nonlinear polynomial systems via symbolic-numeric elimination method. In *In Proceedings of the International Conference on Polynomial System Solving*, pages 50–53, 2004.

Integration of Algebraic Functions: A Simple Heuristic for Finding the Logarithmic Part

Manuel Kauers^{*}
RISC-Linz
Johannes Kepler Universität
A-4040 Linz, Austria
mkauers@risc.uni-linz.ac.at

ABSTRACT

A new method is proposed for finding the logarithmic part of an integral over an algebraic function. The method uses Gröbner bases and is easy to implement. It does not have the feature of finding a closed form of an integral whenever there is one. But it very often does, as we will show by a comparison with the built-in integrators of some computer algebra systems.

Categories and Subject Descriptors

I.1.2 [Computing Methodologies]: Symbolic and Algebraic Manipulation—*Algorithms*

General Terms

Algorithms

Keywords

Symbolic Integration, Algebraic Functions

1. INTRODUCTION

In 1979, Norman and Davenport wrote [10]

Ten years ago an integration program could be judged by the proportion of some set of (known integrable) examples it could cope with, whereas now failure to solve an easy problem is seen as a bug.

They characterize here the development from early heuristic approaches to solid algebraic algorithms for integration. Indeed, there is now a complete algorithmic solution to the problem of indefinite integration of elementary functions, “complete” in the sense that there is an algorithm which is guaranteed to find a closed form of an integral in terms of

^{*}Supported by the Austrian science foundation FWF, grants P19462-N18 and SFB F1305.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC’08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

elementary functions whenever there is one [11, 12, 1, 5]. In this situation, why should we still care about incomplete heuristic approaches?

Parts of the complete integration algorithm are rather complicated. Implementors need a thorough mathematical understanding of the underlying theory and a lot of time to create a complete implementation of the complete algorithm. As a matter of fact, all of today’s computer algebra systems only contain a partial implementation of the complete algorithm. Difficult parts of the algorithm that are only needed for certain types of integrals are often considered not worth the effort of implementing and are therefore left out. A simple heuristic may be an attractive way of filling these gaps. And even if a complete code is desired, a fast heuristic could be executed as a preprocessor before the complete code is entered. This may enhance the overall performance.

These reasons were stated by Bronstein [6, 5] in favor of parallel integration, an incomplete integration alternative to the Risch algorithm for integration of transcendental elementary functions. An incomplete (yet very successful) integrator of his based on this approach requires less than 100 lines of Maple code [4].

In this paper, we propose an incomplete method for finding the logarithmic part of an algebraic integral. As implementing our method requires no more than about ten lines of code in Maple or Mathematica, it might be interesting in situations where a full implementation of Davenport’s [8] or Trager’s [13] algorithm, which both involve complicated and time-consuming algebraic computations, is not adequate.

We use Gröbner bases to compute potential contributions to the logarithmic part. Very often, we can find the complete logarithmic part in this way. At least some components of the logarithmic part can usually be found, and some unintegrated part of the integrand may be left. We cannot give any proofs as to how often our method succeeds, but instead we measure its usefulness by comparing it with the integrators of Axiom, Maple, and Mathematica. We do believe that the integrators of these systems would benefit from including our method.

2. TRAGER’S ALGORITHM

Let k be a field, $\mathbb{Q} \subseteq k$. Let $m \in k[x, y]$ with $d := \deg_y m \geq 2$ irreducible over $\bar{k}(x)$. We consider the differential field $K := k(x)[y]/\langle m \rangle$ with the derivation D defined via $Dc = 0$ ($c \in k$), $Dx = 1$. Elements of K whose minimal polynomial has a leading coefficient with respect to y that is

free of x are called integral. We may assume that y itself is integral in K (otherwise, choose a different generator). For polynomials p we will write $\bar{p}$ for its residue class modulo an ideal that will be clear from the context. We will sloppily write $p(x, y, \gamma)$ for the polynomial obtained from p by substituting γ for the last indeterminate, and in similar situations. All these definitions and conventions will be used throughout the paper.

Suppose an integrand $f \in K$ is given. If the integral $\int f$ is elementary, then, according to Liouville's theorem, there exist $g \in K$ and $\gamma_1, \dots, \gamma_r \in \bar{k}$ and $p_1, \dots, p_r \in K(\gamma_1, \dots, \gamma_r)$ such that

$$f = Dg + \gamma_1 \frac{Dp_1}{p_1} + \dots + \gamma_r \frac{Dp_r}{p_r},$$

and hence $\int f = g + \gamma_1 \log(p_1) + \dots + \gamma_r \log(p_r)$. Here, g is called the algebraic part and the sum of the $\gamma_i \log(p_i)$ is called the logarithmic part of the integral.

Following Trager [13], g , the γ_i and the p_i may be computed as follows.

1. Choose a point $x_0 \in k$ where f has no pole or branch point, and perform the change of variables $x = 1/(x' - x_0)$, $dx = -1/x'^2 dx'$. We call the new integrand again f and rename x' back to x .
2. Compute an integral basis $\omega = (\omega_1, \dots, \omega_d)$ of K . This is a basis which generates the subring $\mathcal{O}_{k[x]} \subseteq K$ of all integral elements as a $k[x]$ -module.
3. Write

$$f = \frac{a_1 \omega_1 + \dots + a_d \omega_d}{b}$$

for $a_1, \dots, a_d, b \in k[x]$ such that $\gcd(a_1, \dots, a_d, b) = 1$. Then, using a generalization of Hermite reduction, compute $g, h \in K$ such that h has simple poles only and

$$f = Dg + h.$$

4. If the algebraic function $x^2 h(1/x)$ has a pole at the origin, return “not integrable”; the integral is not elementary in this case.
5. Write

$$h = \frac{u_1 + \dots + u_d y^{d-1}}{v}$$

for $u_1, \dots, u_d, v \in k[x]$ such that $\gcd(u_1, \dots, u_d, v) = 1$. Compute the splitting field of

$$\text{res}_x(\text{res}_y(u_1 + \dots + u_d y^{d-1} - tDv, m), v) \in k[t],$$

say $\gamma_1, \dots, \gamma_r \in \bar{k}$ generate this field as a vector space over $\mathbb{Q}$.

6. For each γ_i , construct an ideal $\mathfrak{a}_i \trianglelefteq \mathcal{O}_{k[x]}$ that encodes the finite places and multiplicities of the singularities a potential $\log$ and p_i with coefficient γ_i has to have.
7. For each $\mathfrak{a}_i$, determine whether there is a positive integer n_i such that $\mathfrak{a}_i^{n_i}$ is a principal ideal. If so, say $\mathfrak{a}_i^{n_i} = \langle p_i \rangle$ for some $p_i \in \mathcal{O}_{k[x]}$, then $\frac{\gamma_i}{n_i} \log(p_i)$ is the desired contribution to the logarithmic part of the integral. If no n_i exists, then return “not integrable”; the integral is not elementary in this case.

8. After undoing the substitution of Step 1, return

$$g + \frac{\gamma_1}{n_1} \log(p_1) + \dots + \frac{\gamma_r}{n_r} \log(p_r).$$

The computation of an integral basis in Step 2 is easy if y is a radical over $k(x)$, i.e., if $m = ay^d - b$ for some $a, b \in k[x]$ [3]. Otherwise, the computation is not trivial, but algorithms for this purpose are known [13, 14]. The integral basis is needed for the Hermite reduction and also in Step 7. For a modified Hermite reduction [2, 3], however, an integral basis is not needed in the first place. The standard basis $1, y, \dots, y^{d-1}$ can be used instead as a first “approximation” to the integral basis. The modified Hermite reduction then returns, in addition to g and h , a refined basis $\omega_1, \dots, \omega_d$, which is just as close to an integral basis as was necessary for finding the desired g and h .

For details about steps 5, 6, and 7, we refer to Trager's thesis [13]. These are parts that are often only partially implemented (if at all), and for these we will give a simple alternative below.

3. CZICHOWSKI'S OBSERVATION

For the integration of rational functions $f \in k(x)$, Czychowski [7] observed that the logarithmic part can be read off directly from a certain Gröbner basis. Let $f = u/v$ with $u, v \in k[x]$ such that $\gcd(u, v) = \gcd(v, Dv) = 1$ and $\deg u < \deg v$, and consider the Gröbner basis

$$G = \{g_0, g_1, \dots, g_n\} \subseteq k[x, t]$$

of the ideal $\langle v, u - tDv \rangle \subseteq k[x, t]$ with respect to the lexicographic order eliminating x . Denote by $c_i := \text{cont}_x(g_i) \in k[t]$ and $p_i := \text{pp}_x(g_i) \in k[x, t]$ the contents and the primitive parts of the g_i with respect to x ($i = 0, \dots, n$), and suppose the g_i are sorted according to ascending leading terms. Then:

- $p_0 = c_n = 1$
- c_0 is the square free part of the Rothstein-Trager resultant $\text{res}_x(v, u - tDv) \in k[t]$.
- $c_i \mid c_{i-1}$ ($i = 1, \dots, n$)
- for $q_i := c_{i-1}/c_i \in k[t]$ ($i = 1, \dots, n$) we have

$$\int \frac{u}{v} = \sum_{i=1}^n \sum_{\gamma: q_i(\gamma)=0} \gamma \log(p_i(x, \gamma)).$$

EXAMPLE 1. For $u = x^3 + 9x^2 - 18x + 9$ and $v = x^4 - 17x^2 - 18$ we have

$$G = \{(2t-1)(8t^2-9), (2t-1)(x-4t), 7x^2-152t^2+45\}$$

and therefore

$$\begin{aligned} & \int \frac{x^3 + 9x^2 - 18x + 9}{x^4 - 17x^2 - 18} dx \\ &= \sum_{\gamma: 8\gamma^2-9=0} \gamma \log(x-4\gamma) + \sum_{\gamma: 2\gamma-1=0} \gamma \log(7x^2-152\gamma^2+45) \\ &= \frac{3}{\sqrt{8}} \log(x-3\sqrt{2}) - \frac{3}{\sqrt{8}} \log(x+3\sqrt{2}) + \frac{1}{2} \log(7x^2+7) \end{aligned}$$

The following facts are immediate consequences of Czychowski's observation.

- If c_0 is irreducible then $G = \{c_0, p_1\}$ with

$$p_1(x, \gamma) = \gcd(u - \gamma Dv, v)$$

where γ is a root of c_0 .

- More generally, if q is an irreducible factor of c_0 , then the Gröbner basis of $\langle q, u - tDv, v \rangle \in k[x, t]$ with respect to an order eliminating x will have the form $\{q, p\}$ with $p(x, \gamma) = \gcd(u - \gamma Dv, v)$ where γ is a root of q .
- Consequently, if $q^{(1)}, \dots, q^{(m)} \in k[t]$ are all the irreducible factors of c_0 and if $p^{(i)} \in k[x, t]$ is the corresponding element in the Gröbner basis of $\langle q^{(i)}, u - tDv, v \rangle$ ($i = 1, \dots, m$), then

$$\int \frac{u}{v} = \sum_{i=1}^m \sum_{\gamma: q^{(i)}(\gamma)=0} \gamma \log(p^{(i)}(x, \gamma)).$$

4. THE ALGEBRAIC CASE

Our goal is to extract the logarithmic part of an algebraic function integral from a Gröbner basis, similar as Czichowski does it for a rational function integral. Suppose, to this end, that for a given $f \in K = k(x)[y]/\langle m \rangle$ we have executed Trager's algorithm up to, and including, step 4 (cf. Section 2). The remaining integrand is then of the form u/v for some $u \in k[x, y]$ and $v \in k[x]$. We may assume that u/v has at least a double root at infinity.

In the rational case, we can identify the ideals $\langle q, v, u - tDv \rangle = \langle q, p \rangle \trianglelefteq k[x, t]$ for irreducible $q \in k[t]$ with principal ideals $\langle \bar{p} \rangle \trianglelefteq k[x, t]/\langle q \rangle$. These ideals give rise to a contribution $\sum_{\gamma: q(\gamma)=0} \gamma \log(p(x, \gamma))$ to the logarithmic part. Likewise, if in the algebraic case we can find some $p \in k[x, y]$ with $\langle q, m, v, u - tDv \rangle = \langle q, m, p \rangle \trianglelefteq k[x, y, t]$, then this ideal can be identified with the principal ideal $\langle \bar{p} \rangle \trianglelefteq k[x, y, t]/\langle q, m \rangle$. It gives rise to a contribution $\sum_{\gamma: q(\gamma)=0} \gamma \log(p(x, y, \gamma))$ to the logarithmic part of the integral.

EXAMPLE 2. For $u = y$ and $v = x^4 + 1$ with $m = y^2 - (x^2 + 1)$ and using $q = 128t^4 + 16t^2 + 1$ we find

$$\{128t^4 + 16t^2 + 1, y^2 - 16t^2 - 2, x - 32t^3y\}$$

as the Gröbner basis of $\langle q, m, v, u - tDv \rangle$ with respect to lexicographic order $x > y > t$. It is easily checked that in fact

$$\langle v, u - tDv, m \rangle = \langle q, m, x - 32t^3y \rangle$$

(both ideals have the same Gröbner basis). Indeed,

$$\int \frac{\sqrt{x^2+1}}{x^4+1} dx = \sum_{\gamma: 128\gamma^4+16\gamma^2+1=0} \gamma \log(x - 32\gamma^3 \sqrt{x^2+1}).$$

The Gröbner basis of $\langle m, v, u - tDv \rangle \trianglelefteq k[x, y, t]$ with respect to an order eliminating x and y will contain one univariate polynomial in t . For an irreducible polynomial $q \in k[t]$, the ideal $\langle q, m, v, u - tDv \rangle$ will be nontrivial iff q is a divisor of this polynomial. This restricts q to finitely many candidates that can be considered one after the other.

EXAMPLE 3. For $u = ix + y$ and $v = x^4 + 1$ with $m = y^2 - (x^2 + 1)$ we find

$$\begin{aligned} &\{(16t^2 - 8t + (2-i))(16t^2 + 8t + (2+i)), \\ &128it^3 + 16t^2 - 16it + 7y^2 - 1, \\ &192yt^3 - 80iyt^2 + 4yt + 14x - 9iy\} \end{aligned}$$

as the Gröbner basis of $\langle m, v, u - tDv \rangle \trianglelefteq k[x, y, t]$. For $q = 16t^2 - 8t + (2-i)$, we have

$$\langle q, v, u - tDv \rangle = \langle q, m, 2x + (1-i)(4t-1)y \rangle,$$

while for $q = 16t^2 + 8t + (2+i)$, we have

$$\langle q, v, u - tDv \rangle = \langle q, m, 2x + (1+i)(4t+1)y \rangle.$$

Indeed,

$$\begin{aligned} &\int \frac{ix + \sqrt{x^2+1}}{x^4+1} dx \\ &= \sum_{\gamma: 16\gamma^2-8\gamma+(2-i)=0} \gamma \log(2x + (1-i)(4\gamma-1)\sqrt{x^2+1}) \\ &\quad + \sum_{\gamma: 16\gamma^2+8\gamma+(2+i)=0} \gamma \log(2x + (1+i)(4\gamma+1)\sqrt{x^2+1}). \end{aligned}$$

For a given ideal $\langle q, m, v, u - tDv \rangle \trianglelefteq k[x, y, t]$, there may or may not exist a $p \in k[x, y, t]$ such that $\langle q, m, v, u - tDv \rangle = \langle q, m, p \rangle$. It is not trivial to decide whether such a “principal generator” p exists, and even if it is known that there is one it is not obvious how to find it. But it turns out that often a principal generator p will belong to the Gröbner basis of $\langle q, m, v, u - tDv \rangle$ with respect to a block order $[x, y] > [t]$ which orders the block $[x, y]$ by a degree order, say degrevlex. We therefore suggest to compute this Gröbner basis and consider its elements as candidates for p . For each candidate p , we can simply check whether the ideal $\langle q, m, p \rangle$ has the same Gröbner basis, and if so, we have found a contribution to the logarithmic part. We know of no convincing algebraic justification of this heuristic, but we can assert that it does succeed in many cases (see the next section).

In order to give rise to a contribution to the logarithmic part of an integral, it is not necessary that $\mathfrak{a} := \langle \bar{v}, \bar{u} - tD\bar{v} \rangle \trianglelefteq k[x, y, t]/\langle q, m \rangle$ itself is a principal ideal. It suffices that some power of it is. For if $n \in \mathbb{N}$ and $p \in k[x, y, t]$ are such that $\mathfrak{a}^n = \langle \bar{p} \rangle$, this would give a contribution $\sum_{\gamma: q(\gamma)=0} \frac{\gamma}{n} \log(p(x, y, \gamma))$ to the logarithmic part. Therefore, if we fail to find a p with $\langle q, m, v, u - tDv \rangle = \langle q, m, p \rangle$ we check $\langle q, m \rangle + \langle v, u - tDv \rangle^2$, $\langle q, m \rangle + \langle v, u - tDv \rangle^3$, and so on. One of the main difficulties in the construction of a complete integration procedure is finding a bound on the exponent n above which it can be asserted that the integral is not elementary. But as our approach is heuristic anyway, we need not bother about finding a rigorous bound but choose some fixed number, say 12, and give up if we exceed this power without having found anything. (If the divisor is over the rationals, the choice of 12 is in fact a rigorous bound, as pointed out by one of the referees.)

EXAMPLE 4. For $u = y$ and $v = x^3 + 1$ with $m = y^2 - (x^2 + 1)$ we find

$$\begin{aligned} &\{(9t^2 - 2)(9t^2 + 1), y + 9yt^2 - 27t^3 - 3t, \\ &x + 3ty - 1, y^2 + 3yt - 9t^2 - 2\} \end{aligned}$$

as the Gröbner basis of $\langle m, v, u - tDv \rangle$. The Gröbner basis of $\langle 9t^2 + 1, m, v, u - tDv \rangle$ is

$$\{9t^2 + 1, x + 3ty - 1, y^2 + 3ty - 1\},$$

and it turns out that

$$\langle 9t^2 + 1, m, v, u - tDv \rangle = \langle 9t^2 + 1, m, x + 3ty - 1 \rangle.$$

The Gröbner basis of $\langle 9t^2 - 2, m, v, u - tDv \rangle$ is

$$\{9t^2 - 2, y - 3t, x + 1\},$$

but

$$\langle 9t^2 - 2, m, v, u - tDv \rangle \neq \langle 9t^2 - 2, m, y - 3t \rangle \quad \text{and}$$

$$\langle 9t^2 - 2, m, v, u - tDv \rangle \neq \langle 9t^2 - 2, m, x + 1 \rangle,$$

so there seems to be no principal generator for $n = 1$. The Gröbner basis for $\langle 9t^2 - 2, m \rangle + \langle v, u - tDv \rangle^2$ is

$$\{9t^2 - 2, x + 3ty - 1, y^2 - 6ty + 2\},$$

and it turns out that

$$\langle 9t^2 - 2, m \rangle + \langle v, u - tDv \rangle^2 = \langle 9t^2 - 2, m, x + 3ty - 1 \rangle.$$

Indeed,

$$\begin{aligned} \int \frac{\sqrt{x^2 + 1}}{x^3 + 1} dx &= \sum_{\gamma: 9\gamma^2 + 1 = 0} \gamma \log(x - 1 + 3\gamma\sqrt{x^2 + 1}) \\ &+ \sum_{\gamma: 9\gamma^2 - 2 = 0} \frac{\gamma}{2} \log(x - 1 + 3\gamma\sqrt{x^2 + 1}). \end{aligned}$$

We have implicitly assumed so far that the integral closure $\mathcal{O}_{k[x]}$ of $k[x]$ in K is just $k[x, y]$, but it may actually be more than this. Just using $k[x, y]$ then in a computation may cause some contributions to the logarithmic part to be overlooked.

EXAMPLE 5. Consider the integral

$$\int \frac{1}{(4x + 4 - \sqrt[3]{x^2(x+1)}) \sqrt[3]{x^2(x+1)}} dx.$$

Let

$$\begin{aligned} u &= 4(x+1)x^2 + x^2y + 16(x+1)y^2, \\ v &= x^2(x+1)(7x+8)(9x+8) \end{aligned}$$

and $m = y^3 - x^2(x+1)$. Then the integrand can be written $u(x, \sqrt[3]{x^2(x+1)})/v(x)$.

Note that despite of the double factor x in v , the integrand has simple poles only. It turns out that no principal generators can be found in the Gröbner bases of ideals $\langle q, m \rangle + \langle v, u - tDv \rangle^n \subseteq k[x, y, t]$ for $n = 1, \dots, 30$ and any irreducible $q \in k[t]$.

By adjoining new formal elements to $k[x, y]$ and stipulating appropriate relations between them, the whole integral closure $\mathcal{O}_{k[x]}$ can be taken into account during the computation.

EXAMPLE 6. Continuing Example 5, an integral basis for $K = k(x)[y]/\langle m \rangle$ is given by $(1, y, y^2/x)$. Introducing a new indeterminate z referring to y^2/x , the integrand may be written r/s where

$$\begin{aligned} r &= 4(x+1)x + xy + 16(x+1)z \\ s &= x(x+1)(7x+8)(9x+8). \end{aligned}$$

Let $\mathfrak{m} = \langle y^3 - x^2(x+1), z^3 - x(x+1)^2, zx - y^2 \rangle \subseteq k[x, y, z, t]$. Then $\mathfrak{m} + \langle s, r - tDs \rangle \cap k[t] = \langle (4t-3)(4t+3) \rangle$ and we have

$$\begin{aligned} \langle 4t-3 \rangle + \mathfrak{m} + \langle s, r - tDs \rangle^3 &= \langle 4t-3 \rangle + \mathfrak{m} + \langle 8+9x+6y+12z \rangle, \\ \langle 4t+3 \rangle + \mathfrak{m} + \langle s, r - tDs \rangle^3 &= \langle 4t+3 \rangle + \mathfrak{m} + \langle 8+7x+6y-12z \rangle. \end{aligned}$$

The principal generators were found in the Gröbner basis of the ideals on the left with respect to a block order $[x, y, z] > [t]$ ordering $[x, y, z]$ by degrevlex. Indeed,

$$\begin{aligned} &\int \frac{1}{(4x+4-\sqrt[3]{x^2(x+1)}) \sqrt[3]{x^2(x+1)}} dx \\ &= \frac{1}{4} \log(8+9x+6\sqrt[3]{x^2(x+1)}+12\sqrt[3]{x(x+1)^2}) \\ &\quad - \frac{1}{4} \log(8+7x+6\sqrt[3]{x^2(x+1)}-12\sqrt[3]{x(x+1)^2}). \end{aligned}$$

When y is not a radical over $k(x)$ the computation of an integral basis is a difficult and expensive task. Lazy Hermite reduction [2, 3] does not require an integral basis as input but begins the computation with $(1, y, \dots, y^{d-1})$ and refines this basis as much as necessary to complete the reduction. The approximation to the integral basis thus obtained may be used in search of the logarithmic part, as described above. No full integral basis is necessary to proceed this way, but of course the finer the approximation the more likely a logarithmic part will be found.

Additional formal elements can also be used if the integrand involves several different algebraic functions: instead of computing with a primitive element, each algebraic function arising in the integrand may be represented by an individual indeterminate.

EXAMPLE 7. For

$$\begin{aligned} u &= (x-1)(4x+1)y + (x+1)(4x-3)z - 2(2x-1)yz, \\ v &= (x-1)(x+1)(4x-5) \end{aligned}$$

with $\mathfrak{m} = \langle y^2 - (x+1), z^2 - (x-1) \rangle$ we find first

$$\mathfrak{m} + \langle v, u - tDv \rangle \cap k[t] = \langle t^2(t-1)(t+3) \rangle$$

and then

$$\begin{aligned} \langle t-1 \rangle + \mathfrak{m} + \langle v, u - tDv \rangle^2 &= \langle t-1 \rangle + \mathfrak{m} \\ &\quad + \langle 16-20x - (9-12x)y + (13+4x)z - 12yz \rangle \\ \langle t+3 \rangle + \mathfrak{m} + \langle v, u - tDv \rangle^2 &= \langle t+3 \rangle + \mathfrak{m} + \langle 4+3y-z \rangle. \end{aligned}$$

Indeed,

$$\begin{aligned} &\int \left((x+1)(4x-3)\sqrt{x-1} + (x-1)(4x+1)\sqrt{x+1} \right. \\ &\quad \left. - 2(2x-1)\sqrt{x-1}\sqrt{x+1} \right) / (x-1)(x+1)(4x-5) dx \\ &= \frac{1}{2} \log(16-20x - (9-12x)\sqrt{x+1} \\ &\quad + (13+4x)\sqrt{x-1} - 12\sqrt{x+1}\sqrt{x-1}) \\ &\quad + \frac{1}{2} \log(4+3\sqrt{x+1} - \sqrt{x-1}). \end{aligned}$$

Alternatively, we may rephrase the integrand in terms of a primitive element. Let q be an indeterminate representing $\sqrt{x+1} + \sqrt{x-1}$ and $m = q^4 - 4xq^2 + 4$. Then the integrand may be written as $r/s \in k(x)[q]/\langle m \rangle$ with

$$\begin{aligned} r &= (2x-1)q^3 - 2(2x-1)q^2 + 2(x-2)q + 4x(2x-1), \\ s &= 4(x-1)(x+1)(4x-5). \end{aligned}$$

We find $\langle m, s, r - tDs \rangle \cap k[t] = \langle t^2(2t-1)(2t+3) \rangle$ and

$$\begin{aligned} \langle 2t-1, m, s, r - tDs \rangle &= \langle 2t-1, m, q^3 - 2q^2 - q + 2 \rangle \\ \langle 2t+3, m, s, r - tDs \rangle &= \langle 2t+3, m, q + 2 \rangle. \end{aligned}$$

Substituting $\sqrt{x+1} + \sqrt{x-1}$ for q leads to the alternative closed form

$$\begin{aligned} & \log(2 - 4x - (3 - 4x)\sqrt{x+1} \\ & \quad + (1 + 4x)\sqrt{x-1} - 4\sqrt{x+1}\sqrt{x-1}) \\ & + \log(\sqrt{x+1} + \sqrt{x-1} + 2). \end{aligned}$$

Summarizing, our procedure for finding the logarithmic part of an integral over an algebraic function u/v is as follows.

```
G := GröbnerBasis({v, u - tDv} ∪ M); int = 0
for all irreducible factors q of min G do
  A := {1}
  for n from 1 to 12 do
    A := GröbnerBasis((A · G) ∪ {q} ∪ M)
    for all p in A do
      if A = GröbnerBasis({q, p} ∪ M) then
        int := int + ∑γ:q(γ)=0  $\frac{2}{n}$  log(p(x, y, γ))
      next q
return int
```

NOTATIONAL REMARKS.

- GröbnerBasis is meant to compute a Gröbner basis in $k[x, y, t]$ with respect to a block order $[x, y] > [t]$ using the degrevlex order for breaking ties in the block $[x, y]$.
- M is meant to contain the relations among the generators, typically $M = \{m\}$, but there may be more relations (and more indeterminates) in the case of non-trivial integral closures or multiple algebraic functions in the integrand.
- $\min G$ is meant to refer to the element of G with the lowest leading term. This is the unique element of G involving t but not x or y .
- If the polynomial $\min G$ has repeated factors, it is understood that the outer loop takes into account the multiplicities by repeating the body of a repeated factor an according number of times (cf. Section 6.3).
- $A \cdot G$ is meant to refer to the ideal product, i.e., if $A = \{a_1, \dots, a_r\}$ and $G = \{g_1, \dots, g_t\}$ then $A \cdot G := \{a_i g_j : i = 1, \dots, r, j = 1, \dots, t\}$.
- The statement “next q ” is meant to break the two inner loops and proceed with the next iteration of the outermost loop.

It is an easy matter to implement this procedure in an actual computer algebra system. In the appendix, we give a sample code for Mathematica.

5. EXPERIMENTS

Though Norman and Davenport may find it antiquated, we will judge our method by the proportion of some set of (known integrable) examples it can cope with. We compare its performance to the built-in integrators of Maple 10, Mathematica 5.2, and Axiom 3.9, knowing that any comparison of this sort is unfair to some extent.

We chose four different algebraic functions and randomly generated 1000 logarithmic expressions for each, which we

differentiated to obtain candidate integrands in $\mathbb{Q}(x, y)$. A typical integrand from our collection is

$$\begin{aligned} & \left(-214632x^7 - 90880x^5 - 39020x^4 - 238623x^3 \right. \\ & \quad + 4160x + 7378 + (-128496x^4 - 143517x^3 + 1300x \\ & \quad \left. + 12614)\sqrt{x^4+1} \right) / \\ & \left(17424x^8 + 7040x^6 + 6138x^5 + 40414x^4 \right. \\ & \quad + 7040x^2 + 6138x + 22990 + (9504x^5 + 17061x^4 \\ & \quad \left. + 2200x^2 + 10494x + 17666)\sqrt{x^4+1} \right). \end{aligned}$$

The integrands were generated such as to admit a closed form in terms of a linear combination of three logarithms with logands of total degree 1 with respect to x and y . Typically, two of the logarithms involve constants from a quadratic extension of $\mathbb{Q}$. For example, the integral over the algebraic function above admits the closed form

$$\begin{aligned} & \frac{1}{44}(13 + i\sqrt{359}) \log((9 - i\sqrt{359})x + 11 - 33\sqrt{x^4+1}) \\ & + \frac{1}{44}(13 - i\sqrt{359}) \log((9 + i\sqrt{359})x + 11 - 33\sqrt{x^4+1}) \\ & - \frac{27}{4} \log(5 - 16\sqrt{x^4+1}). \end{aligned}$$

If an integrator did not deliver a logarithmic part in this form, but with higher degree logands, or with arctan expressions, we also accepted these as correct answers. The integrands were chosen at random, but simple rational functions in x produced occasionally by the test case generator were discarded.

We have implemented an integrator for algebraic functions in Mathematica that executes steps 1–4 of Trager’s algorithm (as described in Section 2), followed by an implementation of our procedure for finding the logarithmic part. Singular was used as Gröbner basis engine. Our code for finding the logarithmic part is given in the appendix. We then presented the test case integrands to Maple (Mpl), Mathematica (Mma), Axiom (Ax) and our procedure (P). Each integrator got 180 seconds per integrand to perform the integration. The results are summarized below.

5.1 Integrals involving $\sqrt{x^2+1}$

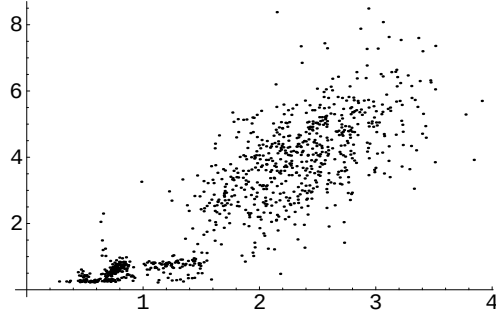
	P	Ax	Mpl	Mma
Success	100.0%	100.0%	100.0%/40.5%	11.6%
Timeout	—	—	—/59.5%	—
Failure	—	—	—/—	88.4%

Maple’s standard integrator does not appear to search for elementary closed forms of algebraic integrals by default. The first percentage shown for Maple refers to the default integration procedure, the second percentage refers to the case where an elementary closed form was explicitly requested.

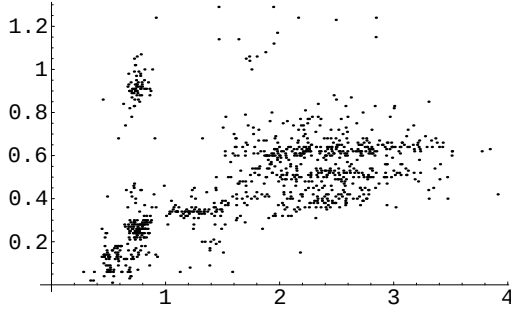
The integrals that Mathematica could not do were returned unevaluated. For the other three integrators a runtime comparison might be interesting. One might think that iterated computation of Gröbner basis in P will make this integrator much slower than his competitors. But this is not the case.

In the following diagram, a dot at the point (x, y) represents an integral that was evaluated by P in x seconds and by Ax in y seconds. We see that the runtime is somewhat correlated, with an advantage for Ax on fast examples and

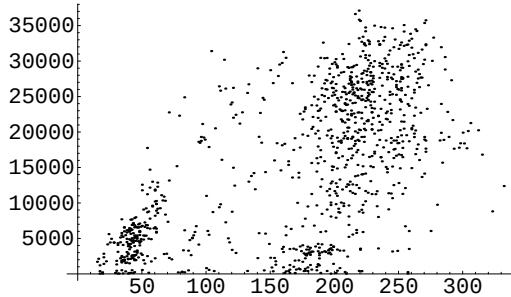
a slight advantage for P on slower examples. The center is at (1.8, 2.8).



The corresponding figure for P vs. Mpl is shown next.



Mpl is much faster than P, but it should be noted that the results produced by Mpl are a whole lot messier than those of P or Ax. In the next figure, a point (x, y) corresponds to an integral for which P produced a logarithmic part of length x and Mpl produced a logarithmic part of length y . (For the “length” of a result, we simply counted the number of characters in Mathematica and Maple syntax, respectively.)



5.2 Integrals involving $\sqrt{x^3 + 1}$

	P	Ax	Mpl	Mma
Success	99.6%	24.3%	.2%/22.0%	2.2%
Timeout	.3%	6.6%	—/78.0%	.1%
Failure	.1%	69.1%	99.8%/—	97.7%

The Failure line covers different behaviors. For Ax, a failure was counted when the integrator aborted with an error (e.g. “implementation incomplete”). Mpl never failed in this way, but always returned non-elementary closed forms involving elliptic functions. Mma returned the integrals unevaluated. Ways how P can fail are discussed in the next section.

The percentage of timeouts and failures increases more rapidly for Ax, Mpl and Mma than for P. Runtime comparisons for the cases in which at least two integrators succeed

lead to less expressive diagrams, which are therefore not shown here.

5.3 Integrals involving $\sqrt{x^4 + 1}$

	P	Ax	Mpl	Mma
Success	99.5%	12.3%	1.8%/24.7%	1.8%
Timeout	.1%	16.4%	—/75.3%	.3%
Failure	.4%	71.3%	98.2%/—	97.9%

Concerning the Failure line, the same remarks apply as for 5.2. The main difference to the previous case is the increase in timeouts for Ax. Also P slows down for this case: the runtime median here (counting successful cases as well as timeouts and failures) is 68.0 sec, compared to 16.0 sec in the previous case.

5.4 Integrals involving $\sqrt[3]{x^2 + 1}$

	P	Ax	Mpl	Mma
Success	67.9%	4.2%	—/18.1%	2.4%
Timeout	29.2%	95.8%	—/81.9%	—
Failure	2.9%	—	100.0%/—	97.6%

Failure here means for both Mpl and Mma that the integral was returned unevaluated.

Although the success rate of P drops considerably in this case, it is still superior to Ax, Mpl, and Mma. The substitution carried out in the first step of the integrator might lead to an algebraic function field with a nontrivial integral basis. The new indeterminates introduced in this case may be responsible for slowing down the Gröbner basis computations and raising the timeout percentage for P compared to the previous test cases.

6. REASONS FOR FAILURE

Our procedure will never detect that an algebraic function cannot be integrated in terms of elementary functions. Even if an elementary integral exists, there is no guarantee that our procedure will find it. It can fail for the following reasons.

6.1 The Principal Generator is Hidden

We search the principal generator p of an ideal among the elements of a Gröbner basis. Though the principal generator can often be found there, this is not always the case.

EXAMPLE 8. In Example 4 we found that the Gröbner basis of $\mathfrak{a} := \langle 9t^2 - 2, m, v, u - tDv \rangle$ is

$$\{9t^2 - 2, y - 3t, x + 1\},$$

and since

$$\langle 9t^2 - 2, m, v, u - tDv \rangle \neq \langle 9t^2 - 2, m, y - 3t \rangle \quad \text{and}$$

$$\langle 9t^2 - 2, m, v, u - tDv \rangle \neq \langle 9t^2 - 2, m, x + 1 \rangle,$$

we concluded that there is no $p \in \mathfrak{a}$ such that $\mathfrak{a} = \langle 9t^2 - 2, m, p \rangle$. However, $p = x + 3ty - 1 \in \mathfrak{a}$ and we do have $\mathfrak{a} = \langle 9t^2 - 2, m, p \rangle$.

It does not harm if we overlook a principal generator in some ideal $\mathfrak{a}$ if we find a generator in one of its powers $\mathfrak{a}^n$. In the example above, the principal generator is found in the next step, as a member of the Gröbner basis for

$$\langle 9t^2 - 2, m \rangle + \langle v, u - tDv \rangle^2$$

(cf. Ex. 4). In our experiments, we have never observed a failure because the principal generator could not be found in any of the powers of an ideal.

6.2 The Power Bound is Exceeded

An obvious source of failures is that the fixed bound on the powers of $\langle v, u - tDv \rangle$ that are inspected is exceeded without that a principal generator was found. Of course, there could still be some principal ideal beyond the bound. In our experiments, we have used the bound 12 but this bound was never reached.

6.3 The Ideal is not Radical

For rational function integrands $u/v \in k(x)$, Czichowski has shown that the ideal $\langle v, u - tDv \rangle$ is always radical, in particular the polynomial in its Gröbner basis which is free of x will always be square free. For algebraic function integrands, this need no longer be true. Branch places over roots of the denominator may cause multiple factors.

In many of these cases, the correct result is obtained when the contributed logarithms are multiplied with the multiplicity of the corresponding factors of the univariate polynomial in t .

EXAMPLE 9. For $u = 1 + y$ and $v = (x^2 + 1)(x + 1)$ with $m = y^2 - (x^2 + 1)$ we find

$$\begin{aligned} & \{(4t^2 + 4t - 1)(8t^2 - 4t + 1)^2, \\ & -2560t^5 - 640t^4 + 1776t^3 - 1136t^2 + 306t + 9y - 41, \\ & 1792t^5 + 576t^4 - 1184t^3 + 744t^2 - 160t + 9x + 19\} \end{aligned}$$

as the Gröbner basis of $\langle m, v, u - tDv \rangle \leq k[x, y, t]$.

For $q = 4t^2 + 4t - 1$ we find

$$\langle q, m \rangle + \langle v, u - tDv \rangle^2 = \langle q, m, 1 - x - (1 + 2t)y \rangle.$$

For $q = 8t^2 - 4t + 1$ we find

$$\langle q, m \rangle + \langle v, u - tDv \rangle^2 = \langle q, m, 4t - 1 + x \rangle.$$

As $8t^2 - 4t + 1$ is a double factor, we count the contribution from the latter ideal twice. Indeed,

$$\begin{aligned} & \int \frac{x + \sqrt{x^2 + 1}}{(x^2 + 1)(x + 1)} dx = 2 \sum_{\gamma: 8\gamma^2 - 4\gamma + 1 = 0} \frac{\gamma}{2} \log(4\gamma - 1 + x) \\ & + \sum_{\gamma: 4\gamma^2 + 4\gamma - 1 = 0} \frac{\gamma}{2} \log(1 - x - (1 + 2\gamma)\sqrt{x^2 + 1}). \end{aligned}$$

Handling multiple factors this way might, however, not be correct. All the failures reported in Section 5 for our integrator are of this kind. The logarithmic expression int returned by the integrator for an integrand u/v may then be viewed as a partial closed form, leaving $\frac{u}{v} - D(\text{int})$ as unintegrated remainder. It usually pays off to apply the integrator to this remainder once again, for it may well succeed in integrating it in a second attempt.

EXAMPLE 10. For $u = 2x^3 + 6x^2 - 7x - 7 - (x - 1)(3x + 1)y$ and $v = (x^2 - 1)x(x^2 - x - 1)$ with $m = y^2 - (x + 1)$ we find that the Gröbner basis of $\langle m, v, u - tDv \rangle$ contains the polynomial $(t - 3)(t - 2)^2t(t + 6)(t + 8)$. We have

$$\begin{aligned} \langle t - 3, m, v, u - tDv \rangle &= \langle t - 3, m, x - 1 \rangle, \\ \langle t - 2, m, v, u - tDv \rangle &= \langle t - 2, m, 1 + x + xy \rangle, \\ \langle t + 6, m, v, u - tDv \rangle &= \langle t + 6, m, y - 1 \rangle, \\ \langle t + 8, m, v, u - tDv \rangle &= \langle t + 8, m, y + 1 \rangle. \end{aligned}$$

However,

$$\begin{aligned} & \frac{u}{v} - D\left(3\log(x - 1) - 6\log(y - 1) \right. \\ & \quad \left. - 8\log(1 + y) + 4\log(1 + x + xy)\right) \\ &= -\frac{2x^2 + x - 1 - (x + 2)y}{(x + 1)(x^2 - x - 1)} \neq 0. \end{aligned}$$

Applying the procedure to this nonzero remainder gives the result $-2\log(x + y)$. Indeed,

$$\begin{aligned} & \int \frac{2x^3 + 6x^2 - 7x - 7 - (x - 1)(3x + 1)\sqrt{x + 1}}{(x^2 - 1)x(x^2 - x - 1)} dx \\ &= 3\log(x - 1) - 6\log(\sqrt{x + 1} - 1) - 8\log(1 + \sqrt{x + 1}) \\ & \quad + 4\log(1 + x + x\sqrt{x + 1}) - 2\log(x + \sqrt{x + 1}). \end{aligned}$$

In the examples in Section 5.2 and 5.3, when the integrator is applied twice, there are no failures any more. In the examples in Section 5.4, when the integrator is applied twice the failure rate drops from 2.9% to .1%. When it is applied once more there are no failures. In 26 of the 29 observed failures in 5.4, the “unintegrable” remainder was just a rational function in x , in 22 of these cases it was just a constant multiple of $1/x$.

7. CONCLUSION

We have described a procedure for finding the logarithmic part of an integral over an algebraic function. Our procedure is simple and efficient, and, although there is no guarantee, it finds the correct results in a great many cases. It has come off well in a comparison with the built-in integrators of Axiom, Maple, and Mathematica.

We do not see how our procedure could be turned into a complete algorithm. In particular, we have no convincing argument that would justify our observation that principal generators often show up in the Gröbner bases we compute. Obviously, any step towards an explanation of this phenomenon would also be a step towards a new complete algorithm, and therefore be highly interesting.

8. REFERENCES

- [1] Manuel Bronstein. On the integration of elementary functions. *Journal of Symbolic Computation*, 9:117–173, 1990.
- [2] Manuel Bronstein. The lazy Hermite reduction. Technical Report 3562, INRIA, 1998.
- [3] Manuel Bronstein. Symbolic integration tutorial. ISSAC’98, 1998.
- [4] Manuel Bronstein. Poorman’s integrator. Maple package, available at <http://www-sop.inria.fr/cafe/Manuel.Bronstein/pmint/>, 2005.
- [5] Manuel Bronstein. *Symbolic Integration I*, volume 1 of *Algorithms and Computation in Mathematics*. Springer, 2nd edition, 2005.
- [6] Manuel Bronstein. Structure theorems for parallel integration. *Journal of Symbolic Computation*, 42(7):757–769, 2007.
- [7] Günter Czichowski. A note on Gröbner bases and integration of rational functions. *Journal of Symbolic Computation*, 20(2):163–167, 1995.

- [8] James H. Davenport. *On the Integration of Algebraic Functions*. Lecture Notes in Computer Science 102. Springer, 1981.
- [9] Manuel Kauers and Viktor Levandovskyy. An interface between Singular and Mathematica. Technical Report 2006-29, SFB F013, Johannes Kepler Universität, 2006.
- [10] A.C. Norman and J.H. Davenport. Symbolic integration – the dust settles? In *LNCS 72*, pages 398–407. Springer, 1979. Proceedings of ISSAC’79.
- [11] Robert H. Risch. The problem of integration in finite terms. *Transactions of the American Mathematical Society*, 139:167–189, 1969.
- [12] Robert H. Risch. The solution of the problem of integration in finite terms. *Bulletin of the American Mathematical Society*, 79:605–608, 1970.
- [13] Barry M. Trager. *On the Integration of Algebraic Functions*. PhD thesis, MIT, 1984.
- [14] Mark van Hoeij. An algorithm for computing an integral basis in an algebraic function field. *Journal of Symbolic Computation*, 18(4):353–363, 1994.

Appendix: Mathematica Code

The following ten lines form a Mathematica 5.2 implementation of the procedure described in this paper. Gröbner basis computations are done with Singular, accessed via an interface package [9]. The function to be integrated is num/den , where den is a squarefree polynomial in the integration variable. The list v contains the formal variables (typically, x and y , but perhaps additional indeterminates for encoding a nontrivial integral basis). The first element of v is taken as the integration variable. The list u is an ideal basis for the relations among the variables in v (typically, the minimal polynomial of y). An electronic version of this code can be found on the author’s homepage.

```
LogarithmicPart[num_, den_, u_List, v_List] := Module[{G, t, factors, f, r, i, F, p},
  G = SingularGroebner[Join[{den, num - t D[den, First[v]]}, u], v, {t}, MonomialOrder->"dp"];
  factors = DeleteCases[Rest[FactorList[First[G]]], {t, _}];
  F[p_, g_] := (t/#2*Log[#1])&@@PrincipalPower[g, Append[u, p], v, {t}];
  Plus@@Apply[r[f[#1], f[#2 F[#1, G]]&, factors, {1}] /. {t -> #, f -> Function, r -> RootSum}];

PrincipalDivisor[gb_, u_, v_] := Module[{G = SingularGroebner[Join[gb, u], v, MonomialOrder->"dp"]},
  First[Append[Select[G, (SingularGroebner[Append[u, #], v, MonomialOrder->"dp"]===G)&, 1], 1]]];

PrincipalPower[gb_, u_, v_List, bound_Integer:12] := Module[{id = gb, p, n = 1},
  While[ n < bound && (p = PrincipalDivisor[id, u, v])===1, n++];
  id = SingularGroebner[Join[SingularTimes[id, gb, v], u], v, MonomialOrder->"dp"]; Return[{p, n}]];
```

Example.

```
In[1]:= LogarithmicPart[y,x(x^8+1),{y^2-(x^8+1)},{x,y}]
```

```
Out[1]= 
$$\frac{\text{Log}[-1 + y]}{8} - \frac{\text{Log}[1 + y]}{8}$$

```


Expressing a Fraction of Two Determinants as a Determinant*

Erich Kaltofen

Department of Mathematics
North Carolina State University
Raleigh, North Carolina 27695-8205, USA
kaltofen@math.ncsu.edu
http://www.kaltofen.us

Pascal Koiran

Laboratoire de l'Informatique du Parallélisme (LIP)[†]
École Normale Supérieure de Lyon
Université de Lyon, France
Pascal.Koiran@ens-lyon.fr
http://perso.ens-lyon.fr/pascal.koiran/

ABSTRACT

Suppose the polynomials f and g in $K[x_1, \dots, x_r]$ over the field K are determinants of non-singular $m \times m$ and $n \times n$ matrices, respectively, whose entries are in $K \cup \{x_1, \dots, x_r\}$. Furthermore, suppose $h = f/g$ is a polynomial in $K[x_1, \dots, x_r]$. We construct an $s \times s$ matrix C whose entries are in $K \cup \{x_1, \dots, x_r\}$, such that $h = \det(C)$ and $s = \gamma(m+n)^6$, where $\gamma = O(1)$ if K is an infinite field or if for the finite field $K = \mathbb{F}_q$ with q elements we have $m = O(q)$, and where $\gamma = (\log_q m)^{1+o(1)}$ if $q = o(m)$. Our construction utilizes the notion of skew circuits by Toda and weakly-skew circuits by Malod and Portier. Our problem was motivated by resultant formulas derived from Chow forms.

Additionally, we show that divisions can be removed from formulas that compute polynomials in the input variables over a sufficiently large field within polynomial formula size growth.

Categories and Subject Descriptors: I.1.1 [Symbolic and Algebraic Manipulation]: Expressions and Their Representation; F.1.1 [Theory of Computation]: Models of Computation

General Terms: algorithms, theory

Keywords: Algebraic complexity theory, formula complexity, Strassen's removal of divisions, Toda's skew circuits, Valiant's universality of determinants

1. INTRODUCTION

1.1 Motivation

The problem that we investigate was motivated by the question of resultant formulas without division. Originally,

*This material is based on work supported in part by the National Science Foundation under Grant CCF-0514585 (Kaltofen).

[†]Unité mixte de recherche (UMR) 5668 ENS Lyon, CNRS, Université Claude Bernard Lyon 1 (UCBL), INRIA.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

the resultant of a set of t homogeneous polynomial equations $f_1 = \dots = f_t = 0$ in t variables has been expressed as a GCD of determinants, whose matrices have the coefficients of the polynomials as entries. Macaulay [13] gave a formula of a quotient of two such determinants. In special cases, one can remove the division and construct a single determinant that is the resultant [10, 6, 11]. Those constructions use properties of exact sequences of exterior algebras. However, an algebraic complexity theoretic approach can remove the division in the general case in an entirely different manner. Take Macaulay's formula $\text{resultant}(f_1, \dots, f_t) = \det(A)/\det(B)$, where the larger matrix A has dimensions $m \times m$. One converts the determinants to straight-line programs, removes the division by Strassen's [18] method, or computes their GCD [8], parallelizes the straight-line program to $O((\log m)^2)$ depth [20, 15], converts the resulting division-free straight-line program to a formula of quasi-polynomial size $m^{O(\log m)}$, and finally writes the resultant formula as the projection of a determinant of a matrix C of dimensions $k \times k$ where $k = m^{O(\log m)}$ [21]. Note that C with $\det(C) = \det(A)/\det(B)$ has as entries the coefficients of the original polynomials or constants. Here we shall show that there is a matrix $\tilde{C}$ of dimensions $O(m^6)$ whose entries are the coefficients of the original polynomials or constants with $\det(\tilde{C}) = \det(A)/\det(B)$.

1.2 Results and Approach Used

Our main result is the following theorem.

Theorem 1 *Let $f, g, h \in K[x_1, \dots, x_r] \setminus \{0\}$, where K is a field, such that $f/g = h$ and f is a projection à la Valiant [21] of an $m \times m$ determinant and g is a projection of an $n \times n$ determinant ($n \leq m$ or $n > m$), meaning that there are matrices $A \in K[x_1, \dots, x_r]^{m \times m}$ and $B \in K[x_1, \dots, x_r]^{n \times n}$, whose entries are in $K \cup \{x_1, \dots, x_r\}$ with $f = \det(A)$ and $g = \det(B)$. Then there exists an $s \times s$ matrix C whose entries are in $K \cup \{x_1, \dots, x_r\}$ such that $h = f/g = \det(C)$ and s is polynomial in $m+n$, that is, the exact quotient of f and g is a projection of a determinant of polynomial dimension. More precisely,*

- (i) *if K is infinite or if $m = O(|K|)$ we can take $s = O((m+n)^6)$;*
- (ii) *if K is a small finite field, we can take $s = O((m+n)^6 \cdot M(\log_{|K|} m))$, where $M(l) = l \cdot (\log l) \cdot (\log \log l)$.*

We prove our result via the notion of *weakly-skew* division-free arithmetic circuits by Malod and Portier [14]. We consider division-free arithmetic circuits (straight-line programs), which are directed acyclic graphs (DAGs) whose nodes have fan-in at most two and which perform addition, subtraction and multiplication. The operands are the values in previous nodes, constant scalars or input variables. The values of designated output nodes are multivariate polynomials in the input variables. The size of the graph is the number of arithmetic operations (sequential complexity) performed. This definition of size is consistent with [21], for instance. Malod and Portier [14] work with a slightly different definition of size, in which input nodes (variables or constants) are counted along with arithmetic nodes. To avoid any confusion, we will call this second notion *fat size*. The fat size of a circuit is therefore equal to the sum of its size and of the number of input nodes. It is bounded by 3 times the size since the number of input nodes is equal to at most twice the number of arithmetic nodes.

Toda [19] introduces *skew* division-free arithmetic circuits, which have the property that at least one of the two operands in each multiplication node is either a scalar constant or an input variable. Toda proves that the determinant polynomial of an $m \times m$ matrix can be computed by a skew circuit of size $O(m^{20})$. In weakly-skew circuits at least one of the two operands to any multiplication node α must be computed by a separate subcircuit C_α . By separate we mean that the edge connecting C_α to α is the only edge between C_α and the remainder of the circuit. Figure 1 shows an example of a weakly-skew circuit. The separate subcircuits for the operands of multiplication nodes are marked by dashed boxes.

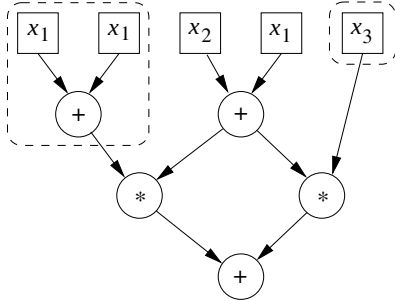


Figure 1: A weakly-skew circuit

Valiant [21] proves that every formula of size s is the projection of an $(s+2) \times (s+2)$ determinant. In formulas, both operands to all nodes are computed by separate formulas. Valiant's proof can be generalized to show that every polynomial that is computed by a weakly-skew circuit of fat size s is the projection of an $(s+1) \times (s+1)$ determinant [14, Lemma 6]. In that paper, the nonconstant entries of the resulting determinant are negated input variables (i.e., they are of the form $-x_i$ where x_i is an input variable). We explain how to get rid of these negations in Section 2.

Furthermore, the division-free parallel circuits by Berkowitz [1] for the characteristic polynomial of an $m \times m$ matrix can be converted to weakly-skew circuits of size $O(m^5)$ [14, Proposition 5]. In Section 3 we apply Strassen's [18] technique for elimination of divisions directly to the characteristic polynomials derived from A and B in Theorem 1

above, thus obtaining a division-free weakly-skew circuit for $h = \det(A)/\det(B)$. This already establishes (using the quantities of Theorem 1) the following result:

Lemma 1 *The exact quotient $h = f/g$ can be computed by weakly-skew circuit of size s .*

To complete the proof of Theorem 1 we note that the resulting circuit is the projection of a determinant. Note that the technique in [2], which is not directly applicable, assumes that the diagonals of A and B hold a single separate variable, implying also $n < m$.

Along the way we show in Section 2 that weakly-skew circuits can be simulated by skew circuits with an increase in size by a constant factor only; and that Toda's $O(m^{20})$ bound can be reduced to $O(m^5)$.

Since weakly-skew circuits are projections of determinants, we have the following corollary to Lemma 1 and our transformation results to skew circuits.

Corollary 1 *Let $f, g, h \in K[x_1, \dots, x_r] \setminus \{0\}$, where K is a field with $\deg(g) = O(|K|)$ such that $f/g = h$ and f and g are computed by a weakly-skew circuit of size s with inputs $x_1, \dots, x_r$. Then h can be computed by a skew circuit of size $O(s^6)$.*

Suppose in the above corollary that f and g are computed by division-free formulas of size $\leq s$. Then $h = f/g$ is a projection of a determinant of a matrix of dimension $O(s^6)$. In Section 4 we show that there exists a division-free formula of size (s^5) that computes $h = f/g$. As a consequence of the latter result, we can show that a polynomial in $K[x_1, \dots, x_r]$ that is computed by a formula of size s with additions, subtractions, multiplications and divisions, where K is a sufficiently large field and $x_1, \dots, x_r$ are input variables, can be computed by a division-free formula of size $s^{O(1)}$ (see Theorem 3 on page).

2. FROM WEAKLY-SKEW TO SKEW CIRCUITS

In this section we show that weakly-skew circuits can be efficiently simulated by skew circuits. This fact will be used in the proof of Theorem 1, part (ii) given in Section 3.2.

Two algorithms for converting weakly-skew circuits into skew circuits are already described by Malod and Portier [14]. The polynomial computed by a weakly-skew circuit of fat size m can be converted into a determinant of size $m+1$ by [14, Lemma 6]. One can then apply Toda's algorithm, which evaluates a determinant of an $m \times m$ matrix by a skew circuit of size $O(m^{20})$. It is observed in [14, Section 5.2] that the role of the determinant in this first algorithm can be played by the polynomial $F_m = \text{Trace}(X^m)$. The method described below is even simpler, and more efficient. As a byproduct, we obtain an improvement to $O(m^5)$ of Toda's original $O(m^{20})$ bound.

Remark 1 In the determinant constructed in [14, Lemma 6], all input variables are negated. This is not a problem for converting a weakly-skew circuit into a skew circuit with the algorithm described in the paragraph above. However, the occurrence of negated input variables makes their construction unsuitable for the proof of Theorem 1. To circumvent this difficulty, one can modify slightly Malod and Portier's

condition to obtain a matrix of size $m + 1$ without negated variables. This can be done in two steps:

1. In their proof of Lemma 6 Malod and Portier first construct a matrix B of size m such that $\det B = -f$, where f is the polynomial computed by a weakly-skew circuit of size m . In this matrix all variables are negated (and all diagonal entries are equal to 1 except the first, which is 0). Now let $C = -B$: we have $\det C = \pm f$, and there are no negated variables in C .
2. If m is odd we have $\det C = -f$. As Malod and Portier, we can add one last row and one last column full of 0's (except for an entry equal to -1 in the bottom right corner) to obtain a matrix of size $m+1$ whose determinant is equal to f .

We shall work with acyclic edge-weighted directed graphs. We recall that the weight of a path in such a graph is defined as the product of the weights of the edges appearing in the path. If s and t are two vertices of G , the weight of (s, t) in G is defined as the sum of the weights of all paths from s to t .

Lemma 2 *Let W be a weakly-skew circuit of fat size m . There exists an acyclic directed graph G , with two distinguished vertices s and t such that:*

- (i) *The weight of (s, t) in G is the polynomial computed by W , and G is of size at most $m + 1$.*
- (ii) *Every vertex in G other than s has either a single incoming edge, of weight equal to an input of W or to the constant 2, or two incoming edges, each of weight 1.*

This is essentially Lemma 5 of Malod and Portier [14]. In that lemma the authors prove the existence of a graph G satisfying (i). An inspection of their proof shows that the graph that they construct also satisfies the second property.

Proposition 1 *Let W be a weakly-skew circuit of fat size m . There exists a skew circuit W' which is equivalent to W (i.e., computes the same polynomial), has the same number of input nodes as W , and has at most m arithmetic nodes (W' is therefore of fat size at most $2m$).*

The algorithmic idea behind this result is quite simple: for each vertex v in the graph G of Lemma 2 we compute the weight $\omega(v)$ of the pair (s, v) . If v has a single incoming edge of weight x connecting s to v then of course $\omega(v) = x$. If v has a single incoming edge of weight x connecting a vertex $v' \neq s$ to v , we can apply the formula $\omega(v) = x \times \omega(v')$ if x is an input of W . If x is the constant 2, we apply the formula $\omega(v) = \omega(v') + \omega(v')$ (one could of course use the formula $\omega(v) = 2 \times \omega(v')$ instead, at the cost of introducing one additional constant input in W'). Finally, if v has two incoming edges of weight 1, connecting the vertices v_1 and v_2 to v , we have $\omega(v) = \omega(v_1) + \omega(v_2)$. The resulting circuit satisfies the requirements of Proposition 1.

Corollary 2 *The determinant of a $m \times m$ matrix can be computed by a skew circuit of size $O(m^5)$.*

This is a significant improvement over the $O(m^{20})$ bound given by Toda [19]. Corollary 2 is an immediate consequence of Proposition 1 since, as pointed out in Section 1, the determinant of a $m \times m$ matrix can be computed by a weakly-skew circuit of size $O(m^5)$.

Skew circuits will be useful in Section 3.2 due to the following proposition and the subsequent remarks.

Proposition 2 *Let K be a field and let $L = K(\theta)$ be an algebraic extension of K of degree $d = [L : K]$. Let $f \in L[x_1, \dots, x_r]$ be a polynomial computed by a skew circuit W_L of size m , with input nodes labeled by variables from $\{x_1, \dots, x_r\}$ or constants from L . Let us expand f according to the powers of θ : one can write $f = \sum_{j=0}^{d-1} \theta^j f_j$, where $f_j \in K[x_1, \dots, x_r]$.*

There exists a skew circuit W_K of size $O(d^2 m)$ with at most d output nodes which computes simultaneously all the polynomials f_j . Moreover, W_K uses only constants from K .

PROOF. Let α be a node of W_L computing a polynomial $f_\alpha \in L[x_1, \dots, x_r]$. We use a standard technique: in W_K , f_α will be represented by d nodes computing polynomials $f_{0,\alpha}, \dots, f_{d-1,\alpha} \in K[x_1, \dots, x_r]$ such that $f_\alpha = \sum_{j=0}^{d-1} \theta^j f_{j,\alpha}$. If α is an input node labeled by some variable x_i we can take $(f_{0,\alpha}, \dots, f_{d-1,\alpha}) = (x_i, 0, \dots, 0)$; if α is labeled by the constant $\sum_{j=0}^{d-1} a_j \theta^j$ we can take $(f_{0,\alpha}, \dots, f_{d-1,\alpha}) = (a_0, a_1, \dots, a_{d-1})$. Assume now that α is an addition node with inputs coming from nodes β and γ . In this case we simply perform componentwise additions since $f_{j,\alpha} = f_{j,\beta} + f_{j,\gamma}$.

Finally, the case where α is a multiplication node can be split in two subcases since W_L is skew: multiplication by a variable x_i , or multiplication by a constant from L . In the first subcase, assume that α multiplies the output of node β by x_i . We have $f_{j,\alpha} = x_i f_{j,\beta}$. Observe that the d resulting multiplications are skew. In the second subcase, assume that α multiplies the output of node β by a constant $\eta \in L$. Multiplication by a constant is a K -linear operation. The tuple $(f_{0,\alpha}, \dots, f_{d-1,\alpha})$ can therefore be obtained from $(f_{0,\beta}, \dots, f_{d-1,\beta})$ by multiplication by an appropriate $d \times d$ matrix A_η with entries in K . The corresponding matrix-vector product can be computed in $O(d^2)$ arithmetic operations, and once again the resulting multiplications are all skew. $\square$

Remark 2 We will apply this result in Section 3.2 in a situation where we know that the output of W_L lies in fact in $K[x_1, \dots, x_r]$. In this case, the subcircuit associated to the first output node of W_K computes the same polynomial as W_L .

Remark 3 Proposition 2 also applies to weakly-skew rather than skew circuits. First one converts the weakly-skew circuit into a skew circuit using Proposition 1. One can then apply Proposition 2 to the skew circuit.

Remark 4 The size of W_K in Proposition 2 can be reduced to $O(m d (\log d) \log \log d)$ by fast polynomial multiplication algorithms [3] and fast division with remainder algorithms [7, Section 9.1].

3. ELIMINATION OF DIVISIONS

In this section we provide a proof of Lemma 1. As explained before, Theorem 1 then follows by applying [14, Lemma 6] and Remark 1 to the resulting circuit.

3.1 Large Coefficient Fields

Our symbolic determinants live in a multivariate polynomial domain $\mathbb{K}[x_1, \dots, x_r]$, where $\mathbb{K}$ is a sufficiently large field. Consider we have non-singular matrices

$$A(x_1, \dots, x_r) \in \mathbb{K}[x_1, \dots, x_r]^{m \times m}$$

and

$$B(x_1, \dots, x_r) \in \mathbb{K}[x_1, \dots, x_r]^{n \times n},$$

whose entries are either variables or constants, i.e.,

$$\begin{aligned} \forall i, j, k, l \text{ with } 1 \leq i, j \leq m, 1 \leq k, l \leq n: \\ (A)_{i,j}, (B)_{k,l} \in \mathbb{K} \cup \{x_1, \dots, x_r\}. \end{aligned}$$

Here $(M)_{i,j}$ denotes the element in row i and column j in the matrix M .

We suppose now that $\det(A)/\det(B) \in \mathbb{K}[x_1, \dots, x_r]$, that is the polynomial division by $\det(B)$ is exact. We construct a division-free weakly-skew arithmetic circuit W of size $O((m+n)^6)$, i.e., *polynomial* in the dimensions of A and B , that computes the polynomial $\det(A)/\det(B)$.

The construction follows Strassen's [18], using Berkowitz's [1][‡] and Chistov's [4] weakly-skew arithmetic circuits for the characteristic polynomial. Let $u_1, \dots, u_r \in \mathbb{K}$ be such that both $U_A = A(u_1, \dots, u_r)$ and $U_B = B(u_1, \dots, u_r)$ are non-singular. Such values always exist if $|\mathbb{K}| > m \geq \deg(\det(A))$ [17, 22].

Now consider

$$\left. \begin{aligned} & \det(U_B - \lambda(U_B - B)) \\ &= \det(U_B) \cdot \det(I_n - \underbrace{U_B^{-1} \lambda(U_B - B)}_{\lambda \bar{B}}) \\ &\in \mathbb{K}[x_1, \dots, x_r][\lambda], \end{aligned} \right\} \quad (1)$$

where I_n denotes an n -dimensional identity matrix. Now the coefficient of λ^i in (1) is the homogeneous part, $\text{hom}_T(\det(B), i)$, of total degree i of $\det(B)$ represented in the term basis

$$T = \{(u_1 - x_1)^{d_1} \cdots (u_r - x_r)^{d_r} \mid d_j \geq 0\}, \quad (2)$$

namely

$$\det(B) = \sum_{i=0}^n \underbrace{\sum_{d_1+\dots+d_r=i} c_{d_1,\dots,d_r} (u_1 - x_1)^{d_1} \cdots (u_r - x_r)^{d_r}}_{\text{hom}_T(\det(B), i)}$$

where $c_{d_1,\dots,d_r} \in \mathbb{K}$. Note that evaluating (1) at $\lambda = 1$ gives $\det(B)$. We compute $1/\det(U_B - \lambda(U_B - B))$ as a truncated power series in $\mathbb{K}(u_1 - x_1, \dots, u_r - x_r)[[\lambda]]$. Because the constant coefficient of (1) as a polynomial in λ is in $\mathbb{K}$, the coefficients of λ^i in the power series for the reciprocal are homogeneous polynomials of degree i in the basis (2). We present a weakly-skew circuit for the coefficients of λ^i .

Let $M_{1\dots l, 1\dots l}$ denote the top left $l \times l$ principal submatrix

[‡]Sasaki and Murao [16] compute the characteristic polynomial of an $n \times n$ matrix with entries in a commutative ring in $n^{\omega+1+o(1)}$ ring operations, and Berkowitz [1, Section 4] conjectures that there exists a division-free arithmetic circuits of size $O(n^\omega)$ for the characteristic polynomial, given that $n \times n$ matrices can be multiplied with $O(n^\omega)$ operations. We can set $\omega = 2.375477$ [5]. The Sasaki&Murao/Berkowitz problem remains open. The best division-free complexity for the characteristic polynomial is $O(n^{2.697263})$ [9].

of a matrix M . Chistov's algorithm is based on the identities

$$\begin{aligned} & \frac{1}{\det(I_n - \lambda \bar{B})} \\ &= \prod_{j=1}^n \frac{\det(I_{j-1} - \lambda \bar{B}_{1\dots j-1, 1\dots j-1})}{\det(I_j - \lambda \bar{B}_{1\dots j, 1\dots j})} \\ &= \prod_{l=1}^n ((I_l - \lambda \bar{B}_{1\dots l, 1\dots l})^{-1})_{l,l} \\ &= \prod_{l=1}^n \left(\sum_{k=0}^{\infty} \lambda^k \bar{B}_{1\dots l, 1\dots l}^k \right)_{l,l} \\ &\equiv \prod_{l=1}^n \left(\sum_{k=0}^m (\bar{B}_{1\dots l, 1\dots l}^k)_{l,l} \lambda^k \right) \pmod{\lambda^{m+1}} \quad (3) \\ &\equiv 1 + q_1 \lambda + \cdots + q_m \lambda^m \pmod{\lambda^{m+1}}. \quad (4) \end{aligned}$$

We use weakly-skew circuits to compute the coefficients q_k . Each $(\bar{B}_{1\dots l, 1\dots l}^k)_{l,l}$ in (3) is computed as

$$\left(\bar{B}_{1\dots l, 1\dots l} \cdot (\bar{B}_{1\dots l, 1\dots l} \cdot (\cdots (\bar{B}_{1\dots l, 1\dots l} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}) \cdots) \right)_l$$

by weakly-skew circuits of size $O(mn^2)$. The weakly-skew circuits for carrying out the modular product (4) require no more than $m+1$ copies of the circuits for each coefficient $(\bar{B}_{1\dots l, 1\dots l}^k)_{l,l}$. Thus a weakly-skew circuit W_1 of size $O(m^2 n^3)$ computes all q_k in (4).

Malod and Portier [14, Proposition 5] compute the coefficients of

$$1 + p_1 \lambda + \cdots + p_m \lambda^m = \det(I_m - U_A^{-1} \lambda(U_A - A))$$

via a weakly-skew circuit W_2 of size $O(m^5)$ by Berkowitz's algorithm. Alternatively and less efficiently, one could as above compute

$$\begin{aligned} & \frac{1}{\det(I_m - U_A^{-1} \lambda(U_A - A))} \\ &\equiv 1 + \bar{p}_1 \lambda + \cdots + \bar{p}_m \lambda^m \pmod{\lambda^{m+1}}, \end{aligned}$$

and compute the truncated power series of the reciprocal as

$$\begin{aligned} & \frac{1}{1 + \bar{p}_1 \lambda + \cdots + \bar{p}_m \lambda^m} \\ &\equiv \sum_{l=0}^m (-\bar{p}_1 \lambda - \cdots - \bar{p}_m \lambda^m)^l \pmod{\lambda^{m+1}} \quad (5) \\ &\equiv 1 + p_1 \lambda + \cdots + p_m \lambda^m \pmod{\lambda^{m+1}}, \end{aligned}$$

again repeating the circuits which compute the coefficients $\bar{p}_i$ in the truncated polynomial powers.

A weakly-skew circuit then carries out the multiplication

$$\begin{aligned} & \frac{\det(U_A - \lambda(U_A - A))}{\det(U_B - \lambda(U_B - B))} \\ &\equiv \frac{\det(U_A)}{\det(U_B)} (1 + p_1 \lambda + \cdots + p_m \lambda^m) \\ &\quad \cdot (1 + q_1 \lambda + \cdots + q_m \lambda^m) \pmod{\lambda^{m+1}} \quad (6) \\ &\equiv \varrho_0 + \varrho_1 \lambda + \cdots + \varrho_m \lambda^m \pmod{\lambda^{m+1}}. \end{aligned}$$

Here and before, the truncation could be performed at order $\deg(\det(A)/\det(B)) + 1 \leq m+1$. Again, we need no more

than $m + 1$ copies of the circuit W_1 for the coefficients q_k or of W_2 for p_k . Because $h = \det(A)/\det(B)$ has total degree $\leq \det(A) \leq m$ and $\rho_i = \text{hom}_T(h, i)$, we can compute

$$\frac{\det(A)}{\det(B)} = \varrho_0 + \varrho_1 + \dots + \varrho_m$$

by a division-free weakly-skew arithmetic circuit of size

$$O(\min\{m^3 n^3 + m^5, m^2 n^3 + m^6\}).$$

Remark 5 The above techniques also yield a single determinant of a matrix of polynomially-sized dimensions for a fraction of products of determinants $(\prod_i A^{[i]})/(\prod_j B^{[j]}) \in K[x_1, \dots, x_r]$, with and without using block diagonal matrices. Such fractions occur when computing the resultant via Koszul complexes [Ágnes Szántó has pointed this application out to us].

3.2 Small Coefficient Fields

When the coefficient field K has few elements, divisions by zero may occur at all values $u_1, \dots, u_r \in K$ and there is no no-singular U_B in Section 3. We can nonetheless obtain the bound of Theorem 1.(ii) using a field extension.

Indeed, from Section 3 we know that the quotient $h = f/g$ can be computed by a polynomial size weakly-skew circuit W_L with constants from L if L is an extension of K with at least $m + 1$ elements. We can therefore work with an extension of degree $O(\log_{|K|}(m))$. Since h is actually a polynomial with coefficients in K , by applying Proposition 2 and the two subsequent remarks to W_L we obtain a skew circuit W_K which computes h using constants from K only.

4. FORMULAS WITH DIVISIONS

We shall prove the following theorem.

Theorem 2 *Let $f, g, h \in K[x_1, \dots, x_r] \setminus \{0\}$, where K is a field with $|K| > \deg(f) + \deg(g) \deg(h)$, such that $f/g = h$ and f is computed by a division-free formula of size s_f and g is computed by a division-free formula of size s_g with inputs $x_1, \dots, x_r$. Then h can be computed by division-free formula of size $O((s_f + s_g)^5)$.*

PROOF. We use interpolation (cf. [8, Section 5]). Denote by $\delta_f, \delta_g, \delta_h$ the total degrees of f, g and h . Note that $\delta_f \leq s_f$ and $\delta_g \leq s_g$. As in Section 3, let $u_1, \dots, u_r \in K$ be such that $g_0 = g(u_1, \dots, u_r) \neq 0$. Again as in (5) and (6) using the term basis (2), we compute (7) in Figure 2. We compute the polynomial H in (7) of degree in λ of no more than $D = \delta_f + \delta_g \delta_h$ by interpolation at $\lambda = v_0, \dots, v_D \in K$ as an exact polynomial before truncating modulo λ^{δ_h+1} . Note that for all j we have

$$\begin{aligned} \sum_{i=0}^{\delta_f} \text{hom}_T(f, i) v_j^i \\ = f(u_1 - v_j(u_1 - x_1), \dots, u_r - v_j(u_1 - x_r)) \end{aligned}$$

and

$$\begin{aligned} \sum_{i=1}^{\delta_g} \text{hom}_T(g, i) v_j^i \\ = g(u_1 - v_j(u_1 - x_1), \dots, u_r - v_j(u_1 - x_r)) - g_0, \end{aligned} \quad (8)$$

$$\begin{aligned} \sum_{i=0}^{\delta_h} \text{hom}_T(h, i) \lambda^i \\ \equiv \frac{\sum_{i=0}^{\delta_f} \text{hom}_T(f, i) \lambda^i}{\sum_{i=0}^{\delta_g} \text{hom}_T(g, i) \lambda^i} \pmod{\lambda^{\delta_h+1}} \\ \equiv \frac{1}{g_0} \left(\sum_{i=0}^{\delta_f} \text{hom}_T(f, i) \lambda^i \right) \\ \cdot \underbrace{\sum_{l=0}^{\delta_h} \left(-\frac{1}{g_0} \right)^l \left(\sum_{i=1}^{\delta_g} \text{hom}_T(g, i) \lambda^i \right)^l}_{H(x_1, \dots, x_r, \lambda)} \pmod{\lambda^{\delta_h+1}}. \end{aligned} \quad (7)$$

Figure 2: Formula division

and therefore can obtain the values in the inputs $x_1, \dots, x_r$ by formulas. We have formulas for each $H(x_1, \dots, x_r, v_j)$ by repeating the formulas (8) no more than $\delta_h(\delta_h + 1)/2$ many times. Interpolation is a matrix times vector product. The vector under consideration is the vector

$$(H(x_1, \dots, x_r, v_j))_{1 \leq j \leq D+1},$$

and the matrix is the inverse of the Vandermonde matrix $\text{Vand}(v_1, \dots, v_{D+1})$. The product again is done by repeating the formulas for $H(x_1, \dots, x_r, v_j)$ no more than $D + 1$ times. Finally, we add the thus obtained first $\delta_h + 1$ coefficients in λ of H . Note that all divisions are by scalars independent on $x_1, \dots, x_r$. $\square$

Theorem 2 together with the well-known parallel circuits for formula evaluation allows the removal of divisions in formulas altogether. When there are division nodes in formulas with inputs $x_1, \dots, x_r$, it is assumed that all rational functions in $K(x_1, \dots, x_r)$ by which is divided are non-zero. Formulas with a divisions by a generic 0 are naturally excluded. For certain values in K for the inputs $x_1, \dots, x_r$ a zero division can occur.

Theorem 3 *There exists a real constant $\gamma > 0$ with the following properties. Let $h \in K[x_1, \dots, x_r]$ be computed by a formula (with divisions) of size s with inputs $x_1, \dots, x_r$. Assume K is a field with $|K| > s^\gamma$. Then h can be computed by division-free formula of size $O(s^\gamma)$.*

In the following proof γ_1, γ_2 and γ_3 are fixed positive real constants. The proof observes that h is computed by a circuit V_1 of fan-in at most 2 with divisions of depth $\leq \gamma_1 \log(s)$ ([12] and the references there). By computing unreduced numerator and denominator polynomials for each node separately, we have two division-free circuits V_2 and V_3 of depth $\leq \gamma_2 \log(s)$ that compute polynomials f and g such that $f/g = h$. We can convert V_2 and V_3 into division-free formulas of depth $\leq \gamma_2 \log(s)$, hence of size $< 2^{\gamma_2 \log(s)+1} = O(s^{\gamma_3})$, which also bounds the degrees of f and g . Applying Theorem 2 to both formulas yields Theorem 3.

5. REFERENCES

- [1] BERKOWITZ, S. J. On computing the determinant in small parallel time using a small number of processors. *Inform. Process. Letters* 18 (1984), 147–150.
- [2] CANNY, J. Generalized characteristic polynomials. *J. Symbolic Comput.* 9, 3 (1990), 241–250.
- [3] CANTOR, D. G., AND KALTOFEN, E. On fast multiplication of polynomials over arbitrary algebras. *Acta Inform.* 28, 7 (1991), 693–701.
- [4] CHISTOV, A. L. Fast parallel calculation of the rank of matrices over a field of arbitrary characteristic. In *Proc. FCT '85* (Heidelberg, Germany, 1985), vol. 199 of *Lect. Notes Comput. Sci.*, Springer Verlag, pp. 63–69.
- [5] COPPERSMITH, D., AND WINOGRAD, S. Matrix multiplication via arithmetic progressions. *J. Symbolic Comput.* 9, 3 (1990), 251–280. Special issue on complexity theory.
- [6] EISENBUD, D., SCHREYER, F., AND WEYMAN, J. Resultants and Chow forms via exterior syzygies. *J. Amer. Math. Soc.* 16 (2003), 537–579.
- [7] VON ZUR GATHEN, J., AND GERHARD, J. *Modern Computer Algebra*. Cambridge University Press, Cambridge, New York, Melbourne, 1999. Second edition 2003.
- [8] KALTOFEN, E. Greatest common divisors of polynomials given by straight-line programs. *J. ACM* 35, 1 (1988), 231–264.
- [9] KALTOFEN, E., AND VILLARD, G. On the complexity of computing determinants. *Computational Complexity* 13, 3-4 (2004), 91–130.
- [10] KHETAN, A. Determinantal formula for the Chow form of a toric surface. In *Proc. 2002 Internat. Symp. Symbolic Algebraic Comput. (ISSAC'02)* (New York, N. Y., 2002), T. Mora, Ed., ACM Press, pp. 145–150.
- [11] KHETAN, A., SONG, N., AND GOLDMAN, R. Sylvester A-resultants for bivariate polynomials with planar newton polygons. In *ISSAC 2004 Proc. 2004 Internat. Symp. Symbolic Algebraic Comput.* (New York, N. Y., 2004), J. Gutierrez, Ed., ACM Press, pp. 205–212.
- [12] KOSARAJU, S. R., AND DELCHER, A. L. Optimal parallel evaluation of tree-structured computations by raking. In *Proc. AWOOC 88* (Heidelberg, Germany, 1988), vol. 319 of *Lect. Notes Comput. Sci.*, Springer Verlag, pp. 101–110.
- [13] MACAULAY, F. S. *The Algebraic Theory of Modular Systems*. No. 19 in Cambridge Tracts. The University Press, Cambridge, Great Britain, 1916. Reissued in the Cambridge Mathematical Library with an Introduction by Paul Roberts, 1994.
- [14] MALOD, G., AND PORTIER, N. Characterizing Valiant's algebraic complexity classes. *Journal of Complexity* 24, 1 (Feb. 2008), 16–38. URL <http://www.ens-lyon.fr/LIP/Pub/Rapports/RR/RR2005/RR2005-44.pdf>.
- [15] MILLER, G. L., RAMACHANDRAN, V., AND KALTOFEN, E. Efficient parallel evaluation of straight-line code and arithmetic circuits. *SIAM J. Comput.* 17, 4 (1988), 687–695.
- [16] SASAKI, T., AND MURAO, H. Efficient Gaussian elimination method for symbolic determinants and linear systems. *ACM Trans. Math. Software* 8, 3 (1982), 277–289.
- [17] SCHWARTZ, J. T. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM* 27 (1980), 701–717.
- [18] STRASSEN, V. Vermeidung von Divisionen. *J. reine u. angew. Math.* 264 (1973), 182–202. In German.
- [19] TODA, S. Classes of arithmetic circuits capturing the complexity of computing the determinant. *IEICE Transactions on Information and Systems E75-D*, 1 (Jan. 1992), 116–124.
- [20] VALIANT, L., SKYUM, S., BERKOWITZ, S., AND RACKOFF, C. Fast parallel computation of polynomials using few processors. *SIAM J. Comp.* 12, 4 (1983), 641–644.
- [21] VALIANT, L. G. Completeness classes in algebra. In *Proc. 11th Annual ACM Symp. Theory Comput.* (New York, N.Y., 1979), ACM, pp. 249–261.
- [22] ZIPPEL, R. Probabilistic algorithms for sparse polynomials. In *Symbolic and Algebraic Computation* (Heidelberg, Germany, 1979), vol. 72 of *Lect. Notes Comput. Sci.*, Springer Verlag, pp. 216–226. Proc. EUROSAM '79.

Symbolic Optimization of Algebraic Functions

Masaaki Kanno
CREST, Japan Science and
Technology Agency
4-1-8, Honcho,
Kawaguchi-shi, Saitama
332-0012, Japan
M.Kanno.99@cantab.net

Hirokazu Anai
Fujitsu Laboratories Ltd
4-1-1 Kamikodanaka,
Nakahara-ku, Kawasaki
211-8588, Japan
anai@jp.fujitsu.com

Kazuhiro Yokoyama
Rikkyo University
3-34-1 Nishi Ikebukuro,
Toshima-ku, Tokyo
171-8501, Japan
yokoyama@rkmath.rikkyo.ac.jp

Shinji Hara
The University of Tokyo
7-3-1 Hongo,
Bunkyo-ku, Tokyo
113-8656, Japan
shinji_hara@ipc.i.u-
tokyo.ac.jp

ABSTRACT

This paper attempts to establish a new framework of symbolic optimization of algebraic functions that is relevant to possibly a wide variety of practical application areas. The crucial aspects of the framework are (i) the suitable use of algebraic methods coupled with the discovery and exploitation of structural properties of the problem in the conversion process into the framework, and (ii) the feasibility of algebraic methods when performing the optimization. As an example an algebraic approach is developed for the discrete-time polynomial spectral factorization problem that illustrates the significance and relevance of the proposed framework. A numerical example of a particular control problem is also included to demonstrate the development.

Categories and Subject Descriptors

I.1 [Symbolic and Algebraic Manipulation]: Algorithms;
J.2 [Physical Sciences and Engineering]

General Terms

Algorithm, Experimentation

Keywords

Parametric optimization, Gröbner basis, quantifier elimination, polynomial spectral factorization

1. INTRODUCTION

Whilst numerical computation based on floating-point arithmetic is prevalent in the applied science and engineering

fields, algebraic methods and algebraic algorithms have been attracting much attention from those application areas due to computed results of different quality such approaches can potentially offer. The early days saw results of theoretical interest, but even a tiny toy problem could hardly be solved because of lack of effective algorithms and implementation and also because of the limited computation power available at that time. Algorithmic development, coupled with the ever increasing computation capability, made it possible to find solutions for (sometimes more than) toy problems. The achievement further stimulated both computer algebraists and people on the application side. Solution for larger, more realistic problems has been envisaged and formulation of a wider class of practical problems in the algebraic framework pursued.

This paper attempts to establish a new framework of symbolic optimization which has relevance to practical problems. Emphasized in the development is the significance of the appropriate use of algebraic methods and of the discovery of structural properties inherent in original problems in the application areas. More specifically algebraic methods play a vital rôle in converting the original problem into the suggested optimization framework and moreover in solving the resulting optimization problem. As an example we consider a particular control problem which falls into the framework, but the framework in fact encompasses a wide class of problems in signal processing and control. The authors believe that a large number of problems from the application side can be dealt with in a similar fashion and that the framework indeed embraces an extensive area of problems of practical significance. It is thus hoped that this paper will stimulate the research in various fields of application and also the computer algebra community into the search of latent structures and exciting algorithmic improvements.

The rest of the paper is organized as follows. Section 2 is devoted to the formulation of the new framework this paper proposes, and some discussions are given on how to achieve the framework and to solve the formulated optimization problem effectively are given. Section 3 takes up an example of the discrete-time polynomial spectral factorization

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

problem, which can help convert various problems in signal processing and control into the suggested framework and thus indicates the relevance of the framework in the context of the solution of problems arising from application fields. Then a numerical example of a control problem is given in Section 4. Some concluding remarks are made in Section 5.

2. NEW FRAMEWORK OF SYMBOLIC OPTIMIZATION

Two algebraic techniques most commonly relied upon to solve problems from applied fields may be Gröbner bases and quantifier elimination (QE). After the formal establishment of the notion of Gröbner bases, endeavours based on Gröbner basis theory have been made for the solution of various problems. For instance the power of Gröbner basis theory is exerted to observe the structure of the solution set of algebraic equations [15], and also to reduce the number of variables (or elucidate free variables) for parametrization of solutions [27]. The technique of QE, originated by Tarski in 1950s, has a long history of application examples, but more recently benefits from the algorithmic development [8, 9, 10, 32] and easy-to-use implementation [5] and also from the computation capability of modern computers. Typical usages of QE include computation of algebraic expressions for feasible regions of parameters [2, 3, 11, 13, 16] and (possibly non-convex) optimization of a cost function which is rational in parameters under some algebraic constraints on parameters [4, 20, 31]. Possibilities of other methods are explored as well and new application examples have been constantly appearing [7].

However, in order to make such algebraic methods more useful in practice, it is of crucial importance to greatly reduce computation time and moreover to allow treatment of a wider class of problems that are relevant to practical applications. For the reduction of the computation burden, it is never sufficient to solely expect improvements in algorithms. Extensive efforts are also to be made to pursue methods that exploit the structural properties of the problems that are being solved, and furthermore the discovery of better representations of the problems that reveal desirable structures is to be made. More importantly several techniques need to be organically blended so that sundry problems of practical significance may be formulated appropriately in the algebraic framework.

In this paper a new framework of symbolic optimization is proposed that may overcome current limitations and proceed in the direction mentioned above. The purpose of the paper is twofold:

- to show the significance of combining various algebraic techniques in reducing a practical problem to a new framework of symbolic optimization suggested in this paper and further performing optimization; and
- to indicate that the suggested framework has relevance to practical problems by giving a particular example in control.

The suggested formulation is written as

$$\begin{cases} \text{maximize/minimize } \phi(\mathbf{p}) \\ \text{subject to } \mathcal{F}(\mathbf{p}) \end{cases} \quad (1)$$

where $\mathbf{p} = (p_1, p_2, \dots)$ is a vector of real parameters, $\phi(\mathbf{p}) \in L$ with $L/\mathbb{Q}(\mathbf{p})$ being a finite algebraic extension of $\mathbb{Q}(\mathbf{p})$,

and $\mathcal{F}(\mathbf{p})$ is a set of algebraic constraints on $\mathbf{p}$. In this paper we call $\phi(\mathbf{p})$ a real algebraic function for simplicity (and hence the title of the paper). Namely, $\phi(\mathbf{p})$ is a rational function in ζ and $\mathbf{p}$, where ζ is some algebraic number defined as a particular root of a minimal polynomial over $\mathbb{Q}(\mathbf{p})$. By $\mathcal{F}$, the feasible region of parameters are specified algebraically, e.g., $-1 \leq p_1 < 2$ or $p_1^2 + 2p_2^2 \leq 1$. Furthermore, as a special case of (1), we consider the situation where the cost function $\phi(\mathbf{p})$ is the *largest real root* of a polynomial whose coefficients are polynomials/rational functions in $\mathbf{p}$.

The problem formulation (1) may be in fact contained in a general framework stated in [4], but there are two distinctive points in our framework: (i) formulation in this way and (ii) solution of this optimization problem. It can hardly be expected that a realistic optimization problem arising from a practical application is readily given in the form of (1). Indeed it is often the case that a practical problem is tackled by solving a series of equations and thus that the input to an equation depends on the output (i.e., solution) of the preceding equations. With parameters the situation is more problematic. Algebraic methods can assist reformulation of the original optimization problem in the form of (1). We will see as an example that sophisticated Gröbner basis techniques turn a control problem into this form.

It goes without saying that the computation cost is always the issue when an algebraic method is utilized. We further point out the significance of the exploitation of structural properties of the problem and of the pursuit of more desirable parametrization of variables for accomplishing this part, which cannot be overstressed.

Once converted into (1), various optimization algorithms, both numerical and algebraic [23], may be applicable, and extensive efforts have been made for improvement of the efficacy of such algorithms. It is indicated that this optimization can also be performed by means of the QE approach [4, 20, 31]. The optimization problem in (1) is in general non-convex and thus it is not an easy task for a numerical optimization algorithm to find the global optimum. Algebraic methods based on, e.g., QE, have an advantage that they can in principle find the global optimum. This paper shows that a wide class of realistic practical problems can be formulated as in (1) and thus that QE-based optimization has immense significance in the application field.

More specifically, write $\phi(\mathbf{p}) = \bar{\phi}(\zeta, \mathbf{p})$, where ζ is defined as a particular real root of a polynomial over $\mathbb{Q}(\mathbf{p})$ and $\bar{\phi}(\zeta, \mathbf{p})$ is a rational function in ζ and $\mathbf{p}$. That ζ is a particular real root can be stated in an equivalent condition on ζ that there are a particular number of real roots between ζ and $+\infty$. Therefore, ζ can be specified algebraically by way of the defining polynomial and the condition from the Sturm-Habicht sequence [14]; see [1, 20] for more details. The optimization problem is thus stated as a QE problem:

$$\begin{aligned} \exists \zeta \exists \mathbf{p} \ (\ \eta = \bar{\phi}(\zeta, \mathbf{p}) \ \wedge \ \mathcal{F}(\mathbf{p}) \\ \wedge \ [\text{Condition from the Sturm-Habicht sequence}] \) . \end{aligned}$$

After the elimination of variables ζ and $\mathbf{p}$, the condition only on η results, which indicates the range of the values $\bar{\phi}(\zeta, \mathbf{p})$ (equivalently, $\phi(\mathbf{p})$) can take when parameters $\mathbf{p}$ change subject to $\mathcal{F}(\mathbf{p})$. It is immediate to find the minimum/maximum of $\phi(\mathbf{p})$ from the result.

We point out that, in this part, further exploitation of structural properties is also possible for speeding up optimization when $\phi(\mathbf{p})$ is defined as the largest real root of a

polynomial over $\mathbb{Q}(\mathbf{p})$ [18]. This further expands the practical applicability of algebraic approaches.

It is emphasized here that all the points stated above have a significant meaning to practical applications and are indispensable for devising effective algebraic approaches. Indeed control problems considered in [19, 20] are suitably transformed in the form of (1) and also solved via the QE-based optimization approach. In Section 3, we see another example of a control problem that falls into the framework suggested in this section. It is believed that these problems are convincing evidences of the versatility of this framework.

3. DISCRETE-TIME POLYNOMIAL SPECTRAL FACTORIZATION

The problem of polynomial spectral factorization is of crucial importance in signal processing [12] and control [22] for finite dimensional linear systems. The pursuit of its solution approaches thus has a long history [28]. Early results includes approaches using constant matrix factorization or the Newton-Raphson method. A typical approach in recent times may be to reduce the problem to an algebraic Riccati equation and solve it by means of numerical linear algebra. The authors of this paper pointed out an intriguing and useful relationship between the *continuous-time* polynomial spectral factorization and Gröbner basis theory [17] and discussed an algebraic approach to parametric polynomial spectral factorization based on this discovery [1, 20]. The key ingredient in this approach is a quantity called the sum of roots (SoR), and it is further indicated that this quantity also has an essential meaning in control theory [19].

This section develops a similar algebraic approach to the *discrete-time* case. Since modern signal processing and control systems utilize digital computers to fulfil severe requirements for high-level performances and accomplish complicated tasks, analysis and synthesis of discrete-time systems have become of practical significance [26]. It is thus beneficial to develop mathematical tools for discrete-time systems.

In general, given a result for the *continuous-time* case, a conceivable approach is to make use of the bilinear transform (Tustin transform) [26], which is a conformal mapping that transforms the *continuous-time* representation to the *discrete-time* one and vice versa. Indeed this approach works for the polynomial spectral factorization problem, and parametric *discrete-time* polynomial spectral factorization is feasible via the approach to the *continuous-time* counterpart. There is however a drawback in such an approach because it may obscure a fundamental quantity in the discrete-time domain.

The aim of this section is thus to develop a more direct method for parametric *discrete-time* polynomial spectral factorization which preserves a quantity that has an essential meaning in control theory, just like the SoR in the *continuous-time* case. The development turns out to be analogous to the continuous-time case [1, 20] and the approach is fairly effective. Nevertheless it is emphasized that this analogy is achieved by means of a novel parametrization. The development is thus believed to serve as a persuasive evidence for the importance of seeking nice representations.

3.1 Problem Formulation

Consider the polynomial¹ of degree $2n$ in $\mathbb{R}[x]$ of the fol-

¹The polynomial (2) is obviously *not* a polynomial, but it

lowing form:

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 + \frac{a_1}{x} + \cdots + \frac{a_{n-1}}{x^{n-1}} + \frac{a_n}{x^n}, \quad a_n \neq 0, \quad (2)$$

where $a_i \in \mathbb{R}$, $i = 0, 1, \dots, n$. Here, for the brevity of the exposition, we first assume that the coefficients of $f(x)$ are real constants and do not contain parameters. The discussion on the parametric case is deferred until Subsection 3.3. The polynomial $f(x)$ is called *self-reciprocal* since $f(x)$ and its reciprocal polynomial², $f(\frac{1}{x})$, are coincident: $f(x) = f(\frac{1}{x})$. Suppose that $f(x)$ has no roots of unit modulus, i.e., has no roots on the unit circle. Due to the self-reciprocal property, if α is a root of $f(x)$, then so is $\frac{1}{\alpha}$. All the roots are situated symmetrically about the unit circle and there are n roots each inside and outside the unit circle. The task in the discrete-time polynomial spectral factorization problem is to decompose $f(x)$ as a product of two polynomials, a polynomial that captures all the roots inside the unit circle (namely, ‘stable’ roots) and its reciprocal.

Definition 1. The *spectral factorization* of $f(x)$ in (2) is a decomposition of $f(x)$ of the following form:

$$f(x) = g(x) g\left(\frac{1}{x}\right), \quad (3)$$

where

$$g(x) = b_n x^n + b_{n-1} x^{n-1} + \cdots + b_1 x + b_0 \in \mathbb{R}[x], \quad b_n > 0, \quad (4)$$

and $g(x)$ has roots strictly *inside* the unit circle only. The polynomial $g(x)$ is called the *spectral factor* of $f(x)$.

Some investigation on the structural properties of the problem is made. Let α_i , $i = 1, 2, \dots, n$, be the n roots of $f(x)$ inside the unit circle (i.e., $|\alpha_i| < 1$). The remaining n roots located outside of the unit circle can then be written as $\frac{1}{\alpha_i}$, $i = 1, 2, \dots, n$. Using α_i ’s, we can express $f(x)$ and $g(x)$ as

$$\begin{aligned} f(x) &= a_n \prod_{i=1}^n (x - \alpha_i) \left(1 - \frac{1}{\alpha_i x}\right), \\ g(x) &= b_n \prod_{i=1}^n (x - \alpha_i). \end{aligned} \quad (5)$$

Unlike the continuous-time case, the leading coefficient b_n of $g(x)$ is not immediately determined from $f(x)$, but, comparing the leading coefficients of the both sides of (3), we can observe the following relationship:

$$a_n = b_n b_0 = b_n^2 \prod_{i=1}^n (-\alpha_i). \quad (6)$$

Now, just as the SoR in the continuous-time case, let us introduce the quantity called the *product of roots* (PoR):

$$\pi := (-\alpha_1)(-\alpha_2) \cdots (-\alpha_n). \quad (7)$$

The naming may be obvious since π is the product of roots

can be easily converted to a polynomial: $x^n f(x)$. We thus regard (2) as a polynomial to follow the convention in signal processing and control and also to simplify the notation in the development in this paper.

²Again, to be precise, this is *not* a polynomial, but we also consider this one as a polynomial.

of $g(x)$ up to sign. As will be seen below, π is a crucial quantity in carrying out discrete-time polynomial spectral factorization. Since $|\alpha_i| < 1$, and also any non-real root of $f(x)$ (resp., $g(x)$) has its complex conjugate as a root of $f(x)$ (resp., $g(x)$) and their product becomes real, the following fact is immediately derived.

Fact 1. The quantity π is real and its modulus (absolute value) is strictly less than 1.

A naïve approach to polynomial spectral factorization may be to first calculate the roots α_i 's of $f(x)$ inside the unit circle, compute b_n from (6) and then construct $g(x)$ using (5). Our main target is nevertheless parametric $f(x)$ and we pursue an algebraic approach that computes π without explicitly finding α_i 's. Also the approach is expected to reduce the polynomial spectral factorization problem in essence to the computation of π , just as in the continuous-time case where the crucial part of the approach is to obtain the SoR. In preparation for the development some polynomials which has π as one of their roots are introduced.

Definition 2. Let $\mathcal{P} = \{(\epsilon_1, \epsilon_2, \dots, \epsilon_n) \mid \epsilon_i \in \{1, -1\}\}$, and $C(\epsilon_1, \epsilon_2, \dots, \epsilon_n) = (-\alpha_1)^{\epsilon_1} \cdot (-\alpha_2)^{\epsilon_2} \dots (-\alpha_n)^{\epsilon_n}$ for each $(\epsilon_1, \epsilon_2, \dots, \epsilon_n)$ in $\mathcal{P}$. The *characteristic polynomial* $S_f(y)$ of π is defined as

$$S_f(y) = \prod_{(\epsilon_1, \epsilon_2, \dots, \epsilon_n) \in \mathcal{P}} (y - C(\epsilon_1, \epsilon_2, \dots, \epsilon_n)) .$$

Notice that

$$\pi = C(1, 1, \dots, 1) ,$$

and the degree of $S_f(y)$ is 2^n . It is further noted that $S_f(y)$ belongs to $\mathbb{R}[y]$. In the case where $f(x)$ belongs to $\mathbb{Q}[x]$, $S_f(y)$ belongs to $\mathbb{Q}[y]$, as well (while $g(x) \notin \mathbb{Q}[x]$ in general).

The following lemma can be deduced in a manner analogous to the continuous-time case.

LEMMA 1. The PoR π defined in (7) coincides with the real root of $S_f(y)$ with the smallest modulus. Moreover, under the assumption that $f(x)$ does not have roots on the unit circle, π is always a simple root.

In the discrete-time case we are dealing with in this section, we need an extra trick to facilitate solution of the polynomial spectral factorization problem. Using π , we can write the relationship in (6) as

$$b_n^2 = \frac{a_n}{\pi} \quad \left(\pi = \frac{a_n}{b_n^2} \right) . \quad (8)$$

Then,

$$\tilde{S}_f(\tilde{y}) := \tilde{y}^{2^{n+1}} S_f\left(\frac{a_n}{\tilde{y}^2}\right) \quad (9)$$

is a polynomial in $\tilde{y}$ of degree 2^{n+1} whose largest real root is equal to b_n (remember that we assume that $b_n > 0$). Also, b_n is always a simple root of $\tilde{S}_f(\tilde{y})$ when $f(x)$ has not roots on the unit circle.

The relationship (8) permits us to treat π and b_n interchangeably. The advantage of dealing with b_n (rather than π) is twofold: it allows us to solve the problem of polynomial spectral factorization effectively by means of Gröbner bases, which we will see in the following subsection; and the quantity b_n is directly related to some essential quantity in control theory [30].

The next subsection develops an algebraic approach to the problem of discrete-time polynomial spectral factorization. Since the structure of roots of $f(x)$ is clear, we can make an extensive use of its properties we have investigated in this subsection. We stress here the importance of the discovery of useful structural properties and the exploitation thereof for acquiring effective algebraic approaches.

3.2 Solution Approach

If we compare the coefficients of the both sides of (3), a system of quadratic polynomial equations in b_i 's is obtained. Write as $\bar{g}_k$ the coefficient of the k -th order term of $g(x)g(\frac{1}{x}) - f(x)$, that is,

$$g(x)g\left(\frac{1}{x}\right) - f(x) = \sum_{k=-n}^n \bar{g}_{|k|} x^k . \quad (10)$$

Then, for each k , $k = 0, 1, \dots, n$, we have

$$\bar{g}_k = \sum_{i=0}^{n-k} b_i b_{i+k} - a_k .$$

The polynomial spectral factorization problem thus reduces to finding a particular zero of an ideal. Considering b_i , $i = 0, 1, \dots, n$, as variables and letting $\mathbf{B}_0 = \{b_0, b_1, \dots, b_n\}$, we have an ideal $\langle \mathcal{G}_0 \rangle$ in $\mathbb{R}[\mathbf{B}_0]$ generated by $\mathcal{G}_0 = \{\bar{g}_0, \bar{g}_1, \dots, \bar{g}_n\}$. Then each zero of $\langle \mathcal{G}_0 \rangle$ corresponds to the sign of b_n and $(\epsilon_1, \epsilon_2, \dots, \epsilon_n) \in \mathcal{P}$ through the relationships $g(x) = b_n \prod_{i=1}^n (x - \alpha_i^{\epsilon_i})$ and $a_n = b_n^2 \prod_{i=1}^n (-\alpha_i^{\epsilon_i})$. Hence the ideal $\langle \mathcal{G}_0 \rangle$ is 0 dimensional and has at most 2^{n+1} distinct zeros.

In the *continuous-time* case, the set of polynomials derived in an analogous fashion forms a Gröbner basis of the ideal generated by itself and we can immediately employ various results from Gröbner basis theory [17, 20]. Unfortunately the same is not true for the *discrete-time* case under investigation. However we can show that a different representation of the coefficients of $g(x)$ and some manipulation of the set of polynomials yield a desired Gröbner basis.

Another representation of $g(x)$ we use is

$$g(x) = \beta_n (x+1)^n + \beta_{n-1} (x+1)^{n-1} + \dots + \beta_0 . \quad (11)$$

Notice that b_i and β_j are related as $b_i = \sum_{j=i}^n \binom{j}{i} \beta_j$ and $\beta_j = \sum_{i=j}^n \binom{i}{j} (-1)^{i-j} b_i$, $j = 0, 1, \dots, n$, where $\binom{j}{i}$ is the binomial coefficient for $i, j \in \mathbb{N}$. Denote $\{\beta_0, \beta_1, \dots, \beta_n\}$ by $\mathbf{B}$. Then the conversion between $\mathbf{B}_0$ and $\mathbf{B}$ is linear and there is a one-to-one relationship. Also notice that $\beta_n = b_n (> 0)$, and we develop our approach around β_n .

We can transform each $\bar{g}_k(\mathbf{B}_0)$ to a polynomial in $\mathbf{B}$, which we denote by $\bar{g}_k(\mathbf{B})$. The set of polynomials $\{\bar{g}_0, \bar{g}_1, \dots, \bar{g}_n\}$ is still not a Gröbner basis, but a Gröbner basis can be computed in a very simple manner from $\bar{g}_k$ without resorting to algorithms such as Buchberger's algorithm. Let $c_{k,\ell}$, $k = 0, 1, \dots, n$, $k \leq \ell \leq n$, be

$$\begin{cases} c_{k,k} = 1 & k = 0, 1, \dots, n , \\ c_{0,\ell} = (-1)^\ell 2 & \ell = 1, 2, \dots, n , \\ c_{k,\ell} = (-1)^{k+\ell} \cdot \frac{2}{(2k)!} \cdot \frac{(k+\ell-1)!}{(\ell-k)!} \cdot \ell & \begin{cases} k = 1, 2, \dots, n , \\ k < \ell \leq n . \end{cases} \end{cases}$$

LEMMA 2. Let

$$\bar{\bar{g}}_k := \sum_{\ell=k}^n c_{k,\ell} \bar{g}_\ell \quad \left(= \bar{g}_k + \sum_{\ell=k+1}^n c_{k,\ell} \bar{g}_\ell \right) , \quad k = 0, 1, \dots, n .$$

Then the set of polynomials

$$\mathcal{G} := \{\bar{g}_0, \bar{g}_1, \dots, \bar{g}_n\}, \quad (12)$$

forms the reduced Gröbner basis of the ideal generated by $\{\bar{g}_0, \bar{g}_1, \dots, \bar{g}_n\}$ in $\mathbb{R}[\mathbf{B}]$ with respect to the graded reverse lexicographic order $\beta_n \succ \beta_{n-1} \succ \dots \succ \beta_0$, with β_k^2 being the leading monomial of $\bar{g}_k$. (The leading coefficients are 1.)

The proof is omitted due to space limitation but may be found in [21].

We call the ideal $\langle \mathcal{G} \rangle$ of $\mathbb{R}[\mathbf{B}]$ the *ideal of spectral factorization*. The set of the leading monomials of the elements of $\mathcal{G}$ is $\{\beta_0^2, \beta_1^2, \dots, \beta_n^2\}$. Therefore,

$$\mathcal{LB} := \{\beta_0^{d_0} \beta_1^{d_1} \dots \beta_n^{d_n} \mid d_k \in \{0, 1\}\}$$

forms a basis of the residue class ring $\mathbb{R}[\mathbf{B}]/\langle \mathcal{G} \rangle$ as an $\mathbb{R}$ -linear space, and $\dim_{\mathbb{R}} \mathbb{R}[\mathbf{B}]/\langle \mathcal{G} \rangle = \#\mathcal{LB} = 2^{n+1}$. These facts lead to the following lemma.

LEMMA 3. *The ideal of spectral factorization is 0 dimensional and the number of its zeros with multiplicities counted is 2^{n+1} .*

Once this ideal of spectral factorization is established based on the unconventional representation (11) of $g(x)$, the rest of the development follows essentially in the same line as the *continuous-time* case. Here some vital points are presented. Readers are referred to [1, 20] for full details of the continuous-time case. If $f(x)$ does not have multiple roots, there are exactly 2^n distinct possible root combinations and thus 2^{n+1} different polynomials $g(x)$ that satisfy (3). This also implies that there are exactly 2^{n+1} zeros of $\langle \mathcal{G} \rangle$. The ‘true’ $g(x)$, namely the spectral factor, corresponds to a zero with the largest real β_n . In order to simplify the search for this particular zero, we convert the Gröbner basis $\mathcal{G}$ into the so-called shape basis.

The ideal of spectral factorization for the discrete-time system has two cases, the *generic* case and the *singular* case, just as for the continuous-time system. Almost all $f(x)$ arising from practical applications fall into the *generic* case and we mainly discuss the generic case; some remark on the *singular* case will be made at the end of this subsection.

Definition 3. Given $f(x)$, when distinct $(\epsilon_1, \epsilon_2, \dots, \epsilon_n) \in \mathcal{P}$ give distinct $C(\epsilon_1, \epsilon_2, \dots, \epsilon_n)$, we call the situation a *generic case*. Otherwise it is called a *singular case*.

In either case we can show by using ‘generic coefficient’ and Lemma 4 in the next subsection that $\tilde{S}_f(\tilde{y})$ in (9) is identical to the characteristic polynomial of β_n modulo $\langle \mathcal{G} \rangle$, i.e., the characteristic polynomial of the linear map derived from the multiplication map [33]:

$$\mathbb{R}[\mathbf{B}]/\langle \mathcal{G} \rangle \ni g \mapsto \beta_n g \in \mathbb{R}[\mathbf{B}]/\langle \mathcal{G} \rangle.$$

Moreover, in the generic case, $\tilde{S}_f(\tilde{y})$ is square-free and its degree is 2^{n+1} . Noting that the ideal of spectral factorization has at most 2^{n+1} distinct zeros, we can immediately deduce that β_n is a separating element [25]. Due to the facts that $\langle \mathcal{G} \rangle$ is 0 dimensional and radical and that β_n is a separating element, we can obtain a special Gröbner basis called the *shape basis* in the discrete-time case as well.

THEOREM 1. *In the generic case the ideal of spectral factorization has a Gröbner basis so-called shape basis with respect to any elimination ordering $\{\beta_0, \beta_1, \dots, \beta_{n-1}\} \succ \beta_n$:*

$$\mathcal{F} := \{\tilde{S}_f(\beta_n), \beta_{n-1} - \tilde{h}_{n-1}(\beta_n), \dots, \beta_0 - \tilde{h}_0(\beta_n)\},$$

where $\tilde{S}_f$ is a polynomial of degree exactly 2^{n+1} and $\tilde{h}_i$ ’s are polynomials of degree strictly less than 2^{n+1} .

The theorem guarantees that, in the discrete-time case, just like the continuous-time case, all the coefficients of the spectral factor can be expressed as polynomials in β_n and therefore that the problem of polynomial spectral factorization can in essence be solved by finding the largest real root of $\tilde{S}_f(\tilde{y})$. Last but not least, since $\mathcal{G}$ is already a Gröbner basis, we can effectively compute a shape basis from $\mathcal{G}$ by way of the basis conversion (change-of-order) technique [25].

Here it should be emphasized that the proposed approach has properties favourable to practical applications. A Gröbner basis with respect to the graded reverse lexicographic order is obtained almost instantly from the problem formulation and the investigation of the properties of the ideal is thus possible. It is then seen that a shape basis is obtainable, and that relatively easily. Lastly we only have to examine the largest real root of $\tilde{S}_f(\tilde{y})$. A typical scenario is that one need to resort to an expensive calculation to derive a shape basis and then one has to investigate all the roots of the characteristic polynomial in search of the true solution. Those advantages are acquired by discovering the representation (11) that yields helpful structural properties.

Before closing this subsection we briefly discuss the *singular* case, where the characteristic polynomial $\tilde{S}_f(\tilde{y})$ has multiple roots. Even in this case we are still able to derive a shape basis in the same manner as in [1]. The key point is to add the *simple part* $\tilde{T}(\tilde{y})$ of $\tilde{S}_f(\tilde{y})$ to the ideal, where the simple part $\tilde{T}(\tilde{y})$ is the factor of $\tilde{S}_f(\tilde{y})$ obtained as the product of $\tilde{y} - \gamma_i$ for all simple roots γ_i of $\tilde{S}_f(\tilde{y})$. Note that $\tilde{T}(\tilde{y})$ can be computed by GCD of $\tilde{S}_f$ and its derivative. Then we can show the following.

THEOREM 2. *The ideal $\langle \mathcal{G}, \tilde{T}(\beta_n) \rangle$ has a shape basis with respect to any elimination ordering $\{\beta_0, \dots, \beta_{n-1}\} \succ \beta_n$:*

$$\{\tilde{T}(\beta_n), \beta_{n-1} - \tilde{h}_{n-1}(\beta_n), \dots, \beta_0 - \tilde{h}_0(\beta_n)\},$$

where $\tilde{h}_i$ ’s are polynomials of degree strictly less than that of $\tilde{T}$.

3.3 Parametric Case

This section deals with the case where each coefficient a_k is some polynomial in real parameters $\mathbf{p} = (p_1, p_2, \dots, p_m)$ over $\mathbb{Q}$. Now the polynomial $f(x)$ is considered as a multivariate one $f(x, \mathbf{p})$ in $\mathbb{Q}[x, \mathbf{p}]$. Even in the parametric case it often happens that the ideal of spectral factorization is generic for almost all combinations of parameter values. In such a case we can apply efficient *parametric basis conversion*, where we can compute the shape basis directly over the rational function field of parameters, as pointed out in [1] (See also Remark 1 below). Nevertheless we need to pay special attention to singular situations so that optimization that follows may be carried out thoroughly. It is shown here that such singularities can also be dealt with systematically. To this end the notion of ‘comprehensive Gröbner system’ is crucial and we can apply several techniques for its computation.

For each element $\mathbf{c} = (c_1, c_2, \dots, c_m)$ in $\mathbb{R}^m$, we denote by $\varphi_{\mathbf{c}}$ the ring homomorphism from $\mathbb{Q}[\mathbf{p}][\mathbf{B}]$ to $\mathbb{R}[\mathbf{B}]$ obtained by substitution of $\mathbf{p}$ with $\mathbf{c}$. For simplicity we denote by $f_{\mathbf{c}}(x)$ the polynomial $\varphi_{\mathbf{c}}(f) (= f(x, \mathbf{c}))$. To perform spectral factorization, we consider the following semi-algebraic set.

Definition 4. A semi-algebraic set $\mathbf{C} \subset \mathbb{R}^m$ is called a *regular region* if, for any $\mathbf{c} \in \mathbf{C}$, $a_n(\mathbf{c}) \neq 0$ and there exist no roots of unit modulus in $f_{\mathbf{c}}(x)$.

In the same way as the continuous-time case, the condition that f has no roots of unit modulus can be computed by the *quantifier elimination technique* or *real root counting* methods. Notice that a polynomial in x over $\mathbb{R}$ has a root of unit modulus if and only if it has a factor of the form of $x^2 + cx + 1$, $x - 1$, or $x + 1$ over $\mathbb{R}$. Due to the special structure (2), factors such as $x - 1$ and $x + 1$ in fact appear in $f(x)$ as their squares and thus are represented by $x^2 + cx + 1$. Substitute c with a new variable z . Then the resultant of $x^n f(x)$ and $x^2 + zx + 1$ becomes a polynomial $R(z)$ in z over $\mathbb{Q}[\mathbf{p}]$. It can be shown that, for a parameter value $\mathbf{c} \in \mathbb{R}^m$, $\varphi_{\mathbf{c}}(f)$ has no roots of unit modulus if and only if $\varphi_{\mathbf{c}}(R(z))$ has no real roots.

Now assume that $\mathbf{C}$ is a regular region for $f(x, \mathbf{p})$. We can compute the polynomial set $\mathcal{G}$ as in Lemma 2, where all polynomials are treated as ones over $\mathbb{Q}[\mathbf{p}]$. Also, for each $\mathbf{c} \in \mathbf{C}$, we can compute the polynomial set, say $\mathcal{G}_{\mathbf{c}}$, for $f_{\mathbf{c}}$ as in Lemma 2. Then, $\mathcal{G}_{\mathbf{c}} = \varphi_{\mathbf{c}}(\mathcal{G})$. Lemma 2 now implies that $\mathcal{G}$ and $\mathcal{G}_{\mathbf{c}}$ are still Gröbner bases over $\mathbb{Q}(\mathbf{p})$ and $\mathbb{Q}$ for ideals generated by themselves with respect to the graded reverse lexicographic order $\beta_n \succ \dots \succ \beta_0$.

We now consider the ideal $\langle \mathcal{G} \rangle$ of spectral factorization in $\mathbb{Q}(\mathbf{p})[\mathbf{B}]$ and the ideal $\langle \mathcal{G}_{\mathbf{c}} \rangle$ of spectral factorization in $\mathbb{R}[\mathbf{B}]$. We note that all arguments in the previous subsections can be applied to the ideals in $\mathbb{Q}(\mathbf{p})[\mathbf{B}]$, as $\mathbb{Q}(\mathbf{p})$ is a field.

An important computational property of the characteristic polynomial of β_n can be derived as follows: $\mathcal{LB} = \{\beta_0^{d_0} \beta_1^{d_1} \dots \beta_n^{d_n} \mid d_k \in \{0, 1\}\}$ is still a linear basis for $\mathbb{Q}(\mathbf{p})[\mathbf{B}]/\langle \mathcal{G} \rangle$, and, for the linear map $\mathbb{Q}(\mathbf{p})[\mathbf{B}]/\langle \mathcal{G} \rangle \ni g \rightarrow \beta_n g \in \mathbb{Q}(\mathbf{p})[\mathbf{B}]/\langle \mathcal{G} \rangle$, its matrix representation $M_{\mathbf{p}}$ with respect to $\mathcal{LB}$ is a matrix over $\mathbb{Q}[\mathbf{p}]$, since all the leading coefficients of the elements of $\mathcal{G}$ are 1 as polynomials in $\mathbb{Q}[\mathbf{p}, \mathbf{B}]$. Thus the characteristic polynomial $\tilde{S}_f$ is a polynomial in $\tilde{y}$ over $\mathbb{Q}[\mathbf{p}]$.

In the same manner, for each $\mathbf{c} \in \mathbf{C}$, we can compute the characteristic polynomial $\tilde{S}_{f_{\mathbf{c}}}$ as the characteristic polynomial of the matrix $M_{\mathbf{c}}$ derived from the linear map. Then, $M_{\mathbf{c}}$ coincides with the matrix obtained from $M_{\mathbf{p}}$ by substituting $\mathbf{p}$ with $\mathbf{c}$, and thus, $\varphi_{\mathbf{c}}(\tilde{S}_{f(x, \mathbf{p})}) = \tilde{S}_{f_{\mathbf{c}}}$.

LEMMA 4. *The characteristic polynomial $\tilde{S}_f$ is a monic polynomial over $\mathbb{Q}[\mathbf{p}]$, and, for each $\mathbf{c} \in \mathbf{C}$, the characteristic polynomial $\tilde{S}_{f_{\mathbf{c}}}$ can be computed by $\tilde{S}_{f_{\mathbf{c}}}(\beta_n) = \varphi_{\mathbf{c}}(\tilde{S}_f(\beta_n))$.*

Again there are the *generic case* and the *singular case* for the ideal spectral factorization over $\mathbb{Q}(\mathbf{p})$. In the generic case, $\tilde{S}_f(y)$ is square-free over $\mathbb{Q}(\mathbf{p})$ and the ideal $\langle \mathcal{G} \rangle$ is radical. Then we have the shape basis $\mathcal{F}$ with respect to an elimination ordering $\{\beta_0, \dots, \beta_{n-1}\} \succ \beta_n$ over $\mathbb{Q}(\mathbf{p})$ as in Theorem 1. For the singular case we compute the simple part $\tilde{T}$ of $\tilde{S}_f$ by using GCD computation, and consider the ideal $\langle \mathcal{G} \cup \{\tilde{T}\} \rangle$ instead of the original one. Then, over $\mathbb{Q}(\mathbf{p})$, we have the shape basis as in Theorem 2. Thus, from now on, we only deal with the generic case.

Now we outline our approach for the computation of the shape basis for the parametric case. Using *comprehensive Gröbner systems* [24, 29], we can compute the Gröbner basis of the ideal of spectral factorization with respect to the elimination ordering, where the region $\mathbf{C}$ may be decomposed

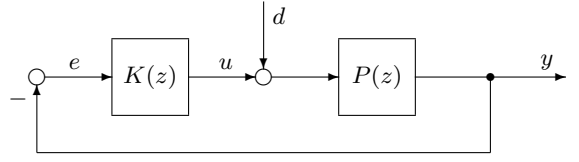


Figure 1: Unity feedback system configuration.

to a number of cells:

$$\mathbf{C} = \bigcup_{i \in I} \mathbf{C}_i \text{ and } \#I < \infty, \quad (13)$$

where each $\mathbf{C}_i$ is given as a semi-algebraic set and its associated polynomial set $\mathcal{G}_i$ in $\mathbb{Q}(\mathbf{p})[\mathbf{B}]$ is computed such that $\varphi_{\mathbf{c}}(\mathcal{G}_i)$ forms a Gröbner basis of $\langle \mathcal{G}_{\mathbf{c}} \rangle$ for any $\mathbf{c} \in \mathbf{C}_i$. We can make the above comprehensive Gröbner system *reduced*, that is, for each $\mathbf{c} \in \mathbf{C}$, $\varphi_{\mathbf{c}}(\mathcal{G}_i)$ is reduced. Then, from the discussion in the previous subsection, it follows that, for each cell $\mathbf{C}_i$ in (13), $\mathcal{G}_i$ is a shape basis if $\tilde{S}_{f_{\mathbf{c}}}$ is square-free for any $\mathbf{c}$ in $\mathbf{C}_i$.

Definition 5. In the resulted comprehensive Gröbner system (13), if $\mathcal{G}_i$ is a shape basis, we say that $\mathbf{C}_i$ is *generic*.

Thus, examining the forms of $\mathcal{G}_i$'s, we can extract all generic cells $\mathbf{C}_i$'s where $\mathcal{G}_i$'s are shape bases.

Remark 1. To extract such a generic cell more efficiently, we can use the technique of *parametric basis conversion* proposed in [1]. Over $\mathbb{Q}(\mathbf{p})$, both $\mathcal{G}$ and its shape basis $\mathcal{F}$ are Gröbner bases (with respect to different orderings) and so it can be shown that, for any $\mathbf{c} \in \mathbb{R}^m$, if the denominators of elements of $\mathcal{F}$ do not vanish on $\mathbf{c}$, then $\varphi_{\mathbf{c}}(\mathcal{F})$ is also the shape basis of $\langle \varphi_{\mathbf{c}}(\mathcal{G}) \rangle$.

For a non-generic cell $\mathbf{C}_i$, we can compute a certain shape basis by Theorem 2. In this case, by introducing new variable γ , we can extract the simple part of $\tilde{S}_f$. Let $\tilde{\mathcal{G}} := \mathcal{G} \cup \{\gamma \frac{d\tilde{S}_f}{d\beta_n} - 1\}$. Since the additional polynomial excludes multiple roots of $\tilde{S}_f$, computing the comprehensive Gröbner system for $\tilde{\mathcal{G}}$, we have a desired shape basis whose first element has β_n as its largest real root. Also we can apply efficient technique in [1].

4. NUMERICAL EXAMPLE

This section considers a particular control problem to demonstrate the discrete-time polynomial spectral factorization algorithm developed in Section 3 and further to show the applicability of the optimization framework proposed in this paper. The problem we deal with is the $\mathcal{H}_2$ regulation problem with input penalty, which is often employed in a research direction in control called the characterization of performance limitations [6]. The problem is formulated as follows. In the feedback configuration in Figure 1, $P(z)$ is a plant, i.e., a system we want to control, and $K(z)$ is a controller that we have to design. The disturbance signal $d(k)$ is taken to be a unit pulse signal and the cost function we employ is

$$E := \sum_{k=0}^{\infty} (|y(k)|^2 + |u(k)|^2).$$

The task is then to find a controller that achieves the smallest value of E among all stabilizing controllers and also the smallest value of E :

$$E^* := \min_{K \text{ stabilizing}} E.$$

Given a fixed $P(z)$, a standard numerical procedure allows us to find the value of E^* and the controller that achieves this. Our focus is thus on a plant with parameters $P(z; \mathbf{p})$, where $\mathbf{p}$ is a vector of real parameters that can be tuned, and we aim to find the minimum value of E^* :

$$\min_{\mathbf{p} \in \mathcal{Q}} E^*,$$

where $\mathcal{Q}$ is the feasible region of $\mathbf{p}$, specified as $\mathcal{F}(\mathbf{p})$ (cf. (1)). Write an n -th order single-input-single-output plant $P(z)$ as

$$P(z) = \frac{P_N(z)}{P_D(z)},$$

where $P_N(z)$ and $P_D(z)$ are coprime polynomials and $P_D(z)$ is n -th order and monic. Construct a self-reciprocal polynomial

$$P_N(z)P_N\left(\frac{1}{z}\right) + P_D(z)P_D\left(\frac{1}{z}\right) \quad (14)$$

(which corresponds to $f(x)$ in (2)), and write its spectral factor as

$$M_D(z) = m_n(x+1)^n + m_{n-1}(x+1)^{n-1} + \dots + m_0$$

(which corresponds to $g(x)$ in Definition 1 represented in the form of (11)). Under the assumptions that $P(z)$ is strictly proper and minimum-phase, E^* can be expressed as [30]

$$E^* = m_n^2 - 1. \quad (15)$$

Notice that (15) satisfies the condition for $\phi(\mathbf{p})$ in (1) since the development in Section 3 allows us to get a polynomial which has m_n as a real root and whose coefficients are polynomial in $\mathbf{p}$. It is also noted that, using the expressions of m_j 's, we can get an expression for the optimal controller. Thus the essential part of the solution is to compute the shape basis for m_j 's. It is emphasized that, although it is not immediate to see the relationship between the $\mathcal{H}_2$ regulation problem and the problem formulation (1), Gröbner basis theory can reformulate the control problem into (1) and thus that the formulation (1) is relevant to the control problem. Other important things to be noted here are that m_n (b_n and β_n in Section 3) represents an essential quantity in control and that it is beneficial to develop a polynomial spectral factorization algorithm in the discrete-time domain.

As a numerical example we employ the following:

$$P(z; p_1, p_2) = \frac{z - (p_1 - \frac{1}{8})p_1}{z^2 + (1 + \frac{1}{100}p_2)z + p_2^2 + \frac{1}{4}},$$

$$(p_1, p_2) \in \mathcal{Q} := \{(p_1, p_2) \mid p_1 \in [-\frac{1}{4}, \frac{1}{4}], p_2 \in [-\frac{1}{2}, \frac{1}{2}]\}.$$

Firstly it can be confirmed that the region $\mathcal{Q}$ is regular. Comparing the coefficients of (14) and $M_D(z)M_D(\frac{1}{z})$, we obtain a set of polynomials. Using Lemma 2, we can get a Gröbner basis with respect to the graded reverse lexicographic order $m_2 \succ m_1 \succ m_0$:

$$\begin{aligned} &\{m_0^2 - p_1^4 - p_2^4 + \frac{1}{4}p_1^3 + \frac{1}{50}p_2^3 - \frac{129}{64}p_1^2 - \frac{5001}{10000}p_2^2 + \frac{1}{4}p_1 + \frac{1}{200}p_2 - \frac{17}{16}, \\ &m_1^2 - 2m_2m_0 + m_1m_0 - \frac{1}{100}p_2^3 + p_1^2 + 3p_2^2 - \frac{1}{8}p_1 - \frac{1}{80}p_2 - \frac{1}{4}, \\ &m_2^2 + m_2m_1 + m_2m_0 - p_2^2 - \frac{1}{4}\}. \end{aligned}$$

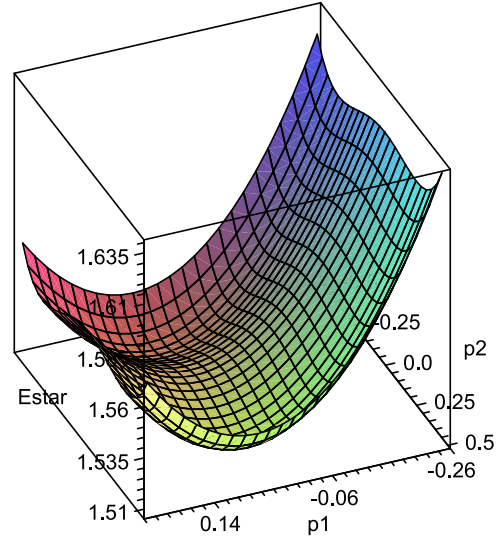


Figure 2: Plot of E^* .

It can further be converted into a shape basis (which is not included due to space limitation) by using the basis conversion computation over the rational function field of parameters. None of the denominators vanishes inside $\mathcal{Q}$ and thus $\mathcal{Q}$ is generic (see Remark 1). Lemma 4 then confirms that the first element is the correct characteristic polynomial of m_2 for all $\mathbf{p} \in \mathcal{Q}$. In fact, since we are focusing on m_2 only in this problem, it suffices to construct the multiplication matrix of m_2 and to compute its characteristic polynomial. From the characteristic polynomial of m_2 , we can obtain a polynomial in λ whose largest real root is equal to m_2^2 :

$$\begin{aligned} &\lambda^4 + (-p_1^4 - p_2^4 + \frac{1}{4}p_1^3 - \frac{1}{64}p_1^2 + \frac{14999}{10000}p_2^2 - \frac{1}{50}p_2 - \frac{41}{16})\lambda^3 \\ &+ (-2p_1^4p_2^2 - \frac{19999}{10000}p_2^6 + \frac{1}{2}p_1^3p_2^2 - \frac{1}{50}p_1^2p_2^3 + \frac{1}{50}p_2^5 + \frac{1}{2}p_1^4 - \frac{65}{32}p_1^2p_2^2 \\ &+ \frac{1}{400}p_1p_2^3 + \frac{30001}{20000}p_2^4 - \frac{1}{8}p_1^3 - \frac{1}{40}p_1^2p_2 + \frac{1}{4}p_1p_2^2 + \frac{1}{100}p_2^3 \\ &- \frac{319}{128}p_1^2 + \frac{1}{320}p_1p_2 - \frac{459983}{160000}p_2^2 + \frac{5}{16}p_1 + \frac{17}{800}p_2 + \frac{5}{32})\lambda^2 \\ &+ (-p_1^4p_2^2 - p_2^8 + \frac{1}{4}p_1^3p_2^4 - \frac{1}{2}p_1^4p_2^2 - \frac{1}{64}p_1^2p_2^4 + \frac{9999}{10000}p_2^6 \\ &+ \frac{1}{8}p_1^3p_2^2 - \frac{1}{50}p_2^5 - \frac{1}{16}p_1^4 - \frac{1}{128}p_1^2p_2^2 - \frac{37501}{20000}p_2^4 + \frac{1}{64}p_1^3 \\ &- \frac{1}{100}p_2^3 - \frac{1}{1024}p_1^2 - \frac{190001}{160000}p_2^2 - \frac{1}{800}p_2 - \frac{41}{256})\lambda \\ &+ p_2^8 + p_2^6 + \frac{3}{8}p_2^4 + \frac{1}{16}p_2^2 + \frac{1}{256}. \end{aligned}$$

By means of a special QE algorithm [18] and the relationship (15), we can find that the global optimum of E^* is

$$\min_{\mathbf{p} \in \mathcal{Q}} E^* \simeq 1.508,$$

and that this is achieved at

$$\mathbf{p}_{\text{opt}} \simeq (0.0625, -0.3457).$$

It is emphasized that the optimal value and $\mathbf{p}_{\text{opt}}$ are the true global optimal values (cf. Figure 2) and moreover can be obtained as algebraic numbers and that they can be computed with arbitrary accuracy. The computation times required for obtaining the result are

Computation of the shape basis :	0.01 [sec]
Optimization of E^* :	403.57 [sec]

by programs implemented in Maple running on a 1.33GHz PC with Intel Core Solo U1500.

5. CONCLUSIONS

This paper has proposed a new framework of symbolic optimization that may encompass a wide range of problems arising from practical applications. Algebraic approaches play a crucial rôle in the framework: formulation into this framework and solution of the problem formulated in this framework. Also emphasized is the significance of the exploitation of the structural properties inherent (sometimes in an obscure way) in the problem. As an example this paper has developed an algebraic approach to discrete-time polynomial spectral factorization, which is believed to deserve attention in its own right. It is expected that the development will find many applications in the area of signal processing.

6. ACKNOWLEDGMENTS

The authors would like to thank Ms. S. Gandy and Mr. H. Tanaka for their help in preparing the numerical example.

7. REFERENCES

- [1] H. Anai, S. Hara, M. Kanno, and K. Yokoyama. Parametric polynomial spectral factorization using the sum of roots and its application to a control design problem. Technical Report METR 2008-04, Department of Mathematical Informatics, The University of Tokyo, January 2008. Also submitted to the *Journal of Symbolic Computation*.
- [2] H. Anai, H. Yanami, K. Sakabe, and S. Hara. Fixed-structure robust controller synthesis based on symbolic-numeric computation: Design algorithms with a CACSD toolbox (invited paper). In *Proceedings of CCA/ISIC/CACSD 2004*, pages 1540–1545, Taipei, Taiwan, 2004.
- [3] B. D. O. Anderson, N. K. Bose, and E. I. Jury. Output feedback stabilization and related problems—solution via decision methods. *IEEE Transactions on Automatic Control*, AC-20(1):53–66, February 1975.
- [4] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 2nd edition, 2006.
- [5] C. W. Brown. QEPAD B: A program for computing with semi-algebraic sets using CADs. *ACM SIGSAM Bulletin*, 37(4):97–108, December 2003.
- [6] J. Chen and R. H. Middleton, editors. *IEEE Transactions on Automatic Control: Special Section on New Developments and Applications in Performance Limitation of Feedback Control*, volume 48, number 8. IEEE Control Systems Society, August 2003.
- [7] A. M. Cohen, editor. *Computer Algebra in Industry: Problem Solving in Practice*. John Wiley & Sons, Chichester, 1993.
- [8] G. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In H. Brakhage, editor, *Automata Theory and Formal Languages — 2nd GI Conference*, volume 33 of *Lecture Notes in Computer Science*, pages 134–183. Springer-Verlag, Berlin, 1975.
- [9] G. E. Collins and H. Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *Journal of Symbolic Computation*, 12(3):299–328, September 1991.
- [10] A. Dolzmann, A. Seidl, and T. Sturm. Efficient projection orders for CAD. In *Proceedings of the ACM SIGSAM International Symposium on Symbolic and Algebraic Computation, ISSAC2004*, pages 111–118. ACM Press, 2004.
- [11] P. Dorato, W. Yang, and C. Abdallah. Robust multi-objective feedback design by quantifier elimination. *Journal of Symbolic Computation*, 24(2):153–159, August 1997.
- [12] B. Dumitrescu. *Positive Trigonometric Polynomials and Signal Processing Applications*. Signals and Communication Technology. Springer, Dordrecht, The Netherlands, 2007.
- [13] I. A. Fotiou, P. Rostalski, P. A. Parrilo, and M. Morari. Parametric optimization and optimal control using algebraic geometry methods. *International Journal of Control*, 79(11):1340–1358, November 2006.
- [14] L. González-Vega, T. Recio, H. Lombardi, and M.-F. Roy. Sturm-Habicht sequences determinants and real roots of univariate polynomials. In B. Caviness and J. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 300–316. Springer, Wien, New York, 1998.
- [15] B. Hanzon and J. M. Maciejowski. Constructive algebra methods for the L_2 -problem for stable linear systems. *Automatica*, 32(12):1645–1657, December 1996.
- [16] H. Hoon and R. Liska, editors. *Journal of Symbolic Computation: Special Issue on Application of Quantifier Elimination*, volume 24, number 2. Academic Press, August 1997.
- [17] M. Kanno, H. Anai, and K. Yokoyama. On the relationship between the sum of roots with positive real parts and polynomial spectral factorization. In T. Boyanov et al., editors, *Numerical Methods and Applications — 6th International Conference, NMA 2006, Borovets, Bulgaria, August, 2006, Revised Papers*, volume 4310 of *Lecture Notes in Computer Science*, pages 320–328. Springer-Verlag, Heidelberg, 2007.
- [18] M. Kanno, S. Gandy, H. Anai, and K. Yokoyama. Optimizing the maximal real root of a polynomial by a special cylindrical algebraic decomposition. Presented at *Mathematical Aspects of Computer and Information Sciences 2007*, Paris, France, December 2007.
- [19] M. Kanno, S. Hara, H. Anai, and K. Yokoyama. Sum of roots, polynomial spectral factorization, and control performance limitations. In *Proceedings of the 46th IEEE Conference on Decision and Control*, pages 2968–2973, New Orleans, Louisiana USA, December 2007.
- [20] M. Kanno, K. Yokoyama, H. Anai, and S. Hara. Parametric optimization in control using the sum of roots for parametric polynomial spectral factorization. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation, ISSAC 2007*, pages 211–218, Waterloo, Ontario, Canada, July-August 2007.
- [21] M. Kanno, K. Yokoyama, H. Anai, and S. Hara. Symbolic optimization of algebraic functions. Technical Report METR 2008, Department of Mathematical Informatics, The University of Tokyo, April 2008.
- [22] V. Kučera. A tutorial on H_2 control theory: The continuous time case. In M. J. Grimble and V. Kučera, editors, *Polynomial Methods for Control Systems Design*, pages 1–55. Springer, London, 1996.
- [23] D. Lazard and F. Rouillier. Solving parametric polynomial systems. *Journal of Symbolic Computation*, 42(6):636–667, June 2007.
- [24] A. Montes. A new algorithm for discussing Gröbner bases with parameters. *Journal of Symbolic Computation*, 33(2):183–208, February 2002.
- [25] M. Noro and K. Yokoyama. A modular method to compute the rational univariate representation of zero-dimensional ideals. *Journal of Symbolic Computation*, 28(1-2):243–264, July 1999.
- [26] K. Ogata. *Discrete-time control systems*. Prentice-Hall International, London, 2nd edition, 1995.
- [27] H. Park. Optimal design of synthesis filters in multidimensional perfect reconstruction FIR filter banks using Gröbner bases. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 49(6):843–851, June 2002.
- [28] A. H. Sayed and T. Kailath. A survey of spectral factorization methods. *Numerical Linear Algebra with Applications*, 8(6-7):467–496, September-November 2001.
- [29] A. Suzuki and Y. Sato. A simple algorithm to compute comprehensive Gröbner bases using Gröbner bases. In *Proceedings of the ACM SIGSAM International Symposium on Symbolic and Algebraic Computation, ISSAC2006*, pages 326–331. ACM Press, 2006.
- [30] H. Tanaka, M. Kanno, and K. Tsumura. Expressions for discrete-time $\mathcal{H}_2$ control performance limitations based on poles and zeros. In *Proceedings of the SICE 8th Annual Conference on Control Systems (CD-Rom)*, Kyoto, Japan, March 2008.
- [31] V. Weispfenning. Simulation and optimization by quantifier elimination. *Journal of Symbolic Computation*, 24(2):189–208, August 1997.
- [32] V. Weispfenning. A new approach to quantifier elimination for real algebra. In B. Caviness and J. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 376–392. Springer, Wien, 1998.
- [33] K. Yokoyama, M. Noro, and T. Takeshima. Solutions of systems of algebraic equations and linear maps on residue class rings. *Journal of Symbolic Computation*, 14(4):399–417, October 1992.

Exact Certification of Global Optimality of Approximate Factorizations Via Rationalizing Sums-Of-Squares with Floating Point Scalars*

Erich Kaltofen
Dept. of Mathematics, NCSU
Raleigh, North Carolina
27695-8205, USA
kaltofen@math.ncsu.edu
<http://www.kaltofen.us>

Bin Li
Key Laboratory of Mathematics
Mechanization
AMSS, Beijing 100190, China
bli@mmrc.iss.ac.cn

Zhengfeng Yang
Dept. of Mathematics, NCSU
Raleigh, North Carolina
27695-8205, USA
zyang4@math.ncsu.edu

Lihong Zhi
Key Laboratory of Mathematics Mechanization
AMSS, Beijing 100190, China
lzhi@mmrc.iss.ac.cn <http://www.mmrc.iss.ac.cn/~lzhi>

ABSTRACT

We generalize the technique by Peyrl and Parillo [Proc. SNC 2007] to computing lower bound certificates for several well-known factorization problems in hybrid symbolic-numeric computation. The idea is to transform a numerical sum-of-squares (SOS) representation of a positive polynomial into an exact rational identity. Our algorithms successfully certify accurate rational lower bounds near the irrational global optima for benchmark approximate polynomial greatest common divisors and multivariate polynomial irreducibility radii from the literature, and factor coefficient bounds in the setting of a model problem by Rump (up to $n = 14$, factor degree = 13).

The numeric SOSes produced by the current fixed precision semi-definite programming (SDP) packages (SeDuMi, SOSTOOLS, YALMIP) are usually too coarse to allow successful projection to exact SOSes via Maple 11's exact linear algebra. Therefore, before projection we refine the SOSes by rank-preserving Newton iteration. For smaller problems the starting SOSes for Newton can be guessed without SDP ("SDP-free SOS"), but for larger inputs we additionally appeal to sparsity techniques in our SDP formulation.

Categories and Subject Descriptors: I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms; G.1.6 [Numerical Analysis]: Global optimization

*This material is based on work supported in part by the National Science Foundation under Grants CCF-0514585 and DMS-0532140 (Kaltofen and Yang) and OISE-0456285 (Kaltofen and Zhi). This research was partially supported by NKBRPC (2004CB318000) and the Chinese National Natural Science Foundation under Grant 10401035 (Li and Zhi).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

General Terms: algorithms, experimentation

Keywords: semidefinite programming, sum-of-squares, validated output, approximate factorization, hybrid method

1. INTRODUCTION

1.1 Motivation

Minimizing the deformations on inexact floating point scalars in the inputs to symbolic computation problems so that the resulting deformed inputs yield non-trivial outputs, such as common roots, factors or sparse interpolants, has resulted in a plethora of numerical optimization algorithms for those hybrid symbolic-numeric tasks (see, e.g., the survey [46]). We have successfully deployed structure preserving total least squares algorithms based on Newton iteration and the method of Lagrangian multipliers for the approximate polynomial greatest common divisor (GCD) problem [16], the approximate multivariate polynomial factorization problem [15], and the sparse multivariate polynomial and rational function interpolation [17] problem. Those optimization techniques efficiently produce local minima, which with randomizing start values and projections experimentally appear to actually be the global optima.

Semidefinite programming (SDP) is a far-reaching generalization to the interior-point methods of linear programming: theoretically, the SDPs can be solved in polynomial-time and can produce approximations to classical combinatorial optimization problems that were unachieved before. In addition and important to our setting, via sum-of-squares (SOS) and truncated moment techniques, SDP is applicable to global polynomial optimization problems (POPs) of the form

$$\left. \begin{array}{l} \min_{x \in \mathbb{R}^n} p(x) \\ \text{s. t. } q_1(x) \geq 0, \dots, q_l(x) \geq 0 \\ \text{where } p, q_1, \dots, q_l \in \mathbb{R}[X_1, \dots, X_n] \end{array} \right\} \quad (1)$$

(if the q_j are omitted, the problem is unconstrained).

The idea of bringing SDP-based polynomial optimization into hybrid symbolic-numeric computation was discussed

among James Demmel, Erich Kaltofen and Lihong Zhi at the October 2005 BIRS Workshop “Challenges in Linear and Polynomial Algebra in Symbolic Computation Software.” We now have a growing body of work [28, 13, 22, 23]. At this moment in time, we can draw the following conclusion: SDP is significantly more complex than our local (upper bound) algorithms, but one also gets more: namely an SOS certificate à la Putinar for the global infimum $r \in \mathbb{R}$ of (1) in the form of polynomials $u_i, v_{j,k} \in \mathbb{R}[X_1, \dots, X_n]$, $0 \leq i \leq m$, $1 \leq j \leq l$, $1 \leq k \leq \nu_j$ that satisfy the polynomial identity

$$p(X) - r = \sum_{i=1}^m u_i(X)^2 + \sum_{j=1}^l \left(q_j(X) \cdot \sum_{k=1}^{\nu_j} v_{j,k}(X)^2 \right) \quad (2)$$

Note that the above Positivstellensatz (2) [30] places restrictions on the constraints. Without constraints, not all such polynomials have a polynomial SOS (e.g., Motzkin’s polynomial), and either polynomial relaxations or a common square polynomial denominator u_0^2 in the $u_i^2, v_{j,k}^2$ (Emil Artin’s theorem) become necessary.

That is what the theory promises. However, an underlying assumption is that the interior-point iteration is performed with sufficiently high bigfloat precision, which current implementations do not support. The SOS certificate is numeric: it is subject to round-off errors of the floating point scalars in the computed equation (the exact solution may have irrational algebraic coefficients) and, more consequentially, to the numerical error from the fixed precision SDP solver itself. In fact, we will demonstrate on Siegfried Rump’s model problem how an SDP solver can become quite unstable with scale, see Table 1. In summary, (2) is satisfied only approximately by the polynomials SDP produces.

Our goal is to convert the imprecise and possibly invalid SOS certificate into an exact SOS certificate with exact rational scalars and polynomials, i.e., a proof, by bringing exact symbolic methods into play. Then the floating point precision in the SDP solver can be fixed, and further relaxations such as sparseness and “cheap-and-dirty” heuristics can be introduced without concern for the returned answer. One may even proceed by computing the numeric SOS certificate using Newton iteration directly rather than formulating a corresponding SDP. Since the global optimum can be an algebraic number in an extension of high degree [29], we shall verify a nearby rational lower bound $\tilde{r} \approx r$, $\tilde{r} \in \mathbb{Q}$. We assume that the coefficients of the polynomials p and q_j in (1) are represented exactly, for example as exact rational numbers. If they are floating point numbers, we can take their rational values. If the coefficients are irrational algebraic numbers with an exact representation, our methods also work, see Remark 2. Unlike Rump’s and Villard’s fully analyzed and validated numerical approach (see [43] and its references), our paradigm allows for unchecked numerical techniques and low precision floating point numbers because our exact rationalization catches all false certificates.

1.2 Used Approach and Results

We can formulate the problem of computing for an absolutely irreducible input polynomial h of total degree t in $\mathbb{R}[Z_1, \dots, Z_s]$ the nearest polynomial with a (real) factor of given total degree k as an unconstrained polynomial opti-

mization problem [15, 23]:

$$\min_{h_1, h_2 \in \mathbb{R}[Z_1, \dots, Z_s], \deg(h_1) = k, \deg(h_2) = t - k} \|h(Z) - h_1(Z)h_2(Z)\|^2 \quad (3)$$

with $n = \binom{s+k}{s} + \binom{s+t-k}{s}$ unknown coefficients. (With $\|\cdot\|$ we always denote the 2-norm of the coefficient vector.) Or we can eliminate the coefficients of h_2 by a symbolic least squares ansatz and obtain a multivariate rational function optimization problem:

$$\min_{x \in \mathbb{R}^n} \frac{f(x)}{g(x)} \quad (\text{where } g(x) > 0 \text{ for all } x \in \mathbb{R}^n) \quad (4)$$

with $n = \binom{s+k}{s}$ [13]. Since any polynomial can always be viewed as a rational function with the denominator as 1, in the following, we focus on the more general problem of minimization of a rational function by SOS:

$$\left. \begin{aligned} r^* &:= \sup_{r \in \mathbb{R}, W} r \\ \text{s. t. } & f(X) - rg(X) = m_d(X)^T \cdot W \cdot m_d(X) \\ & W \succeq 0, W^T = W \end{aligned} \right\} \quad (5)$$

where $m_d(X)$ is the column vector of all terms in $X_1, \dots, X_n$ up to degree d . The dimension of $m_d(X)$ is $\binom{n+d}{d}$. From (5) we know W is a symmetric positive semidefinite real matrix and the program is an SOS. We refer to [33, 31, 19, 28, 12] for description of SOS relaxations and their dual problems.

The problem of finding the rational SOS is equivalent to finding a rational positive semidefinite symmetric matrix W solving SOS problem (5) (see (13) below). Inspired by the method in [34], we start with a numerical solution W computed by semidefinite programming solving problem (5). However, we refine the matrix W further by using Newton iteration. In the next step, we lower r^* to a rational bound $\tilde{r}$, convert W to a rational matrix and project the matrix onto the affine linear hyperplane (cf. (5))

$$\mathcal{X} = \{A \mid A^T = A, f(X) - \tilde{r}g(X) = m_d(X)^T \cdot A \cdot m_d(X)\} \quad (6)$$

denoting the nearest matrix in $\mathcal{X}$ by $\widetilde{W}$. The projection is done by solving a rational least squares problem (11) exactly. The hope is that $\widetilde{W}$ is positive semidefinite, yielding a SOS proof (2) for the lower bound $\tilde{r}$. If $\widetilde{W}$ is not positive semidefinite, then we may increase the precision or reduce $\tilde{r}$ further and try again. It is always possible to compute a valid rational solution using sufficiently many digits, provided the optimal W remains positive semidefinite in a neighborhood [34, Proposition 3.1].

We demonstrate our exact certification strategy on three problems, the approximate greatest common divisor problem, the model problem by Rump, and the approximate factorization problem. Thus we are able to certify for several non-trivial problems from the literature minimum required distances to the nearest solvable problem that are fairly near their upper bounds. In fact, within minutes of computing we can prove a lower bound for our approximate GCD problem [16, Example 4.2] that is accurate to 5 decimal mantissa digits (6 decimal places). The lower bound establishes that the corresponding ill-conditioned Sylvester matrix is structurally well-conditioned. Rump’s [38] problem minimizes

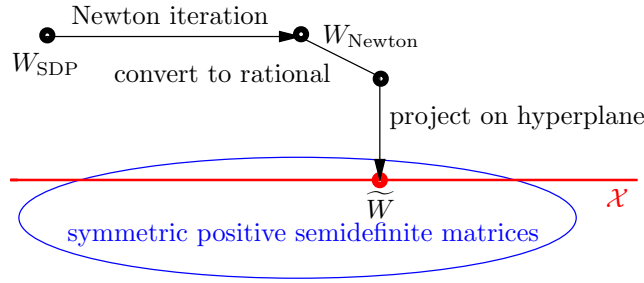


Figure 1: Rationalization of SOS

the factor coefficient bound

$$\begin{aligned} \mu_n = \min_{P, Q} \frac{1}{B_{n-1}} \quad & \left(\frac{1}{\mu_n} = \max_{P, Q} B_{n-1} \right) \\ \text{s. t. } \|P(Z)\|^2 \cdot \|Q(Z)\|^2 = B_{n-1} \|P(Z) \cdot Q(Z)\|^2 \\ P, Q \in \mathbb{R}[Z] \setminus \{0\}, \deg(P) \leq n-1, \deg(Q) \leq n-1 \end{aligned}$$

Mignotte’s [25] bound is $\frac{1}{\mu_n} \leq \binom{2n-2}{n-1}^2$. We can prove accurate lower bounds for μ_n as high as $n = 14$ and upper bounds at least as large as $n = 63$. Our upper bounds are obtained by Newton-Lagrange optimization with an incremental number of decimal mantissa digits (see also [1]). For $n = 14$, we have $1.62 \cdot 10^{11} < \frac{1}{\mu_{14}} < 3.12 \cdot 10^{11}$ with Digits := 16, while $\binom{26}{13}^2 \approx 1.08 \cdot 10^{14}$. The SOS-based lower bound for μ_{14} is at the limits of our fixed precision (15 decimal mantissa digits) SDP solvers. Finally, we have certified the irreducibility radius of [14, Example 3]. Our certified lower bound computed in under one minute is accurate to 5 decimal mantissa digits (8 decimal places).

Our method has also produced the following exact SOS representation of the non-negative polynomial $\mathcal{D}(\lambda)$ in [5] (the benchmark “Vor1” in Safey El Din’s paper in these Proceedings [40]): $16(au + au^2)^2 + (ay + a\beta + 2auy + 4a\beta u - a^2x - a^2\alpha + 4a\beta u^2 - 2a^2\alpha u)^2 + (y + \beta + 2\beta u - ax - a\alpha - 2aux - 4a\alpha u - 4a\alpha u^2)^2$. The YALMIP numeric sparse SOS + Gauss-Newton refinement + rational projection takes no more than 2 seconds in total.

In section 2 we illustrate how to extend the method in [34] to find the rational sums of squares of polynomials. The initial symmetric positive semidefinite matrix W can be obtained by solving the SDP problem (5). We also explore the possibility of starting at a random symmetric positive semidefinite matrix with given rank as W . We have thus successfully certified, without using SDP, the lower bounds for Rump’s problem up to $n = 10$. The size of $m_d(X)$ increases very fast with d and the number of variables. So the sparsity of the SDP problem (5) has to be exploited [36, 32, 44, 20, 27, 23]. See section 3.2.2 for a detailed discussion.

2. RATIONALIZING AN SOS DERIVED FROM SDP AND NEWTON ITERATION

The SOS program (5) can be solved efficiently by algorithms in SOSTOOLS [35], YALMIP [24] and SeDuMi [41]. However, since we are running fixed precision SDP solvers in Matlab, we can only obtain a numerical positive semidefinite matrix W and floating point number r^* which satisfy approximately

$$f(X) - r^*g(X) \approx m_d(X)^T \cdot W \cdot m_d(X), \quad W \succeq 0. \quad (7)$$

So r^* is a lower bound of $f(x)/g(x)$, $x \in \mathbb{R}^n$, approximately! For some applications, such as Rump’s model problem (16), due to the numerical error, the computed lower bounds can even be much bigger than upper bounds, see Table 1. This motivates us to consider how to use exact linear algebra tools to certify the lower bounds computed by SDP.

The lower bound $\tilde{r}$ is certified if $\tilde{r}$ and $\tilde{W}$ hold the following conditions exactly:

$$f(X) - \tilde{r}g(X) = m_d(X)^T \cdot \tilde{W} \cdot m_d(X), \quad \tilde{W} \succeq 0. \quad (8)$$

In the following subsections, we start with using Gauss-Newton iterations to refine r^* and W which satisfy (7) approximately, then compute the rational number $\tilde{r}$ and rational positive semidefinite symmetric matrix $\tilde{W}$ which satisfy (8) exactly. The projection steps are shown in Figure 1.

2.1 Newton Iteration

Let θ denote the backward error:

$$\theta = \|f(X) - rg(X) - m_d(X)^T \cdot W \cdot m_d(X)\|.$$

The floating point number r can be computed by solving the SOS program (5) or by other local optimization methods [6, 15]. In order to derive a satisfactory certified lower bounds, it is important to start with an accurate lower bound, denoted by r^* .

For the refined r^* , the positive semidefinite symmetric matrix W can be obtained by solving the following SOS program:

$$\left. \begin{aligned} \inf_W \quad & \text{Trace}(W) \\ \text{s. t. } \quad & f(X) - r^*g(X) = m_d(X)^T \cdot W \cdot m_d(X) \\ & W \succeq 0, W^T = W \end{aligned} \right\} \quad (9)$$

(here $\text{Trace}(W)$ acts as a dummy objective function that is commonly used in SDP for optimization problem without an objective function.)

We expand the quadratic form obtained from the SOS decomposition:

$$f(X) - r^*g(X) \approx \sum_{i=1}^k \left(\sum_{\alpha} c_{i,\alpha} X^\alpha \right)^2 \in \mathbb{R}[X]. \quad (10)$$

Here k is the rank of the matrix W . The rank deficiency of W corresponds to the number of global optima. For some applications, if the number of global optima is known, then k is known too. For Rump’s model problem (18), the rank deficiency of W is 1 if n is even; and 2 if n is odd. For the given tolerance, we compute the rank k by using the singular value decomposition of W .

We apply Gauss-Newton iteration to compute $\Delta c_{i,\alpha} X^\alpha$ such that

$$f(X) - r^* g(X) = \sum_{i=1}^k \left(\sum_{\alpha} c_{i,\alpha} X^\alpha + \Delta c_{i,\alpha} X^\alpha \right)^2 + O\left(\sum_{i=1}^k \left(\sum_{\alpha} \Delta c_{i,\alpha} X^\alpha \right)^2\right).$$

We update the matrix W accordingly to $W + \Delta W$ and the iteration is stopped when θ is less than the given tolerance τ . If θ remains greater than the given tolerance τ after several Gauss-Newton iterations, we may increase the precision of the SDP and Gauss-Newton iteration computations or use a smaller r^* and try the computations again.

The total number of X^α in $m_d(X)$ is $\binom{n+d}{d}$. So the computation of Gauss-Newton iteration is very heavy. It is necessary to exploit the sparsity of the SOS program (9). Fortunately, for many optimization problems arising from approximate polynomial computation, the sparsity can be discovered by analyzing the Newton polytope. We show in section 3.2.2 how to explore the sparsity for the problem (18).

Remark 1 It is possible to construct W as a random symmetric positive semidefinite matrix satisfying some given conditions such as rank, sparsity. However, if the backward error θ is large, we may need a big number of Gauss-Newton iterations to reduce θ below τ .

2.2 Rationalizing an SOS

In [34], a Macaulay 2 package is presented to compute an exact SOS decomposition from a numerical solution for non-negative polynomials with rational coefficients. We extend their technique to construct an exact rational SOS decomposition for the polynomial $f(X) - \tilde{r}g(X)$. Then $\tilde{r}$ is certified to be the lower bound of the minimization problem (4).

Suppose W has been refined by Gauss-Newton iterations such that $\|f(X) - r^*g(X) - m_d(X)^T \cdot W \cdot m_d(X)\| < \tau$. We approximate r^* by a nearby rational number $\tilde{r} \approx r^*$ and convert W to a rational matrix. Then we orthogonally project the refined matrix W to the rational matrix $\tilde{W}$ on the hyperplane $\mathcal{X}$ in (6). The projection is achieved by solving exactly the following least squares problems:

$$\left. \begin{array}{l} \min_{\tilde{W}} \|W - \tilde{W}\|_F^2 \\ \text{s. t. } f(X) - \tilde{r}g(X) = m_d(X)^T \cdot \tilde{W} \cdot m_d(X) \end{array} \right\} \quad (11)$$

It is equivalent to solve a set of smaller least squares problems:

$$\left. \begin{array}{l} \min_{\tilde{W}} \sum_{\alpha} \sum_{\beta+\gamma=\alpha} (W_{\beta,\gamma} - \tilde{W}_{\beta,\gamma})^2 \\ \text{s. t. } f_{\alpha} - \tilde{r}g_{\alpha} = \sum_{\beta+\gamma=\alpha} \tilde{W}_{\beta,\gamma} \end{array} \right\} \quad (12)$$

By solving the least squares problem (12) for each α , we get the minimal rational solution, denoted by $\tilde{W}$. Then we compute the exact L^TDL-decomposition [7] to check whether $\tilde{W}$ is a symmetric positive semidefinite matrix. The optimum $\tilde{r}$ is verified as the lower bound if

$$\left. \begin{array}{l} f(X) - \tilde{r}g(X) = m_d(X)^T \cdot \tilde{W} \cdot m_d(X) \\ = m_d(X)^T \cdot L^T \cdot D \cdot L \cdot m_d(X), \\ \text{such that } \forall i: D_{i,i} \geq 0, \end{array} \right\} \quad (13)$$

where $D_{i,i}$ is the i -th diagonal entry of the diagonal matrix D . If $\tilde{W}$ is not positive semidefinite, i.e., one entry, usually the last, in the diagonal matrix D is negative, we can decrease $\tilde{r}$ or increase the precision of Newton iterations to repeat the above computation. In our experiments, we usually choose $\rho \in [0.1 \cdot \theta, 0.5 \cdot \theta]$, and lower r^* to $\tilde{r} = r^* - \rho$. In our experience, the matrix $\tilde{W}$ becomes a positive semidefinite matrix.

Remark 2 The crucial property for verification is that the L^TDL-factorization of $\tilde{W}$ can be computed by an algorithm using rational arithmetic. The corresponding SOS is exact, but not rational:

$$f(X) - \tilde{r}g(X) = \sum_i (\sqrt{D_{i,i}} \cdot L_i \cdot m_d(X))^2, \quad (14)$$

where $D_{i,i} > 0$ is rational and L_i is the i -th row of L . We do not need this in our certificate, but if all coefficients are rational, a rational SOS exists [10]. If the coefficients of f and g are in an exact (not necessarily totally) real algebraic extension $K = \mathbb{Q}[\eta]/(\varphi(\eta))$, where $\varphi(\eta) \in \mathbb{Q}[\eta]$ and the symbol η represents a designated real root of φ , the certificate is verified exactly in the same manner by arithmetic in K . Note that $\tilde{r}$ is still chosen in $\mathbb{Q}$.

Algorithm Lower Bound Verification

Input: $\triangleright f(X_1, \dots, X_n), g(X_1, \dots, X_n) \in \mathbb{Q}[X_1, \dots, X_n]$: the numerator and denominator of a multivariate rational function.
 $\triangleright r^*$ (optional): the approximate optimum of the minimization problem.
 $\triangleright \tau \in \mathbb{R}_{>0}$: the given tolerance.
Output: $\triangleright \tilde{r}$: the verified lower bound.

1. Gauss-Newton refinement

(a) Get an approximate SOS decomposition.

Case SDP is used to compute W :

A. If r^* is given, set up SDP to compute W such that W satisfy (7).
Otherwise, set up SDP to compute r^* and W such that they satisfy (7).

B. Compute the numerical rank k of W and exploit the sparsity structure of polynomials in the SOS.

Case SDP is not used to compute W :

Construct the symmetric positive semidefinite matrix W randomly that satisfies the rank condition and given structures.

(b) Apply Gauss-Newton method to refine (10) and compute θ .

(c) If $\theta < \tau$, then get the refined matrix W .

Otherwise, decrease r^* and go back to step 1(a)A.

2. Compute the exact SOS

(a) Lower r^* to a rational number $\tilde{r}$ and convert W as a rational matrix.

(b) Compute the rational matrix $\tilde{W}$ by solving (11).

(c) Check whether $\tilde{W}$ is positive semidefinite. If so, return $\tilde{r}$. Otherwise, choose $\rho \in [0.1 \cdot \theta, 0.5 \cdot \theta]$, let $\tilde{r} = r^* - \rho$ and go back to step 2b.

Remark 3 Our projection method tries to achieve positive semidefiniteness for a rational $\tilde{r}$ and $\tilde{W}$ such that $\tilde{r}$ is as close as possible to r^* . We apply Gauss-Newton refinement to W for r^* (or a lowered r^*) and project using the even smaller $\tilde{r}$. Refinement with the actual target $\tilde{r}$ seems to bring W too close to the boundary of the cone of positive semidefinite matrices, and orthogonal projection fails to preserve that property.

3. THREE APPLICATIONS

3.1 Approximate Greatest Common Divisors

Approximate Greatest Common Divisors (GCDs) of univariate and multivariate polynomials can be transformed as global minimization of rational function (see [23] and the reference there). An approximate global optimum r^* can be attained by structured total least norm method [16] or SOS relaxations [23, 27]. By Algorithm Lower Bound Verification, we can find the rational number $\tilde{r}$ which is verified as the lower bound.

Example 1 (Example 4.2 in [16]) Consider the polynomials

$$1000Z_1^{10} + Z_1^3 - 1 \text{ and } Z_1^2 - \frac{1}{100}.$$

The local minimum computed by our STLN method is

$$r^* = 0.0421579164 \quad (15)$$

which can also be attained by SOS relaxations [23]. We decrease r^* by 4×10^{-8} to $r^* = 0.0421578636$ and then verify this lower bound. We reformulate the approximate GCD problem as rational function minimization problem. The numerator and denominator $f(X), g(X) \in \mathbb{Q}[X]$ can be obtained from the formulation in [18, 11, 23].

The matrix W is computed by solving the SOS program (5) with $\dim(m_d(X)) = 13$. We check that

$$\begin{aligned} \theta &= \|f(X) - r^*g(X) - m_d(X)^T \cdot W \cdot m_d(X)\| \\ &= 1.9430e-7. \end{aligned}$$

Case 1: Without applying Gauss-Newton iteration, we lower r^* to the rational lower bound

$$\tilde{r}_1 = 1414583 \cdot 2^{-27} \approx 0.0421578586.$$

Our SDP-SOS took 0.02 seconds* to obtain the rational matrix $\tilde{W}$ by solving (11) with the exact computation in Maple 11 with Digits:=20. By L^TDL-decomposition we find that $\tilde{W}$ is a positive semidefinite matrix:

$$f(X) - \tilde{r}_1 g(X) = m_d(X)^T \cdot \tilde{W} \cdot m_d(X) \geq 0.$$

Hence, $\tilde{r}_1$ is verified as the lower bound.

Case 2: Applying the Gauss-Newton iterations, the error is reduced to

$$\begin{aligned} \theta &= \|f(X) - r^*g(X) - m_d(X)^T \cdot W \cdot m_d(X)\| \\ &= 1.8140e-13. \end{aligned}$$

We lower r^* to the rational lower bound

$$\tilde{r}_2 = 45266661 \cdot 2^{-30} \approx 0.0421578633.$$

Our SDP-SOS took 47.6 seconds to obtain the rational positive semidefinite matrix $\tilde{W}$; $\tilde{r}_2$ is also verified as the lower bound.

Comparing the two cases, $\tilde{r}_2$ is a bit better than $\tilde{r}_1$. So we let $\tilde{r} = \tilde{r}_2$ as the certified lower bound for this GCD problem. And the local minimum computed by STLN method in [16] is the approximate global optimum.

Note that in [16] we had the globality of (15) verified by minimization with interval arithmetic [45]. The verification of the minimum was important because the Sylvester matrix of (1) is highly ill-conditioned, while the structured distance to singularity is large, a phenomenon that has been studied more since our discovery [2, 21].

3.2 Siegfried Rump's Model Problem

In this section, we solve the Rump's model problem by using sparse semidefinite programming (sparse SDP). The Newton-Lagrange method is applied to refine the global optima computed by SDP. Furthermore, we certify exact lower bounds by rationalizing sum-of-squares (SOS) decompositions.

3.2.1 Problem and Mathematical Background

Rump's model problem [39, 38] is that of computing the global minimum μ_n :

$$\mu_n = \min\{\|PQ\|^2 \mid P, Q \in \mathbb{R}[Z], \|P\| = \|Q\| = 1 \text{ and } \deg(P) = \deg(Q) = n-1\}. \quad (16)$$

For a non-singular system of linear equations $Ax = b$, we denote the Toeplitz condition number by $\kappa^{\text{Toep}}(A, x)$. It characterizes the sensitivity of the solution x with respect to infinitely small Toeplitz structured perturbations of the matrix A and perturbations of b . It has been proved in [37] that the ratio between the Toeplitz and the unstructured condition number satisfies

$$\frac{\kappa^{\text{Toep}}(A, x)}{\kappa(A, x)} = \alpha \frac{\|A^{-1}J\Psi_x\|}{\|A^{-1}\| \|x\|} \geq \frac{1}{\sqrt{n}} \frac{\sigma_{\min}(\Psi_x)}{\|x\|},$$

where the matrix Ψ_x is defined by

$$\Psi_x := \begin{bmatrix} x_1 & \dots & x_n & & \\ & & \dots & & \\ & & & x_1 & \dots & x_n \end{bmatrix} \in \mathbb{R}^{n \times (2n-1)},$$

$J \in \mathbb{R}^{n \times n}$ is the permutation matrix mapping $(1, \dots, n)^T$ into $(n, \dots, 1)^T$ and $\frac{1}{\sqrt{n}} \leq \alpha \leq \sqrt{2}$. It was shown in [39] that

$$\begin{aligned} \sqrt{\mu_n} &= \min_{\|x\|=1} \sigma_{\min}(\Psi_x) = \min_{\|x\|=\|y\|=1} \|\Psi_x y\| \\ &= \min_{\|x\|=\|y\|=1} \|\Psi_y x\| \end{aligned}$$

for all n . The upper bound of μ_n can be computed very efficiently. The challenge is to compute rigorous lower bounds for μ_n . In [38], verified lower bounds are given for n up to 8. In the following, we show that by rationalizing SOS computed from sparse SOS, we can efficiently compute the certified lower bounds for n up to 14.

3.2.2 Rational Function Minimization by Sparse SOS

The problem (16) is equivalent to the rational function

*All reported timings are on a 4 CPU (3GHz Xeon) MacPro with 9GB of memory under Linux 2.6.22-14 (Ubuntu).

n	μ_n^* from fixed prec. SDP	Newton-Lagrange	Rump's upper bound
3	0.111111111111111132	0.11111111111111112	0.11111111111111113
4	0.0174291733214352	0.01742917332143266	0.01742917332143269
5	0.00233959554819155	0.002339595548155594	0.002339595548155599
6	0.00028973187528375	0.0002897318752796807	0.0002897318752796843
7	0.0000341850701964797	0.00003418506980008289	0.00003418506980008323
8	0.00000390543564465773	0.000003905435649755721	0.000003905435649755845
9	4.36004072290608e-007	4.360016539181021e-007	4.360016539181362e-007
10	4.78395278113997e-008	4.783939568771179e-008	4.783939568772086e-008
11	5.18272812166654e-009	5.178749097446552e-009	5.178749097451150e-009
12	5.54188889223539e-010	5.545881831162859e-010	5.545881831173105e-010
13	4.06299438537872e-011	5.886688081195787e-011	5.886688081216679e-011
14	2.26410681869460e-010	6.202444992001861e-012	6.202444992172272e-012

Table 1: Rump's upper bounds and the optima computed by SDP and Newton-Lagrange

minimization:

$$\mu_n = \min \left\{ \frac{\|PQ\|^2}{\|P\|^2\|Q\|^2} \mid \begin{aligned} P(Z) &= \sum_{i=1}^n p_i Z^{i-1} \in \mathbb{R}[Z], \\ Q(Z) &= \sum_{i=1}^n q_i Z^{i-1} \in \mathbb{R}[Z]. \end{aligned} \right\} \quad (17)$$

The numbers of unknowns in coefficients of P, Q are $n(P) = n(Q) = n$. It has been shown in [39] that polynomials P, Q realizing the polynomials achieving μ_n must be symmetric (self-reciprocal) or skew-symmetric. Thus the problem can be rewritten into three optimization problems with three different constraints

$$\begin{aligned} p_{n+1-i} &= p_i, & q_{n+1-i} &= q_i, & 1 \leq i \leq n, \\ p_{n+1-i} &= p_i, & q_{n+1-i} &= -q_i, & 1 \leq i \leq n, \\ p_{n+1-i} &= -p_i, & q_{n+1-i} &= -q_i, & 1 \leq i \leq n, \end{aligned}$$

and the smallest of three minima is equal to μ_n . We are going to minimize the rational function $f(X)/g(X)$ with

$$\begin{aligned} f(X) &= \|PQ\|^2 = \sum_{k=2}^{2n} \left(\sum_{i+j=k} p_i q_j \right)^2, \\ g(X) &= \|P\|^2 \|Q\|^2 = \left(\sum_{i=1}^n p_i^2 \right) \left(\sum_{j=1}^n q_j^2 \right) \end{aligned}$$

and the variables

$$X = \{p_1, \dots, p_{n(P)}\} \cup \{q_1, \dots, q_{n(Q)}\},$$

where $n(P) = n(Q) = \lceil n/2 \rceil$. An SOS-relaxed lower bound of μ_n can be obtained by solving the following SOS program:

$$\left. \begin{aligned} \mu_n^* &:= \sup_{r \in \mathbb{R}, W} r \\ \text{s. t. } & f(X) - rg(X) = m_d(X)^T \cdot W \cdot m_d(X) \\ & W \succeq 0, W^T = W \end{aligned} \right\} \quad (18)$$

here $\deg(f) = \deg(g) = 4$ and $m_d(X)$ is the column vector of all terms up to degree $d = 2$. The dimension of real symmetric matrix W is $\binom{n(P)+n(Q)+2}{2}$ and there are $\binom{n(P)+n(Q)+4}{4}$ equality constraints.

Similar to [23], we exploit the sparsity in the SOS program (18). Let $C(p)$ denote the convex hull of $\text{supp}(p) = \{\alpha \mid p_\alpha \neq 0\}$ for polynomial $p(X) = \sum_\alpha p_\alpha X^\alpha$. Suppose

$$f(X) - rg(X) = m_d(X)^T \cdot W \cdot m_d(X) = \sum_k u_k(X)^2.$$

Let the polynomial $h(X) = \sum_{i=1}^{n(P)} \sum_{j=1}^{n(Q)} p_i q_j$, and X^α be any term in the SOS, then

$$\alpha \in \frac{1}{2} C(f(X) - rg(X)) \subseteq \frac{1}{2} C(h(X)^2) = C(h(X)).$$

According to the property of convex hulls, there exist non-negative numbers $\lambda_{i,j}$ for $1 \leq i \leq n(P), 1 \leq j \leq n(Q)$ such that

$$X^\alpha = \prod_{i,j} (p_i q_j)^{\lambda_{i,j}} = \prod_i p_i^{\sum_j \lambda_{i,j}} \prod_j q_j^{\sum_i \lambda_{i,j}}, \quad \sum_{i,j} \lambda_{i,j} = 1.$$

So X^α must be $p_i q_j$ for some i, j . The sparse SOS program of the rational function minimization problem (17) is:

$$\left. \begin{aligned} \mu_n^* &:= \sup_{r \in \mathbb{R}, W} r \\ \text{s. t. } & f(X) - rg(X) = m_G(X)^T \cdot W \cdot m_G(X) \\ & W \succeq 0, W^T = W \end{aligned} \right\} \quad (19)$$

where $m_G(X)$ is the column vector containing all terms $p_i q_j$ for $1 \leq i \leq n(P), 1 \leq j \leq n(Q)$. The dimension of the real symmetric matrix W is $n(P)n(Q)$ and the number of equality constraints is $\binom{n(P)+1}{2} \binom{n(Q)+1}{2}$. For $n = 14$, the sparse SOS program (19) has 784 equality constraints and $\dim(W) = 49$ while the SOS program (18) has 3060 equality constraints and $\dim(W) = 120$.

The SOS program (19) can be solved efficiently by algorithms in SOSTOOLS [35], YALMIP [24] and SeDuMi [41]. However, we can see from the Table 1, the lower bounds computed by the fixed precision SDP solver can be even larger than the upper bounds given in [38]. There are different techniques to improve the output of SDP solver, for example, Newton iteration.

Again, let $f(X) = \|PQ\|^2$, $c_1(X) = \|P\|^2 - 1$ and $c_2(X) = \|Q\|^2 - 1$, we apply Newton-Lagrange method to solve the equality-constrained problem

$$\begin{aligned} \min_x & f(x) \\ \text{s. t. } & c_1(x) = c_2(x) = 0. \end{aligned}$$

The initial values of X are the coefficients of the polynomials P, Q computed by solving the sparse SOS program (19).

From the Table 1, we can see that the global optima computed by solving the sparse SOS program (19) in Matlab and refined by Newton-Lagrange method in Maple 11 with Digits := 16 are strictly smaller than Rump's upper bounds [38]. Actually, we can apply the Newton-Lagrange method to any initial guess for the polynomials P and Q , it is not necessary to start with solving the SOS program (19).

n	time(s)	Digits	certified lower bound	Rump's lower bound
3	0.028	20	0.11111111111111111111	0.1111111111111111083
4	0.368	20	0.017429173321432652	0.01742917332143174
5	8.128	20	0.002339595548155591	0.002339595548155278
6	182.8	20	0.0002897318752796800	0.0002897318752795867
7	837.4	20	0.00003418506980008203	0.00003418506980004407
8	2.112	15	0.000003905435649455700	0.000003905435649743504
9	7.008	15	4.36001623918100e-007	?
10	34.14	15	4.78393556877000e-008	?
11	27.93	15	5.17774909740000e-009	?
12	174.1	15	5.51588183110000e-010	?
13	181.2	15	5.78668808100000e-011	?
14	1556	15	3.20244499200000e-012	?

Table 2: The certified lower bounds and Rump's lower bounds

However, the output of (19) provides us an initial positive semidefinite matrix W which can be used in the following lower bound certification.

Remark 4 Boyd and Collins have investigated the maximization of the single factor coefficient bound for integer polynomials with respect to infinity norm [3, and its references]. Collins establishes that the ratio of (the minimal height among irreducible factors)/(height of their product) can be ≥ 2.2005 . In [1] we compute from the Rump polynomials (see Table 1) with $n = 67$ an integer polynomial of degree 132 with the record-breaking ratio ≥ 3.30846991 .

3.2.3 Lower Bound Certification

Given the rational number $r_n = \mu_n^* - \rho_n$ for the small positive number ρ_n . We use the matrix W computed by solving the sparse SOS program (19) as the initial approximate positive semidefinite matrix. Applying the Newton iteration method to the system of equations obtained from the sparse SOS decomposition

$$\begin{aligned} f(X) - r_n g(X) &= m_G(X)^T \cdot W \cdot m_G(X) \\ &= \sum_k (\sum_\alpha c_{k,\alpha} X^\alpha)^2 \in \mathbb{R}[X], \end{aligned}$$

we get the refined positive semidefinite matrix W . The numerical matrix W is converted to the rational matrix W . By solving the least squares problem

$$\begin{aligned} \min_W \sum_{i,j} (w_{i,j} - \tilde{w}_{i,j})^2 \\ \text{s. t. } f(X) - r_n g(X) &= m_G(X)^T \cdot \tilde{W} \cdot m_G(X) \end{aligned}$$

we project W to the hyperplane defined by the SOS decomposition. If the computed projected rational matrix $\tilde{W}$ is positive semidefinite, then we get the rational SOS decomposition and r_n is the certified lower bound for μ_n . If it's not, then we can increase the precision for solving SDP and Newton iteration or try smaller r_n .

In the Table 2, we compare the certified lower bounds obtained by our algorithm applying Gauss-Newton iterations with the lower bounds given in [38]. For $n \leq 7$ we used Maple 11's Digits:=20, while for $n \geq 8$ we used machine precision. The case of $n = 14$ is our most difficult certificate (see also Remark 3). The numerators and denominators of its rational numbers have up to 945 decimal digits: http://www4.ncsu.edu/~kaltoven/software/certif/model14_SOS.txt.

Remark 5 We have also tried to avoid SDP altogether and to start with a random positive semidefinite matrix with

rank deficiency 1 for n being even and 2 for n being odd. For n up to 10, we can certify the same lower bounds listed in Table 2.

3.3 Approximate Polynomial Factorizations

We can formulate the problem of computing for an absolutely irreducible input polynomial of total degree t in $\mathbb{R}[Z_1, \dots, Z_s]$ the nearest polynomial with a factor P of given total degree k and cofactor Q of given total degree $t - k$ as a polynomial optimization problem $\min_{X \in \mathbb{R}^n} f(X)$ with $n = \binom{s+k}{s} + \binom{s+t-k}{s}$. It has been pointed out in [13], without exploring the sparsity of the SOS problem (5), it seems difficult to certify the lower bound even for Example 3 in [14], which originally was posed in [26, Example 1]. But the sparsity of the SOS problem for the approximate factorization can be exploited (see [23]). We have the following sparse SOS program:

$$\left. \begin{aligned} r^* &:= \sup_{r \in \mathbb{R}, W} r \\ \text{s. t. } & \begin{cases} f(X) - r = m_G(X)^T \cdot W \cdot m_G(X) \\ W \succeq 0, W^T = W \end{cases} \end{aligned} \right\} \quad (20)$$

where $m_G(X)$ is the column vector containing 1 and all terms $p_i q_j$ for $1 \leq i \leq \binom{s+k}{s}$ and $1 \leq j \leq \binom{s+t-k}{s}$, where p_i and q_j are unknown coefficients of polynomials P and Q respectively.

Example 2 (Example 3 in [14]) Consider the polynomial

$$(Z_1^2 + Z_2 Z_1 + 2Z_2 - 1)(Z_1^3 + Z_2^2 Z_1 - Z_2 + 7) + \frac{1}{5} Z_1$$

we are going to certify the lower bounds for polynomial having a degree one factor or a degree two factor. The global minima r^* computed by solving the sparse SOS program (20) is 0.2947781736518 for $k = 1$, and 0.000413702070 for $k = 2$. The corresponding upper bound in [15, Table 1, Example 1] is 0.00041370181014226, which is lower than the numeric SOS bound. The objective function $f(X) \in \mathbb{Q}[X]$ can be obtained from the formulation (3) (see also in [23]).

For $k = 1$, the matrix W is computed by solving the sparse SOS program (20) with $\dim(m_G(X)) = 46$. We check that

$$\theta = \|f(X) - r^* - m_G(X)^T \cdot W \cdot m_G(X)\| = 1.06591917\text{e-}11.$$

Without applying the Gauss-Newton iterations, we lower r^* to a rational number

$$\tilde{r} = 633031307 \cdot 2^{-31} \approx 0.2947781733237$$

and obtain the rational matrix $\widetilde{W}$ by solving (11) with the exact computation in Maple 11. By L^TDL -decomposition we find that $\widetilde{W}$ is a positive semidefinite matrix:

$$f(X) - \tilde{r}g(X) = m_G(X)^T \cdot \widetilde{W} \cdot m_G(X) \geq 0.$$

Hence, $\tilde{r}$ is verified as the lower bound. And the local minimum computed by solving the sparse SOS program (20) is the approximate global optimum. The algorithm takes 0.56 second to certify the lower bound.

For $k = 2$, the matrix W is computed by solving the sparse SOS program (20) with $\dim(m_G(X)) = 61$. We check that

$$\theta = \|f(X) - r^* - m_G(X)^T \cdot W \cdot m_G(X)\| = 1.04543742e-11.$$

Without applying the Gauss-Newton iterations, we lower r^* to a rational number

$$\tilde{r} = 111052 \cdot 2^{-28} \approx 0.000413700938$$

and obtain the rational matrix $\widetilde{W}$ by solving (11) with the exact computation in Maple 11. By L^TDL -decomposition we find that $\widetilde{W}$ is a positive semidefinite matrix:

$$f(X) - \tilde{r}g(X) = m_G(X)^T \cdot \widetilde{W} \cdot m_G(X) \geq 0.$$

Hence, $\tilde{r}$ is verified as the lower bound. And the local minimum computed by the approximate factorizer is the approximate global optimum. The algorithm takes 2.56 seconds to certify the lower bound. This is, due to 61 squares, our largest certificate. The numerators and denominators of its rational numbers have up to 1132 decimal digits: http://www4.ncsu.edu/~kaltofen/software/certif/factor_SOS.txt.

Remark 6 Even though our certificates seem to contain rational numbers with many digits in their numerators and denominators (their logarithmic height), the exact linear algebra (Steps 2b and 2c) does not contribute significantly to the running time. In fact, once a floating point matrix $\widetilde{W}$ that projects to a positive semidefinite rationalization $\widetilde{W}$ is found (see Remark 3), our algorithm completes quickly, in part because very fast exact linear algebra methods are available (see, e.g., [8, 47]). In fact, assuming that the floating point numbers in the numeric SOS are reasonably bounded in magnitude, the number of digits in the rational scalars of the certificate grow no more than quadratically (least squares + L^TDL factorization) in the dimension of the W matrix (see (14)). Note that if the rank of W , i.e., the number of squares in the SOS, is much smaller than the dimension (see Step 1a), the number of digits is also proportionally less. The best worst case logarithmic height bounds for exact symbolic methods appear to grow exponentially in the number of variables in the arising polynomial systems [4]. The logarithmic height of our certificates is reduced because we certify a nearby rational lower bound of small height and we project an SOS of fixed floating point precision. The latter places a form of well-conditionedness restriction on the optimization problems that we currently can certify.

4. CONCLUSION

By rationalizing a numeric sum-of-squares representation to an exact certificate we can eliminate numerical inaccuracies from our stated lower bounds. We have certified lower bounds for several problems in hybrid symbolic-numeric computation that previously remained unproven. The limiting

requirement of our approach is that the rationalized moment matrix remains positive semidefinite, so that an exact sum-of-squares representation can be obtained. We achieve the former by refining SDP-based numeric SOSes by rank and structure-preserving Gauss-Newton iteration, and the latter by relaxing the lower bound. It is not clear that one needs SDP to seed the Gauss-Newton iteration, or that orthogonal projection is the best for preserving positive semidefiniteness (see Remark 3). Finally, the SOSes can be tightened to Artin-style rational function sums-of-squares, or so-called “big-ball” constraints can be added (we know non-negative polynomials f for which $f + \delta$ are not SOSes for all real $\delta \geq 0$). We plan to investigate those directions further, and we can freely deploy wild and unanalyzed numerical optimization heuristics, because our truly hybrid paradigm yields exact symbolic certificates that leave no doubt.

Acknowledgments: We thank Kartik Sivaramakrishnan for suggesting the exploitation of sparsity in our SDPs.

Lihong Zhi is also grateful to the IMA in Minneapolis for hosting her during the winter quarter of the IMA thematic year on applications of algebraic geometry.

5. REFERENCES

- [1] BOYD, D., KALTOFEN, E., AND ZHI, L. Integer polynomials with large single factor coefficient bounds. In preparation, 2008.
- [2] CHÈZE, G., AND YAKOUBSOHN, J.-C. Distance computation to the resultant variety, Nov. 2007. Talk at the Journées 2007 da l’ANR GECKO, URL: <http://www-sop.inria.fr/galaad/conf/07gecko/Yakoubshon.J.C.pdf>.
- [3] COLLINS, G. E. Single-factor coefficient bounds. *J. Symbolic Comput.* 38, 6 (2004), 1507–1521.
- [4] DAHAN, X., AND SCHOST, É. Sharp estimates for triangular sets. In Gutierrez [9], pp. 103–110.
- [5] EVERETT, H., LAZARD, D., LAZARD, S., AND SAFEY EL DIN, M. The Voronoi diagram of three lines in r^3 . In *SoCG ’07: Proceedings of the 23-rd Annual Symposium on Computational Geometry* (2007), ACM, New York, USA, pp. 255–264.
- [6] GAO, S., KALTOFEN, E., MAY, J. P., YANG, Z., AND ZHI, L. Approximate factorization of multivariate polynomials via differential equations. In Gutierrez [9], pp. 167–174.
- [7] GOLUB, G. H., AND VAN LOAN, C. F. *Matrix Computations*, third ed. Johns Hopkins University Press, Baltimore, Maryland, 1996.
- [8] GREGORY, B., AND KALTOFEN, E. Analysis of the binary complexity of asymptotically fast algorithms for linear system solving. *SIGSAM Bulletin* 22, 2 (Apr. 1988), 41–49.
- [9] GUTIERREZ, J., Ed. *ISSAC 2004 Proc. 2004 Internat. Symp. Symbolic Algebraic Comput.* (New York, N. Y., 2004), ACM Press.
- [10] HILLAR, C. Sums of polynomial squares over totally real fields are rational sums of squares. *Proc. American Math. Society* (2008). To appear. URL: <http://www.math.tamu.edu/~chillar/files/totallyrealsos.pdf>.
- [11] HITZ, M. A., AND KALTOFEN, E. Efficient algorithms for computing the nearest polynomial with constrained roots. In *Proc. 1998 Internat. Symp. Symbolic Algebraic Comput. (ISSAC’98)* (New York, N. Y., 1998), O. Gloor, Ed., ACM Press, pp. 236–243.
- [12] JIBETAN, D., AND DE KLERK, E. Global optimization of rational functions: a semidefinite programming approach. *Math. Program.* 106, 1 (2006), 93–109.
- [13] KALTOFEN, E., LI, B., SIVARAMAKRISHNAN, K., YANG, Z., AND ZHI, L. Lower bounds for approximate factorizations via semidefinite programming (extended abstract). In Verschelde and Watt [42], pp. 203–204.
- [14] KALTOFEN, E., AND MAY, J. On approximate irreducibility of polynomials in several variables. In *ISSAC 2003 Proc. 2003 Internat. Symp. Symbolic Algebraic Comput.* (New York, N. Y., 2003), J. R. Sendra, Ed., ACM Press, pp. 161–168.
- [15] KALTOFEN, E., MAY, J., YANG, Z., AND ZHI, L. Approximate factorization of multivariate polynomials using singular value decomposition. *J. Symbolic Comput.* 43, 5 (2008), 359–376.

- [16] KALTOFEN, E., YANG, Z., AND ZHI, L. Approximate greatest common divisors of several polynomials with linearly constrained coefficients and singular polynomials. In *ISSAC MMVI Proc. 2006 Internat. Symp. Symbolic Algebraic Comput.* (New York, N. Y., 2006), J.-G. Dumas, Ed., ACM Press, pp. 169–176.
- [17] KALTOFEN, E., YANG, Z., AND ZHI, L. On probabilistic analysis of randomization in hybrid symbolic-numeric algorithms. In Vershelde and Watt [42], pp. 11–17.
- [18] KARMARKAR, N. K., AND LAKSHMAN Y. N. On approximate GCDs of univariate polynomials. *J. Symbolic Comput.* 26, 6 (1998), 653–666. Special issue on Symbolic Numeric Algebra for Polynomials S. M. Watt and H. J. Stetter, editors.
- [19] LASSERRE, J. B. Global optimization with polynomials and the problem of moments. *SIAM J. on Optimization* 11, 3 (2000), 796–817.
- [20] LASSERRE, J. B. Global SDP-relaxations in polynomial optimization with sparsity. *SIAM J. on Optimization* 17, 3 (2006), 822–843.
- [21] LI, B., LIU, Z., AND ZHI, L. Structured condition numbers of Sylvester matrices (extended abstract), Dec. 2007. Presented at MACIS 2007, URL: http://www-spiral.lip6.fr/MACIS2007/Papers/submission_17.pdf.
- [22] LI, B., NIE, J., AND ZHI, L. Approximate GCDs of polynomials and SOS relaxation. In Vershelde and Watt [42], pp. 205–206.
- [23] LI, B., NIE, J., AND ZHI, L. Approximate GCDs of polynomials and sparse SOS relaxations. Manuscript, 16 pages. Submitted, 2007.
- [24] LÖFBERG, J. YALMIP : A toolbox for modeling and optimization in MATLAB. In *Proc. IEEE CCA/ISIC/CACSD Conf.* (Taipei, Taiwan, 2004). URL: <http://control.ee.ethz.ch/~joloef/yalmip.php>.
- [25] MIGNOTTE, M. Some useful bounds. In *Computer Algebra*, B. Buchberger, G. Collins, and R. Loos, Eds., 2 ed. Springer Verlag, Heidelberg, Germany, 1982, pp. 259–263.
- [26] NAGASAKA, K. Towards certified irreducibility testing of bivariate approximate polynomials. In *Proc. 2002 Internat. Symp. Symbolic Algebraic Comput. (ISSAC'02)* (New York, N. Y., 2002), T. Mora, Ed., ACM Press, pp. 192–199.
- [27] NIE, J., AND DEMMEL, J. Sparse SOS relaxations for minimization functions that are summation of small polynomials, 2007.
- [28] NIE, J., DEMMEL, J., AND GU, M. Global minimization of rational functions and the nearest GCDs. *ArXiv Mathematics e-prints* (Jan. 2006). URL: <http://arxiv.org/pdf/math/0601110>.
- [29] NIE, J., RANESTAD, K., AND STURMFELS, B. The algebraic degree of semidefinite programming, 2006. URL: <http://www.citebase.org/abstract?id=oai:arXiv.org:math/0611562>.
- [30] NIE, J., AND SCHWEIGHOFER, M. On the complexity of Putinar's Positivstellensatz. *J. Complexity* 23 (2007), 135–70.
- [31] PARRILO, P. A. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, May 2000. URL: <http://www.cds.caltech.edu/~pablo/>.
- [32] PARRILO, P. A. Exploiting algebraic structure in sum of squares programs. *Lecture Notes in Control and Information Sciences* 312 (2005), 181–194.
- [33] PARRILO, P. A., AND STURMFELS, B. Minimizing polynomial functions. *DIMACS series in discrete mathematics and theoretical computer* 60 (2003), 83. URL: <http://www.citebase.org/abstract?id=oai:arXiv.org:math/0103170>.
- [34] PEYRL, H., AND PARRILO, P. A. A Macaulay 2 package for computing sum of squares decompositions of polynomials with rational coefficients. In Vershelde and Watt [42], pp. 207–208.
- [35] PRAJNA, S., PAPACHRISTODOULOU, A., AND PARRILO, P. A. SOSTOOLS: Sum of squares optimization toolbox for MATLAB. URL: <http://www.cds.caltech.edu/sostools>.
- [36] REZNICK, B. Extremal PSD forms with few terms. *Duke Mathematical Journal* 45, 2 (1978), 363–374.
- [37] RUMP, S. M. Structured perturbations part I: Normwise distances. *SIAM J. Matrix Anal. Applic.* 25, 1 (2003), 1–30.
- [38] RUMP, S. M. Global optimization: a model problem, 2006. URL: <http://www.ti3.tu-harburg.de/rump/Research/ModelProblem.pdf>.
- [39] RUMP, S. M., AND SEKIGAWA, H. The ratio between the Toeplitz and the unstructured condition number, 2006. To appear. URL: <http://www.ti3.tu-harburg.de/paper/rump/RuSe06.pdf>.
- [40] SAFEY EL DIN, M. Computing the global optimum of a multivariate polynomial over the reals. In *ISSAC 2008 Proc. 2008 Internat. Symp. Symbolic Algebraic Comput.* (New York, N. Y., 2008), D. Jeffrey, Ed., ACM Press. These Proceedings.
- [41] STURM, J. F. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software* 11/12 (1999), 625–653.
- [42] VERSHELDE, J., AND WATT, S. M., Eds. *SNC'07 Proc. 2007 Internat. Workshop on Symbolic-Numeric Comput.* (New York, N. Y., 2007), ACM Press.
- [43] VILLARD, G. Certification of the QR factor R and of lattice basis reducedness. In *ISSAC 2007 Proc. 2007 Internat. Symp. Symbolic Algebraic Comput.* (New York, N. Y., 2007), C. W. Brown, Ed., ACM Press, pp. 361–368.
- [44] WAKI, H., KIM, S., KOJIMA, M., AND MURAMATSU, M. Sums of squares and semidefinite programming relaxations for polynomial optimization problems with structured sparsity. *SIAM Journal on Optimization* 17 (2006), 218–242.
- [45] ZHANG, T., XIAO, R., AND XIA, B. Real solution isolation based on interval Krawczyk operator. In *Proc. of the Seventh Asian Symposium on Computer Mathematics* (Seoul, South Korea, 2005), S. Pae and H. Park, Eds., Korea Institute for Advanced Study, pp. 235–237. Extended abstract.
- [46] ZHI, L. Numerical optimization in hybrid symbolic-numeric computation. In Vershelde and Watt [42], pp. 33–35.
- [47] ZHOU, W., AND JEFFREY, D. J. Fraction-free matrix factors: new forms for LU and QR factors. In *Frontiers of Computer Science in China* (2008). To appear. URL <http://www.apmaths.uwo.ca/~djeffrey/Offprints/FFLUQR.pdf>.

Numerical Primary Decomposition

Anton Leykin
Institute for Mathematics and its Applications
University of Minnesota
Minneapolis, MN, USA
leykin@ima.umn.edu

ABSTRACT

Consider an ideal $I \subset R = \mathbb{C}[x_1, \dots, x_n]$ defining a complex affine variety $X \subset \mathbb{C}^n$. We describe the components associated to I by means of *numerical primary decomposition* (NPD).

The method is based on the construction of *deflation ideal* $I^{(d)}$ that defines the *deflated variety* $X^{(d)}$ in a complex space of higher dimension. For every embedded component there exists d and an isolated component $Y^{(d)}$ of $I^{(d)}$ projecting onto Y . In turn, $Y^{(d)}$ can be discovered by existing methods for prime decomposition, in particular, the *numerical irreducible decomposition*, applied to $X^{(d)}$.

The concept of NPD gives a full description of the scheme $\text{Spec}(R/I)$ by representing each component with a *witness set*. We propose an algorithm to produce a collection of witness sets that contains a NPD and that can be used to solve the *ideal membership problem* for I .

Categories and Subject Descriptors

Mathematics of Computing [G.0].

General Terms

Algorithms, Theory.

Keywords

Primary decomposition, numerical algebraic geometry, polynomial homotopy continuation, deflation

1. INTRODUCTION

Throughout the paper we use the following notation. Let I be an ideal in the polynomial ring $R = \mathbb{C}[\mathbf{x}] = \mathbb{C}[x_1, \dots, x_n]$ generated by polynomials $f_1, \dots, f_N \in R$. The ideal I defines the a variety $X = V(I)$, which, set-theoretically, is the set of points in $\mathbb{C}^n$ annihilated by all polynomials in ideal I .

The basic description of the affine scheme $\text{Spec}(R/I)$ is given by the set of *associated prime ideals* $\text{Ass}(I)$ consisting

of all ideals that appear as annihilators of elements of the R -module R/I .

There exist symbolic algorithms (see, e.g., [6, 7, 3, 5, 19, 16]) to compute an *irredundant primary decomposition* of I , which is by definition a decomposition

$$I = J_1 \cap \dots \cap J_r \quad (1.1)$$

such that all ideals J_i are primary and their radicals $I_i = \sqrt{J_i} \in \text{Ass}(I)$ are pairwise distinct.

The set of subvarieties defined by the prime ideals

$$\text{VAss}(I) = \{V(P) \mid P \in \text{Ass}(I)\}$$

is called the *components* associated to I . A component is said to be *isolated* if it is maximal with respect to inclusion and *embedded* if it is not. The existing methods of *numerical algebraic geometry* can “see” only the isolated components.

The concept of *numerical primary decomposition* (NPD) that we propose describes *all* components in terms of the (generalized) *witness sets* (see Definition 4.1). Such a witness set for a component $Y \in \text{VAss}(I)$ describes Y completely: one can sample the component, determine its degree, determine whether a given point belongs to Y , etc. Moreover, with a NPD in hand one can solve the *ideal membership problem*, i.e., given a polynomial $f \in R$ decide if $f \in I$. This is discussed in Subsection 4.2.

The idea of our method lies in the construction of the *deflation ideal* $I^{(d)}$ (see Definition 2.3) that defines the *deflated variety* $X^{(d)} = V(I^{(d)})$ of order d in a higher-dimensional space. The latter is a stratified vector bundle over X that comes with the natural projection $\pi_d : X^{(d)} \rightarrow X$ onto the base. This construction is related to a deflation technique for 0-dimensional isolated components [12, 13, 9].

The main theoretical result, Theorem 3.8, shows that an embedded component $Y \in \text{VAss}(I)$ becomes *visible* for some deflation order d , i.e., there is an isolated component $Z \in \text{VAss}(I^{(d)})$ such that $\pi_d(Z) = Y$.

Based on this theorem, we outline a straightforward algorithm, Algorithm 3.11, to compute all visible components up to the given order. The advantage of this algorithm over other known techniques is that the problem of finding all components is reduced to that of finding only isolated components without performing any saturation steps.

Algorithm 5.3 specializes this general algorithm to a numerical method for prime decomposition, namely, the *numerical irreducible decomposition* (NID) [17].

The area of *numerical algebraic geometry* comprises novel approaches to computational algebraic geometry based on

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

the numerical polynomial homotopy continuation methods. The recent book [18] may serve as a good introduction to the area.

While these methods involve computations that are approximate, a typical output of the algorithms consists of not only approximations of exact solutions, which are algebraic numbers, but also exact discrete information about the input. For instance, the main concept introduced in this paper, a *numerical primary decomposition* of an ideal provides *exact* data such as the dimensions and the degrees of the associated components. Another example is a solution to the ideal membership problem given via NPD. Although we do not provide a certification procedure for this method, such can be developed in theory, provided that results of our numerical computation can be refined to an arbitrary precision.

One advantage of hybrid numerical homotopy continuation techniques over purely symbolic methods, such as Gröbner bases, is that the former is easily parallelizable and the algorithms for computing the latter are intrinsically serial.

Due to a very small amount of data being stored on or transferred between the computational nodes and *embarassing parallelism*, one can easily achieve linear speedups for numerical homotopy continuation on a parallel computing system with any architecture. This comes in contrast with the high interdependency of tasks performed in Gröbner bases computation and the phenomena of intermediate expression swell. (Although there is no doubt that limited speedups are possible in parallel Gröbner computation, the claims of good scalability are substantiated with experiments on preselected classes of problems and methods based on non-optimal serial algorithms. See, e.g., [1] for critique of such claims.)

We live in the age when the clock speed of processors has stopped growing fast and the computational capacity of computers increases mainly through building either multicore or distributed systems. This tendency implies both great present and even better future for the algorithms of numerical algebraic geometry.

The paper is structured as follows. After the introduction (Section 1), we make the main definitions of the paper – those of deflated ideal and variety – in Section 2. Next, we overview the dual space approach to looking at the structure of a polynomial ideal; Section 3 culminates in the proof of Theorem 3.8, the main theoretical result of the paper. Section 4 proposes a new numerical representation of an ideal called *numerical primary decomposition* (NPD) via (generalized) *witness sets*. In Section 5, we give a skeleton of an NPD algorithm and examples. We discuss the future of growing meat on the skeleton in conclusion, Section 6.

Acknowledgements. I would like to thank IMA (Minneapolis, USA) for providing an exceptional working environment during the thematic year on Applications of Algebraic Geometry, as well as RISC and RICAM (Linz, Austria) for hosting me for two months during the special semester on Gröbner bases (2006). My thanks also go to the referees for their thoughtful comments.

2. DEFLATION

A variant of the matrix defined below appears in the *deflation* method for regularizing singular isolated solutions of

a polynomial system described in [13], as well as the computation of the multiplicity of an isolated point [4].

DEFINITION 2.1. *The deflation matrix of order d of an ideal I generated by polynomials $f_1, \dots, f_N \in R$ is a matrix $A_I^{(d)}$ with*

- entries in R ;
- rows indexed by $\mathbf{x}^\alpha f_j$, where $|\alpha| < d$ and $j = 1, 2, \dots, N$;
- columns indexed by partial differential operators $\partial^\beta = \frac{\partial^{|\beta|}}{\partial x_1^{\beta_1} \dots \partial x_n^{\beta_n}}$, where $|\beta| \leq d$;

and with the entry at row $\mathbf{x}^\alpha f_j$ and column ∂^β set to be

$$\partial^\beta \cdot (\mathbf{x}^\alpha f_j) = \frac{\partial^{|\beta|}(\mathbf{x}^\alpha f_j)}{\partial \mathbf{x}^\beta}. \quad (2.1)$$

If a point $\mathbf{x}$ is an isolated solution, in other words, $\{\mathbf{x}\}$ is 0-dimensional irreducible component, then its multiplicity – i.e., the vector space dimension of the ring R/I localized at $\mathbf{x}$ – equals corank $A_I^{(d)}(\mathbf{x})$ for a large enough d . This fact is shown, for example, in [4] where a method for computing the multiplicity using $A_I^{(d)}(\mathbf{x})$ is introduced.

REMARK 2.2. *A variant of the deflation matrix – the column of $A_I^{(d)}$ labeled with ∂^0 is dropped – is used in the (higher-order) deflation in [13] to construct an augmented system of equations for which the multiplicity of a given isolation solution $\mathbf{x}$ of the original system drops. For computational purposes there is no need to keep the aforementioned column, however, this paper's definition of deflation matrix makes our argument in Section 3 more compact.*

DEFINITION 2.3. *Let $I = (f_1, \dots, f_N) \subset R$ and let $\mathbf{a} = (a_\beta)$, $|\beta| \leq d$, be a vector of indeterminates.*

The ideal generated by $f_1, \dots, f_N$ and the entries of the vector $A^{(d)} \mathbf{a}^T$ in the ring $\mathbb{C}[\mathbf{x}, \mathbf{a}]$ is called the deflation ideal of I of order d and denoted by $I^{(d)}$.

The deflated variety of order d is defined as

$$X^{(d)} = V(I^{(d)}) \subset \mathbb{C}^{B(n,d)},$$

where $B(n,d) = \dim \mathbb{C}[\mathbf{x}, \mathbf{a}] = n + \binom{n+d-1}{d}$.

Given an ideal I , the variety $X^{(d)}$ is well defined in view of the following proposition, the proof of which is self-contained.

LEMMA 2.4. *For every $g \in I$, the deflated variety $Y^{(d)}$ of its hypersurface $Y = V(g)$ contains $X^{(d)} = V(I^{(d)})$.*

PROOF. Let $Q_{\mathbf{a}} = \sum a_\beta \partial^\beta \in \mathbb{C}[\mathbf{a}, \partial]$ be the linear differential operator corresponding to the vector $\mathbf{a} = (a_\beta)$ of indeterminates. It is enough to show that for every point $\mathbf{x} \in X$ and every vector $\mathbf{a}$ in the kernel of $A_f^{(d)}(\mathbf{x})$ the expression $Q_{\mathbf{a}} g$ vanishes at $\mathbf{x}$. This would imply that the fiber of $Y^{(d)} \rightarrow Y$ over $\mathbf{x}$ contains the fiber of $X^{(d)} \rightarrow X$ over $\mathbf{x}$, thus proving the statement.

Let us write $g = \sum_{\alpha, 1 \leq i \leq N} c_{i,\alpha} \mathbf{x}^\alpha f_i$, where $c_{i,\alpha} \in \mathbb{C}$. Given a point $\mathbf{x} \in \mathbb{C}^n$ we can rewrite this as two sums:

$$g = \sum_{|\alpha| \geq d, 1 \leq i \leq N} c'_{i,\alpha} (\mathbf{x} - \mathbf{x})^\alpha f_i + \sum_{|\alpha| < d, 1 \leq i \leq N} c'_{i,\alpha} \mathbf{x}^\alpha f_i.$$

Now, apply the operator $Q_{\mathbf{a}}$ to g . The first sum above vanishes at $\mathbf{x}$, since the order of vanishing of every summand is at least $d+1$. The second sum vanishes, since $\mathbf{a} \in \ker A_f^{(d)}(\mathbf{x})$. $\square$

PROPOSITION 2.5. If $I_1 \subset I_2 \subset R$, then $V(I_2^{(d)}) \subset V(I_1^{(d)})$ for all d .

In particular, the deflation variety $X^{(d)}$ does not depend on the set of generators $f_1, \dots, f_N$ of I chosen to construct $A_I^{(d)}$.

PROOF. This follows from Lemma 2.4, since for a fixed set of generators $f_1, \dots, f_N$ of I , by definition,

$$V(I^{(d)}) = V(I) \cap V((f_1)^{(d)}) \cap \dots \cap V((f_N)^{(d)}).$$

□

Moreover, a stronger Proposition 2.7 holds; its proof requires a global argument less transparent than the local argument of Lemma 2.4 and 2.5.

LEMMA 2.6. Let $Q_a = \sum_{|\beta| \leq d} a_\beta \partial^\beta$ and let $\chi_i Q_a$ is a formal derivative of Q_a with respect to ∂_i for $1 \leq i \leq n$.

Then $\chi^\alpha Q_a \cdot f_j \in I^{(d)}$ for every j and any α , where the deflation ideal $I^{(d)}$ is defined using $f_1, \dots, f_N$.

PROOF. In case $|\alpha| \geq d$, the expression $\chi^\alpha Q_a$ is a constant in ∂ , hence, $\chi^\alpha Q_a \cdot f_j \in I^{(d)}$, since $f_j \in I^{(d)}$.

Assume $|\alpha| < d$. By generalized Leibnitz rule

$$Q_a \cdot (\chi^\alpha f_j) = \sum_\gamma \frac{1}{\gamma!} \frac{\partial^{|\gamma|} \chi^\alpha}{\partial \mathbf{x}^\gamma} \chi^\gamma Q_a \cdot f_j. \quad (2.2)$$

The expression $\frac{\partial^{|\gamma|} \chi^\alpha}{\partial \mathbf{x}^\gamma}$ is zero if not $\gamma = \alpha$ or $|\gamma| < |\alpha|$. Assuming that the claim is established for degrees lower than $|\alpha|$, it follows that $\chi^\gamma Q_a \cdot f_j \in I^{(d)}$ for $|\gamma| < |\alpha|$. Since the left hand side of (2.2) belongs to $I^{(d)}$ by definition, it follows that $\chi^\alpha Q_a \cdot f_j$, the summand corresponding to $\gamma = \alpha$, belongs to $I^{(d)}$ as well. □

PROPOSITION 2.7. The deflation ideal $I^{(d)}$ does not depend on the set of generators I chosen to construct it.

PROOF. By Lemma 2.6, we have $Q_a \cdot (\chi^\alpha f_j) \in I^{(d)}$, therefore, $Q_a \cdot h \in I^{(d)}$ for every $h \in I$, since h is a linear combination of terms $\chi^\alpha f_j$ and the action of Q_a is linear.

The conclusion of the proposition follows immediately. □

EXAMPLE 2.8. Let $I = (x_1^2, x_1 x_2 x_3) \subset \mathbb{C}[x_1, x_2, x_3]$. The radical $\sqrt{I} = (x_1)$, hence, the only isolated component is $V(x_1)$.

Compute the first order ($d = 1$) deflation: multiply the deflation matrix

$$A_I^{(1)}(\mathbf{x}) = \begin{matrix} & id & \partial_1 & \partial_2 & \partial_3 \\ \begin{matrix} x_1^2 \\ x_1 x_2 x_3 \end{matrix} & \begin{bmatrix} x_1^2 & 2x_1 & 0 & 0 \\ x_1 x_2 x_3 & x_2 x_3 & x_1 x_3 & x_1 x_2 \end{bmatrix} \end{matrix}$$

by the column vector $(a_0, a_1, a_2, a_3)^T$. The deflation ideal then is

$$I^{(1)} = (x_1^2, x_1 x_2 x_3, a_1 x_1, a_1 x_2 x_3 + a_2 x_1 x_3 + a_3 x_1 x_2).$$

Observe that

$$\sqrt{I^{(1)}} = (x_1, a_1 x_2 x_3) = (x_1, a_1) \cap (x_1, x_2) \cap (x_1, x_3),$$

therefore, projecting isolated components of $X^{(1)}$ to X gives two more (embedded) components: $V(x_1, x_2)$ and $V(x_1, x_3)$.

REMARK 2.9. Denote by $\pi_d : X^{(d)} \rightarrow X$ the natural projection induced by the projection $\pi_d : \mathbb{C}^{B(n,d)} \rightarrow \mathbb{C}^n$, which maps $(\mathbf{x}, \mathbf{a}) \mapsto \mathbf{x}$.

The deflated variety $X^{(d)}$ is a stratified vector bundle over the variety X with the projection π_d onto the base. The strata of X over which $X^{(d)}$ is locally trivial are constructible algebraic subsets of X .

EXAMPLE 2.10. Let L be a plane in $\mathbb{C}^n$ of dimension $n-k$ defined by the vanishing of $I = (x_{j_i})$ for some $1 \leq j_1 \leq \dots \leq j_k \leq n$.

Then it is not hard to establish that

$$I^{(d)} = I + (\{a_\beta : |\beta| \leq d \text{ and } \beta_{j_i} \neq 0 \text{ for all } i\}).$$

Therefore, $L^{(d)}$ is a plane in $\mathbb{C}^{B(n,k)}$ of dimension $n-k + \binom{n-k+d-1}{d}$.

Every plane in $\mathbb{C}^n$ can be brought to the above form with an affine change of coordinates. This change would result in a linear transformation on the derivations ∂^β ; hence, a deflated variety of order d of a plane of codimension k is always a plane of dimension $n-k + \binom{n-k+d-1}{d}$.

3. VISIBLE COMPONENTS

In this section we describe briefly the dual space (a.k.a. inverse system) approach to the local description of a point scheme. While from the computational point of view this approach has not been exploited yet beyond the 0-dimensional case, it comes handy in the proof of the main theorem of the section and the paper.

3.1 Dual spaces

DEFINITION 3.1. For $\mathbf{x} \in \mathbb{C}^n$, let $\Delta_{\mathbf{x}}^\beta : R \rightarrow \mathbb{C}$ be a linear functional defined by

$$\Delta_{\mathbf{x}}^\beta(f) = (\partial^\beta \cdot f)(\mathbf{x}) = \frac{\partial^{|\beta|} f}{\partial \mathbf{x}^\beta}(\mathbf{x}), \quad f \in R.$$

The dual space $D_{\mathbf{x}}[I]$ of ideal I at a point $\mathbf{x}$ is the subspace of the $\mathbb{C}$ -span of functionals $\Delta_{\mathbf{x}}^\beta$ consisting of the ones that annihilate all polynomials in I .

The dual space defined as above has been introduced under other names in literature; notably it is the *local inverse system*, the term going back to Macaulay [14].

The dual space $D_{\mathbf{x}}[I]$ has a filtration

$$D_{\mathbf{x}}^{(0)}[I] \subset D_{\mathbf{x}}^{(1)}[I] \subset D_{\mathbf{x}}^{(2)}[I] \subset \dots$$

where $D_{\mathbf{x}}^{(d)}[I]$ is the set of functionals of order at most d . For computational purposes, it is convenient to think of $D_{\mathbf{x}}^{(d)}[I]$ as $\ker A_I^{(d)}(\mathbf{x})$: indeed, given a vector $\mathbf{a} \in \ker A_I^{(d)}(\mathbf{x})$, one can view it naturally as the functional $\sum_{|\beta| \leq d} a_\beta \Delta_{\mathbf{x}}^\beta$. The natural isomorphism $\ker A_I^{(d)}(\mathbf{x}) \simeq D_{\mathbf{x}}^{(d)}[I]$ could be shown then by inspection of the definition of the deflation matrix. In particular, for the case of an isolated point $\mathbf{x}$, the whole dual space $D_{\mathbf{x}}[I] \simeq \ker A_I^{(d)}(\mathbf{x})$ for $d \gg 0$, which is shown, for example, in [4, Theorem 1].

Let e_i be an n -vector with one in the i -th position and zeros in the rest. On the space $D_{\mathbf{x}}[0]$ of all differential functionals at a point $\mathbf{x}$ we define

1. operators of integration $\delta_i : D_{\mathbf{x}}^{(d)}[0] \rightarrow D_{\mathbf{x}}^{(d+1)}[0]$, $i = 1, \dots, n$, induced by the multiplication by ∂_i in the Definition 3.1, i.e.:

$$\delta_i(\Delta_{\mathbf{x}}^\beta)(f) = (\partial_i \partial^\beta \cdot f), \quad f \in R,$$

in other words,

$$\delta_i(\Delta_{\mathbf{x}}^\beta) = \Delta_{\mathbf{x}}^{\beta+e_i};$$

2. operators of differentiation $\chi_i : D_{\mathbf{x}}^{(d+1)}[0] \rightarrow D_{\mathbf{x}}^{(d)}[0]$, $i = 1, \dots, n$, induced by the multiplication by $(x_i - \mathbf{x}_i)$ in the Definition 3.1, i.e.:

$$\begin{aligned} \chi_i(\Delta_{\mathbf{x}}^\beta)(f) &= ((x_i - \mathbf{x}_i)\partial^\beta \cdot f) \\ &= ((\partial^\beta(x_i - \mathbf{x}_i) + \beta_i\partial^{\beta-e_i}) \cdot f), \end{aligned}$$

therefore, after specializing $\mathbf{x} = \mathbf{x}$,

$$\chi_i(\Delta_{\mathbf{x}}^\beta) = \beta_i\Delta_{\mathbf{x}}^{\beta-e_i}.$$

Since $\Delta_{\mathbf{x}}^\beta = \delta^\beta\Delta_{\mathbf{x}}^0$, for a fixed $\mathbf{x}$ we may look at the space of linear functionals $D_{\mathbf{x}}[0]$ as $\mathbb{C}[\delta] = \mathbb{C}[\delta_1, \dots, \delta_n]$ with differentiation operators χ_i acting as formal derivations $\frac{\partial}{\partial \delta_i}$. Note that $D_{\mathbf{x}}[I] \subset \mathbb{C}[\delta_1, \dots, \delta_n]$ is stable under this action of χ_i for every ideal $I \subset R$.

In fact, by (naturally) extending the space of linear functionals to the formal power series $\mathbb{C}[[\delta]]$ one may obtain the following fact (see, e.g., [15, Proposition 2.6]).

PROPOSITION 3.2. *The ideals of R are in one-to-one correspondence with the vector subspaces of $\mathbb{C}[[\delta]]$ stable under differentiation and closed in (δ) -adic topology.*

For our purposes, given an ideal I , we need just a local statement regarding $(R/I)_{\mathbf{x}} = (R/I)_{\mathbf{m}_{\mathbf{x}}}$, the quotient local ring at a point $\mathbf{x}$, where $\mathbf{m}_{\mathbf{x}} = (x_1 - \mathbf{x}_1, \dots, x_n - \mathbf{x}_n) \subset R$.

One may show that the subspaces of $\mathbb{C}[[\delta]]$ stable under the differentiation are in one-to-one correspondence with the ideals of the local ring $R_{\mathbf{x}}$. The following lemma proves this statement in one direction.

LEMMA 3.3. *The image of $f \in R$ is zero*

1. *in $(R/I + \mathbf{m}_{\mathbf{x}}^{d+1})_{\mathbf{x}}$ iff $Q \cdot f = 0$ for all $Q \in D_{\mathbf{x}}^{(d)}[I]$;*
2. *in $(R/I)_{\mathbf{x}}$ iff $Q \cdot f = 0$ for all $Q \in D_{\mathbf{x}}[I]$.*

PROOF. Statement (2) follows from (1).

To prove (1), notice that $D_{\mathbf{x}}[I + \mathbf{m}_{\mathbf{x}}^{d+1}]$ is exactly $D_{\mathbf{x}}^{(d)}[I]$. Indeed, every functional of the latter annihilates not only I , but also all elements of $\mathbf{m}_{\mathbf{x}}^{d+1}$; on the other hand, a functional of order larger than d does not kill the entire $\mathbf{m}_{\mathbf{x}}^{d+1}$. $\square$

REMARK 3.4. *In case of an isolated point $\mathbf{x} \in X$, it follows from the lemma that the dimensions of the $\mathbb{C}$ -spaces $D_{\mathbf{x}}[I]$ and $(R/I)_{\mathbf{x}}$ are equal. Stronger statements are available: for example, see [13, Theorem 3.1] and [15, Theorem 3.2]. The latter provides a recipe for constructing the $\mathbf{m}_{\mathbf{x}}$ -primary component of the ideal I from a basis of $D_{\mathbf{x}}[I]$.*

3.2 A visible deflation of a component

Recall the deflated variety $X^{(d)} = V(I^{(d)}) \subset \mathbb{C}^{B(n,d)}$ and the projection $\pi_d : X^{(d)} \rightarrow X$ in the Definition 2.3.

DEFINITION 3.5. *We define the deflation of order d of a component $Y \in \text{VAss}(I)$ as $Y^{(d)} = \overline{\pi_d^{-1}Y^\circ} \subset X^{(d)}$, where Y° is the subset of generic points, that is all smooth points that do not belong to other components that do not contain Y .*

PROPOSITION 3.6. *A deflation of a component is an irreducible subvariety.*

PROOF. As was mentioned in Remark 2.9 $X^{(d)}$ is a stratified bundle. The generic locus Y° is a stratum in the stratification; indeed, the bundle is locally trivial at every generic point. Since $\pi_d^{-1}Y^\circ$ is an open subset of $Y^{(d)}$, the latter is irreducible. $\square$

DEFINITION 3.7. *We say that $Y \in \text{VAss}(I)$ is visible at order d , if $Y^{(d)}$ is an isolated component of $X^{(d)}$.*

THEOREM 3.8. *Every component is visible at some order.*

PROOF. Note that an isolated component Y is visible for any $d \geq 0$, since $Y^{(d)} = \overline{\pi_d^{-1}Y^\circ}$ can not be contained in any other component in $\text{VAss}(I^{(d)})$.

Let $I = \bigcap_{Z \in \text{VAss}(I)} J_Z$ be an irredundant primary decomposition of I , where $V(J_Z) = Z$ for all Z . Fix an embedded component Y with a generic point $\mathbf{y} \in Y^\circ$ and consider $I_{>Y} = \bigcap_{Y \subsetneq Z \in \text{VAss}(I)} J_Z$. Then $I_{>Y} \supsetneq I$, moreover, the local quotient ring $(R/I_{>Y})_{\mathbf{y}}$ is a proper quotient of $(R/I)_{\mathbf{y}}$. Hence, according to Lemma 3.3 there is a proper inclusion $D_{\mathbf{y}}[I] \supsetneq D_{\mathbf{y}}[I_{>Y}]$.

For any $Z \in \text{VAss}(I)$ containing Y the fiber of its deflation $\pi_d^{-1}(\mathbf{y}) \cap Z^{(d)}$ is contained in $D_{\mathbf{y}}^{(d)}[I_{>Y}]$, since only the fibers over the generic locus Z° define $Z^{(d)}$. Now, there exists d such that

$$\pi_d^{-1}(\mathbf{y}) \cap Y^{(d)} = D_{\mathbf{y}}^{(d)}[I] \supsetneq D_{\mathbf{y}}^{(d)}[I_{>Y}] \supset \pi_d^{-1}(\mathbf{y}) \cap Z^{(d)}$$

that implies immediately that $Z^{(d)}$ does not contain $Y^{(d)}$. $\square$

REMARK 3.9. *If Y is visible at order d then it is visible at any order $d' \geq d$. Indeed, the component*

$$Y^{(d')} = \overline{\pi_{d'}^{-1}Y^\circ} = \overline{\pi_{d' \rightarrow d}^{-1}(Y^{(d)})^\circ},$$

where $\pi_{d' \rightarrow d} : X^{(d')} \rightarrow X^{(d)}$ is the natural projection, has to be isolated if $Y^{(d)}$ is isolated.

Since there is a finite number of components there is an order, at which all components are visible.

EXAMPLE 3.10. *Consider two ideals of $\mathbb{C}[x_1, x_2, x_3]$:*

$$I = (x_1^2, x_1x_2x_3)$$

$$J = (x_1^2, x_1x_2^2x_3, x_1x_2x_3^2).$$

The first order deflation of I has been computed in Example 2.8. For J we have

$$A_J^{(1)} = \begin{matrix} & \text{id} & \partial_1 & \partial_2 & \partial_3 \\ \begin{matrix} x_1^2 \\ x_1x_2^2x_3 \\ x_1x_2x_3^2 \end{matrix} & \begin{bmatrix} x_1^2 & 2x_1 & 0 & 0 \\ x_1x_2^2x_3 & x_1x_2^2x_3 & x_2^2x_3 & 2x_1x_2x_3 \\ x_1x_2x_3^2 & x_1x_2x_3^2 & x_2x_3^2 & x_1x_3^2 \end{bmatrix} \end{matrix}$$

The entries of $A_J^{(1)}(a_0, a_1, a_2, a_3)^T$ together with the original generators of J generate

$$\begin{aligned} J^{(1)} = (& x_1^2, x_1x_2^2x_3, x_1x_2x_3^2, a_1x_1, \\ & a_1x_2^2x_3 + 2a_2x_1x_2x_3 + a_3x_1x_2^2, \\ & a_1x_2x_3^2 + a_2x_1x_3^2 + 2a_3x_1x_2x_3). \end{aligned}$$

It is easy to see that

$$\sqrt{J^{(1)}} = \sqrt{I^{(1)}} = (x_1, a_1x_2x_3) = (x_1, a_1) \cap (x_1, x_2) \cap (x_1, x_3),$$

therefore, we need a higher order deflation to distinguish J from I .

One may check that the second order deflation uncovers another embedded component in $\text{VAss}(J)$, the origin, which is not a component in $\text{VAss}(J)$. However, still $\sqrt{J^{(2)}} = \sqrt{I^{(2)}}$, for I the origin is a pseudo-component!

It is not until the third deflation that we see the difference and here is why. The difference may be seen by comparing the dual spaces at the origin: $D_0[I]$ and $D_0[J]$. First of all, $D_0[I] \subset D_0[J]$, since $J \subset I$. This inclusion is proper as $\Delta_0^{(1,1,1)} \in D_0[J] \setminus D_0[I]$, a functional of order 3. On the other hand, $D_0^{(2)}[I] = D_0^{(2)}[J]$.

Here we would like to propose the most general algorithm for primary decomposition, which take the order of deflation d as a parameter and returns all components for $d \gg 0$.

ALGORITHM 3.11. $\mathcal{N} = \text{VisibleComponents}(I, d)$

Require: I , ideal of $R = \mathbb{K}[x]$ where $\mathbb{K}$ is a field of characteristic 0; $d > 0$.

Ensure: $\mathcal{C}$, the set components visible at order d .

$\mathcal{C} = \pi_d \left\{ \text{isolated components of } I^{(d)} \right\}$.

The correctness of the algorithm is assured by Theorem 3.8; although we use $\mathbb{K} = \mathbb{C}$, the argument holds for an arbitrary algebraically closed field of characteristic 0.

In fact, by passing to the algebraic closure, one can make the same conclusion for an arbitrary $\mathbb{K}$ of characteristic 0. The only serious issue in this case is that a component that is irreducible over $\mathbb{K}$ may become reducible over its algebraic closure. However, the deflation procedure preserves irreducibility and the property of algebraic closedness is exploited only locally in the proof of Theorem 3.8.

While any routine for prime decomposition can be used to compute the isolated components of the deflated variety, in what follows we concentrate on a numerical approach, which is applicable only for $\mathbb{K} = \mathbb{C}$.

4. WITNESS SETS AND NUMERICAL PRIMARY DECOMPOSITION

4.1 Generalized and classical witness sets

All numerical algorithms based on homotopy continuation boil down to the computation of approximations to points of a 0-dimensional variety. That is why for every component Y we need to invent a presentation that would consist of a finite number of points and, perhaps, some additional (finite) information.

DEFINITION 4.1. A witness set $W = W_Y$ of a component $Y \in \text{VAss}(I)$ is a triple $(d, L, w) = (d_Y, L_Y, w_Y)$ consisting of

1. an order d , such that Y is visible at order d ;
2. a generic $(\text{codim } Y^{(d)})$ -plane $L \subset \mathbb{C}^{B(n,d)}$;
3. the (finite) set of witness points $w = Y^{(d)} \cap L$;

All items can be presented with finite data: in particular, L can be represented by a linear basis. We do not include as elements of the witness set generators of I , we assume that those are fixed and available.

We also assume that there is a procedure $\mathcal{H}_{L,L'}$ that for another generic $(\text{codim } Y^{(d)})$ -plane $L' \subset \mathbb{C}^{B(n,d)}$ takes the witness points w_Y as input and produces a new set of witness points $w_{Y'}$ forming a witness set $(d, L', w_{Y'})$. In numerical algebraic geometry such a procedure is provided by a sufficiently randomized homotopy continuation that deforms L into L' without encountering an intermediate plane that is singular with respect to $Y^{(d)}$ and a numerical routine that tracks the paths starting at the witness points w_Y .

We would like to remark that, in principle, it is enough to store only one witness point, since the rest can be obtained due to the action of the *monodromy group*, which is transitive on w_Y . In practice, this can be done by following a random homotopy cycle $\mathcal{H}_{L,L}$ a finite number of times. However, it is, of course, more practical to store the whole set w_Y .

REMARK 4.2. An isolated component is visible at order $d = 0$. In this case, the plane L and the set w give what we would call a classical witness set, a concept which is used, for example, throughout [18] where a generalization of it is made (page 237) in relation to the deflation of an isolated (but multiple) component.

EXAMPLE 4.3. The components of $I = (x_3^2, x_3(x_2 + x_1^2)) \subset \mathbb{C}[x_1, x_2, x_3]$ are $Z = V(x_3)$ and $Y = V(x_3, x_2 + x_1^2)$.

The isolated component Z can be presented by the witness set using any line L that is not parallel to Z and such that $Z \cap L \not\subseteq Y$.

To represent Y , we have to look at the first order deflation:

$$\begin{aligned} I^{(1)} &= (x_3^2, x_3(x_2 + x_1^2), a_3x_3, \\ &\quad 2a_1x_1x_3 + a_2x_3 + a_3(x_2 + x_1^2)) \\ &\subset \mathbb{C}[x_1, x_2, x_3, a_0, a_1, a_2, a_3] \\ X^{(1)} &= V(x_3, a_3(x_2 + x_1^2)) \subset \mathbb{C}^7 \end{aligned}$$

The deflated variety $X^{(1)}$ is 5-dimensional; we take the following 5 equations for L :

$$\begin{aligned} x_2 &= -3x_1 + 2; \\ a_3 &= x_1 - 3; \\ a_i &= c_i x_1 + d_i, \quad c_i, d_i \in \mathbb{C}, \quad i = 0, 1, 2. \end{aligned}$$

The first two equations together with the second defining equation of $X^{(1)}$ give

$$(x_1 - 3)(x_1 - 2)(x_1 - 1) = 0.$$

The set $\pi_1(X^{(1)} \cap L) = \{(3, -7, 0), (2, -4, 0), (1, -1, 0)\}$ contains projections of two subsets of witness points,

$$\begin{aligned} \pi_1(w_Z) &= \{(3, -7, 0)\}, \\ \pi_1(w_Y) &= \{(2, -4, 0), (1, -1, 0)\}, \end{aligned}$$

of the witness sets of the first order $W_Z = (1, L, w_Z)$ and $W_Y = (1, L, w_Y)$, respectively.

REMARK 4.4. For a witness point $(\mathbf{y}, \mathbf{a}) \in w_Y$, the vector $\mathbf{a}$ translates into a functional $Q_{\mathbf{a}} \in D_{\mathbf{y}}^{(d)}[I]$. For a fixed generic $\mathbf{y} \in Y$ the set $\{Q_{\mathbf{a}} \mid (\mathbf{y}, \mathbf{a}) \in Y^{(d)}\}$ equals the dual space $D_{\mathbf{y}}^{(d)}[I]$. Therefore, in practice, we can compute $D_{\mathbf{y}}^{(d)}[I]$ from the witness set (d, L, w) of Y by tracking a homotopy that creates another witness set (d, L_i, w_i) , where the plane L_i is a random plane such that $\pi_d(L) \cap Y \ni \mathbf{y}$.

If the procedure is carried out for sufficiently many L_i , then the functionals $\{Q_a \mid (\mathbf{y}, \mathbf{a}) \in w_i\}$ span $D_{\mathbf{y}}^{(d)}[I]$.

4.2 Numerical primary decomposition and the ideal membership problem

DEFINITION 4.5. A collection of witness sets is called a numerical primary decomposition (NPD) of I if it contains precisely one witness set for each component in $\text{VAss}(I)$.

NPD contains exhaustive information about the ideal I and, in particular, the scheme $\text{Spec}(R/I)$ due to the possibility of solving the *ideal membership problem*. This is so, since for every Y the projection $\pi_{d_Y}(\mathbf{z})$ of a witness point $\mathbf{z} \in w_Y$ gives a generic point of Y and in the view of the following:

THEOREM 4.6. A polynomial $g \in R$ is contained in the ideal I iff for all $Y \in \text{VAss}(I)$ and every (any) generic point $y \in Y$, all functionals in the dual space $D_{\mathbf{y}}^{(\deg_Y g)}[I]$ annihilate g .

PROOF. A polynomial $g \in I$ iff its image in R/I is zero or, equivalently, for all $\mathbf{x} \in X = V(I)$ its image in $(R/I)_{\mathbf{x}}$ vanishes. It suffices to check the latter statement for one generic point per component for all components. $\square$

It follows that, in practice, we can solve the ideal membership problem by checking the condition in the theorem at a finite number of points. Therefore, we have the following algorithm.

ALGORITHM 4.7. $b = \text{IMP}(g, I, \mathcal{N})$

Require: I , ideal of R represented by a finite set of generators; $\mathcal{N}$, a NPD of I .

Ensure: $b = "g \in I"$, a boolean value.

Let $d = \deg g$ and $g = \sum_{|\beta| \leq d} c_{\beta} \mathbf{x}^{\beta}$;

for all $(d', L, w) \in \mathcal{N}$ **do**

 Pick $\mathbf{x} \in \pi_{d'}(w)$;

 Compute a linear basis K of $\ker A_I^{(d)}(\mathbf{x})$;

if $Q \cdot f \neq 0$ (*) for some $Q \in D_{\mathbf{x}}^{(d)}[I]$ corresponding to an element of K **then**

 Return **false**;

end if

end for

Return **true**.

According to Theorem 4.6 $f \in I$ iff there is no $x \in X$, for which the condition (*) holds. In fact, it suffices to check (*) for a set of generic points (one per component).

REMARK 4.8. In case the above algorithm is executed numerically, i.e., only approximations of points on the components are generated, we would like to point out several practical issues.

First, having the whole NPD it easy to generate other points on any given component and recheck the condition (*) at as many points as desired, therefore, lowering the probability of this algorithm returning an incorrect result due to picking a non-generic point.

Second, since the approximations of generic points can be refined to an arbitrary precision, the condition (*) can be effectively checked by “zooming in” on the exact point for which

the computation is carried out. However, a rigorous certification procedure has to be developed in the future in the spirit of the alpha-test for an approximate zero of a univariate polynomial.

Algorithm 4.7 is practical only for polynomials of low degrees due to the high complexity of construction of the deflation matrix of order d and computing its kernel. However, we are confident that improvements can be made as this matrix is highly structured and the fact that it is enough to check the condition (*) for only one generic element in the kernel.

5. ALGORITHM FOR NUMERICAL PRIMARY DECOMPOSITION

In the description of our NPD algorithm, we assume the following subroutines are at our disposal:

ALGORITHM 5.1. $\mathcal{M} = \text{NID}(I)$

Require: I , ideal of R .

Ensure: $\mathcal{M}$, the set of classical witness sets (L_Y, w_Y) for all isolated components $Y \in \text{VAss}(I)$.

There are two approaches to NID in the numerical algebraic geometry: the “top-down” method described in detail in [18, Chapter 15] and “equation-by-equation” method the philosophy of which is outlined in [18, §16.2].

ALGORITHM 5.2. $b = \text{IsInComponent}(\mathbf{y}, W_Z)$

Require: $\mathbf{y} \in \mathbb{C}^n$; $W_Z = (d, L, w)$ a witness set for $Z \in \text{VAss}(I)$.

Ensure: $b = "y \in Z"$, a boolean value.

Pick a generic $(\text{codim } Z)$ -plane $M \subset \mathbb{C}^n$ and a $(\dim L)$ -plane $L' \subset \mathbb{C}^{B(n,d)}$ such that $\pi_d(L') = M \ni \mathbf{y}$.

Use a procedure similar to the usual containment test routine [18, §15.1], i.e., track the points w along a generic homotopy $\mathcal{H}_{L,L'}$.

Return “ $\mathbf{y} \in \pi_d(\mathcal{H}_{L,L'}(w))$ ”.

There are also two subroutines, for which finding efficient algorithms is an open problem:

- $\text{StopCriterion}(d, I, \mathcal{N})$ implements a termination criterion that guarantees that all components in $\text{VAss}(I)$ are visible at order less than d .
- $\text{IsComponent}((d, L, w), I, \mathcal{N})$ is used to filter out witness sets (d, L, w) that represent *false components* that appear due to singularities. As a parameter it takes partial NPD $\mathcal{N}$ that includes the witness sets for all components visible at order d of dimensions higher than the dimension of the alleged component.

Both routines will be discussed later in this section.

Now we are ready to outline the main algorithm of this paper.

ALGORITHM 5.3. $\mathcal{N} = \text{NPD}(I)$

Require: I , ideal of R .

Ensure: $\mathcal{N}$, the set of witness sets for all $Y \in \text{VAss}(I)$ visible at order d .

$\mathcal{N} = \emptyset$;

```

repeat
   $\mathcal{N} = \{(L, w) \in \text{NID}(I^{(d)}) :$ 
    not IsInComponent( $\pi_d(w_1), W$ ) for all  $W \in \mathcal{C}\}$ ;
  for all  $(L, w) \in \mathcal{N}$  in an order
    of decreasing  $\dim Y_{(L, w)}$  do
    if IsComponent( $(d, L, w), I, \mathcal{N}$ ) then
       $\mathcal{N} = \mathcal{N} \cup \{(d, L, w)\}$ ;
    end if
  end for
   $d = d + 1$ ;
until StopCriterium( $d, I, \mathcal{N}$ )

```

The order of deflation sufficient to discover all components can be bound by the maximum of the regularities of the (local) Hilbert functions at the points of X . For the latter a crude bound exists, which is doubly exponential in the number of variables. Obviously, one can not use StopCriterium based on this bound for practical purposes.

In reality, for many nontrivial examples *all* embedded components are discovered by the deflation of order as low as 1. While finding a reasonable termination criterion is the matter of the future, currently it makes sense to run a *truncated* computation with

$$\text{StopCriterium}(d, I, \mathcal{N}) = "d > d_{\max}",$$

where $d_{\max}$ is the maximal deflation order considered.

As to IsComponent, first of all, we remark that elimination of fake components from $\mathcal{N}$ may also be done at the end of the algorithm. There is a way to do this by checking whether for large enough d all functionals in the $D_y^{(d)}[I]$ for a generic y in an alleged component Y “come” from the components that contain Y . We do not describe the procedure here as it is quite technical and not practical at the moment, since the question how large d should be relates to the question of finding a good StopCriterium.

REMARK 5.4. *Establishing an efficient StopCriterium has a higher priority (over IsComponent), since $\mathcal{N}$ containing additional witness sets of fake components can be used instead of a true NPD for many tasks. In particular, our IMP routine (Algorithm 4.7) would still work.*

Below is an example of a “truncated computation”, where only the first order deflation is computed.

EXAMPLE 5.5. *Consider the cyclic 4-roots problem:*

$$I = (x_1 + x_2 + x_3 + x_4, x_1x_2 + x_2x_3 + x_3x_4 + x_4x_1, x_1x_2x_3 + x_2x_3x_4 + x_3x_4x_1 + x_4x_1x_2, x_1x_2x_3x_4 - 1).$$

The calculation of associated primes via symbolic software (we used Macaulay 2 [8]) gives:

$$\text{Ass}(I) = \left\{ \begin{array}{l} (x_2 + x_4, x_1 + x_3, x_3x_4 + 1), \\ (x_2 + x_4, x_1 + x_3, x_3x_4 - 1), \\ (x_4 - 1, x_3 + 1, x_2 + 1, x_1 - 1), \\ (x_4 - 1, x_3 - 1, x_2 + 1, x_1 + 1), \\ (x_4 + 1, x_3 + 1, x_2 - 1, x_1 - 1), \\ (x_4 + 1, x_3 - 1, x_2 - 1, x_1 + 1), \\ (x_3 + x_4, x_2 + x_4, x_1 - x_4, x_4^2 + 1), \\ (x_3 - x_4, x_2 + x_4, x_1 + x_4, x_4^2 + 1) \end{array} \right\}$$

The first two ideals correspond to the irreducible curves that are the two isolated components. The rest are embedded 0-dimensional components; note that the last two ideals are irreducible over the ground field $\mathbb{Q}$, but not $\mathbb{C}$.

Over complex numbers, there are 8 embedded components that are all visible at order 1; The numerical computation of the irreducible components of the first deflation finds all components. An excerpt from the numerical output is given below: we list the projections of the witness points for the components $\text{VAss}(I^{(1)})$.

```

>>> projections of witness points for
>>> component #1:
[x1 = -4.4882+2.0260*I, x2 = .18509+.83550e-1*I,
x3 = 4.4882-2.0260*I, x4 = -.18509-.83550e-1*I]
[x1 = .52885e-1-.87608*I, x2 = -.68654e-1-1.1373*I,
x3 = -.52885e-1+.87608*I, x4 = .68654e-1+1.1373*I]
[x1 = -.15083+.49191*I, x2 = .56975+1.8582*I,
x3 = .15083-.49191*I, x4 = -.56975-1.8582*I]
[x1 = .41488+.24720*I, x2 = -1.7788+1.0599*I,
x3 = -.41488-.24720*I, x4 = 1.7788-1.0599*I]
>>> component #2:
[x1 = -.95775+.36799*I, x2 = -.90980-.34957*I,
x3 = .95775-.36799*I, x4 = .90980+.34957*I]
[x1 = .71538+.12328*I, x2 = 1.3576-.23395*I,
x3 = -.71538-.12328*I, x4 = -1.3576+.23395*I]
[x1 = -3.7686+1.7072*I, x2 = -.22017-.99738e-1*I,
x3 = 3.7686-1.7072*I, x4 = .22017+.99738e-1*I]
[x1 = -.16036-.30943*I, x2 = -1.3202+2.5476*I,
x3 = .16036+.30943*I, x4 = 1.3202-2.5476*I]
>>> component #3:
[x1 = -1.0-.53734e-17*I, x2 = 1.0-.20045e-16*I,
x3 = 1.0+.89149e-17*I, x4 = -1.0+.18026e-17*I]
...
>>> component #10:
[x1 = -.59351e-17+1.0*I, x2 = -.46995e-16+1.0*I,
x3 = .16158e-16-1.0*I, x4 = .22439e-16-1.0*I]

```

Note that there are 2 witness sets of 4 points corresponding to the 2 isolated curves and 8 singletons for the embedded points.

See the webpage [10] for the scripts in *Macaulay 2* and *Maple* (using *PHCmaple* package [11]) that perform prime decomposition and numerical irreducible decomposition, respectively, for the first deflation ideal in this example.

6. DISCUSSION AND CONCLUSION

We consider this paper as one laying a theoretical foundation to the method that at this point works only on small examples. However, it is our believe that this technique would be able to solve problems unsolvable by purely symbolic methods in the future. The improvements are expected to be made both in the software and in the theory.

We remark that the software in the area of numerical algebraic geometry is as young as the area itself. For the purposes of numerical irreducible decomposition there exist only two software options: *PHCpack* [20] that we use via *PCHmaple* [11] and *Bertini* [2]. The practical computation using the ideas in this paper is limited by the capabilities of these software systems; we expect the implementations of numerical irreducible decomposition algorithms to improve. Both *PHCpack* and *Bertini* move towards the throughout parallelization; as we argued in the introduction, easy parallelization is a crucial feature of numerical methods that distinguishes them from the symbolic ones.

The future theoretical work should, in particular, concentrate on the construction of special homotopy methods to

tackle deflated ideals (systems), which possess an obvious multihomogeneous structure: they are linear in the additional variables. Also, while the global algorithms such as NID have been well established, there are still no efficient local procedures, e.g., for determining the local dimension at a given point on a variety. The same can be said about the local dual space computation: while it is possible to compute the truncation at some degree using the deflation matrix, an efficient description of the whole (possibly infinite-dimensional) dual space and ways to create such description are yet to be found.

7. REFERENCES

- [1] G. Attardi and C. Traverso. Strategy-accurate parallel Buchberger algorithms. *J. Symbolic Comput.*, 21(4-6):411–425, 1996.
- [2] D. J. Bates, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler. Bertini: software for numerical algebraic geometry. Available at <http://www.nd.edu/~sommese/bertini>.
- [3] M. Caboara, P. Conti, and C. Traverso. Yet another ideal decomposition algorithm. In *Applied algebra, algebraic algorithms and error-correcting codes (Toulouse, 1997)*, volume 1255 of *Lecture Notes in Comput. Sci.*, pages 39–54. Springer, Berlin, 1997.
- [4] B. Dayton and Z. Zeng. Computing the multiplicity structure in solving polynomial systems. In M. Kauers, editor, *Proceedings of the 2005 International Symposium on Symbolic and Algebraic Computation*, pages 116–123. ACM, 2005.
- [5] W. Decker, G.-M. Greuel, and G. Pfister. Primary decomposition: algorithms and comparisons. In *Algorithmic algebra and number theory (Heidelberg, 1997)*, pages 187–220. Springer, Berlin, 1999.
- [6] D. Eisenbud, C. Huneke, and W. Vasconcelos. Direct methods for primary decomposition. *Invent. Math.*, 110(2):207–235, 1992.
- [7] P. Gianni, B. Trager, and G. Zacharias. Gröbner bases and primary decomposition of polynomial ideals. *J. Symbolic Comput.*, 6(2-3):149–167, 1988. Computational aspects of commutative algebra.
- [8] D. R. Grayson and M. E. Stillman. Macaulay 2, a software system for research in algebraic geometry. Available at <http://www.math.uiuc.edu/Macaulay2/>.
- [9] G. Lecerf. Quadratic Newton iteration for systems with multiplicity. *Found. Comput. Math.*, 2:247–293, 2002.
- [10] A. Leykin. Numerical primary decomposition (webpage). www.math.umn.edu/~leykin/NPD.
- [11] A. Leykin. PHCmaple: A Maple interface to the numerical homotopy algorithms in PHCpack. www.math.umn.edu/~leykin/PHCmaple.
- [12] A. Leykin, J. Verschelde, and A. Zhao. Newton’s method with deflation for isolated singularities of polynomial systems. *Theoretical Computer Science*, 359(1-3):111–122, 2006.
- [13] A. Leykin, J. Verschelde, and A. Zhao. Higher-order deflation for polynomial systems with isolated singular solutions. In A. Dickstein, F.-O. Schreyer, and A. J. Sommese, editors, *Algorithms in Algebraic Geometry*, volume 146 of *The IMA Volumes in Mathematics and its Applications*. Springer, 2008.
- [14] F. S. Macaulay. *The algebraic theory of modular systems*. Cambridge Mathematical Library. Cambridge University Press, Cambridge, 1994. Revised reprint of the 1916 original, With an introduction by Paul Roberts.
- [15] B. Mourrain. Isolated points, duality and residues. *J. Pure Appl. Algebra*, 117/118:469–493, 1997. Algorithms for algebra (Eindhoven, 1996).
- [16] T. Shimoyama and K. Yokoyama. Localization and primary decomposition of polynomial ideals. *J. Symbolic Comput.*, 22(3):247–277, 1996.
- [17] A. Sommese, J. Verschelde, and C. Wampler. Numerical decomposition of the solution sets of polynomial systems into irreducible components. *SIAM J. Numer. Anal.*, 38(6):2022–2046, 2001.
- [18] A. J. Sommese and C. W. Wampler, II. *The numerical solution of systems of polynomials*. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2005.
- [19] A. Steel. Conquering inseparability: primary decomposition and multivariate factorization over algebraic function fields of positive characteristic. *J. Symbolic Comput.*, 40(3):1053–1075, 2005.
- [20] J. Verschelde. Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Trans. Math. Softw.*, 25(2):251–276, 1999. Software available at <http://www.math.uic.edu/~jan>.

Computational D-module Theory with SINGULAR, Comparison with Other Systems and Two New Algorithms

Viktor Levandovskyy
Lehrstuhl D für Mathematik, RWTH Aachen
Templergraben 64, 52062 Aachen, Germany
Viktor.Levandovskyy@math.rwth-
aachen.de

Jorge Martín Morales^{*}
Department of Mathematics.
University of Zaragoza
C/ Pedro Cerbuna, 12 - 50009, Zaragoza, Spain
jorge@unizar.es

ABSTRACT

We present the new implementation of core functions for the computational D -module theory. It is realized as a library `dmod.lib` in the computer algebra system SINGULAR. We show both theoretical advances, such as the LOT and `check-Root` algorithms as well as the comparison of our implementation with other packages for D -modules in computer algebra systems KAN/SM1, ASIR and MACAULAY2. The comparison indicates, that our implementation is among the fastest ones. With our package we are able to solve several challenges in D -module theory and we demonstrate the answers to these problems.

Categories and Subject Descriptors

G.4 [Mathematical Software]: Algorithm design and analysis; H.4 [Information Systems Applications]: Miscellaneous; I.1 [Computing Methodologies]: Symbolic and Algebraic Manipulation

General Terms

Algorithms, Design, Theory

Keywords

D -modules, Bernstein-Sato polynomial, annihilator, Groebner bases, non-commutative Groebner bases, intersection with subalgebra, preimage of ideal

1. INTRODUCTION

Through the article we assume $\mathbb{K}$ to be a field of characteristic 0. By R_n we denote the ring of polynomials $\mathbb{K}[x_1, \dots, x_n]$

^{*}Partially supported by the projects MTM2007-67908-C02-01 and FQM-333 of Spain and by the DFG Graduiertenkolleg "Hierarchie und Symmetrie in mathematischen Modellen" at RWTH Aachen, Germany.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

in n variables over $\mathbb{K}$ and by D_n we denote the ring of $\mathbb{K}$ -linear partial differential operators with coefficients in R_n , that is the n -th Weyl algebra [17]. The ring D_n is the associative $\mathbb{K}$ -algebra generated by the partial derivative operators ∂_i and the multiplication operators x_i subject to relations

$$\{\partial_i x_j = x_j \partial_i + \delta_{ij}, x_j x_i = x_i x_j, \partial_j \partial_i = \partial_i \partial_j \mid 1 \leq i, j \leq n\}.$$

That is, the only non-commuting pairs of variables are (x_i, ∂_i) ; they satisfy the relation $\partial_i x_i = x_i \partial_i + 1$. We use the Lie bracket notation $[a, b] := ab - ba$ for operators a, b , then e.g. the latter relation can be written as $[\partial_i, x_i] = 1$.

Finally, we denote by $D_n[s]$ the ring of polynomials in one variable s with coefficients in the n -th Weyl algebra, i.e. $D_n[s] = D_n \otimes_{\mathbb{K}} \mathbb{K}[s]$.

Let us recall Bernstein's construction. Given a non-zero polynomial $f \in R_n$ in n variables, we consider $M = R_n[s, \frac{1}{f}] \cdot f^s$ which is by definition the free $R_n[s, \frac{1}{f}]$ -module of rank one generated by the formal symbol f^s . Then M has a natural structure of left $D_n[s]$ -module. Here the differential operators act in a natural way,

$$\partial_i(g(s, x) \cdot f^s) = \left(\frac{\partial g}{\partial x_i} + sg(s, x) \frac{\partial f}{\partial x_i} \frac{1}{f} \right) \cdot f^s \in M \quad (1.1)$$

THEOREM 1 (BERNSTEIN [3]). *For every polynomial $f \in R_n$ there exists a non-zero polynomial $b(s) \in \mathbb{K}[s]$ and a differential operator $P(s) \in D_n[s]$ such that*

$$P(s)f \cdot f^s = b(s) \cdot f^s \in R_n[s, \frac{1}{f}] \cdot f^s = M. \quad (1.2)$$

One is interested in the commutative ring $R_n[\frac{1}{f}]$ which is also a left D_n -module with the usual partial derivative operators on $R_n[\frac{1}{f}]$. Actually, $R_n[\frac{1}{f}]$ is a submodule of M as a left D_n -module with the action given in (1.1). Constructively, one needs to find a left ideal J_f in D_n such that $R_n[\frac{1}{f}]$ is isomorphic to D_n/J_f as a left D_n -module.

For the following definition, given $\alpha \in \mathbb{K}$, we also need to consider $N = R_n[\frac{1}{f}] \cdot f^\alpha$ which is by definition the free $R_n[\frac{1}{f}]$ -module of rank one generated by the formal symbol f^α . Then N has a natural structure of left D_n -module coming from substituting s by α in the $D_n[s]$ -module M .

Definition 1. Using the above notation, for a given non-zero polynomial $f \in R_n$, we define the following data:

1. The Bernstein-Sato polynomial of f (also called the global b -function of f), denoted by $b_f(s)$, is defined

to be the monic polynomial of the lowest degree in s satisfying the equation (1.2).

2. An operator $P(s) \in D_n[s]$, satisfying the equation (1.2) for $b(s) = b_f(s)$ is called a *b-operator associated with f* .
3. We denote by $\text{Ann}_{D_n[s]}(f^s)$ the left ideal of all elements $P(s) \in D_n[s]$ such that $P(s) \cdot f^s = 0$ and we call it *the annihilator of f^s in $D_n[s]$* .
4. We denote by s_0 *the minimal integral root of the b-function of f* .
5. Given $\alpha \in \mathbb{K}$, we denote by $\text{Ann}_{D_n}(f^\alpha)$ the left ideal in D_n of all elements $P \in D_n$ such that $P \cdot f^\alpha = 0$ and we call it *the annihilator of f^α in D_n* .

Following Kashiwara, $R_n[\frac{1}{f}]$ is a holonomic left D_n -module, generated by f^{s_0} as left D_n -module, where s_0 is the minimal integral root of $b_f(s)$. Thus, this left module is isomorphic to $D_n / \text{Ann}_{D_n}(f^{s_0})$. The annihilator of f^{s_0} in D_n can be computed just by substituting s with s_0 in the left ideal $\text{Ann}_{D_n[s]}(f^s)$.

It should be noted, that in general, the annihilator of f^α in D_n even for $\alpha \in \mathbb{Z}$ cannot be computed just by substituting s with α in the left ideal $\text{Ann}_{D_n[s]}(f^s)$. There is an algorithm for the computation of $\text{Ann}_{D_n[s]}(f^\alpha)$ [23], which requires the knowledge of $\text{Ann}_{D_n[s]}(f^s)$.

We provide the following implementations for computing the data from the Def. 1 in the SINGULAR library `dmod.lib` [14] for algebraic D -modules:

- The algorithm of Oaku and Takayama [21, 23]
- Levandovskyy's modification of the Oaku-Takayama algorithm (a new development, see Section 2.3 below)
- The algorithm of Briançon-Maisonobe [4]

We refer to these methods as to OT, LOT and BM methods respectively. Let us recall these methods in some more details.

Let D_n be the n -th Weyl algebra, generated by $\{x_i, \partial_i\}$ as it has been described before.

Step 1. Compute the annihilator of f^s in $D_n[s]$

OT. Take $D_n\langle t, \partial_t \rangle[u, v] := D_n\langle t, \partial_t \mid \partial_t t = t\partial_t + 1 \rangle \otimes_{\mathbb{K}} \mathbb{K}[u, v]$ and the following left ideal in $D_n\langle t, \partial_t \rangle[u, v]$:

$$I = \left\langle \{t - uf, \partial_i + u \frac{\partial f}{\partial x_i} \partial_t, uv - 1\}, i = 1, \dots, n \right\rangle$$

Then $\text{Ann}_{D_n[s]}(f^s) = I \cap D_n[-\partial_t t]$ and hence the latter can be obtained by computing a left Gröbner basis of I with respect to an elimination ordering for u, v first, removing the entries containing u, v . Then one performs certain substitutions and, moreover, substitutes every appearance of the monomial $t\partial_t$ by $-s - 1$.

LOT. Let $D_n\langle t, \partial_t, s \rangle$ be the $\mathbb{K}$ -algebra, obtained by adjoining a new variable s to the $(n+1)$ -th Weyl algebra $D_n\langle t, \partial_t \rangle$ subject to relations $[s, t] = -t$ and $[s, \partial_t] = \partial_t$. Consider the following left ideal in $D_n\langle t, \partial_t, s \rangle$:

$$I = \left\langle \{t - f, \partial_i + \frac{\partial f}{\partial x_i} \partial_t, s + \partial_t t\}, i = 1, \dots, n \right\rangle$$

Then $\text{Ann}_{D_n[s]}(f^s) = I \cap D_n[s]$ and therefore we need to compute a Gröbner basis of I with respect to an elimination ordering for t, ∂_t .

BM. Let $D_n\langle \partial_t, s \rangle$ be the extension of the algebra $D_n[\partial_t]$ with the new variable s subject to the non-commutative relation $[s, \partial_t] = \partial_t$. Now, consider the left ideal

$$I = \left\langle \{s + f\partial_t, \partial_i + \frac{\partial f}{\partial x_i} \partial_t\}, i = 1, \dots, n \right\rangle \subseteq D_n\langle \partial_t, s \rangle.$$

Then $\text{Ann}_{D_n[s]}(f^s) = I \cap D_n[s]$ and thus a left Gröbner basis with respect to an elimination ordering for ∂_t will give us the answer.

Step 2. As soon as there are $\{P_1(s), \dots, P_k(s)\} \subset D_n[s]$, which generate the annihilator of f^s , the Bernstein-Sato polynomial can be obtained by means of another elimination.

$$\langle b_f(s) \rangle = (\text{Ann}_{D_n[s]}(f^s) + \langle f \rangle) \cap \mathbb{K}[s]. \quad (1.3)$$

Hence, we need to compute a left Gröbner basis of $\{P_1(s), \dots, P_k(s), f\}$ with respect to an elimination ordering for the variables x_i, ∂_i .

Step 3. Factorize $b_f(s) \in \mathbb{Q}[s]$ and determine its minimal integral root s_0 .

Step 4. Substitute s_0 in $\text{Ann}_{D_n[s]}(f^s)$ and thus obtain the generators $\{P_1(s_0), \dots, P_k(s_0)\}$ of $\text{Ann}_{D_n}(f^{s_0})$.

We should note, that challenging problems appear in the steps 1 and 2. It is known, that the elimination in the step 2 is very hard in general, while computations in the step 1 are somewhat easier. In what follows, we compare the performance of computer algebra packages on steps 1 and 2.

It should be observed that we can treat the case when $\mathbb{K}$ is any field of characteristic 0, since, due to Kashiwara [12], the rationality of the roots of the Bernstein-Sato polynomial is true over any such field. The crucial fact is that if $E \subset \mathbb{K}$ is a subfield containing all the coefficients of f , then the coefficients of $b_f(s)$ computed over $\mathbb{K}$ belong to E . Now note that one can embed E into $\mathbb{C}$ and thus the Bernstein-Sato polynomial of f over E is the same as over $\mathbb{C}$, (see [16, Prop. 2.11] for instance). However, in practice and in examples, we will assume that the base field $\mathbb{K}$ is the field of rational numbers $\mathbb{Q}$.

Non-commutative abilities and Gröbner bases in a wide class on non-commutative algebras, available in SINGULAR:PLURAL, [9] (since 2005 this extension has become an integral part of SINGULAR [11]), attracted D -module theorists already at the early stages of its development. Experimental versions of SINGULAR:PLURAL have been used in e.g. [6, 7]. Our work towards the library of procedures for D -modules started in 2004.

2. INGREDIENTS

2.1 Elimination Orderings

In computing elimination, that is an intersection of a left ideal with the subalgebra in our context, the choice of elimination ordering is even more important than in the commutative case. As it is described in detail in [13], working in

non-commutative G -algebras imposes constraints both on the ordering and on the subalgebra one works with. The situation, when no admissible elimination ordering exists, is by no means artificial [13]. However, in the realm of computational D -module theory we have less problems, compared with the general situation. It results in the existence of a big family of elimination orderings. Our experience showed, that generically best results are achieved, when one applies a possibly nested block elimination orderings with extra weight vectors. In the language of SINGULAR system such orderings are described as $(\mathbf{a}(\mathbf{w}), \mathbf{a}(\mathbf{v}), \mathbf{ord})$ (see [10] for more details), where $\mathbf{v}, \mathbf{w}$ are integral vectors and $\mathbf{ord}$ stays for a degree well-ordering, such as the degree reverse lexicographical one. For some particular cases, the pure lexicographical ordering scores very good results. However, we cannot explain this phenomenon yet. It would be interesting to apply the ideas of Tran (produce particular ideal-dependent elimination orderings, [25]) to Malgrange ideals in D -module theory.

2.2 Slim Gröbner bases

Slimgb [5] is a variant of the Buchberger's algorithm implemented in SINGULAR by M. Brickenstein. The algorithm is designed to keep polynomials *slim*, that is short with small coefficients. It features parallel reductions and a strategy to minimize the (possibly weighted) lengths of polynomials. The weighted length of the polynomials can be seen as measure for the intermediate expression swell. Weighted length functions can consider not only the number of terms, but also their coefficients and degrees. Considering the degrees of the terms inside the polynomials, **slimgb** can often directly (that is, without using Gröbner Walk or similar algorithms) compute Gröbner bases with respect to e.g. elimination orderings. The principles of **slimgb** apply to both commutative and non-commutative cases. In particular, **slimgb** shows very good performance on examples from the realm of D -modules, which heavily uses elimination. In most of non-commutative examples, **slimgb** outperforms the standard **std** routine for Gröbner bases in SINGULAR. The Hilbert-driven Gröbner basis algorithm [10], known for its strength in the commutative case has not yet been implemented in SINGULAR. Even more, we are not aware of any generalization of this algorithm to non-commutative G -algebras and any other implementation. It would be very interesting to compare the performance of the latter with **slimgb**.

2.3 The LOT Algorithm

Consider the so-called *Malgrange ideal*, associated to f in the algebra $D_n \langle t, \partial_t \mid \partial_t t = t \partial_t + 1 \rangle$:

$$I_f = \left\langle \{ t - f, \partial_i + \frac{\partial f}{\partial x_i} \partial_t \}, i = 1, \dots, n \right\rangle.$$

We want to compute $\text{Ann}_{D_n[s]}(f^s) = I_f \cap D_n[-\partial_t t]$. Consider the morphism, called *algebraic Mellin transform* [23]

$$\phi : D_n[s] = D_n \otimes_{\mathbb{K}} \mathbb{K}[s] \rightarrow D_n \otimes_{\mathbb{K}} \mathbb{K} \langle t, \partial_t \mid \partial_t t = t \partial_t + 1 \rangle,$$

where ϕ is the identity mapping on D_n and $\phi(s) = -\partial_t t = -t \partial_t - 1$. By aiming at $I_f \cap D_n[-\partial_t t]$ we are looking for the preimage $\phi^{-1}(I_f) \subset D_n[s]$ of the left ideal I_f indeed.

Recall the algorithm for computing the preimage of a left ideal under a homomorphism of G -algebras from [13].

Definition 2. Let $\mathbb{K}$ be a field and A be a quotient of the free associative algebra $\mathbb{K} \langle x_1, \dots, x_n \rangle$ by the two-sided ideal I , generated by the finite set $\{x_j x_i - x_i x_j - d_{ij} \mid \forall 1 \leq i < j \leq n, \text{ where } d_{ij} \in \mathbb{K}[x_1, \dots, x_n]\}$. A is called a *G -algebra of Lie type* [15], if

- $\forall 1 \leq i < j < k \leq n$ the expression $d_{ij} x_k - x_k d_{ij} + x_j d_{ik} - d_{ik} x_j + d_{jk} x_i - x_i d_{jk}$ reduces to zero modulo I and
- there exists a monomial ordering $\prec$ on $\mathbb{K}[x_1, \dots, x_n]$, such that $\text{lm}(d_{ij}) \prec x_i x_j$ for each $i < j$, that is the leading monomial of d_{ij} is smaller than $x_i x_j$.

THEOREM 2 (PREIMAGE OF A LEFT IDEAL). *Let A, B be G -algebras of Lie type, generated by $\{x_i \mid 1 \leq i \leq n\}$ and $\{y_j \mid 1 \leq j \leq m\}$ respectively, subject to finite sets of relations R_A, R_B as in Def. 2. Let $\phi : A \rightarrow B$ be a homomorphism of $\mathbb{K}$ -algebras. Define I_ϕ to be the (A, A) -bimodule $A \langle \{x_i - \phi(x_i) \mid 1 \leq i \leq n\} \rangle_A \subset A \otimes_{\mathbb{K}} B$. Suppose, that there exists an elimination ordering for B on $A \otimes_{\mathbb{K}} B$, satisfying the following conditions*

$$1 \leq i \leq n, 1 \leq j \leq m, \quad \text{lm}(y_j \phi(x_i) - \phi(x_i) y_j) \prec x_i y_j.$$

Then there are the following statements.

- 1) Define $A \otimes_{\mathbb{K}}^{\phi} B$ to be a K -algebra, generated by $\{x_1, \dots, x_n, y_1, \dots, y_m\}$ subject to the finite set of relations, composed of R_A, R_B and $\{y_j x_i - x_i y_j - y_j \phi(x_i) + \phi(x_i) y_j\}$. Then $A \otimes_{\mathbb{K}}^{\phi} B$ is a G -algebra of Lie type.
- 2) Let $J \subset B$ be a left ideal, then

$$\phi^{-1}(J) = (I_\phi + J) \cap A \subset A \otimes_{\mathbb{K}}^{\phi} B \cap A.$$

Moreover, this computation can be done by means of elimination.

Now we apply the theorem to the Malgrange ideal and the algebraic Mellin transform. Assume D_n is generated by $\{x_i, \partial_i\}$ subject to usual relations on n -th Weyl algebra. Then $A = D_n[s]$, where we denote the generators by $\{X_i, D_i\}$ and $B = D_n \langle t, \partial_t \mid \partial_t t = t \partial_t + 1 \rangle$. Hence on $E := A \otimes_{\mathbb{K}}^{\phi} B = \mathbb{K} \langle X_i, D_i, s, x_i, \partial_i, t, \partial_t \rangle$ there are both relations $\{D_i X_i = X_i D_i + 1, \partial_i x_i = x_i \partial_i + 1, \partial_t t = t \partial_t + 1\}$, inherited from A and B , and additional relations $st = ts - t, s \partial_t = \partial_t s + \partial_t$.

According to Theorem 2, $I_\phi = \langle \{X_i - x_i, D_i - \partial_i, s + t \partial_t + 1\} \rangle \subset E$. Now, we use the fact that $\phi|_{D_n} = \text{id}_{D_n}$ and move to the smaller algebra $E' = \mathbb{K} \langle t, \partial_t, x_i, \partial_i, s \rangle$ with the part of relations from E . Similarly, $I_\phi \subset E$ can be simplified to the form $I'_\phi = \langle \{s + t \partial_t + 1\} \rangle \subset E'$.

Since $[t, s] = t$ and $[\partial_t, s] = -\partial_t$, the ordering conditions of the Theorem 2 take the form $t \prec st$ and $\partial_t \prec s \partial_t$, which are satisfied as soon as $1 \prec t, \partial_t, s$. In the elimination ordering for $\{t, \partial_t\}$ we have only to ensure that $1 \prec s$ holds. The construction of such an ordering is trivial.

Then, by the Theorem 2, $\phi^{-1}(I_f) = (I_\phi + I_f) \cap A \subset E \cap A$. Using the simplification above, we obtain $\phi^{-1}(I_f) = (I'_\phi + I_f) \cap A \subset E' \cap A$. Hence, in the algebra E' we have to eliminate $\{t, \partial_t\}$ from the ideal

$$I'_\phi + I_f = \langle t - f, s + t \partial_t + 1, \{ \frac{\partial f}{\partial x_i} \partial_t + \partial_i \} \rangle.$$

Of course, there is a multivariate generalization of the LOT algorithm, that is an algorithm for $f = f_1 \cdot \dots \cdot f_p$,

going along the same lines. Notably, in the LOT algorithm we do not use any homogenization, similar to the OT algorithm. This decreases the total number of extra variables by one half (instead of $\{u_i, v_i\}$ we work with $\{s_i\}$). On the other hand, we do not homogenize the generators of I_f , but work with the original generators. Much more important is the fact, that we use just one elimination in the algorithm for getting the complete result, while in the OT algorithm one has to eliminate $\{u_i, v_i\}$ first and then perform the intersection with a subalgebra by a trick of Oaku, which needs some nontrivial work to be done.

Indeed, the approach, presented here, leads us to a new proof of the Briançon-Maisonobe algorithm, which will be published in another article.

2.4 The CHECKROOT algorithm

As we have mentioned, several algorithms for computing the b -function associated with a polynomial have been known. However, in general it is very hard from computational point of view to obtain this polynomial, and in the actual computation only a reduced number of examples can be treated. That is why we suggest the following idea in order to enhance the computation of the Bernstein-Sato polynomial via Gröbner bases.

1. Obtain an upper bound for $b_f(s)$: find $B(s) \in \mathbb{K}[s]$ such that $b_f(s)$ divides $B(s)$.

$$B(s) = \prod_{i=1}^d (s - \alpha_i)^{m_i}$$

2. Check whether α_i is a root of the b -function.
3. Compute the multiplicity of α_i as a root of $b_f(s)$.

There exist some well-known methods to obtain an upper bound for the Bernstein-Sato polynomial of a hypersurface singularity once we know, for instance, an embedded resolution of such singularity [12]. However, as far as we know, there is no algorithm for computing the b -function from this upper bound. In this section we present an algorithm for checking whether a given rational number is a root of the b -function and for computing its multiplicity. As an application we could obtain $b_f(s)$ for some interesting non-isolated singularities, see Example 1 below.

From the definition of the b -function it is clear that

$$\langle b_f(s) \rangle = (\text{Ann}_{D_n[s]}(f^s) + \langle f \rangle) \cap \mathbb{K}[s]. \quad (2.1)$$

In fact, this is another way of defining the Bernstein-Sato polynomial. This equation was used to prove the main result of this section, see Theorem 3 below.

THEOREM 3. *Let R be a ring whose center contains $\mathbb{K}[s]$ as a subring. Let us consider $q(s) \in \mathbb{K}[s]$ a polynomial in one variable and I a left ideal in R satisfying $I \cap \mathbb{K}[s] \neq 0$. Then $(I + R\langle q(s) \rangle) \cap \mathbb{K}[s] = I \cap \mathbb{K}[s] + \mathbb{K}[s]\langle q(s) \rangle$. In particular, using the above equation (2.1), we have*

$$\begin{aligned} & (\text{Ann}_{D_n[s]}(f^s) + D_n[s] \cdot \langle f, q(s) \rangle) \cap \mathbb{K}[s] = \\ & = \langle b_f(s), q(s) \rangle = \langle \gcd(b_f(s), q(s)) \rangle. \end{aligned}$$

PROOF. Let $b(s) \neq 0$ be a generator of $I \cap \mathbb{K}[s]$. First suppose that $h(s) \in (I + \langle q(s) \rangle) \cap \mathbb{K}[s]$. Then we have

$$h(s) = P(s) + Q(s)q(s) \quad (2.2)$$

with $P(s) \in I$ and $Q(s) \in R$. Let $d(s)$ be the greatest common divisor of $b(s)$ and $q(s)$. There exist $b_1(s)$ and $q_1(s)$ such that $d(s)b_1(s) = b(s)$ and $d(s)q_1(s) = q(s)$, and hence $b_1(s)q(s) = q_1(s)b(s)$. Since s commutes with all elements in R , multiplying the equation (2.2) by $b_1(s)$, we obtain

$$b_1(s)h(s) = b_1(s)P(s) + Q(s)q_1(s)b(s) \in I$$

Thus, $b_1(s)h(s) \in I \cap \mathbb{K}[s] = \langle b(s) \rangle$ and therefore $h(s) \in \langle b(s) \rangle : \langle b_1(s) \rangle = \langle d(s) \rangle = I \cap \mathbb{K}[s] + \langle q(s) \rangle$. The other inclusion follows obviously. $\square$

Corollary 1. Let $\{P_1(s), \dots, P_k(s)\}$ be a system of generators of the annihilator of f^s in $D_n[s]$. The following conditions are equivalent:

1. $\alpha \in \mathbb{Q}_{>0}$ is a root of $b_f(-s)$.
2. $D_n[s] \cdot \langle P_1(s), \dots, P_k(s), f, s + \alpha \rangle \neq D_n[s]$.
3. $D_n[s] \cdot \langle P_1(-\alpha), \dots, P_k(-\alpha), f \rangle \neq D_n[s]$.

Moreover, in such a case $D_n[s] \cdot \langle P_1(s), \dots, P_k(s), f, s + \alpha \rangle \cap \mathbb{K}[s] = \langle s + \alpha \rangle$.

PROOF. Take $J = D_n[s] \cdot \langle P_1(s), \dots, P_k(s), f, s + \alpha \rangle$ and $K = J \cap D_n = \langle P_1(-\alpha), \dots, P_k(-\alpha), f \rangle$. Since $J = D[s] \iff J \cap \mathbb{K}[s] = \mathbb{K}[s] \iff K = D_n$ and $\gcd(b_f(s), s + \alpha) = 1$ if and only if $b_f(-\alpha) \neq 0$, the result follows from applying Theorem 3 using $q(s) = s + \alpha$. $\square$

Corollary 2. Let m_α be the multiplicity of α as a root of $b_f(-s)$ and let us consider the ideals $J_i = \text{Ann}_{D_n[s]}(f^s) + \langle f, (s + \alpha)^{i+1} \rangle \subseteq D_n[s]$, $i = 0, \dots, n$. The following conditions are equivalent:

1. $m_\alpha > i$.
2. $J_i \cap \mathbb{K}[s] = \langle (s + \alpha)^{i+1} \rangle$.
3. $(s + \alpha)^i \notin J_i$.

Moreover if $D_n[s] \supsetneq J_0 \supsetneq J_1 \supsetneq \dots \supsetneq J_{m-1} = J_m$, then $m_\alpha = m$. In particular, $m \leq n$ and $J_{m-1} = J_m = \dots = J_n$.

PROOF. $1 \iff 2$. Since $m_\alpha > i$ iff $\gcd(b_f(s), (s + \alpha)^{i+1}) = (s + \alpha)^{i+1}$, the equivalence follows by applying Theorem 3 using $q(s) = (s + \alpha)^{i+1}$.

$2 \implies 3$. If $(s + \alpha)^i \in J_i \cap \mathbb{K}[s]$, then $J_i \cap \mathbb{K}[s] \neq \langle (s + \alpha)^{i+1} \rangle$. $3 \implies 2$. Let $h(s) \in \mathbb{K}[s]$ be the monic generator of the ideal $J_i \cap \mathbb{K}[s]$. Since $(s + \alpha)^{i+1} \in J_i \cap \mathbb{K}[s] = \langle h(s) \rangle$, there exists $j \leq i + 1$ such that $h(s) = (s + \alpha)^j$. Suppose that $j \leq i$. Then, $(s + \alpha)^i = (s + \alpha)^{i-j}(s + \alpha)^j = (s + \alpha)^{i-j}h(s) \in J_i$. That, however, contradicts 3 and thus $j = i + 1$.

The rest of the assertion follows by applying the above result using $i = m$ and $i = m - 1$, since $(s + \alpha)^m \in J_m$ and $(s + \alpha)^{m-1} \notin J_{m-1}$ from the hypothesis. $\square$

Once we know a system of generators of the annihilator of f^s in $D_n[s]$, the last two corollaries provide an algorithm for checking whether a given rational number is a root of the b -function of f and for computing its multiplicity, using Gröbner bases for differential operators.

Algorithm 1 CHECKROOT (checks whether $\alpha \in \mathbb{Q}_{>0}$ is a root of $b_f(-s)$ and computes its multiplicity)

Input 1: $\{P_1(s), \dots, P_k(s)\} \subseteq D_n[s]$, a system of generators of $\text{Ann}_{D_n[s]}(f^s)$,
Input 2: f a polynomial in R_n , α a number in $\mathbb{Q}_{>0}$;
Output: m_α , the multiplicity of α as a root of $b_f(-s)$;

```

for  $i = 0$  to  $n$  do
   $J := D_n[s] \cdot \langle P_1(s), \dots, P_k(s), f, (s + \alpha)^{i+1} \rangle$ ;  $\triangleright J_i$ 
   $G :=$  Gröbner basis of  $J$  w.r.t. ANY term ordering;
   $r :=$  normal form of  $(s + \alpha)^i$  with respect to  $G$ ;
  if  $r = 0$  then
     $m_\alpha := i$ ;  $\triangleright r = 0 \implies (s + \alpha)^i \in J_i$ 
    break  $\triangleright$  leave the for block
  end if
end for
return  $m_\alpha$ ;

```

PROOF. (of Algorithm 1).

Termination: The algorithm CHECKROOT clearly terminates and one only has to consider the loop from 0 to n because the multiplicity of every root of $b_f(s)$ is less than or equal to n , see [12].

Correctness: Corollaries 1 and 2 imply the correctness of CHECKROOT. $\square$

This algorithm is much faster, than the computation of the whole Bernstein polynomial via Gröbner bases, because no elimination ordering is needed for computing a Gröbner basis of J . Also, the element $(s + \alpha)^{i+1}$, added as a generator, seems to simplify tremendously such a computation. Actually, when $i = 0$ it is possible to eliminate the variable s in advance and we can perform the whole computation in D_n , see Corollary 1 (3) above. Let us see an example.

Example 1. Let $f = (xy + z)(y^4 + z^5 + yz^4) \in \mathbb{Q}[x, y, z]$ and $B_1(s) = b_{x^5}(s)b_{y^{18}}(s)b_{z^{24}}(s)$. Since every root of $b_f(-s)$ belongs to the real interval $(0, 2)$ [27], [22], computing an embedded resolution of the singularity and using Kashiwara's result [12], we obtain that $B(s) = B_1(s)B_1(s+1)B_1(s+2)$ is an upper bound for $b_f(s)$. Once we know a system of generator of the annihilator of f^s in $D_n[s]$, checking whether each root of the upper bound is a root of the Bernstein-Sato polynomial or not was easy. It took less than 5 seconds except for those ones which appear in the table below. At the moment, to the best of our knowledge, this example is intractable by any computer algebra system.

The running time is given in the format *minutes:seconds*.

Root of $B(-s)$	Running time	Root of $b_f(-s)$?
5/4	29:16	Yes
31/24	26:16	Yes
29/24	7:51	Yes
9/8	0:35	Yes

We close this section by referring to H. Nakayama who gives in [18] an algorithm for computing local b -functions. One step in his algorithm uses a bound for the multiplicity of a given rational root of the global b -function. Then the algorithm checks if this multiplicity agrees with the local one. This approach is very similar to our `checkRoot` algorithm.

3. STRUCTURE OF THE LIBRARY

The library `dmod.lib` contains implementations of core functions, used in computational D -module theory. We provide top-level procedures `annfs`, `Sannfs`, written with the “user’s minimal knowledge” principle as well as specialized procedures, on which, in particular, the top-level procedures are built.

The library is well documented and includes a set of non-trivial examples for each of its procedures. It combines user-friendliness with the functional flexibility and demonstrates good performance, see the details below.

We index the names of similar procedures by the algorithm, which is used. We use the following acronyms for the algorithms: *OT* for Oaku and Takayama, *LOT* for Levandovskyy’s modification of Oaku and Takayama and *BM* for Briançon-Maisonobe. Moreover, it is possible to specify the desired Gröbner basis engine (`std` or `slimgb`) via the optional argument.

There are the following functions for a polynomial $F \in R_n = \mathbb{K}[x_1, \dots, x_n]$:

- `annfs(F)` (corresponds to the Step 1 and Step 2) computes the Bernstein-Sato polynomial $b_F(s)$ and the ideal $\text{Ann}_D F^{s_0} \subset D$, the D -module structure on $\mathbb{K}[x, F^{-1}]$. This top-level procedure calls one of the `annfsOT`, `annfsLOT`, `annfsBM` procedures with the `annfsBM` as the default option.
- `Sannfs(F)` computes $\text{Ann}_{D[s]} F^s \subset D[s]$ (corresponds to the Step 1) and, like the procedure before, uses one of the `SannfsOT`, `SannfsLOT`, `SannfsBM` procedures.
- `annfs0(I, F)` computes the Bernstein-Sato polynomial $b_F(s)$, its minimal integral root s_0 and the ideal $\text{Ann}_D F^{s_0} \subset D$ (corresponds to the Steps 2-4) from the given $I = \text{Ann}_{D[s]} F^s \subset D[s]$ and F .
- `annfspecial(I, F, s_0, r)` computes the annihilator ideal $\text{Ann}_D F^r \subset D$ for arbitrary rational number r from the knowledge of $I = \text{Ann}_{D[s]} F^s \subset D[s]$ and minimal integral root s_0 of Bernstein-Sato polynomial.
- `operatorBM(F)` computes all the D -module data relevant to F : $\text{Ann}_{D[s]} F^s, b_F(s), s_0, \text{Ann}_D F^{s_0}, P_F(s)$.
- `Sannfslog(F)` computes a left ideal, generated by the elements in $\text{Ann}_{D[s]} F^s$, coming from logarithmic derivations.

For a product of polynomials, there is the function `annfsBMI`, which computes the D -module structure of $R_n[[F^{-1}]] \cdot F^s$, where $F^s = F_1^{s_1} \dots F_p^{s_p}$, according to the multivariate algorithm of Briançon-Maisonobe. Along with the ideal $\text{Ann}_{D_n[s_1, \dots, s_p]} F^s$, the procedure outputs a Bernstein-Sato ideal in $\mathbb{K}[s_1, \dots, s_p]$, which need not be principal in general.

4. COMPARISON

The approach of Oaku and Takayama (possibly with its modifications) is implemented in computer algebra systems ASIR ([20, 2]), KAN/SM1 ([24, 2]) and MACAULAY2 [8].

These systems have special packages for working with D -modules: `cohom.sm1` for KAN/SM1, `bfct` for ASIR and `D-modules.m2` for MACAULAY2 [26]. Above we have described our implementation in `dmod.lib` of SINGULAR.

Among the vast collection of available examples ([6, 19, 23] etc.) we have chosen a subset, consisting of problems, which can be solved by all three computer algebra packages for D -modules. We have noticed a curious fact: it is hard to find examples of medium complexity, since most of available examples are either easy or very hard.

The tests were performed on AMD Athlon 64 Processor 3400+ (2400 MHz) with 4 GB RAM running Fedora 8 Linux.

We have used KAN/SM1 release 3.050615, RISA/ASIR version 20070806 with `bfct`, MACAULAY2 1.1 with `Dmodules.m2` and SINGULAR 3-0-4 with `dmod.lib` version 1.26.

In the tests, we would like to compute both steps of OT approach. In the first one we get the $\text{Ann}_{D[s]} F^s$, and in the second one – the Bernstein-Sato polynomial (preferably in a factorized form), its minimal integral root s_0 and $\text{Ann}_D F^{s_0}$. Both SINGULAR and KAN/SM1 return these data, although KAN/SM1 outputs non-factorized form, while the factorization is done internally. During the computation of the Bernstein-Sato polynomial with MACAULAY2, the ideal $\text{Ann}_{D[s]} F^s$ is computed and used internally, however it is not possible to output the ideal with the tools, available to the user. ASIR uses a very different method (see [19]) for computing the Bernstein-Sato polynomial, hence we compare with ASIR only in the computation of $\text{Ann}_{D[s]} F^s$.

4.1 Example scripts, used in tests

Below we place the typical scripts we used for testing. In these scripts we are using the polynomial

$$x^4z - xy^3z + x^3y - y^4 = (xz + y) \cdot (x^3 - y^3).$$

4.1.1 RISA/ASIR

The computation of $\text{Ann}_{D[s]} F^s$ is done as follows.

```
load("bfct");
ann(x^4*z-x*y^3*z+x^3*y-y^4);
```

4.1.2 KAN/SM1

The classical KAN/SM1 commands, which we use, are

```
(cohom.sm1) run ::
(oxasir.sm1) run ::
[(x^4 z - x y^3 z + x^3 y - y^4) (x,y,z)] annfs ::
```

4.1.3 MACAULAY2

In order to follow the OT approach, we proceed as follows.

```
load "D-modules.m2"; // load the package
R = QQ[x,y,z]; // define a commutative ring
D = makeWA R; // create Weyl algebra w.r.t. R
f = x^4*z-x*y^3*z+x^3*y-y^4;
b = globalBFunction (f, Strategy => ViaAnnFs);
B = factor(b); print(B); // factor+print BS poly
```

4.1.4 SINGULAR

Working with SINGULAR, one can set `printlevel=1` to obtain the very detailed information on the computation. Setting `printlevel=2` results in printing all the intermediate

data additionally. Hence, it suffices to run `annfs` function, which returns the Bernstein-Sato polynomial and $\text{Ann} F^{s_0}$, since $\text{Ann}_{D[s]} F^s$ is computed in the first step of the algorithm internally and can be seen with the above option activated. The function `Sannfs` computes $\text{Ann}_{D[s]} F^s$ only.

```
LIB "dmod.lib"; // load the library
ring R = 0,(x,y,z),dp; // define a commutative ring
poly F = x^4*z-x*y^3*z+x^3*y-y^4;
def B = annfs(F); // run the computation
setring B;
BS; // output BS (factorized BS-polynomial)
LD; // output LD (ideal Ann F^s0)
```

4.2 Tests and Timings

We have used the following polynomials for test examples:

Example	Polynomial
cnuN	$(xz + y)(x^N - y^N)$
ttNM	$x^N + y^N + z^N - (xyz)^M$
ha1	$xyz(x + y)(x + z)$
ha2	$xyzw(x + y)(x + z)(x + w)$
ha25	$xyz(x - z)(y - x)(3x + 2y + z)(y + z)$
ha28	$xyz(x + y)(x + z)(y + z)(y + z - x)$
ha30	$xyz(x - y)(y + z)(x - 2z)(x - y - z)$
ha31	$xyz(x - y)(y + z)(2x - z)(x - y - z)$
abNM	$(z^N + w^M)(Nz^{N-1}x + Mw^{M-1}y)$
xc45	$(xz + y)(x^4 + y^5)$
reiffen11	$x^{11} + y^{11} + xy^{10}$
reiffenNM	$x^N + y^M + xy^{M-1}$

In the tests for KAN/SM1, MACAULAY2 and SINGULAR we request the computation of both $\text{Ann}_{D[s]} f^s$ and the Bernstein-Sato polynomial of f .

The running time is given in format *minutes:seconds*.

Example	KAN/SM1	MACAULAY2	SINGULAR
cnu3	0:24	0:04	0:01
cnu4	0:54	0:11	0:02
cnu5	5:27	1:04	0:18
ha1	21:41	1:13	0:04
reiffen11	19:02	2:22	0:32
reiffen66	0:25	0:03	0:01
reiffen77	0:38	0:06	0:01
reiffen88	1:28	0:14	0:03
reiffen99	4:09	0:30	0:07
tt32	0:25	0:03	0:01
tt42	0:24	0:05	0:01
tt43	3:25	0:21	0:02

We have to make a remark on KAN/SM1. Indeed, it uses ASIR for the polynomial factorization and thus there is extra time, used for communication. But even if we divide the total running time (as above) by a reasonable factor two, we see, that this does not change the general behaviour of KAN/SM1. Therefore we do not use KAN/SM1 for harder tests.

In the tests for MACAULAY2, ASIR and three algorithms (OT, LOT and BM), implemented in SINGULAR, we ask for the computation of $\text{Ann}_{D[s]} f^s \subset D[s]$.

The running time is given in format *[hours[h]]:min:sec*.

Example	MACAULAY2	ASIR	SINGULAR		
			OT	LOT	BM
ha25	1:31	1:26	1:02	0:46	0:21
ha28	2:00	2:03	1:10	1:17	0:35
ha30	1:58	2:08	1:15	1:36	0:42
ha31	1:23	2:07	1:11	1:11	0:33
reiffen89	3:37	5:20	0:36	0:09	0:03
ab23	4:08	0:25	0:07	0:04	0:02
xc45	16:59	4:09	1:54	1:39	0:59
ha2	6h:25:17	1h:37:08	57:27	52:16	39:20

5. SOLVED CHALLENGES

In [6, 7], there appeared several challenges for the computation of Bernstein-Saito ideals (denoted by $\mathcal{B}$ in [6]) for products of polynomials. In particular, the following two cases of pairs of polynomials were “...intractable by any computer algebra system” at that time. In our implementation the needed computation is performed by the function `annfsBMI` (see Section 3) with the multivariate method of Briançon-Maisonobe.

Let us denote $\Delta := \{5, 7, 9, 11, 13\}$. For the $F = f_1 \cdot f_2$, where $f_1 = x^3 + y^2$, $f_2 = x^2 + y^3$, we spent 68 sec to obtain a principal Bernstein-Saito ideal

$$(s_1 + 1) \cdot (s_2 + 1) \cdot \prod_{i \in \Delta} (4s_1 + 6s_2 + i) \cdot (6s_1 + 4s_2 + i).$$

As we can see, the integral solutions are $(-1, k)$ and $(k, -1)$ for $k \in \mathbb{Z}$. These sets are intersecting at the solution $(-1, -1)$.

For the $F = f_1 \cdot f_2$, where $f_1 = x^2 + y^2 + y^3$, $f_2 = x^2 + y^3$, we spent 233 sec to obtain a principal Bernstein-Saito ideal

$$(s_1 + 1) \cdot (s_2 + 1) \cdot (s_1 + s_2 + 1) \cdot (s_1 + s_2 + 2) \cdot (2s_1 + 2s_2 + 3) \cdot (2s_1 + 2s_2 + 5) \cdot \prod_{i \in \Delta} (4s_1 + 6s_2 + i).$$

For the $F = f_1 \cdot f_2$, where $f_1 = x^2 + y^2 + y^3$, $f_2 = x^3 + y^2$, we spent 453 sec to obtain the ideal, which coincides with the ideal as above. Note, that despite this coincidence the annihilator ideals are quite different. In the results above we decided to present generators of principal ideal in a non-monic form for better readability.

6. CONCLUSION

As we have demonstrated in Section 4.2, our implementation is very fast in the realm of D -module algorithms, based on elimination.

With our implementation and experiments above we confirm the facts, that the algorithm of Briançon-Maisonobe, computing the $\text{Ann}_{D[s]} f^s$, is generically the fastest one, while the LOT algorithm is faster than the original algorithm due to Oaku and Takayama.

Our work on `dmod.lib` continues in several directions. Having a fast implementation of core algorithms, it is natural to implement more algorithmic applications like localization, restriction and integration of D -modules and even more sophisticated algorithms as de Rham cohomology, local cohomology, Weyl closure and so on. On the other hand, there are many questions, arising in connection with various

generalizations of the D -module theory to the case, when the polynomial f factorizes (we have briefly mentioned it) as well as in cases, when we are given not just a hypersurface, but an arbitrary variety.

We are going to investigate and enhance both LOT and `checkRoot` algorithms and apply the developed techniques to further mathematical problems.

There is a strong need for creating a unified collection of examples, available online, and the tools for presenting the examples in different formats of various computer algebra systems. It seems possible to adapt the tools of SYMBOLIC-DATA [1] project for this purpose in the future.

We would like to stress, that we develop the library using easy, C-like interpreted language of SINGULAR, hence a potential contributor can join our project at virtually any stage of it.

7. ACKNOWLEDGMENTS

We would like to thank Francisco Castro-Jiménez, José-María Ucha, Gert-Martin Greuel, Enrique Artal and José-Ignacio Cogolludo for their constant support and motivation in our work over years.

We are grateful to Masayuki Noro, Nobuki Takayama and Anton Leykin for their help and careful explanations, concerning computer algebra systems ASIR, KAN/SM1 and MACAULAY2 respectively. Moreover, we appreciate the valuable suggestions of anonymous referees, especially the examples **ha25** to **ha31** of hyperplane arrangements.

We wish to express our gratitude to Michael Brickenstein and Hans Schönemann for the tight collaboration and for the implementation of `slimgb` and for constant advise concerning SINGULAR respectively.

The authors thank to the DFG Graduiertenkolleg “Hierarchie und Symmetrie in mathematischen Modellen” at RWTH Aachen, Germany and projects MTM2007-67908-C02-01 and FQM-333 of Spain for the partial financial support.

8. REFERENCES

- [1] The SYMBOLICDATA project, 2000–2008. Available at <http://www.SymbolicData.org>.
- [2] OPENXM (Open message eXchange for Mathematics). A project to integrate mathematical software systems, 2007. Available at <http://www.openxm.org>.
- [3] I. N. Bernšteĭn. Analytic continuation of generalized functions with respect to a parameter. *Funkcional. Anal. i Priložen.*, 6(4):26–40, 1972.
- [4] J. Briançon and P. Maisonobe. Remarques sur l’idéal de Bernstein associé à des polynômes. *Preprint no. 650, Univ. Nice Sophia-Antipolis*, 2002.
- [5] M. Brickenstein. Slimgb: Gröbner Bases with Slim Polynomials. In *Rhine Workshop on Computer Algebra*, pages 55–66, 2006. Proceedings of RWCA’06, Basel, March 2006.
- [6] F. Castro-Jiménez and J. Ucha-Enríquez. On the computation of Bernstein-Sato ideals. *J. Symbolic Computation*, 37:629–639, 2004.

- [7] J. Gago-Vargas, M. Hartillo-Hermoso, and J. Ucha-Enríquez. Comparison of theoretical complexities of two methods for computing annihilating ideals of polynomials. *J. Symbolic Computation*, 40(3):1076–1086, 2005.
- [8] D. Grayson and M. Stillman. MACAULAY 2, a software system for research in algebraic geometry, 2005. Available at <http://www.math.uiuc.edu/Macaulay2>.
- [9] G.-M. Greuel, V. Levandovskyy, and H. Schönemann. PLURAL. A SINGULAR 3.0 Subsystem for Computations with Non-commutative Polynomial Algebras. Centre for Computer Algebra, University of Kaiserslautern, 2006. Available at <http://www.singular.uni-kl.de>.
- [10] G.-M. Greuel and G. Pfister. *A SINGULAR Introduction to Commutative Algebra*. Springer, 2nd edition, 2008.
- [11] G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR 3.0. A Computer Algebra System for Polynomial Computations. Centre for Computer Algebra, University of Kaiserslautern, 2005. Available at <http://www.singular.uni-kl.de>.
- [12] M. Kashiwara. B -functions and holonomic systems. Rationality of roots of B -functions. *Invent. Math.*, 38(1):33–53, 1976/77.
- [13] V. Levandovskyy. Intersection of ideals with non-commutative subalgebras. In *Proc. of the International Symposium on Symbolic and Algebraic Computation (ISSAC'06)*, pages 212–219. ACM Press, 2006.
- [14] V. Levandovskyy and J. Morales. A SINGULAR 3.0 library for computations with algebraic D -modules `dmod.lib`, 2008. Available at <http://www.singular.uni-kl.de>.
- [15] V. Levandovskyy and H. Schönemann. Plural – a computer algebra system for noncommutative polynomial algebras. In *Proc. of the International Symposium on Symbolic and Algebraic Computation (ISSAC'03)*, pages 176–183. ACM Press, 2003.
- [16] A. Leykin. Constructibility of the set of polynomials with a fixed Bernstein-Sato polynomial: an algorithmic approach. *J. Symbolic Comput.*, 32(6):663–675, 2001. Effective methods in rings of differential operators.
- [17] J. McConnell and J. Robson. *Noncommutative Noetherian rings. With the cooperation of L. W. Small*. Graduate Studies in Mathematics. 30. Providence, RI: American Mathematical Society (AMS), 2001.
- [18] H. Nakayama. An Algorithm Computing the Local b Function by an Approximate Division Algorithm in $\hat{D}$. Available at <http://arxiv.org/abs/math/0606437v1>.
- [19] M. Noro. An efficient modular algorithm for computing the global b -function. In A. M. Cohen, editor, *Mathematical software*. World Scientific, 2002. Proceedings of the 1st international congress on mathematical software, Beijing, China, August 2002.
- [20] M. Noro, T. Shimoyama, and T. Takeshima. RISA/ASIR, an open source general computer algebra system, 2006. Available at <http://www.math.kobe-u.ac.jp/Asir>.
- [21] T. Oaku. Algorithms for the b -function and D -modules associated with a polynomial. *J. Pure Appl. Algebra*, 117/118:495–518, 1997.
- [22] M. Saito. On microlocal b -function. *Bull. Soc. Math. France*, 122(2):163–184, 1994.
- [23] M. Saito, B. Sturmfels, and N. Takayama. *Gröbner deformations of hypergeometric differential equations*, volume 6 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 2000.
- [24] N. Takayama. KAN/SM1, a Gröbner engine for the ring of differential and difference operators, 2003. Available at <http://www.math.kobe-u.ac.jp/KAN>.
- [25] Q.-N. Tran. A new class of term orders for elimination. *J. Symbolic Computation*, 42(5):533–548, 2007.
- [26] H. Tsai and A. Leykin. D -modules package for MACAULAY 2 – algorithms for D -modules, 2006. Available at <http://www.ima.umn.edu/~leykin/Dmodules>.
- [27] A. N. Varčenko. Asymptotic Hodge structure on vanishing cohomology. *Izv. Akad. Nauk SSSR Ser. Mat.*, 45(3):540–591, 688, 1981.

Complex Brackets, Balanced Complex Differences, and Applications in Symbolic Geometric Computing

Hongbo Li

Mathematics Mechanization Key Lab
AMSS, Chinese Academy of Sciences
Beijing 100080, China
hli@mmrc.iss.ac.cn

Lei Huang

Mathematics Mechanization Key Lab
AMSS, Chinese Academy of Sciences
Beijing 100080, China
holoqgyz@yahoo.com.cn

ABSTRACT

In advanced invariant algebras such as null bracket algebra (NBA), symmetries of algebraic operators are the most important devices of encoding and employing syzygies of advanced geometric invariants. The larger the symmetry group, the more powerful the computing devices.

In this paper, the largest symmetry group of the two kinds of bracket operators in the NBA of plane geometry is found. An algorithm of complexity $O(N \log N)$ is proposed to reduce a bracket of length N to its normal form, and then decide the congruence of two brackets of length N .

By writing the two bracket operators as the real and pure imaginary parts of a complex bracket operator, their normal forms can be translated into a class of complex polynomials whose variables are first-order differences, called balanced complex difference (BCD) polynomials.

BCD polynomials provide a complex-numbers-based invariant language for advanced algebraic manipulations of geometric problems. A simplification algorithm is proposed for making symbolic geometric computing with NBA and BCD polynomials, with the unique feature of controlling the expression size by avoiding multilinear expansions of the first-order difference variables of complex polynomials.

ACM Computing Classification: I.1.1 [Symbolic and Algebraic Manipulation]: Expressions and Their Representation; G.4 [Mathematical Software]: Efficiency.

General Terms: Theory; algorithm.

Keywords: Bracket Algebra, Complex Numbers Method, Theorem Proving, Geometric Algebra, Graph Theory.

1. INTRODUCTION

Invariant algebras introduce compact algebraic representations for geometric objects and constraints [1], [2], [7], [9]. Due to the presence of syzygies among invariants, algebraic manipulations in invariant algebras are more tricky than in coordinate polynomials. To keep more geometric symmetries in algebraic manipulations, at the same time maintain

the expressions in manipulation to small size, advanced invariant algebras such as null bracket algebra (NBA) and null Grassmann-Cayley algebra (NGC) [6] are proposed. In these algebras, simplifications rely more on symmetries within algebraic operators than on syzygies among basic invariants.

NBA grows out of the combination of the conformal model and Clifford algebra. It is generated as a ring by two kinds of long brackets: angular bracket $\langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle$ and square bracket $[\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{n+2l}]$, where the $\mathbf{a}_i$ are null vectors in $(n+2)$ D Minkowski space $\mathbb{R}^{n+1,1}$ representing points in n D (n -dimensional) Euclidean space. The two brackets exhibit two general symmetries — shift symmetry and reversion symmetry:

$$\begin{aligned} \langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle &= \langle \mathbf{a}_2 \cdots \mathbf{a}_{2k} \mathbf{a}_1 \rangle \\ &= \langle \mathbf{a}_{2k} \cdots \mathbf{a}_2 \mathbf{a}_1 \rangle, \\ [\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k}] &= (-1)^{n-1} [\mathbf{a}_2 \cdots \mathbf{a}_{2k} \mathbf{a}_1] \\ &= (-1)^{\frac{n(n-1)}{2}} [\mathbf{a}_{2k} \cdots \mathbf{a}_2 \mathbf{a}_1]. \end{aligned} \quad (1.1)$$

Since symmetries are the most important devices of symbolic manipulations in NBA, it is natural to ask the following question: what are the largest groups of symmetries of the two bracket operators respectively? The answer for $n > 2$ is still open, although there are evidences indicating that (1.1) generates all the symmetries for generic vectors $\mathbf{a}_i$.

The answer for $n = 2$ is given in Theorem 6 of this paper: the angular bracket of $\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k}$, together with its square bracket if the normal form of the square bracket is unique, each have $S_k \times S_k \times S_2$ as the largest symmetry group up to a change of sign of the bracket, where S_k denotes the permutation group of k elements. The action of the group upon the bracket is as follows: first the $2k$ elements are written as a sequence of $2k$ vector pairs: $(\mathbf{a}_1, \mathbf{a}_2), (\mathbf{a}_2, \mathbf{a}_3), \dots, (\mathbf{a}_{2k-1}, \mathbf{a}_{2k}), (\mathbf{a}_{2k}, \mathbf{a}_1)$, then an S_k acts upon the set of vector pairs of even positions, another S_k acts upon the set of vector pairs of odd positions, and the non-identity element of S_2 interchanges the two sets.

Hence, two angular or square brackets of generic null vectors are equal if and only if after reduction with respect to the symmetry group, they share a common normal form. If the brackets are of length N , then it takes $O(N \log N)$ transpositions and comparisons to decide their congruence.

In contrast, if deciding the congruence by first expanding the two brackets using Caianiello's expansion [3], then straightening the results [10], the complexity is exponential. Likewise, if first substituting the coordinates of the null vectors into the brackets, then expanding the brackets into coordinate polynomials, the complexity is also exponential.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

The proof of Theorem 6 is based on two new techniques called *complex brackets* and *BCD monomials*. For null vectors $\mathbf{a}_1, \dots, \mathbf{a}_{2k} \in \mathbb{R}^{3,1}$, their complex bracket is

$$\langle \mathbf{a}_1 \cdots \mathbf{a}_{2k} \rangle_{\mathbb{C}} := \langle \mathbf{a}_1 \cdots \mathbf{a}_{2k} \rangle + i[\mathbf{a}_1 \cdots \mathbf{a}_{2k}] \in \mathbb{C}. \quad (1.2)$$

The complex multiplication of several complex brackets is a monomial composed of N linear factors of the form $\mathbf{a}_i - \mathbf{a}_j$, and N conjugate-linear factors of the form $\overline{\mathbf{a}_k - \mathbf{a}_l}$, where $\mathbf{a}_i$ is the complex representation of the planar point represented by $\mathbf{a}_i$. Such a monomial is called a *balanced complex difference* (BCD) monomial.

We prove that any BCD monomial of length $2N$ equals the ratio of a complex bracket to several inner products of null vectors, and propose a graph-theoretic algorithm to make the translation in $O(N^2)$ transpositions and comparisons.

That graph theory comes into the study of NBA is not a coincidence. We show that for any $n \geq 4$, a bracket in n D NBA can be represented by a unique closed walk in one of the following graphs, depending on the value of $n \bmod 4$: usual graph, directed graph, edge-labeled graph of two labels, and edge-labeled directed graph of two labels.

BCD polynomials are complex polynomial translations of homogeneous bracket polynomials in NBA, where the null vectors representing planar points are changed into usual complex numbers representations. The monoid of BCD polynomials inherits the Euclidean invariance of long brackets, and provides a new platform for describing and manipulating Euclidean geometry with complex numbers.

The BCD method is distinguished from the traditional complex numbers method [11] in plane geometry by the employing and manipulating of first-order differences of complex indeterminates. While the traditional complex numbers method is very efficient in handling geometric problems related to the unit circle, it is much less efficient when there is more than one circle. The reason, from the symbolic computation point of view, is that *translation in the plane increases the expression size exponentially by multilinear expansion*: for a single complex monomial $\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_m$ of degree m , a translation by complex number $\mathbf{b}$ changes it into a polynomial of 2^m terms:

$$(\mathbf{a}_1 + \mathbf{b})(\mathbf{a}_2 + \mathbf{b}) \cdots (\mathbf{a}_m + \mathbf{b}) = \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_m + \dots + \mathbf{b}^m. \quad (1.3)$$

BCD polynomials are invariant under translations and rotations because of the first-order differences and pairing of linear and conjugate-linear factors. Their algebraic manipulations do not resort to expanding the product of their first-order difference factors, thus having good control of the expression size. As a result, BCD polynomials can be used to generate elegant complex numbers proofs of geometric theorems when combined with NGC and NBA techniques.

An algorithm is proposed to make simplifications in symbolic geometric computing with BCD polynomials. It is applied to more than twenty theorems for which no monomial proof is found by the method of [6]. The proofs generated by the new method are longer but still within ten terms. Remarkable benefits include the simplification of algebraic manipulations brought about by the commutativity of complex numbers, and systematic graph-theoretic manipulations of bracket expressions in making simplifications.

Section 2 introduces Clifford Algebra and its relation to complex numbers. Sections 3 and 4 investigate NBA, complex brackets and BCD monomials. Section 5 is on bracket normalization, and Section 6 is some geometric applications.

2. CLIFFORD ALGEBRA AND BRACKETS

The invariant and coordinate-free language to describe and manipulate orthogonal geometry is *Clifford algebra*, an associative algebra equipped with a multilinear product called the *geometric product* [5], denoted by juxtaposition. The Clifford algebra over an n D inner-product space $\mathcal{V}^n$ is denoted by $CL(\mathcal{V}^n)$. For any $\mathbf{x}, \mathbf{y} \in \mathcal{V}^n$, its geometric product and inner product are related by $\mathbf{x}\mathbf{y} + \mathbf{y}\mathbf{x} = 2\mathbf{x} \cdot \mathbf{y}$.

$CL(\mathcal{V}^n)$ is linearly isomorphic to the *exterior algebra* $\Lambda(\mathcal{V}^n)$, whose product is called the *outer product*. Since $\Lambda(\mathcal{V}^n)$ is $\mathbb{Z}$ -graded, so is $CL(\mathcal{V}^n)$. The linear isomorphism from $CL(\mathcal{V}^n)$ to $\Lambda(\mathcal{V}^n)$ is called *grading*, and its inverse is *ungrading*.

In exterior algebra, the outer product of finitely many vectors, if nonzero, is called a *Grassmann monomial*, and the number of participating vectors is the *grade* of the monomial. Similarly, in Clifford algebra, the geometric product of finitely many vectors is called a *Clifford monomial*.

For vectors $\mathbf{a}_i \in \mathcal{V}^n$,

$$\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_k = \sum_{l=0}^{\lfloor \frac{k}{2} \rfloor} \langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_k \rangle_{k-2l} \quad (2.1)$$

is the grading of Clifford monomial $\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_k$, where $\langle \rangle_m$ is the *m-grading operator* extracting the m -graded part. So ungrading is just the elimination of grading operators.

When $n = 2$, there are the following ungradings:

$$\begin{aligned} \langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle_0 &= \frac{1}{2}(\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} + (\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k})^\dagger), \\ \langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle_2 &= \frac{1}{2}(\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} - (\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k})^\dagger), \end{aligned} \quad (2.2)$$

where “ $\dagger$ ” is the *reversion operator* in $CL(\mathcal{V}^n)$ defined by

$$(\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_l)^\dagger := \mathbf{a}_l \cdots \mathbf{a}_2 \mathbf{a}_1, \quad \forall \mathbf{a}_i \in \mathcal{V}^n. \quad (2.3)$$

In $CL(\mathcal{V}^n)$, the *angular bracket* of element $\mathbf{A}$ is its scalar part, and the *square bracket* of $\mathbf{A}$ is the coordinate of its n -graded part:

$$\langle \mathbf{A} \rangle := \langle \mathbf{A} \rangle_0, \quad [\mathbf{A}] := \langle \mathbf{A} \rangle_n \mathbf{I}_n^{-1}, \quad (2.4)$$

where $\mathbf{I}_n$ is a fixed nonzero element in the 1D subspace of n -graded elements in $CL(\mathcal{V}^n)$, serving as the basis of the subspace. In particular,

$$\langle \mathbf{a}_1 \mathbf{a}_2 \rangle = \mathbf{a}_1 \cdot \mathbf{a}_2, \quad [\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_n] = \det(\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n), \quad (2.5)$$

and $[\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_n]$ is called a *classical bracket*.

The geometry of coplanar angles is described by $CL(\mathbb{R}^2)$ as follows. Let $\mathbf{e}_1, \mathbf{e}_2$ be an orthonormal basis of $\mathbb{R}^2$. Then

$$\mathbf{I}_2 = \mathbf{e}_2 \wedge \mathbf{e}_1 \quad (2.6)$$

is invariant under any orthogonal transformation of $\mathbb{R}^2$, called the *unit orientation* of the plane. For any $\mathbf{a}, \mathbf{b} \in \mathbb{R}^2$,

$$\begin{aligned} \mathbf{a}\mathbf{b} &= \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \wedge \mathbf{b} \\ &= |\mathbf{a}||\mathbf{b}|(\cos \angle(\mathbf{a}, \mathbf{b}) + \mathbf{I}_2 \sin \angle(\mathbf{a}, \mathbf{b})) \\ &= |\mathbf{a}||\mathbf{b}| \exp(\mathbf{I}_2 \angle(\mathbf{a}, \mathbf{b})), \end{aligned} \quad (2.7)$$

where the exponential map in $CL(\mathcal{V}^n)$ is defined by

$$\exp(\mathbf{A}) := 1 + \mathbf{A} + \frac{\mathbf{A}^2}{2!} + \frac{\mathbf{A}^3}{3!} + \dots, \quad \forall \mathbf{A} \in CL(\mathcal{V}^n). \quad (2.8)$$

Consequently, for any unit vectors $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{2k} \in \mathbb{R}^2$,

$$\begin{aligned} \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} &= \exp(\mathbf{I}_2 \sum_{l=1}^k \angle(\mathbf{a}_{2l-1}, \mathbf{a}_{2l})), \\ \langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle &= \cos(\sum_{l=1}^k \angle(\mathbf{a}_{2l-1}, \mathbf{a}_{2l})), \\ [\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k}] &= \sin(\sum_{l=1}^k \angle(\mathbf{a}_{2l-1}, \mathbf{a}_{2l})). \end{aligned} \quad (2.9)$$

The unit orientation $\mathbf{I}_2$ satisfies $\mathbf{I}_2^2 = -1$, resembling the imaginary unit i . Denote by $CL^+(\mathbb{R}^2)$ the 2D subalgebra generated by 1 and $\mathbf{I}_2$. Then

$$\mathbf{g}(x_1 + \mathbf{I}_2 x_2) := x_1 + i x_2, \quad \forall x_i \in \mathbb{R} \quad (2.10)$$

is an isomorphism of $CL^+(\mathbb{R}^2)$ and $\mathbb{C}$ as real algebras. Under this isomorphism, the reversion operator is identical to the complex conjugate:

$$\mathbf{g}((x_1 + \mathbf{I}_2 x_2)^\dagger) = x_1 - i x_2 = \overline{x_1 + i x_2}. \quad (2.11)$$

The mapping

$$\mathbf{g}'(\mathbf{x}) := \mathbf{x} \mathbf{e}_1, \quad \forall \mathbf{x} \in \mathbb{R}^2 \quad (2.12)$$

changes $(x_1, x_2) \in \mathbb{R}^2$ into $x_1 + \mathbf{I}_2 x_2$. Denote

$$\mathbf{h} = \mathbf{g} \circ \mathbf{g}' : \mathbb{R}^2 \longrightarrow \mathbb{C}. \quad (2.13)$$

Then $\mathbf{h}$ is the *complex numbers representation* of $\mathbb{R}^2$, and

$$\mathbf{h}(\mathbf{a}) \overline{\mathbf{h}(\mathbf{b})} = \mathbf{g}(\mathbf{a} \mathbf{b}), \quad \forall \mathbf{a}, \mathbf{b} \in \mathbb{R}^2. \quad (2.14)$$

Notice that if the orientation of the plane is set to be $\mathbf{I}_2 = \mathbf{e}_1 \wedge \mathbf{e}_2$ instead of (2.6), then (2.14) should be replaced by $\overline{\mathbf{h}(\mathbf{a})} \mathbf{h}(\mathbf{b}) = \mathbf{g}(\mathbf{a} \mathbf{b})$.

For any $\mathbf{a}_1, \dots, \mathbf{a}_{2k} \in \mathbb{R}^2$, their *complex bracket* is

$$\begin{aligned} \langle \mathbf{a}_1 \cdots \mathbf{a}_{2k} \rangle_{\mathbb{C}} &:= \mathbf{g}(\mathbf{a}_1 \cdots \mathbf{a}_{2k}) \\ &= \langle \mathbf{a}_1 \cdots \mathbf{a}_{2k} \rangle + i [\mathbf{a}_1 \cdots \mathbf{a}_{2k}]. \end{aligned} \quad (2.15)$$

By (2.4), (2.10), (2.11) and (2.14), we get from (2.2) the following complex numbers representations of the brackets:

$$\begin{aligned} \langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle_{\mathbb{C}} &= \mathbf{h}(\mathbf{a}_1) \mathbf{h}(\mathbf{a}_3) \cdots \mathbf{h}(\mathbf{a}_{2k-1}) \overline{\mathbf{h}(\mathbf{a}_2) \mathbf{h}(\mathbf{a}_4) \cdots \mathbf{h}(\mathbf{a}_{2k})}, \\ \langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle &= \frac{1}{2} \left(\prod_{l=1}^k \mathbf{h}(\mathbf{a}_{2l-1}) \overline{\mathbf{h}(\mathbf{a}_{2l})} + \prod_{l=1}^k \mathbf{h}(\mathbf{a}_{2l}) \overline{\mathbf{h}(\mathbf{a}_{2l-1})} \right), \\ [\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k}] &= \frac{1}{2i} \left(\prod_{l=1}^k \mathbf{h}(\mathbf{a}_{2l-1}) \overline{\mathbf{h}(\mathbf{a}_{2l})} - \prod_{l=1}^k \mathbf{h}(\mathbf{a}_{2l}) \overline{\mathbf{h}(\mathbf{a}_{2l-1})} \right). \end{aligned} \quad (2.16)$$

3. CONFORMAL MODEL AND NBA

In the Cartesian coordinate model $\mathbb{R}^n$ and the homogeneous coordinate model $\mathbb{R}^{n+1}$ of Euclidean space $\mathbb{E}^n$, a translation can only be represented by the difference of two points, which is a 2-termed vector. Multilinear expansions such as (1.3) are usually indispensable in manipulating translations.

The *conformal model* of $\mathbb{E}^n$ is more appropriate in handling Euclidean geometry than both $\mathbb{R}^n$ and $\mathbb{R}^{n+1}$ [4]. In

this model, $\mathbb{R}^n$ is mapped isometrically onto the set

$$\mathcal{N}_e = \{\mathbf{a} \in \mathbb{R}^{n+1,1} \mid \mathbf{a} \cdot \mathbf{a} = 0, \mathbf{a} \cdot \mathbf{e} = -1\}, \quad (3.1)$$

where $\mathbf{e}$ is a fixed null vector in the orthogonal complement $\mathbb{R}^{1,1}$ of $\mathbb{R}^n$ in $(n+2)$ D Minkowski space $\mathbb{R}^{n+1,1}$, as follows:

$$\mathbf{f}(\mathbf{x}) := \mathbf{e}_0 + \mathbf{x} + \frac{\mathbf{x}^2}{2} \mathbf{e}, \quad \forall \mathbf{x} \in \mathbb{R}^n, \quad (3.2)$$

where $\mathbf{e}_0 \in \mathcal{N}_e \cap \mathbb{R}^{1,1}$ is fixed. By *isometry* we mean $(\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2))^2 = (\mathbf{x}_1 - \mathbf{x}_2)^2$, or equivalently, by (3.2),

$$\mathbf{f}(\mathbf{x}_1) \cdot \mathbf{f}(\mathbf{x}_2) = -\frac{1}{2} d_{\mathbf{x}_1 \mathbf{x}_2}^2, \quad (3.3)$$

where $d_{\mathbf{x}_1 \mathbf{x}_2}$ is the distance between points $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$.

In the conformal model, $\mathbf{e}$ represents the unique *conformal point at infinity*, which is used to compactify $\mathbb{E}^n$. Vector $\mathbf{e}_0$ represents the origin of $\mathbb{R}^n$ via mapping $\mathbf{f}$. Any point of $\mathbb{E}^n$ is represented by a null vector of $\mathbb{R}^{n+1,1}$.

For example, when $n = 2$, let $\mathbf{e} = (0, 0, 1, 1)$ and $\mathbf{e}_0 = (0, 0, -1/2, 1/2)$ in $\mathbb{R}^{3,1}$. Then for any $(x_1, x_2) \in \mathbb{R}^2$,

$$\mathbf{f}(x_1, x_2) = (x_1, x_2, \frac{x_1^2 + x_2^2 - 1}{2}, \frac{x_1^2 + x_2^2 + 1}{2}). \quad (3.4)$$

The expression seems rather strange in real coordinates. When setting $\mathbf{a} = x_1 + i x_2$, then under the following constant unitary transformation

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & i & & \\ & 1 & -i & \\ & & 1 & 1 \\ & & -1 & 1 \end{pmatrix},$$

the real coordinates of $\mathbf{f}(x_1, x_2)$ in $\mathbb{R}^{3,1}$ are changed into the following nice complex coordinates in $\mathbb{C}^4$: $(\mathbf{a}, \bar{\mathbf{a}}, \mathbf{a}\bar{\mathbf{a}}, 1)/\sqrt{2}$.

The Clifford algebra over the conformal model of $\mathbb{E}^n$ is called *conformal geometric algebra* [8]. The angular and square bracket operators when acted upon Clifford monomials of null vectors, generate an algebra of advanced invariants, called *null bracket algebra* (NBA).

Let $\mathbf{a}_1, \dots, \mathbf{a}_m$ be symbolic null vectors in $\mathbb{R}^{n+1,1}$ generating an NBA. Let $\mathbf{u}$ be a generic symbolic null vector in $\mathbb{R}^{n+1,1}$ that is algebraically independent of the generators of the NBA. For any $\mathbf{a}, \mathbf{b} \in \mathbb{R}^{n+1,1}$, set

$$\partial(\mathbf{a}) := \mathbf{u} \cdot \mathbf{a}, \quad \partial(\mathbf{a} \mathbf{b}) := \partial(\mathbf{a}) \mathbf{b} - \partial(\mathbf{b}) \mathbf{a}. \quad (3.5)$$

For simplicity we have suppressed the subscript $\mathbf{u}$ of the operator ∂ .

Lemma 1. For any null vector $\mathbf{u}$ and vectors $\mathbf{a}, \mathbf{b} \in \mathcal{V}^n$,

$$\mathbf{u} \mathbf{a} \mathbf{u} = 2\partial(\mathbf{a}) \mathbf{u}, \quad \mathbf{u} \mathbf{a} \mathbf{b} \mathbf{u} = 2\partial(\mathbf{a} \mathbf{b}) \mathbf{u}. \quad (3.6)$$

Proof. We only prove the first equality.

$$\mathbf{u} \mathbf{a} \mathbf{u} = (2\mathbf{u} \cdot \mathbf{a} - \mathbf{a} \mathbf{u}) \mathbf{u} = 2(\mathbf{u} \cdot \mathbf{a}) \mathbf{u}.$$

Consider $\langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle$, where the $\mathbf{a}_i$ are null vectors. By (1.1) and (3.6), for any null vector $\mathbf{u}$ algebraically independent of the $\mathbf{a}_i$,

$$\begin{aligned} \langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle &= \frac{\langle (\mathbf{a}_1 \mathbf{u} \mathbf{a}_1) (\mathbf{a}_2 \mathbf{u} \mathbf{a}_2) \cdots (\mathbf{a}_{2k} \mathbf{u} \mathbf{a}_{2k}) \rangle}{2^{2k} (\mathbf{u} \cdot \mathbf{a}_1) (\mathbf{u} \cdot \mathbf{a}_2) \cdots (\mathbf{u} \cdot \mathbf{a}_{2k})} \\ &= \frac{\langle \partial(\mathbf{a}_1 \mathbf{a}_2) \partial(\mathbf{a}_2 \mathbf{a}_3) \cdots \partial(\mathbf{a}_{2k-1} \mathbf{a}_{2k}) \mathbf{u} \mathbf{a}_{2k} \mathbf{a}_1 \rangle}{2\partial(\mathbf{a}_1) \partial(\mathbf{a}_2) \cdots \partial(\mathbf{a}_{2k})}. \end{aligned}$$

Since $\mathbf{u} \cdot \partial(\mathbf{a}\mathbf{b}) = \partial(\mathbf{a})\mathbf{u} \cdot \mathbf{b} - \partial(\mathbf{b})\mathbf{u} \cdot \mathbf{a} = 0$ for any vectors $\mathbf{a}, \mathbf{b}$, we have

$$\langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle = \frac{\langle \partial(\mathbf{a}_1 \mathbf{a}_2) \partial(\mathbf{a}_2 \mathbf{a}_3) \cdots \partial(\mathbf{a}_{2k-1} \mathbf{a}_{2k}) \partial(\mathbf{a}_{2k} \mathbf{a}_1) \rangle}{2\partial(\mathbf{a}_1)\partial(\mathbf{a}_2)\cdots\partial(\mathbf{a}_{2k})}. \quad (3.7)$$

When $\mathbf{a}_i \in \mathcal{N}_{\mathbf{u}}$, i.e., $\partial(\mathbf{a}_i) = -1$, by fixing a vector $\mathbf{v} \in \mathcal{N}_{\mathbf{u}}$ and denoting by $\mathbb{R}^{1,1}$ the 2D subspace spanned by $\mathbf{u}, \mathbf{v}$, we get an orthogonal decomposition $\mathbb{R}^{n+1,1} = \mathbb{R}^{1,1} \oplus \mathbb{R}^n$. Let $\mathbf{p}$ be the orthogonal projection into $\mathbb{R}^n$, and denote

$$\overrightarrow{\mathbf{a}_i \mathbf{a}_j} := \mathbf{p}(\mathbf{a}_j) - \mathbf{p}(\mathbf{a}_i) \in \mathbb{R}^n. \quad (3.8)$$

Then (3.7) becomes

$$\langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle = \frac{1}{2} \langle \overrightarrow{\mathbf{a}_1 \mathbf{a}_2} \overrightarrow{\mathbf{a}_2 \mathbf{a}_3} \cdots \overrightarrow{\mathbf{a}_{2k-1} \mathbf{a}_{2k}} \overrightarrow{\mathbf{a}_{2k} \mathbf{a}_1} \rangle. \quad (3.9)$$

The reason why we use $(\mathbf{u}, \mathbf{v})$ instead of $(\mathbf{e}, \mathbf{e}_0)$ is that $\mathbf{e}$ or $\mathbf{e}_0$ often occurs in an input expression, and setting $\mathbf{u} = \mathbf{e}$ or $\mathbf{v} = \mathbf{e}_0$ destroys the highly important property that for any $1 \leq i \neq j \leq 2k$, $\mathbf{a}_i$ and $\mathbf{a}_j$ occur equal number of times on the right side of (3.7), i.e., on the right side of (3.9).

Similarly, let $\mathbf{I}_n$ be the unit orientation of $\mathbb{R}^n$ defining the square bracket operator in $CL(\mathbb{R}^n)$, then $\mathbf{u} \wedge \mathbf{v} \wedge \mathbf{I}_n$ is the unit orientation of $\mathbb{R}^{n+1,1}$ defining the square bracket operator in $CL(\mathbb{R}^{n+1,1})$, and

$$[\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{n+2l}] = \frac{\langle \partial(\mathbf{a}_1 \mathbf{a}_2) \partial(\mathbf{a}_2 \mathbf{a}_3) \cdots \partial(\mathbf{a}_{n+2l-1} \mathbf{a}_{n+2l}) \rangle_n \cdot \mathbf{I}_n^{-1}}{(-1)^n 2\partial(\mathbf{a}_1)\partial(\mathbf{a}_2)\cdots\partial(\mathbf{a}_{n+2l})}. \quad (3.10)$$

When $\mathbf{a}_i \in \mathcal{N}_{\mathbf{u}}$, (3.10) becomes

$$[\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{n+2l}] = \frac{(-1)^n}{2} [\overrightarrow{\mathbf{a}_1 \mathbf{a}_2} \overrightarrow{\mathbf{a}_2 \mathbf{a}_3} \cdots \overrightarrow{\mathbf{a}_{n+2l-1} \mathbf{a}_{n+2l}}]. \quad (3.11)$$

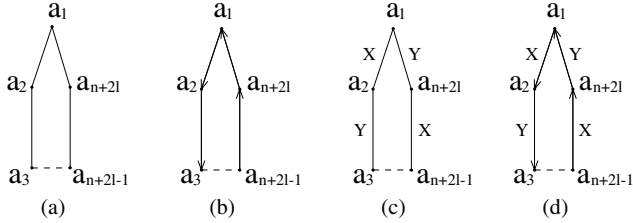


Figure 1: Graph representations of $[\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{n+2l}]$ in the NBA over $\mathbb{R}^{n+1,1}$: (a) $n = 4h - 1$, (b) $n = 4h - 3$, (c) $n = 4h - 2$, (d) $n = 4h$.

(i) By (1.1) and (3.9), $\langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle$ can be represented by a closed walk in the graph of vertices $\mathbf{a}_1, \dots, \mathbf{a}_{2k}$ and containing edges $\{\mathbf{a}_l \mathbf{a}_{l+1} \mid 1 \leq l \leq 2k\}$, where $\mathbf{a}_{2k+1} = \mathbf{a}_1$.

(ii) When $n = 4h - 1$, $[\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{n+2l}]$ is represented by a closed walk in the graph G of vertices $\mathbf{a}_1, \dots, \mathbf{a}_{n+2l}$ and containing edges $\{\mathbf{a}_l \mathbf{a}_{l+1} \mid 1 \leq l \leq n+2l\}$, cf. Figure 1(a).

(iii) When $n = 4h - 3$, graph G must be directed because of the reversion antisymmetry. The square bracket is represented by a closed directed walk, cf. Figure 1(b).

An *XY-graph* refers to an edge-labeled graph with two labels X and Y . A *closed labeled walk* in an *XY-graph* is of the form (label $(\mathbf{a}_1 \mathbf{a}_2), \mathbf{a}_1 \mathbf{a}_2, \mathbf{a}_2 \mathbf{a}_3, \dots, \mathbf{a}_{n+2l} \mathbf{a}_1$), where edges $\mathbf{a}_{2j-1} \mathbf{a}_{2j}$ have the same label, and edges $\mathbf{a}_{2j} \mathbf{a}_{2j+1}$ have the same opposite label, cf. Figure 1(c).

(iv) When $n = 4h - 2$, graph G must be edge-labeled with two labels because of the shift antisymmetry. The square bracket is represented by a closed labeled walk.

(v) When $n = 4h$, graph G must be both edge-labeled and directed. The square bracket is represented by a closed labeled directed walk, cf. Figure 1(d).

When $n = 2$, the *complex bracket* of $\mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k}$, where the $\mathbf{a}_i$ are null vectors in $\mathbb{R}^{3,1}$, is defined by (1.2). If furthermore $\mathbf{a}_i \in \mathcal{N}_{\mathbf{u}}$, by (2.15), (3.9) and (3.11),

$$2\langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle_{\mathbb{C}} = \frac{\langle \overrightarrow{\mathbf{a}_1 \mathbf{a}_2} \overrightarrow{\mathbf{a}_2 \mathbf{a}_3} \cdots \overrightarrow{\mathbf{a}_{2k} \mathbf{a}_1} \rangle_{\mathbb{C}}}{\mathbf{g}(\overrightarrow{\mathbf{a}_1 \mathbf{a}_2} \overrightarrow{\mathbf{a}_2 \mathbf{a}_3} \cdots \overrightarrow{\mathbf{a}_{2k} \mathbf{a}_1})}. \quad (3.12)$$

The following symmetries are consequences of (3.12): let

$$\mathbf{B} = \mathbf{a}_{i_1} \mathbf{a}_{i_2} \cdots \mathbf{a}_{i_{2k-1}}, \quad \mathbf{C} = \mathbf{a}_{j_1} \mathbf{a}_{j_2} \cdots \mathbf{a}_{j_{2h-1}},$$

then for any null vector $\mathbf{a} \in \mathbb{R}^{3,1}$,

$$\begin{aligned} \langle \mathbf{a} \mathbf{B} \rangle_{\mathbb{C}} &= \langle \mathbf{B}^\dagger \mathbf{a} \rangle_{\mathbb{C}} = \overline{\langle \mathbf{B} \mathbf{a} \rangle_{\mathbb{C}}}, \\ \langle \mathbf{a} \mathbf{B} \mathbf{a} \mathbf{C} \rangle_{\mathbb{C}} &= 2\langle \mathbf{a} \mathbf{B} \rangle_{\mathbb{C}} \langle \mathbf{a} \mathbf{C} \rangle_{\mathbb{C}}. \end{aligned} \quad (3.13)$$

4. BALANCED COMPLEX DIFFERENCES

From now on we always assume $n = 2$, and use $\mathbf{a}_i$ to denote the complex representation of the planar point represented by null vector $\mathbf{a}_i \in \mathcal{N}_{\mathbf{u}}$. By (3.12),

$$\langle \mathbf{a}_1 \mathbf{a}_2 \cdots \mathbf{a}_{2k} \rangle_{\mathbb{C}} = \frac{1}{2} \prod_{l=1}^k (\mathbf{a}_{2l-1} - \mathbf{a}_{2l}) (\overline{\mathbf{a}_{2l} - \mathbf{a}_{2l+1}}), \quad (4.1)$$

where $\mathbf{a}_{2k+1} := \mathbf{a}_1$. It provides a complex numbers explanation of the brackets in the NBA of plane geometry.

For example, when $k = 1$, the left side of (4.1) equals $\langle \mathbf{a}_1 \mathbf{a}_2 \rangle = -d_{\mathbf{a}_1 \mathbf{a}_2}^2/2$, and the right side equals

$$\frac{1}{2} (\mathbf{a}_1 - \mathbf{a}_2) (\overline{\mathbf{a}_2 - \mathbf{a}_1}) = -\frac{1}{2} |\mathbf{a}_1 - \mathbf{a}_2|^2.$$

When $k = 2$, the imaginary part of (4.1) is

$$\begin{aligned} [\mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_4] &= \frac{1}{4i} \{ (\mathbf{a}_1 - \mathbf{a}_2) (\mathbf{a}_3 - \mathbf{a}_4) (\overline{\mathbf{a}_2 - \mathbf{a}_3}) (\overline{\mathbf{a}_4 - \mathbf{a}_1}) \\ &\quad - (\overline{\mathbf{a}_1 - \mathbf{a}_2}) (\overline{\mathbf{a}_3 - \mathbf{a}_4}) (\mathbf{a}_2 - \mathbf{a}_3) (\mathbf{a}_4 - \mathbf{a}_1) \}. \end{aligned} \quad (4.2)$$

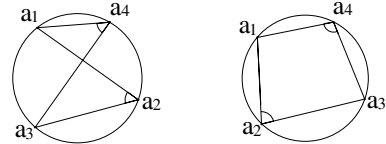


Figure 2: $\angle(\mathbf{a}_1 \mathbf{a}_2, \mathbf{a}_2 \mathbf{a}_3) + \angle(\mathbf{a}_3 \mathbf{a}_4, \mathbf{a}_4 \mathbf{a}_1) = 0$ or π .

By (2.9) and (3.11), the left side of (4.2) equals zero if and only if $\sin(\angle(\mathbf{a}_1 \mathbf{a}_2, \mathbf{a}_2 \mathbf{a}_3) + \angle(\mathbf{a}_3 \mathbf{a}_4, \mathbf{a}_4 \mathbf{a}_1)) = 0$. As shown in Figure 2, this is equivalent to the four planar points represented by $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4$ being either cocircular or collinear. The right side of (4.2) equals zero if and only if the *complex cross-ratio*

$$\frac{(\mathbf{a}_1 - \mathbf{a}_2) (\overline{\mathbf{a}_1 - \mathbf{a}_4})}{(\mathbf{a}_3 - \mathbf{a}_2) (\overline{\mathbf{a}_3 - \mathbf{a}_4})}$$

is real.

Consider the product of several complex brackets.

Definition 1. Let $\mathbf{a}_1, \dots, \mathbf{a}_{2k}$ be a sequence of complex indeterminates such that $\mathbf{a}_{2l-1} \neq \mathbf{a}_{2l}$ for any $1 \leq l \leq k$. Let σ be a permutation of indices $\{1, \dots, 2k\}$. Then

$$\prod_{l=1}^k \lambda(\mathbf{a}_{2l-1} - \mathbf{a}_{2l}) (\overline{\mathbf{a}_{\sigma(2l-1)} - \mathbf{a}_{\sigma(2l)}}) \quad (4.3)$$

is called a *balanced complex difference* (BCD) monomial, where coefficient $\lambda \in \mathbb{C}$. $\mathbf{a}_{2l-1} - \mathbf{a}_{2l}$ is called a *linear factor*, and $\overline{\mathbf{a}_{\sigma(2l-1)} - \mathbf{a}_{\sigma(2l)}}$ is called a *conjugate-linear factor* of the monomial. A $\mathbb{C}$ -linear combination of BCD monomials having the same multiset of complex indeterminates in their linear factors is called a *BCD polynomial*.

An XY -graph is said to be *balanced*, if at every vertex of the graph, the number of incident edges with label X equals that with label Y . BCD monomial (4.3) can be represented up to coefficient by a balanced XY -graph as follows: the $2k$ vertices are the $\mathbf{a}_j$, the k edges $\mathbf{a}_{2l-1}\mathbf{a}_{2l}$ are labeled by X , and the other k edges $\mathbf{a}_{\sigma(2l-1)}\mathbf{a}_{\sigma(2l)}$ are labeled by Y .

In particular, a 2-cycle composed of two edges of the same vertices $\mathbf{a}_u, \mathbf{a}_v$ but opposite labels corresponds to $(\mathbf{a}_u - \mathbf{a}_v)(\overline{\mathbf{a}_u - \mathbf{a}_v}) = -d_{\mathbf{a}_u\mathbf{a}_v}^2/2$. Such a monomial is called an *inner-product pair*.

Theorem 2. Any BCD monomial is the product of one or several complex brackets. It is a complex bracket if and only if its XY -graph is connected. If it is not a complex bracket, then it becomes a complex bracket when multiplied with several inner-product pairs whose number is less than the number of connected components of the graph.

The proof is based on graph theory and is sketched below.

Lemma 3. A balanced XY -graph satisfies the following:

- (1) It has a closed labeled walk.
- (2) It is still balanced if removing a closed labeled walk.
- (3) It is the sum of several closed labeled walks.
- (4) It is a closed labeled walk if and only if it is connected.

Lemma 4. For two closed labeled walks,

- (a) if they have a common vertex, then they can connect at the vertex to form one closed labeled walk;
- (b) else, by choosing two different vertices, one from each walk, they can be connected by the new 2-cycle between the two vertices to become a single closed labeled walk.

Lemma 5. Let the number of edges of a balanced XY -graph be N . There is an algorithm that can change the graph into a closed labeled walk by adding 2-cycles, in $O(N^2)$ transpositions and comparisons of the vertices.

Sketch of the proof of Lemma 5:

Starting from any vertex of nonzero degree in the graph, extend any edge from the vertex to a labeled walk arbitrarily, until the walk comes back to the vertex. In every step of the walk, the edge in use is deleted from the graph, together with any vertex of it that becomes degree-zero after the deletion. The result is a closed labeled walk scraped off the graph.

Scraping off all such closed labeled walks requires $O(N)$ transpositions and comparisons. Connecting them by using Lemma 4 takes $O(N^2)$ transpositions and comparisons.

Example 1.

$$\begin{aligned} & \frac{(\mathbf{a}_1 - \mathbf{a}_2)(\mathbf{a}_3 - \mathbf{a}_4)(\mathbf{a}_5 - \mathbf{a}_6)(\mathbf{a}_7 - \mathbf{a}_8)}{(\mathbf{a}_2 - \mathbf{a}_3)(\mathbf{a}_1 - \mathbf{a}_4)(\mathbf{a}_5 - \mathbf{a}_7)(\mathbf{a}_6 - \mathbf{a}_8)} \\ &= -4\langle \mathbf{a}_1\mathbf{a}_2\mathbf{a}_3\mathbf{a}_4 \rangle_{\mathbb{C}} \langle \mathbf{a}_5\mathbf{a}_6\mathbf{a}_7\mathbf{a}_8 \rangle_{\mathbb{C}} \\ &= -\frac{1}{\mathbf{a}_4 \cdot \mathbf{a}_5} \langle \mathbf{a}_1\mathbf{a}_2\mathbf{a}_3\mathbf{a}_4\mathbf{a}_5\mathbf{a}_6\mathbf{a}_7\mathbf{a}_8 \rangle_{\mathbb{C}}. \end{aligned} \quad (4.4)$$

All BCD polynomials of complex indeterminates $\{\mathbf{a}_i \mid 1 \leq i \leq m\}$ form a commutative monoid by complex multiplication, called a *BCD monoid*, whose identity is integer 1. They

cannot form a ring, because for two BCD polynomials, their sum is a BCD polynomial if and only if they have the same multiset of complex indeterminates in their factors.

BCD monoid has the following basic syzygy relation:

$$(\mathbf{a}_{l_1} - \mathbf{a}_{l_2})(\mathbf{a}_{l_3} - \mathbf{a}_{l_4}) = (\mathbf{a}_1 - \mathbf{a}_{l_3})(\mathbf{a}_{l_2} - \mathbf{a}_{l_4}) - (\mathbf{a}_{l_1} - \mathbf{a}_{l_4})(\mathbf{a}_{l_2} - \mathbf{a}_{l_3}). \quad (4.5)$$

The 3-termed equality has the property that any term determines the other two terms uniquely. The rewriting from the left to the right of (4.5) is called a *BCD split*, and the reverse rewriting is called a *BCD contraction*.

The real and imaginary parts of any BCD monomial $\mathfrak{B}$ are defined to be its *angular* and *square brackets* respectively, called *BCD brackets*:

$$2\langle \mathfrak{B} \rangle := \mathfrak{B} + \overline{\mathfrak{B}}, \quad 2i[\mathfrak{B}] := \mathfrak{B} - \overline{\mathfrak{B}}. \quad (4.6)$$

By (2.9) and (2.16), for BCD monomials $\mathfrak{B}_1, \mathfrak{B}_2$,

$$\begin{aligned} 2\langle \mathfrak{B}_1 \rangle \langle \mathfrak{B}_2 \rangle &= \langle \mathfrak{B}_1 \mathfrak{B}_2 \rangle + \langle \mathfrak{B}_1 \overline{\mathfrak{B}_2} \rangle, \\ 2[\mathfrak{B}_1][\mathfrak{B}_2] &= \langle \mathfrak{B}_1 \overline{\mathfrak{B}_2} \rangle - \langle \mathfrak{B}_1 \mathfrak{B}_2 \rangle, \\ 2\langle \mathfrak{B}_1 \rangle [\mathfrak{B}_2] &= [\mathfrak{B}_1 \mathfrak{B}_2] - [\mathfrak{B}_1 \overline{\mathfrak{B}_2}]. \end{aligned} \quad (4.7)$$

The transformations from the left to the right are called *trigonometric expansions* of brackets, and the reverse transformations are *trigonometric factorizations*. For example, the trigonometric expansions are equivalent to $2\cos\theta_1\cos\theta_2 = \cos(\theta_1 + \theta_2) + \cos(\theta_1 - \theta_2)$ and the like.

The following formulas are used to factorize a bracket:

(i) If $\langle \mathfrak{B}_1 \rangle = 0$, then

$$\langle \mathfrak{B}_1 \mathfrak{B}_2 \rangle = -[\mathfrak{B}_1][\mathfrak{B}_2], \quad [\mathfrak{B}_1 \mathfrak{B}_2] = [\mathfrak{B}_1]\langle \mathfrak{B}_2 \rangle. \quad (4.8)$$

(ii) If $[\mathfrak{B}_1] = 0$, e.g., $\mathfrak{B}_1 = (\mathbf{a}_1 - \mathbf{a}_2)(\overline{\mathbf{a}_2 - \mathbf{a}_3})(\mathbf{a}_3 - \mathbf{a}_4)(\overline{\mathbf{a}_4 - \mathbf{a}_1})$ where the four points $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4$ are cocircular or collinear, or $\mathfrak{B}_1 = (\mathbf{a}_1 - \mathbf{a}_2)(\overline{\mathbf{a}_1 - \mathbf{a}_2})$, then

$$\langle \mathfrak{B}_1 \mathfrak{B}_2 \rangle = \langle \mathfrak{B}_1 \rangle \langle \mathfrak{B}_2 \rangle, \quad [\mathfrak{B}_1 \mathfrak{B}_2] = \langle \mathfrak{B}_1 \rangle [\mathfrak{B}_2]. \quad (4.9)$$

5. NORMALIZATION OF BRACKETS

A BCD monomial can be written as

$$\mathfrak{B} = \lambda P_{\mathfrak{B}} \overline{Q_{\mathfrak{B}}}, \quad (5.1)$$

where $\lambda \in \mathbb{C}$, $P_{\mathfrak{B}}$ is the product of linear factors, and $\overline{Q_{\mathfrak{B}}}$ is the product of conjugate-linear factors.

Let the complex indeterminates $\mathbf{a}_1, \dots, \mathbf{a}_m$ be arbitrarily ordered. In a BCD monomial, if rearranging the positions of the two indeterminates in each linear or conjugate-linear factor, then each linear factor and the conjugate of each conjugate-linear factor can be put in the form of a lower-ordered indeterminate minus a high-ordered one. The signs of the re-orderings are absorbed into the coefficient. A BCD monomial of this form is said to be *ascending*.

Theorem 6. Let $P_1 \overline{Q_1}$ and $P_2 \overline{Q_2}$ be two monic ascending BCD monomials, where any two complex indeterminates are either identical or algebraically independent of each other. Then

1. $\langle P_1 \overline{Q_1} \rangle = \mu \langle P_2 \overline{Q_2} \rangle$ for some $\mu \in \mathbb{R}$ if and only if $\{P_1, Q_1\} = \{P_2, Q_2\}$.

2. $[P_1 \overline{Q_1}] = \mu [P_2 \overline{Q_2}] \neq 0$ for some $\mu \in \mathbb{R}$ if and only if either

(i) $\{P_1, Q_1\} = \{P_2, Q_2\}$, or

(ii) $P_1 \neq Q_1$, $P_2 \neq Q_2$, and there are four complex indeterminates $\mathbf{a}_{l_1} \prec \mathbf{a}_{l_2} \prec \mathbf{a}_{l_3} \prec \mathbf{a}_{l_4}$ and a monomial C composed of ascending linear factors, such that

$$\{P_1, Q_1, P_2, Q_2\} = \{(\mathbf{a}_{l_1} - \mathbf{a}_{l_2})(\mathbf{a}_{l_3} - \mathbf{a}_{l_4})C, (\mathbf{a}_{l_1} - \mathbf{a}_{l_3})(\mathbf{a}_{l_2} - \mathbf{a}_{l_4})C, (\mathbf{a}_{l_1} - \mathbf{a}_{l_4})(\mathbf{a}_{l_2} - \mathbf{a}_{l_3})C\}. \quad (5.2)$$

Let the total degrees of P_1, P_2 be the same, say N . The order among the indeterminates $\mathbf{a}_l$ induces a lexicographic order among the ascending linear factors formed by them, and it takes $O(N \log N)$ transpositions to arrange the linear factors in P_1 (or P_2) in non-decreasing order. The identification of either N or $N - 2$ elements between the two non-decreasing sequences of P_1 and P_2 takes $O(N)$ comparisons. As a corollary of Theorem 7, deciding the congruence of two brackets takes $O(N \log N)$ transpositions and comparisons.

Lemma 7. Let $P\overline{Q}$ be a monic ascending BCD monomial. Then (a) $\langle P\overline{Q} \rangle \neq 0$, (b) $[P\overline{Q}] = 0$ if and only if $P = Q$.

Proof. Since $\mathbb{Z}[\{\mathbf{a}_l, \overline{\mathbf{a}}_l \mid 1 \leq l \leq m\}]$ is a UFD, if $\langle P\overline{Q} \rangle = 0$, i.e., $P\overline{Q} = -Q\overline{P}$, then $P = Q$ because they are monic and ascending. Then $P\overline{Q} = 0$, which is impossible. The second statement can be proved similarly.

For a polynomial $\mathfrak{B}$ in variables $\{\mathbf{a}_l, \overline{\mathbf{a}}_l \mid 1 \leq l \leq m\}$, its *holomorphization* is the following holomorphic polynomial:

$$\mathfrak{M}(\mathfrak{B}) := \mathfrak{B} \big|_{\overline{\mathbf{a}}_l = \mathbf{a}_l, l=1..m}. \quad (5.3)$$

Proof of Theorem 6.

1. $\langle P_1\overline{Q_1} \rangle = \mu \langle P_2\overline{Q_2} \rangle$ can be written as

$$P_1\overline{Q_1} + Q_1\overline{P_1} - \mu P_2\overline{Q_2} - \mu Q_2\overline{P_2} = 0. \quad (5.4)$$

The holomorphization of the left side is $2(P_1Q_1 - \mu P_2Q_2)$. So $P_1Q_1 = \mu P_2Q_2$, and in fact $\mu = 1$. Then (5.4) becomes

$$P_1\overline{Q_1} - P_2\overline{Q_2} = -\overline{(P_1\overline{Q_1} - P_2\overline{Q_2})}. \quad (5.5)$$

Substituting $Q_2 = P_1Q_1/P_2$ into it, we get that

$$i(P_1\overline{P_2} - \overline{P_1P_2})\overline{Q_1}/\overline{P_2} \quad (5.6)$$

is real-valued. So either $P_1\overline{P_2}$ or Q_1/P_2 is real-valued. In the former case, $[P_1\overline{P_2}] = 0$, and by Lemma 7, $P_1 = P_2$; in the latter case, $Q_1 = P_2$.

2. $[P_1\overline{Q_1}] = \mu[P_2\overline{Q_2}]$ can be written as

$$P_1\overline{Q_1} - Q_1\overline{P_1} - \mu P_2\overline{Q_2} + \mu Q_2\overline{P_2} = 0. \quad (5.7)$$

We only need to consider the case where P_1, Q_1, P_2, Q_2 are *coprime*, i.e., they have no common factors.

Statement 1: P_1 and Q_1 are coprime, so are P_2 and Q_2 .

Proof of the statement. If $L = \mathbf{a}_1 - \mathbf{a}_2$ is a common linear factor of P_2 and Q_2 , by setting $\mathbf{a}_1 = \mathbf{a}_2$ in (5.7), we get $(P_1 - Q_1)|_{\mathbf{a}_1=\mathbf{a}_2} = 0$, so $L \mid (P_1 - Q_1)$. Let $Q_1 = P_1 + LR$, where R is a holomorphic polynomial. Substituting it into (5.7), we get $\overline{LR}P_1 = LU$, where U is a non-holomorphic polynomial. So $L \mid P_1$, hence $L \mid Q_1$ by (5.7), contradicting with the assumption that P_1, Q_1, P_2, Q_2 are coprime.

Statement 2: P_1, Q_1, P_2, Q_2 have the same degree in any complex indeterminate $\mathbf{a}_l$. The proof is omitted.

Statement 3: If $P_1Q_1 = P_2Q_2$, then $\{P_1, Q_1\} = \{P_2, Q_2\}$. The proof is omitted.

Statement 4: Let $L = \mathbf{a}_1 - \mathbf{a}_2$. If for some $r > 0$, $L^r \mid P_1Q_1$ but $L^{r+1} \nmid P_1Q_1$, then either (a) $L \nmid P_2Q_2$, or (b) $L^r \mid P_2Q_2$ but $L^{r+1} \nmid P_2Q_2$. The proof is omitted.

Notation: By Statement 2, $\mathbf{a}_l$ ($1 \leq l \leq m$) occurs the same number of times in the linear factors of any of P_1, Q_1, P_2, Q_2 , denoted by t_l . Let $2N$ be the sum of t_l over $1 \leq l \leq m$. Then P_1, Q_1, P_2, Q_2 each have total degree N . Let F be the result of multilinearly expanding the left side of (5.7) but not combining like terms.

Statement 5: If $P_1Q_1 \neq P_2Q_2$, then (5.2) holds.

Proof. Now that $P_1Q_1 \neq P_2Q_2$, without loss of generality, assume that it is $L = \mathbf{a}_1 - \mathbf{a}_2$ such that $L^r \mid Q_1$ but $L^r \nmid P_2Q_2$ for some $r > 0$. By Statements 1 and 4, $L \nmid P_1P_2Q_2$.

All the terms in F that are multiples of $\mathbf{a}_1^{t_1}\mathbf{a}_2^{t_2}\overline{\mathbf{a}}_1^{t_1}\overline{\mathbf{a}}_2^{t_2}$ are from $-\mu P_2\overline{Q_2} + \mu Q_2\overline{P_2}$, and the sum of these terms equals $-\mu \mathbf{a}_1^{t_1}\mathbf{a}_2^{t_2}\overline{\mathbf{a}}_1^{t_1}\overline{\mathbf{a}}_2^{t_2}(P_2'\overline{Q_2'} - Q_2'\overline{P_2'})$. So $P_2'\overline{Q_2'} = Q_2'\overline{P_2'}$; hence $P_2' = Q_2' = 1$, and $t_1 + t_2 = N$.

Since $(\mathbf{a}_1 - \mathbf{a}_2) \nmid P_1P_2Q_2$, every linear factor of P_1, P_2, Q_2 must contain either $\mathbf{a}_1$ or $\mathbf{a}_2$. At least one linear factor of Q_1 , say $\mathbf{a}_3 - \mathbf{a}_4$, contains neither $\mathbf{a}_1$ nor $\mathbf{a}_2$. Similar to the arguments on factor $\mathbf{a}_1 - \mathbf{a}_2$, it can be proved that $t_3 + t_4 = N$, and every linear factor of P_1, P_2, Q_2 must contain either $\mathbf{a}_3$ or $\mathbf{a}_4$, but cannot contain both. Let $\mathbf{a}_1 \prec \mathbf{a}_2 \prec \mathbf{a}_3 \prec \mathbf{a}_4$.

Now all factors of P_1, P_2, Q_2 must be among

$$\{\mathbf{a}_1 - \mathbf{a}_3, \mathbf{a}_1 - \mathbf{a}_4, \mathbf{a}_2 - \mathbf{a}_3, \mathbf{a}_2 - \mathbf{a}_4\}. \quad (5.8)$$

Since P_2, Q_2 are coprime, they each have two and only two of the four kinds of factors. Say $(\mathbf{a}_1 - \mathbf{a}_3) \nmid Q_2$. Then for some k_1, k_2, k_3, k_4 ,

$$(\mathbf{a}_1 - \mathbf{a}_4)^{k_1}(\mathbf{a}_2 - \mathbf{a}_3)^{k_2} = Q_2, \quad (\mathbf{a}_1 - \mathbf{a}_3)^{k_3}(\mathbf{a}_2 - \mathbf{a}_4)^{k_4} = P_2.$$

So $k_l = t_l = N/2$ for $1 \leq l \leq 4$, and N is even.

If any of (5.8), say $\mathbf{a}_1 - \mathbf{a}_3$, is in Q_1 , then it cannot be in P_1 , so $(\mathbf{a}_1 - \mathbf{a}_3)^{N/2} \nmid P_1Q_1$ but $(\mathbf{a}_1 - \mathbf{a}_3)^{N/2} \mid P_2$; by Statement 4, $(\mathbf{a}_1 - \mathbf{a}_3) \nmid Q_1$, contradiction. Thus we get

$$Q_1 = (\mathbf{a}_1 - \mathbf{a}_2)^{N/2}(\mathbf{a}_3 - \mathbf{a}_4)^{N/2}.$$

Last, consider P_1 . If $(\mathbf{a}_1 - \mathbf{a}_3)^{N/2} \nmid P_1$, by Statement 4, $(\mathbf{a}_1 - \mathbf{a}_3) \nmid P_1$, so $P_1 = Q_2$; else, $P_1 = P_2$. By symmetry, consider only the latter case. Substituting $P_2 = P_1$ into (5.7), we get $P_1(\overline{Q_1} - \mu\overline{Q_2}) = \overline{P_1}(Q_1 - \mu Q_2)$, so $P_1 = \tau(Q_1 - \mu Q_2)$ for some $\tau \in \mathbb{C}$. Substituting into it the expressions of P_1, Q_1, Q_2 , and using BCD split (4.5), we get

$$\begin{aligned} & \{(\mathbf{a}_1 - \mathbf{a}_2)(\mathbf{a}_3 - \mathbf{a}_4) + (\mathbf{a}_1 - \mathbf{a}_4)(\mathbf{a}_2 - \mathbf{a}_3)\}^{N/2} \\ &= \tau((\mathbf{a}_1 - \mathbf{a}_2)(\mathbf{a}_3 - \mathbf{a}_4))^{N/2} - \mu\tau((\mathbf{a}_1 - \mathbf{a}_4)(\mathbf{a}_2 - \mathbf{a}_3))^{N/2}. \end{aligned}$$

As $\mathbf{a}_1$ to $\mathbf{a}_4$ are generic, we deduce $\tau = 1 = -\mu$ and $N = 2$.

Remark: Theorem 6 implies that the translation of an NBA bracket B into a BCD bracket is unique up to a complex conjugate except for the case $B = [\mathbf{a}_1\mathbf{a}_2\mathbf{a}_3\mathbf{a}_4]$, which has three different translations, and the equality between any two is by a BCD split. When there are many BCD split candidates, since each split increases the expression size, an optimal translation is needed for a bracket polynomial involving classical brackets.

A BCD polynomial is said to be *equi-modular*, if for any two terms $\mathfrak{B}_1, \mathfrak{B}_2$ of it, $\mathfrak{M}(\mathfrak{B}_1) = \mathfrak{M}(\mathfrak{B}_2)$ up to coefficient. All equi-modular BCD polynomials (MBCDs) form a submonoid of the BCD monoid. It turns out that translating a homogeneous bracket polynomial into an MBCD is beneficial to generating like terms and making factorization.

Proposition 8. Let $\mathfrak{B}_1 + \mathfrak{B}_2$ be a 2-termed MBCD. Let $\mathfrak{B}_j = \lambda_j P_j \overline{Q_j}$ for $j = 1, 2$ be ascending, where P_1 and P_2 are coprime, and Q_1 and Q_2 are coprime. Then $P_1 = Q_2$, $P_2 = Q_1$, and $P\overline{Q}$ and $P_1\overline{Q_1}$ are both BCD monomials.

Proof. Since $P_1Q_1 = P_2Q_2$, by (??), $P_1 = Q_2$ and $P_2 = Q_1$. PP_1 and QQ_1 have the same degree in any complex indeterminate $\mathbf{a}_i$, so are PQ_1 and QP_1 . So P_1, Q_1 have the same degree in $\mathbf{a}_i$, and the same is true for P, Q .

Example 2. Proposition 8 when used together with trigonometric factorizations (4.7), can be used to factorize a bracket of an MBCD polynomial into the product of several brackets of BCD polynomials. In the following expression, $\mathbf{e}, 1, 2, 3, 4$ are complex numbers representations of $\mathbf{e}, 1, 2, 3, 4 \in \mathcal{N}_u$ respectively:

$$\begin{aligned} & 2^{-5}(\mathbf{e}-1)(\mathbf{e}-2)(\mathbf{e}-3)(\mathbf{e}-4)\overline{(\mathbf{e}-1)(\mathbf{e}-2)(\mathbf{e}-3)(\mathbf{e}-4)} \\ & \{ -\langle (\mathbf{e}-2)(\mathbf{e}-4)(1-3)(1-4)(2-3) \rangle \\ & \quad \overline{(\mathbf{e}-1)(\mathbf{e}-3)(1-2)(2-4)(3-4)} \rangle \\ & -\langle (\mathbf{e}-1)(\mathbf{e}-4)(1-2)(2-3)(3-4) \rangle \\ & \quad \overline{(\mathbf{e}-2)(\mathbf{e}-3)(1-3)(1-4)(2-4)} \rangle \\ & -\langle (\mathbf{e}-1)(\mathbf{e}-2)(1-4)(2-3)(3-4) \rangle \\ & \quad \overline{(\mathbf{e}-3)(\mathbf{e}-4)(1-2)(1-3)(2-4)} \rangle \\ & +\langle (\mathbf{e}-1)(\mathbf{e}-4)(1-3)(2-3)(2-4) \rangle \\ & \quad \overline{(\mathbf{e}-2)(\mathbf{e}-3)(1-2)(1-4)(3-4)} \rangle \\ & +\langle (\mathbf{e}-3)(\mathbf{e}-4)(1-2)(1-4)(2-3) \rangle \\ & \quad \overline{(\mathbf{e}-1)(\mathbf{e}-2)(1-3)(2-4)(3-4)} \rangle \\ & +\langle (\mathbf{e}-1)(\mathbf{e}-3)(1-4)(2-3)(2-4) \rangle \\ & \quad \overline{(\mathbf{e}-2)(\mathbf{e}-4)(1-2)(1-3)(3-4)} \rangle \}. \end{aligned} \quad (5.9)$$

Denote the last factor of (5.9) by f . It is the angular bracket of an MBCD. By (4.7), each pair of terms of f can be factorized into two bracket factors, so $C_6^2 \times 2 = 30$ such brackets can be generated. Among them, there are three brackets corresponding to the same classical bracket **[1234]**, in the following three of its six BCD bracket forms:

$$\begin{aligned} 2[\mathbf{1234}] &= [(1-4)(2-3)\overline{(1-2)(3-4)}] \\ &= -[(1-2)(3-4)\overline{(1-3)(2-4)}] \\ &= [(1-4)(2-3)\overline{(1-3)(2-4)}]. \end{aligned} \quad (5.10)$$

There are another three brackets corresponding to the same classical bracket **[e123]**. No three of the 24 brackets left correspond to the same classical bracket anymore.

The three forms in (5.10) are generated by the following pairing in f : terms 1 and 6, 2 and 5, 3 and 4. The result by applying (4.7) to the three pairs is

$$\begin{aligned} f &= 2\{ [(1-4)(2-3)\overline{(1-2)(3-4)}] \\ & \quad \overline{[(\mathbf{e}-2)(\mathbf{e}-4)(1-3)(\mathbf{e}-1)(\mathbf{e}-3)(2-4)]} \\ & + [(1-2)(3-4)\overline{(1-3)(2-4)}] \\ & \quad \overline{[(\mathbf{e}-1)(\mathbf{e}-4)(2-3)(\mathbf{e}-2)(\mathbf{e}-3)(1-4)]} \\ & + [(1-4)(2-3)\overline{(1-3)(2-4)}] \\ & \quad \overline{[(\mathbf{e}-1)(\mathbf{e}-2)(3-4)(\mathbf{e}-3)(\mathbf{e}-4)(1-2)]} \} \\ &= 2^2[\mathbf{1234}] \\ & \quad \{ [(\mathbf{e}-2)(\mathbf{e}-4)(1-3)(\mathbf{e}-1)(\mathbf{e}-3)(2-4)] \\ & \quad + [(\mathbf{e}-2)(\mathbf{e}-3)(1-4)(\mathbf{e}-1)(\mathbf{e}-4)(2-3)] \\ & \quad + [(\mathbf{e}-1)(\mathbf{e}-2)(3-4)(\mathbf{e}-3)(\mathbf{e}-4)(1-2)] \}. \end{aligned} \quad (5.11)$$

The last factor in (5.11), denoted by g , is not an MBCD;

its factorization is more involved. We first need a BCD split

$$(\mathbf{e}-2)(1-3) = (\mathbf{e}-1)(2-3) + (\mathbf{e}-3)(1-2) \quad (5.12)$$

to change g into 4 terms, then apply the following two BCD contractions

$$\begin{aligned} (\mathbf{e}-1)(2-4) - (\mathbf{e}-2)(1-4) &= -(\mathbf{e}-4)(1-2), \\ (\mathbf{e}-3)(2-4) - (\mathbf{e}-2)(3-4) &= -(\mathbf{e}-4)(2-3) \end{aligned} \quad (5.13)$$

to its two pairs of terms respectively. The procedure is

$$\begin{aligned} g &= [(\mathbf{e}-1)(\mathbf{e}-4)(2-3)\overline{(\mathbf{e}-1)(\mathbf{e}-3)(2-4)}] \\ & + [(\mathbf{e}-3)(\mathbf{e}-4)(1-2)\overline{(\mathbf{e}-1)(\mathbf{e}-3)(2-4)}] \\ & + [(\mathbf{e}-2)(\mathbf{e}-3)(1-4)\overline{(\mathbf{e}-1)(\mathbf{e}-4)(2-3)}] \\ & + [(\mathbf{e}-1)(\mathbf{e}-2)(3-4)\overline{(\mathbf{e}-3)(\mathbf{e}-4)(1-2)}] \\ & \stackrel{\text{contract}}{=} (\mathbf{e}-4)\overline{(\mathbf{e}-4)}[(\mathbf{e}-3)(1-2)\overline{(\mathbf{e}-1)(2-3)}] \\ & \quad - [(\mathbf{e}-1)(2-3)\overline{(\mathbf{e}-3)(1-2)}] \\ & \stackrel{\text{translate}}{=} -2^3(\mathbf{e} \cdot 4)[\mathbf{e123}]. \end{aligned} \quad (5.14)$$

The factorization of (5.11) is not successful if starting with a BCD split of any of $(\mathbf{e}-4)(1-2)$, $(\mathbf{e}-4)(1-3)$, $(\mathbf{e}-4)(2-3)$.

Theorem 9. Any BCD polynomial when multiplied with some inner products and classical brackets of null vectors, can be translated into an MBCD.

Proof of Theorem 9. By induction, we only need to prove the theorem for a BCD binomial $\mathfrak{B}_1 + \mathfrak{B}_2$. Let G_j for $j = 1, 2$ be the graph whose vertices are the complex indeterminates and whose edges are the factors in $\mathfrak{B}_j$. Label all edges in G_1 by X and all edges in G_2 by Y .

The *algebraic sum* $H_1 + H_2$ of any two XY -graphs H_1, H_2 is defined as an XY -graph whose vertices are the union of those of H_1, H_2 , and whose edges are the *algebraic sum* of those of H_1, H_2 , in the sense that if in H_j , an edge E is labeled r_j times by X and s_j times by Y , then in $H_1 + H_2$, if $t = r_1 + r_2 - s_1 - s_2 \geq 0$, E occurs t times and with label X , else it occurs $-t$ times and with label Y .

Consider XY -graph $G_1 + G_2$. It is balanced and has no labeled 2-cycle. $\mathfrak{B}_1 + \mathfrak{B}_2$ is an MBCD if and only if $G_1 + G_2$ is a graph without any edge. By Lemma 3, $G_1 + G_2$ is the algebraic sum of closed labeled walks. By induction, we only need to consider one closed labeled walk W in $G_1 + G_2$.

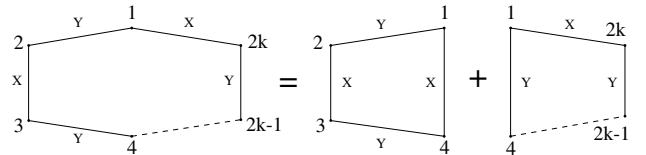


Figure 3: Decomposition of a closed labeled walk.

As shown in Figure 3, W is the algebraic sum of labeled 4-cycles, and every decomposition of a labeled $(2k)$ -cycle into a labeled 4-cycle and a labeled $(2k-2)$ -cycle where $k > 2$, can be realized by multiplying $\mathfrak{B}_1 + \mathfrak{B}_2$ with an inner-product pair.

As shown in Figure 4, any labeled 4-cycle 12341 is the algebraic sum of two single-labeled 4-cycles of the same vertices but different labels. The latter two 4-cycles are graphs of the same bracket **[1234]**, and the decomposition can be realized by multiplying $\mathfrak{B}_1 + \mathfrak{B}_2$ with the bracket. With all such multiplications, every edge in $G_1 + G_2$ can be canceled.

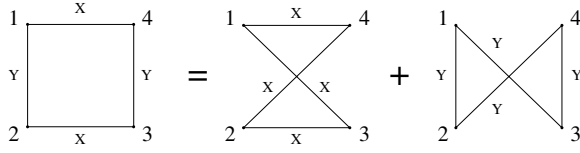


Figure 4: Decomposition of a labeled 4-cycle.

6. GEOMETRIC APPLICATIONS

BCD and MBCD can be used together with NGC and NBA in geometric theorem proving. A general program of making symbolic geometric computing by algebraic simplifications consists of the following steps:

Input: A geometric configuration formed by a sequence of constructions each having an NGC expression; a conclusion expression *conc* in NBA.

Output: Every algebraic manipulation made to *conc*.

Step 1. Eliminations in *conc* by substituting NGC expressions of the constructions into it and making Cayley expansions. In the end of this step, *conc* is a bracket polynomial in NBA.

Step 2. Translation of *conc* from NBA to MBCD.

Step 3. Simplifications of *conc* in MBCD, including bracket-factor extractions, BCD factorizations and contractions, trigonometric expansions and factorizations, etc.

Step 4. Translation of *conc* from MBCD back to NBA.

Example 3. [Simson's Theorem, Figure 5(a)] Let **1, 2, 3, 4** be free points in the plane, **5, 6, 7** be feet drawn from **4** to lines **12, 23, 31** respectively. Then if **1, 2, 3, 4** are cocircular, points **5, 6, 7** must be collinear.

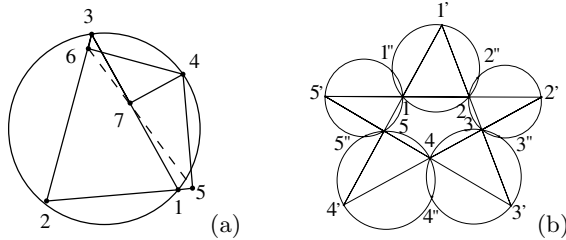


Figure 5: (a): Simson's Theorem; (b): Miguel's 5-Circle Theorem.

First proof. Use $[1234] = 0$ to prove $[e567] = 0$.

Intersections **5, 6, 7** have the following NGC expressions:

$$\begin{aligned} 5 &= \langle e24e12 \rangle 1 - \langle e14e12 \rangle 2 \mod e, \\ 6 &= \langle e34e23 \rangle 2 - \langle e24e23 \rangle 3 \mod e, \\ 7 &= \langle e14e31 \rangle 3 - \langle e34e31 \rangle 1 \mod e. \end{aligned} \quad (6.1)$$

Substituting them into conclusion expression $[e567]$, we get

$$[e567] = [e123](\langle e24e12 \rangle \langle e34e23 \rangle \langle e14e31 \rangle - \langle e14e12 \rangle \langle e24e23 \rangle \langle e34e31 \rangle). \quad (6.2)$$

Translating the last factor on the right side of (6.2) into BCD and then making trigonometric expansions (4.7), result in a 6-termed MBCD which is just (5.9).

As $[1234] = 0$, we can use (4.9) to extract factor $\langle 1234 \rangle$ out of each term of (5.9). After the extractions, the six terms cancel pairwise and the proof is finished.

Second proof. Remove $[1234] = 0$ from the hypotheses, factorize $[e567]$ to get factor $[1234]$. The procedure from (5.9) on is already shown in Example 2.

Example 4. [Miguel's 5-Circle Theorem] In Figure 5(b), the five circles are circumcircles of the five triangular wedges of a 5-star; they meet sequentially at $1'', 2'', 3'', 4'', 5''$ respectively. Then the five points are cocircular.

In [6], the configuration is constructed by taking **1, 2, 3, 4, 5** as free points, constructing $1', 2', 3', 4', 5'$ as intersections of lines, then constructing $1'', 2'', 3'', 4'', 5''$ as intersections of circles. By symmetry, proving $[1''2''3''4''5''] = 0$ is sufficient. After all the eliminations, we get that the conclusion is equivalent to

$$\begin{aligned} &-4(2 \cdot 3)(4 \cdot 5)[e215][e34e51] \\ &+4(2 \cdot 3)(1 \cdot 5)[e345][e12e45] \\ &+ [e154][12345e23e5] = 0. \end{aligned} \quad (6.3)$$

In [6], it takes five nontrivial steps to establish (6.3). Later on, four new proofs were discovered, in which establishing (6.3) takes two to three easy steps of null contractions.

When translating the left side of (6.3) into an MBCD, we get six terms that cancel pairwise, so (6.3) is trivially established. That we adopt the translations of $[e215]$ and $[e154]$ instead of $[e125]$ and $[e145]$ is the result of another algorithm based on graph theory, which is a complete algorithm for the MBCD translation problem, but has no room to be presented here.

The simplification algorithm was tested by over 20 examples on Euclidean geometric theorem proving, which have been tested by the method of [6] but are not given any monomial proofs. In the proofs generated by the MBCD method, the expression size is bigger but still within 10 terms. The benefit is the commutativity of complex numbers, and systematic graph-theoretic considerations of bracket expressions.

7. REFERENCES

- [1] Chou, S.-C., Gao, X.-S. and Zhang, J.-Z. *Machine Proofs in Geometry*. World Scientific, Singapore, 1994.
- [2] Crapo, H. and Richter-Gebert, J. Automatic Proving of Geometric Theorems. In [10], pp. 107–139, 1994.
- [3] Crapo, H. and Senato D. (eds.), *Alg. Combinatorics and Computer Science*, Springer, Milano, pp. 151–196, 2001.
- [4] Havel, T.F. Geometric Algebra and Möbius Sphere Geometry as a Basis for Euclidean Invariant Theory. In [10], pp. 245–256, 1995.
- [5] Hestenes, D. and Sobczyk, G. *Clifford Algebra to Geometric Calculus*. D. Reidel, Dordrecht, Boston, 1984.
- [6] Li, H. A Recipe for Symbolic Geometric Computing. In: *Proc. ISSAC 2007*, pp. 261–268.
- [7] B. Mourrain and N. Stolfi, Computational Symbolic Geometry. In [10], pp. 107–139, 1995.
- [8] Sommer, G. (ed). *Geometric Computing with Clifford Algebras*. Springer, Heidelberg, 2001.
- [9] D. Wang. Geometric Reasoning with Geometric Algebra. In: *Geometric Algebra with Appl. in Science and Engineering*, pp. 89–109, Birkhauser, Boston, 2001.
- [10] White, N. (ed). *Invariant Methods in Discrete and Computational Geometry*. D. Reidel, Dordrecht, 1995.
- [11] Yaglom, M. *Complex Numbers in Geometry*. Academic, London, 1968.

The Complete Root Classification of a Parametric Polynomial on an Interval

Songxin Liang
Department of Applied
Mathematics, The University
of Western Ontario, London,
Ontario, Canada
sliang22@uwo.ca

David J. Jeffrey
Department of Applied
Mathematics, The University
of Western Ontario, London,
Ontario, Canada
djeffrey@uwo.ca

Marc Moreno Maza
Department of Computer
Science, The University of
Western Ontario, London,
Ontario, Canada
moreno@csd.uwo.ca

ABSTRACT

Given a real parametric polynomial $p(x)$ and an interval $(a, b) \subset \mathbb{R}$, the Complete Root Classification (CRC) of $p(x)$ on (a, b) is a collection of all possible cases of its root classification on (a, b) , together with the conditions its coefficients must satisfy for each case. In this paper, a new algorithm is proposed for the automatic computation of the complete root classification of a parametric polynomial on an interval. As a direct application, the new algorithm is applied to some real quantifier elimination problems.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms—*Algebraic algorithms*

General Terms

Algorithms

Keywords

Complete root classification, real root, parametric polynomial, interval, real quantifier elimination

1. INTRODUCTION

The counting and classifying of the roots of a polynomial have been the subject of many investigations. This paper concerns the complete root classification of a parametric polynomial on an interval.

RC and CRC. Let $p(x)$ be a real polynomial with constant coefficients. The *root classification (RC)* of $p(x)$ on $\mathbb{R}$ is denoted by

$$[L_1, L_2] = [[n_1, n_2, \dots], [m_1, -m_1, m_2, -m_2, \dots]],$$

where n_k are the multiplicities of the distinct real roots of $p(x)$ on $\mathbb{R}$, and m_k are the multiplicities of the distinct complex conjugate pairs of $p(x)$, and $L_1 = [n_1, n_2, \dots]$ is called

the *real RC* of $p(x)$ on $\mathbb{R}$. Let $a, b \in \mathbb{R} \cup \{-\infty, +\infty\}$. The *RC* of $p(x)$ on (a, b) is denoted by a list $L = [n_1, n_2, \dots]$, where $n_1, n_2, \dots$ are the multiplicities of the distinct real roots of $p(x)$ on (a, b) . For a real polynomial $p(x)$ with parametric coefficients, the *complete root classification (CRC)* of $p(x)$ on (a, b) is a collection of all possible cases of its RC on (a, b) , together with the conditions its coefficients must satisfy for each case.

The history of CRC is short. The CRC of a real parametric quartic polynomial on $\mathbb{R}$ was found by Arnon in 1988 [2]; the first method for establishing the CRC of a real parametric polynomial of any degree on $\mathbb{R}$ was given by Yang, Hou and Zeng in 1996 [10]. They illustrated their method by computing the CRC of a reduced sextic polynomial. The first automatic generation of CRCs was described and implemented by Liang and Zhang [7], with some improvements added in [5]. Further improvements to the algorithm were made in [6] by replacing the ‘revised sign lists’ (Definition 4 below) with the direct use of ‘sign lists’. As well as offering greater efficiency, the new algorithm offers a better filter for eliminating non-realizable conditions.

All works above are on $\mathbb{R}$, and applications often need CRC on an interval. For example, in robust control [1] and problems concerning program termination [12], we have to determine the conditions on the parametric coefficients of $p(x)$ such that $\forall x > 0, p(x) > 0$, or the conditions such that $\forall x \in (a, b), p(x) \neq 0$. Therefore, it is meaningful to develop an algorithm for computing the CRC of a parametric polynomial on an interval.

However, in order to develop such an algorithm, we have to face two challenging problems. The first problem is the determination of the conditions for a parametric polynomial having a given number of real roots on an interval. One naturally thinks of the well-known Sturm sequence. The Sturm sequence of a polynomial with known, constant coefficients is a good tool for computing the number of real roots on an interval, but it is inconvenient and inefficient when the given polynomial has parametric coefficients. A better solution uses the fact that we know how to determine the conditions for a polynomial having a given number of real roots on $\mathbb{R}$ [6], and converts the problem of the determination of conditions on an interval into a problem on $\mathbb{R}$. This is done in Section 3, where Theorem 4 is given. Let $p \in \mathbb{R}[x]$ with $p(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_0$ and $a_n \neq 0$. Let $a, b \in \mathbb{R}$ such that $p(a) \neq 0$ and $p(b) \neq 0$. Let $\Psi_1(x) = (1 - x)^n p\left(\frac{b - ax}{1 - x}\right)$ and $\Psi_2(x) = \Psi_1(-x^2) =$

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC’08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

$(1+x^2)^n p\left(\frac{b+ax^2}{1+x^2}\right)$. Then $L = [r_1, r_2, \dots, r_k]$ is the RC of $p(x)$ on (a, b) , if and only if $L_2 = [r_1, r_1, r_2, r_2, \dots, r_k, r_k]$ is the real RC of $\Psi_2(x)$ on $\mathbb{R}$. Therefore, the conditions for $p(x)$ having L as its RC on an interval can be obtained by computing the conditions for $\Psi_2(x)$ having L_2 as its real RC on $\mathbb{R}$.

The second problem is the computation of the Δ -sequence of Ψ_2 (Definition 5). We try to determine the conditions for Ψ_2 having L_2 as its real RC on $\mathbb{R}$. The set of all possible sign lists of Ψ_2 can be determined by Theorem 2 and 3. Now, in order to make the multiplicities of the $2k$ distinct real roots of Ψ_2 be $r_1, r_1, r_2, r_2, \dots, r_k, r_k$ respectively, we also have to determine the possible sign lists of the polynomials in the Δ -sequence of Ψ_2 . According to Proposition 1, $\Delta^1(\Psi_2)$ can be determined by the maximal index ℓ of non-vanishing members in the sign list of Ψ_2 which actually is the total number of distinct (real and complex) roots of Ψ_2 . Since L_2 does not contain information about the number of distinct complex-conjugate roots of Ψ_2 , the maximal index ℓ is not uniquely determined. Therefore, unlike the case of RC on $\mathbb{R}$ [6], there may be more than one $\Delta^1(\Psi_2)$ for the real RC L_2 , and consequently the conditions for $\Psi_2(x)$ having L_2 as its real RC on $\mathbb{R}$ would be more complicated. So the question is how to determine these $\Delta^1(\Psi_2)$ and corresponding conditions.

In this paper, a new algorithm for the automatic computation of the CRC of a parametric polynomial on an interval is proposed. The new algorithm has been implemented in MAPLE. As an immediate application, the new algorithm has been applied to some real quantifier elimination problems. However, it should be emphasized that the CRC of a parametric polynomial on an interval contains more information than is needed for these problems, and consequently it has more potential applications than the examples given here.

2. PRELIMINARY

In this section, we review some definitions and theorems which mainly come from [10] and [6]. They are necessary for the new algorithm. Let $p(x) \in \mathbb{R}[x]$ with $p(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_0$ and $a_n \neq 0$.

DEFINITION 1. The $2n \times 2n$ matrix M

$$\begin{pmatrix} a_n & a_{n-1} & a_{n-2} & \dots & a_0 & \\ 0 & na_n & (n-1)a_{n-1} & \dots & a_1 & \\ & a_n & a_{n-1} & \dots & a_1 & a_0 \\ & 0 & na_n & \dots & 2a_2 & a_1 \\ & & & \dots & \dots & \\ & & & \dots & \dots & \\ & & & & a_n & a_{n-1} & \dots & a_0 \\ & & & & 0 & na_n & \dots & a_1 \end{pmatrix}$$

is called the discrimination matrix of p .

DEFINITION 2. For $1 \leq k \leq 2n$, let M_k be the k th principal minor of M , and let $D_k = M_{2k}$. The n -tuple $D = [D_1, D_2, \dots, D_n]$ is called the discriminant sequence of p .

DEFINITION 3. If $\text{sgn } x$ is the signum function, $\text{sgn } 0 = 0$, then the list $[s_1, s_2, \dots, s_n] = [\text{sgn } D_1, \text{sgn } D_2, \dots, \text{sgn } D_n]$ is called the sign list of p .

DEFINITION 4. The revised sign list $[e_1, e_2, \dots, e_n]$ of $p(x)$ is constructed from the sign list $s = [s_1, s_2, \dots, s_n]$ of p as

follows. If $[s_i, s_{i+1}, \dots, s_{i+j}]$ is a section of s , where $s_i \neq 0$, $s_{i+1} = s_{i+2} = \dots = s_{i+j-1} = 0$ and $s_{i+j} \neq 0$, then we replace the subsection $[s_{i+1}, \dots, s_{i+j-1}]$ by

$$[-s_i, -s_i, s_i, s_i, -s_i, -s_i, s_i, s_i, \dots],$$

i.e., let $e_{i+r} = (-1)^{\lfloor (r+1)/2 \rfloor} s_i$, for $r = 1, 2, \dots, j-1$, and keep other elements unchanged, i.e., let $e_k = s_k$. The revised sign list of p (resp. s) is denoted by $\text{rsl}(p)$ (resp. $\text{rsl}(s)$).

Yang, Hou and Zeng used the following theorem to calculate the number of distinct complex-conjugate roots and real roots.

THEOREM 1. Suppose a polynomial $p \in \mathbb{R}[x]$ has revised sign list $\text{rsl}(p)$. If the number of non-vanishing members of $\text{rsl}(p)$ is s , and the number of sign changes in $\text{rsl}(p)$ is v , then $p(x)$ has v pairs of distinct complex-conjugate roots and $s - 2v$ distinct real roots.

In order to calculate the multiplicities of roots, Yang, Hou and Zeng used the following definitions and propositions.

DEFINITION 5. Let $\Delta(p)$ denote $\gcd(p(x), p'(x))$, and let $\Delta^0(p) = p(x)$, $\Delta^j(p) = \Delta(\Delta^{j-1}(p))$, $j = 1, 2, \dots$. Then $\Delta^0(p), \Delta^1(p), \Delta^2(p), \dots$ is called the Δ -sequence of p .

PROPOSITION 1. If $\text{rsl}(p)$ contains k zeros, equivalently, $D_n = \dots = D_{n-k+1} = 0$ but $D_{n-k} \neq 0$, then $\gcd(p, p') = P_k(p, p')$, where $P_k(p, p')$ is the k th subresultant of $p(x)$ and $p'(x)$.

The relationship between the RC of $\Delta^j(p)$ and the RC of its 'repeated part' $\Delta^{j+1}(p)$ is given by the following propositions.

PROPOSITION 2. If $\Delta^j(p)$ has k distinct roots with respective multiplicities $n_1, n_2, \dots, n_k$, then $\Delta^{j+1}(p)$ has at most k distinct roots with respective multiplicities $n_1 - 1, n_2 - 1, \dots, n_k - 1$.

PROPOSITION 3. If $\Delta^j(p)$ has k distinct roots with respective multiplicities $n_1, n_2, \dots, n_k$, and $\Delta^{j-1}(p)$ has m distinct roots, then $m \geq k$, and the multiplicities of these m distinct roots are $n_1 + 1, n_2 + 1, \dots, n_k + 1, 1, \dots, 1$ respectively.

However, the old algorithms [5] and the methods above have to work with revised sign list which is a major source of inefficiency, since we have to transfer the output conditions in terms of revised sign lists to conditions in terms of sign lists. The transferring process is usually very difficult and full of opportunities for including non-realizable conditions. This consideration motivated the authors to propose a new algorithm for overcoming these disadvantages [6]. The new algorithm offers improved efficiency and a new test for non-realizable conditions. The improvement lies in the direct use of sign lists, rather than revised sign lists.

The algorithm uses the following definitions and theorems, where "PmV" means "generalized Permanences minus Variations" [3].

DEFINITION 6. Let $s = [s_n, \dots, s_0]$ be a finite list of elements in $\mathbb{R}$ such that $s_n \neq 0$. Let $m < n$ such that $s_{n-1} = \dots = s_{m+1} = 0$, and $s_m \neq 0$, and $s' = [s_m, \dots, s_0]$.

If there is no such m , then s' is the empty list. We define inductively

$$\text{PmV}(s) = \begin{cases} 0, & s' = \emptyset, \\ \text{PmV}(s') + \epsilon_{n-m} \text{sgn}(s_n s_m), & n - m \text{ odd}, \\ \text{PmV}(s'), & n - m \text{ even}, \end{cases}$$

where $\epsilon_{n-m} = (-1)^{(n-m)(n-m-1)/2}$.

The following theorem gives the number of distinct roots in terms of sign lists.

THEOREM 2. Let $D = [D_1, \dots, D_n]$ be the discriminant sequence of a real polynomial $p(x)$ of degree n , and ℓ be the maximal index such that $D_\ell \neq 0$. If $\text{PmV}(D) = r$, then $p(x)$ has $r+1$ distinct real roots and $\frac{1}{2}(\ell - r - 1)$ pairs of distinct complex conjugate roots.

The next theorem can be used to detect the non-realizable sign lists in output conditions.

THEOREM 3. Let $S = [s_1, \dots, s_n]$ and $R = [r_1, \dots, r_n]$ be the sign list and the revised sign list of $p(x)$ respectively. Then $\text{PmV}(S) = \text{PmV}(R)$.

At last, we review a result given by Yang and Xia [9][11] for computing the number of real roots on intervals, which gives us some clue for solving the first problem mentioned in Section 1.

Let $p \in \mathbb{R}[x]$ with $p(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_0$ and $a_n \neq 0$. Let $a, b \in \mathbb{R}$ such that $p(a) \neq 0$ and $p(b) \neq 0$. Let $\Psi_1(x) = (1-x)^n p\left(\frac{b-ax}{1-x}\right)$ and $\Psi_2(x) = \Psi_1(-x^2) = (1+x^2)^n p\left(\frac{b+ax^2}{1+x^2}\right)$. Then, it is easy to see that $\text{coeff}(\Psi_1, x, n) = (-1)^n p(a) \neq 0$, $\text{coeff}(\Psi_2, x, 2n) = p(a) \neq 0$ and $\Psi_1(0) = \text{coeff}(\Psi_1, x, 0) = \text{coeff}(\Psi_2, x, 0) = p(b) \neq 0$. Furthermore

PROPOSITION 4. $\#\{x \in (a, b) | p(x) = 0\} = \#\{x < 0 | \Psi_1(x) = 0\} = \frac{1}{2} \#\{x \in \mathbb{R} | \Psi_2(x) = 0\}$.

3. BASIS OF THE ALGORITHM

In this section, we establish the basis for the new algorithm. The main idea is that we transfer the computation of CRC for a parametric polynomial on an interval to the computation of CRC for a parametric polynomial on $\mathbb{R}$.

THEOREM 4. Let $p(x), \Psi_1(x), \Psi_2(x)$ be defined as in Section 2. Then, $[r_1, r_2, \dots, r_k]$ is the RC of $p(x)$ on (a, b) , if and only if $[r_1, r_1, r_2, r_2, \dots, r_k, r_k]$ is the real RC of $\Psi_2(x)$ on $\mathbb{R}$.

PROOF. Since $[r_1, r_2, \dots, r_k]$ is the RC of $p(x)$ on (a, b) , we can decompose $p(x)$ in $\mathbb{C}$ as

$$p(x) = a_n \prod_{i=1}^k (x - \alpha_i)^{r_i} \prod_{j=1}^s (x - \beta_j)^{f_j} \prod_{u=1}^t (x - \gamma_u)^{h_u} (x - \overline{\gamma_u})^{h_u},$$

where, $\alpha_i (i = 1, \dots, k), \beta_j (j = 1, \dots, s) \in \mathbb{R}$ are all distinct real roots of $p(x)$, $\gamma_u, \overline{\gamma_u} (u = 1, \dots, t) \in \mathbb{C}$ are all pairs of distinct complex-conjugate roots of $p(x)$, and $a < \alpha_i < b$, $\beta_j \notin [a, b]$. Then, since a, b are not roots of $p(x)$, we have $\Psi_1(x) =$

$$a_n \prod_{i=1}^k (\alpha_i - a)^{r_i} \left(x - \frac{\alpha_i - b}{\alpha_i - a}\right)^{r_i} \prod_{j=1}^s (\beta_j - a)^{f_j} \left(x - \frac{\beta_j - b}{\beta_j - a}\right)^{f_j}.$$

$$\prod_{u=1}^t (\gamma_u - a)^{h_u} \left(x - \frac{\gamma_u - b}{\gamma_u - a}\right)^{h_u} (\overline{\gamma_u} - a)^{h_u} \left(x - \frac{\overline{\gamma_u} - b}{\overline{\gamma_u} - a}\right)^{h_u}.$$

Notice that $\frac{\alpha_i - b}{\alpha_i - a} \neq \frac{\alpha_j - b}{\alpha_j - a} (i \neq j)$, $\frac{\alpha_i - b}{\alpha_i - a} < 0$, $\frac{\beta_j - b}{\beta_j - a} > 0$ and $\frac{\gamma_u - b}{\gamma_u - a}, \frac{\overline{\gamma_u} - b}{\overline{\gamma_u} - a} \notin \mathbb{R}$ because of $\alpha_i \neq \alpha_j$, $a < \alpha_i < b$, $\beta_j \notin [a, b]$ and $a \neq b$ respectively.

Based on the facts above, we can conclude that $\Psi_1(x)$ in $(-\infty, 0)$ has exactly k distinct real roots of respective multiplicities $r_1, \dots, r_k$. Therefore, the RC of $\Psi_1(x)$ on $(-\infty, 0)$ is $[r_1, r_2, \dots, r_k]$, if and only if the RC of $p(x)$ on (a, b) is $[r_1, r_2, \dots, r_k]$. Noticing $\Psi_1(0) = p(b) \neq 0$, by similar discussion, we can conclude that the real RC of $\Psi_2(x)$ on $\mathbb{R}$ is $[r_1, r_1, r_2, r_2, \dots, r_k, r_k]$, if and only if the RC of $\Psi_1(x)$ on $(-\infty, 0)$ is $[r_1, r_2, \dots, r_k]$. Finally, the desired result follows. $\square$

From Theorem 4, for intervals $(-\infty, a)$ and $(a, +\infty)$, we have the following corollary.

COROLLARY 1. Let notations be as in Theorem 4. Then, $[r_1, r_2, \dots, r_k]$ is the RC of $p(x)$ on $(-\infty, a)$ iff $[r_1, r_1, r_2, r_2, \dots, r_k, r_k]$ is the real RC of $\Psi_2(x) := p(-x^2 + a)$ on $\mathbb{R}$; $[r_1, r_2, \dots, r_k]$ is the RC of $p(x)$ on $(a, +\infty)$ iff $[r_1, r_1, r_2, r_2, \dots, r_k, r_k]$ is the real RC of $\Psi_2(x) := p(x^2 + a)$ on $\mathbb{R}$.

Based on Theorem 4 and Corollary 1, the conditions for $p(x)$ having $[r_1, r_2, \dots, r_k]$ as its RC on an interval can be obtained by computing the conditions for $\Psi_2(x)$ having $[r_1, r_1, r_2, r_2, \dots, r_k, r_k]$ as its real RC on $\mathbb{R}$.

4. THE ALGORITHM

Let $p(x), \Psi_2(x)$ be defined as in Section 2. In this section, we propose an algorithm for computing the CRC of $p(x)$ on (a, b) , where $a, b \in \mathbb{R} \cup \{-\infty, +\infty\}$. We need some functions to present the algorithm.

- **ClassifySL:** Input a set of sign lists and a positive integer i ; output the subset of sign lists of which the maximal index of non-vanishing members is i . For example, if the input is $S = \{[1, 1, 1, -1, 0, 0], [1, 0, 0, -1, 0, 0], [1, 1, 1, -1, 1, 1], [1, -1, 0, 0, 1, 1]\}$, then $\text{ClassifySL}(S, 4) = \{[1, 1, 1, -1, 0, 0], [1, 0, 0, -1, 0, 0]\}$, $\text{ClassifySL}(S, 6) = \{[1, 1, 1, -1, 1, 1], [1, -1, 0, 0, 1, 1]\}$, but $\text{ClassifySL}(S, 1) = \text{ClassifySL}(S, 2) = \text{ClassifySL}(S, 3) = \text{ClassifySL}(S, 5) = \emptyset$.
- **MinusOne:** Input an RC $[r_1, \dots, r_k]$; output $[r_1 - 1, \dots, r_k - 1]$ (all elements with value 0 are removed).
- **Op:** Input a set $\{a_1, \dots, a_n\}$ or a list $[a_1, \dots, a_n]$; output the sequence $a_1, \dots, a_n$.
- **DRC:** input an RC $[r_1, \dots, r_k]$; output $[r_1, r_1, \dots, r_k, r_k]$.
- **AllRC:** Input $n \in \mathbb{N}$; output the union of the sets of partitions of $0, 1, \dots, n$.

By Theorem 4 and Corollary 1, the conditions for p having $L = [r_1, r_2, \dots, r_k]$ as its RC on (a, b) can be obtained by computing the conditions for Ψ_2 having $L_2 = [r_1, r_1, r_2, r_2, \dots, r_k, r_k]$ as its real RC on $\mathbb{R}$. So, what we have to do is to find the conditions for Ψ_2 having L_2 as its real RC on $\mathbb{R}$.

We first compute all possible sign lists of Ψ_2 for Ψ_2 having L_2 as its real RC on $\mathbb{R}$.

PolySL.

Input: Ψ_2 and L_2 .

Output: The set of all possible sign lists of Ψ_2 .

Procedure:

- Compute the discriminant sequence $D = [D_1, \dots, D_{2n}]$ of Ψ_2 .
- Compute the set S_0 of all possible sign lists from D : for $1 \leq k \leq 2n$, if $D_k \in \mathbb{R}$, then $D_k \rightarrow \text{sgn}(D_k)$; otherwise, $D_k \rightarrow \{-1, 0, 1\}$. For example, if $D = [1, -2, a]$, then $S_0 = \{[1, -1, -1], [1, -1, 0], [1, -1, 1]\}$.
- Compute $S = \{s \in S_0 \mid \text{PmV}(s) = \text{PmV}(\text{rsl}(s)) = 2k - 1\}$, where $\#L_2 = 2k$.
- Return S .

Then $S = \text{PolySL}(\Psi_2, L_2)$ is the set of all possible sign lists of Ψ_2 for Ψ_2 having L_2 as its real RC on $\mathbb{R}$. In order to make the multiplicities of the $2k$ distinct real roots of Ψ_2 be $r_1, r_1, r_2, r_2, \dots, r_k, r_k$ respectively, we also have to determine the possible sign lists of the polynomials in the Δ -sequence of Ψ_2 . According to Proposition 1, $\Delta^1(\Psi_2)$ can be determined by the maximal index ℓ of non-vanishing members in the sign list of Ψ_2 , which actually is the total number of distinct (real and complex) roots of Ψ_2 . Since L_2 does not contain information about the number of distinct complex roots of Ψ_2 , the maximal index ℓ is not uniquely determined. Therefore, unlike the case of RC on $\mathbb{R}$ [6], there may be more than one $\Delta^1(\Psi_2)$ for the real RC L_2 , and consequently the conditions for $\Psi_2(x)$ having L_2 as its real RC on $\mathbb{R}$ would be more complicated. The following observations can be used to solve the problem.

Let $G_i = \text{ClassifySL}(S, i)$. Then $S = \bigcup_{i=k}^n G_{2i}$ because Ψ_2 has at least $2k$ distinct roots and, by Theorem 4 and its proof, the number of distinct roots of Ψ_2 is always a multiple of 2. For $k \leq i \leq n$, if the sign list of Ψ_2 belongs to G_{2i} , then the number of distinct roots of Ψ_2 is $2i$. So by Proposition 1, $\Delta^1(\Psi_2) = P_{2n-2i}(\Psi_2, \Psi'_2)$. It is a polynomial of degree $2n - 2i$. By Proposition 2, the real RC of $\Delta^1(\Psi_2)$ on $\mathbb{R}$ is $\text{MinusOne}(L_2)$. Then the computation above can be repeated for $\Delta^1(\Psi_2)$ and $\text{MinusOne}(L_2)$ until the termination conditions (see below) are reached. Therefore, unlike the case of RC on $\mathbb{R}$, we have to divide all possible sign lists of Ψ_2 into different groups. For each group, there is a distinct $\Delta^1(\Psi_2)$.

For the following three *termination conditions*, we can determine just by S that the real RC of Ψ_2 on $\mathbb{R}$ is L_2 without further computation for the sign lists of other polynomials in the Δ -sequence of Ψ_2 .

1. $k = n$,
2. $k = 0$,
3. the last entry of the sign list of Ψ_2 is nonzero.

Therefore, the algorithm for generating the necessary and sufficient conditions for $p(x)$ having $L = [r_1, r_2, \dots, r_k]$ as its RC on (a, b) is as follows. Let Ψ_2 and L_2 be defined as above. The output conditions are a sequence of *mixed lists*. Each mixed list consists of a polynomial in the Δ -sequence of Ψ_2 , followed by all of its possible sign lists. We denote the empty sequence by NULL. Notice that if the empty sequence is returned, then L_2 , and consequently L are not realizable.

IntCond

Input: Ψ_2 and a real RC L_2 .

Output: A sequence of mixed lists (the conditions for Ψ_2 having L_2 as its real RC on $\mathbb{R}$).

Procedure:

```

 $t \leftarrow \#L_2$ 
 $m \leftarrow \deg(\Psi_2)$ 
 $S \leftarrow \text{PolySL}(\Psi_2, L_2)$ 
if  $S = \emptyset$  then
    return NULL
elif  $t = 0$  or  $t = m$  or  $\text{ClassifySL}(S, m) = S$  then
    return  $[\Psi_2, \text{Op}(S)]$  (*)
else
    for  $i$  from  $t$  to  $m$  by 2 do
         $G_i \leftarrow \text{ClassifySL}(S, i)$ 
        if  $G_i \neq \emptyset$  then
            if  $i = m$  then
                return  $[\Psi_2, \text{Op}(G_i)]$  (**)
            else
                 $C_i \leftarrow \text{IntCond}(\Delta^1(\Psi_2), \text{MinusOne}(L_2))$ 
                if  $C_i \neq \text{NULL}$  then
                    return  $[\Psi_2, \text{Op}(G_i)], C_i$ 
                else
                    return NULL

```

Remark 1. Notice that $\Delta^1(\Psi_2)$ depends on i . By Proposition 1, it is a polynomial of degree $2n - i$. In the algorithm, t is a nonnegative integer representing the number of distinct real roots of Ψ_2 . m is a positive integer representing the degree of Ψ_2 . S is a set of sign lists representing all possible sign lists of Ψ_2 for Ψ_2 having L_2 as its real RC on $\mathbb{R}$. G_i is the subset of S such that the maximal index of non-vanishing members in the sign lists of G_i is i . Finally, C_i is a sequence of mixed lists representing the conditions for $\Delta^1(\Psi_2)$ having $\text{MinusOne}(L_2)$ as its real RC on $\mathbb{R}$.

PROOF. First, observe that the number of recursions in the algorithm is finite. So the algorithm will terminate in finite steps.

Second, the output conditions can be viewed as a couple of trees whose nodes consist of mixed lists. Their roots are those mixed lists of which the first member is Ψ_2 , and their leaves are the mixed lists at points (*) or (**) in the algorithm.

For the case of RC on $\mathbb{R}$ [6], the necessary and sufficient conditions for a polynomial having a given RC is a single chain of mixed lists connected using conjunction ($\wedge$). Now suppose the leaves are labeled by i ($i = 1, \dots, N$). Let Θ_i be the chain of mixed lists obtained by connecting the nodes from the i th leaf to its root using conjunction. Then, the output conditions can be expressed as the disjunction of these chains, $\bigvee_{i=1}^N \Theta_i$.

Using Theorem 2 and Proposition 2 and 3, as we did in [6], we can prove that $\bigvee_{i=1}^N \Theta_i$ is the necessary and sufficient conditions for Ψ_2 having L_2 as its real RC on $\mathbb{R}$.

Finally, by Theorem 4, the correctness of IntCond follows. $\square$

Based on the algorithm IntCond above, we can now propose an algorithm for generating the CRC of a parametric polynomial $p(x)$ on an interval (a, b) .

IntCRC

Input: A real parametric polynomial $p(x)$, and $a, b \in \mathbb{R} \cup \{-\infty, +\infty\}$.

Output: The CRC of $p(x)$ on (a, b) .

Procedure:

```

 $\mathcal{L} \leftarrow \text{AllRC}(\deg(p))$ 
compute  $\Psi_2$ 
for  $L$  in  $\mathcal{L}$  do
   $C \leftarrow \text{IntCond}(\Psi_2, \text{DRC}(L))$ 
  if  $C \neq \text{NULL}$  then
    return  $L$  and  $C$ 

```

Optimization of algorithm. Finally we discuss the optimization of the algorithm. In comparison with the case on $\mathbb{R}$, the output conditions of the CRC of a parametric polynomial on an interval is usually large, especially when the parametric polynomial has a general form. So there remains the work of condensing the output conditions. Suppose $[D_1, \dots, D_{2n}]$ is the discriminant sequence of Ψ_2 , and S is the set of all possible sign lists of Ψ_2 for Ψ_2 having $L_2 = [r_1, r_1, \dots, r_k, r_k]$ as its real RC on $\mathbb{R}$. Notice that the length of any sign list in S is $2n$ and the first element of it can always be assumed to be 1.

For $S_1, S_2, S_3 \in S$, if there is an index i such that for any index $j \neq i$, we have $S_1[j] = S_2[j] = S_3[j]$ and $\{S_1[i], S_2[i], S_3[i]\} = \{-1, 0, 1\}$, then S_1, S_2, S_3 can be represented by a single list $[S_1[1], \dots, S_1[i-1], *, S_1[i+1], \dots, S_1[2n]]$, where ‘*’ means that D_i is unconstrained.

For $S_1, S_2 \in S$, if there is an index i such that for any index $j \neq i$, we have $S_1[j] = S_2[j]$ and $\{S_1[i], S_2[i]\} = \{0, 1\}$, then S_1, S_2 can be represented by a single list $[S_1[1], \dots, S_1[i-1], 0+, S_1[i+1], \dots, S_1[2n]]$, where ‘0+’ means that $D_i \geq 0$.

Similarly, for $S_1, S_2 \in S$, if there is an index i such that for any index $j \neq i$, we have $S_1[j] = S_2[j]$ and $\{S_1[i], S_2[i]\} = \{0, -1\}$, then S_1, S_2 can be represented by a single list $[S_1[1], \dots, S_1[i-1], 0-, S_1[i+1], \dots, S_1[2n]]$, where ‘0-’ means that $D_i \leq 0$.

At last, for $S_1, S_2 \in S$, if there is an index i such that for any index $j \neq i$, we have $S_1[j] = S_2[j]$ and $\{S_1[i], S_2[i]\} = \{1, -1\}$, then S_1, S_2 can be represented by a single list $[S_1[1], \dots, S_1[i-1], <>, S_1[i+1], \dots, S_1[2n]]$, where ‘<>’ means that $D_i \neq 0$.

In this way, S can be condensed into a smaller set S' . For the sake of simplicity, we still call the elements in S' sign lists. For example, if $S = \{[1, 0, 1, 1], [1, 0, 0, 1], [1, 0, -1, 1]\}$, then $S' = \{[1, 0, *, 1]\}$.

Complexity analysis. We give an upper bound for the running time of the algorithm **IntCond**. We count the number of operations $T(n)$ on the coefficients, regarding all polynomials as univariate in x . We recall that the input polynomial Ψ_2 has degree $2n$. We first estimate the call $S = \text{PolySL}(\Psi_2, L_2)$ and make three observations: (1) computing the discriminant sequence D of Ψ_2 can be done in $O(n^2)$ coefficient operations, (2) computing the set S_0 of all possible sign lists from D requires $O(3^{2n})$ operations on coefficients (and integers), (3) for each $s \in S_0$, checking whether s belongs to the output set S can be done in linear time (w.r.t. n), and there are 3^{2n} sign lists in S_0 . So computing S requires $O(n3^{2n})$ operations.

Consequently the call $S = \text{PolySL}(\Psi_2, L_2)$ runs in $O(n3^{2n})$ operations on coefficients. Returning to the body of the al-

gorithm **IntCond**, we observe that all the calls $\text{ClassifySL}(S, i)$ together have a linear cost w.r.t. the size of S , hence these costs can be absorbed in the cost for creating S . The remaining costs in the body of **IntCond** are (at most) $n-1$ recursive calls with respective costs $T(n-1), T(n-2), \dots, T(1)$. Therefore, we have

$$T(n) \leq O(n3^{2n}) + T(n-1) + T(n-2) + \dots + T(2) + T(1).$$

Unrolling the recurrence, performing elementary computations we obtain: $T(n) \in O(n9^n)$.

5. EXAMPLES

In this section, we present some examples of the CRCs for some parametric polynomials on intervals. As a direct application, we show how to apply the CRC to questions concerning real quantifier elimination. As with many symbolic computations, solving general cases of high degree polynomials usually would result in very large solutions. Therefore, we only focus on sparse parametric polynomials. All computations were performed with Maple 10 running on a 1.6 GHz Pentium CPU. Notice that, in the output conditions below, different chains of mixed lists (see the proof of **IntCond**) are separated by semicolons (;).

Example 1. The following is the CRC of $p_3 = x^3 + ax + b$ on $(0, 2)$, where the initial conditions correspond to $p_3(0) \neq 0$ and $p_3(2) \neq 0$. It takes 1.18 seconds to generate the CRC. After the output, there is an explanation of its interpretation.

(*) $p_3 = x^3 + ax + b$

The CRC of p_3 on $(0, 2)$ is:

([polynomial, its all possible sign lists])

(1) [], if and only if $[P_6, [1, -1, 0, 0, <>, -1], [1, -1, -1, 1, 0, 0], [1, -1, -1, 0, +, -1], [1, -1, -1, 1, -1, -1], [1, -1, -1, *, 1, -1], [1, 0, 0, -1, 1, -1], [1, 1, 1, -1, 1, -1]]$

(2) [1], if and only if $[P_6, [1, 1, 1, -1, 0, 0], [1, 0, 0, -1, 0, 0], [1, -1, -1, -1, 0, 0], [P_6, [1, 1, 1, -1, 1, 1], [1, -1, 0, 0, 1, 1], [1, 0, -1, -1, 1, 1], [1, 1, 1, <>, -1, 1], [1, 1, 1, 0, 0, 1], [1, -1, -1, 1, 1, 1], [1, -1, -1, 0, 0, 1], [1, -1, -1, -1, -1, 1], [1, 0, 0, -1, <>, 1]]$

(3) [2], if and only if $[P_6, [1, 1, 1, -1, 0, 0], [1, 0, 0, -1, 0, 0], [1, -1, -1, -1, 0, 0], [P_6, [1, 1]]$

(4) [1, 1], if and only if $[P_6, [1, -1, -1, -1, -1, -1], [1, 1, 1, -1, -1, -1], [1, 0, 0, -1, -1, -1]]$, where,

$P_6 = 8+2*a+4*a*x^2+2*a*x^4+b+3*b*x^2+3*b*x^4+b*x^6$,
 $P_62 = b*a^2*x^2+9*b^2*x^2+2*a^3*x^2+6*b*a*x^2$
 $+b*a^2+8*a^2+2*a^3$.

And the initial conditions are

$b <> 0, 8+2*a+b <> 0$

Let us explain the CRC of p_3 on $(0, 2)$. We assume that the initial conditions $b \neq 0$ and $2a+b+8 \neq 0$ hold. All possible cases of RC for p_3 on $(0, 2)$ are [], [1], [2], [1, 1], [3], [1, 2], [1, 1, 1].

The conditions for p_3 having [1] (one single root) as its RC in $(0, 2)$ can be obtained by computing the conditions for P_6 (which is Ψ_2 in the algorithm) having [1, 1] as its real RC on $\mathbb{R}$.

All possible sign lists of P_6 would be $[1, 1, 1, -1, 0, 0]$, $[1, 0, 0, -1, 0, 0]$, $[1, -1, -1, -1, 0, 0]$, $[1, 1, 1, -1, 1, 1]$, $[1, -1, 0, 0, 1, 1]$, $[1, 0, -1, -1, 1, 1]$, $[1, 1, 1, <>, -1, 1]$, $[1, 1, 1, 0, 0, 1]$, $[1, -1, -1, 1, 1, 1]$, $[1, -1, -1, 0, 0, 1]$, $[1, -1, -1, -1, -1, 1]$, $[1, 0, 0, -1, <>, 1]$.

Now these sign lists of P_6 can be divided into two groups: $G_4 = \{[1, 1, 1, -1, 0, 0], [1, 0, 0, -1, 0, 0], [1, -1, -1, -1, 0, 0]\}$ and $G_6 = \{[1, 1, 1, -1, 1, 1], [1, -1, 0, 0, 1, 1], [1, 0, -1, -1, 1, 1], [1, 1, 1, <>, -1, 1], [1, 1, 1, 0, 0, 1], [1, -1, -1, 1, 1, 1], [1, -1, -1, 0, 0, 1], [1, -1, -1, -1, -1, 1], [1, 0, 0, -1, <>, 1]\}$.

If the sign list of P_6 belongs to G_4 , then the number of distinct roots of P_6 is 4. So the 'repeated part' $\Delta^1(P_6) = P_{62}$ and the RC of P_{62} is MinusOne($[1, 1]$) = $[-1]$. For P_{62} and $[-1]$, IntCond is called again, obtaining that the condition for P_{62} having $[-1]$ as its real RC on $\mathbb{R}$ is its sign list being $[1, -1]$.

At this point, the termination condition 2 is satisfied, so IntCond terminates. If the sign list of P_6 belongs to G_6 , then the termination condition 3 is satisfied, and IntCond terminates.

In summary, p_3 has $[1]$ as its RC on $(0, 2)$, if and only if the sign list of P_6 belongs to G_4 and the sign list of P_{62} is $[1, -1]$, or the sign list of P_6 belongs to G_6 . The cases $[-1]$, $[2]$ and $[1, 1]$ can be explained similarly.

For the cases $[3]$, $[1, 2]$, $[1, 1, 1]$, since the output of IntCond is the empty sequence NULL, they are not realizable. Based on the CRC of p_3 , we can answer some questions concerning real quantifier elimination. The discriminant sequence of P_6 is $[1, D_2, D_3, D_4, D_5, D_6]$, where

$$\begin{aligned} D_2 &= -3b^2 - 2ab, \\ D_3 &= -a^2b(2a + 3b), \\ D_4 &= a^2b(a^2b + 9b^2 + 2a^3 + 6ab), \\ D_5 &= -b(a^2b + 2a^3 + 6ab + 9b^2)(4a^3 + 27b^2), \\ D_6 &= -(8 + 2a + b)b(4a^3 + 27b^2)^2. \end{aligned}$$

The necessary and sufficient condition for $\forall x \in (0, 2)[p_3 \neq 0]$ is that case (1) holds, and case (1) holds iff the sign list of P_6 be one of the following: $[1, -1, 0, 0, <>, -1]$, $[1, -1, -1, 1, 0, 0]$, $[1, -1, -1, 0, 0, -1]$, $[1, -1, -1, 1, -1, -1]$, $[1, -1, -1, *, 1, -1]$, $[1, 0, 0, -1, 1, -1]$, $[1, 1, 1, -1, 1, -1]$. Therefore, the necessary and sufficient condition for $\forall x \in (0, 2)[p_3 \neq 0]$ is

$[D_2 < 0 \wedge D_3 = 0 \wedge D_4 = 0 \wedge D_5 \neq 0 \wedge D_6 < 0] \vee [D_2 < 0 \wedge D_3 < 0 \wedge D_4 > 0 \wedge D_5 = 0 \wedge D_6 = 0] \vee [D_2 < 0 \wedge D_3 < 0 \wedge D_4 \geq 0 \wedge D_5 = 0 \wedge D_6 < 0] \vee [D_2 < 0 \wedge D_3 < 0 \wedge D_4 > 0 \wedge D_5 < 0 \wedge D_6 < 0] \vee [D_2 < 0 \wedge D_3 < 0 \wedge D_5 > 0 \wedge D_6 < 0] \vee [D_2 = 0 \wedge D_3 = 0 \wedge D_4 < 0 \wedge D_5 > 0 \wedge D_6 < 0] \vee [D_2 > 0 \wedge D_3 > 0 \wedge D_4 < 0 \wedge D_5 > 0 \wedge D_6 < 0]$.

Similarly, the necessary and sufficient condition for p_3 having two distinct real roots of multiplicities 1 on $(0, 2)$ is that case (4) holds. That is

$[D_2 < 0 \wedge D_3 < 0 \wedge D_4 < 0 \wedge D_5 < 0 \wedge D_6 < 0] \vee [D_2 > 0 \wedge D_3 > 0 \wedge D_4 < 0 \wedge D_5 < 0 \wedge D_6 < 0] \vee [D_2 = 0 \wedge D_3 = 0 \wedge D_4 < 0 \wedge D_5 < 0 \wedge D_6 < 0]$.

Example 2. The following is the CRC of $p_4 = x^4 + ax + b$ on $(-\infty, -1)$, where the initial condition corresponds to $p_4(-1) \neq 0$. It takes 0.75 second to generate the CRC.

(*) $p_4 := x^4 + a*x + b$
The CRC of p_4 on $(-\infty, -1)$ is:
([polynomial, its all possible sign lists])

(1) $[-1]$, if and only if

$[P8, [1, -1, 0, 0, 0, 0, 0, 0], [1, -1, 0, 0, -1, -1, -1, 1], [1, -1, 0, 0, 0, 0, <>, 1], [1, -1, 0, 0, <>, -1, 0, 0], [1, -1, 0, 0, 1, *, -1, 1], [1, -1, 0, 0, <>, -1, 1, 1]]$

(2) $[1]$, if and only if

$[P8, [1, -1, 0, 0, 0, 0, -1, -1], [1, -1, 0, 0, 1, -1, -1, -1], [1, -1, 0, 0, 1, 0, 0, -1], [1, -1, 0, 0, 1, 1, 1, -1], [1, -1, 0, 0, -1, -1, -1, -1]]$

(3) $[1, 1]$, if and only if

$[P8, [1, -1, 0, 0, 1, 1, 1, 1]]$

Where,

(#1) $P8 := 1 + 4*x^2 + 6*x^4 + 4*x^6 + x^8 - a*a*x^2 + b$,
and the initial condition is
 $1 - a + b < 0$

Example 3. Find the conditions on a, b, c such that $\forall x > 0, p_5 = x^5 + ax^2 + bx + c > 0$.

This problem was studied by Yang in [9]. Here we try to use the CRC method to solve this problem. First, we compute the CRC of p_5 on $(0, +\infty)$. It takes 6.33 seconds.

(*) $p_5 := x^5 + a*x^2 + b*x + c$
The CRC of p_5 on $(0, +\infty)$ is:
([polynomial, its all possible sign lists])

(1) $[-1]$, if and only if

$[P10, [1, 0, 0, 0, 0, -1, 1, *, -1, -1], [1, 0, 0, 0, 0, *, -1, 1, 0, 0], [1, 0, 0, 0, 0, -1, 1, 1, 1, -1], [1, 0, 0, 0, 0, -1, 1, 1, 0, 0], [1, 0, 0, 0, 0, -1, 0, 0, -1, -1], [1, 0, 0, 0, 0, *, -1, 0, +, 0, -1], [1, 0, 0, 0, 0, 0, +, 0, -1, 1, -1], [1, 0, 0, 0, 0, 0, -, 0, 1, *, -1], [1, 0, 0, 0, 0, *, -1, *, 1, -1], [1, 0, 0, 0, 0, -1, 0, 0, 0, 0], [1, 0, 0, 0, 0, 1, 1, -1, 1, -1], [1, 0, 0, 0, 0, *, -1, 1, -1, -1], [1, 0, 0, 0, 0, *, 0, 0, 1, -1], [1, 0, 0, 0, 0, -, 0, 0, 0, -1], [1, 0, 0, 0, 0, -1, 1, 0, +, 0, -1], [1, 0, 0, 0, 0, 0, -, 0, 1, 0, 0]]$

(2) $[1, 2]$, if and only if

$[P10, [1, 0, 0, 0, 0, 1, 1, 1, 0, 0]], [P102, [1, 1]]$

(3) $[1]$, if and only if

$[P10, [1, 0, 0, 0, 0, 1, 0, 0, 0, 0]], [P104, [1, -1, 0, 0], [1, -1, 1, 1], [1, 0, 0, 1], [1, -1, 0, 1], [1, 1, -1, 1], [1, 0, -1, 1], [1, -1, -1, 1]]$; $[P10, [1, 0, 0, 0, 0, 1, 1, -1, 0, 0], [1, 0, 0, 0, 0, *, -1, -1, 0, 0], [1, 0, 0, 0, 0, 0, +, 0, -1, 0, 0]], [P102, [1, -1]]$; $[P10, [1, 0, 0, 0, 0, -1, 1, 1, 1, 1], [1, 0, 0, 0, 0, 1, 1, 0, -, 0, 1], [1, 0, 0, 0, 0, 1, 1, -1, 1, 1], [1, 0, 0, 0, 0, *, -1, 0, -, 0, 1], [1, 0, 0, 0, 0, 1, 0, 0, -1, 1], [1, 0, 0, 0, 0, 1, 1, *, -1, 1], [1, 0, 0, 0, 0, *, 0, 0, 1, 1], [1, 0, 0, 0, 0, -, 0, 1, 1, 1], [1, 0, 0, 0, 0, *, -1, *, 1, 1], [1, 0, 0, 0, 0, *, -1, -1, -1, 1], [1, 0, 0, 0, 0, 0, +, 0, 0, 0, 1], [1, 0, 0, 0, 0, 0, +, 0, -1, *, 1]]$

(4) $[2]$, if and only if

$[P10, [1, 0, 0, 0, 0, 1, 0, 0, 0, 0]], [P104, [1, 0, -1, -1], [1, -1, -1, -1], [1, 1, -1, -1], [1, 1, 1, -1], [1, 1, 0, -1], [1, 0, 0, -1]]$; $[P10, [1, 0, 0, 0, 0, 0, -1, 0, 0], [1, 0, 0, 0, 0, 1, -1, -1, 0, 0], [1, 0, 0, 0, 0, -, 1, -1, 0, 0], [1, 0, 0, 0, 0, -1, -1, -1, 0, 0], [1, 0, 0, 0, 0, 1, 1, -1, 0, 0], [1, 0, 0, 0, 0, 1, 0, -1, 0, 0]], [P102, [1, -1]]$

(5) $[3]$, if and only if $[P10, [1, 0, 0, 0, 0, 1, 0, 0, 0, 0]], [P104, [1, 1, 0, 0]], [P1042, [1, 1]]$

(6) $[1, 1]$, if and only if

[P10, [1,0,0,0,0,1,1,1,0,0]], [P102, [1,-1]]; [P10,
[1,0,0,0,0,1,1,1,-1], [1,0,0,0,0,1,0,0,-1,-1],
[1,0,0,0,0,1,1,*, -1,-1], [1,0,0,0,0,1,1,0+,0,-1],
[1,0,0,0,0,0+,0,-1,-1,-1], [1,0,0,0,0,*, -1,-1,-1,-1]]

(7) [1,1,1], if and only if
[P10, [1,0,0,0,0,1,1,1,1,1]]

Where,

(#1) P1042: $= -2*b*x^2 - 5*c$
(#2) P102: $= 54*a^4*c + 27*b*a^4*x^2 - 225*x^2*c^2*a^2$
 $+ 600*a*c^2*b + 720*a*x^2*c*b^2 - 320*c*b^3 - 256*x^2*b^4$,
(#3) P10: $= x^{10} + a*x^4 + b*x^2 + c$,
(#4) P104: $= -3*a*x^4 - 4*b*x^2 - 5*c$,
and the initial condition is
 $c < 0$

The discriminant sequence of P_{10} is $[1, 0, 0, 0, 0, D_6, D_7, D_8, D_9, D_{10}]$, where

$$\begin{aligned} D_6 &= -a^5, & D_7 &= -a^3(27a^4 + 300abc - 160b^3), \\ D_8 &= (300bac - 160b^3 + 27a^4)(720acb^2 - 256b^4 + 27a^4b - 225a^2c^2), \\ D_9 &= -(720acb^2 - 256b^4 + 27a^4b - 225a^2c^2)(-1600b^3ca + 256b^5 - 27a^4b^2 + 2250ba^2c^2 + 3125c^4 + 108a^5c), \\ D_{10} &= -c(-1600b^3ca + 256b^5 - 27a^4b^2 + 2250ba^2c^2 + 3125c^4 + 108a^5c)^2 \end{aligned}$$

Again, we assume that the initial condition $c \neq 0$ holds. Then $(\forall x > 0)[p_5 = x^5 + ax^2 + bx + c > 0]$ iff case (1) holds. That is $[D_6 < 0 \wedge D_7 > 0 \wedge D_9 < 0 \wedge D_{10} < 0] \vee [D_7 < 0 \wedge D_8 > 0 \wedge D_9 = 0 \wedge D_{10} = 0] \vee [D_6 < 0 \wedge D_7 > 0 \wedge D_8 > 0 \wedge D_9 > 0 \wedge D_{10} < 0] \vee [D_6 < 0 \wedge D_7 > 0 \wedge D_8 > 0 \wedge D_9 = 0 \wedge D_{10} = 0] \vee [D_6 < 0 \wedge D_7 = 0 \wedge D_8 < 0 \wedge D_9 < 0 \wedge D_{10} < 0] \vee [D_7 < 0 \wedge D_8 \geq 0 \wedge D_9 = 0 \wedge D_{10} < 0] \vee [D_6 \geq 0 \wedge D_7 = 0 \wedge D_8 < 0 \wedge D_9 > 0 \wedge D_{10} < 0] \vee [D_6 \leq 0 \wedge D_7 = 0 \wedge D_8 > 0 \wedge D_{10} < 0] \vee [D_7 < 0 \wedge D_9 > 0 \wedge D_{10} < 0] \vee [D_6 < 0 \wedge D_7 = 0 \wedge D_8 = 0 \wedge D_9 = 0 \wedge D_{10} = 0] \vee [D_6 > 0 \wedge D_7 > 0 \wedge D_8 < 0 \wedge D_9 > 0 \wedge D_{10} < 0] \vee [D_7 < 0 \wedge D_8 > 0 \wedge D_9 < 0 \wedge D_{10} < 0] \vee [D_7 = 0 \wedge D_8 = 0 \wedge D_9 > 0 \wedge D_{10} < 0] \vee [D_6 \leq 0 \wedge D_7 > 0 \wedge D_8 = 0 \wedge D_9 = 0 \wedge D_{10} < 0] \vee [D_6 < 0 \wedge D_7 > 0 \wedge D_8 \geq 0 \wedge D_9 = 0 \wedge D_{10} < 0] \vee [D_6 \leq 0 \wedge D_7 = 0 \wedge D_8 > 0 \wedge D_9 = 0 \wedge D_{10} = 0]$.

This solution appears to be different from that given by Yang [9]. To obtain the same form as he gave, one might apply additional transformations to the result. For example, one writes $D_7 = -a^3(27a^4 + 300abc - 160b^3) = a^3d_8$ with d_8 taking the obvious value. Then the condition $D_7 > 0$ can be written as $(a^3 > 0 \wedge d_8 > 0) \vee (a^3 < 0 \wedge d_8 < 0)$. It is now possible to use Boolean algebra to transform the above result into Yang's form. Furthermore, the CRC of p_5 on $(0, +\infty)$ can be used to solve other problems easily. For example, the necessary and sufficient condition for p_5 having two distinct positive roots of multiplicities 1 is that case (6) holds. Thus the condition is

$$\begin{aligned} &[D_6 > 0 \wedge D_7 > 0 \wedge D_8 > 0 \wedge D_9 = 0 \wedge D_{10} = 0 \wedge E_2 < 0] \vee [D_6 > 0 \wedge D_7 > 0 \wedge D_8 > 0 \wedge D_9 > 0 \wedge D_{10} < 0] \vee [D_6 > 0 \wedge D_7 = 0 \wedge D_8 = 0 \wedge D_9 < 0 \wedge D_{10} < 0] \vee [D_6 > 0 \wedge D_7 > 0 \wedge D_9 < 0 \wedge D_{10} < 0] \vee [D_6 > 0 \wedge D_7 > 0 \wedge D_8 \geq 0 \wedge D_9 = 0 \wedge D_{10} < 0] \vee [D_6 \geq 0 \wedge D_7 = 0 \wedge D_8 < 0 \wedge D_9 < 0 \wedge D_{10} < 0] \vee [D_7 < 0 \wedge D_8 < 0 \wedge D_9 < 0 \wedge D_{10} < 0], \end{aligned}$$

where $[1, E_2]$ is the discriminant sequence of P_{102} , and $E_2 = -c(300bac - 160b^3 + 27a^4)(720acb^2 - 256b^4 + 27a^4b - 225a^2c^2)$.

Also, we can conclude that it is impossible for p_5 to have four distinct positive roots.

6. CONCLUSION

In this paper, we have proposed a new algorithm for the automatic computation of the complete root classification of a parametric polynomial on an interval. However, some issues deserve further consideration. Although Theorem 3 is used in the algorithm to filter non-realizable sign lists, it is not guaranteed that all non-realizable sign lists are detected and deleted. Furthermore, the output conditions are basically equalities and inequalities in terms of the parametric coefficients. A further step would be to determine what are the possible values of the parametric coefficients such that the conditions described are satisfied. This is essentially the problem of solving semi-algebraic systems, a problem well-known to be difficult. This problem may be addressed using interval analysis [4] or method based on Gröbner basis [8]. We will leave these issues in further work.

7. REFERENCES

- [1] H. Anai and H. Yanami. SyNRAC: A Maple-package for solving real algebraic constraints. *Lecture Notes in Computer Science*, 2657:828-837, 2003.
- [2] D. S. Arnon. Geometric reasoning with logic and algebra. *Artificial Intelligence*, 37:37-60, 1988.
- [3] S. Basu, R. Pollack, M.-F. Roy. *Algorithms in Real Algebraic Geometry*, 2nd edition. Springer-Verlag, 2003.
- [4] A. Colagrossi and A. M. Miola. Computing real zeros of polynomials with parametric coefficients. *ACM SIGSAM Bulletin*, 17(1):12-15, 1983.
- [5] S. Liang and D. J. Jeffrey. An Algorithm for computing the complete root classification of a parametric polynomial. *Lecture Notes in Computer Science*, 4120:116-130, 2006.
- [6] S. Liang and D. J. Jeffrey. The automatic computation of the complete root classification for a parametric polynomial. Electronic proceedings of MEGA 2007, www.ricam.oeaw.ac.at/mega2007/electronic/30.pdf.
- [7] S. Liang and J. Zhang. A complete discrimination system for polynomials with complex coefficients and its automatic generation. *Science in China (Series E)*, 42(2):113-128, 1999.
- [8] F. Rouillier. On solving parametric systems. In Workshop on *Challenges in Linear and Polynomial Algebra in Symbolic Computation Software*. Banff International Research Center, 2005.
- [9] L. Yang. Recent advances on determining the number of real roots of parametric polynomials. *Journal of Symbolic Computation*, 28:225-242, 1999.
- [10] L. Yang, X. Hou, and Z. Zeng. Complete discrimination system for polynomials. *Science in China (Series E)*, 39(6):628-646, 1996.
- [11] L. Yang and B. Xia. Explicit criterion to determine the number of positive roots of a polynomial. *MM Research Preprints*, 15:134-145, 1997.
- [12] L. Yang and B. Xia. Quantifier elimination for quartics. *Lecture Notes in Computer Science*, 4120:131-145, 2006.

Hilbert's Nullstellensatz and an Algorithm for Proving Combinatorial Infeasibility

Jesús A. De Loera*
Dept. of Mathematics
Univ. of California, Davis
Davis, CA
deloera@math.ucdavis.edu

Peter N. Malkin
Dept. of Mathematics
Univ. of California, Davis
Davis, CA
malkin@math.ucdavis.edu

Jon Lee
Mathematical Programming
IBM T.J. Watson Res. Cntr.
Yorktown Heights, New York
jonlee@us.ibm.com

Susan Margulies
Dept. of Computer Science
Univ. of California, Davis
Davis, CA
smargulies@ucdavis.edu

ABSTRACT

Systems of polynomial equations over an algebraically-closed field $\mathbb{K}$ can be used to concisely model many combinatorial problems. In this way, a combinatorial problem is feasible (e.g., a graph is 3-colorable, hamiltonian, etc.) if and only if a related system of polynomial equations has a solution over $\mathbb{K}$. In this paper, we investigate an algorithm aimed at proving combinatorial infeasibility based on the observed low degree of Hilbert's Nullstellensatz certificates for polynomial systems arising in combinatorics and on large-scale linear-algebra computations over $\mathbb{K}$. We report on experiments based on the problem of proving the non-3-colorability of graphs. We successfully solved graph problem instances having thousands of nodes and tens of thousands of edges.

Categories and Subject Descriptors

G.2.1 [Combinatorics]: Combinatorial algorithms

General Terms

Algorithms

1. INTRODUCTION

It is well known that systems of polynomial equations over a field can yield small models of difficult combinatorial problems. For example, it was first noted by D. Bayer that the 3-colorability of graphs can be modeled via a system of polynomial equations [2]. More generally, one can easily prove the following:

*Research supported in part by an IBM Open Collaborative Research Award and by NSF grant DMS-0608785

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

LEMMA 1.1. *A graph G is k -colorable iff the system of $n + m$ equations in n variables $x_i^k - 1 = 0, \forall i \in V(G)$, and $\sum_{l=0}^{k-1} x_i^{k-1-l} x_j^l = 0, \forall \{i, j\} \in E(G)$ has a complex solution. Moreover, the number of solutions equals the number of distinct k -colorings multiplied by $k!$.*

Although such polynomial system encodings have been used to prove combinatorial results (see [1, 6] and therein), they have not been widely used for computation. The key issue that we investigate here is the use of such polynomial systems to effectively decide whether a graph, or other combinatorial structure, has a property captured by the polynomial system and its associated ideal. We call this the *combinatorial feasibility problem*. We are particularly interested in whether this can be accomplished in practice for large combinatorial structures such as graphs with many nodes.

Certainly, using standard tools in computational algebra such as Gröbner bases, one can answer the combinatorial feasibility problem by simply solving the system of polynomials. Nevertheless, it has been shown by experiments that current Gröbner bases implementations often cannot directly solve polynomial systems with hundreds of polynomials. This paper proposes another approach that relies instead on the nice low degree of the Hilbert's Nullstellensatz for combinatorial polynomial systems and on large-scale linear-algebra computation.

For a hard combinatorial problem (e.g., 3-colorability of graphs), we associate a system of polynomial equations $J = \{f_1(x) = 0, \dots, f_s(x) = 0\}$ such that the system J has a solution if and only if the combinatorial problem has a feasible solution. The Hilbert Nullstellensatz (see e.g., [5]) states that the system of polynomial equations has *no* solution over an algebraically-closed field $\mathbb{K}$ if and only if there exist polynomials $\beta_1, \dots, \beta_s \in \mathbb{K}[x_1, \dots, x_n]$ such that $1 = \sum \beta_i f_i$. Thus, if the polynomial system J has no solution, then there exists a *certificate* that J has no solution, and thus a certificate that the combinatorial problem is infeasible.

The key idea that we explore in this article is to use the Nullstellensatz to generate a finite sequence of linear algebra systems, of increasing size, which will eventually become *feasible* if and only if the combinatorial problem is *infeasible*. Given a system of polynomial equations, we fix a tentative degree k for the coefficient polynomials β_i in the certificates.

We can decide whether there is a Nullstellensatz certificate with coefficients of degree $\leq k$ by solving a system of linear equations over the field $\mathbb{K}$ whose variables are in bijection with the coefficients of the monomials of the polynomials $\beta_1, \dots, \beta_s$. If this linear system has a solution, we have found a certificate; otherwise, we try a higher degree for the polynomials β_i . This process is guaranteed to terminate because, for a Nullstellensatz certificate to exist, the degrees of the polynomials β_i cannot be more than known bounds (see [10] and references therein). We explain the details of the algorithm, which we call NullA, in Section 2.

Our method can be seen as a general-field variation of work by Lasserre [11], Laurent [13] and Parrilo [16] and many others, who studied the problem of minimizing a general polynomial function $f(x)$ over a real algebraic variety with finitely many points. Laurent proved that when the variety consists of the solutions of a zero-dimensional radical ideal I , one can set up the optimization problem $\min\{f(x) : x \in \text{variety}(I)\}$ as a finite sequence of semidefinite programs terminating with the optimal solution (see [13]). There are two key observations that speed up practical calculations considerably: (1) when dealing with feasibility, instead of optimization, linear algebra replaces semidefinite programming and (2) there are ways of controlling the length of the sequence of linear-algebra systems including finite field computation instead of calculations over the reals and the reduction of matrix size by symmetries. See Section 3 for details.

Our algorithm has good practical performance and numerical stability. Although known theoretical bounds for degrees of the Nullstellensatz coefficients are doubly-exponential in the size of the polynomial system (and indeed there exist examples that attain such a large bound and make NullA useless in general), our experiments demonstrate that often low degrees suffice for systems of polynomials coming from graphs. We have implemented an exact-arithmetic linear system solver optimized for these Nullstellensatz-based systems. We performed many experiments using NullA, focusing on the problem of deciding graph 3-colorability (note that the method is applicable to any combinatorial problem whose polynomial system encoding is known). We conclude with a report on these experiments in Section 4.

2. NULLSTELLENSATZ LINEAR ALGEBRA (NULLA) ALGORITHM

Recall that Hilbert's Nullstellensatz states that a system of polynomial equations $f_1(x) = 0, \dots, f_s(x) = 0$, where $f_i \in \mathbb{K}[x_1, \dots, x_n]$ and $\mathbb{K}$ is an algebraically closed field, has no solution in $\mathbb{K}^n$ if and only if there exist polynomials $\beta_1, \dots, \beta_s \in \mathbb{K}[x_1, \dots, x_n]$ such that $1 = \sum \beta_i f_i$ [5]. The polynomial identity $1 = \sum \beta_i f_i$ is called a *Nullstellensatz certificate*, which has degree d if $\max\{\deg(\beta_i)\} = d$.

The Nullstellensatz Linear Algebra (NullA) algorithm accepts as input a system of polynomial equations and outputs either a *yes answer*, if the system of polynomial equations has a solution, or a *no answer*, along with a Nullstellensatz infeasibility certificate, if the system has no solution. Before stating the algorithm in pseudocode, we clarify the connection to linear algebra. Suppose the input polynomial system is infeasible over $\mathbb{K}$, and suppose further that an oracle has told us the certificate has degree d but that we do not know the actual coefficients of the degree d polynomials β_i . Thus,

we have the polynomial identity $1 = \sum \beta_i f_i$. If we expand the identity into monomials, the coefficients of a monomial are linear expressions in the coefficients of the β_i . Since two polynomials over a field are identical precisely when the coefficients of corresponding monomials are identical, from the $1 = \sum \beta_i f_i$, we get a system of linear equations whose variables are the coefficients of the β_i . Here is an example:

EXAMPLE 2.1. *Consider the polynomial system $x_1^2 - 1 = 0, x_1 + x_2 = 0, x_1 + x_3 = 0, x_2 + x_3 = 0$. This system has no solution, and a Nullstellensatz certificate of degree one.*

$$\begin{aligned} 1 &= \underbrace{(c_0x_1 + c_1x_2 + c_2x_3 + c_3)}_{\beta_1} \underbrace{(x_1^2 - 1)}_{f_1} \\ &\quad + \underbrace{(c_4x_1 + c_5x_2 + c_6x_3 + c_7)}_{\beta_2} \underbrace{(x_1 + x_2)}_{f_2} \\ &\quad + \underbrace{(c_8x_1 + c_9x_2 + c_{10}x_3 + c_{11})}_{\beta_3} \underbrace{(x_1 + x_3)}_{f_3} \\ &\quad + \underbrace{(c_{12}x_1 + c_{13}x_2 + c_{14}x_3 + c_{15})}_{\beta_4} \underbrace{(x_2 + x_3)}_{f_4}. \end{aligned}$$

Expanding the tentative Nullstellensatz certificate into monomials and grouping like terms, we arrive at the following polynomial equation:

$$\begin{aligned} 1 &= c_0x_1^3 + c_1x_1^2x_2 + c_2x_1^2x_3 + (c_3 + c_4 + c_8)x_1^2 - c_3 \\ &\quad + (c_{10} + c_{14})x_3^2 + (c_4 + c_5 + c_9 + c_{12})x_1x_2 + (c_5 + c_{13})x_2^2 \\ &\quad + (c_6 + c_8 + c_{10} + c_{12})x_1x_3 + (c_6 + c_9 + c_{13} + c_{14})x_2x_3 \\ &\quad + (c_7 + c_{15} - c_1)x_2 + (c_{11} + c_{15} - c_2)x_3 + (c_7 + c_{11} - c_0)x_1. \end{aligned}$$

From this, we extract a system of linear equations. Since a Nullstellensatz certificate is identically one, all monomials except the constant term must be equal to zero; namely:

$$c_0 = 0, \quad c_{11} + c_{15} - c_2 = 0, \quad -c_3 = 1.$$

By solving the system of linear equations, we reconstruct the Nullstellensatz certificate from the solution. Indeed

$$1 = \frac{1}{2}x_1(x_1 + x_2) - \frac{1}{2}x_1(x_2 + x_3) + \frac{1}{2}x_1(x_1 + x_3) - (x_1^2 - 1)$$

In general, one does not know the degree of the Nullstellensatz certificate in advance. What one can do is to start with a tentative degree, say start at degree one, produce the corresponding linear system, and solve it. If the system has a solution, then we have found a Nullstellensatz certificate demonstrating that the original input polynomials do not have a common root. Otherwise, we increment the degree until we can be sure that there will not be a Nullstellensatz certificate at all, and thus we can conclude the system of polynomials has a solution. The number of iterations of the above steps determines the running time of NullA. For this, there are well-known upper bounds on the degree of the Nullstellensatz certificate [10]. These upper bounds for the degrees of the coefficients β_i in the Hilbert Nullstellensatz certificates for *general* systems of polynomials are doubly-exponential in the number of input polynomials and their degree. Unfortunately, these bounds are known to be sharp for some specially-constructed systems. Although this immediately says that NullA is not practical for arbitrary polynomial systems, we have observed in practice that polynomial systems for combinatorial questions are extremely specialized, and the degree growth is often *very* slow — enough to deal with large graphs or other combinatorial structures. Now we describe NullA in pseudocode:


```

*****
ALGORITHM: Nullstellensatz Linear Algebra (NulLA)
           Algorithm)
INPUT: A system of polynomial equations
        $F = \{f_1(x) = 0, \dots, f_s(x) = 0\}$ 
OUTPUT: YES, if  $F$  has solution, else NO along with a
        Nullstellensatz certificate of infeasibility.

 $d \leftarrow 1$ .
 $K \leftarrow$  known upper bounds on degree of Nullstellensatz
for  $F$  (see e.g., [10])
while  $d \leq K$  do
  CERT  $\leftarrow \sum_{i=1}^s \beta_i f_i$  (where  $\beta_i$  are polynomials of
  degree  $d$ , with unknowns for their coefficients).
  Extract a system of linear equations from CERT with
  columns corresponding to unknowns,
  and rows corresponding to monomials.
  Solve the linear system.
  if the linear system is consistent then
    CERT  $\leftarrow \sum_{i=1}^s \beta_i f_i$  (with unknowns in  $\beta_i$ 
    replaced with linear system solution values.)
    print "The system of equations  $F$  is infeasible."
    return NO with CERT.
  end if
   $d \leftarrow d + 1$ .
end while
print "The system of equations  $F$  is feasible."
return YES.
*****

```

This opens several theoretical questions. It is natural to ask about lower bounds on the degree of the Nullstellensatz certificates. Little is known, but recently it was shown in [6], that for the problem of deciding whether a given graph G has an independent set of a given size, a minimum-degree Nullstellensatz certificate for the non-existence of an independent set of size greater than $\alpha(G)$ (the size of the largest independent set in G) has degree equal to $\alpha(G)$, and it is very dense; specifically, it contains at least one term per independent set in G . For polynomial systems coming from logic there has also been an effort to show degree growth in related polynomial systems (see [3, 8] and the references therein). Another question is to provide tighter, more realistic upper bounds for concrete systems of polynomials. It is a challenge to settle it for any concrete family of polynomial systems.

3. FOUR MATHEMATICAL IDEAS TO OPTIMIZE NULLA

Since we are interested in practical computational problems, it makes sense to explore refinements and variations that make NulLA robust and much faster for concrete challenges. The main computational component of NulLA is to construct and solve linear systems for finding Nullstellensatz certificates of increasing degree. These linear systems are typically very large for reasonably-sized problems, even for certificate degrees as low as four, which can produce linear systems with millions of variables (see Section 4). Furthermore, the size of the linear system increases dramatically with the degree of the certificate. In particular, the number of variables in the linear system to find a Nullstellensatz certificate of degree d is precisely $s \binom{n+d}{d}$ where n is the number of variables in the polynomial system and s is the number of polynomials. Note that $\binom{n+d}{d}$ is the number of possible monomials of degree d or less. Also, the number of non-zero entries in the constraint matrix is precisely $M \binom{n+d}{d}$ where M is the sum over the number of monomials in each polynomial of the system.

For this reason, in this section, we explore mathematical approaches for solving the linear system more efficiently and robustly, for decreasing the size of the linear system for a given degree, and for decreasing the degree of the Nullstellensatz certificate for infeasible polynomial systems thus significantly reducing the size of the largest linear system that we need to solve to prove infeasibility. Note that these approaches to reduce the degree do not necessarily decrease the available upper bound on the degree of the Nullstellensatz certificate required for proving feasibility.

It is certainly possible to significantly decrease the size of the linear system by preprocessing the given polynomial system to remove redundant polynomial equations and also by preprocessing the linear system itself to eliminate many variables. For example, in the case of 3-coloring problems for connected graphs, since $(x_i^3 + 1) = (x_j^3 + 1) + (x_i + x_j)(x_i^2 + x_i x_j + x_j^2)$, we can remove all but one of the vertex polynomials by tracing paths through the graph. However, preprocessing alone is not sufficient to enable us to solve some large polynomial systems.

The mathematical ideas we explain in the rest of this section can be applied to arbitrary polynomial systems, but to implement them, one has to look for the right structures in the polynomials.

3.1 NulLA over Finite Fields

The first idea is that, for combinatorial problems, one can often carry out calculations over finite fields instead of relying on unstable floating-point calculations. We illustrate this with the problem of deciding whether the vertices of a graph permit a proper 3-coloring. The following encoding (a variation of [2] over the complex numbers) allows us to compute over $\mathbb{F}_2$, which is robust and much faster in practice (also see [7]):

LEMMA 3.1. *The graph G is 3-colorable if and only if the zero-dimensional system of equations $x_i^3 + 1 = 0, \forall i \in V(G)$, and $x_i^2 + x_i x_j + x_j^2 = 0, \forall \{i, j\} \in E(G)$, has a solution over $\mathbb{F}_2$, the algebraic closure of $\mathbb{F}_2$.*

Before we prove Lemma 3.1, we introduce a convenient notation: Let α be an algebraic element over $\mathbb{F}_2$ such that $\alpha^2 + \alpha + 1 = 0$. Thus, although $x_i^3 + 1$ has only one root over $\mathbb{F}_2$, since $x_i^3 + 1 = (x_i + 1)(x_i^2 + x_i + 1)$, the polynomial $x_i^3 + 1$ has three roots over $\mathbb{F}_2$, which are 1, α and $\alpha + 1$.

PROOF. If the graph G is 3-colorable, simply map the three colors to 1, α and $\alpha + 1$. Clearly, the vertex polynomial equations $x_i^3 + 1 = 0$ are satisfied. Furthermore, given an edge $\{i, j\}$, $x_i + x_j \neq 0$ since variable assignments correspond to a proper 3-coloring and adjacent vertices are assigned different roots. This implies that $x_i^3 + x_j^3 = (x_i + x_j)(x_i^2 + x_i x_j + x_j^2) = 1 + 1 = 0$. Therefore, $x_i^2 + x_i x_j + x_j^2 = 0$ and the edge polynomial equations are satisfied.

Conversely, suppose that there exists a solution to the system of polynomial equations. Clearly, every vertex is assigned either 1, α or $\alpha + 1$. We will show that adjacent vertices are assigned different values. Our proof is by contradiction: Assume that two adjacent vertices i, j are assigned the same value β . Then, $0 = x_i^2 + x_i x_j + x_j^2 = \beta^2 + \beta^2 + \beta^2 = 3\beta^2 \neq 0$. Therefore, adjacent vertices are assigned different roots, and a solution to the system corresponds directly to a proper 3-coloring. $\square$

We remark that this result can be extended to k -colorability and $\mathbb{F}_q$, when q is relatively prime to k . The following computational lemma will allow us to certify graph non-3-colorability very rapidly over $\mathbb{F}_2$ instead of working over its algebraic closure.

LEMMA 3.2. *Let $\mathbb{K}$ be a field and $\overline{\mathbb{K}}$ its algebraic closure. Given $f_1, f_2, \dots, f_s \in \mathbb{K}[x_1, \dots, x_n]$, there exists a Nullstellensatz certificate $1 = \sum \beta_i f_i$ where $\beta_i \in \mathbb{K}[x_1, \dots, x_n]$ if and only if there exists a Nullstellensatz certificate $1 = \sum \beta'_i f_i$ where $\beta'_i \in \mathbb{K}[x_1, \dots, x_n]$.*

PROOF. If there exists a Nullstellensatz certificate $1 = \sum \beta_i f_i$ where $\beta_i \in \mathbb{K}[x_1, \dots, x_n]$, via NullA, construct the associated linear system and solve. Since $f_i \in \mathbb{K}[x_1, \dots, x_n]$, the coefficients in the linear system will consist only of values in $\mathbb{K}$. Thus, solving the linear system relies only on computations in $\mathbb{K}$, and if the free variables are chosen from $\mathbb{K}$ instead of $\overline{\mathbb{K}}$, the resulting Nullstellensatz certificate $1 = \sum \beta'_i f_i$ has $\beta'_i \in \mathbb{K}[x_1, \dots, x_n]$. The reverse implication is trivial. $\square$

Therefore, we have the following corollary:

COROLLARY 3.3. *A graph G is non-3-colorable if and only if there exists a Nullstellensatz certificate $1 = \sum \beta_i f_i$ where $\beta_i \in \mathbb{F}_2[x_1, \dots, x_n]$ where the polynomials $f_i \in \mathbb{F}_2[x_1, \dots, x_n]$ are as defined in Lemma 3.1.*

This corollary enables us to compute over $\mathbb{F}_2$, which is extremely fast in practice (see Section 4).

Finally, the degree of Nullstellensatz certificates necessary to prove infeasibility can be lower over $\mathbb{F}_2$ than over the rationals. For example, over the rationals, every odd-wheel has a minimum non-3-colorability certificate of degree four [6]. However, over $\mathbb{F}_2$, every odd-wheel has a Nullstellensatz certificate of degree one. Therefore, not only are the mathematical computations more efficient over $\mathbb{F}_2$ as compared to the rationals, but the algebraic properties of the certificates themselves are sometimes more favorable for computation as well.

3.2 NullA with symmetries

Let us assume that the input polynomial system $F = \{f_1, \dots, f_s\}$ has maximum degree q and that n is the number of variables present. As we observed in Section 2, for a given fixed positive integer d serving as a tentative degree for the Nullstellensatz certificate, the Nullstellensatz coefficients come from the solution of a system of linear equations. We now take a closer look at the matrix equation $M_{F,d} y = b_{F,d}$ defining the system of linear equations. First of all, the matrix $M_{F,d}$ has one row per monomial x^α of degree less than or equal to $q + d$ on the n variables and one column per polynomial of the form $x^\delta f_i$, i.e., the product of a monomial x^δ of degree less than or equal to d and a polynomial $f_i \in F$. Thus, $M_{F,d} = (M_{x^\alpha, x^\delta f_i})$ where $M_{x^\alpha, x^\delta f_i}$ equals the coefficient of the monomial x^α in the polynomial $x^\delta f_i$. The variable y has one entry for every polynomial of the form $x^\delta f_i$ denoted $y_{x^\delta f_i}$, and the vector $b_{F,d}$ has one entry for every monomial x^α of degree less than or equal to $q + d$ where $(b_{F,d})_{x^\alpha} = 0$ if $\alpha \neq 0$ and $(b_{F,d})_1 = 1$.

EXAMPLE 3.4. *Consider the complete graph K_4 . The shape of a degree-one Hilbert Nullstellensatz certificate over $\mathbb{F}_2$ for non-3-colorability is as follows:*

$$1 = c_0(x_1^3 + 1)$$

$$\begin{aligned} &+ (c_{12}^1 x_1 + c_{12}^2 x_2 + c_{12}^3 x_3 + c_{12}^4 x_4)(x_1^2 + x_1 x_2 + x_2^2) \\ &+ (c_{13}^1 x_1 + c_{13}^2 x_2 + c_{13}^3 x_3 + c_{13}^4 x_4)(x_1^2 + x_1 x_3 + x_3^2) \\ &+ (c_{14}^1 x_1 + c_{14}^2 x_2 + c_{14}^3 x_3 + c_{14}^4 x_4)(x_1^2 + x_1 x_4 + x_4^2) \\ &+ (c_{23}^1 x_1 + c_{23}^2 x_2 + c_{23}^3 x_3 + c_{23}^4 x_4)(x_2^2 + x_2 x_3 + x_3^2) \\ &+ (c_{24}^1 x_1 + c_{24}^2 x_2 + c_{24}^3 x_3 + c_{24}^4 x_4)(x_2^2 + x_2 x_4 + x_4^2) \\ &+ (c_{34}^1 x_1 + c_{34}^2 x_2 + c_{34}^3 x_3 + c_{34}^4 x_4)(x_3^2 + x_3 x_4 + x_4^2) \end{aligned}$$

Note that we have preprocessed the certificate by removing the redundant polynomials $x_i^3 + 1$ where $i \neq 1$ and removing some variables that we know a priori can be set to zero, which results in a matrix with less columns. As we explained in Section 2, this certificate gives a linear system of equations in the variables c_0 and c_{ij}^k (note that k is a superscript and not an exponent). This linear system can be captured as the matrix equation $M_{F,1} c = b_{F,1}$ where the matrix $M_{F,1}$ is as follows.

	c_0	c_{12}^1	c_{12}^2	c_{12}^3	c_{12}^4	c_{13}^1	c_{13}^2	c_{13}^3	c_{13}^4	$\dots$	c_{34}^1	c_{34}^2	c_{34}^3	c_{34}^4
1	1	0	0	0	0	0	0	0	0	$\dots$	0	0	0	0
x_1^3	1	1	0	0	0	1	0	0	0	$\dots$	0	0	0	0
$x_1^2 x_2$	0	1	1	0	0	0	1	0	0	$\dots$	0	0	0	0
$x_1^2 x_3$	0	0	0	1	0	1	0	1	0	$\dots$	0	0	0	0
$x_1^2 x_4$	0	0	0	0	1	0	0	0	1	$\dots$	0	0	0	0
$x_1 x_2^2$	0	1	1	0	0	0	0	0	0	$\dots$	0	0	0	0
$x_1 x_2 x_3$	0	0	0	1	0	0	1	0	0	$\dots$	0	0	0	0
$x_1 x_2 x_4$	0	0	0	0	1	0	0	0	0	$\dots$	0	0	0	0
$x_1 x_3^2$	0	0	0	0	0	1	0	1	0	$\dots$	1	0	0	0
$x_1 x_3 x_4$	0	0	0	0	0	0	0	0	1	$\dots$	1	0	0	0
$x_1 x_4^2$	0	0	0	0	0	0	0	0	0	$\dots$	1	0	0	0
x_2^3	0	0	1	0	0	0	0	0	0	$\dots$	0	0	0	0
$x_2^2 x_3$	0	0	0	1	0	0	0	0	0	$\dots$	0	0	0	0
$x_2^2 x_4$	0	0	0	0	1	0	0	0	0	$\dots$	0	0	0	0
$x_2 x_3^2$	0	0	0	0	0	0	1	0	0	$\dots$	0	1	0	0
$x_2 x_3 x_4$	0	0	0	0	0	0	0	0	0	$\dots$	0	1	0	0
$x_2 x_4^2$	0	0	0	0	0	0	0	0	0	$\dots$	0	1	0	0
x_3^3	0	0	0	0	0	0	0	1	0	$\dots$	0	0	1	0
$x_3^2 x_4$	0	0	0	0	0	0	0	0	1	$\dots$	0	0	1	1
$x_3 x_4^2$	0	0	0	0	0	0	0	0	0	$\dots$	0	0	1	1
x_4^3	0	0	0	0	0	0	0	0	0	$\dots$	0	0	0	1

Certainly the matrix $M_{F,d}$ is rather large already for small systems of polynomials. The main point of this section is to demonstrate how to reduce the size of the matrix by using a group action on the variables, e.g., using symmetries or automorphisms in a graph. Suppose we have a finite permutation group G acting on the variables $x_1, \dots, x_n$. Clearly G induces an action on the set of monomials with variables $x_1, x_2, \dots, x_n$ of degree t . We will assume that the set F of polynomials is invariant under the action of G , i.e., $g(f_i) \in F$ for each $f_i \in F$. Denote by x^δ , the monomial $x_1^{\delta_1} x_2^{\delta_2} \dots x_n^{\delta_n}$, a monomial of degree $\delta_1 + \delta_2 + \dots + \delta_n$. Denote by $Orb(x^\alpha), Orb(x^\delta f_i)$ the orbit under G of monomial x^α and, respectively, the orbit of the polynomial obtained as the product of the monomial x^δ and the polynomial $f_i \in F$.

We now introduce a new matrix equation $\bar{M}_{F,d,G} \bar{y} = \bar{b}_{F,d,G}$. The rows of the matrix $\bar{M}_{F,d,G}$ are indexed by the orbits of monomials $Orb(x^\alpha)$ where x^α is a monomial of degree less than or equal to $q + d$, and the columns of $\bar{M}_{F,d,G}$ are indexed by the orbits of polynomials $Orb(x^\delta f_i)$ where $f_i \in F$ and the degree of the monomial x^δ less than or equal to d . Then, let $\bar{M}_{F,d,G} = (\bar{M}_{Orb(x^\alpha), Orb(x^\delta f_i)})$ where

$$\bar{M}_{Orb(x^\alpha), Orb(x^\delta f_i)} = \sum_{x^\gamma f_j \in Orb(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j}.$$

Note that $M_{x^\alpha, x^\delta f_i} = M_{g(x^\alpha), g(x^\delta f_i)}$ for all $g \in G$ meaning that the coefficient of the monomial x^α in the polynomial $x^\delta f_i$ is the same as the coefficient of the monomial $g(x^\alpha)$ in the polynomial $g(x^\delta f_i)$. So, $\forall x^\alpha \in \text{Orb}(x^\alpha)$,

$$\sum_{x^\gamma f_j \in \text{Orb}(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j} = \sum_{x^\gamma f_j \in \text{Orb}(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j},$$

and thus, $\bar{M}_{\text{Orb}(x^\alpha), \text{Orb}(x^\delta f_i)}$ is well-defined. We call the matrix $\bar{M}_{F,d,G}$ the *orbit matrix*. The variable $\bar{y}$ has one entry for every polynomial orbit $\text{Orb}(x^\delta f_i)$ denoted $\bar{y}_{\text{Orb}(x^\delta f_i)}$. The vector $\bar{b}_{F,d}$ has one entry for every monomial orbit $\text{Orb}(x^\alpha)$, and let $(\bar{b}_{F,d})_{\text{Orb}(x^\alpha)} = (b_{F,d})_{x^\alpha} = 0$ if $\alpha \neq 0$ and $(\bar{b}_{F,d})_{\text{Orb}(1)} = (b_{F,d})_1 = 1$. The main result in this section is that, under some assumptions, the system of linear equations $\bar{M}_{F,d,G} \bar{y} = \bar{b}_{F,d,G}$ has a solution if and only if the larger system of linear equations $M_{F,d} y = b_{F,d}$ has a solution.

THEOREM 3.5. *Let $F = \{f_1, \dots, f_s\} \subset \mathbb{K}[x_1, \dots, x_n]$, be a polynomial system with $\mathbb{K}$ an algebraically-closed field, and a finite group of permutations $G \subset S_n$. Let $M_{F,d}, \bar{M}_{F,d,G}$ denote the matrices defined above. Suppose that the polynomial system F is closed under the action of the group G permuting the indices of variables $x_1, \dots, x_n$. Suppose further that the order of the group $|G|$ and the characteristic of the field $\mathbb{K}$ are relatively prime. The degree d Nullstellensatz linear system of equations $M_{F,d} y = b_{F,d}$ has a solution over $\mathbb{K}$ if and only if the system of linear equations $\bar{M}_{F,d,G} \bar{y} = \bar{b}_{F,d,G}$ has a solution over $\mathbb{K}$.*

PROOF. To simplify notation, let $M = M_{F,d}$, $b = b_{F,d}$, $\bar{M} = \bar{M}_{F,d,G}$ and $\bar{b} = \bar{b}_{F,d,G}$. First, we show that if the linear system $My = b$ has a solution, then there exists a *symmetric* solution y of the linear system $My = b$ meaning that $y_{x^\delta f_i}$ is the same for all $x^\delta f_i$ in the same orbit, i.e., $y_{x^\gamma f_j} = y_{x^\delta f_i}$ for all $x^\gamma f_j \in \text{Orb}(x^\delta f_i)$. The converse is also trivially true.

Since the rows and columns of the matrix M are labeled by monomials x^α and polynomials $x^\delta f_i$ respectively, we can think of the group G as acting on the matrix M , permuting the entries M , i.e., applying $g \in G$ to M gives the permuted matrix $g(M)$ where

$$g(M)_{g(x^\alpha), g(x^\delta f_i)} = M_{x^\alpha, x^\delta f_i}.$$

Moreover, since $M_{x^\alpha, x^\delta f_i} = M_{g(x^\alpha), g(x^\delta f_i)}$ for all $g \in G$, we must have $g(M) = M$, so the matrix M is invariant under the action of the group G . Also, since the entries of the variable y are labeled by polynomials of the form $x^\alpha f_i$, we can also think of the group G as acting on the vector y , permuting the entries of the vector y , i.e., applying $g \in G$ to y gives the permuted vector $g(y)$ where $g(y)_{g(x^\delta f_i)} = y_{x^\delta f_i}$. Similarly, G acts on the vector b , and in particular, $g(b) = b$. Next, we show that if $My = b$, then $Mg(y) = b$ for all $g \in G$ accordingly:

$$My = b \Rightarrow g(My) = g(b) \Rightarrow g(M)g(y) = b \Rightarrow Mg(y) = b,$$

for all $g \in G$. Now, let

$$y' = \frac{1}{|G|} \sum_{g \in G} g(y).$$

Note we need that $|G|$ is relatively prime to the characteristic of the field $\mathbb{K}$ so that $|G|$ is invertible. Then,

$$My' = \frac{1}{|G|} \sum_{g \in G} Mg(y) = \frac{1}{|G|} \sum_{g \in G} b = b,$$

so y' is a solution. Also, $y'_{x^\delta f_i} = \frac{1}{|G|} \sum_{g \in G} y_{g(x^\delta f_i)}$, so $y'_{x^\delta f_i} = y'_{x^\gamma f_j}$ for all $x^\gamma f_j \in \text{Orb}(x^\delta f_i)$. Therefore, y' is a symmetric solution as required.

Now, assume that there exists a solution of $My = b$. By the above argument, we can assume that the solution is symmetric, i.e., $y_{x^\delta f_i} = y_{x^\gamma f_j}$ where $g(x^\delta f_i) = x^\gamma f_j$ for some $g \in G$. From this symmetric solution of $My = b$, we can find a solution of $\bar{M}\bar{y} = \bar{b}$ by setting

$$\bar{y}_{\text{Orb}(x^\delta f_i)} = y_{x^\delta f_i}.$$

To show this, we check that $(\bar{M}\bar{y})_{\text{Orb}(x^\alpha)} = \bar{b}_{\text{Orb}(x^\alpha)}$ for every monomial x^α .

$$\begin{aligned} (\bar{M}\bar{y})_{\text{Orb}(x^\alpha)} &= \sum_{\text{all } \text{Orb}(x^\delta f_i)} \bar{M}_{\text{Orb}(x^\alpha), \text{Orb}(x^\delta f_i)} \bar{y}_{\text{Orb}(x^\delta f_i)} \\ &= \sum_{\text{all } \text{Orb}(x^\delta f_i)} \left(\sum_{x^\gamma f_j \in \text{Orb}(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j} \right) \bar{y}_{\text{Orb}(x^\delta f_i)} \\ &= \sum_{\text{all } \text{Orb}(x^\delta f_i)} \left(\sum_{x^\gamma f_j \in \text{Orb}(x^\delta f_i)} M_{x^\alpha, x^\gamma f_j} y_{x^\gamma f_j} \right) \\ &= \sum_{\text{all } x^\delta f_i} M_{x^\alpha, x^\delta f_i} y_{x^\delta f_i} = (My)_{x^\alpha}. \end{aligned}$$

Thus, $(\bar{M}\bar{y})_{\text{Orb}(x^\alpha)} = \bar{b}_{\text{Orb}(x^\alpha)}$ since $(My)_{x^\alpha} = b_{x^\alpha} = \bar{b}_{\text{Orb}(x^\alpha)}$.

Next, we establish the converse more easily. Recall that the columns of $\bar{M}$ are labeled by orbits. If there is a solution for $\bar{M}\bar{y} = \bar{b}$, then to recover a solution of $My = b$, we set

$$y_{x^\delta f_i} = \bar{y}_{\text{Orb}(x^\delta f_i)}.$$

Note that y is a symmetric solution. Using the same calculation as above, we have that $(My)_{x^\alpha} = (\bar{M}\bar{y})_{\text{Orb}(x^\alpha)}$, and thus, $My = b$. $\square$

EXAMPLE 3.6 (CONTINUATION OF EXAMPLE 3.4). *Now consider the action of the symmetry group G generated by the cycle $(2,3,4)$ (a cyclic group of order three). The permutation of variables permutes the monomials and yields a matrix $M_{F,1,G}$. We have now grouped together monomials and terms within orbit blocks in the matrix below. The blocks will be later replaced by a single entry, shrinking the size of the matrix.*

	c_0	c_{12}	c_{13}	c_{14}	c_{12}^2	c_{13}^2	c_{14}^2	c_{12}^3	c_{13}^3	c_{14}^3	c_{12}^4	c_{13}^4	c_{14}^4	c_{12}^5	c_{13}^5	c_{14}^5	c_{12}^6	c_{13}^6	c_{14}^6
1	1	0	0	0	0	0	0	0	0	0	$\cdots$	0	0	0	0	0	0	0	0
x_1^3	1	1	1	1	0	0	0	0	0	0	$\cdot$	0	0	0	0	0	0	0	0
$x_2^2 x_2$	0	1	0	0	1	0	0	0	0	1	$\cdots$	0	0	0	0	0	0	0	0
$x_1^3 x_3$	0	0	1	0	0	1	0	1	0	0	$\cdots$	0	0	0	0	0	0	0	0
$x_1^2 x_4$	0	0	0	1	0	0	1	0	1	0	$\cdots$	0	0	0	0	0	0	0	0
$x_1 x_2^3$	0	1	0	0	1	0	0	0	0	0	$\cdots$	0	0	0	0	0	0	0	0
$x_1 x_2^3 x_3$	0	0	1	0	0	1	0	0	0	0	$\cdots$	0	0	0	0	0	0	0	0
$x_1 x_2^3 x_4$	0	0	0	1	0	0	1	0	0	0	$\cdots$	0	0	0	0	0	0	0	0
$x_1 x_2 x_3$	0	0	0	0	0	0	0	1	0	0	$\cdots$	0	0	0	0	0	0	0	0
$x_1 x_2 x_4$	0	0	0	0	0	0	0	0	0	1	$\cdots$	0	0	0	0	0	0	0	0
$x_1 x_3 x_4$	0	0	0	0	0	0	0	0	0	1	$\cdots$	0	0	0	0	0	0	0	0
x_3^3	0	0	0	0	1	0	0	0	0	0	$\cdots$	0	0	0	0	0	0	0	0
$x_3^3 x_3$	0	0	0	0	0	1	0	0	0	0	$\cdots$	0	0	0	0	0	0	0	0
$x_3^3 x_4$	0	0	0	0	0	0	1	0	0	0	$\cdots$	0	0	0	0	0	0	0	0
$x_2^2 x_3$	0	0	0	0	0	0	0	1	0	0	$\cdots$	0	1	0	0	0	0	0	0
$x_3^3 x_4$	0	0	0	0	0	0	0	0	1	0	$\cdots$	0	0	0	1	0	0	0	1
$x_2 x_2^2$	0	0	0	0	0	0	0	0	0	0	$\cdots$	0	0	0	1	0	0	0	1
$x_2 x_2^2 x_3$	0	0	0	0	0	0	0	0	0	0	$\cdots$	1	0	0	0	0	0	0	0
$x_2 x_3 x_4$	0	0	0	0	0	0	0	0	0	0	$\cdots$	0	0	1	0	0	0	1	0
$x_2 x_3 x_4$	0	0	0	0	0	0	0	0	0	0	$\cdots$	1	1	1	1	1	1	1	1

The action of the symmetry group generated by the cycle $(2,3,4)$ yields an orbit matrix $\bar{M}_{F,q,G}$ of about a third the size of the original one:

	$\bar{c}_0$	$\bar{c}_{12}^1$	$\bar{c}_{12}^2$	$\bar{c}_{12}^3$	$\bar{c}_{12}^4$	$\bar{c}_{23}^1$	$\bar{c}_{23}^2$	$\bar{c}_{23}^3$	$\bar{c}_{23}^4$	$\bar{c}_{24}^1$	$\bar{c}_{24}^2$	$\bar{c}_{24}^3$	$\bar{c}_{24}^4$
$Orb(1)$	1	0	0	0	0	0	0	0	0	0	0	0	0
$Orb(x_1^3)$	1	3	0	0	0	0	0	0	0	0	0	0	0
$Orb(x_1^2 x_2)$	0	1	1	1	1	0	0	0	0	0	0	0	0
$Orb(x_1 x_2^2)$	0	1	1	0	0	2	0	0	0	0	0	0	0
$Orb(x_1 x_2 x_3)$	0	0	0	1	1	1	0	0	0	0	0	0	0
$Orb(x_2^3)$	0	0	1	0	0	0	1	1	0	0	0	0	0
$Orb(x_2^2 x_3)$	0	0	0	1	0	0	1	1	1	1	1	1	1
$Orb(x_2^2 x_4)$	0	0	0	0	1	0	1	0	1	1	1	1	1
$Orb(x_2 x_3 x_4)$	0	0	0	0	0	0	0	0	0	0	0	3	3

(mod 2)

	$\bar{c}_0$	$\bar{c}_{12}^1$	$\bar{c}_{12}^2$	$\bar{c}_{12}^3$	$\bar{c}_{12}^4$	$\bar{c}_{23}^1$	$\bar{c}_{23}^2$	$\bar{c}_{23}^3$	$\bar{c}_{23}^4$	$\bar{c}_{24}^1$	$\bar{c}_{24}^2$	$\bar{c}_{24}^3$	$\bar{c}_{24}^4$
$Orb(1)$	1	0	0	0	0	0	0	0	0	0	0	0	0
$Orb(x_1^3)$	1	1	0	0	0	0	0	0	0	0	0	0	0
$Orb(x_1^2 x_2)$	0	1	1	1	1	0	0	0	0	0	0	0	0
$Orb(x_1 x_2^2)$	0	1	1	0	0	0	0	0	0	0	0	0	0
$Orb(x_1 x_2 x_3)$	0	0	0	1	1	1	0	0	0	0	0	0	0
$Orb(x_2^3)$	0	0	1	0	0	0	1	1	0	0	0	0	0
$Orb(x_2^2 x_3)$	0	0	0	1	0	0	1	1	1	1	1	1	1
$Orb(x_2^2 x_4)$	0	0	0	0	1	0	1	1	1	1	1	1	1
$Orb(x_2 x_3 x_4)$	0	0	0	0	0	0	0	0	0	0	0	1	1

If $|G|$ is not relatively prime to the characteristic of the field $\mathbb{K}$, then it is still true that, if $\bar{M}y = \bar{b}$ has a solution, then $My = b$ has a solution. Thus, even if $|G|$ is not relatively prime to the characteristic of the field $\mathbb{K}$, we can still prove that the polynomial system F is infeasible by finding a solution of the linear system $\bar{M}y = \bar{b}$.

3.3 Reducing the Nullstellensatz degree by appending polynomial equations

We have discovered that by *appending* certain valid but redundant polynomial equations to the system of polynomial equations described in Lemma 3.1, we have been able to *decrease* the degree of the Nullstellensatz certificate necessary to prove infeasibility. A valid but redundant polynomial equation is any polynomial equation $g(x) = 0$ that is true for all the zeros of the polynomial system $f_1(x) = 0, \dots, f_s(x) = 0$, i.e., $g \in \sqrt{I}$, the radical ideal of I , where I is the ideal generated by $f_1, \dots, f_s$. In fact, we only really require that $g(x) = 0$ holds for at least one of zeros of the polynomial system $f_1(x) = 0, \dots, f_s(x) = 0$ if a zero exists. We refer to a redundant polynomial equation appended to a system of polynomial equations, with the goal of reducing the degree of a Nullstellensatz certificate, as a *degree-cutter*.

For example, for 3-coloring, consider a triangle described by the vertices $\{x, y, z\}$. Whenever a triangle appears as a subgraph in a graph, the vertices of the triangle must be colored differently. We capture that additional requirement with the equation

$$x^2 + y^2 + z^2 = 0, \quad (1)$$

which is satisfied if and only if $x \neq y \neq z \neq x$ since x, y and z are third roots of unity. Note that the equation $x + y + z = 0$ also implies $x \neq y \neq z \neq x$, but we use the equation $x^2 + y^2 + z^2 = 0$, which is homogeneous of degree two, because the edge equations from Lemma 3.1 are also homogeneous of degree two, and this helps preserve the balance of monomials in the final certificate.

Consider the Koester graph [9] from Figure 1, a graph with 40 vertices and 80 edges. This graph has chromatic number four, and a corresponding non-3-colorability certificate of degree four. The size (after preprocessing) of the associated linear system required by NullA to produce this certificate was $8,724,468 \times 10,995,831$ and required 5 hours and 17 minutes of computation time.

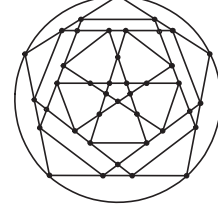


Figure 1: Koester graph

When we inspect the Koester graph in Figure 1, we can see that this graph contains 25 triangles. When we append these additional 25 equations to the system of polynomial equations describing this graph, the degree of the Nullstellensatz certificate drops from four to one, and now, with the addition of the 25 triangle equations, NullA only needs to solve a $4,626 \times 4,346$ linear system to produce a degree one certificate, which takes 0.2 seconds of computation time. Note that even though we have *appended* equations to the system of polynomial equations, because the degree of the overall certificate is *reduced*, the size of the resulting linear system is still much, much less.

These degree-cutter equations for 3-colorability (1) can be extended to k -colorability. A $(k-1)$ -clique implies that all nodes in the clique have a different color. Then, given the $(k-1)$ -clique with the vertices x_1 through x_{k-1} , the equation $x_1^{k-1} + x_2^{k-1} + \dots + x_{k-1}^{k-1} = 0$ is valid. We conjecture that these equations may also decrease the minimal degree of the Nullstellensatz certificate if one exists.

The degree-cutter equations for 3-colorability (1) are not always sufficient to reduce the degree of the Nullstellensatz. Consider the graph from Figure 2. Using only the polynomials from Lemma 3.1, the graph in Figure 2 has a degree four certificate. The graph contains three triangles: $\{1, 2, 6\}$, $\{2, 5, 6\}$ and $\{2, 6, 7\}$. In this case, after appending the degree-cutter equations for 3-colorability (1) the degree of the minimal Nullstellensatz certificate for this graph is still four.

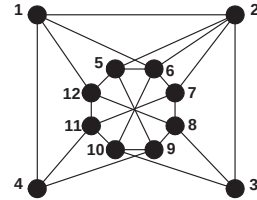


Figure 2: A graph with a degree four certificate.

The difficulty with the degree-cutter approach is in finding candidate degree-cutters and in determining how many of the candidate degree-cutters to append to the system. There is an obvious trade-off here between the time spent finding

degree-cutters together with the time penalty incurred related to the increased size of the linear system that must be solved versus the benefit of reducing the degree of the Nullstellensatz certificate.

3.4 Alternative Nullstellensätze

There is another approach we have found to decrease the minimal degree of the Nullstellensatz certificate. We now introduce the idea of an *alternative Nullstellensatz*, which follows from the Hilbert Nullstellensatz.

COROLLARY 3.7 (ALTERNATIVE NULLSTELLENSATZ). *A system of polynomial equations $f_1(x) = 0, \dots, f_s(x) = 0$ where $f_i \in \mathbb{K}[x_1, \dots, x_n]$ and $\mathbb{K}$ is an algebraically closed field has no solution in $\mathbb{K}^n$ if and only if there exist polynomials $\beta_1, \dots, \beta_s \in \mathbb{K}[x_1, \dots, x_n]$ and $g \in \mathbb{K}[x_1, \dots, x_n]$ such that $g = \sum \beta_i f_i$ and the system $f_1(x) = 0, \dots, f_s(x) = 0$ and $g(x) = 0$ has no solution.*

The Hilbert Nullstellensatz is a special case of this alternative Nullstellensatz where $g(x) = 1$. We can easily adapt the NulLA algorithm to use this alternative Nullstellensatz given the polynomial g . Here, the polynomial g determines the constant terms of the linear system that we need to solve to find a certificate of infeasibility. The idea here is that the minimal degree of the alternative Nullstellensatz certificate is sometimes smaller than the minimal degree of the ordinary Nullstellensatz certificate.

In the case of 3-colorability (and also more generally k -colorability), we may choose g as any non-trivial monomial since $g(x) = 0$ implies that $x_i = 0$ for some $i = 1, \dots, n$, which contradicts that $x_i^3 - 1 = 0$. For the graph in Figure 2, if we choose $g(x) = x_1 x_8 x_9$ (among others), then the minimal degree of the Nullstellensatz certificate drops to one.

The apparent difficulty in using the alternative Nullstellensatz approach is in choosing $g(x)$. One solution to this problem is to try and find a Nullstellensatz certificate for a set of $g(x)$ including $g(x) = 1$. For example, for the graph in Figure 2, we tried to find a certificate of degree one for the set of all possible monomials of degree 3. Since choosing different $g(x)$ only means changing the constant terms of the linear system in NulLA (the other coefficients remain the same), solving for a set of $g(x)$ can be accomplished very efficiently.

4. EXPERIMENTAL RESULTS

In this section, we present our experimental results, including a comparison between NulLA and other graph coloring algorithms such as the Alon-Tarsi method [1], the Gröbner basis method, DSATUR and *Branch-and-Cut* [12]. To summarize, almost all of the graphs tested by NulLA had degree one certificates. This algebraic property, coupled with our ability to compute over $\mathbb{F}_2$, allowed us to prove the non-3-colorability of graphs with over a thousand nodes.

4.1 Methods

Our computations were performed on machines with dual Opteron nodes, 2 GHz clock speed, and 12 GB of RAM. No degree-cutter equations or alternative Nullstellensatz certificates were used. We preprocessed the linear systems by removing redundant vertex polynomials via $(x_i^3 + 1) = (x_j^3 + 1) + (x_i + x_j)(x_i^2 + x_i x_j + x_j^2)$. Since the graphs that we tested are connected, via the previous equality, we can remove all but one of the vertex polynomial equations by

tracing paths from an arbitrarily selected “origin” vertex. We also eliminated unnecessary monomials from the system.

4.2 Test cases

We tested the following graphs:

1. **DIMACS:** The graphs from the DIMACS Computational Challenge (1993, 2002) are described in detail at <http://mat.gsia.cmu.edu/COLORING02/>. This set of graphs is the standard benchmark for graph coloring algorithms. We tested every DIMACS graph whose associated NulLA matrix could be instantiated within 12 GB of RAM. For example, we did *not* test C4000.5.c1q, which has 4,000 vertices and 4,000,268 edges, yielding a degree one NulLA matrix of 758 million non-zero entries and 1 trillion columns.
2. **Mycielski:** The Mycielski graphs are known for the gap between their clique and chromatic number. The Mycielski graph of order k is a triangle-free graph with chromatic number k . The first few instances and the algorithm for their construction can be seen at <http://mathworld.wolfram.com/MycielskiGraph.html>.
3. **Kneser:** The nodes of the Kneser- (t, r) graph are represented by the r -subsets of $\{1, \dots, t\}$. Two nodes are adjacent iff their subsets are disjoint.
4. **Random:** We tested random graphs in 16 nodes with an edge probability of .27. This probability was experimentally selected based on the boundary between 3-colorable and non-3-colorable graphs and is explained in detail in Section 4.3.

4.3 Results

In this section, we present our experimental results on graphs with and without 4-cliques. We also point out certain properties of NulLA-constructed certificates, and conclude with tests on random graphs. Surprisingly, all but four of the DIMACS, Mycielski and Kneser graphs tested with NulLA have degree one certificates.

The DIMACS graphs are primarily benchmarks for graph k -colorability, and thus contain many graphs with large chromatic number. Such graphs often contain 4-cliques. Although testing for graph 3-colorability is well-known to be NP-Complete, there exist many efficient (and even trivial), polynomial-time algorithms for finding 4-cliques in a graph. Thus, we break our computational investigations into two tables: Table 1 contains graphs *without* 4-cliques, and Table 3 contains graphs *with* 4-cliques (considered “easy” instances of 3-colorability). In the tables below, the certificate degree is one, n is the number of vertices and m is the number of edges. For space considerations, we only display representative results for graphs of varying size for each family. The size of the linear systems involved range from $15,737 \times 15,681$ up to $45,980,650 \times 46,378,333$ (for the $(8, 3)$ -Kneser and $(13, 5)$ -Kneser graphs, respectively).

However, not all of the DIMACS challenge graphs had degree one certificates. We were not able to produce certificates for `mug88_1`, `mug88_25`, `mug100_1` or `mug100_25`, even when using degree-cutters and searching for alternative Nullstellensatz certificates. When testing for a degree four certificate, the smallest of these graphs (88 vertices and 146 edges) yielded a linear system with 1,170,902,966 non-zero

Graph	n	m	sec
Mycielski 7	95	755	.46
Mycielski 9	383	7,271	268.78
Mycielski 10	767	22,196	14835
(8, 3)-Kneser	56	280	.07
(10, 4)-Kneser	210	1,575	3.92
(12, 5)-Kneser	792	8,316	466.47
(13, 5)-Kneser	1,287	36,036	216105
ash331GPIA	662	4,185	13.71
ash608GPIA	1,216	7,844	34.65
ash958GPIA	1,916	12,506	90.41
1-Insertions_5	202	1,227	1.69
2-Insertions_5	597	3,936	18.23
3-Insertions_5	1,406	9,695	83.45

Table 1: Graphs without 4-cliques.

entries and 390,340,149 columns. A matrix of this size is not computationally tractable at this time.

Recall that the certificates returned by NullA consist of a single vertex polynomial (via preprocessing), and edge polynomials describing either the original graph in its entirety, or a non-3-colorable subgraph from the original graph. For example, if the graph contains a 4-clique as a subgraph, often the Nullstellensatz certificate will only display the edges contained in the 4-clique. In this case, we say that NullA *isolates* a non-3-colorable subgraph from the original graph. The size difference between these subgraphs and the input graphs is often dramatic, as shown in Table 2.

Graph	n	m	subgraph vertices	subgraph edges
miles1500	128	10,396	6	10
hamming8-4	256	20,864	19	33
Mycielski 10	767	22,196	11	20
(12, 5)-Kneser	792	8,316	53	102
dsjc1000.1	1,000	49,629	15	24
ash608GPIA	1,216	7,844	23	44
3-Insertions_5	1,406	9,695	56	110
ash958GPIA	1,916	12,506	24	45

Table 2: Input G vs. non-3-colorable subgraph.

An overall analysis of these computational experiments shows that NullA performs best on sparse graphs. For example, the 3-Insertions_5 graph (with 1,406 nodes and 9,695 edges) runs in 83 seconds, while the 3-FullIns_5 graph (with 2,030 nodes and 33,751 edges) runs in 15027 seconds. Another example is p_hat700-2 (with 700 nodes and 121,728 edges) and will199GPIA (with 701 nodes and 7,065 edges). NullA proved the non-3-colorability of will199GPIA in 35 seconds, while p_hat700-2 took 30115 seconds.

Finally, as an informal measure of the distribution of degree one certificates, we generated random graphs of 16 nodes with edge probability .27. We selected this probability because it lies on the boundary between feasible and infeasible instances. In other words, graphs with edge probability less than .27 were almost always 3-colorable, and graphs with edge probability greater than .27 were almost always non-3-colorable. However, we experimentally found that an edge probability of .27 created a distribution that was almost exactly half and half. Of 100 trials, 48 were infeasible. Of those 48 graphs, 40 had degree one certificates and 8 had degree four certificates. Of these remaining 8 instances, we were able to find degree one certificates for all 8 by append-

ing degree-cutters or by finding alternative Nullstellensatz certificates. This tentative measure indicates that non-3-colorability certificates of degrees greater than one may be rare.

Graph	n	m	sec
miles500	128	2,340	1.35
miles1000	128	6,432	7.52
miles1500	128	10,396	24.23
multsol.i.5	197	3,925	6
zeroin.i.1	211	4,100	6
queen16_16	256	12,640	106
hamming8-4	256	20,864	621.1
school1_nsh	352	14,612	210.74
MANN_a27	378	70,551	9809.22
brock400_4	400	59,765	4548.59
gen400_p0.9_65	400	71,820	9608.85
le450_5d	450	9,757	304.84
fpsol2.i.1	496	11,654	93.8
C500.9	500	112,332	72752
homer	561	3,258	8
p_hat700-2	700	121,728	30115
will199GPIA	701	7,065	35
initx.i.1	864	18,707	1021.76
qg.order30	900	26,100	13043
wap06a	947	43,571	1428
dsjc1000.1	1,000	49,629	2981.91
5-FullIns_4	1,085	11,395	200.09
3-FullIns_5	2,030	33,751	15027.9

Table 3: Graphs with 4-cliques.

4.4 NullA vs. other algorithms

In this section, we compare NullA to two other algebraic methods for detecting 3-colorability; the Alon-Tarsi (AT) method, and the Gröbner basis (GB) method. We also briefly comment on NullA’s relation to well-known graph coloring heuristics such as DSATUR and Branch-and-Cut [12]. We implemented the Alon-Tarsi method in C++, and used CoCoA Lib [4] to test the Gröbner basis method. For brevity, we do not record any “internal data” about the various algorithmic runs, such as the size of the underlying linear systems solved by NullA or the maximum number of monomials in the normal forms produced by the Alon-Tarsi method. In the tables below, all certificates have degree one, n refers to the number of vertices, m refers to the number of edges and a “-” signifies that the method was terminated after 4 hours of computation.

The Gröbner basis method refers to simply taking the Gröbner basis of the ideal defined in Lemma 3.1. By Hilbert’s Nullstellensatz, the Gröbner basis is a constant if and only if the graph is non-3-colorable.

The Alon-Tarsi method is based on the following (see Section 7 of [1] and references therein):

THEOREM 4.1. *Given a graph G with n vertices, let $I_G = \langle x_1^3 - 1, \dots, x_n^3 - 1 \rangle$. Additionally, let*

$$P_G = \prod_{(i,j) \in E(G)} (x_i - x_j)$$

Then $P_G \in I_G$ if and only if G is non-3-colorable

In order to compute with the Alon-Tarsi method, we note that the set $B = \{x_1^3 - 1, \dots, x_n^3 - 1\}$ is a Gröbner basis for I_G . Thus, we simply take the normal form of P_G with

respect to B . If the normal form is zero, $P_G \in I_G$, and the graph is non-3-colorable. The efficiency of the Alon-Tarsi method can be increased by incrementally constructing P_G [7]: we order the edges, and then find the normal form of $(x_{i_1} - x_{j_1})$ with respect to B , and then the normal form of $(x_{i_1} - x_{j_1})(x_{i_2} - x_{j_2})$ with respect to B , etc.

We compared NullA to the Gröbner basis and Alon-Tarsi methods on graphs with and without 4-cliques; results are displayed in Tables 6 and 7, respectively.

NullA consistently greatly out-performed the Gröbner basis method. For example, on `zeroin.i.1`, NullA ran in 6 seconds, while CoCoA Lib took almost one hour. These experimental results indicate that NullA scales better than the Gröbner basis method.

NullA also compared extremely favorably with the Alon-Tarsi method, which usually did not terminate within the requisite time bounds. However, in the special case when the first few vertices and edges of the graph happen to describe a non-3-colorable subgraph (such as a 4-clique, or the Grötzsch graph), the Alon-Tarsi method ran very quickly, because of the iterative approach incorporated during implementation. Consider the example of the ninth Mycielski graph (383 vertices and 7,271 edges): the Alon-Tarsi method terminated in .24 seconds, but after we permuted the vertices and edges, the method consumed 9 GB of RAM over 4 hours of computation and only processed 30 edges. This example shows that the Alon-Tarsi method is extremely sensitive to the vertex and edge ordering. If a similar iterative approach was incorporated either into NullA or the Gröbner basis method, these algorithms would likewise terminate early in this special case.

As another example of the draw-backs of the Alon-Tarsi method, we considered edge-critical graphs, where the entire input must be read. For example, the odd wheels form a trivial family of edge-critical non-3-colorable graphs. The Alon-Tarsi method was unable to determine the non-3-colorability of the 17-odd-wheel (18 vertices and 34 edges): after two hours of computation, the normal form contained over 19 million monomials, and had consumed over 8 GB of RAM. The experimental results are displayed in Table 4.

<i>Wheels</i>	<i>n</i>	<i>m</i>	<i>NullA</i>	<i>GB</i>	<i>AT</i>
9	10	18	0	0	.05
11	12	22	0	0	.74
13	14	26	0	0	8.47
15	16	30	0	0	369.45
17	18	34	0	0	—
151	152	302	.21	2.21	—
501	502	1,002	15.58	126.83	—
1001	1,002	2,002	622.73	1706.69	—
2001	2,002	4,002	12905.6	—	—

Table 4: NullA, GB and AT on wheel graphs.

We conclude with a short comment about NullA’s relation to DSATUR and Branch-and-Cut [12]. These heuristics return bounds on the chromatic number. In Table 5 (data taken from [12]), we display the bounds returned by Branch-and-Cut (B&C) and DSATUR, respectively. In the case of these graphs, NullA determined non-3-colorability very rapidly (establishing a lower bound of four), while the two heuristics returned lower bounds of three and two, respectively. Thus, NullA returned a tighter lower bound on the chromatic number than B&C or DSATUR.

			B&C		DSATUR		NullA
<i>Graph</i>	<i>n</i>	<i>m</i>	<i>lb</i>	<i>up</i>	<i>lb</i>	<i>up</i>	<i>sec</i>
4-Insertions.3	79	156	3	4	2	4	0
3-Insertions.4	281	1,046	3	5	2	5	1
4-Insertions.4	475	1,795	3	5	2	5	3
2-Insertions.5	597	3,936	3	6	2	6	12
3-Insertions.5	1,406	9,695	3	6	2	6	83

Table 5: NullA vs. Branch-and-Cut and DSATUR.

<i>Graph</i>	<i>n</i>	<i>m</i>	<i>NullA</i>	<i>GB</i>	<i>AT</i>
miles500	128	2,340	1.35	133.91	.07
miles1000	128	6,432	7.52	802.23	0
miles1500	128	10,396	24.23	2598.84	.01
mulsol.i.5	197	3,925	6	18804.5	0
zeroin.i.1	211	4,100	6	2753.37	0
queen16_16	256	12,640	106	59466.9	0
hamming8-4	256	20,864	621.1	—	—
le450_5d	450	9,757	304.84	—	—
homer	561	3,258	8	—	—
dsgc1000.1	1,000	49,629	2981.91	—	—
5-FullIns.4	1,085	11,395	200.09	—	557.12
3-FullIns.5	2,030	33,751	15027.9	—	3.97

Table 6: NullA, GB, AT on graphs with 4-cliques.

<i>Graph</i>	<i>n</i>	<i>m</i>	<i>NullA</i>	<i>GB</i>	<i>AT</i>
Mycielski 4	11	20	0	.01	.22
Mycielski 5	23	71	0	.08	.23
Mycielski 6	47	236	.04	3.99	.22
Mycielski 7	95	755	.46	179.94	.23
Mycielski 8	191	2,360	7.72	9015.06	.23
Mycielski 9	383	7,271	268.78	—	.22
Mycielski 9 permuted	383	7,271	497.47	—	—
(6, 2)-Kneser	15	45	0	.03	1.87
(8, 3)-Kneser	56	280	.07	18.39	—
(10, 4)-Kneser	210	1,575	3.92	9771.76	—
(12, 5)-Kneser	792	8,316	466.47	—	—
ash331GPIA	662	4,185	13.71	—	—
1-Insertions.4	67	232	.04	3.71	—
2-Insertions.4	149	541	.26	32.42	—
1-Insertions.5	202	1,227	1.69	940.7	—
3-Insertions.4	281	1,046	.97	237.69	—
4-Insertions.4	475	1,795	3.02	1596.35	—
2-Insertions.5	597	3,936	18.23	—	—

Table 7: NullA, GB, AT on graphs w/o 4-cliques.

5. CONCLUSION

We presented a general algebraic method to prove combinatorial infeasibility. We showed that even though the worst-case known Nullstellensatz degree upper bounds are doubly-exponential, in practice for useful combinatorial systems, they are often much smaller and can be used to solve even large problem instances. Our experimental results illustrated that many benchmark non-3-colorable graphs have degree one certificates; indeed, non-3-colorable graphs with certificate degrees larger than one appear to be rare. We also showed that NullA compares well with other algebraic methods and popular heuristics for colorability.

6. REFERENCES

- [1] N. Alon, Combinatorial Nullstellensatz, *Combinatorics, Probability and Computing*, 8, (1999), 7–29.

- [2] D.A. Bayer, “The Division Algorithm and the Hilbert Scheme,” Ph.D. Thesis, Harvard University, (1982).
- [3] S. Buss and T. Pitassi, Good degree bounds on nullstellensatz refutations of the induction principle, *IEEE Conference on Computational Complexity*, (1996), 233–242.
- [4] CoCoATeam, CoCoA: a system for doing Computations in Commutative Algebra, available at <http://cocoa.dima.unige.it>.
- [5] D. Cox, J. Little and D. O’Shea, “Ideals, Varieties and Algorithms,” Springer Undergraduate Texts in Mathematics, Springer-Verlag, New York, (1992).
- [6] J.A. De Loera, J. Lee, S. Margulies, S. Onn, Expressing Combinatorial Optimization Problems by Systems of Polynomial Equations and the Nullstellensatz, submitted manuscript 2007, see <http://arxiv.org/abs/0706.0578>
- [7] C.J. Hillar and T. Windfeldt, “An algebraic characterization of uniquely vertex colorable graphs,” *Journal of Combinatorial Theory Series B*, 98 (2008), 400–414.
- [8] P. Impagliazzo, P. Pudlák and J. Sgall, Lower bounds for polynomial calculus and the Groebner basis algorithm, *Computational Complexity*, 8, (1999), 127–144.
- [9] E. Koester, On 4-critical planar graphs with high edge density, *Discrete Mathematics*, 98, (1991), 147–151.
- [10] J. Kollar, Sharp effective Nullstellensatz, *Journal of the AMS*, 1(4), (1988), 963–975.
- [11] J.B. Lasserre, Polynomials nonnegative on a grid and discrete optimization, *Transactions of the AMS*, 354(2), (2001), 631–649.
- [12] I. Méndez-Díaz and P. Zabala, “A branch-and-cut algorithm for graph coloring,” *Discrete Applied Mathematics*, Vol. 154(5), (2006), 826–847.
- [13] M. Laurent, Semidefinite representations for finite varieties, *Mathematical Programming*, 109, (2007), 1–26.
- [14] L. Lovász, Stable sets and polynomials, *Discrete Mathematics*, 124, (1994), 137–153.
- [15] S. Margulies, “Computer Algebra, Combinatorics, and Complexity: Hilbert’s Nullstellensatz and NP-Complete Problems,” UC Davis Ph.D. dissertation, in preparation, (2008).
- [16] P. Parrilo, Semidefinite programming relaxations for semialgebraic problems,” *Mathematical Programming, Series B*, 96(2), (2003), 293–320.

When does $\langle T \rangle$ equal $\text{sat}(T)$?

François Lemaire

Laboratoire d'Informatique Fondamentale de
Lille (LIFL)
Université de Lille, France
Francois.Lemaire@lifl.fr

Wei Pan

ORCCA, University of Western Ontario (UWO)
London, Ontario, Canada
wpan9@csd.uwo.ca

Marc Moreno Maza

ORCCA, University of Western Ontario (UWO)
London, Ontario, Canada
moreno@csd.uwo.ca

Yuzhen Xie

ORCCA, University of Western Ontario (UWO)
London, Ontario, Canada
yxie@csd.uwo.ca

ABSTRACT

Given a regular chain T , we aim at finding an efficient way for computing a system of generators of $\text{sat}(T)$, the saturated ideal of T . A natural idea is to test whether the equality $\langle T \rangle = \text{sat}(T)$ holds, that is, whether T generates its saturated ideal. By generalizing the notion of primitivity from univariate polynomials to regular chains, we establish a necessary and sufficient condition, together with a Gröbner basis free algorithm, for testing this equality. Our experimental results illustrate the efficiency of this approach in practice.

Categories and Subject Descriptors:

I.1.2 [Computing Methodologies]: Symbolic and Algebraic Manipulation – *Algebraic Algorithms*

General Terms:

Algorithms, Theory

Keywords:

Regular chain, Saturated ideal, Primitivity of polynomials.

1. INTRODUCTION

Triangular decompositions are one of the most studied techniques for solving polynomial systems symbolically. Invented by J.F. Ritt in the early 30's for systems of differential polynomials, their stride started in the late 80's with the method of W.T. Wu [21] dedicated to algebraic systems. Different concepts and algorithms extended the work of Wu. In the early 90's, the notion of a *regular chain*, introduced independently by M. Kalkbrener in [13] and, by L. Yang and J. Zhang [22], led to important algorithmic discoveries.

In Kalkbrener's vision, regular chains are used to represent the generic zeros of the irreducible components of an algebraic variety. In the original work of Yang and Zhang, they are used to decide whether a hypersurface intersects a quasi-variety (given by a regular chain). Regular chains have, in fact, several interesting properties and are the key notion in many algorithms for decomposing systems of algebraic or differential equations.

Regular chains have been investigated in many papers, among

them are [2, 14, 7]. Several surveys [4, 11] are also available on this topic. The abundant literature on the subject can be explained by the many equivalent definitions of a regular chain. Actually, the original formulation of Kalkbrener is quite different from that of Yang and Zhang. Two papers [5, 20] provide bridges between the point of view of Kalkbrener and that of Yang and Zhang.

The key algebraic object associated with a regular chain is its *saturated ideal*. Let us review its definition. Let $\mathbf{k}$ be a field and $x_1 \prec \dots \prec x_n$ be ordered variables. For a regular chain $T \subset \mathbf{k}[x_1, \dots, x_n]$, the saturated ideal of T , denoted by $\text{sat}(T)$ is defined by $\text{sat}(T) := \langle T \rangle : h^\infty$, where h is the product of the initial polynomials of T . (The next section contains a detailed review of these notions.) Given a polynomial $p \in \mathbf{k}[x_1, \dots, x_n]$, the memberships $p \in \text{sat}(T)$ and $p \in \sqrt{\text{sat}(T)}$ can be decided by means of pseudo-divisions and GCD computations, respectively. One should observe that these computations can be achieved without computing a system of generators of $\text{sat}(T)$. In some sense, the regular chain T is a “black box representation” of $\text{sat}(T)$ since the assertions $p \in \text{sat}(T)$ and $p \in \sqrt{\text{sat}(T)}$ can be evaluated without using an explicit representation of $\text{sat}(T)$.

Being able to compute a system of generators of $\text{sat}(T)$ remains, however, a fundamental question. For instance, given a second regular chain $U \subset \mathbf{k}[x_1, \dots, x_n]$, the only general method to decide the inclusion $\text{sat}(T) \subseteq \text{sat}(U)$ goes through the computation of a system of generators of $\text{sat}(T)$ by means of Gröbner bases. Unfortunately, such computations can be expensive (see [3]) whereas one would like to obtain an inclusion test which could be used intensively in order to remove redundant components when computing the triangular decompositions of Kalkbrener's algorithm or those arising in differential algebra. Note that for other kinds of triangular decompositions, such as those of [17, 20], this question has been solved in [6].

Therefore, testing the inclusion $\text{sat}(T) \subseteq \text{sat}(U)$ without Gröbner basis computation is a very important question in practice. Moreover, this can be regarded as an *algebraic version* of the Ritt problem in differential algebra. One case presents no difficulties: if $\text{sat}(T)$ is a zero-dimensional ideal, the product of the initial polynomials of T is invertible modulo $\langle T \rangle$ (see Proposition 5 in [18]) and thus T generates $\text{sat}(T)$. In this case the inclusion test for saturated ideals reduces to the membership problem mentioned above.

In positive dimension, however, the ideal $\text{sat}(T)$ could be strictly larger than that generated by T . Consider for instance $n = 4$ and $T = \{x_1x_3 + x_2, x_2x_4 + x_1\}$, we have

$$\langle T \rangle = \langle x_1, x_2 \rangle \cap \langle x_1x_3 + x_2, -x_3x_4 + 1 \rangle.$$

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Thus, we have

$$\text{sat}(T) = \langle T \rangle : (x_1x_2)^\infty = \langle x_1x_3 + x_2, -x_3x_4 + 1 \rangle.$$

In this article, we give a necessary and sufficient condition for the equality $\langle T \rangle = \text{sat}(T)$ to hold. Looking at the above example, one can feel that the ideal $\langle x_1, x_2 \rangle$ can be regarded as a “sort of content” of the ideal $\langle T \rangle$, which is discarded when computing $\text{sat}(T)$. We observe also that the polynomials $x_1x_3 + x_2$ and $x_2x_4 + x_1$ are primitive in $(\mathbf{k}[x_1, x_2])[x_3]$ and $(\mathbf{k}[x_1, x_2])[x_4]$ respectively. Thus, the “usual notion” of primitivity (for a univariate polynomial over a UFD) is not sufficient to guarantee the equality $\langle T \rangle = \text{sat}(T)$. This leads us to the following two definitions.

Let R be a commutative ring with unity. We say that a non-constant polynomial $p = a_e x^e + \dots + a_0 \in R[x]$ is *weakly primitive* if for any $\beta \in R$ such that a_e divides $\beta a_{e-1}, \dots, \beta a_0$ then a_e divides β as well. This notion and its relations with similar concepts are discussed in Sections 3, 4, and 5.

We say that the regular chain $T = \{p_1, \dots, p_m\}$ is *primitive* if for all $1 \leq k \leq m$, the polynomial p_k is weakly primitive in $R[x_j]$, where x_j is the main variable of p_k and R is the residue class ring $\mathbf{k}[x_1, \dots, x_{j-1}]/\langle p_1, \dots, p_{k-1} \rangle$.

The first main result of this paper is the following: *the regular chain T generates its saturated ideal if and only if T is primitive.* This result, generalizing the concept of primitivity from univariate polynomials to regular chains, is established in Section 4.

In Section 6, looking at regular chains from the point of view of regular sequences, we obtain our second main result: an algorithm to decide whether a regular chain generates its saturated ideal or not. The pseudo-code and its proof are presented in Section 6. This algorithm relies on a procedure for computing triangular decompositions. However, being applied to input systems which are regular sequences and “almost regular chains”, this procedure reduces simply to an iterated resultant computation. As a result, the proposed algorithm performs very well in practice and is Gröbner basis free.

In Section 7 we report on experimentation, where we confirm the efficiency of the algorithm. Meanwhile, we observe that primitive regular chains are often present in the output of triangular decompositions. The paper is concluded with a few remarks.

2. PRELIMINARIES

2.1 Triangular set and regular chain

We denote by $\mathbf{k}[\mathbf{x}]$ the ring of multivariate polynomials with coefficients in a field $\mathbf{k}$ and with ordered variables $\mathbf{x} = x_1 \prec \dots \prec x_n$. For a non-constant polynomial $p \in \mathbf{k}[\mathbf{x}]$, the greatest variable appearing in p is called *main variable*, denoted by $\text{mvar}(p)$. We regard p as a univariate polynomial in its main variable. The degree, the leading coefficient, the leading monomial and the reductum of p as a univariate polynomial in $\text{mvar}(p)$ are called *main degree*, *initial*, *rank* and *tail* of p ; they are denoted by $\text{mdeg}(p)$, $\text{init}(p)$, $\text{rank}(p)$ and $\text{tail}(p)$ respectively. Thus we have $p = \text{init}(p)\text{rank}(p) + \text{tail}(p)$.

Let R be a commutative ring with unity and F be a subset of R . Denote by $\langle F \rangle$ the ideal it generates, by $\sqrt{\langle F \rangle}$ the radical of $\langle F \rangle$, and by $R/\langle F \rangle$ the residue class ring of R with respect to $\langle F \rangle$. For an element p in R , we say that p is zero modulo $\langle F \rangle$ if p belongs to $\langle F \rangle$, that is, p is zero as an element in $R/\langle F \rangle$. An element $p \in R$ is a *zerodivisor* modulo $\langle F \rangle$, if there exists $q \in R$ such that $p \notin \langle F \rangle$ and $q \notin \langle F \rangle$ but $pq \in \langle F \rangle$. We say that p is *regular* modulo $\langle F \rangle$ if it is neither zero, nor a zerodivisor modulo $\langle F \rangle$. Furthermore, p is *invertible* in R if there exists a $q \in R$ such that $pq = 1$.

Example 2.1 Consider the polynomials in $\mathbf{k}[x_1, x_2, x_3]$

$$p_1 = x_2^2 - x_1^2, p_2 = (x_2 - x_1)x_3 \text{ and } p_3 = x_2x_3^3 - x_1.$$

The above notions are illustrated in the following table.

	mvar	init	mdeg	rank	tail
p_1	x_2	1	2	x_2^2	$-x_1^2$
p_2	x_3	$x_2 - x_1$	1	x_3	0
p_3	x_3	x_2	3	x_3^3	$-x_1$

The initial $x_2 - x_1$ of p_2 is a zerodivisor modulo $\langle p_1 \rangle$, since $(x_2 + x_1)(x_2 - x_1)$ is in $\langle p_1 \rangle$, while neither $x_2 + x_1$ nor $x_2 - x_1$ belongs to $\langle p_1 \rangle$. However, the initial x_2 of p_3 is regular modulo $\langle p_1 \rangle$.

In what follows, we recall the notions of regular chain and saturated ideal, which are the main objects in our study.

A set T of non-constant polynomials in $\mathbf{k}[\mathbf{x}]$ is called a *triangular set*, if for all $p, q \in T$ with $p \neq q$ we have $\text{mvar}(p) \neq \text{mvar}(q)$. For a nonempty triangular set T , we define the *saturated ideal* $\text{sat}(T)$ of T to be the ideal $\langle T \rangle : h^\infty$, that is,

$$\text{sat}(T) := \langle T \rangle : h^\infty = \{q \in \mathbf{k}[\mathbf{x}] \mid \exists e \in \mathbb{Z}_{\geq 0} \text{ s.t. } h^e q \in \langle T \rangle\},$$

where h is the product of the initials of the polynomials in T . The empty set is also regarded as a triangular set, whose saturated ideal is the trivial ideal $\langle 0 \rangle$.

One way of solving (or decomposing) a polynomial set $F \subseteq \mathbf{k}[\mathbf{x}]$ is to compute triangular sets $T_1, \dots, T_e \subseteq \mathbf{k}[\mathbf{x}]$ such that $\sqrt{\langle F \rangle}$ equals the intersection of $\sqrt{\text{sat}(T_1)}, \dots, \sqrt{\text{sat}(T_e)}$. It is thus desirable to require that $\text{sat}(T_1), \dots, \text{sat}(T_e)$ be proper ideals. This observation has led to the notion of a regular chain which was introduced independently in [13] and [22].

Definition 2.2 (Regular chain) Let T be a triangular set in $\mathbf{k}[\mathbf{x}]$. If T is empty, then it is a regular chain. Otherwise, let p be the polynomial of T with the greatest main variable and let C be the set of other polynomials in T . We say that T is a regular chain, if C is a regular chain and $\text{init}(p)$ is regular modulo $\text{sat}(C)$.

In commutative algebra (See [10]) there is a closely related concept called *regular sequence* which is a sequence $r_1, \dots, r_s$ of nonzero elements in the ring $\mathbf{k}[\mathbf{x}]$ satisfying

- (i) $\langle r_1, \dots, r_s \rangle$ is a proper ideal of $\mathbf{k}[\mathbf{x}]$;
- (ii) r_i is regular modulo $\langle r_1, \dots, r_{i-1} \rangle$, for each $2 \leq i \leq s$.

When we sort polynomials in a regular chain by increasing main variable, the following example says that the resulting sequence may not be a regular sequence of $\mathbf{k}[\mathbf{x}]$.

Example 2.3 Let $T = \{t_1, t_2\}$ be a triangular set in $\mathbf{k}[x_1, x_2, x_3]$ with $t_1 = x_1x_2$ and $t_2 = x_1x_3$. Then $\{t_1\}$ is a regular chain with $\text{sat}(\{t_1\}) = \langle x_1x_2 \rangle : x_1^\infty = \langle x_2 \rangle$. Since $\text{init}(t_2) = x_1$ is regular modulo $\text{sat}(\{t_1\})$, the triangular set T is a regular chain with

$$\text{sat}(T) = \langle x_1x_2, x_1x_3 \rangle : x_1^\infty = \langle x_2, x_3 \rangle.$$

However, t_1, t_2 is not a regular sequence since $t_2 = x_1x_3$ is not regular modulo $\langle x_1x_2 \rangle$. Here, the saturation operation discards the content introduced by the initials.

2.2 Properties of regular chains

We recall several important results on regular chains and saturated ideals, which will be used throughout this paper. Pseudo-division and iterated resultant are fundamental tools in this context.

Let p and q be polynomials of $\mathbf{k}[x]$, with $q \notin \mathbf{k}$. Denote by $\text{prem}(p, q)$ and $\text{pquo}(p, q)$ the *pseudo-remainder* and the *pseudo-quotient* of p by q , regarding p and q as univariate polynomials in $x = \text{mvar}(q)$. Using these notations, we have

$$\text{init}(q)^e p = \text{pquo}(p, q)q + \text{prem}(p, q), \quad (1)$$

where $e = \max\{\deg(p, x) - \deg(q, x) + 1, 0\}$; moreover either $r := \text{prem}(p, q)$ is null or $\deg(r, x) < \deg(q, x)$. Pseudo-division generalizes as follows given a polynomial p and a regular chain T :

$$\text{prem}(p, T) = \begin{cases} p & \text{if } T = \emptyset, \\ \text{prem}(\text{prem}(p, t), T') & \text{if } T = T' \cup \{t\}, \end{cases}$$

where t is the polynomial in T with greatest main variable. We have the *pseudo-division formula* [21]: there exist non-negative integers $e_1, \dots, e_s$ and polynomials $q_1, \dots, q_s \in \mathbf{k}[x]$ such that

$$h_1^{e_1} \dots h_s^{e_s} p = \sum_{i=1}^s q_i t_i + \text{prem}(p, T), \quad (2)$$

where $T = \{t_1, \dots, t_s\}$ and $h_i = \text{init}(t_i)$, for $1 \leq i \leq s$.

We denote by $\text{res}(p, q)$ the *resultant* of p and q regarding them as univariate polynomials in $\text{mvar}(q)$. Note that $\text{res}(p, q)$ may be different from $\text{res}(q, p)$, if they have different main variables. For a polynomial p and a regular chain T , we define the *iterated resultant* of p w.r.t. T , denoted by $\text{ires}(p, T)$, as follows:

$$\text{ires}(p, T) = \begin{cases} p & \text{if } T = \emptyset, \\ \text{ires}(\text{res}(p, t), T') & \text{if } T = T' \cup \{t\}, \end{cases}$$

where t is the polynomial in T with greatest main variable.

Theorem 2.4 *For a regular chain T and a polynomial p we have:*

- (1) p belongs to $\text{sat}(T)$ if and only if $\text{prem}(p, T) = 0$,
- (2) p is regular modulo $\text{sat}(T)$ if and only if $\text{ires}(p, T) \neq 0$,
- (3) p is a zerodivisor modulo $\text{sat}(T)$ if and only if $\text{ires}(p, T) = 0$ and $\text{prem}(p, T) \neq 0$.

For the proofs, we refer to [2] for item (1), and to [20, 5] for item (2). Item (3) is a direct consequence of (1) and (2).

Remark 2.5 *Theorems 2.4 and 2.6 highlight the structure of the associated primes of $\text{sat}(T)$ which makes regularity test easier than with an arbitrary polynomial ideal. In general, deciding if a polynomial p is regular modulo an ideal I is equivalent to checking if p does not belong to any associated primes of I .*

An ideal in $\mathbf{k}[x]$ is *unmixed*, if all its associated primes have the same dimension. In particular, an unmixed ideal has no embedded associated primes.

Theorem 2.6 *Let $T = C \cup \{t\}$ be a regular chain in $\mathbf{k}[x]$ with t having greatest main variable in T . The following properties hold:*

- (1) $\text{sat}(T)$ is an unmixed ideal with dimension $n - |T|$,
- (2) $\text{sat}(T \cap \mathbf{k}[x_1, \dots, x_i]) = \text{sat}(T) \cap \mathbf{k}[x_1, \dots, x_i]$,
- (3) $\text{sat}(T) = \langle \text{sat}(C) \cup \{t\} \rangle : \text{init}(t)^\infty$.

For the proofs, we refer to [4, 7] for item (1), to [2] for item (2), and to [14] for item (3). From (1), we deduce that the saturated ideal of a regular chain T consisting of n polynomials has dimension 0.

3. PRIMITIVITY OF POLYNOMIALS

In this section, we introduce the notion of weak primitivity of a polynomial in a general univariate polynomial ring, and then present several of its properties.

The following Lemma 3.1 may be seen as a generalization of Gauss lemma over an arbitrary commutative ring with unity. It will be used in the proof of our main theorem. We found that this lemma is not new and can be deduced from the Dedekind-Mertens Lemma (See [1, 9, 8] and the references therein). For the sake of reference, we include our direct proof here. In the sequel, the ring R is a commutative Noetherian ring with unity. We say that p divides q , denoted by $p \mid q$, if there exists r such that $q = pr$ holds.

Lemma 3.1 *Let $p = \sum_{i=0}^m a_i y^i$ and $q = \sum_{i=0}^n b_i y^i$ be polynomials in $R[y]$ with $\deg(p) = m \geq 0$ and $\deg(q) = n \geq 0$. Then for each $h \in R$,*

- (i) $h \mid pq$ implies $h \mid b_0 a_i^{n+1}$ for $0 \leq i \leq m$,
- (ii) $h \mid pq$ implies $h \mid b_n a_i^{n+1}$ for $0 \leq i \leq m$.

PROOF. First, we prove (i). Considering first the special case $m = 0$, we observe that $h \mid pq$ implies $h \mid a_0 b_0$ and the conclusion follows. Now we assume that $m > 0$ holds.

For $i = 0$, the claim is also clear, for the same reason as the case $m = 0$. For $1 \leq i \leq m$, we introduce the polynomials A_i and B_i below in order to simplify our expressions:

$$A_i = \sum_{j=0}^{i-1} a_j y^j, \text{ and } B_i = - \sum_{j=i}^m a_j y^j. \quad (3)$$

Clearly, we have $p = A_i - B_i$. The key observation is to consider the polynomial $\tilde{p} = A_i^{n+1} - B_i^{n+1}$, as suggested by the forms of our claims. To avoid talking about the degree of a zero polynomial, we assume that both A_i^{n+1} and B_i^{n+1} are nonzero polynomials.

According to the construction of A_i and B_i in (3), we have the following degree estimates:

$$\deg(A_i^{n+1}) \leq \deg(A_i)(n+1) \leq (i-1)(n+1), \quad (4)$$

$$\text{trdeg}(B_i^{n+1}) \geq \text{trdeg}(B_i)(n+1) \geq i(n+1), \quad (5)$$

where $\text{trdeg}(\cdot)$ denotes the trailing degree, that is, the degree of the term with lowest degree in a polynomial. Therefore there is no term cancellation between A_i^{n+1} and B_i^{n+1} . With the assumption that A_i and B_i nonzero, the polynomial $\tilde{p}$ is nonzero too. Now we write $\tilde{p}$ in the form

$$\tilde{p} = (A_i - B_i)(A_i^n + \dots + B_i^n) = p(A_i^n + \dots + B_i^n).$$

It follows that $p \mid \tilde{p}$ holds. Therefore $h \mid \tilde{p}q$ holds since we have $h \mid pq$. Observe now that if qA_i^{n+1} is nonzero, then

$$\deg(qA_i^{n+1}) \leq (i-1)(n+1) + n < i(n+1). \quad (6)$$

Similarly, if qB_i^{n+1} is nonzero, then its trailing degree is bounded

$$\text{trdeg}(qB_i^{n+1}) \geq i(n+1). \quad (7)$$

Combining (6) with (7), we know that in $q\tilde{p} = qA_i^{n+1} - qB_i^{n+1}$, the polynomial qA_i^{n+1} only contributes to terms with degree smaller than $i(n+1)$. Thus we have

$$\text{coeff}(q\tilde{p}, y^{i(n+1)}) = \text{coeff}(-qB_i^{n+1}, y^{i(n+1)}) = b_0 a_i^{n+1} \quad (8)$$

which implies $h \mid b_0 a_i^{n+1}$, as desired.

Now we handle the special cases where $A_i^{n+1} = 0$ and $B_i^{n+1} = 0$. It is easy to see that $A_i^{n+1} = 0$ does not affect the proof above.

When $B_i^{n+1} = 0$, simply we have $a_i^{n+1} = 0$, and then the claim is also clear.

Finally, we prove (ii). Let $P = y^m p(1/y)$ and $Q = y^n q(1/y)$. Since $h \mid pq$, h will also divide $PQ = y^{m+n}(pq)(1/y)$. Assume that

$$\begin{aligned} a_0 &= \cdots = a_{r-1} = 0, a_r \neq 0, \\ b_0 &= \cdots = b_{s-1} = 0, b_s \neq 0. \end{aligned}$$

Then $r \leq m$ and $s \leq n$ hold. According to (i), for any $r \leq i \leq m$, $h \mid b_n a_i^{s+1}$. It follows that $h \mid b_n a_i^{n+1}$ for any $0 \leq i \leq m$. $\square$

Definition 3.2 Let $p = a_0 + \cdots + a_e x^e \in R[x]$ with $e \geq 1$. The polynomial p is strongly primitive if the ideal generated by $\{a_0, \dots, a_e\}$ is the whole ring R . The polynomial p is weakly primitive if for any $\beta \in R$ such that $a_e \mid \beta a_i$ holds for all $0 \leq i \leq e-1$, we have $a_e \mid \beta$ as well.

Proposition 3.3 Strong primitivity implies weak primitivity.

PROOF. We use the same notation as in Definition 3.2. Let p be strongly primitive. Then there exist $c_e, \dots, c_0 \in R$ such that $c_e a_e + \cdots + c_0 a_0 = 1$. Let $\beta \in R$ such that for $0 \leq j \leq e-1$, we have $a_e \mid \beta a_j$. Then there exist $d_0, \dots, d_{e-1} \in R$ such that $a_e d_j = \beta a_j$. Since $\beta c_e a_e + \cdots + \beta c_0 a_0 = \beta$, we have $a_e(\beta c_e + d_{e-1} c_{e-1} + \cdots + d_0 c_0) = \beta$. Thus, we have $a_e \mid \beta$, and therefore p is weakly primitive. $\square$

Remark 3.4

- (1) If any a_i is invertible, then p is strongly primitive and then is weakly primitive. As a particular case, p is weakly primitive if one of its coefficients is a nonzero constant of a field.
- (2) Weak primitivity does not imply strong primitivity. For example, let $R = \mathbb{Z}[t]$ and $p = tx + 2 \in \mathbb{Z}[t][x]$. Then p is not strongly primitive, since $\langle t, 2 \rangle \neq \langle 1 \rangle_R$. In $R[x]$, the polynomial p is weakly primitive. If $t \mid 2\beta$, then $t \mid \beta$ must hold.
- (3) The definition of strongly primitive does not depend on the order of the coefficients in p . However, the definition of weakly primitive relies on it. Indeed, let $R = \mathbb{Z}_4[t]$, $p = 2x + t$ and $q = tx + 2$. Then we have
 - (i) p is weakly primitive in $R[x]$. For any $\beta \in R[x]$, if $2 \mid t\beta$ then $2 \mid \beta$.
 - (ii) q is not weakly primitive in $R[x]$. Let $\beta = t + 2 \in R[x]$. Then we have $t \mid 2(t + 2) = 2t$, and $t \nmid (t + 2)$.
- (4) Weak primitivity may not be extended. That is to say, if p is weakly primitive, assuming that $\deg(p) = e > 0$, then $\bar{p} = p + qx^{e+1}$ may not be weakly primitive. For example, let $R = \mathbb{Z}_4[t]$, $p = 2x + t$ and $\bar{p} = p + tx^2 = tx^2 + 2x + t$. Then p is weakly primitive, and $\bar{p}$ is not weakly primitive. Indeed taking $\beta = t + 2$, we have $t \mid t\beta$ and $t \mid 2\beta$, but $t \nmid \beta$.

According to Proposition 3.5 the notion of weak primitivity turns out to be a generalization of the ordinary notion of primitivity (the gcd of the coefficients of a univariate polynomial is 1).

Proposition 3.5 Let R be a UFD and $p = \sum_{i=0}^e a_i x^i \in R[x]$ with $a_e \neq 0$ and $e \geq 1$. Then, the following statements are equivalent

- (i) p is weakly primitive in $R[x]$.

(ii) $\text{content}(p) := \gcd(a_0, \dots, a_e) = 1$.

PROOF. We prove (i) $\Rightarrow$ (ii). Assume that $\gcd(a_0, \dots, a_e) \neq 1$. Then there is a prime factor f of $\gcd(a_0, \dots, a_e)$. Let $\beta = a_e/f$. Then $a_e \mid \beta a_i$, for $0 \leq i \leq e-1$. Since $a_e \nmid \beta$, p is not weakly primitive, a contradiction.

We prove (ii) $\Rightarrow$ (i). Assume that there exists $\beta \in R$ such that

$$(\forall 0 \leq j \leq e-1) \quad a_e \mid \beta a_j \quad \text{and} \quad a_e \nmid \beta.$$

Then $a_e \mid \text{content}(\beta p) = \beta \text{content}(p)$. Since $a_e \nmid \beta$, some prime factor f of a_e divides $\text{content}(p)$, a contradiction. $\square$

The following property on weak primitivity will be used in the next section. It states the following fact: if one raises each coefficient of a weakly primitive polynomial p to some power, then the resulting polynomial is still weakly primitive. To avoid the cancellation of the leading coefficient of p , we assume that this coefficient is a regular element of the ground ring.

Proposition 3.6 Let $p = \sum_{i=0}^e a_i x^i \in R[x]$ with a_e being regular in R , and $\{n_i \mid 0 \leq i \leq e\}$ be a set of non-negative integers. Define $q = \sum_{i=0}^e a_i^{n_i} x^i$. Then if p is weakly primitive, q is also weakly primitive.

The proof directly follows from the following two lemmas.

Lemma 3.7 Let $p = a_0 + \cdots + a_e x^e \in R[x]$ with a_e being regular in R and n be a non-negative integer. If p is weakly primitive, then $p_n = a_0 + \cdots + a_{e-1} x^{e-1} + a_e^n x^e$ is also weakly primitive.

PROOF. By induction on $n \geq 0$. The case $n = 0$ follows from Remark 3.4. So we assume that the claim is true for $n-1$, that is, p_{n-1} is weakly primitive, with $n \geq 1$. Let $\beta \in R$ such that $a_e^n \mid a_i \beta$, for $0 \leq i \leq e-1$. There exist $h_0, \dots, h_{e-1} \in R$ such that we have

$$a_e^n h_i = a_i \beta, \quad 0 \leq i \leq e-1. \quad (9)$$

Since p_{n-1} is weakly primitive and since we have $a_e^{n-1} \mid a_i \beta$, we deduce $a_e^{n-1} \mid \beta$, that is, there exists $h' \in R$ such that

$$a_e^{n-1} h' = \beta. \quad (10)$$

With (9) and (10) we have $a_e^n h_i = a_i a_e^{n-1} h'$, and then $a_e h_i = a_i h'$, since a_e is regular. Hence $a_e \mid a_i h'$. By the weak primitivity of p , $a_e \mid h'$ holds, that is, there exists $h'' \in R$ such that

$$a_e h'' = h'. \quad (11)$$

By (10) and (11) we have $a_e^n h'' = \beta$. So $a_e^n \mid \beta$ and p_n is weakly primitive. $\square$

Lemma 3.8 Let $p = a_0 + \cdots + a_e x^e \in R[x]$ with $a_e \neq 0$ and n be a non-negative integer. Let j be an index such that $0 \leq j \leq e-1$. Define $q = a_0 + \cdots + a_j^n x^j + \cdots + a_e x^e = p + (a_j^n - a_j) x^j$. If p is weakly primitive, then q is also weakly primitive.

PROOF. The claim is clear if $n = 0$, so we assume $n \geq 1$. Let $\beta \in R$ such that, for $0 \leq i \leq e-1$ and $i \neq j$

$$a_e \mid a_i \beta, \quad \text{and} \quad a_e \mid a_j^n \beta. \quad (12)$$

We prove that $a_e \mid \beta$ holds. We have, for $0 \leq i \leq e-1$ and $i \neq j$

$$a_e \mid a_i (a_j^{n-1} \beta), \quad \text{and} \quad a_e \mid a_j (a_j^{n-1} \beta).$$

Define $\beta' = a_j^{n-1} \beta$. Hence $a_e \mid \beta'$ holds, since p is weakly primitive. With (12), for $0 \leq i \leq e-1$ and $i \neq j$ we have

$$a_e \mid a_i \beta, \quad \text{and} \quad a_e \mid a_j^{n-1} \beta. \quad (13)$$

We deduce that $a_e \mid a_j^{n-2} \beta$ holds. Continuing in this manner, we reach $a_e \mid \beta$. Thus q is also weakly primitive. $\square$

4. PRIMITIVE REGULAR CHAIN

In this section, we generalize the notion of primitivity to any regular chain T . Then we prove that $\text{sat}(T) = \langle T \rangle$ holds if and only if T is primitive.

Definition 4.1 Let $T = \{p_1, \dots, p_m\} \subset \mathbf{k}[x] = \mathbf{k}[x_1, \dots, x_n]$ be a regular chain with $\text{mvar}(p_1) \prec \dots \prec \text{mvar}(p_m)$. We say that T is primitive if for all $1 \leq k \leq m$, p_k is weakly primitive in $R[x_j]$ where $x_j = \text{mvar}(p_k)$ and

$$R = \mathbf{k}[x_1, \dots, x_{j-1}] / \langle p_1, \dots, p_{k-1} \rangle.$$

Proposition 4.2 (Base case of Theorem 4.4)

Let $p = a_e x^e + \dots + a_0 \in \mathbf{k}[y][x]$ and $c = \gcd_{\mathbf{k}[y]}(a_0, \dots, a_e)$, where $e \geq 1$ and y is a finite set of variables. Then we have $\langle p \rangle = \langle p \rangle : a_e^\infty \iff c = 1$.

PROOF. First we prove that $\langle p \rangle \subsetneq \text{sat}(p) := \langle p \rangle : a_e^\infty$ if $c \neq 1$. Denote $\bar{p} = p/c$. Then $a_e \bar{p} = a_e p/c \in \langle p \rangle$, hence $\bar{p} \in \text{sat}(p)$. Assume that $\bar{p}$ is in $\langle p \rangle$. Then there exists $q \in \mathbf{k}[y][x]$ such that $p/c = \bar{p} = pq$. It follows that $qc = 1$ which is a contradiction since $c \notin \mathbf{k}$. Therefore $\bar{p}$ is in $\text{sat}(p)$ but not in $\langle p \rangle$.

Conversely, we prove that if $c = 1$ then $\text{sat}(p) \subseteq \langle p \rangle$. For any $q \in \text{sat}(p)$, there exist $n \in \mathbb{Z}_{\geq 0}$ and $\beta \in \mathbf{k}[y][x]$ such that $a_e^n q = \beta p$. Taking the content w.r.t. x , we have

$$\begin{aligned} a_e^n \text{content}(q, x) &= \text{content}(\beta, x) \text{content}(p, x) \\ &= \text{content}(\beta, x) \end{aligned}$$

Thus $a_e^n \mid \beta$. There exists $\beta' \in \mathbf{k}[y][x]$ such that $\beta = a_e^n \beta'$. So we have $a_e^n q = \beta p = a_e^n \beta' p$, and then $q = \beta' p$, that is, $q \in \langle p \rangle$. $\square$

Remark 4.3 Let $T = \{p_1\}$ be a regular chain consisting of a single polynomial. By definition, T is primitive if and only if p_1 is weakly primitive in $R = \mathbf{k}[x_1, \dots, x_{j-1}]$, where $x_j = \text{mvar}(p_1)$. Since R is a UFD, it follows from Proposition 3.5, that T is primitive if and only if p_1 is primitive in ordinary sense, that is, whenever the gcd of the coefficients of p_1 (as a univariate polynomial in $R[x_j]$) is 1. Therefore, the notion of primitivity for a regular chain extends that of primitivity for a polynomial.

Theorem 4.4 Let $T \subset \mathbf{k}[x_1, \dots, x_n]$ be a regular chain. Then T is primitive if and only if $\langle T \rangle = \text{sat}(T)$.

PROOF. We prove the theorem by induction on the number of polynomials in T . The base case is Proposition 4.2, where $|T| = 1$. Now assume that $T = \{p_1, \dots, p_m\}$ consists of $m \geq 2$ polynomials with $\text{mvar}(p_1) \prec \dots \prec \text{mvar}(p_m)$. We denote by T_k the regular chain consisting of the first k polynomials in T .

First, assume indirectly that T is not primitive. We need to prove that $\langle T \rangle$ is a proper subset of $\text{sat}(T)$. Let k be the smallest integer such that p_k is not weakly primitive in $R[y]$, where $y = x_j = \text{mvar}(p_k)$ and $R = \mathbf{k}[x_1, \dots, x_{j-1}] / \langle T_{k-1} \rangle$. By Proposition 4.2, we know $k \geq 2$.

Let $p_k = a_e y^e + \dots + a_0$. By induction, $\text{sat}(T_{k-1}) = \langle T_{k-1} \rangle$ holds and thus a_e is regular in R . Since p_k is not weakly primitive over R , there exists $\beta \in \mathbf{k}[x_1, \dots, x_{j-1}]$ such that, in R , we have

$$(\forall 0 \leq r \leq e-1) \quad a_e \mid \beta a_r \quad \text{and} \quad a_e \nmid \beta.$$

Define $q_k = \beta p_k / a_e$. Then $q_k \in R[y]$, since

$$\frac{\beta}{a_e} p_k = \beta y^e + \sum_{0 \leq r < e} \frac{\beta a_r}{a_e} y^r.$$

We claim that $q_k \in \langle p_k \rangle : a_e^\infty$ and $q_k \notin \langle p_k \rangle$ in $R[y]$, which leads to $\text{sat}(T_k) \neq \langle T_k \rangle$.

Indeed, we have $a_e q_k = \beta p_k \in \langle p_k \rangle$ in $R[y]$. Thus, $q_k \in \langle p_k \rangle : a_e^\infty$. Now if $q_k \in \langle p_k \rangle$, there exists $\alpha \in R[y]$ such that $q_k = \alpha p_k$ in $R[y]$. By the construction of q_k , $\deg(q_k, y)$ equals $\deg(p_k, y)$. Hence $\alpha \in R$ and $\beta - \alpha a_e = 0$ in R . This contradicts $a_e \nmid \beta$.

Secondly, we assume that T is primitive and show $\langle T \rangle = \text{sat}(T)$. By induction, $\text{sat}(T_{k-1}) = \langle T_{k-1} \rangle$ holds. We shall prove that $\text{sat}(T_k) = \langle T_k \rangle$ holds, too. To do so, we consider $p \in \text{sat}(T_k)$ and show that we have $p \in \langle T_k \rangle$. Let $\text{mvar}(p) = x_i$ and $\text{mvar}(p_k) = x_j$. If $i > j$, then $p \in \text{sat}(T_k)$ if and only if all coefficients of p w.r.t. x_i are in $\text{sat}(T_k)$, since T_k is a regular chain. So we can concentrate on the case $p \in \mathbf{k}[x_1, \dots, x_j]$.

Let h_{p_k} be the leading coefficient of p_k w.r.t. $y = x_j$, that is, w.r.t. the main variable of p_k . By virtue of Theorem 2.6 we have

$$\begin{aligned} \text{sat}(T_k) &= \langle \text{sat}(T_{k-1}), p_k \rangle : h_{p_k}^\infty \\ &= \langle \langle T_{k-1} \rangle, p_k \rangle : h_{p_k}^\infty. \end{aligned}$$

By virtue of Theorem 2.4 we have $\text{prem}(p, T_k) = 0$, since $p \in \text{sat}(T_k)$. Consequently, $\text{prem}(p, p_k)$ is in $\text{sat}(T_{k-1}) = \langle T_{k-1} \rangle$. Now the pseudo-division formula (1) in Section 2 leads to

$$h_{p_k}^\alpha p = p \text{quo}(p, p_k) p_k + \text{prem}(p, p_k), \quad (14)$$

where $\alpha = \max\{0, \deg(p, y) - \deg(p_k, y) + 1\}$. If $\deg(p, y) < \deg(p_k, y)$, then $p = \text{prem}(p, p_k) \in \langle T_{k-1} \rangle \subset \langle T_k \rangle$ holds and we are done. From now on, we assume $\deg(p, y) \geq \deg(p_k, y)$ and we write $\alpha = \deg(p, y) - \deg(p_k, y) + 1$. With (14) we observe that we have the following equation in $R[y]$

$$h_{p_k}^\alpha p = q p_k. \quad (15)$$

We consider a more general situation: let $s \in \text{sat}(T_k)$, let δ be a non-negative integer and let $u \in \mathbf{k}[x_1, \dots, x_n]$ such that

$$h_{p_k}^\delta s = u p_k \quad (16)$$

holds in $R[y]$. In order to prove that $p \in \langle T_k \rangle$ holds, we prove that $s \in \langle T_k \rangle$ by induction on the number of terms in u . For simplicity, we denote

$$p_k = \sum_{i=0}^e a_i y^i \quad \text{and} \quad u = \sum_{i=0}^f b_i y^i,$$

with $a_e \neq 0$ and $b_f \neq 0$. Note that $a_e = h_{p_k}$.

If $u = 0$ in $R[y]$, then $a_e^\delta s = 0$ in $R[y]$. Since a_e is regular in R , we deduce $s = 0$ in $R[y]$, that is, $s \in \langle T_{k-1} \rangle$ and thus $s \in \langle T_k \rangle$. Assume $u \neq 0$ in $R[y]$. Let f' be the largest integer such that $b_{f'} \notin \langle T_{k-1} \rangle$ and write $u' = \sum_{i=0}^{f'} b_i y^i$. We have

$$a_e^\delta s = u' p_k \text{ in } R[y]. \quad (17)$$

By Lemma 3.1, for any $0 \leq i \leq e$, we have $a_e^\delta \mid b_{f'} a_i^{f'+1}$ in R . Since p_k is weakly primitive in $R[y]$, by Proposition 3.6 we have $a_e^\delta \mid b_{f'}$ in R . Thus there exists $\gamma \in \mathbf{k}[x_1, \dots, x_{j-1}]$, $\gamma \neq 0$ in R , such that

$$a_e^\delta \gamma = b_{f'} \quad \text{in } R. \quad (18)$$

We define

$$s' = s - \gamma y^{f'} p_k. \quad (19)$$

Since $s \in \text{sat}(T_k)$ we have $s' \in \text{sat}(T_k)$. Moreover we have

$$u' = a_e^\delta \gamma y^{f'} + \text{tail}(u').$$

Therefore, the following holds in $R[y]$:

$$a_e^\delta s' = \text{tail}(u') p_k. \quad (20)$$

By induction hypothesis we have $s' \in \langle T_k \rangle$. With (19) we conclude $s \in \langle T_k \rangle$, as desired. $\square$

5. WEAK PRIMITIVITY TEST

In this section, we point out the componentwise nature of weak primitivity. That is, if R can be written as a direct product of rings, then checking weak primitivity over R reduces to checking weak primitivity over each of its “components”.

Lemma 5.1 *Let $R_1, \dots, R_n$ be commutative rings with 1. Let $R = \prod_{i=1}^n R_i$ be their direct product and let π_k be the canonical projection from R to R_k . Let $a, b \in R$. Then $a \mid b$ in R if and only if $\pi_k(a) \mid \pi_k(b)$ for each $1 \leq k \leq n$.*

The proof of this lemma is straightforward, and thus is omitted.

Proposition 5.2 *Let $R = \prod_{i=1}^n R_i$ be a direct product of rings and let π_k be the canonical projection from R to R_k and τ_k be the canonical injection from R_k to R . Let $p = \sum_{i=0}^e a_i x^i \in R[x]$ be a polynomial with a_e being regular in R . Then p is weakly primitive in $R[x]$ if and only if $\pi_k(p) = \sum_{i=0}^e \pi_k(a_i) x^i$ is weakly primitive in $R_k[x]$ for each $1 \leq k \leq n$.*

PROOF. For any $1 \leq k \leq n$, denote $p_k = \pi_k(p)$. Since a_e is regular in R , $\pi_k(a_e) \neq 0$ for each k , and then each p_k is a polynomial of degree e .

First we prove that if all p_k are weakly primitive then p is also weakly primitive. Let $\beta \in R$ satisfying $a_e \mid a_i \beta$ for $0 \leq i \leq e-1$. By definition, we need to prove that $a_e \mid \beta$ in R .

Applying π_k to $a_e \mid a_i \beta$, we have $\pi_k(a_e) \mid \pi_k(a_i) \pi_k(\beta)$, for $0 \leq i \leq e-1$. By the weak primitivity of p_k , we have $\pi_k(a_e) \mid \pi_k(\beta)$. So there exists $u_k \in R_k$ such that $\pi_k(a_e) u_k = \pi_k(\beta)$. Define $u = (u_1, \dots, u_n) \in \prod_{i=1}^n R_i$. Then $\pi_k(u) = u_k$, and hence $\pi_k(a_e) \pi_k(u) = \pi_k(\beta)$, for each $1 \leq k \leq n$. By Lemma 5.1, $a_e \mid \beta$ in R . We proved that p is weakly primitive in $R[x]$.

On the other hand, we prove that, if p_k is not weakly primitive over R_k for some $1 \leq k \leq n$ then p is not weakly primitive over R . For simplicity, we assume $k = 1$. So, there exists $\beta_1 \in R_1$ such that $\pi_1(a_e) \nmid \pi_1(a_i) \beta_1$ for $0 \leq i \leq e-1$, but $\pi_1(a_e) \mid \pi_1(\beta)$. Define $\beta = \tau_1(\beta_1) = (\beta_1, 0, \dots, 0) \in R$. Then we claim that $a_e \nmid \beta$ and $a_e \mid a_i \beta$ for $0 \leq i \leq e-1$. This implies that p is not weakly primitive over R , as desired.

Indeed, first we have $a_e \nmid \beta$, since $\pi_1(a_e) \nmid \pi_1(\beta) = \beta_1$. Second, to prove $a_e \mid a_i \beta$ for $0 \leq i \leq e-1$, by Lemma 5.1, we need to prove that $\pi_k(a_e) \mid \pi_k(a_i \beta)$ for $1 \leq k \leq n$ and $0 \leq i \leq e-1$. If $k = 1$, it follows from the choice of β_1 . If $2 \leq k \leq n$, we have

$$\pi_k(a_i \beta) = \pi_k(a_i) \pi_k(\beta) = \pi_k(a_i) \cdot 0 = 0$$

for $1 \leq i \leq e-1$. Thus $\pi_k(a_e) \mid \pi_k(a_i \beta)$ holds for $1 \leq i \leq e-1$. $\square$

Example 5.3 *Let $T = \{p_1, p_2\}$ be a regular chain in $\mathbb{Q}[t \prec x \prec y]$ with $p_1 = x(x-t)$, $p_2 = (x+t)y+t$. Since $p_1 = x^2 - tx$ is strongly primitive in $(\mathbb{Q}[t])[x]$, p_1 is weakly primitive in $(\mathbb{Q}[t])[x]$. Let $R = \mathbb{Q}[t, x]/\langle x(x-t) \rangle$. Then we have*

$$R = R_1 \times R_2 = \mathbb{Q}[x, t]/\langle x \rangle \times \mathbb{Q}[x, t]/\langle x-t \rangle \simeq \mathbb{Q}[t] \times \mathbb{Q}[t].$$

Over R_1 , $p_2 = ty + t$ is not weakly primitive, since t is not invertible over R_1 and according to the definition we can choose $\beta = 1$. Hence T is not a primitive regular chain.

In order to generalize the construction of the above example into an algorithm, one would need to use algebraic factorization. In the next section, we propose a primitivity test for regular chains which avoids algebraic factorization, relying instead on polynomial GCDs modulo regular chains. Based on the algorithms and software tools available today we view it as a practical solution, as confirmed in Section 7.

6. A PRIMITIVITY TEST ALGORITHM

In Section 4, we define the notion of primitive regular chain which generalizes that of primitive polynomial over a UFD. In this section, we present another characterization on primitivity in terms of regularity of a polynomial. As a consequence, we obtain an algorithm to test whether a regular chain is primitive or not.

Lemma 6.1, 6.2, 6.3 and 6.4 are well-known facts. The proofs of Lemma 6.1 and 6.4 are straightforward. Lemma 6.2 can be found as Lemma 9.2.3 in [12] whereas Lemma 6.3 is Lemma 7 in [8].

Lemma 6.1 *Let I be a proper ideal of R and let h be an element of R . Then h is regular modulo I if and only if $I = I : h^\infty$ holds.*

Lemma 6.2 *Let a and b be two regular elements of R . Assume that a and b are not invertible. If a is regular modulo $\langle b \rangle$ then b is also regular modulo $\langle a \rangle$.*

Lemma 6.3 (Mc Coy Lemma) *A non-zero polynomial $f \in R[x]$ is a zerodivisor if and only if there exists a non-zero element $a \in R$ such that $af = 0$ holds.*

Lemma 6.4 *Let $f \in R[x]$ be a non-constant polynomial. If its leading coefficient is a regular element in R , then f is not a unit.*

Proposition 6.5 *Let R be a Noetherian commutative ring with 1. Consider a polynomial $f = \sum_{i=0}^n a_i x^i \in R[x]$. Assume that n is at least 1 and a_n is regular in R . Then $\langle f \rangle = \langle f \rangle : a_n^\infty$ holds if and only if a_n is invertible in R , or $\text{tail}(f)$ is regular modulo $\langle a_n \rangle$.*

PROOF. If a_n is invertible in R , then clearly $\langle f \rangle : a_n^\infty = \langle f \rangle$ holds. So we assume that a_n is not invertible in R . Note that both a_n and f are regular in $R[x]$; this follows from Lemma 6.3. Since a_n is not invertible in R , a_n is not invertible in $R[x]$ either. Since a_n is regular in R , it follows from Lemma 6.4 that f is not invertible in $R[x]$. Then, applying Lemma 6.1 and 6.2, we deduce

$$\begin{aligned} \langle f \rangle = \langle f \rangle : a_n^\infty &\iff a_n \text{ is regular modulo } \langle f \rangle \\ &\iff f \text{ is regular modulo } \langle a_n \rangle \\ &\iff \text{tail}(f) \text{ is regular modulo } \langle a_n \rangle. \end{aligned}$$

This completes the proof. $\square$

The following corollary may be seen as another characterization of the primitivity of a regular chain. This also provides an algorithm for checking whether a regular chain is primitive or not.

Corollary 6.6 (Primitivity test of a regular chain)

Let $T \subset \mathbf{k}[x_1, \dots, x_{s-1}]$ be a primitive regular chain. Let $p = \sum_{i=0}^e a_i x_s^i \in \mathbf{k}[x_1, \dots, x_s]$ with a_e being regular modulo $\text{sat}(T)$. Denote $\text{tail}(p) = \sum_{i=0}^{e-1} a_i x_s^i$. Then $T \cup \{p\}$ is a primitive regular chain if and only if a_e is invertible modulo $\text{sat}(T)$, or $\text{tail}(p)$ is a regular polynomial modulo $\langle T \cup \{a_e\} \rangle$.

PROOF. This is a direct consequence of Proposition 6.5, Theorem 4.4 and the definition of a regular chain. $\square$

Thus the problem of checking whether a regular chain $T \cup \{p\}$ is primitive or not, reduces to checking whether the polynomial $\text{tail}(p)$ is regular or not modulo $\langle T, a_e \rangle$. We next show that $\langle T, a_e \rangle$ in Corollary 6.6 generates an unmixed ideal; this result is crucial in view of Algorithm 1 below. Indeed, it allows us to deal with the following subtle point: a polynomial p regular modulo the radical $\sqrt{I}$ of an ideal I may not be regular modulo I . For example, consider $p = y$ and $I = \langle xy, x^2 \rangle$. Then y is a zerodivisor modulo I but y is regular modulo $\sqrt{I} = \langle x \rangle$. If I is unmixed, then p is regular modulo I if and only if p is regular modulo $\sqrt{I}$.

Lemma 6.7 Let $R = \mathbf{k}[x_1, \dots, x_n]$ and T be a primitive regular chain of R . If $t \in R$ is regular but not invertible modulo $\text{sat}(T)$, then (T, t) is a regular sequence of R and the ideal $\langle T, t \rangle$ is unmixed with dimension $n - |T| - 1$.

PROOF. Denote $T_i = T \cap \mathbf{k}[x_1, \dots, x_i]$. Since T is primitive, $\text{sat}(T_i) = \langle T_i \rangle$ holds for each i . Thus T is already a regular sequence of R . Now since t is regular but not invertible modulo $\text{sat}(T) = \langle T \rangle$, by definition (T, t) is a regular sequence.

Let $I = \langle T, t \rangle$ and $d = |T|$. According to the Principal Ideal Theorem (See Theorem 10.2 of [10]) the dimension $\dim(I)$ of I is at least $n - (d + 1)$. On the other hand, since (T, t) is a regular sequence of length $d + 1$, the dimension of I is at most $n - (d + 1)$. Hence, $\dim(I) = n - (d + 1)$ and then I is unmixed, by Macaulay Unmixedness Theorem (See Theorem 5.7 of [19]). $\square$

Algorithm 1 IsPrimitive

Input: T , a regular chain of $\mathbf{k}[x_1, \dots, x_n]$.

Output: true if T is primitive, false otherwise.

```

1: if  $|T| = 1$  then
2:    $t \leftarrow$  the defining polynomial of  $T$ 
3:   if  $\text{content}(t, \text{mvar}(t)) \in \mathbf{k}$  then return true else return false
4: else
5:   write  $T$  as  $T' \cup \{t\}$ , where  $t$  has the greatest main variable
6:   if not IsPrimitive( $T'$ ) then
7:     return false
8:   else
9:      $h \leftarrow \text{init}(t)$ ,  $r \leftarrow \text{tail}(t)$ 
10:    for  $U \in \text{Triangularize}(T' \cup \{h\})$  do
11:      if  $\text{ires}(r, U) = 0$  then return false
12:    end for
13:    return true
14:  end if
15: end if

```

Remark 6.8 (on the procedure IsPrimitive)

- (1) The function **Triangularize** decomposes a polynomial system F into a finite set of regular chains U_i such that $\sqrt{\langle F \rangle} = \bigcap_i \sqrt{\text{sat}(U_i)}$ holds; this is called a triangular decomposition of F in the sense of Kalkbrener [3]. According to the above specification, the set of the associated primes of $\sqrt{\langle F \rangle}$ are “implicitly” represented by U_i ’s.

Triangularize is one of the core functions in the REGULARCHAINS library [15]; it implements the triangular decomposition algorithm of [17]. While computing in Kalkbrener’s sense, it has the same specification as the function **solve_n** in Kalkbrener [13], although the algorithms of [17] and [13] are quite different.

Apart from Kalkbrener’s sense, **Triangularize** can also work in Lazard’s sense [3], where all solutions of the input systems will be explicitly represented by means of regular chains. In general, this function runs faster in Kalkbrener’s sense, since only generic solutions will be represented explicitly.

- (2) The use of **Triangularize** seems hard to avoid. The purpose is to represent all associated primes of the ideal $\langle T \cup \{h\} \rangle$ by means of regular chains. Geometrically, it is the intersection of the zero set of T with the hypersurface defined by h .

- (3) Algorithm 1 can be optimized using Item (1) of Remark 3.4: if a coefficient a_i of $t = a_e x^e + \dots + a_0$ is a nonzero constant, then lines 10-12 can be skipped since t is strongly primitive.

PROOF OF ALGORITHM ISPRIMITIVE. Termination of the algorithm follows from the fact that in each recursive call the number of polynomials in the input regular chain decreases by 1.

For the correctness, we proceed by induction on the number of polynomials in the regular chain T . When $|T| = 1$, the specification follows from Remark 4.3. So we assume $|T| > 1$. Definition 4.1 and Theorem 4.4 imply that if T is primitive then T' is also primitive. So we assume that T' is primitive and branch to line 9.

Let $\mathcal{U}$ be the output of **Triangularize** in line 10 and let $I = \langle T' \cup \{h\} \rangle$. From the specification of **Triangularize**, we have

$$\bigcap_{U \in \mathcal{U}} \sqrt{\text{sat}(U)} = \sqrt{I}.$$

By Corollary 6.6, we need to distinguish two cases: h is invertible (resp. not invertible) modulo $\langle T' \rangle = \text{sat}(T')$.

If h is invertible modulo $\langle T' \rangle$ then $\mathcal{U}$ is empty, and the algorithm correctly returns true. Assume from now on that h is not invertible modulo $\langle T' \rangle$. In this case by Lemma 6.7, the triangular decomposition $\mathcal{U}$ is not empty. So T is primitive if and only if r is regular modulo I . By Lemma 6.7 again, the ideal I is unmixed and therefore T is primitive if and only if r is regular modulo $\sqrt{I}$. This holds if and only if r is regular modulo $\text{sat}(U)$ for each $U \in \mathcal{U}$. Finally, the correctness of Algorithm 1 follows from Theorem 2.4. $\square$

Example 6.9 Let $R = \mathbf{k}[z \prec y \prec x]$ be a polynomial ring and $T = \{t_1, t_2\}$ be a regular chain of R with $t_1 = y^3 - z^2$, $t_2 = yx - z$. Clearly, $\{t_1\}$ is a primitive regular chain. Let $I = \langle t_1, \text{lc}(t_2) \rangle = \langle t_1, y \rangle = \langle y, z^2 \rangle$. In Algorithm 1 the call to **Triangularize** will produce $\sqrt{I} = \text{sat}(U)$ where $U = \{z, y\}$ is a regular chain. Thus, the computation

$$\text{ires}(\text{tail}(t_2), U) = \text{ires}(-z, U) = 0$$

implies that $\text{tail}(t_2) = -z$ is a zerodivisor modulo I . Thus T is not primitive. In fact, $\text{sat}(T) = \langle xy - z, xz - y^2, x^2 - y \rangle$ defines the twisted cubic which can not be generated by only two polynomials.

The above example implies that not every prime ideal can be generated by a primitive regular chain.

7. EXPERIMENTATION

We implemented the algorithm **IsPrimitive** on top of the REGULARCHAINS library [15] in MAPLE. The experimentation, described hereafter, was conducted on well-known problems used in [5]¹, and the tests were performed in MAPLE 11 on an Intel Pentium 4 machine (3.20GHz CPU, 2.0GB memory).

First, we computed their triangular decompositions using the **Triangularize** command in the sense of Kalkbrener. Then, we applied the **IsPrimitive** algorithm to each regular chain in the output.

In Table 1, we summarize the features of the problems and our experimental results. The name of the problems are listed in the first column. The second column gives the number n of variables and the maximal total degree d . For each triangular decomposition (which is a list of regular chains) we record the total running time (in seconds) of **IsPrimitive** in the third column. The last column is the result of mapping **IsPrimitive** to each triangular decomposition: in each of these patterns Y stands for *true* and N for *false*.

¹The defining polynomial systems can be found at <http://www.orcca.on.ca/~panwei/issac08/>

These data show that the procedure **IsPrimitive** is efficient in practice. This agrees with the fact that, in Algorithm 1, the input polynomial set in each call to **Triangularize** is rather structured. We also observe that primitive regular chains appear quite often in the output of triangular decompositions.

Table 1: Tests for IsPrimitive on 14 examples

System	(n, d)	Time	Pattern
KdV575	(26, 3)	3.525	[Y, Y, Y, Y, Y, Y, Y]
MontesS11	(6, 4)	.001	[Y]
MontesS16	(15, 2)	.103	[Y, Y, Y, N, Y, Y, Y]
Wu-Wang2	(13, 3)	0.099	[Y, N, Y, Y, Y]
MontesS10	(7, 3)	.145	[N]
Lazard2001	(7, 4)	2.314	[Y, Y, Y, N, Y, N]
Lanconelli	(11, 3)	.062	[N, Y]
Wang93	(5, 3)	.142	[N]
Leykin-1	(8, 4)	.228	[Y, Y, Y, Y, Y, Y, Y, Y, Y, Y, N]
MontesS14	(5, 4)	1.171	[Y, N, N]
MontesS15	(12, 2)	.312	[N]
MacLane	(10, 2)	.157	[Y, Y, N, Y, N]
MontesS12	(8, 2)	.042	[N]
Liu-Lorenz	(5, 2)	1.117	[N, Y]

8. CONCLUDING REMARK

We have generalized the notion of primitivity from univariate polynomials to regular chains. This has allowed us to establish a necessary and sufficient condition for a regular chain T to generate its saturated ideal $\text{sat}(T)$. Assume that T is not empty and write $T = T' \cup \{p\}$ where p is the polynomial of T with largest main variable. Theorem 4.4 states that the equality $\langle T \rangle = \text{sat}(T)$ holds whenever $\langle T' \rangle = \text{sat}(T')$ holds and the polynomial p is *weakly primitive* over $\mathbf{k}[x]/\langle T' \rangle$. This latter property is a generalization of the usual notion of primitivity for polynomials over a UFD.

Examining the proof of Theorem 4.4, we make the following observation. When p is not weakly primitive over $\mathbf{k}[x]/\langle T' \rangle$, the proof exhibits a polynomial q which belongs to $\text{sat}(T)$ but not to $\langle T \rangle$. When p is weakly primitive over $\mathbf{k}[x]/\langle T' \rangle$, the proof shows that every polynomial q of $\text{sat}(T)$ belongs to $\langle T \rangle$. The argument is constructive providing that one has at hand an algorithm for dividing a by b modulo $\langle T' \rangle$, where b is a polynomial regular modulo $\langle T' \rangle$ and is a multiple of the polynomial a modulo $\langle T' \rangle$. This can be done via Gröbner basis computations, see [16]. An algorithmic solution based on the algorithms of the REGULARCHAINS library is an ongoing research work.

Theorem 4.4 and its proof do not lead directly to an algorithm for testing the equality $\langle T \rangle = \text{sat}(T)$. Algorithm 1 provides such a decision procedure. This algorithm reduces to testing whether a polynomial is regular modulo an ideal. Fortunately the involved ideal is unmixed which allows us to rely on the algorithms of the REGULARCHAINS library avoiding Gröbner basis computations. Our experimentation illustrates the practical efficiency of Algorithm 1.

An application of this procedure is in the removal of redundant components for triangular decompositions in the sense of Kalkbrener. However, this procedure provides only a criterion for removing redundant components. Obtaining an algorithm, free of Gröbner basis computations, for testing the inclusion of saturated ideals remains an open problem.

9. REFERENCES

- [1] J. Arnold and R. Gilmer. On the contents of polynomials. *Proc. Amer. Math. Soc.*, 24:556–562, 1970.
- [2] P. Aubry, D. Lazard, and M. Moreno Maza. On the theories of triangular sets. *J. Symb. Comput.*, 28(1-2):105–124, 1999.
- [3] P. Aubry and M. Moreno Maza. Triangular sets for solving polynomial systems: A comparative implementation of four methods. *J. Symb. Comp.*, 28(1-2):125–154, 1999.
- [4] F. Boulier, F. Lemaire, and M. Moreno Maza. Well known theorems on triangular systems and the D5 Principle. In *Transgressive Computing 2006*, Universidad de Granada, 2006.
- [5] C. Chen, F. Lemaire, O. Golubitsky, M. Moreno Maza, and W. Pan. Comprehensive triangular decomposition. In *Proc. of CASC 2007. Lecture Notes in Computer Science*, volume 4770, pages 73–101. Springer-Verlag, 2007.
- [6] C. Chen, F. Lemaire, M. Moreno Maza, W. Pan, and Y. Xie. Efficient computations of irredundant triangular decompositions with the REGULARCHAINS library. In *Proc. of CASA2007. Lecture Notes in Computer Science*, volume 4488, pages 268–271. Springer-Verlag, 2007.
- [7] S.C. Chou and X.S. Gao. On the dimension of an arbitrary ascending chain. *Chinese Bull. of Sci.*, 38:799–804, 1991.
- [8] T. Coquand, L. Ducos, H. Lombardi, and C. Quitté. L’idéal des coefficients du produit de deux polynômes. *Revue des Mathématiques de l’enseignement Supérieur*, 113(3):25–39, 2003.
- [9] A. Corso, W. V. Vasconcelos, and R. H. Villarreal. On the contents of polynomials. *J. Pure. Appl. Algebra*, 125(1-3):117–127, 1998.
- [10] D. Eisenbud. *Commutative Algebra*. Springer-Verlag, 1994.
- [11] E. Hubert. Notes on triangular sets and triangulation-decomposition algorithms i: Polynomial systems. In *SNSC*, pages 1–39, 2001.
- [12] F. Ischebeck and R. A. Rao. *Ideals and reality. Projective modules and number of generators of ideals*. Springer-Verlag, 2005.
- [13] M. Kalkbrener. A generalized euclidean algorithm for computing triangular representations of algebraic varieties. *J. Symb. Comp.*, 15:143–167, 1993.
- [14] M. Kalkbrener. Algorithmic properties of polynomial rings. *J. Symb. Comput.*, 26(5):525–581, 1998.
- [15] F. Lemaire, M. Moreno Maza, and Y. Xie. The REGULARCHAINS library in MAPLE 10. In Ilias S. Kotsireas, editor, *Maple Conference 2005*, pages 355–368, 2005.
- [16] M. Monagan and R. Pearce. Rational simplification modulo a polynomial ideal. In *ISSAC ’06*, pages 239–245. ACM, 2006.
- [17] M. Moreno Maza. On triangular decompositions of algebraic varieties. Technical Report TR 4/99, NAG Ltd, Oxford, UK, 1999. Presented at the MEGA-2000 Conference, Bath, England. <http://www.csd.uwo.ca/~moreno>.
- [18] M. Moreno Maza and R. Rioboo. Polynomial gcd computations over towers of algebraic extensions. In *Proc. AAECC-11, Lecture Notes in Computer Science*, volume 948, pages 365–382. Springer, 1995.
- [19] B. Sturmfels. *Solving Systems of Polynomial Equations*. Amer. Math. Soc., 2002.
- [20] D. Wang. Computing triangular systems and regular systems. *J. Symb. Comput.*, 30(2):221–236, 2000.
- [21] W.T. Wu. On zeros of algebraic equations – an application of Ritt principle. *Kexue Tongbao*, 31(1):1–5, 1986.
- [22] L. Yang and J. Zhang. Searching dependency between algebraic equations: an algorithm applied to automated reasoning. Technical report ic/91/6, International Atomic Energy Agency, Miramare, Trieste, Italy, 1991.

Deciding Polynomial-Exponential Problems

Melanie Achatz
Faculty of Mathematics and
Informatics
University of Passau
D-94030 Passau, Germany
Melanie_Achatz@gmx.de

Scott McCallum
Department of Computing
Macquarie University
NSW 2109, Australia
scott@ics.mq.edu.au

Volker Weispfenning
Faculty of Mathematics and
Informatics
University of Passau
D-94030 Passau, Germany
weispfen@uni-passau.de

ABSTRACT

This paper presents a decision procedure for a certain class of sentences of first order logic involving integral polynomials and the exponential function in which the variables range over the real numbers. The inputs to the decision procedure are prenex sentences in which only the outermost quantified variable can occur in the exponential function. The decision procedure has been implemented in the computer logic system REDLOG. Closely related work is reported in [2, 7, 16, 20, 24].

Categories and Subject Descriptors

G.4 [Mathematics of Computation]: Mathematical software

General Terms

Algorithms, Theory

Keywords

Decision procedure, exponential polynomials

1. INTRODUCTION

In 1948 Tarski published a proof that the first order theory of the real numbers is decidable: indeed he exhibited a decision method for this theory [23]. In his monograph Tarski briefly considered an extended system in which one introduces a unary function symbol for exponentiation with respect to a fixed base. He remarked that the decision problem for such a system, which was still an open problem in 1948, is of great theoretical and practical interest. Over the following years many efforts were made to resolve this decision problem. The problem was conditionally solved in the positive sense in 1996 by Macintyre and Wilkie [15]. Their solution relies upon the plausible yet unproven Schanuel's conjecture in transcendental number theory. Their solution also is an indirect one, using powerful model-theoretic machinery; so it is not well suited for implementation.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

A number of papers have addressed, without recourse to Schanuel's conjecture and with practical implementability in mind, decision problems for fragments of the full first order theory of the reals with a specific transcendental function. For example, each of the works [2, 24] shows how to decide a certain kind of linear-transcendental problem using a relatively elementary and explicit approach well suited for implementation. Examples of related work are provided by a line of research initiated by Richardson [20]. This line of work addresses the computation of one and two dimensional polynomial-exponential systems using a variant of Sturm theory [16, 17]. Further examples of related work, in which algorithms are proposed (though not implemented) and relevant complexity bounds derived, can be found in works such as [19].

The present paper is in the spirit of [2, 24] in that its method addresses a certain fragment of the first order theory of the reals with exponentiation, relies on no unproven conjecture, and has been implemented in the computer logic system REDLOG [10]. In particular an unconditional and implementable decision method is presented for prenex sentences (that is, prenex formulas containing no free variables) having bound variables $x_1, x_2, \dots, x_n$, in which only x_1 occurs as argument of the exponential function. This non-trivial fragment of the extended system of real algebra is thus decided in both theory and practice. Our method and the REDLOG module embodying it thus provide an extension of computer algebra tools for real algebra into real analysis. Further extension to problems with several exponential variables appears difficult while Schanuel's conjecture remains unresolved.

Our decision method is based upon an algorithm for isolating the real zeros of a univariate exponential polynomial $p(x, \exp(x))$ (where $p(x, y)$ is a given bivariate integral polynomial). Our recursive real root isolation algorithm uses pseudodifferentiation and Rolle's theorem in the spirit of [8], and also relies upon a classical result of Lindemann [22] (see Section 2). The root isolation method of [16], in contrast, is based upon the construction of local Sturm sequences. We have reason to believe that our root isolation method is more efficient than that of [16] (see Section 3).

The remainder of the paper is organised as follows. Section 2 presents the formal framework and recalls the most essential background material for reading the paper. In Section 3 we outline an algorithm which decides *univariate* polynomial-exponential problems. This involves a careful study of real zero isolation for exponential polynomials. Section 4 presents a decision procedure for more general

polynomial-exponential problems which uses the method of Section 3 as a subalgorithm. Section 5 reports on the RED-LOG implementation of the decision procedure of Section 4, and discusses some examples. Section 6 discusses planned extensions of the work reported.

2. FORMAL FRAMEWORK AND BACKGROUND MATERIAL

In this section we introduce exponential polynomials, define the extension of the first order theory of the reals with which we shall be concerned, and specify some properties of the exponential function and exponential polynomials.

2.1 Exponential Polynomials

The symbol $\exp$ will denote the exponential function $x \mapsto e^x$ defined and analytic for all $x \in \mathbb{R}$.

Definition 1. (exponential polynomial)

Let $p(x, y)$ be an integral polynomial in x and y , i.e. $p(x, y) \in \mathbb{Z}[x, y]$. Putting $y = \exp(x)$ we obtain the associated **exponential polynomial** $p^*(x) := p(x, \exp(x))$, analytic on the whole real line.

The following result is not difficult to prove.

THEOREM 2.1. *Let $p(x, y)$ be an integral polynomial in x and y , and let $p^*(x) = p(x, \exp(x))$ be the associated exponential polynomial. Then the mapping $p(x, y) \mapsto p^*(x)$ is an injective homomorphism of $\mathbb{Z}[x, y]$ into the ring of all real valued analytic functions defined on the whole real line.*

Definition 2. (pseudodegree, pseudoderivative)

Let $p^*(x) = p(x, \exp(x))$ be the exponential polynomial associated to $p(x, y) = \sum_{i=0}^n p_i(x)y^i$. We define the following concepts:

- The **pseudodegree** of $p^*(x) \neq 0$, denoted by $\text{pdeg } p^*$, is the pair $(m, n) \in \mathbb{N}^2 = \mathbb{N} \times \mathbb{N}$, where $m = \deg_y p(x, y)$ and $n = \deg_x p(x, 0)$. (If $p(x, 0) = 0$, we set $n = 0$).
- The **pseudoderivative** of $p(x, y)$ is, with a slight abuse of notation, the polynomial $p'(x, y) = \partial p / \partial x + y \partial p / \partial y$ (cf. the notion of false derivative in [16, 20]). Notice that the derivative $(p^*)'$ of p^* thus satisfies $(p^*)'(x) = (p')^*(x)$.

2.2 Extension of the First-Order Theory of the Reals

The *extension of the first-order theory of the real numbers* with which we shall be concerned, denoted by $\mathcal{T}_{\exp}$, is a certain class of true sentences for the structure $\langle \mathbb{R}, \exp \rangle$, where $\mathbb{R} = \langle \mathbb{R}; +, -, \cdot, 0, 1, < \rangle$ denotes the real ordered field. Sentences of $\mathcal{T}_{\exp}$ are expressed in the language $\mathcal{L}_{\exp}$ in which the variables are $x_1, x_2, \dots$, the constant symbols are $0, 1$, the binary function symbols are $+$, $-$ and $\cdot$, the unary function symbol is $\exp$, to be applied *only* to the variable x_1 , and the binary order relation symbol is $<$. *Terms* of this language are integral polynomials in the variables $x_1, x_2, \dots$ and another variable y , where every occurrence of y is replaced by $\exp(x_1)$. By an *atomic formula* we mean an equation, inequation or inequality of the form $\tau = 0$, $\tau \neq 0$, $\tau > 0$, etc. where τ is a term. *Formulas* are constructed from atomic

formulas using boolean connectives and quantifiers. A *sentence* is a formula without free variables, usually expressed in prenex form:

$$(Q_1 x_1)(Q_2 x_2) \dots (Q_n x_n) \psi(x_1, x_2, \dots, x_n)$$

where ψ is a quantifier-free formula, and the $(Q_i x_i)$ are quantifiers.

EXAMPLES 2.2. (prenex sentences):

1. $(\forall x_1)(\exists x_2)[(x_1 + x_2) + x_2^2 y \neq x_1 y^2 \vee x_1 - x_2 = 0]$, where $y = \exp(x_1)$.
2. $(\exists x_1)[y - x_1 - 1 = 0 \wedge x_1 > 0]$, where $y = \exp(x_1)$.

In Sections 3 and 4 we shall present a decision method for $\mathcal{T}_{\exp}$.

2.3 Properties of the Exponential Function and Exponential Polynomials

Here we introduce the concept of an *admissible* function and show that the exponential function is admissible. Moreover, we establish crucial properties of exponential polynomials which we will need in our algorithm that isolates the real zeros of such functions and our algorithm that decides univariate polynomial-exponential problems.

Recall that a complex number is *transcendental* if it is not algebraic. F. Lindemann proved the following important result [22] concerning the transcendence of the values of the exponential function:

THEOREM 2.3. (Lindemann's theorem)

If z is a nonzero algebraic number, then e^z is transcendental.

The following concept was introduced in [2]:

Definition 3. (strongly transcendental function)

A real or complex analytic function $f(x)$ is called **strongly transcendental** (with exceptional points $\alpha_1, \alpha_2, \dots, \alpha_k$) if for all real or complex numbers x excluding the α_i ($x \neq \alpha_i$) not both x and $f(x)$ are algebraic.

COROLLARY 2.4. *The complex function e^z is strongly transcendental with exceptional point 0.*

The following concept is intended to capture the requirement on a real analytic, strongly transcendental function *trans*(x) which makes conceivable the provision of a finite description of the zeros of any polynomial function of *trans*(x).

Definition 4. (admissible function)

A strongly transcendental real valued function *trans*(x), with exceptional points $\alpha_1, \dots, \alpha_n$, analytic on the whole real line is called an **admissible** function with exceptional points α_i if for every nonzero integral polynomial $p(x, t)$, the associated real analytic function $p^*(x) = p(x, \text{trans}(x))$ has only finitely many zeros in the real line.

THEOREM 2.5. *The exponential function $\exp(x)$ is admissible with the exceptional point 0.*

PROOF. By Corollary 2.4, the exponential function $\exp(x)$ is strongly transcendental. According to Hardy [[11], page 17] a one-variable exponential function which does not vanish identically has only finitely many real zeros. Hence, for every nonzero $p(x, y) \in \mathbb{Z}[x, y]$, the associated real analytic function $p^*(x)$ has only finitely many real zeros. $\square$

We present some important properties of exponential polynomials.

LEMMA 2.6. *Let $p(x, y)$ be an irreducible element of $\mathbb{Z}[x, y]$ of positive degree in y . Let $p'(x, y)$ be the pseudoderivative of $p(x, y)$ (defined in Sec. 2.1). Let $r(x) = \text{res}_y(p, p')$. Then $r(x) \neq 0$ unless $p(x, y) = \pm y$ or $p'(x, y) = 0$.*

PROOF. Suppose $r(x) = 0$. Then p and p' have a common factor with positive degree in y . Since p is irreducible it follows that $p|p'$. Hence $p(x, 0)|p'(x, 0)$. But $p'(x, 0)$ is the ordinary derivative (with respect to x) of $p(x, 0)$. Hence we must have $p'(x, 0) = 0$. Therefore y is a factor of $p'(x, y)$. Let $q = \frac{p'}{y}$. Since $p|yq$ and p is irreducible, either $p|y$ or $p|q$. In the former case $p = \pm y$. In the latter case, observe that the degree in y of q is less than that of p , hence $q = 0$. Therefore $p' = 0$ also. $\square$

THEOREM 2.7. *Let $p(x, y)$ be an irreducible element of $\mathbb{Z}[x, y]$. Then the only possible non-simple real zero α of $p^*(x)$ is the exceptional point of $\exp(x)$ i.e. $\alpha = 0$.*

PROOF. p^* has no non-simple zeros if p has degree 0 in y . So suppose that p has positive degree in y . With $p'(x, y)$ denoting the pseudoderivative of $p(x, y)$, let $r(x) = \text{res}_y(p, p')$ and suppose that $r(x) = 0$. By the previous lemma, $p(x, y) = \pm y$ or $p'(x, y) = 0$. In the former case, $p^*(x) = \pm \exp(x)$. In the latter case, $p^*(x)$ is a nonzero constant. In both these cases $p^*(x)$ has no real zeros, hence no non-simple real zeros. So henceforth suppose that $r(x) \neq 0$. Let α be a non-simple zero of $p^*(x)$. Then $p^*(\alpha) = (p^*)'(\alpha) = 0$. Hence, with $\beta = \exp(\alpha)$, we have

$$p(\alpha, \beta) = p'(\alpha, \beta) = 0.$$

Therefore α is a root of $r(x) \neq 0$, hence algebraic. Now β is a root of the polynomial $p(\alpha, y)$, which is nonzero by the irreducibility of p . Hence β is also algebraic. By Lindemann's theorem (see Theorem 2.3) α must be an exceptional point of $\exp(x)$ (i.e. $\alpha = 0$). $\square$

THEOREM 2.8. *Let $p(x, y)$ and $q(x, y)$ be relatively prime nonzero elements of $\mathbb{Z}[x, y]$. Then the only possible common real zero α of $p^*(x)$ and $q^*(x)$ is the exceptional point of $\exp(x)$ i.e. $\alpha = 0$.*

PROOF. $p^*(x)$ and $q^*(x)$ have no common zeros if both p and q have degree 0 in y . So suppose that at least one of p and q has positive degree in y . Let α be a common zero of $p^*(x)$ and $q^*(x)$ which is not the exceptional point of $\exp(x)$, i.e. $\alpha \neq 0$. Then, with $\beta = \exp(\alpha)$, we have

$$p(\alpha, \beta) = q(\alpha, \beta) = 0.$$

Therefore α is a root of the resultant $\text{res}_y(p, q)$, which is a nonzero polynomial since p and q are assumed relatively prime. Hence β is also algebraic, since β is a root of the polynomials $p(\alpha, y)$ and $q(\alpha, y)$, at least one of which is nonzero by relative primality. But this contradicts Lindemann's theorem (see Theorem 2.3), which implies that not both α and β can be algebraic. $\square$

COROLLARY 2.9. *Let $p(x, y) \in \mathbb{Z}[x, y]$ be nonzero and square-free. Then the only possible non-simple real zero α of $p^*(x)$ is the exceptional point of $\exp(x)$ i.e. $\alpha = 0$.*

PROOF. Since $p(x, y)$ is squarefree it is a product of pairwise relatively prime irreducible elements of $\mathbb{Z}[x, y]$. The corollary follows immediately by application of the two preceding theorems. $\square$

3. DECIDING UNIVARIATE POLYNOMIAL-EXPONENTIAL PROBLEMS

The goal of this section is to present a decision method for those sentences of the theory $\mathcal{T}_{\exp}$ defined in Sec. 2 which involve only the variable x_1 . Indeed, we shall describe an algorithm DUPEP that decides univariate polynomial-exponential problems. In this section, for simplicity, we use the variable x instead of x_1 . The section is organized as follows. First we consider the problem of determining bounds for the real zeros of exponential polynomials. Next we present a key subalgorithm ISOL that isolates the real zeros of an exponential polynomial. Finally, a full description of the algorithm DUPEP is given.

3.1 Real Zero Bound for Exponential Polynomials

Our real zero isolation algorithm ISOL will require the determination of a real zero bound for exponential polynomials.

THEOREM 3.1. (upper bound for real zeros)
Let $p(x, y) = \sum_{i=0}^n p_i(x)y^i$ with $p_i(x) \in \mathbb{Z}[x]$ and $p_n(x) \neq 0$. Then an upper bound C for the real zeros of $p^*(x)$ can be obtained with the following procedure:

1. Find $C_1 > 0$ such that for all $x > C_1$, $|p_n(x)| \geq 1$.
2. Find $C_2 > 0$ and $k \in \mathbb{N}$ such that for all i in the range $0 \leq i < n$ and for all $x > C_2$, $|p_i(x)| \leq \frac{x^k}{n}$.
3. Find $C_3 > 0$ such that for all $x > C_3$, $x^k < \exp(\frac{x}{2})$.
4. Set $C \leftarrow \max\{C_1, C_2, C_3\}$.

PROOF. Let $x > C$. Then we can derive the inequality

$$\left| \sum_{i=0}^{n-1} p_i(x) \exp(x)^i \right| < |p_n(x)| e^{nx}$$

by applying 1, 2 and 3. Therefore, for $x > C$, we have $p^*(x) \neq 0$. Thus C is an upper bound for the real zeros of $p^*(x)$. Complete details concerning the determination of the numbers C_1, C_2, C_3 can be found in [1]. $\square$

A lower bound for the real zeros of $p^*(x)$ can be obtained by applying an analogous procedure to the exponential polynomial $g^*(y) = e^{ny}p^*(-y)$. The details can be found in [1]. See [16] for an alternative method for finding a real zero bound for exponential polynomials.

3.2 Isolating Real Zeros of Exponential Polynomials

We shall describe an algorithm to isolate the real zeros of a nonzero exponential polynomial $p^*(x) = p(x, \exp(x))$. This algorithm is based on differentiation and recursion on the pseudodegree $\text{pdeg } p^*$ of $p^*(x) \neq 0$. (Recall that the pseudodegree of $p^*(x)$ was defined in Section 2.1.) Related algorithms for polynomial real root isolation can be found in [9, 13]. We shall use the lexicographic order $\leq$ on $\mathbb{N}^2$ (defined, for example, in Exercise 4.61 of [4]): $(k, l) \leq (m, n)$ means that $k < m$ or $(k = m \text{ and } l < n)$ or $(k = m \text{ and } l = n)$. Theorem 4.62 of [4] implies that this linear (indeed admissible) order on $\mathbb{N}^2$ is a *well-order*, that is, every non-empty subset S of $\mathbb{N}^2$ has a least element. Hence the principle of *noetherian induction* can be used to prove a claim of

the kind, “ $P(m, n)$ is true for all $(m, n) \in \mathbb{N}^2$ ”, as explained in [4]. Our real zero isolation algorithm uses recursion on the pseudodegree, and we will demonstrate its validity using noetherian induction.

Our algorithm uses the concept of a *modulus of continuity* (*moc*) [5, 6] for a real valued function $f(x)$ on a nonempty compact interval $[a, b]$ of the real line. A positive real valued function δ defined on the set of all positive real numbers is called a *moc* for $f(x)$ on $[a, b]$ if for all $\epsilon > 0$ and for all $x, y \in [a, b]$ $|x - y| \leq \delta(\epsilon)$ implies $|f(x) - f(y)| \leq \epsilon$. By definition $f(x)$ is uniformly continuous on $[a, b]$ if and only if there is a moc for $f(x)$ on $[a, b]$. Moreover by a standard theorem of analysis the assertion of the preceding sentence remains valid if one omits the modifier ‘uniformly’. The following theorem provides an explicit linear moc for a continuously differentiable function.

THEOREM 3.2. *Let $f(x)$ be real valued and continuously differentiable on $I = [a, b]$. Let M be a positive number with $M \geq \max_{x \in I} |f'(x)|$. Then a linear moc δ for $f(x)$ on I can be obtained by putting $\delta(\epsilon) = \epsilon/M$.*

PROOF. Let $\epsilon > 0$, let $x, y \in I$, and suppose that $|x - y| \leq \delta(\epsilon)$. By the mean value theorem, $f(x) - f(y) = f'(c)(x - y)$, for some c between x and y . Therefore

$$|f(x) - f(y)| = |f'(c)||x - y| \leq M|x - y| \leq M\delta(\epsilon) = M\epsilon/M = \epsilon.$$

This completes the proof. $\square$

We define some terminology. An *isolation list* for a real-valued function $f(x)$ defined on the whole real line is a list $L = (I_1, I_2, \dots, I_k)$, such that

- (a) k is the number of distinct real zeros of f ;
- (b) each $I_j = (a_j, b_j)$, where a_j and b_j are binary rational numbers;
- (c) each I_j contains a unique zero of f ; and
- (d) $a_1 < b_1 \leq a_2 < b_2 \leq \dots \leq a_k < b_k$.

ALGORITHM 3.3. (real root isolation of exponential polynomials)

$$L \leftarrow \text{ISOL}(p)$$

Input: $p(x, y) \in \mathbb{Z}[x, y]$, a nonzero squarefree integral polynomial.

Output: L , an isolation list for $p^*(x) = p(x, \exp(x))$.

- (1) [Basis.] Set $(m, n) \leftarrow \text{pdeg } p^*(x)$. If $(m, n) = (0, 0)$ then {Set $L \leftarrow ()$. Return}.
- (2) [Recursion.] Compute $p'(x, y)$, the pseudoderivative of $p(x, y)$. If $n > 0$ then set $s(x, y) \leftarrow \text{gsfd}(p'(x, y))$ else {Set $\hat{p}(x, y) \leftarrow p'(x, y)/y$. Set $s(x, y) \leftarrow \text{gsfd}(\hat{p}(x, y))$. Set $L' \leftarrow \text{ISOL}(s)$. [In this step ‘gsfd’ denotes ‘greatest squarefree divisor’. By construction, $\text{pdeg } s(x, y) < \text{pdeg } p(x, y)$. By noetherian induction hypothesis, L' is an isolation list for $s^*(x)$, hence for $(p')^*(x) = (p^*)'(x)$.]
- (3) [Bound for real zeros.] Set $\gamma \leftarrow$ a binary rational real zero bound for $p^*(x)$, using the method of Section 3.1. [Every real zero α of $p^*(x)$ satisfies $|\alpha| < \gamma$.]

- (4) [Prepare for induction step.] Let $L' = (I_1, I_2, \dots, I_k)$, with $I_j = (a_j, b_j)$. Let $\alpha'_0 = -\infty$ and $\alpha'_{k+1} = \infty$, and for $1 \leq j \leq k$, let α'_j denote the unique zero of $(p^*)'(x)$ in I_j . Observe that, by Corollary 2.9, $s^*(a_j)s^*(b_j) < 0$, for every j in the range $1 \leq j \leq k$, unless $\alpha'_j = 0$. Without loss of generality assume that, in case $k > 0$, $-\gamma \leq a_1$ and $b_k \leq \gamma$. Put $b_0 = -\gamma$ and $a_{k+1} = \gamma$. For $i = 0, 1, \dots, k$, put $J_i = (b_i, a_{i+1})$. [In case $b_i = a_{i+1}$ J_i is empty.] [By Rolle’s theorem, each interval $[\alpha'_i, \alpha'_{i+1}]$ contains at most one zero of $p^*(x)$. Hence each complementary interval $J_i = (b_i, a_{i+1})$ contains at most one such zero. Moreover, by Corollary 2.9, $p^*(x)$ has no non-simple zeros, with the possible exception of 0. Hence neither α'_i nor α'_{i+1} is a zero of $p^*(x)$, unless $\alpha'_i = 0$ or $\alpha'_{i+1} = 0$. The next step will ensure that, after suitable refinement of the I_j s, no $[a_j, b_j]$ contains a zero of $p^*(x)$, unless $\alpha'_j = 0$.] Set $L \leftarrow ()$.
- (5) [Interval refinement.] For $j = 1, 2, \dots, k$ {If $\alpha'_j = 0$ and $p^*(0) = 0$ then insert I_j into L else {Compute a linear moc δ for $p^*(x)$ on the (initial) interval $[a_j, b_j]$. Repeatedly bisect $I_j = (a_j, b_j)$, always retaining the subinterval of I_j which contains α'_j , that is, always maintaining the invariance of the relation $s^*(a_j)s^*(b_j) < 0$. But if $s^*(m_j) = 0$, with $m_j = (a_j + b_j)/2$, then retain the subinterval $(a_j + (b_j - a_j)/4, a_j + 3(b_j - a_j)/4)$ centred at $m_j = \alpha'_j$. Where $\epsilon = \min(|p^*(a_j)|, |p^*(b_j)|)/2$, terminate the bisection process when $\epsilon > 0$ and (then) $b_j - a_j \leq \delta(\epsilon)$.} } [For each j the repeated bisection process must terminate. For suppose that this is not the case. Then the repeated bisection process defines infinite sequences of values for a_j and b_j , hence for ϵ . Observe $p^*(\alpha'_j) \neq 0$, by Corollary 2.9. Now the sequence of values of ϵ tends to the limit $|p^*(\alpha'_j)|/2$, which is positive by the above observation. Hence the sequence of values of $\delta(\epsilon)$ tends to the limit $\delta(|p^*(\alpha'_j)|/2) > 0$ (as δ is a linear function). Hence eventually $\epsilon > 0$ and $b_j - a_j \leq \delta(\epsilon)$, contradicting the assumption about non-termination. Now $|b_j - \alpha'_j| < b_j - a_j$. Hence upon termination $|p^*(b_j) - p^*(\alpha'_j)| \leq \epsilon < |p^*(b_j)|$. Therefore $\text{sign}(p^*(\alpha'_j)) = \text{sign}(p^*(b_j)) \neq 0$. Hence $[\alpha'_j, b_j]$ contains no zero of $p^*(x)$. We can similarly show that $[a_j, \alpha'_j]$ contains no zero of $p^*(x)$. Combining two of our conclusions we see that $[a_j, b_j]$ contains no zero of $p^*(x)$.]
- (6) [Completion of induction step.] For $i = 0, 1, \dots, k$ {If $p^*(b_i)p^*(a_{i+1}) < 0$ then insert J_i into L }. [By step 5, $(\alpha'_i, \alpha'_{i+1})$ contains a zero of $p^*(x)$ if and only if J_i does, which occurs if and only if $p^*(b_i)p^*(a_{i+1}) < 0$.] Return.

Computation of a linear moc δ for $p^*(x)$ on $[a_j, b_j]$, as required by step 5 of the above algorithm, could be done by applying Theorem 3.2. (To apply this theorem here one has to obtain a bound M for $|(p^*)'(x)|$ on $[a_j, b_j]$. Such a bound could be obtained using the triangle inequality and appropriate estimates for the component terms of $|(p^*)'(x)|$ on $[a_j, b_j]$.) The astute reader will note that ISOL requires a method to evaluate the sign of an exponential polynomial $s^*(x)$ at a binary rational number, say r , in case $s^*(r) \neq 0$. Such a method could be readily obtained by iterating sufficiently often a standard numerical procedure for computing

an interval $[u, w]$ of specified length $\epsilon > 0$ guaranteed to contain the value $v = s^*(r)$.

For the record, we assert the validity of our algorithm.

THEOREM 3.4. *For all $(m, n) \in \mathbb{N}^2$, the following statement $P(m, n)$ is true: for every valid input $p(x, y)$, with $\text{pdeg } p = (m, n)$, ISOL returns an isolation list L for $p^*(x) = p(x, \exp(x))$.*

PROOF. The main ingredients of the proof of this theorem by noetherian induction on (m, n) are supplied as comments in the above description of ISOL. $\square$

For ease of exposition and proof we have kept our description of ISOL conceptually simple. In practice there are elementary improvements which could be made to enhance the efficiency of the method. For example, we could insert an initial algorithm step (step 0, say) which finds the content $c(x)$ of $p(x, y)$ with respect to y and computes an isolation list L_0 for $c(x)$ using any highly efficient real root isolation algorithm for $\mathbb{Z}[x]$ [9, 13]. After setting $p(x, y) \leftarrow p(x, y)/c(x)$ we proceed with steps 1-6 (in which step 5 could be simplified slightly, by Theorem 3.5 below). We could then append a final algorithm step (step 7, say) which refines the isolating intervals in $L \leftarrow L_0 \cup L$ into an isolation list for the original $p^*(x)$.

Some comparison of ISOL with the root isolation method of [16] is warranted. Actually [16] is chiefly concerned with counting the number of real roots of a given exponential polynomial $p^*(x)$ in some interval (a, b) . The method is based on construction of local Sturm sequences. It requires computation of the real roots of two auxiliary univariate polynomials whose degrees are high relative to $\deg_x p$ and $\deg_y p$. It also involves evaluation of the signs of a list of exponential polynomials at rational points. So, although no systematic empirical comparison of the two root isolation algorithms has yet been undertaken, we think that our method is more efficient than that of [16].

3.3 Sample Points

Let $f^*(x)$ be the exponential polynomial associated to some given $f(x, y) \in \mathbb{Z}[x, y]$. We make no assumption about the squarefreeness of $f(x, y)$, so we cannot directly apply algorithm ISOL to isolate the zeros of $f^*(x)$. Let us nevertheless consider the zeros of $f^*(x)$, which determine a decomposition of the real line, which we term an *f^* -invariant* decomposition. Assuming that $f^*(x)$ has n zeros, then the decomposition consists of n 0-cells (the zeros) and $n + 1$ 1-cells (the open intervals between the zeros). A *sample point* for a cell is an exact representation of a particular algebraic or non-algebraic (i.e. transcendental) number belonging to that cell. Suppose that we could somehow obtain an isolation list for $f^*(x)$. (A method for doing so, which uses the algorithm ISOL, will be described in the next subsection.) Then as sample points for the 1-cells we use appropriately chosen rational endpoints from the isolating intervals obtained. If a 0-cell α is an algebraic number we use a standard representation for α by its minimal polynomial and an isolating interval, as described in [14]. For a 0-cell α which is not an algebraic number we represent α by the irreducible factor $p(x, y)$ of $f(x, y)$ for which $p^*(\alpha) = 0$ and an isolating interval for α .

We now describe how to determine the sign of a given exponential polynomial $g^*(x)$ at a given sample point $\alpha \neq 0$

of such an f^* -invariant decomposition of the real line. Such sign determination will be an important component of the algorithm DUPEP, described in the next subsection.

Suppose first that α is algebraic. For determining whether or not $g^*(\alpha) = 0$ the following theorem is relevant.

THEOREM 3.5. *Let $h(x, y) \in \mathbb{Z}[x, y]$ be primitive and of positive degree in y . Let $\alpha \neq 0$ be an algebraic number. Then $h^*(\alpha) \neq 0$.*

PROOF. Suppose that $h^*(\alpha) = 0$. Let $\beta = \exp(\alpha)$. Then $h(\alpha, \beta) = h^*(\alpha) = 0$. But $h(\alpha, y) \neq 0$, by the primitivity of $h(x, y)$. Therefore β is algebraic, contradicting Theorem 2.3. $\square$

We find the content $c(x)$ and the primitive part $h(x, y)$ of $g(x, y)$. Then $g^*(\alpha) = 0$ if and only if $c(\alpha) = 0$, by the above theorem. Suppose now that $g^*(\alpha) \neq 0$. Let I be an isolating interval for α . Using a method analogous to that of step 5 of ISOL, one refines I about α so that, after refinement, I contains no zero of $g^*(x)$. Then, by evaluating g^* at the left or right endpoint of the refined I , we can determine the sign of $g^*(\alpha)$.

Suppose on the other hand that α is not algebraic. Recall that α is represented by the irreducible factor $p(x, y)$ of $f(x, y)$ for which $p^*(\alpha) = 0$ and an isolating interval say I for α . Clearly $g^*(\alpha) = 0$ if p is a factor of g . So suppose that p is not a factor of g . Then, by Theorem 2.8, $g^*(\alpha) \neq 0$. Using a method analogous to that of step 5 of ISOL, one refines I about α so that, after refinement, I contains no zero of $g^*(x)$. Then, by evaluating g^* at the left or right endpoint of the refined I , we can determine the sign of $g^*(\alpha)$.

3.4 The Algorithm DUPEP

ALGORITHM 3.6. (deciding univariate polynomial-exponential problems)

$$v \leftarrow \text{DUPEP}(\varphi)$$

Input: A prenex sentence φ in $\mathcal{L}_{\text{exp}}$ involving only x .

Output: The truth value v of φ over the reals.

1. **Extraction.** The input φ is of the following form

$$(Qx)\psi(x)$$

where ψ is a quantifier-free formula of $\mathcal{L}_{\text{exp}}$ and (Qx) is a quantifier. Extract the list $P := \{p_1, p_2, \dots, p_n\}$ of those polynomials $p_i(x, y) \in \mathbb{Z}[x, y]$ for which $p_i^*(x)$ occurs in $\psi(x)$.

2. **Contents and primitive parts.** Compute the set $\text{cont}_y(P)$ of contents (w.r.t. y) of the elements of P and the set $\text{pp}_y(P)$ of primitive parts (w.r.t. y) of elements of P of positive degree in y .
3. **Squarefree bases.** Compute squarefree bases K and Q of $\text{cont}_y(P)$ and $\text{pp}_y(P)$, respectively.
4. **Root isolation.** Apply algorithm ISOL to each polynomial $q(x, y)$ in Q and to each polynomial $c(x)$ in K , individually.
5. **Isolation list for product.** By their relative primality, for any pair of distinct elements p and q of $K \cup Q$, p^* and q^* have common real zeros only at $x = 0$ (see Theorem 2.8). Hence by refining the isolating intervals for the zeros of all the $p^*(x) \in K \cup Q^*$ we obtain an isolation list for the product $f^*(x)$ of all $p^*(x)$.

6. **Sample points.** Use the isolation list for $f^*(x)$ to construct sample points for all of the cells of the decomposition of the real line determined by the zeros of $f^*(x)$, as described in the previous subsection.
7. **Evaluation.** Use the sample points to decide the original function $(Qx)\psi(x)$. [This can be done as follows. For each i express $p_i(x, y)$ as a product of elements of $K \cup Q$. By evaluating the signs of the exponential polynomials associated with the factors of each p_i at each sample point as described in the previous subsection, the sign of each p_i^* at each sample point can be determined. Then the truth value of the original formula φ can be decided.]

4. DECIDING MULTIVARIATE POLYNOMIAL-EXPONENTIAL PROBLEMS

In Section 3 we described an algorithm DUPEP that decides univariate polynomial-exponential problems. In this section we outline an extension to this procedure, i.e. an algorithm that decides polynomial-exponential problems in general.

Our decision algorithm DPEP accepts as input a prenex sentence φ in $\mathcal{L}_{\text{exp}}$. It produces as output the truth value v of the input φ over the real numbers.

The algorithm DPEP has two basic phases.

- (1) **First Phase.** Now the input prenex sentence φ has the form

$$(Q_1x_1)(Q_2x_2) \dots (Q_nx_n)\psi(x_1, x_2, \dots, x_n)$$

where ψ is a quantifier-free formula of $\mathcal{L}_{\text{exp}}$ and the (Q_ix_i) are quantifiers. Recall that ψ involves polynomials in the x_i and $\exp(x_1)$. We apply a quantifier-elimination (QE) algorithm for elementary real algebra (such as QE by CAD [3, 7]) to the following formula

$$\varphi' = (Q_2x_2)(Q_3x_3) \dots (Q_nx_n)\psi(x_1, x_2, \dots, x_n)$$

obtained by removing (Q_1x_1) from φ . (As the variable x_1 is not quantified in φ' , the quantifier elimination algorithm can proceed without any precautions concerning the exponential function.) The output is a quantifier-free formula $\psi_1(x_1)$ which is equivalent to φ' .

- (2) **Second Phase.** We combine the quantifier (Q_1x_1) of the input sentence φ with the output of the QE algorithm $\psi_1(x_1)$, thus obtaining a univariate polynomial-exponential decision problem instance:

$$\varphi'' = (Q_1x_1)\psi_1(x_1).$$

We apply algorithm DUPEP from Section 3 to this problem instance φ'' , obtaining the truth value v of φ'' over the real numbers. Finally we return v as the truth value of the input sentence φ over the real numbers.

The validity of algorithm DPEP is clear.

5. IMPLEMENTATION AND EXAMPLES

The univariate decision procedure DUPEP described in Section 3 was implemented by the first author [1] in the REDLOG package of the computer algebra system REDUCE

[12] under the guidance of the third author. Based on the package for quantifier elimination by cylindrical algebraic decomposition (QE by CAD) in REDLOG [21] this module was extended to an implementation of the more general decision procedure DPEP described in Section 4.

The functions implemented use the logical REDLOG context OFSF for the ordered field of real numbers regarded as a structure for the first-order language of ordered rings [10]. In order to avoid a cumbersome extension of this context for the handling of polynomial-exponential problems the following conventions were used: the variable x_1 is reserved as independent variable of the exponential function; and the value $\exp(x_1)$ of this function is represented by a new free variable y . Sentences φ of the formal language $\mathcal{L}_{\text{exp}}$ are entered into the decision program DPEP in prenex form

$$(Q_1x_1)(Q_2x_2) \dots (Q_nx_n)\psi(x_1, y, x_2, \dots, x_n),$$

where the (Q_ix_i) are quantifiers. Semantically the variables x_i range over the real numbers, the variable y is treated as the value $\exp(x_1)$, and the truth value of φ is evaluated accordingly in the ordered field of real numbers with exponentiation.

In REDLOG the user has the following commands to start, and to obtain verbose output of, the decision procedure: `rldpep`, `rldpepverbose`, and `rldpepivverbose`. The second command turns on the verbose output option. The third command provides an even more detailed trace of the procedure's workings.

The implementation of DPEP was tested on several examples with up to 3 quantifiers and produced correct results with running times in the range of 0.5 s to 12 s on a Pentium 4 (2 Ghz, 128 heap size). Here are a few of the examples used for testing purposes. In each example the correct truth value (true) was obtained in less than a second on the Pentium 4.

$$\begin{aligned} &(\exists x_1)(\exists x_2)[y - x_2^2 = 0 \wedge x_1 - x_2 = 0], \text{ where } y = \exp(x_1) \\ &(\forall x_1)[(1 - x_1) \cdot y \leq 1 \vee x_1 \geq 1], \text{ where } y = \exp(x_1) \\ &(\exists x_1)(\exists x_2)(\exists x_3)[2x_1 - x_2 + x_3 + y^2 = 0 \wedge 3x_2 - x_3 = 0 \wedge \\ &\quad 2x_1 + x_2 + 3x_3 + y = 0], \text{ where } y = \exp(x_1) \end{aligned}$$

Sentences involving polynomials in $x_1, x_2, \dots, x_n$ and a hyperbolic function such as $\cosh(x_1)$ can be massaged into a form for which the program DPEP can be applied. For example suppose we want to decide

$$(\forall x_1)[x_1 > 7 \Rightarrow \cosh(x_1) > p(x_1)]$$

for some integral polynomial $p(x_1)$. Using the defining relation

$$\cosh(x_1) = \frac{\exp(x_1) + \exp(-x_1)}{2}$$

and noting that $\exp(x_1)$ is positive on the real line, we see that the sentence of interest is equivalent to the sentence

$$(\forall x_1)[x_1 > 7 \Rightarrow \exp(x_1)^2 + 1 > 2p(x_1)\exp(x_1)]$$

of $\mathcal{L}_{\text{exp}}$. As a specific example we set $p(x_1) = x_1^3 - 4x_1$. The program DPEP found that the above sentence is true within 17 seconds.

Sentences involving polynomials in x and a Gaussian function such as $\exp(-x^2)$ can also be massaged into sentences of $\mathcal{L}_{\text{exp}}$. For example suppose we want to decide

$$(\exists x)[\exp(-x^2) = p(x)]$$

for some integral polynomial $p(x)$. We introduce a new variable say z to represent $-x^2$ and thus obtain an equivalent sentence:

$$(\exists z)(\exists x)[\exp(z) = p(x) \wedge z + x^2 = 0].$$

Replacing z by x_1 and x by x_2 we obtain an equivalent sentence of $\mathcal{L}_{\text{exp}}$. In a similar manner sentences involving polynomials in x and a more general exponential function of the form $\exp(\pi(x))$, where $\pi(x)$ is a rational polynomial, can be converted into sentences of $\mathcal{L}_{\text{exp}}$.

6. REFINEMENTS AND EXTENSIONS

A refinement of algorithm DPEP on efficiency grounds is contemplated. It would probably be desirable to try to develop a decision method for $\mathcal{T}_{\text{exp}}$ which is analogous to decision by cylindrical algebraic decomposition (CAD). In order to decide a sentence φ in the variables $x_1, x_2, \dots, x_n$ we would first extract the set A of polynomials occurring in φ . Second, by analogy with the CAD method, we would compute a description of a cylindrical *analytic* decomposition D of the space $\mathbb{R}^n$ compatible with the zeros of the polynomials in A . Third, we would use the description of D to decide the truth or falsity of φ by analogy with QE by CAD. Some progress has already been made in this direction [18].

A small extension of the work reported herein is planned. We will endeavour to extend our decision procedure for $\mathcal{T}_{\text{exp}}$ to decide sentences involving polynomials in x and a more general exponential function $\exp(\pi(x))$ (as described at the end of the previous section) *without* introducing a new variable. Some progress has already been made towards this goal [18]. We also plan to investigate the decidability of analogous theories obtained by introducing into elementary algebra other admissible functions such as the inverse trigonometric function $\arctan(x)$. Furthermore, while the trigonometric functions $\sin(x)$ and $\cos(x)$ are technically not admissible we have some optimism that similar results could be obtained for these functions.

Extension of our decision method to the full first order theory of the reals with exponentiation would of course be highly desirable. However this is probably difficult to achieve while Schanuel's conjecture remains unresolved.

7. REFERENCES

- [1] M. Achatz. Deciding polynomial-exponential problems. Diploma thesis, Universität Passau, D-94030 Passau, Germany, Sept. 2006.
- [2] H. Anai and V. Weispfenning. Deciding linear-trigonometric problems. In C. Traverso, editor, *ISSAC'2000*, pages 14–22. ACM-Press, 2000.
- [3] D. S. Arnon, G. E. Collins, and S. McCallum. Cylindrical algebraic decomposition i: Basic algorithm. In B. Caviness and J. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 136–151. Springer, Wien, New York, 1998.
- [4] T. Becker, V. Weispfenning, and H. Kredel. *Gröbner Bases, a Computational Approach to Commutative Algebra*, volume 141 of *Graduate Texts in Mathematics*. Springer, New York, corrected second printing edition, 1998.
- [5] E. Bishop. *Foundations of Constructive Analysis*. McGraw-Hill, New York, 1967.
- [6] E. Bishop and D. Bridges. *Constructive Analysis*. Grundlehren der math. Wissenschaften. Springer-Verlag, Berlin, 1985.
- [7] G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In B. Caviness and J. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 85–121. Springer, Wien, New York, 1998.
- [8] G. E. Collins and R. Loos. Polynomial real root isolation by differentiation. In *Proceedings of the 1976 Symposium on Symbolic and Algebraic Computation SYMSAC'76*, pages 15–25. ACM, 1976.
- [9] G. E. Collins and R. Loos. Real zeros of polynomials. In B. Buchberger, G. E. Collins, R. Loos, and R. Albrecht, editors, *Computer Algebra: Symbolic and Algebraic Manipulation*, pages 83–94. Springer-Verlag, Wien, New York, second edition, 1982.
- [10] A. Dolzmann and T. Sturm. Redlog user manual, edition 3.0. Technical report, University of Passau, 2004.
- [11] G. H. Hardy. *Orders of Infinity*. Cambridge Univ. Press, Cambridge, 1910.
- [12] A. Hearn. Reduce user's manual for version 3.8. Technical report, RAND Corporation, 2004.
- [13] J. R. Johnson. Algorithms for polynomial real root isolation. In B. Caviness and J. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 269–299. Springer, Wien, New York, 1998.
- [14] R. G. K. Loos. Computing in algebraic extensions. In B. Buchberger, G. Collins, and R. Loos, editors, *Computer Algebra: Symbolic and Algebraic Computation*, pages 173–188. Springer, Wien, New York, 1982.
- [15] A. Macintyre and A. Wilkie. On the decidability of the real exponential field. In *Kreiseliana: About and around Georg Kreisel*, pages 441–467. A.K. Peters, 1996.
- [16] A. Maignan. Solving one and two-dimensional exponential polynomial systems. In O. Gloor, editor, *Proceedings of the 1998 International Symposium on Symbolic and Algebraic Computation*, pages 215–221, New York, Aug. 1998.
- [17] A. Maignan. Real solving of elementary-algebraic systems. *Numerical Algorithms*, 27:153–167, 2001.
- [18] S. McCallum and V. Weispfenning. Deciding polynomial-exponential problems. Technical report, Macquarie University, 2006.
- [19] S. Pericleous and N. Vorobjov. New complexity bounds for cylindrical decompositions of sub-pfaffian sets. In B. Mourrain, editor, *Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation*, pages 268–275, New York, July 2001.
- [20] D. Richardson. Towards computing non algebraic cylindrical decompositions. In S. M. Watt, editor, *Proceedings of the 1991 International Symposium on Symbolic and Algebraic Computation*, pages 247–255, Bonn, Germany, July 1991.

- [21] A. Seidl. Cylindrical decomposition under application-oriented paradigms. Doctoral dissertation, Universität Passau, D-94030 Passau, Germany, Nov. 2006.
- [22] A. B. Shidlovskii. *Transcendental Numbers*. Walter de Gruyter, Berlin, New York, 1989.
- [23] A. Tarski. A decision algorithm for elementary algebra and geometry. In B. Caviness and J. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 24–84. Springer, Wien, New York, 1998.
- [24] V. Weispfenning. Deciding linear-transcendental problems. In V. Ganzha, E. Mayr, and E. Vorozhtshov, editors, *Computer Algebra in Scientific Computation - CASC 2000*, pages 423–438. Springer, 2000.

On Generalized LN-Surfaces in 4-Space

Martin Peternell
University of Technology
Vienna, Austria
martin@geometrie.tuwien.ac.at

Boris Odehnal
University of Technology
Vienna, Austria
boris@geometrie.tuwien.ac.at

ABSTRACT

The present paper investigates a class of two-dimensional rational surfaces Φ in $\mathbb{R}^4$ whose tangent planes satisfy the following property: For any three-space E in $\mathbb{R}^4$ there exists a unique tangent plane T of Φ which is parallel to E . The most interesting families of surfaces are constructed explicitly and geometric properties of these surfaces are derived. Quadratically parameterized surfaces in $\mathbb{R}^4$ occur as special cases. This construction generalizes the concept of LN-surfaces in $\mathbb{R}^3$ to two-dimensional surfaces in $\mathbb{R}^4$.

Categories and Subject Descriptors

I.1 [Symbolic and Algebraic Manipulation]: Miscellaneous

General Terms

Theory

Keywords

LN-surface, quadratically parameterized surface, linear congruence of lines, chordal variety, rational parameterization.

1. INTRODUCTION

In $\mathbb{R}^3$ there exists a remarkable class of rational surfaces which are characterized by possessing a field of normal vectors which is linear in the surface parameters. These so-called LN-surfaces [2, 3] possess remarkable properties. Their family of tangent planes are represented by graphs of rational functions [6]. LN-surfaces possess rational offset surfaces [3] and the convolution surface of an LN-surface with an arbitrary rational non-developable surface is always rational [11]. Surprisingly, quadratically parameterized surfaces belong to this class [6].

This article will generalize the concept of LN-surfaces to $\mathbb{R}^4$. From the dual representation the generalization to hypersurfaces in $\mathbb{R}^4$ is evident, see Section 2.1. Interesting

questions occur when determining two-dimensional rational surfaces Φ in $\mathbb{R}^4$ with similar properties. We characterize the class of rational surfaces Φ in $\mathbb{R}^4$ which satisfy the property that for all given 3-spaces E the surface parameters of Φ can be expressed in terms of *rational functions* of the coefficients of E . This implies that for all 3-spaces E there exist a unique tangent plane T parallel to E with a unique point of contact $\mathbf{p} \in \Phi$. The proposed construction presented in Section 3 circumvents the integration and leads to explicit parameterizations and geometric characterizations of these surfaces Φ . Quadratically parameterized surfaces in $\mathbb{R}^4$ occur as special cases.

The motivation for this research is based on the following relation: Considering $\mathbb{R}^4$ as model space of oriented spheres in $\mathbb{R}^3$, it has been shown in [7] that quadratically parameterized surfaces in $\mathbb{R}^4$ correspond to two-parameter families of spheres in $\mathbb{R}^3$ whose envelope surfaces and their offsets are rational surfaces. A criterion for all parameterized surfaces in $\mathbb{R}^4$ whose corresponding two-parameter families of spheres have envelopes which admit rational parameterizations has been given in [4].

As already indicated in [7], the proposed construction to obtain rational parameterizations of the envelopes of the two-parameter families of spheres can not only be performed with quadratically parameterized surfaces but with a much larger class of surfaces in $\mathbb{R}^4$. These surfaces are investigated here without taking into account these relations to sphere geometry but only considering the generalization of the concept of LN-surfaces to $\mathbb{R}^4$.

The paper is organized as follows: Section 2 discusses some necessary facts about lines in 3-space, and presents the concept of LN-surfaces. Section 3 discusses the different cases of surfaces and gives explicit parameterizations. Some examples illustrate the method. In Section 4 we conclude the article and give hints to possible applications.

2. GEOMETRIC BACKGROUND

Points in $\mathbb{R}^n$ are represented by their inhomogeneous coordinate vectors $\mathbf{x} = (x_1, \dots, x_n)$. The projective closure of $\mathbb{R}^n$ is denoted by $\mathbb{P}^n$. Points in $\mathbb{P}^n$ are identified with their *homogeneous coordinate vectors*

$$\mathbf{y}\mathbb{R} = (y_0, y_1, \dots, y_n)\mathbb{R}, \text{ with } \mathbf{y} \neq \mathbf{o}. \quad (1)$$

Let $\omega : y_0 = 0$ be the hyperplane at infinity in $\mathbb{P}^n$. The interchange between homogeneous and Cartesian coordinates for points in $\mathbb{R}^n$ is realized by

$$x_1 = \frac{y_1}{y_0}, x_2 = \frac{y_2}{y_0}, \dots, x_n = \frac{y_n}{y_0}. \quad (2)$$

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Moreover, let $\mathbb{P}^{n*}$ be the dual projective space, whose points are identified with the hyperplanes in $\mathbb{P}^n$. A hyperplane $E : e_0y_0 + e_1y_1 + \dots + e_ny_n = 0$ in $\mathbb{P}^n$ is identified with the homogeneous coordinate vector $\mathbf{e}\mathbb{R} = (e_0, \dots, e_n)\mathbb{R}$.

2.1 LN-curves and LN-surfaces

A rational curve $C \subset \mathbb{R}^2$ is called an LN-curve if its tangent lines admit the representation $T(u) : x_1u + x_2 = f(u)$, where $f(u)$ is a rational function. This implies that the curve C possesses the *linear normal vector field* $\mathbf{n}(u) = (u, 1)$. A parameterization $\mathbf{c}(u)$ of C can be computed by intersecting $T \cap \dot{T}$. This leads to

$$\mathbf{c}(u) = (\dot{f}, f - u\dot{f}), \text{ with } \dot{f} = df/du. \quad (3)$$

Analogously LN-surfaces $F \subset \mathbb{R}^3$ are characterized as rational surfaces possessing a linear normal vector field $\mathbf{n}$. By excluding cylinders we can assume that $\mathbf{n} = (u, v, 1)$. The tangent planes $T(u, v)$ of F admit the representation

$$T : x_1u + x_2v + x_3 = f(u, v), \quad (4)$$

where $f(u, v)$ is a bivariate rational function. A parameterization $\mathbf{f}(u, v)$ of F can be computed as intersection $T \cap T_u \cap T_v$. This leads to

$$\mathbf{f}(u, v) = (f_u, f_v, f - uf_u - vf_v), \quad (5)$$

where $f_u = df/du$ and $f_v = df/dv$. These surfaces first occurred in computer aided geometric design in [2]. They have very special properties concerning envelope computation with respect to two-parameter translational motions, see [11]. LN-surfaces F in $\mathbb{R}^3$ are also characterized by the fact that for any plane ε in $\mathbb{R}^3$ the surface parameters of F can be expressed in terms of rational functions of the coefficients of ε . This implies that for all planes ε there exists a unique tangent plane T parallel to ε with a unique point of contact $\mathbf{f} \in F$.

The generalization of this concept to LN-hypersurfaces F in $\mathbb{R}^4$ is straightforward. Prescribing the tangent hyperplanes by

$$T(u, v, w) : x_1u + x_2v + x_3w + x_4 = f(u, v, w), \quad (6)$$

F is parameterized by $\mathbf{f} = (f_u, f_v, f_w, f - uf_u - vf_v - wf_w)$.

2.2 Basic facts on surfaces in 4-space

Let $\Phi \subset \mathbb{R}^4$ be a two-dimensional surface, and let $\mathbf{p} : (u, v) \in \mathbb{R}^2 \rightarrow \mathbb{R}^4$ be a parameterization of Φ . At a regular surface point $\mathbf{p}(u, v)$ the partial derivative vectors $\mathbf{p}_u$ and $\mathbf{p}_v$ determine the tangent plane T at $\mathbf{p}$. Considering $\mathbb{R}^4$ as Euclidean space with the canonical scalar product, there exist two linearly independent vectors $\mathbf{e}(u, v)$ and $\mathbf{f}(u, v)$ which determine the normal plane N at $\mathbf{p}$. These vectors satisfy the relations $\mathbf{e}^T \mathbf{p}_u = \mathbf{e}^T \mathbf{p}_v = 0$ and $\mathbf{f}^T \mathbf{p}_u = \mathbf{f}^T \mathbf{p}_v = 0$. The tangent plane T at $\mathbf{p}$ can be considered as intersection of the 3-spaces $(\mathbf{x} - \mathbf{p})^T \mathbf{e} = 0$ and $(\mathbf{x} - \mathbf{p})^T \mathbf{f} = 0$.

When studying rational surfaces Φ in $\mathbb{R}^4$ it is advantageous to consider the projective extension $\mathbb{P}^4$ of $\mathbb{R}^4$. Let $\omega = \mathbb{P}^4 \setminus \mathbb{R}^4$ be the ideal 3-space. The ideal lines $g = T \cap \omega$ of the tangent planes T of Φ form a rational two-parameter family of lines $\mathcal{G}$. Likewise the ideal lines $h = N \cap \omega$ of the normal planes N of Φ form a rational two-parameter family of lines $\mathcal{H}$. The rational surfaces which will be constructed in Sect. 3 are characterized by special properties of the families $\mathcal{G}$ and $\mathcal{H}$ in ω . Therefore we point to some facts from

the geometry of lines in $\mathbb{P}^3$. For more details on this topic we refer to [9].

2.3 Lines in projective three-space

In order to introduce coordinates for lines in $\mathbb{P}^3$, let a line g be spanned by two different points $P = \mathbf{p}\mathbb{R}$ and $Q = \mathbf{q}\mathbb{R}$, with $\mathbf{p}$ and $\mathbf{q}$ in $\mathbb{R}^4$. The *Plücker coordinates* $\mathbf{G} = (g_1, \dots, g_6)$ of g are defined by

$$\mathbf{G} = \begin{pmatrix} p_0q_1 - p_1q_0, p_0q_2 - p_2q_0, p_0q_3 - p_3q_0, \\ p_2q_3 - p_3q_2, p_3q_1 - p_1q_3, p_1q_2 - p_2q_1 \end{pmatrix}. \quad (7)$$

The coordinates g_i are homogeneous and independent of the choice of the points P and Q . Thus, they can be interpreted as the coordinates of points $\mathbf{G}\mathbb{R} = (g_1, \dots, g_6)\mathbb{R}$ in $\mathbb{P}^5$. The coordinates g_i are not independent but satisfy the *Plücker relation*

$$g_1g_4 + g_2g_5 + g_3g_6 = 0. \quad (8)$$

A line g in $\mathbb{P}^3$ can also be considered as intersection of two planes ε and φ . Let these planes be given by their homogeneous coordinate vectors $\mathbf{e} = (e_0, e_1, e_2, e_3)\mathbb{R}$ and $\mathbf{f} = (f_0, f_1, f_2, f_3)\mathbb{R}$. The Plücker coordinates $\mathbf{G} = (g_1, \dots, g_6)$ of $g = \varepsilon \cap \varphi$ are computed by

$$\mathbf{G} = \begin{pmatrix} e_2f_3 - e_3f_2, e_3f_1 - e_1f_3, e_1f_2 - e_2f_1, \\ e_0f_1 - e_1f_0, e_0f_2 - e_2f_0, e_0f_3 - e_3f_0 \end{pmatrix}. \quad (9)$$

2.4 Special two-parameter families of lines

We study rational two-parameter families of lines $\mathcal{G}$ with the property that for almost all planes ε in $\mathbb{P}^3$ there exists a unique line $g \in \mathcal{G}$ with $g \subset \varepsilon$. Applying a duality or polarity δ in $\mathbb{P}^3$, the family of lines $\mathcal{G}$ is mapped to a family of lines $\mathcal{H} = \delta(\mathcal{G})$. The family $\mathcal{H}$ has the property that for almost all points $X \in \mathbb{P}^3$ there exists a unique line $h \in \mathcal{H}$ with $X \in h$. Algebraic families of lines of this type have an *exceptional set*, which means that there exists at most a one-parameter family of points where the line $h \in \mathcal{H}$ is not unique.

Since it is more intuitive in some cases we describe rational two-parameter families of lines $\mathcal{H}$ sending a unique line through a generic point. These families are called *congruences of lines* of degree one and class n , denoted as $(1, n)$ -congruences. The degree denotes the number of lines passing through a generic point, and the class denotes the number of lines lying in a generic plane.

It is a result of classical algebraic line geometry [10], that, besides the star of lines, there exist the following types:

- Chordal variety of a spatial cubic: $(1, 3)$ -congruence.
- $(1, n)$ -congruences of the first kind: there exist two different singular curves.
- $(1, n)$ -congruences of the second kind: there exists only one singular curve which is a line.

This classification holds for families of lines in complex projective space $\mathbb{P}^3$. Since we have to focus on families of real lines, we describe seven types of families of lines. Besides the star of lines (type 6) and the chordal variety (type 5) there exist two linear line congruences (type 1 and type 3) with different singular curves and one linear line congruence (type 2) with only one singular curve for $n = 1$. For types 1–3 the families $\mathcal{G}$ and $\mathcal{H}$ are of the same type. Type 4 describes $(1, n)$ -congruences of the first kind are given in type 4. Analogously, type 7 deals with $(1, n)$ -congruences of the second kind.

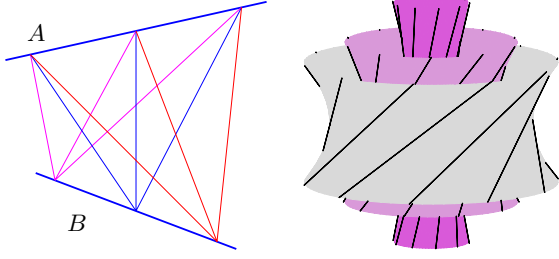


Figure 1: Line congruences of types 1 and 3.

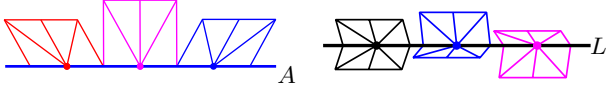


Figure 2: Line congruences of types 2 and 7.

Type 1 – hyperbolic linear line congruence: This family of lines in $\mathbb{P}^3$ consists of those lines intersecting two real skew lines A and B which are called the axes of the congruence. The exceptional set consists of A and B , considered as point set for $\mathcal{H}$ and as pencils of planes for $\mathcal{G}$.

Type 2 – parabolic linear line congruence: It consists of a one-parameter family of pencils of lines in planes through the *axis* A and with vertices on A . The correspondence between the carrier planes and the vertices of the pencils is a projective mapping. The exceptional set consists of the axis A , considered as point set for $\mathcal{H}$ and as pencil of planes for $\mathcal{G}$.

Type 3 – elliptic linear line congruence: This is a family of real lines in $\mathbb{P}^3$ which intersect a pair of skew and conjugate complex lines A and $\bar{A}$. Some authors use the notation *spread* for this family. The exceptional set does not contain real points or planes.

Type 4 – $(1, n)$ -congruence of the first kind: Let C be an algebraic space curve of degree n and let L be a line intersecting C in $n-1$ points. The family $\mathcal{H}$ comprises the lines intersecting both C and L . For any generic point $X \notin C, L$ there exists a plane $X \vee L$ intersecting C in a further point Y and XY is the unique line of the family passing through X . The exceptional set consists of C and L .

Type 5 – chordal variety: This family $\mathcal{H}$ of lines consists of the chords of a spatial cubic C in $\mathbb{P}^3$. It contains also tangent lines of C and lines connecting two conjugate complex points of C . The points of the cubic C form the exceptional set.

Type 6 – star of lines: This family $\mathcal{H}$ consists of the lines through a fixed point P . It sends a unique line through any point $X \neq P$, and P is the exceptional set. The dual family $\mathcal{G}$ is called *ruled plane* and is formed by all lines lying in a fixed plane π , which is the exceptional set.

Type 7 – $(1, n)$ -congruence of the second kind: Let L be a line. There exists a rational correspondence between the points $X \in L$ and the planes ε through L

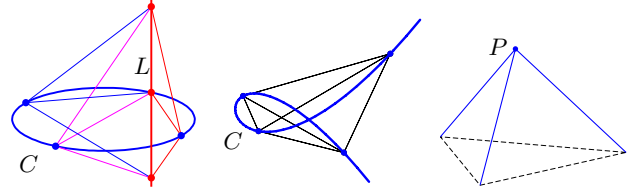


Figure 3: Line congruences of types 4, 5, and 6.

in a way that each point X corresponds to n planes but each plane $\varepsilon \supset L$ corresponds only to one single point. The family consists of pencils of lines with vertices $X \in L$ which lie in planes $\varepsilon \supset L$. The line L is the exceptional set of this family of lines.

For the types 4 and 7 it is not possible to give a complete list. Examples for the families of type 4 for $n = 2$ and $n = 3$ are given in Sect. 3.4. An example for a family of type 7 for $n = 2$ is given in Sect. 3.7.

3. CONSTRUCTION OF THE SURFACES

We present a construction for rational two-dimensional surfaces Φ in $\mathbb{R}^4$ which generalizes LN-surfaces. Let $\mathbf{p}(u, v)$ be a rational parameterization of Φ , and let $E : e_0 + e_1x_1 + e_2x_2 + e_3x_3 + e_4x_4 = 0$ be a 3-space in $\mathbb{R}^4$.

Definition 1. A rational two-dimensional surface Φ in $\mathbb{R}^4$ is called *generalized LN-surface* if for all 3-spaces E the surface parameters u and v can be expressed in terms of rational functions depending on the coefficients e_i of E .

This implies that for all 3-spaces E there exists a unique tangent plane T of Φ which is parallel to E . Although this definition characterizes an affine-invariant class of surfaces, it is convenient to introduce the orthogonality in $\mathbb{R}^4$ determined by the canonical scalar product $\mathbf{x}^T \mathbf{y} = \sum_i x_i y_i$.

Let $\mathbb{P}^4$ be the projective extension of $\mathbb{R}^4$, and let $\omega = \mathbb{P}^4 \setminus \mathbb{R}^4$ be the ideal hyperplane (at infinity). The ideal lines $g = T \cap \omega$ of the tangent planes T of Φ form the family $\mathcal{G}$. Let $\varepsilon = E \cap \omega$ be the ideal plane of a 3-space E . According to Def. 1, these rational surfaces Φ are characterized by the fact that all planes $\varepsilon \in \omega$ carry a unique ideal line $g \in \mathcal{G}$. A duality δ in ω maps planes ε to points X and the family $\mathcal{G}$ to a family of lines $\mathcal{H} = \delta(\mathcal{G})$. If any plane ε carries a unique line $g \in \mathcal{G}$ then the family $\mathcal{H}$ sends a unique line $h \in \mathcal{H}$ through any point X .

Without loss of generality we may choose δ as polarity with respect to the quadric $y_1^2 + y_2^2 + y_3^2 + y_4^2 = 0$. This implies that the canonical scalar product in $\mathbb{R}^4$ is induced by δ . Therefore $\mathcal{H}$ is considered as the family of ideal lines $h = N \cap \omega$ of the normal planes N of Φ . The construction of the surfaces uses the classification of $(1, n)$ -congruences from [10]. For each type of congruence of degree one we obtain an affine-invariant family of generalized LN-surfaces. Besides the surfaces from Sect. 3.6 we restrict our interest to surfaces Φ which span $\mathbb{R}^4$.

The proposed construction interprets a surface Φ in $\mathbb{R}^4$ as envelope of its tangent planes. Each plane $T(u, v)$ is considered as intersection of the 3-spaces

$$\begin{aligned} E(u, v) : \mathbf{e}(u, v)^T \mathbf{x} &= a(u, v), \\ F(u, v) : \mathbf{f}(u, v)^T \mathbf{x} &= b(u, v), \end{aligned} \quad (10)$$

where $\mathbf{e}(u, v)$ and $\mathbf{f}(u, v)$ can be considered as normal vector fields of Φ and $a(u, v)$ and $b(u, v)$ are rational functions. The ideal lines g of T are represented as intersection lines $g = \varepsilon \cap \varphi$ of the ideal planes $\varepsilon = E \cap \omega$ and $\varphi = F \cap \omega$ of E and F . Using $\mathbf{x}$ as homogeneous coordinates in ω , these planes are given by the equations $\varepsilon : \mathbf{e}^T \mathbf{x} = 0$ and $\varphi : \mathbf{f}^T \mathbf{x} = 0$. The question arises, which conditions have to be fulfilled by the functions $\mathbf{e}, \mathbf{f}$ and a, b such that the family of planes $T(u, v)$ possesses an envelope surface?

COROLLARY 1. *Let $T = E \cap F$ be a rational two-parameter family of planes in $\mathbb{R}^4$, where E and F are rational two-parameter families of hyperplanes. The planes T possess a rational envelope surface Φ if and only if the following system of linear equations has a solution,*

$$\begin{aligned} E : \quad \mathbf{e}^T \mathbf{x} &= a, & F : \quad \mathbf{f}^T \mathbf{x} &= b, \\ E_u : \quad \mathbf{e}_u^T \mathbf{x} &= a_u, & F_u : \quad \mathbf{f}_u^T \mathbf{x} &= b_u, \\ E_v : \quad \mathbf{e}_v^T \mathbf{x} &= a_v, & F_v : \quad \mathbf{f}_v^T \mathbf{x} &= b_v. \end{aligned} \quad (11)$$

PROOF. Let $\mathbf{p}(u, v)$ be a solution of the system (11). We have to prove that $T = E \cap F$ is the tangent plane at a regular point $\mathbf{p}(u, v)$. Since T is parameterized by $\mathbf{p} + s\mathbf{p}_u + t\mathbf{p}_v$, this is equivalent to verify the relations

$$\mathbf{e}^T \mathbf{p}_u = 0, \mathbf{e}^T \mathbf{p}_v = 0 \text{ and } \mathbf{f}^T \mathbf{p}_u = 0, \mathbf{f}^T \mathbf{p}_v = 0.$$

Differentiating $\mathbf{e}^T \mathbf{p} = a$ with respect to u and v and taking $\mathbf{e}_u^T \mathbf{p} = a_u$, and $\mathbf{e}_v^T \mathbf{p} = a_v$ into account, leads to $\mathbf{e}^T \mathbf{p}_u = 0$ and $\mathbf{e}^T \mathbf{p}_v = 0$. Analogously we proceed for $\mathbf{f}$, which concludes the proof. $\square$

If the rank of the coefficient matrix of the system (11) is less than four, the solution may degenerate. Since the normal forms of vectors $\mathbf{e}$ and $\mathbf{f}$ we use later always lead to matrices of rank four, we do not consider degenerate cases here.

Generalized LN-surfaces Φ according to Def. 1 can be characterized equivalently in the following way. Let $W : \mathbf{w}^T \mathbf{x} = c$ be an arbitrary 3-space in $\mathbb{R}^4$. The surface Φ is a generalized LN-surface if and only if it possesses a parameterization $\mathbf{p}(u, v)$ in a way that the equations

$$\mathbf{w}^T \mathbf{p}_u(u, v) = 0 \text{ and } \mathbf{w}^T \mathbf{p}_v(u, v) = 0 \quad (12)$$

possess rational solutions $u = \alpha(\mathbf{w})$, $v = \beta(\mathbf{w})$ depending on the coordinates w_i of $\mathbf{w} \in \mathbb{R}^4$.

Remark 1. From the parameterizations $\mathbf{p}(u, v)$ of the seven types of generalized LN-surfaces Φ in $\mathbb{R}^4$ discussed later one arrives at systems of polynomial equations (12). Without taking into account the geometric generation of these surfaces, one would have to use Gröbner basis or other elimination techniques to solve these equations for u and v . The geometric generation (11), however, induces a preferable computation, because we already know that $\mathbf{e}(u, v)$ and $\mathbf{f}(u, v)$ are normal vector fields of Φ . This allows to compute two vector fields $\mathbf{s}(u, v)$ and $\mathbf{t}(u, v)$ spanning the tangent spaces of Φ with methods from linear algebra. Substituting $\mathbf{p}_u$ and $\mathbf{p}_v$ by $\mathbf{s}$ and $\mathbf{t}$, equations (12) simplify. Moreover, the construction itself proves the rationality of the solutions for u and v .

Each of the types of rational families of lines from section 2.4 leads to a family of generalized LN-surfaces Φ in $\mathbb{R}^4$, according to Def. 1. In most cases these surfaces are studied by using special normal forms for functions $\mathbf{e}$ and

$\mathbf{f}$, and conditions to the functions a and b for the existence of envelope surfaces Φ are determined. This is no restriction since the construction of the surfaces is invariant with respect to affine transformations in $\mathbb{R}^4$. In most cases these surfaces admit simple geometric generations and relations to LN-curves and LN-surfaces will appear.

3.1 Surfaces of type 1

If $\mathcal{H}$ is a hyperbolic linear line congruence, the family $\mathcal{G}$ is of the same type. The Plücker coordinates of the axes A and B of $\mathcal{G}$ are chosen as $\mathbf{A} = (0, 0, 0, 0, 1, 0)$ and $\mathbf{B} = (0, 1, 0, 0, 0, 0)$. The two pencils of planes $\varepsilon(u)$ and $\varphi(v)$ passing through A and B are parameterized by

$$\mathbf{e}(u) = (1, 0, u, 0) \text{ and } \mathbf{f}(v) = (0, 1, 0, v).$$

The tangent planes T of Φ have ideal lines $g = \varepsilon \cap \varphi$, and its Plücker coordinates are $\mathbf{G} = (uv, 0, -u, 1, 0, v)$. Since $\mathbf{e}(u)$ and $\mathbf{f}(v)$ are univariate polynomials, $a(u)$ and $b(v)$ have to be chosen as univariate rational functions. They satisfy

$$a_v = 0 \text{ and } b_u = 0. \quad (13)$$

Vectors $\mathbf{e}$ and $\mathbf{f}$ define two independent normal vector fields of Φ . A rational parameterization of Φ is obtained as solution of the system (11), and reads

$$\mathbf{p}(u, v) = (a - ua_u, b - vb_v, a_u, b_v). \quad (14)$$

Thus a surface Φ of type 1 is a translational surface $\mathbf{p}(u, v) = \mathbf{c}(u) + \mathbf{d}(v)$ with profile curves C and D ,

$$\begin{aligned} \mathbf{c}(u) &= (a - ua_u, 0, a_u, 0), \\ \mathbf{d}(v) &= (0, b - vb_v, 0, b_v). \end{aligned} \quad (15)$$

Comparison with (3) shows that C and D are LN-curves. The tangent planes T of Φ are spanned by $\mathbf{p}$ and the vectors $\mathbf{s} = (-u, 0, 1, 0)$ and $\mathbf{t} = (0, -v, 0, 1)$. The partial derivatives can be expressed by

$$\mathbf{p}_u = a_{uu}\mathbf{s} \text{ and } \mathbf{p}_v = b_{vv}\mathbf{t}.$$

According to (12) the equations $\mathbf{w}^T \mathbf{s} = 0$ and $\mathbf{w}^T \mathbf{t} = 0$ result in the rational expressions

$$u = \frac{w_3}{w_1} \text{ and } v = \frac{w_4}{w_2}. \quad (16)$$

THEOREM 1. *A generalized LN-surface Φ in $\mathbb{R}^4$ of type 1 is a translational surface $\mathbf{p}(u, v) = \mathbf{c}(u) + \mathbf{d}(v)$ with planar LN-curves as profile curves. Conversely, any translational surface Φ which spans $\mathbb{R}^4$ and whose profile curves are LN-curves is a surface of type 1.*

PROOF. We only have to show that a translational surface F which spans $\mathbb{R}^4$ obtained by translating LN-curves C and D is equivalent to (14). Since C and D shall span $\mathbb{R}^4$, we may choose an affine coordinate system in a way that C and D are parameterized by (15). Since F is parameterized by $\mathbf{f}(u, v) = \mathbf{c}(u) + \mathbf{d}(v)$, the statement holds. $\square$

Example 1. Choosing quadratic polynomials $a(u) = 1/2u^2$ and $b(v) = 1/2v^2$, the surface of type 1 is a quadratically parameterized surface $\mathbf{p} = (-1/2u^2, -1/2v^2, u, v)$, see [8]. Fig. 4 illustrates the projection $\mathbf{p} = 1/2(-u^2, -v^2, u + v)$ in $\mathbb{R}^3$.

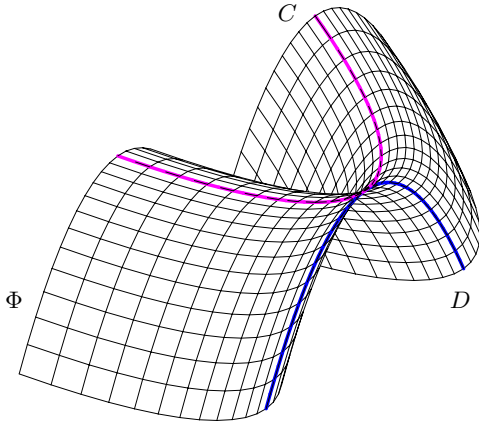


Figure 4: Projection of a surface of type 1 into $\mathbb{R}^3$.

3.2 Surfaces of type 2

If $\mathcal{H}$ is a parabolic linear line congruence, the family $\mathcal{G}$ is of the same type. The lines $g(u, v)$ of $\mathcal{G}$ are the intersection lines of the pencil of planes $\varepsilon(u)$ through the axis A of $\mathcal{G}$ and an appropriate star of planes $\varphi(u, v)$ passing through a vertex $\notin A$. We choose

$$\mathbf{e}(u) = (1, 0, u, 0) \text{ and } \mathbf{f}(u, v) = (0, -1, v, u).$$

The family $\mathcal{G}$ consists of the pencils of lines with vertices $(0, 1, 0, u)\mathbb{R}$ lying in planes $\varepsilon(u)$. The Plücker coordinates of the lines $g(u, v)$ are $\mathbf{G} = (u^2, 0, u, -1, v, u)$. Since $\mathbf{e}_v = (0, 0, 0, 0)$ and $\mathbf{e}_u = \mathbf{f}_v$ holds, the rational functions $a(u)$ and $b(u, v)$ have to satisfy the relations

$$a_v = 0 \text{ and } a_u - b_v = 0. \quad (17)$$

This implies $b(u, v) = va_u - \lambda(u)$. Solving (11), the surface Φ is parameterized by

$$\mathbf{p}(u, v) = (a - ua_u, ub_u + \lambda, a_u, b_u).$$

The tangent planes T of Φ are spanned by $\mathbf{p}$ and the vectors $\mathbf{s} = (-u, v, 1, 0)$ and $\mathbf{t} = (0, u, 0, 1)$. The partial derivatives can be expressed by

$$\mathbf{p}_v = a_{uu}\mathbf{t} \text{ and } \mathbf{p}_u = a_{uu}\mathbf{s} - (\lambda_{uu} - va_{uuu})\mathbf{t}.$$

According to (12) the equations $\mathbf{w}^T\mathbf{s} = 0$ and $\mathbf{w}^T\mathbf{t} = 0$ result in the rational expressions

$$u = -\frac{w_4}{w_2} \text{ and } v = -\frac{w_1w_4 + w_2w_3}{w_2^2}. \quad (18)$$

Since $b_u = va_{uu} - \lambda_u$ holds, $\mathbf{p}(u, v)$ is linear in v and Φ is a ruled surface. Letting $\tilde{v} = b_u$, one obtains the ruled surface parameterization

$$\begin{aligned} \mathbf{p}(u, v) &= \mathbf{c}(u) + \tilde{v}\mathbf{d}(u) \\ &= (a - ua_u, \lambda, a_u, 0) + \tilde{v}(0, u, 0, 1). \end{aligned} \quad (19)$$

The directrix curve $\mathbf{c}(u)$ is a rational curve on a cylinder over the LN-curve $(a - ua_u, 0, a_u)$, and $\mathbf{d}(u)$ is a linear direction vector field along $\mathbf{c}$. The reparameterization $\tilde{v} = b_u$ has influence on the parameterization of $\mathcal{G}$. However, for the geometric properties of surfaces of type 2 this is not relevant.

THEOREM 2. *A generalized LN-surface Φ in $\mathbb{R}^4$ of type 2 is a ruled surface with a rational curve C on a cylinder over an LN-curve as directrix curve and a linear direction*

vector field. Conversely, any ruled surface which spans $\mathbb{R}^4$ and possesses these properties is a surface of type 2.

PROOF. To prove the converse statement one only has to choose an appropriate affine coordinate system and obtains the parameterization (19). $\square$

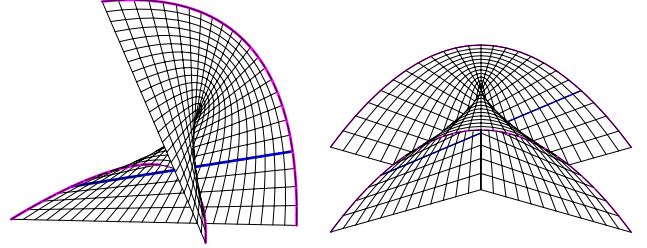


Figure 5: Cayley's surface(left) and Whitney's umbrella(right)

Example 2. For $a(u) = 1/2u^2$ we obtain the surface $\mathbf{f}(u, v) = (-1/2u^2, uv, u, v)$. It can also be represented as quadratically parameterized surface, see [8]. The projection onto $x_4 = 0$ in direction of $(1, 0, 0, -1)$ is recognized as *Cayley's surface* with parameterization $\tilde{\mathbf{f}}(u, v) = (-1/2u^2 + v, uv, u)$. The orthogonal projection onto $x_3 = 0$ is known as *Whitney's umbrella* or *Plücker's conoid* with parameterization $\tilde{\mathbf{f}}(u, v) = (-1/2u^2, uv, v)$.

3.3 Surfaces of type 3

If $\mathcal{H}$ is an elliptic linear line congruence, the family $\mathcal{G}$ is of the same type. In order to describe $\mathcal{G}$, let α be a projective mapping between two stars of planes $\varepsilon(u, v)$ and $\varphi(u, v)$. Their vertices are chosen as $V_e = (0, 1, 0, 0)\mathbb{R}$ and $V_f = (1, 0, 0, 0)\mathbb{R}$, and the planes ε and φ can be determined by

$$\mathbf{e}(u, v) = (1, 0, -u, v) \text{ and } \mathbf{f}(u, v) = (0, -1, v, u),$$

Their intersection lines $g(u, v) \in \mathcal{G}$ are given by the Plücker coordinates $\mathbf{G} = (u^2 + v^2, v, u, 1, -v, -u)$. The projective mapping $\alpha : \varepsilon \mapsto \varphi$ maps the pencil of planes through $V_e V_f$ onto itself, $\eta(t) = (0, 0, t, 1)\mathbb{R} \mapsto \alpha(\eta(t)) = (0, 0, 1, -t)\mathbb{R}$. Since α restricted to the pencil $\eta(t)$ possesses two conjugate complex fixed planes $(0, 0, i, 1)\mathbb{R}$ and $(0, 0, 1, -i)\mathbb{R}$, the family $\mathcal{G}$ of lines $g = \varepsilon \cap \varphi$ is an elliptic linear line congruence.

Because of $\mathbf{e}_u = -\mathbf{f}_v$ and $\mathbf{e}_v = \mathbf{f}_u$, a parameterization $\mathbf{p}(u, v)$ of a surface Φ of type 3 is obtained as solution of (11) for any choice of rational functions $a(u, v)$ and $b(u, v)$ which satisfy

$$a_u = -b_v \text{ and } a_v = b_u. \quad (20)$$

Thus a and b are harmonic conjugate and b is the real part and a is the imaginary part of a univariate polynomial or rational function in the complex variable $z = u + iv$. A rational parameterization of Φ reads

$$\mathbf{p}(u, v) = (a - ua_u - va_v, -b - va_u + ua_v, -a_u, a_v). \quad (21)$$

We show that a surface Φ of type 3 is a translational surface with two conjugate complex profile curves C and $\overline{C}$ with parameterizations $\mathbf{c}(z)$ and $\overline{\mathbf{c}(\overline{z})}$, respectively. Let $f(z)$ be

a rational function in the complex variable z with imaginary part $a(u, v)$ and real part $b(u, v)$. Thus $f = b + ia$. Differentiating f with respect to z and $\bar{z}$ gives

$$\frac{df}{dz} = f_z = a_v + ia_u \text{ and } \frac{df}{d\bar{z}} = 0.$$

It is not difficult to verify that Φ can be represented by

$$\mathbf{p}(u, v) = \frac{1}{2}(\mathbf{c}(z) + \overline{\mathbf{c}(z)}), \text{ with } \mathbf{c}(z) = (i(zf_z - f), (zf_z - f), if_z, f_z), \quad (22)$$

where $\mathbf{c}(z)$ is a planar curve. Its carrier plane is given by $ix_1 + x_2 = ix_3 + x_4 = 0$. Since the tangent lines of C have direction vectors $d\mathbf{c}/dz = f_{zz}(iz, z, i, 1)$, the curves C and $\bar{C}$ are a pair of planar conjugate complex LN-curves. The tangent planes T of Φ are spanned by $\mathbf{p}$ and the vectors $\mathbf{s} = (u, v, 1, 0)$ and $\mathbf{t} = (-v, u, 0, 1)$. The partial derivatives can be expressed by

$$\mathbf{p}_u = -a_{uu}\mathbf{s} + b_{uu}\mathbf{t} \text{ and } \mathbf{p}_v = b_{vv}\mathbf{s} + a_{vv}\mathbf{t}.$$

According to (12) the equations $\mathbf{w}^T\mathbf{s} = 0$ and $\mathbf{w}^T\mathbf{t} = 0$ result in the system of linear equations for u and v ,

$$\begin{pmatrix} w_1 & w_2 \\ w_2 & -w_1 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -w_3 \\ -w_4 \end{pmatrix}. \quad (23)$$

THEOREM 3. *A generalized LN-surface Φ in $\mathbb{R}^4$ of type 3 is a translational surface $\mathbf{p}(z) = 1/2(\mathbf{c}(z) + \overline{\mathbf{c}(z)})$ with a pair of planar conjugate complex LN-curves as profile curves. Conversely, any translational surface Φ whose profile curves are a pair of conjugate complex LN-curves which span $\mathbb{R}^4$ is a surface of type 3.*

PROOF. We only have to show that a translational surface F obtained by translating LN-curves C and $\bar{C}$ is equivalent to (21). Since we require that C and $\bar{C}$ shall span $\mathbb{R}^4$, we can choose an affine coordinate system in a way that C is parameterized by $\mathbf{c}(z)$ from (22). Since F is parameterized by $\mathbf{p}(z) = 1/2(\mathbf{c}(z) + \overline{\mathbf{c}(z)})$, the theorem holds. $\square$

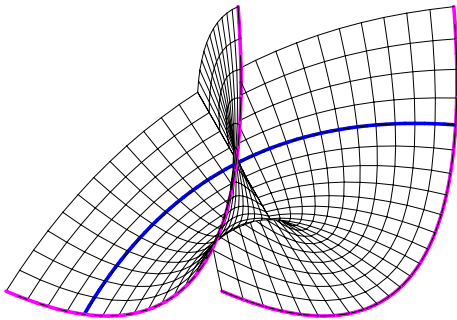


Figure 6: Projection of a surface of type 3.

Example 3. For the choice $a = 1/2(v^2 - u^2)$ and $b = uv$ one obtains the quadratically parameterized surface $\mathbf{p} = (1/2(u^2 - v^2), uv, u, v)$. A projection of this surface onto $x_4 = 0$ is displayed in Fig. 6.

3.4 Surfaces of type 4

Let C be a spatial algebraic curve of degree n in ω which has at least one chord L joining $n - 1$ points of C . The number of points is counted algebraically. The family $\mathcal{H}$ of lines h meeting the curve C and the line L sends a unique line through a generic point $X \in \omega$.

Since there exist more than one family $\mathcal{H}$ for $n > 2$, we only give examples for $n = 2$ and $n = 3$:

- Let C be a conic and let L be a line meeting C at a single point. The family $\mathcal{H}$ consists of all lines meeting both C and L . The polarity maps C to a quadratic cone D , and L to a tangent line M of D .
- Let C be a cubic and let L be a chord of C . The family $\mathcal{H}$ consists of all lines meeting both C and L . The chord L of C can be replaced by a tangent line of C . The polarity maps C to a developable surface D of class three, and L to a line M carrying two tangent planes of D . These two tangent planes are not necessarily real.

The construction of a surface of type 4 is illustrated by the example for $n = 2$.

The planes $\varepsilon(u)$ and $\varphi(v)$ can be represented by

$$\mathbf{e}(u) = (-1, 0, 0, u) \text{ and } \mathbf{f}(v) = (0, u, -v, v^2).$$

The Plücker coordinates of lines $g(u, v) = \varepsilon(u) \cap \varphi(v)$ are $\mathbf{G} = (uv, u^2, 0, -u, v, -v^2)$. The rational functions a and b have to satisfy

$$a_v = 0 \text{ and } ub_u + vb_v - b - v^2a_u = 0. \quad (24)$$

Choosing a as univariate rational function $a(u)$ and with $b(u, v)$ being of the form $b(u, v) = uh(u/v) + v^2/ua(u)$ a parameterization of this particular subclass of surfaces Φ of type 4 reads

$$\mathbf{p}(u, v) = (ua_u - a, b_u, 2va_u - b_v, a_u). \quad (25)$$

The tangent planes T of Φ are spanned by $\mathbf{p}$ and the vectors $\mathbf{s} = (u, 0, v, 1)$ and $\mathbf{t} = (0, v, u, 0)$. The partial derivatives can be expressed by

$$\mathbf{p}_u = a_{uu}\mathbf{s} + \frac{b_{uu}}{v}\mathbf{t} \text{ and } \mathbf{p}_v = \frac{b_{uv}}{v}\mathbf{t}.$$

According to (12) the equations $\mathbf{w}^T\mathbf{s} = 0$ and $\mathbf{w}^T\mathbf{t} = 0$ result in the rational expressions

$$u = \frac{w_2w_4}{w_3^2 - w_1w_2} \text{ and } v = \frac{-w_3w_4}{w_3^2 - w_1w_2}. \quad (26)$$

Example 4. For the choice $a = 1/2u^2$ and $b = 1/2uv^2$ one obtains the quadratically parameterized surface $\mathbf{p} = (1/2u^2, 1/2v^2, uv, u)$. A projection of this surface onto $x_1 = 0$ is the Plücker's conoid as displayed in Fig. 5. Another projection of this surface reads $(p_1 - p_2, p_3, p_4)$, which is displayed in Fig. 6.

3.5 Surfaces of type 5

Let $\mathcal{H}$ be the chordal variety of a spatial cubic C . Applying the polarity δ maps $\mathcal{H}$ to the axes variety $\mathcal{G}$ of a developable surface D of class 3. The family $\mathcal{G}$ consists of the real intersection lines of two tangent planes of D . These planes can also be conjugate complex. In addition the generating lines of D belong to $\mathcal{G}$.

Let $\varepsilon(t)$ be the real tangent planes of D . The intersection lines $g(u, v) = \varepsilon(u) \cap \varepsilon(v)$ form a subset of $\mathcal{G}$. This parameterization however has the disadvantage that each line g is obtained for two different parameter values (u_1, v_1) and (v_1, u_1) . Consequently the uniqueness property does not hold for this representation. This can be avoided by using proper parameterizations of the axes variety $\mathcal{G}$.

The lines $g \in \mathcal{G}$ can be obtained by joining points $\mathbf{x}\mathbb{R} = (u, 0, v, 1)\mathbb{R}$ and $\mathbf{y}\mathbb{R} = (1, v, u, 0)\mathbb{R}$. Because of the linearity in u and v , the correspondence $\alpha : \mathbf{x}\mathbb{R} \mapsto \mathbf{y}\mathbb{R}$ is a projective mapping between the planes $y_1 = 0$ and $y_3 = 0$. The Plücker coordinates of lines g are $\mathbf{G} = (uv, u^2 - v, -1, -u, v, -v^2)$. In order to generate the surface Φ in $\mathbb{R}^4$ as envelope of its tangent planes, lines $g \in \mathcal{G}$ are considered as intersection of planes ε and φ . Their coordinate vectors are chosen as

$$\mathbf{e} = (u, 0, -1, -u^2 + v) \text{ and } \mathbf{f} = (-v, 1, 0, uv). \quad (27)$$

By investigating the equations (11), we realize that the system is solvable if and only if the functions $a(u, v)$ and $b(u, v)$ satisfy the conditions

$$b_u = va_v \text{ and } b_v = -a_u - ua_v. \quad (28)$$

This system is integrable if the function $a(u, v)$ satisfies

$$a_{uu} = -ua_{uv} - 2a_v - va_{vv}. \quad (29)$$

A solution for $b(u, v)$ follows by

$$b = \int va_v du - \int (va_{vv} + a_v) dv + a_u + ua_v dv + C. \quad (30)$$

Reducing our interest to polynomial solutions, the integration of $b(u, v)$ is not problematic. The condition (29) for the function $a(u, v)$ can be satisfied with a polynomial ansatz. The corresponding parameterization of a surface Φ in $\mathbb{R}^4$ of type 5 is

$$\mathbf{p}(u, v) = (ua_v - b_v, b - vb_v, -a + b_u - ub_v, a_v). \quad (31)$$

This representation depends on the choice of $\mathbf{e}$ and $\mathbf{f}$ which determine the parameterization of the axes variety $\mathcal{G}$. Different parameterizations typically lead to different representations for surfaces Φ of type 5. The tangent planes T of Φ are spanned by $\mathbf{p}$ and the vectors $\mathbf{s} = (u, 0, v, 1)$ and $\mathbf{t} = (1, v, u, 0)$. The partial derivatives can be expressed by

$$\mathbf{p}_u = a_{uv}\mathbf{s} - va_{vv}\mathbf{t} \text{ and } \mathbf{p}_v = a_{vv}\mathbf{s} - b_{vv}\mathbf{t}.$$

According to (12) the equations $\mathbf{w}^T \mathbf{s} = 0$ and $\mathbf{w}^T \mathbf{t} = 0$ result in the system of linear equations for u and v ,

$$\begin{pmatrix} w_1 & w_3 \\ w_3 & w_2 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -w_4 \\ -w_1 \end{pmatrix}. \quad (32)$$

THEOREM 4. *A generalized LN-surface Φ in $\mathbb{R}^4$ of type 5 admits explicit rational parameterizations. The tangent planes of Φ have linearly parameterized direction vectors $\mathbf{s} = (u, 0, v, 1)$ and $\mathbf{t} = (1, v, u, 0)$.*

Example 5. Performing a polynomial ansatz for $a(u, v)$ of degree three, a general solution of (28) for a and b depending on the coefficients c_i reads

$$\begin{aligned} a(u, v) &= c_4 + c_3u - c_2u^2 - 1/2c_1u^3 + c_2v + c_1uv, \\ b(u, v) &= c_2uv + 1/2c_1u^2v - c_3v - 1/2c_1v^2. \end{aligned}$$

The coefficients $(c_1, \dots, c_4) = (0, 0, 0, 1)$ give the polynomials $a(u, v) = -1/2u^3 + uv$ and $b(u, v) = 1/2u^2v - 1/2v^2$. Finally the surface Φ of type 5 is a quadratically parameterized

surface, parameterized by $\mathbf{p}(u, v) = (1/2u^2 + v, 1/2v^2, uv, u)$. Fig. 7 displays a projection onto $x_4 = 0$.

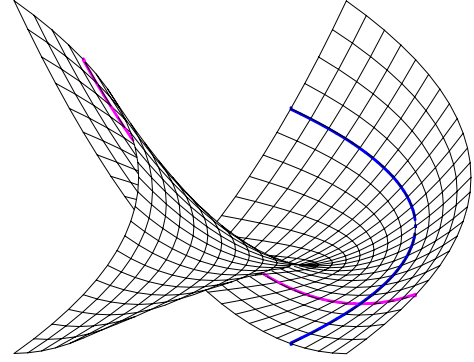


Figure 7: Projection of a surface of type 5.

3.6 Surfaces of type 6

If $\mathcal{H}$ is a star of lines, $\mathcal{G}$ is a ruled plane. The ideal lines $g(u, v)$ of the tangent planes $T(u, v)$ of a surface Φ of type 6 are obtained by intersecting the planes of a star $\varepsilon(u, v)$ with the fixed carrier plane φ of $\mathcal{G}$. We choose

$$\mathbf{e}(u, v) = (1, u, v, 0) \text{ and } \mathbf{f} = (0, 0, 0, 1)$$

as parameterizations of ε and φ , respectively and an arbitrary bivariate rational function $a(u, v)$ and $b = \text{const.}$ Since $\mathcal{G}$ is a ruled plane, Φ is contained in a 3-space. Solving (11) a rational parameterization of a surface Φ of type 6 reads

$$\mathbf{p}(u, v) = (a - ua_u - va_v, a_u, a_v, b). \quad (33)$$

Comparison with section 2.1 shows that Φ is an LN-surface contained in $x_4 = b$. An example for $a(u, v) = u^3 + v^3$ and $b = 0$ is displayed in Fig. 8. The tangent planes T of Φ are spanned by $\mathbf{p}$ and the vectors $\mathbf{s} = (-u, 1, 0, 0)$ and $\mathbf{t} = (-v, 0, 1, 0)$. The partial derivatives can be expressed by

$$\mathbf{p}_u = a_{uu}\mathbf{s} + a_{uv}\mathbf{t} \text{ and } \mathbf{p}_v = a_{uv}\mathbf{s} + a_{vv}\mathbf{t}.$$

According to (12) the equations $\mathbf{w}^T \mathbf{s} = 0$ and $\mathbf{w}^T \mathbf{t} = 0$ result in the rational expressions

$$u = \frac{w_2}{w_1} \text{ and } v = \frac{w_3}{w_1}. \quad (34)$$

THEOREM 5. *A generalized LN-surface Φ in $\mathbb{R}^4$ of type 6 is an LN-surface in a hyperplane of $\mathbb{R}^4$.*

3.7 Surfaces of type 7

Let $\mathcal{H}$ be a $(1, n)$ -congruence of the second kind. It possesses a singular line L and consists of pencils of lines with vertices $X \in L$ with carrier planes $\varepsilon \supset L$. Thereby each point $X \in L$ corresponds to n planes through L but each plane $\varepsilon \supset L$ corresponds only to one point $X \in L$. The case $n = 1$ is exactly type 2, and we give an example for $n = 2$. To represent planes ε and φ we choose

$$\mathbf{e}(u) = (1 - u^2, 0, 0, 2u) \text{ and } \mathbf{f}(u, v) = (0, 1, u, v).$$

The Plücker coordinates of lines $g = \varepsilon \cap \varphi$ are $\mathbf{G} = (-2u^2, 2u, 0, 1 - u^2, u(1 - u^2), v(1 - u^2))$. Investigating the equations

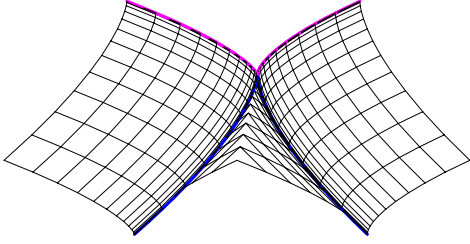


Figure 8: Surface of type 6.

(11), the functions $a(u, v)$ and $b(u, v)$ have to satisfy the conditions

$$a_v = 0 \text{ and } 2b_v(1 + u^2) = a_u(1 - u^2) + 2ua. \quad (35)$$

Thus the solvability of (11) requires a univariate rational function $a(u)$ and b is determined by

$$b(u, v) = \frac{v}{2(1 + u^2)}(a_u(1 - u^2) + 2ua) + \lambda(u).$$

A parameterization $\mathbf{p}$ of Φ is obtained by

$$\mathbf{p}(u, v) = \left(\frac{a - ua_u}{1 + u^2}, b - ub_u - vb_v, b_u, b_v \right). \quad (36)$$

The tangent planes T of Φ are spanned by vectors $\mathbf{s} = (0, -u, 1, 0)$ and $\mathbf{t} = (2u/(u^2 - 1), -v, 0, 1)$. The partial derivatives can be expressed by

$$\mathbf{p}_u = b_{uu}\mathbf{s} + b_{uv}\mathbf{t} \text{ and } \mathbf{p}_v = b_{uv}\mathbf{s}.$$

According to (12) the equations $\mathbf{w}^T \mathbf{s} = 0$ and $\mathbf{w}^T \mathbf{t} = 0$ result in the rational expressions

$$u = \frac{w_3}{w_2} \text{ and } v = \frac{2w_1w_2w_3 + w_4(w_3^2 - w_2^2)}{w_2(w_3^2 - w_2^2)}. \quad (37)$$

Since $b = vb_v + \lambda$ is linear in v , surfaces Φ are ruled surfaces. By letting $\tilde{v} = b_u$ we find the ruled surface parameterization

$$\mathbf{p}(u, v) = \left(\frac{a - ua_u}{1 + u^2}, \lambda, 0, \frac{a_u(1 - u^2) + 2ua}{2(1 + u^2)} \right) + \tilde{v} (0, -u, 1, 0).$$

COROLLARY 2. *A generalized LN-surface Φ in $\mathbb{R}^4$ of type 7 is a ruled surface whose generating lines are parallel to a fixed plane.*

Fig. 9 shows two different projections of the surface for $a = (1 + u^2)^2$ and $\lambda = 0$.

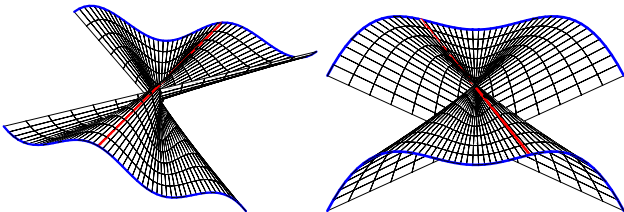


Figure 9: Two projections of a surface of type 7.

4. CONCLUSION

We have presented a class of rational surfaces Φ in $\mathbb{R}^4$ which satisfy the property that for all given vectors $\mathbf{w} \in \mathbb{R}^4$ the surface parameters u, v of Φ can be expressed by rational functions of the coefficients w_i of $\mathbf{w}$. Considering $\mathbb{R}^4$ as model space for the four-parameter family of spheres in $\mathbb{R}^3$, the presented surfaces correspond to two-parameter families of spheres whose envelope surfaces and their offsets admit rational parameterizations. These relations will be elaborated in a separate publication.

Acknowledgments

This work has been funded in parts by the Austrian Science Fund FWF within the research network S92.

5. REFERENCES

- [1] Farouki, R.T., 2002. Pythagorean Hodograph curves, in: *Handbook of Computer Aided Geometric Design*, Farin, G., Hoschek, J., Kim, M.-S. eds.), Elsevier.
- [2] Jüttler, B., 1998. Triangular Bézier surface patches with a linear normal vector field, in: *The Mathematics of Surfaces VIII*, Information Geometers, 431–446.
- [3] Jüttler, B. and Sampoli, M.L., 2000. Hermite interpolation by piecewise polynomial surfaces with rational offsets, *Comp. Aided Geom. Design* **17**, 361–385.
- [4] Kosinka, J. and Jüttler, B., 2007. MOS surfaces: Medial Surface Transforms with Rational Domain Boundaries, in: *The Mathematics of Surfaces XII*, R. Martin, M. Sabin, J. Winkler (eds.), Lecture Notes in Computer Science, Springer, 245–262.
- [5] Krasauskas, R., 2006. Minimal rational parameterization of canal surfaces, *Computing* **79**, 281–290.
- [6] Peternell, M. and Odehnal, B., 2008. Convolution surfaces of quadratic triangular Bézier surfaces, *Comp. Aided Geom. Design* **25**, 116–129.
- [7] Peternell, M., Odehnal, B. and Sampoli, M.L. On quadratic two-parameter families of spheres and their envelope, *Comp. Aided Geom. Design*, in press.
- [8] Peters, J., Reif, U., 1998. The 42 equivalence classes of quadratic surfaces in affine n -space, *Comp. Aided Geom. Design* **15**, 459–473.
- [9] Pottmann, H. and Wallner, J., 2001. *Computational Line Geometry*. Springer-Verlag.
- [10] Sturm, R., 1893. *Die Gebilde ersten und zweiten Grades der Liniengeometrie in synthetischer Behandlung, II. Theil*, Teubner, Leipzig.
- [11] Sampoli, M.L., Peternell, M. and Jüttler, B., 2006. Exact parameterization of convolution surfaces and rational surfaces with linear normals, *Computer Aided Geometric Design* **23**, 179–192.
- [12] Zindler, K., 1921. Algebraische Liniengeometrie, in: *Encykl. d. Math. Wiss.* III, 2.2, 973–1228.

A Rational Decomposition–Lemma for Systems of Linear Differential–Algebraic Equations

Eckhard Pflügel
Kingston University, Faculty of CISM
Penrhyn Road, Kingston upon Thames
Surrey, KT1 2EE, United Kingdom
E.Pfluegel@kingston.ac.uk

ABSTRACT

We give a new decomposition lemma for linear differential–algebraic equations (DAEs). This generalises a result which we have given in [12] for singular linear systems of ordinary differential equations to the class of linear DAEs with formal power series coefficients. The results of this paper are a first step towards a formal reduction algorithm for this type of equations.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms

General Terms

Algorithms

Keywords

Computer Algebra, Linear DAEs, Local Reduction, Splitting Lemma

1. INTRODUCTION

Linear differential-algebraic equations (DAEs) arise frequently in applications, for example in modelling electric circuits [9]. Since early work in this field, it has been acknowledged that symbolic methods are highly appropriate when tackling DAEs. Few symbolic methods have been published specialising in linear DAEs (see [14, 15] and the references therein).

Recent progress in the algorithmic treatment of systems of linear ODEs has resulted in efficient symbolic methods for solving this type of systems (see [11] and references therein). At the heart of these methods is an effective reduction algorithm [13], based on efficient reduction algorithms [7, 4] implementing Moser’s Reduction Principle [10] and extensions of the classical Splitting Lemma [13, 12]. This has led

to implementations in the Computer Algebra system Maple [3].

We would like to extend the methods that are known for linear ODEs to linear DAEs. In this paper, we show that the “Root–Free Splitting Lemma” as introduced in [12] can be generalised to the class of linear differential-algebraic systems with formal power series coefficients. This results in a new decomposition lemma, that includes all previously known variants [13, 12] and is a first step towards a more general reduction algorithm for linear DAEs.

We consider a linear differential-algebraic equation (DAE) of the form

$$A(x)y - B(x)\theta_r(y) = 0 \quad (1)$$

where

$$A = \sum_{j=0}^{\infty} A_j x^j, \quad B = \sum_{j=0}^{\infty} B_j x^j \quad (A_0, B_0 \neq 0)$$

with $A_j, B_j \in \mathbb{C}^{n \times n}$ and $\theta_r = x^{r+1} \frac{d}{dx}$ where r is a positive integer. When studying this type of equations in the context of Computer Algebra, the following questions arise:

- A DAE combines differential and algebraic equations – how can we separate these effectively?
- We would like to find algorithms for computing power series solutions of (1). To our knowledge, there are no specialised algorithms available at present.
- In order to compute such power series solutions in the case of linear systems of ordinary differential equations, the formal reduction is an appropriate algorithmic method. Is it possible to adapt the formal reduction to the class of linear DAEs?

This paper is organised as follows: in Section 2 we give a brief overview of the local analysis of linear systems of ordinary differential equations and we present an important concept, the classical Splitting Lemma, in Section 3. In Section 4 we recall some results about regular matrix pencils that shall be used in the what follows, and establish the link to Linear DAEs. The definition of the *characteristic matrix* in Section 5 has been originally motivated by a paper of Hufford on systems of linear partial differential equations [8], see the final section of this paper for a brief discussion of this. Section 6 states and proves the decomposition lemma, followed by an example in Section 7 and a discussion in the final section.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC’08, July 20–23, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Notations:

Throughout this paper, let $\omega = e^{2\pi i/q}$ be a q th root of unity. We will consider matrices with entries in $C((x^{1/q}))$, the field of q -meromorphic formal power series with ramification index q . Denote by $v_x(a)$ the valuation of an element $a \in C((x^{1/q}))$.

2. LINEAR SYSTEMS OF DIFFERENTIAL EQUATIONS

In this section, we review some results about singular systems of linear differential equations of the form

$$x^{r+1} \frac{dy}{dx} = A(x)y \quad (2)$$

with $r \in \mathbb{N}$ called the *Poincaré-rank* of the system, and

$$A = \sum_{j=0}^{\infty} A_j x^j \quad (A_0 \neq 0)$$

where $A_j \in \mathbb{C}^{n \times n}$. When studying the behaviour of the solutions of this system in the neighbourhood of $x = 0$, one distinguishes between the *regular singular* case ($r = 0$) and the *irregular singular* case ($r > 0$). In the former case, the system admits a formal fundamental matrix which can be computed in a fairly straightforward manner by solving a system of recurrence equations, see for example the descriptions in [1]. The case $r > 0$ requires more work; the approach of formal reduction [13] computes solutions in this case.

Let $T(x)$ be an invertible square matrix depending on the variable x (we will discuss below different types of transformations that are relevant). The change of variable

$$y = Tz \quad (3)$$

transforms the system (2) into a new system

$$x^{r+1} \frac{dz}{dx} = Bz \quad (4)$$

where

$$B = T[A] := T^{-1}AT - x^{r+1}T^{-1} \frac{dT}{dx}. \quad (5)$$

We call the systems (2) and (4) (the matrices A and B respectively) *equivalent*.

We will require different types of transformations. The simplest type is the class of *polynomial transformations*. Albeit simple, they are useful in many applications – most of the algorithms aiming at reducing the Poincaré rank use transformations of this kind [1, 10]. A more general transformation class are *analytic power series*, mainly used in the *classical Splitting Lemma* and its generalisations [13, 12]. The *formal reduction* requires the use of transformations with *meromorphic* and *q-meromorphic* power series coefficients. The particular class of *diagonal (Shearing)* transformations deserves mentioning, as it is often used in classical algorithms, and we will use it as well.

[1] contains a more detailed description and discussion of these different types of transformations. For the purposes of this paper, it is however sufficient to simply distinguish between *rational* and *ramified* transformations. The latter refers to q -meromorphic transformations and the former to

any transformation whose coefficients do not contain fractional powers of x . The integer q is called the *ramification index* of the transformation.

3. CLASSICAL SPLITTING LEMMA

As we have previously mentioned, the Splitting Lemma is a well-known result. Its proof is carried out in a constructive fashion and gives a method for computing the coefficients T_j of a transformation matrix T as in (3). We repeat it here for the sake of completeness. Also, we will formulate it for q -meromorphic systems in preparation of the proof of Theorem 2. Consider a system of the form

$$x \frac{dy}{dx} = \hat{A}(x)y \quad (6)$$

which has a coefficient matrix of the form

$$\hat{A}(x) = x^{-r} \sum_{j=p}^{\infty} \hat{A}_j x^{j/q} \quad (\hat{A}_p \neq 0, p \geq 0) \quad (7)$$

where p is relatively prime to q , $p < q$ and $\hat{A}_p$ has several eigenvalues.

LEMMA 1 ([1, 2, 16]). *Consider the system (6) and assume that $\hat{A}_p$ is block-diagonal*

$$\hat{A}_p = \begin{pmatrix} \hat{A}_p^{11} & 0 \\ 0 & \hat{A}_p^{22} \end{pmatrix}$$

such that

$$\text{spec}(\hat{A}_p^{11}) \cap \text{spec}(\hat{A}_p^{22}) = \emptyset.$$

Then there exists a formal q -meromorphic transformation of the form

$$\hat{T}(x) = \sum_{j=0}^{\infty} \hat{T}_j x^{j/q} \quad (\hat{T}_0 = I)$$

such that the transformed system is block-diagonal with the same block partition as in $\hat{A}_p$.

Although this is a classical result, we repeat the proof here as it prepares the reader for techniques that appear later in this paper, generalising the approach below:

Proof We use a transformation of the special form

$$\hat{T}(x) = \begin{pmatrix} I & \hat{U}(x) \\ \hat{V}(x) & I \end{pmatrix}$$

with $\hat{U}_0 = \hat{V}_0 = 0$. Denote by $\hat{B}$ the matrix $\hat{T}[\hat{A}]$. Inserting the series expansion for $\hat{A}, \hat{B}$ and $\hat{T}$ into the definition of equivalence (5) and comparing coefficients gives the recursion formula for $h > 0$

$$\hat{A}_p \hat{T}_h - \hat{T}_h \hat{A}_p = \sum_{j=1}^h (\hat{T}_{h-j} \hat{B}_{j+p} - \hat{A}_{j+p} \hat{T}_{h-j}) + ((p+h)/q - r) \hat{T}_{p+h-qr} \quad (8)$$

where $\hat{T}_j = 0$ for $j < 0$. Equation (8) is of the form

$$\hat{A}_p \hat{T}_h - \hat{T}_h \hat{A}_p = \hat{B}_{h+p} - \hat{A}_{h+p} + \hat{R}_h \quad (9)$$

where

$$\hat{R}_h = \sum_{j=1}^{h-1} (\hat{T}_{h-j} \hat{B}_{j+p} - \hat{A}_{j+p} \hat{T}_{h-j}) + ((p+h)/q - r) \hat{T}_{p+h-qr}$$

depends only on $\hat{B}_j$ with $j < h + p$ and $\hat{T}_j$ with $j < h$. Imposing the special form of

$$\hat{T}_h = \begin{pmatrix} 0 & \hat{U}_h \\ \hat{V}_h & 0 \end{pmatrix}, \quad \hat{B}_h = \begin{pmatrix} \hat{B}_h^{11} & 0 \\ 0 & \hat{B}_h^{22} \end{pmatrix}$$

and decomposing $\hat{R}_h$ into block-structure accordingly gives the following system of equations:

$$\hat{B}_{p+h}^{11} + \hat{R}_h^{11} = 0, \quad (10)$$

$$\hat{B}_{p+h}^{22} + \hat{R}_h^{22} = 0 \quad (11)$$

where $\hat{B}_{p+h}^{11}$ and $\hat{B}_{p+h}^{22}$ are unknown, and

$$\begin{aligned} \hat{A}_p^{11} \hat{U}_h - \hat{U}_h \hat{A}_p^{22} &= \hat{R}_h^{12}, \\ \hat{A}_p^{11} \hat{V}_h - \hat{V}_h \hat{A}_p^{22} &= \hat{R}_h^{21} \end{aligned} \quad (12)$$

with unknowns $\hat{U}_h$ and $\hat{V}_h$. Given $\hat{R}_h$, the first two equations (10) and (11) can be solved by setting $\hat{B}_{p+h}^{11} = -\hat{R}_h^{11}$ and $\hat{B}_{p+h}^{22} = -\hat{R}_h^{22}$. The remaining equations can be solved uniquely for $\hat{U}_h$ and $\hat{V}_h$ because the matrices $\hat{A}_p^{11}$ and $\hat{A}_p^{22}$ have no eigenvalues in common, see e.g. [6]. $\square$

4. REGULAR MATRIX PENCILS AND LINEAR DAES

In order to generalise the Root-Free Splitting Lemma, we need some results concerning regular matrix pencils. A *matrix pencil* $A - \lambda B$ is a pair of matrices where $A, B \in \mathbb{C}^{m \times n}$ and λ is an indeterminate. A *regular matrix pencil* is a matrix pencil where $m = n$ and there exists $c \in \mathbb{C}$ such that $A - cB$ is invertible. One defines similarity transformations through left- and right multiplication with constant, non-singular matrices: two regular matrix pencils $A - \lambda B$ and $\tilde{A} - \lambda \tilde{B}$ are similar if there exist $S, T \in \text{GL}(n, \mathbb{C})$ such that

$$S(A - \lambda B)T = \tilde{A} - \lambda \tilde{B}.$$

One can define the notion of eigenvalues of regular matrix pencils in the following way: *finite* eigenvalues are the roots of the characteristic polynomial

$$\chi(\lambda) = \det(A - \lambda B) \neq 0.$$

There are also $n - d = n - \deg(\chi)$ eigenvalues *at infinity*. We use the notation $\text{spec}(A - \lambda B)$ for the set of eigenvalues (finite and at infinity) of the regular pencil $A - \lambda B$.

In Section 6, we will use the following result from matrix theory:

THEOREM 1 ([5]). *The “Generalised Sylvester Equation”*

$$(A - \lambda B)X - Y(C - \lambda D) = E - \lambda F$$

has a unique solution (X, Y) for given (A, B, C, D, E, F) if and only if $\text{spec}(A - \lambda B) \cap \text{spec}(C - \lambda D) = \emptyset$.

Similarly as in the case of ordinary linear differential systems, we can consider a change of variable

$$y = Tz$$

and in addition to that, we can multiply a linear DAE from the left with an invertible square matrix S . Using $\theta_r T =$

$\theta_r(T) + T\theta_r$, one can see that this leads to the *equivalent* linear DAE having the operator

$$S(A - B\theta_r)T = (SAT - SB\theta_r(T)) - SBT\theta_r. \quad (13)$$

We can consider for S and T the same types of transformations as explained previously.

In order to understand why matrix pencils arise naturally when analysing linear DAEs, consider the operator $A - B\theta_r$ that corresponds to a linear DAE as in (1). We define the *leading pencil* $A_0 - \lambda B_0$. We then observe the following

LEMMA 2. *If the leading pencil of a linear DAE is regular, then transforming this DAE with a constant transformation leads to an equivalent DAE with similar leading pencil.*

Remark 4.1. It is also clear that the eigenvalues of a regular leading pencil play a crucial role for computing the leading terms of formal solutions.

In [13] we have introduced the concept of k -simple systems and the results presented therein immediately apply to a linear DAE with regular leading pencil. In the next section, we will make a less restrictive assumption and show that it is still possible to find a rational transformation that splits the system into two blocks of smaller size, even if the conditions of [13] are not met.

5. ON THE CHARACTERISTIC MATRIX OF LINEAR DAES

DEFINITION 1. *Consider a linear DAE (1) and let a_{ij} and b_{ij} refer to the entries of A and B . Suppose the existence of vectors of rational numbers $s, t \in \mathbb{Q}^n$ and let q be the least common multiple of the denominators of the entries s_i and t_i of s and t . Assume that*

$$\begin{aligned} \min_{i,j} (v_x(a_{ij}(x)) - s_i + t_j) &=: m \\ \min_{i,j} (v_x(b_{ij}(x)) - s_i + t_j) &= 0 \end{aligned}$$

with $0 \leq m = \frac{p}{q} < r$ and that one has $\gcd(p, q) = 1$. We define the (s, t) -characteristic matrix as the matrix pencil

$$C_{s,t}(\lambda) = ((c_{ij} - \lambda d_{ij}))$$

where c_{ij} is the coefficient of $x^{m+s_i-t_j}$ in $a_{ij}(x)$ and d_{ij} is the coefficient of $x^{s_i-t_j}$ in $b_{ij}(x)$.

We call $C_{s,t}(\lambda)$ *non-trivial*, if $\det(C_{s,t}(\lambda)) \neq 0$.

The following lemma says that the characteristic matrix is the leading pencil of a equivalent linear DAE, using a particular Shearing transformation.

LEMMA 3. *Let $C_{s,t}(\lambda)$ be an (s, t) -characteristic matrix of (1) and*

$$S = \text{diag}(x^{-s_1}, \dots, x^{-s_n}), \quad T = \text{diag}(x^{t_1}, \dots, x^{t_n}).$$

Then, transforming (1) by (S, T) results in

$$\tilde{A}(x)y - \tilde{B}(x)\theta_r(y) = 0 \quad (14)$$

where, with $m = p/q$, we have

$$\tilde{A}(x) = \sum_{j=p}^{\infty} \tilde{A}_j x^{j/q}, \quad \tilde{B}(x) = \sum_{j=0}^{\infty} \tilde{B}_j x^{j/q}$$

and

$$C_{s,t}(\lambda) = \tilde{A}_p - \lambda \tilde{B}_0.$$

Proof We have $\theta_r(T)T^{-1} = x^r \text{diag}(t_1, \dots, t_n)$ hence it follows

$$\tilde{A} = S(A - O(x^r))T = ((\tilde{a}_{ij}))$$

where

$$\begin{aligned} \tilde{a}_{ij} &= a_{ij}^* x^{t_j - s_i + v_x(a_{ij})} + \dots + \sigma_{ij} \\ &= a_{ij}^* x^{m_{ij}} + \dots + \sigma_{ij} \end{aligned}$$

with $\sigma_{ij} \in \mathbb{C}((x))$ all of order $O(x^r)$ and a_{ij}^* is the leading coefficient of the entries of the matrix A , i.e. the coefficient of $x^{v(a_{ij})} = x^{m_{ij} - t_j + s_i}$ in a_{ij} . Set

$$\min_{ij} \{m_{ij}\} =: m = \frac{p}{q}$$

and note $m < r$. Hence, we obtain $\tilde{A}_p = ((b_{ij}^*))$ where

$$b_{ij}^* = \begin{cases} 0 & \text{if } m_{ij} > m \\ a_{ij}^* & \text{otherwise,} \end{cases}$$

which can be regarded as the coefficient of $x^{m - t_j + s_i}$ in $a_{ij}(x)$. The considerations for $\tilde{B}_0$ are done in a very similar manner, from which the lemma follows. $\square$

We will refer to the diagonal transformations S and T as the with the characteristic matrix *associated transformations*. We observe that the transformed linear DAE (14) (and in fact any linear DAE resulting from a Shearing transformation) has a symmetry property. A similar symmetry has been observed in the case of linear ordinary differential systems in [1] and [12].

We shall use the notation $S(e^{2\pi i})$ to denote formally substituting the monomial $x^{j/q}$ with $e^{2\pi i j/q}$ in $S(x)$.

LEMMA 4. Let $P = S(e^{2\pi i})$, $Q = T(e^{2\pi i})$ and $\omega := e^{2\pi i/q}$. Then $\tilde{A}$ and $\tilde{B}$ of (14) satisfies

$$\omega^j(\tilde{A}_j - \lambda \tilde{B}_j) = P(\tilde{A}_j - \lambda \tilde{B}_j)Q \quad \forall j. \quad (15)$$

Proof Following [1], using the notation $S(e^{2\pi i})$ for a formal substitution similar to that in Lemma 4, we have

$$S(e^{2\pi i}) = PS(x) = S(x)P, \quad T(e^{2\pi i}) = QT(x) = T(x)Q$$

and obtain by using the definition of equivalence

$$\tilde{A}(e^{2\pi i}x) - \lambda \tilde{B}(e^{2\pi i}x) = P[\tilde{A}(x) - \lambda \tilde{B}(x)]Q.$$

The lemma follows by inspecting coefficients. $\square$

DEFINITION 2. If a symmetry condition as in (15) holds, we call the pencil $\tilde{A} - \lambda \tilde{B}$ of ramification index q (linear DAE $\tilde{A} - \tilde{B}\theta_r$ respectively) (P, Q) -symmetric.

Remark 5.2. The matrices P and Q in Lemma 4 are diagonal, but we allow any non-singular constant matrices in the above definition.

LEMMA 5. Let $A - B\theta_r$ be (P, Q) -symmetric. Transform this operator with the constant transformation (U, V) and denote by $\tilde{A} - \tilde{B}\theta_r$ the resulting operator. Then $\tilde{A} - \tilde{B}\theta_r$ is $(UPU^{-1}, V^{-1}QV)$ -symmetric.

Proof We compute

$$\begin{aligned} \tilde{A}(e^{2\pi i}x) - \tilde{B}(e^{2\pi i}x)\theta_r &= \\ U(A(e^{2\pi i}x) - B(e^{2\pi i}x)\theta_r)V &= \\ UP(A(x) - B(x)\theta_r)QV &= \\ UPU^{-1}(\tilde{A}(x) - \tilde{B}(x)\theta_r)V^{-1}QV. \end{aligned}$$

This equation leads to the claimed result, by inspecting coefficients as in the proof of Lemma 4. $\square$

LEMMA 6. The spectrum of the characteristic matrix $C_{s,t}(\lambda)$ of a linear DAE that is (P, Q) -symmetric is closed w.r.t multiplication with ω .

Proof We have $\omega^p \tilde{A}_p = P \tilde{A}_p Q$ and $\tilde{B}_0 = P \tilde{B}_0 Q$, thus

$$C_{s,t}(\omega^p \lambda) = \tilde{A}_p - \omega^p \lambda \tilde{B}_0 \sim \omega^p (\tilde{A}_p - \lambda \tilde{B}_0) \sim \tilde{A}_p - \lambda \tilde{B}_0 = C_{s,t}(\lambda)$$

from which the claim follows, since p and q are relatively prime. $\square$

This motivates the definition of an equivalence relation for eigenvalues of a regular matrix pencil

$$\lambda_1 \sim \lambda_2 \iff \exists l \in \{0, \dots, q-1\} : \lambda_1 = \omega^l \lambda_2.$$

Grouping eigenvalues together in their equivalence classes, we define the ω -spectrum

$$\omega\text{-spec}(C_{s,t}(\lambda)) = \{[\mu]_{\sim} \mid \mu \in \text{spec}(C_{s,t}(\lambda))\}$$

where we, slightly abusing notation, will identify $[\mu]_{\sim}$ with μ .

6. THE DECOMPOSITION LEMMA

We have now all the ingredients to formulate and prove the announced decomposition lemma:

THEOREM 2. Consider the linear DAE (1) and assume that its characteristic matrix $C_{s,t}(\lambda)$ is non-trivial, and similar to a block-diagonal pencil of the form

$$\begin{pmatrix} C^{11}(\lambda) & 0 \\ 0 & C^{22}(\lambda) \end{pmatrix}$$

where furthermore

$$\omega\text{-spec}(C^{11}(\lambda)) \cap \omega\text{-spec}(C^{22}(\lambda)) = \emptyset.$$

Then (1) is equivalent to a block-diagonal linear DAE with block sizes matching the blocks of $C^{11}(\lambda)$ and $C^{22}(\lambda)$ such that

1. The transformation is rational and can be computed constructively.
2. The individual systems have non-trivial characteristic matrices equal to $C^{11}(\lambda)$ and $C^{22}(\lambda)$ respectively.

We will prove this theorem in several steps. Note that the case where s and t are vectors of integers follows directly from the Generalised Splitting Lemma as presented in [13]. We will prove the general case in the following way: we will show that the given system can be transformed to an equivalent system, using a ramified transformation, resulting in a new system which has a symmetry property. We will then apply the Generalised Splitting Lemma (in the

form of Proposition 1) and show that the symmetry is preserved. Finally, applying one additional transformation we show that we obtain a rational transformation.

Let S, T be the associated diagonal transformations of $\mathcal{C}_{s,t}(\lambda)$ and U, V such that

$$\tilde{\mathcal{C}}_{s,t}(\lambda) := U\mathcal{C}_{s,t}(\lambda)V = \begin{pmatrix} \mathcal{C}^{11}(\lambda) & 0 \\ 0 & \mathcal{C}^{22}(\lambda) \end{pmatrix}.$$

Denote by $\tilde{A} - \tilde{B}\theta_r$ the operator that corresponds to the DAE resulting from transformation with (U, S, TV) . Using Lemma 5, we can see that $\tilde{A} - \tilde{B}\theta_r$ is (P, Q) -symmetric where $P = US(e^{2\pi i})U^{-1}$ and $Q = V^{-1}T(e^{2\pi i})V$.

LEMMA 7. *The matrices P and Q are block-diagonal, matching the block sizes in $\tilde{\mathcal{C}}_{s,t}(\lambda)$.*

Proof Since (15) is valid for any λ , we derive in particular the two equations $\omega^p \tilde{A}_p = P\tilde{A}_p Q$ and $\tilde{B}_0 = P\tilde{B}_0 Q$. Combining these as matrix pencils and rewriting

$$\omega^p(\tilde{A}_p - \lambda\tilde{B}_0) = P(\tilde{A}_p - \omega^p\lambda\tilde{B}_0)Q$$

as

$$(\omega^p(\tilde{A}_p - \lambda\tilde{B}_0))Q^{-1} = P(\tilde{A}_p - \omega^p\lambda\tilde{B}_0)$$

and decomposing P and Q^{-1} into blocks

$$P = \begin{pmatrix} P^{11} & P^{12} \\ P^{21} & P^{22} \end{pmatrix}, \quad Q^{-1} = \begin{pmatrix} \tilde{Q}^{11} & \tilde{Q}^{12} \\ \tilde{Q}^{21} & \tilde{Q}^{22} \end{pmatrix}$$

matching the block sizes in $\tilde{A}_p - \lambda\tilde{B}_0$, the upper-right blocks yield, after carrying out the matrix multiplications, the following necessary condition:

$$(\omega^p \tilde{A}_p^{11} - \lambda\tilde{B}_0^{11})\tilde{Q}^{12} - P^{12}(\tilde{A}_p^{22} - \lambda\tilde{B}_0^{22}) = 0.$$

Since the ω -spectra of the two regular matrix pencils in this equation have empty intersection, following [5] this Generalised Sylvester Equation has only the trivial solution $\tilde{Q}^{12} = P^{12} = 0$. Using a very similar argument for the lower left block, we obtain that P, Q^{-1} and hence also Q are block-diagonal as desired. $\square$

Using the block-diagonal structure of P and Q , the given symmetry conditions

$$\omega^j(\tilde{A}_j - \lambda\tilde{B}_j) = P(\tilde{A}_j - \lambda\tilde{B}_j)Q \quad (16)$$

give symmetry conditions for the blocks of $\tilde{A}_j - \lambda\tilde{B}_j$, which will be used in the what follows:

$$\begin{aligned} \omega^j(\tilde{A}_j^{11} - \lambda\tilde{B}_j^{11}) &= P^{11}(\tilde{A}_j^{11} - \lambda\tilde{B}_j^{11})Q^{11}, \\ \omega^j(\tilde{A}_j^{12} - \lambda\tilde{B}_j^{12}) &= P^{11}(\tilde{A}_j^{12} - \lambda\tilde{B}_j^{12})Q^{22}, \\ \omega^j(\tilde{A}_j^{21} - \lambda\tilde{B}_j^{21}) &= P^{22}(\tilde{A}_j^{21} - \lambda\tilde{B}_j^{21})Q^{11} \end{aligned}$$

and

$$\omega^j(\tilde{A}_j^{22} - \lambda\tilde{B}_j^{22}) = P^{22}(\tilde{A}_j^{22} - \lambda\tilde{B}_j^{22})Q^{22}.$$

PROPOSITION 1. *The operator $\tilde{A} - \tilde{B}\theta_r$ can be block-diagonalised using a transformation $(\tilde{S}, \tilde{T})$ of ramification index q of the form*

$$\tilde{S} = \begin{pmatrix} I & \tilde{S}^{12} \\ \tilde{S}^{21} & I \end{pmatrix}, \quad \tilde{T} = \begin{pmatrix} I & \tilde{T}^{12} \\ \tilde{T}^{21} & I \end{pmatrix}$$

where

$$\tilde{S} = \sum_{j=0}^{\infty} \tilde{S}_j x^{j/q}, \quad \tilde{T} = \sum_{j=0}^{\infty} \tilde{T}_j x^{j/q},$$

with $\tilde{S}_0^{12} = \tilde{S}_0^{21} = \tilde{T}_0^{12} = \tilde{T}_0^{21} = 0$, obtaining a block-diagonal operator $\tilde{A} - \tilde{B}\theta_r$. Furthermore, the coefficients of $\tilde{S}$ and $\tilde{T}$ have the following properties:

$$\omega^h Q \tilde{T}_h = \tilde{T}_h Q, \quad (17)$$

$$\omega^h \tilde{S}_h P = P \tilde{S}_h. \quad (18)$$

Proof Inserting the power series expansion for $\tilde{A}, \tilde{B}, \tilde{S}$ and $\tilde{T}$ into the definition of equivalence (13) one finds after some calculation for $h > 0$

$$(\tilde{A}_p^{11} - \lambda\tilde{B}_0^{11})\tilde{T}_h^{12} + \tilde{S}_h^{12}(\tilde{A}_p^{22} - \lambda\tilde{B}_0^{22}) = \tilde{R}_h^{12}(\lambda) \quad (19)$$

where

$$\begin{aligned} \tilde{R}_h^{12}(\lambda) = & -(\tilde{A}_{h+p}^{12} - \lambda\tilde{B}_h^{12}) \\ & - \sum_{j=1}^{h-1} ((\tilde{A}_{j+p}^{11} - \lambda\tilde{B}_j^{11})\tilde{T}_{h-j}^{12} + \tilde{S}_{h-j}^{12}(\tilde{A}_{j+p}^{22} - \lambda\tilde{B}_j^{22})) \\ & - \sum_{j=1}^{h-1} \sum_{k=1}^j \tilde{S}_k^{12}(\tilde{A}_{j-k+p}^{21} - \lambda\tilde{B}_{j-k}^{21})\tilde{T}_{h-j}^{12} \\ & + \sum_{j=1}^{h+p-1} \sum_{k=1}^j ((h-j)/q + m - r) \tilde{S}_k^{12} \tilde{B}_{j-k}^{21} \tilde{T}_{h+p-j-rq}^{12} \\ & + \sum_{j=0}^{h+p-1} (j/q - r) \tilde{B}_{h+p-j}^{11} \tilde{T}_{j-rq}^{12} \end{aligned}$$

where $\tilde{S}_j$ and $\tilde{T}_j$ are set to 0 if $j < 0$. Note that $p/q = m < r$ implies in the third sum $h+p-j-rq > 0 \Rightarrow j-k < h$ and in the last sum $j-rq > 0 \Rightarrow h+p-j < h$ hence $\tilde{R}_h^{12}$ only depends on $\tilde{A}_{l+p}, \tilde{B}_l$ with $l \leq h$ and $\tilde{S}_l$ and $\tilde{T}_l$ with $l < h$. Equation (20) is a Generalised Sylvester Equation and can be solved uniquely for $\tilde{T}_0^{12}$ and $\tilde{S}_0^{12}$ given arbitrary values of $\tilde{R}_h^{12}(\lambda)$ since according to the assumptions of the theorem we have $\omega\text{-spec}(\tilde{A}_p^{11} - \lambda\tilde{B}_0^{11}) \cap \omega\text{-spec}(\tilde{A}_p^{22} - \lambda\tilde{B}_0^{22}) = \emptyset$.

A very similar computation yields unique solutions for $\tilde{T}_0^{21}$ and $\tilde{S}_0^{21}$.

We shall now prove (17) and (18) by induction, assuming these equations hold for $j < h$. We shall only prove the properties

$$\omega^h Q^{11} \tilde{T}_h^{12} = \tilde{T}_h^{12} Q^{22}$$

and

$$\omega^h \tilde{S}_h^{12} P^{22} = P^{11} \tilde{S}_h^{12},$$

as (17) and (18) follow from the above and a very similar

property for the blocks $\tilde{S}_h^{21}$ and $\tilde{T}_h^{21}$. One computes

$$\begin{aligned}
\omega^{h+p} \tilde{R}_h^{12}(\lambda) &= \\
&- P^{11}(\tilde{A}_{h+p}^{12} - \omega^p \lambda \tilde{B}_h^{12}) Q^{22} \\
&- \sum_{j=1}^{h-1} \left(P^{11}(\tilde{A}_{j+p}^{11} - \omega^p \lambda \tilde{B}_j^{11}) \omega^{h-j} Q^{11} \tilde{T}_{h-j}^{12} \right. \\
&\quad \left. + \omega^{h-j} \tilde{S}_{h-j}^{12} P^{22}(\tilde{A}_{j+p}^{22} - \omega^p \lambda \tilde{B}_j^{22}) Q^{22} \right) \\
&- \sum_{j=0}^{h-1} \sum_{k=1}^{h-j} \omega^k \tilde{S}_k^{12} P^{22}(\tilde{A}_{j-k+p}^{21} - \omega^p \lambda \tilde{B}_{j-k}^{21}) \omega^{h-j} Q^{11} \tilde{T}_{h-j}^{12} \\
&+ \sum_{j=0}^{h-1} \sum_{k=1}^{h-j} ((h-j)/q - r) \omega^{k+p} \tilde{S}_k^{12} P^{22} \tilde{B}_{j-k}^{21} \\
&\quad \cdot \omega^{h+p-j-rq} Q^{11} \tilde{T}_{h+p-j-rq}^{12} \\
&+ \sum_{j=0}^{h-1} (j/q - r) \omega^{h+p-j} P^{11} \tilde{B}_{h+p-j}^{11} \omega^{j-rq} Q^{11} \tilde{T}_{j-rq}^{12} \\
&= P^{11} \tilde{R}_h^{12}(\omega^p \lambda) Q^{22}.
\end{aligned}$$

Multiplying (19) by ω^{h+p} and using the previous equation, we obtain

$$\begin{aligned}
(\tilde{A}_p^{11} - \lambda \tilde{B}_0^{11}) \omega^{h+p} \tilde{T}_h^{12} + \omega^{h+p} \tilde{S}_h^{12} (\tilde{A}_p^{22} - \lambda \tilde{B}_0^{22}) &= \\
\omega^{h+p} \tilde{R}_h^{12}(\lambda) &= \\
P^{11} \tilde{R}_h^{12}(\omega^p \lambda) Q^{22} &= \\
P^{11} (\tilde{A}_p^{11} - \omega^p \lambda \tilde{B}_0^{11}) \tilde{T}_h^{12} Q^{22} + P^{11} \tilde{S}_h^{12} (\tilde{A}_p^{22} - \omega^p \lambda \tilde{B}_0^{22}) Q^{22}.
\end{aligned}$$

On the other hand, using the fact that $\tilde{A} - \tilde{B}\theta_r$ is (P, Q) -commutative, one derives from (19)

$$\begin{aligned}
P^{11} \tilde{R}_h^{12}(\omega^p \lambda) Q^{22} &= \\
P^{11} (\tilde{A}_p^{11} - \omega^p \lambda \tilde{B}_0^{11}) \omega^h Q^{22} \tilde{T}_h^{12} &+ \\
\omega^h \tilde{S}_h^{12} P^{22} (\tilde{A}_p^{22} - \omega^p \lambda \tilde{B}_0^{22}) Q^{22}.
\end{aligned}$$

Subtracting these two equations and taking into account the condition $\omega\text{-spec}(\tilde{A}_p^{11} - \lambda \tilde{B}_0^{11}) \cap \omega\text{-spec}(\tilde{A}_p^{22} - \lambda \tilde{B}_0^{22}) = \emptyset$ leads to (17) and (18) as desired. $\square$

LEMMA 8. *The matrices P^{11} , Q^{11} , P^{22} and Q^{22} are equivalent to diagonal matrices, with diagonal elements of the form ω^{k_i} with $k_i \in \mathbb{N}$.*

Proof This property clearly holds for the matrices P and Q , as they are similar to the diagonal matrices $S(e^{2\pi i})$ and $T(e^{2\pi i})$ respectively. Being block-diagonal matrices, the property also holds for the individual blocks. $\square$

We now proceed to the proof of our decomposition lemma.

Proof [Theorem 2] Let S and T be the diagonal transformations associated to the system (1). Recall that they are determined by the rational numbers that are entries of the vectors s and t . Let the matrices U , V , $\hat{S}$ and $\hat{T}$ be as introduced previously. Furthermore, let $\hat{U}$, $\hat{V}$ such that (given by the previous lemma)

$$\hat{U}^{-1} P \hat{U} = \text{diag}(\omega^{d_1}, \dots, \omega^{d_n})$$

and

$$\hat{V}^{-1} Q \hat{V} = \text{diag}(\omega^{e_1}, \dots, \omega^{e_n}).$$

Finally, define

$$\hat{S} = \text{diag}(x^{-d_1/q}, \dots, x^{-d_n/q})$$

and

$$\hat{T} = \text{diag}(x^{-e_1/q}, \dots, x^{-e_n/q}).$$

We consider the transformation

$$(S^*, T^*) = (\hat{S} \hat{U} \tilde{S} U S, T V \tilde{T} \hat{V} \hat{T})$$

and will first show that S^* is a rational matrix. Our proof consists of showing the property $S^*(e^{2\pi i} x) = S^*(x)$ which then implies

$$\omega^j S_j^* = S_j^* \quad \forall j$$

For $q > 1$, this then implies (since $\omega = e^{2\pi i/q} \neq 1$) $S_j^* = 0$ if $j \neq 0 \pmod{q}$. If $q = 1$, nothing needs to be shown as the ramification index is 1.

We recall the following properties:

$$\begin{aligned}
S(e^{2\pi i} x) &= S(e^{2\pi i}) S(x), \\
\tilde{S}(e^{2\pi i} x) &= P S(x) P^{-1}
\end{aligned}$$

and we compute

$$\begin{aligned}
\hat{S}(e^{2\pi i} x) &= \hat{S}(x) \text{diag}(\omega^{-d_1}, \dots, \omega^{-d_n}) \\
&= \hat{S}(x) D^{-1}
\end{aligned}$$

where

$$D = \text{diag}(\omega^{d_1}, \dots, \omega^{d_n}).$$

We finally obtain

$$\begin{aligned}
S^*(e^{2\pi i} x) &= \\
\hat{S}(x) D^{-1} \hat{U} P \tilde{S}(x) P^{-1} U S(e^{2\pi i}) S(x) &= \\
\hat{S}(x) D^{-1} \hat{U} P \hat{U}^{-1} \hat{U} \tilde{S}(x) U S(e^{-2\pi i}) U^{-1} U S(e^{2\pi i}) S(x) &= \\
S^*(x)
\end{aligned}$$

where we have used $\hat{U} P \hat{U}^{-1} = D$ and $P = U S(e^{2\pi i}) U^{-1}$. From this, the claim follows.

Similarly, it can be shown that $T^*(e^{2\pi i} x) = T^*(x)$, but we will not detail the proof as it can be done in the same manner. From this it follows that (S^*, T^*) is a rational transformation.

It is easy to see that the given system (1), transformed by (S^*, T^*) is block-diagonal as the transformation $(\tilde{S} U S, T V \tilde{T})$ achieves a block-diagonalisation according to Lemma 1 and the additional transformation with $(\hat{S} \hat{U}, \hat{V} \hat{T})$ preserves this structure (being block-diagonal as well). Since (S^*, T^*) is a rational transformation, this is also true for the resulting system. The last statement in the theorem can be seen by using the matrices $\hat{S}^{-1}$ and $\hat{T}^{-1}$ as associated diagonal transformations. $\square$

7. AN EXAMPLE

The content of this paper being fairly technical, it will be helpful to consider a worked example. We study the linear

DAE of the form (1) with $r = 1$,

$$A = \begin{pmatrix} -x & 3x^3 + 1 + 1/4x^2 & -2x^2 \\ x^3 + x & -3x^5 + x^2 - 1/2x^3 + \frac{15}{4}x^4 & 2x^4 - 2x^3 \\ -x^2 & 3x^4 + x + 1/4x^3 & -2x^3 + x \end{pmatrix}$$

and

$$B = \begin{pmatrix} 1 & -x^2 + 1/2x & x \\ -x^2 + 2x & x^4 - 5/2x^3 + 2x^2 & -x^3 + 2x^2 - x \\ x & -x^3 + 1/2x^2 - x & x^2 + 1 \end{pmatrix}.$$

The matrix B is singular and hence this system cannot be written as a system of ordinary linear differential equations. The leading pencil is

$$A_0 - \lambda B_0 = \begin{pmatrix} -\lambda & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -\lambda \end{pmatrix}$$

which is singular, i.e. one has $\det(A_0 - \lambda B_0) = 0$. One verifies that for the rational number vectors $s = (1, \frac{1}{2}, 1)$ and $t = (1, \frac{1}{2}, 1)$ the (s, t) -characteristic matrix of the system is

$$\mathcal{C}(\lambda) = \begin{pmatrix} -\lambda & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -\lambda \end{pmatrix}.$$

This is a block-diagonal regular matrix pencil and the assumptions of Theorem 2 apply – indeed, we find $\det \mathcal{C}^{11}(\lambda) = 1$, $\det \mathcal{C}^{22}(\lambda) = \lambda$ which shows

$$\omega\text{-spec}(\mathcal{C}^{11}(\lambda)) \cap \omega\text{-spec}(\mathcal{C}^{22}(\lambda)) = \{\infty\} \cap \{0\} = \emptyset.$$

Following the proof of our decomposition lemma, one can construct the rational transformation

$$(S^*, T^*) = \left(\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & x \\ -x & 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 & -x \\ 0 & 1 & 0 \\ 0 & x & 1 \end{pmatrix} \right)$$

which transforms the initial system into a new block-diagonal system with operator $A^* - B^*\theta_r$ given by

$$\begin{pmatrix} -x & 1 + 1/4x^2 & 0 \\ x & 1/2x^2(4+x) & 0 \\ 0 & 0 & x \end{pmatrix} - \begin{pmatrix} 1 & 1/2x & 0 \\ 2x & x^2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \theta_r.$$

Here, $(S^*, T^*) = (S^{-1}\tilde{S}S, T\tilde{T}T^{-1})$ with

$$S = \text{diag}(1, x^{-1/2}, 1), \quad T = \text{diag}(1, x^{1/2}, 1)$$

and $\tilde{S}, \tilde{T}$ are matrices with ramification index $q = 2$, given by applying Lemma 1 to the operator $\tilde{A} - \tilde{B}\theta_r = S(A - B\theta_r)T$. As expected, $\tilde{A} - \tilde{B}\theta_r$ is (P, Q) -symmetric where

$$P = \text{diag}(1, -1, 1), \quad Q = \text{diag}(1, -1, 1).$$

The leading pencil of $\tilde{A} - \tilde{B}\theta_r$ is

$$\tilde{A}_1 - \lambda \tilde{B}_0 = \begin{pmatrix} -\lambda & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -\lambda \end{pmatrix}$$

which equals the characteristic matrix of the system. Note that in this example, calculations are exact and the decomposition lemma yields two systems of smaller size with coefficients in $\mathbb{C}[x]$. In general, computations will have to be done on formal power series coefficients which makes an implementation more difficult – see the next section for a brief discussion of implementation aspects.

8. DISCUSSION

Implementation of our decomposition lemma is currently in progress in the Computer Algebra package ISOLDE [3] which was originally developed for the symbolic resolution of linear ordinary differential systems of the form (2). The main technical challenge is the adaptation of the mechanism of lazy evaluation of power series to manipulating linear DAEs. Once this task is achieved, one can hope for a fairly efficient handling of systems with reasonably complicated coefficients.

The decomposition lemma we have developed in this paper is an additional new member of a family of decomposition lemmas that have been given by us and others in the past. In particular, it specialises to the following existing concepts:

- For a linear DAE of the form (1) with $B = I$, to the Root-free Splitting-Lemma of [12] for linear systems of ordinary differential equations.
- In case of $B = I$ and additionally the vectors $s=t=0$ one is led to the classical Splitting-Lemma [16].
- If B is non-singular and $s=t=0$, one encounters the so called k -simple systems and the Generalised Splitting-Lemma of [13].

An interesting future research programme is investigating the obvious question that the reader may have when reviewing our definition of the characteristic matrix: is it possible to systematically find suitable vectors s and t that lead to a nontrivial characteristic matrix? Even for the case $B = I$ this seems a difficult problem, and Hufford's paper does not give any strategies towards possible solutions. We believe that a generalisation of the Moser-reduction algorithm as defined for systems of ordinary linear differential equations to linear DAEs may lead to an algorithm achieving this for certain types of vectors s and t (only containing integers).

In the general case of rational entries for s and t , it remains to see whether methods for the ODE case that deal with ramifications can be improved to such an extent that it will be possible to generalise them to DAEs.

In any case, as decomposition lemmas are important building blocks of the formal reduction in the ordinary case, one can expect that Theorem 2 will be equally important for the formal reduction of linear DAEs. This could lead to useful applications, such as the computation of formal solutions of this type of equations.

9. REFERENCES

- [1] W. Balser. *Formal Power Series and Linear Systems of Meromorphic Ordinary Differential Equations*. Springer New York, 2000.
- [2] M. Barkatou. An algorithm to compute the exponential part of a formal fundamental matrix solution of a linear differential system. *Journal of App. Alg. in Eng. Comm. and Comp.*, 8(1):1–23, 1997.

- [3] M. Barkatou and E. Pflügel. The ISOLDE package. A SourceForge Open Source project, <http://isolde.sourceforge.net>, 2006.
- [4] M. Barkatou and E. Pflügel. Computing super-irreducible forms of systems of linear differential equations via moser-reduction: A new approach. In *Proceedings of ISSAC*, pages 1–8. ACM, 2007.
- [5] K. Chu. The solution of the matrix equations $AXB - CXD = E$ and $(YA - DZ, YC - BZ) = (E, F)$. *Linear Algebra Appl.*, 93:93–105, 1987.
- [6] F. Gantmacher. *The Theory of Matrices*. Volumes 1 and 2, Chelsea, New York, 1959.
- [7] A. Hilali and A. Wazner. Formes super-irréductibles des systèmes différentiels linéaires. *Numer. Math.*, 50:429–449, 1987.
- [8] G. Hufford. On the characteristic matrix of a matrix of differential operators. *Journal of Diff. Equ.*, (1):27–38, 1965.
- [9] A. H. G.-Q. Z. L. O. Chua, C.W. Wu. A universal circuit for studying and generating chaos, I: Routes to chaos. *IEEE Trans. Circuits. Syst.*, 1993.
- [10] J. Moser. The order of a singularity in Fuchs’ theory. *Math. Z.*, pages 379–398, 1960.
- [11] E. Pflügel. Résolution symbolique des systèmes différentiels linéaires. Ph.D. Thesis, Université Joseph Fourier, oct. 1998.
- [12] E. Pflügel. A root-free splitting-lemma for systems of linear differential equations. Preprint http://dircweb.king.ac.uk/papers/Pfluegel_E.2005_860909/root_split.pdf.
- [13] E. Pflügel. Effective formal reduction of linear differential systems. *Appl. Alg. Eng. Comm. Comp.*, 10(2):153–187, 2000.
- [14] M. P. Quéré and G. Villard. An algorithm for the reduction of linear DAE. In *ISSAC ’95: Proceedings of the 1995 international symposium on Symbolic and algebraic computation*, pages 223–231, New York, NY, USA, 1995. ACM.
- [15] L. W. W. A. Harris, Y. Sibuya. A reduction algorithm for linear differential systems. *Funkcialaj Ekvacioj*, 11:59–67, 1968.
- [16] W. Wasow. *Asymptotic Expansions for Ordinary Differential Equations*. Robert E. Krieger Publishing, 1967.

Good Reduction of Puiseux Series and Complexity of the Newton-Puiseux Algorithm over Finite Fields

Adrien Poteaux, Marc Rybowicz

XLIM - DMI, UMR CNRS 6172

Université de Limoges

adrien.poteaux@xlim.fr, marc.rybowicz@xlim.fr

ABSTRACT

In [12], we sketched a numeric-symbolic method to compute Puiseux series with floating point coefficients. In this paper, we address the symbolic part of our algorithm. We study the reduction of Puiseux series coefficients modulo a prime ideal and prove a good reduction criterion sufficient to preserve the required information, namely Newton polygon trees. We introduce a convenient modification of Newton polygons that greatly simplifies proofs and statements of our results. Finally, we improve complexity bounds for Puiseux series calculations over finite fields, and estimate the bit-complexity of polygon tree computation.

Categories and Subjects Descriptors: I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms; F.2.1 [Analysis of Algorithms and Problem Complexity]: Numerical Algorithms and Problems.

General Terms: Algorithms, Theory.

Keywords: Puiseux Series, Algebraic Functions, Modular Methods, Finite Fields, Complexity, Symbolic-Numeric Algorithms.

1. INTRODUCTION

Let K be a number field and $F(X, Y)$ be a squarefree bivariate polynomial in $K[X, Y]$, monic in Y , such that $\deg_Y(F) = d > 1$ and $\deg_X(F) = n > 0$. Denote by $\Delta_F(X)$ the discriminant of F with respect to Y . A root of Δ_F will be called a *critical point*. The equation $F(X, Y) = 0$ defines d algebraic functions of the variable X , which are analytic in any simply connected domain $\mathcal{D} \subset \mathbb{C}$ free of critical points. If $\mathcal{D}$ is included in a sufficiently small disc centered at a critical point x_0 , it is well-known that numerical values of these functions in $\mathcal{D}$ can be obtained directly via truncated Puiseux series at $X = x_0$ (see Section 2). We have used this fact to devise an algorithm to compute the monodromy of the Riemann sphere covering defined by the curve $F(X, Y) = 0$ [12], a question that has numerous applications, including the determination of Galois

groups, effective versions of Abel-Jacobi's Theorem [5], which in turn are useful in various context (see [5, 4]). Unfortunately, applying a floating point Newton-Puiseux algorithm (see Section 3) to compute Puiseux series above a critical point is doomed to failure. Indeed, if the critical point x_0 is replaced with an approximation, expansion algorithms return approximate series with very small convergence discs and do not retain important information, such as ramification indices. Therefore, the output is not helpful. On the other hand, coefficient growth considerably slows down symbolic methods. Since the degree of Δ_F is in $O(nd)$, Puiseux series coefficients above x_0 belong to a finite extension of K whose degree over K may be in $O(d^2n)$. Moreover, when these coefficients are expressed as linear combination over $\mathbb{Q}$, the size of the rational numbers involved may also be overwhelming. Floating point evaluation of such coefficients must, in some cases, be performed with a high number of digits because spectacular numerical cancellations occur (see examples in [12]). Walsh [18] has shown that, for any $\epsilon > 0$, the singular part of Puiseux series can be computed using $O(d^{32+\epsilon} n^{4+\epsilon} \log h^{2+\epsilon})$ bit operations where h is the height of F . Although this bound is probably not sharp, it is not encouraging and tends to confirm experimental observations. To alleviate these problems, we introduced a symbolic-numeric approach : exact important information is first obtained by means of computations modulo a well chosen prime number p , then this information is used to guide floating point computations. The coefficient size is therefore kept under control while numerical instability is reduced. Exact important data, such as ramification indices and intersection multiplicities of branches, are preserved. Experimental evidences reported in [12] seem to validate this approach. This paper presents several contributions :

- Section 3 introduces “generic Newton polygons” and “polygon trees”. The latter concept captures precisely the symbolic information needed for floating point computations. We explain how polygon trees can be obtained using modular arithmetic. For this task, generic Newton polygons are more convenient than classical ones (see Section 4).
- In Section 4, we study modular reduction of Puiseux series and provide a fully proved and easy to check criterion for choosing a “good prime”.
- Improved complexity bounds for the computation of rational Puiseux expansions over finite fields are given in Section 5. We also deduce bit-complexity estimates for a randomized version of the modular part of our symbolic-numeric method (Section 6).

Because of the lack of space, proofs and many comments

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

have been omitted. For proofs and details, the reader is referred to the extended version [13] available on the Internet.

The following notations and well-known facts will be used throughout the paper :

- If L is a field, $\bar{L}$ will denote an algebraic closure of L .
- For each positive integer e , ζ_e is a primitive e -th root of unity in $\bar{L}$. Primitive roots are chosen so that $\zeta_{ab}^b = \zeta_a$.
- v_X denotes the X -adic valuation of the fractional power series field $L((X^{1/e}))$, normalized with $v_X(X) = 1$. If $S \in L((X^{1/e}))$, we denote by $tc(S)$ the trailing coefficient of S , namely $S = tc(S)X^{v_X(S)} + \text{higher order terms}$.
- If $S = \sum_{k \geq l} \alpha_k X^{k/e}$ is an element of $L((X^{1/e}))$ and r is a rational number, $\tilde{S}^r$ denotes the truncated power series $\tilde{S}^r = \sum_{k=l}^N \alpha_k X^{k/e}$ where $N = \max\{k \in \mathbb{N} \mid \frac{k}{e} \leq r\}$.
- The discriminant of a univariate polynomial U is denoted by Δ_U . If U is a multivariate polynomial, the context will always allow to identify the variable.
- Let f be a polynomial in $L[T]$ with squarefree factorization $f = \prod_{i=1}^r f_i^{k_i}$. We associate to f the partition of $\deg f$ denoted $[f] = (k_1^{\deg f_1} \dots k_r^{\deg f_r})$. Namely, the multiplicity k_i is repeated $\deg f_i$ times in the decomposition of $\deg f$.
- If $H \in L[X, Y]$, then H_X and H_Y are the formal partial derivatives of H .
- For $H(\underline{X}) = \sum_{\underline{k}} \alpha_{\underline{k}} \underline{X}^{\underline{k}} \in \mathbb{C}[\underline{X}] = \mathbb{C}[X_1, \dots, X_n]$, where $\underline{k}$ is a multi-index, we denote $\|H\|_{\infty} = \max_{\underline{k}} |\alpha_{\underline{k}}|$.

2. PUISEUX SERIES

We need to state results over more general fields than K . Throughout this section, L denotes a field of characteristic $p \geq 0$ and F belongs to $L[X, Y]$. Otherwise, we keep the assumptions and notations of Section 1. We also impose the condition :

$$p = 0 \quad \text{or} \quad p > d = \deg_Y(F) \quad (1)$$

After a change of variable $X \leftarrow X + x_0$, we may assume that the critical point is $X = 0$.

2.1 Classical Puiseux series

In this part, we review classical results about Puiseux series. We begin with :

THEOREM 1 (PUISEUX). *Let H be a squarefree polynomial of $L[X, Y]$ such that $\deg_Y(H) = d > 0$. If condition (1) is satisfied, there exist positive integers $e_1, \dots, e_s$ satisfying $\sum_{i=1}^s e_i = d$ such that H , viewed as a polynomial in Y , has d distinct roots in $\bar{L}((X))$ which can be written :*

$$S_{ij}(X) = \sum_k \alpha_{ik} \zeta_{e_i}^{jk} X^{\frac{k}{e_i}}$$

for $1 \leq i \leq s$ and $0 \leq j \leq e_i - 1$. Moreover, the set of coefficients $\{\alpha_{ik}\}$ is included in a finite algebraic extension of L .

DEFINITION 1. *The d fractional Laurent series above are called Puiseux series of H above 0. The integer e_i is the ramification index of S_{ij} . If $e_i > 1$, then S_{ij} is ramified. If $S_{ij} \in \bar{L}[[X^{1/e_i}]]$, we say that S_{ij} is defined at $X = 0$. If $S_{ij}(0) = 0$, we say that S_{ij} vanishes at $X = 0$.*

An arbitrary number of terms of all Puiseux series can be effectively computed using the Newton-Puiseux algorithm (see Section 3). For each positive integer $e \leq d$, hypothesis (1) implies that the Galois group $\mathbb{G}_e$ of $\bar{L}((X^{1/e}))/\bar{L}((X))$ is cyclic and generated by : $X^{1/e} \mapsto \zeta_e X^{1/e}$. Hence, $\mathbb{G}_e$ permutes cyclically the elements of $S_i = \{S_{ij}(X)\}_{0 \leq j \leq e_i - 1}$.

DEFINITION 2. *We call S_i a cycle of H above 0. If an element of S_i (thus, all elements) vanishes at $X = 0$, we say that the cycle vanishes at $X = 0$.*

Since the S_{ij} ($0 \leq j \leq e_i - 1$) can be quickly recovered from any element of S_i , it is sufficient for our purposes to compute a set of representatives for the cycles of H .

DEFINITION 3. *The regularity index r_{ij} of S_{ij} in H is the least integer N such that $\widetilde{S_{ij}}^{\frac{N}{e_i}} = \widetilde{S_{uv}}^{\frac{N}{e_i}}$ implies $(u, v) = (i, j)$. The truncated series $\widetilde{S_{ij}}^{\frac{r_{ij}}{e_i}}$ is called the singular part of S_{ij} in H .*

In other words, r_{ij} is the smallest number of terms necessary to distinguish S_{ij} from the other Puiseux series above 0. It is worth noting that r_{ij} depends not only on S_{ij} , but also on H since H is not assumed irreducible in $L[X, Y]$.

If the singular part of a Puiseux series is known, a change of variable yields a bivariate polynomial for which remaining terms of the series can be computed “fast” using quadratic Newton iterations [11, 17]. Newton iterations can be applied to series with floating point coefficients, therefore we focus on the computation of the singular parts of the S_{ij} . Since it can be shown that all elements of a cycle S_i have the same regularity index, that we denote r_i , the problem reduces to the determination of the singular part of a representative of S_i for $1 \leq i \leq s$.

2.2 The characteristic of a Puiseux series

We derive relations between the discriminant of F and particular coefficients of its Puiseux series that we shall use to define a “good reduction” criterion. Let $S(X) = \sum_{i=0}^{\infty} \alpha_i X^{i/e}$ denote a Puiseux series of F with ramification index $e > 1$. We define a sequence $(B_0, R_0), \dots, (B_g, R_g)$ of integer pairs as follows : $(B_0, R_0) = (0, e)$, and for $j > 0$, if $R_{j-1} > 1$ we set $B_j = \min\{i > B_{j-1} \mid \alpha_i \neq 0 \text{ and } R_{j-1} \nmid i\}$ and $R_j = \gcd(B_j, R_{j-1})$. If $R_{j-1} = 1$, we stop and set $g = j - 1$. Note that $g \geq 1$ and $R_g = 1$.

Finally, we set $Q_j = R_{j-1}/R_j$, $M_j = B_j/R_j$ ($1 \leq j \leq g$) and define H_j to be the largest nonnegative integer such that $B_j + H_j R_j < B_{j+1}$ for $0 \leq j \leq g - 1$. It is clear that $e = Q_1 Q_2 \dots Q_g$ and M_j is an integer prime to Q_j .

With these notations, and up to a new indexing of the coefficients, S can be written in the form :

$$\begin{aligned} S(X) &= \sum_{j=0}^{H_0} \alpha_{0,j} X^j \\ &+ \gamma_1 X^{\frac{M_1}{Q_1}} + \sum_{j=1}^{H_1} \alpha_{1,j} X^{\frac{M_1+j}{Q_1}} \\ &+ \gamma_2 X^{\frac{M_2}{Q_1 Q_2}} + \sum_{j=1}^{H_2} \alpha_{2,j} X^{\frac{M_2+j}{Q_1 Q_2}} \\ &+ \dots + \dots \\ &+ \gamma_g X^{\frac{M_g}{Q_1 Q_2 \dots Q_g}} + \sum_{j=1}^{\infty} \alpha_{g,j} X^{\frac{M_g+j}{Q_1 Q_2 \dots Q_g}} \end{aligned}$$

In the expression above, the monomials of S are ordered by increasing (rational) degree.

DEFINITION 4 ([19, 1]). *The characteristic of S is the tuple of integers $(e; B_1, \dots, B_g)$. The characteristic coefficients are the elements of $(\gamma_1, \dots, \gamma_g)$ and the characteristic monomials are the corresponding monomials of S .*

PROPOSITION 1. *Assume that hypothesis (1) is satisfied. Let $G(X, Y)$ be the minimal polynomial over $\bar{L}((X))$ of a ramified Puiseux series $S \in \bar{L}[[X^{1/e}]]$ as above. Then :*

- $tc(\Delta_G) = \pm \left(\prod_{i=1}^g Q_i^{R_i} \prod_{i=1}^g \gamma_i^{R_{i-1} - R_i} \right)^e$
- $v_X(\Delta_G) = \sum_{i=1}^g B_i(R_{i-1} - R_i)$.

2.3 Rational Puiseux expansions

In order to perform computations in the smallest possible extension of L and to take advantage of conjugacy over L , Duval introduced the notion of “rational Puiseux expansions over L ” [6]. This arithmetical concept is irrelevant in the context of floating point computations, but will prove useful for expansions over finite fields.

DEFINITION 5. *Let H be a polynomial in $L[X, Y]$ with $\deg_Y H > 0$. A parametrization $R(T)$ of H is a pair of non constant power series $R(T) = (X(T), Y(T)) \in \overline{L}((T))^2$ such that $H(X(T), Y(T)) = 0$ in $\overline{L}((T))$. The parametrization is irreducible if there is no integer $u > 1$ such that $R(T) \in \overline{L}((T^u))^2$. The coefficient field of $R(T)$ is the extension of L generated by the coefficients of $X(T)$ and $Y(T)$.*

Assume for a moment that H is irreducible in $L[X, Y]$ so that $\mathcal{K} = L(X)[Y]/(H)$ is an algebraic function field. A parametrization $R(T) = (X(T), Y(T))$ induces a field morphism :

$$\begin{aligned} \phi_R : \quad \mathcal{K} &\rightarrow \overline{L}((T)) \\ f(X, Y) &\mapsto f(X(T), Y(T)) \end{aligned}$$

Composing ϕ_R with the valuation v_T of $\overline{L}((T))$, we obtain a valuation of $\mathcal{K}$ that we denote again by v_T . It is easily seen that the set $\mathfrak{P}_R = \{f \in \mathcal{K} \mid v_T(f) > 0\}$ is a place of $\mathcal{K}$ in the sense of [2] and that $V_R = \{f \in \mathcal{K} \mid v_T(f) \geq 0\}$ is the corresponding V-ring of $\mathcal{K}$. We recall that $\mathfrak{P}_R$ is the unique maximal ideal of V_R and that the residue field of $\mathfrak{P}_R$ is $V_R/\mathfrak{P}_R$, which can be viewed as a finite algebraic extension of L . Therefore, we obtain a mapping Ψ from the set of parametrizations of F onto the set of places of $\mathcal{K}$. Reciprocally, to each place $\mathfrak{P}$ of $\mathcal{K}$ correspond a parametrization of H . Let us denote by $\{\mathfrak{P}_i\}_{1 \leq i \leq r}$ the places of $\mathcal{K}$ dividing X and by k_i the residue field of $\mathfrak{P}_i$.

DEFINITION 6 (RATIONAL PUISEUX EXPANSIONS).

• Assume that H is irreducible in $L[X, Y]$. A system of L -rational Puiseux expansions above 0 of H is a set of irreducible parametrizations $\{R_i\}_{1 \leq i \leq r}$ of the form :

$$R_i(T) = (X_i(T), Y_i(T)) = (\gamma_i T^{e_i}, \sum_k \beta_{ik} T^k) \in \overline{L}((T))^2$$

with $e_i > 0$ such that :

- (i) Ψ is one-to-one from $\{R_i\}_{1 \leq i \leq r}$ to $\{\mathfrak{P}_i\}_{1 \leq i \leq r}$. We assume that the $\mathfrak{P}_i$ are numbered so that $\mathfrak{P}_i = \mathfrak{P}_{R_i} = \Psi(R_i)$.
- (ii) The coefficient field of R_i is isomorphic to k_i .
- Assume that H is squarefree. A system of L -rational Puiseux expansions above 0 of H is the union of systems of L -rational Puiseux expansions for the irreducible factors of H in $L[X, Y]$.

DEFINITION 7. If $Y_i \in \overline{L}[[T]]$, we say that R_i is defined at $T = 0$ and call $(X_i(0), Y_i(0))$ the center of R_i .

The classical formula relating degrees of residue fields and ramification indices of an algebraic function field (see [2]) translates into :

THEOREM 2. Let H be squarefree polynomial of $L[X, Y]$, $\deg_Y H = d > 0$ and $\{R_i\}_{1 \leq i \leq r}$ be a system of L -rational Puiseux expansions above 0 for H . If f_i denotes the degree over L of the coefficient field of R_i , then $\sum_{i=1}^r e_i f_i = d$.

Classical Puiseux series can readily be deduced from a system of rational Puiseux expansions (see [13]). Classical Puiseux series that are defined at $X = 0$ (resp. that vanish at $X = 0$) correspond to rational Puiseux expansions defined at $T = 0$ (resp. centered at $(0, 0)$). Moreover, we note that regularity indices for all Puiseux series corresponding to the same rational Puiseux expansion are equal. Therefore, we define the singular part of a rational Puiseux expansion R_i to be the pair $(\gamma_i T^{e_i}, \sum_{k=-\infty}^{r_i} \beta_{ik} T^k)$, where r_i is the regularity index of a Puiseux series associated to R_i .

Since our initial polynomial F is monic, rational Puiseux expansions of F above zero are all defined at $T = 0$.

3. NEWTON-PUISEUX ALGORITHM

We describe an algorithm to compute singular parts of rational Puiseux expansions. We also briefly recall how to compute classical Puiseux series. Throughout Section 3, L denotes again a field of characteristic $p \geq 0$ and $F \in L[X, Y]$ is a polynomial such that condition (1) is satisfied. Moreover, we keep assumptions and notations of Section 1.

Newton polygons and characteristic polynomials are the crucial tools. We first recall well-known definitions and introduce a variant that will prove more convenient and powerful.

3.1 Generic Newton polygons

Assume that $H(X, Y) = \sum_{i,j} a_{ij} X^i Y^j$ is a polynomial of $L[[X]][Y]$ such that $H(0, Y) \neq 0$.

DEFINITION 8. Denote by $\mathcal{I}(H)$ the nonnegative integer $v_Y(H(0, Y))$ and by $\mathcal{H}$ the convex hull of $\text{Supp}(H) = \{(i, j) \in \mathbb{N}^2 \mid a_{ij} \neq 0\}$. The Newton Polygon $\mathcal{N}(H)$ of H is the lower part of $\mathcal{H}$. Namely :

- If $H(X, 0) \neq 0$, $\mathcal{N}(H)$ is formed by the sequence of edges of $\mathcal{H}$ closest to the origin and joining $(0, v_X(H(X, 0)))$ to $(\mathcal{I}(H), 0)$,
- If $H(X, 0) = 0$, $(0, v_X(H(X, 0)))$ is replaced by the leftmost point of $\mathcal{H}$ with smallest j -coordinate.

We introduce a slightly different object, that we call *generic Newton polygon* for reasons explained in [13]. This variation allows a homogeneous treatment of finite series, clearer specifications for the algorithms and simplifies wording and proofs of results regarding modular reduction.

DEFINITION 9. The generic Newton polygon $\mathcal{GN}(H)$ is obtained by restricting $\mathcal{N}(H)$ to edges with slope no less than -1 and by joining the leftmost remaining point to the vertical axis with an edge of slope -1 .

In other words, we add a fictitious point $(0, j_0)$ to $\text{Supp}(H)$ so as to mask edges with slope less than -1 .

EXAMPLE 1. Consider $H_1(X, Y) = Y^7 + X^2 Y^2 + X Y^4 + X^6 + X^2 Y^3 + X Y^5 + X^4 Y^3$. In Figure 1, the support of H_1 is represented by crosses, $\mathcal{GN}(H_1)$ is drawn with plain lines and the masked edge of $\mathcal{N}(H_1)$ with a dotted line.

EXAMPLE 2. Consider $H_2(X, Y) = Y^8 + 3X^2 Y^3 + X Y^5 + 2X^6 + 4X^3 Y^2 + X^2 Y^5 + X^4 Y^3$ and Figure 1. The edge with slope -1 is prolonged until the vertical axis.

EXAMPLE 3. Assume that $H_3(X, Y) = Y$. The classical polygon $\mathcal{N}(H_3)$ is the trivial polygon $(1, 0)$. But $\mathcal{GN}(H_3)$ is formed of a unique edge joining $(0, 1)$ to $(1, 0)$.

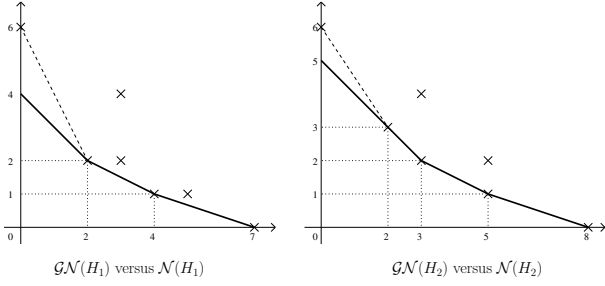


Figure 1: Generic versus classical polygons

REMARK 1. Mark Van Hoeij pointed out to us that his implementation of the Newton-Puiseux algorithm, available since Maple V.5 (`algcures[puiseux]`), implicitly uses the concept of generic polygons. His motivation was to improve efficiency : At each recursive step, it is possible to compute modulo a well-chosen power of X so as to precisely obtain the generic polygon of the next step. This code is used to compute integral bases [16], but the implementation technique has not been published. In essence, Van Hoeij's implementation uses "local truncation orders", while Proposition 6 can be viewed as a "global truncation order". The latter allows to obtain complexity bounds. While local truncation orders may prove more efficient in practice, it is not clear that they lead to a better asymptotic behaviour.

The algorithm first stage requires a special treatment. To this effect, we introduce the following definition :

DEFINITION 10. The exceptional Newton polygon $\mathcal{EN}(H)$ is the unique horizontal edge $[(0,0), (\deg_Y H(0,Y), 0)]$.

In particular, $\mathcal{EN}(F) = [(0,0), (d,0)]$ since F is monic. To an edge Δ of $\mathcal{GN}(H)$ (or $\mathcal{N}(H)$, $\mathcal{EN}(H)$) correspond three nonnegative integers q, m and l with q and m coprime such that Δ is on the line $qj + mi = l$. If Δ is the horizontal edge of $\mathcal{EN}(H)$, $m = l = 0$ and we choose $q = 1$.

DEFINITION 11. We define the characteristic polynomial of Δ as $\phi_\Delta(T) = \sum_{(i,j) \in \Delta} a_{ij} T^{\frac{i-i_0}{q}}$, where i_0 is the smallest value i such that (i,j) belongs to Δ .

Note that if $\mathcal{N}(H)$ is used, $\phi_\Delta(T)$ cannot vanish at $T = 0$, while $\mathcal{GN}(H)$ allows such cancellation if Δ is a fictitious edge (or contains a fictitious part). In this case, the multiplicity of 0 as a root of $\phi_\Delta(T)$ is the length of the fictitious edge (or portion of the fictitious edge) added. For $\mathcal{EN}(H)$, 0 can also be a root of the characteristic polynomial.

The algorithm below performs successive changes of variable, determined by (q, m, l) and the roots of ϕ_Δ . It returns a set of triplets $\{(G_i(X, Y), P_i(X), Q_i(X, Y))\}_i$ such that :

- $G_i, P_i, Q_i \in \overline{L}[X, Y]$,
- $P_i(X)$ is a monomial of the form $\lambda_i X^{e_i}$,
- $Q_i(X, Y) = Q_{i0}(X) + Y X^{r_i}$, where r_i is the regularity index of the expansion and $(P_i(T), Q_{i0}(T))$ is the singular part of a parametrization of F ,
- There exist nonnegative integers L_i such that $G_i(X, Y) = F(P_i(X), Q_i(X, Y))/X^{L_i}$, $G_i(0, 0) = 0$ and $G_{iY}(0, 0) \neq 0$.

3.2 Rational Newton-Puiseux algorithm

We present an algorithm due to Duval to compute singular parts of rational Puiseux expansions above 0 [7]. We also give differences between this algorithm and the classical one. We need two auxiliary algorithms, for which we only provide specifications :

Factor(L, ϕ)

Input: L : A field.

ϕ : A univariate polynomial in $L[T]$.

Output: A set of pairs $\{(\phi_i, k_i)\}_i$ so that ϕ_i is irreducible in $L[T]$ and $\phi = \prod_i \phi_i^{k_i}$.

Bezout(q, m)

Input: q, m : Two coprime positive integers.

Output: A pair of integers (u, v) so that $uq - mv = 1$.
If $q = 1$, enforce $v = 0$ and $u = 1$.

Algorithm `RNPuiseux`(L, H)

Input: L : A field.

H : A squarefree polynomial of degree $d \geq 1$ in $L[X, Y]$, such that $H(0, Y) \neq 0$.

Output: A set of triplets $\{[G_i, P_i, Q_i]\}_i$, which form a set of representatives for :
- L -rational Puiseux expansions of H defined at $T = 0$ for the initial call,
- L -rational Puiseux expansions of H centered at $(0, 0)$ for recursive calls.

Begin

If in a recursive call then

$\mathcal{P} \leftarrow \mathcal{GN}(H)$

If $\mathcal{I}(H) = 1$ then Return $\{[H, X, Y]\}$ End

else

$\mathcal{P} \leftarrow \mathcal{EN}(H)$

End

$\mathcal{R} \leftarrow \{\}$

For each side Δ of $\mathcal{P}$ do

Compute q, m, l and ϕ_Δ

$(u, v) \leftarrow \text{Bezout}(q, m)$

For each (f, k) in `Factor`(L, ϕ_Δ) do

$\xi \leftarrow$ Any root of f

$H'(X, Y) \leftarrow H(\xi^v X^q, X^m(\xi^u + Y))/X^l$

For each $[G, P, Q]$ in `RNPuiseux`($L(\xi), H'$) do

$\mathcal{R} \leftarrow \mathcal{R} \cup \{[G, \xi^v P^q, P^m(\xi^u + Q)]\}$

End

End

End

Return $\mathcal{R}$

End.

Replacing L by $\overline{L}$ and (u, v) by $(1/q, 0)$ in `RNPuiseux`, one obtains the classical algorithm, that we call `CNPuiseux`.

EXAMPLE 4. Let $F(X, Y) = (Y^2 - 2X^3)(Y^2 - 2X^2)(Y^3 - 2X) \in \mathbb{Q}[X, Y]$. Applying `RNPuiseux` over $\mathbb{Q}$ yields three expansions :

$$\begin{aligned} (P_1, Q_1) &= (2X^2, X^0(0 + 2X^2(0 + X(2 + Y)))) \\ &= (2X^2, 4X^3 + 2X^3Y) \\ (P_2, Q_2) &= (4X^3, X^0(0 + X(2 + Y))) \\ &= (4X^3, 2X + 2XY) \\ (P_3, Q_3) &= (X, X^0(0 + X(\sqrt{2} + Y))) \\ &= (X, \sqrt{2}X + XY) \end{aligned}$$

The first two expansions have residue field $\mathbb{Q}$ and ramification index 2 and 3. The third one corresponds to a place with residue field isomorphic to $\mathbb{Q}(\sqrt{2})$. Applying **RNPuiseux** over $\mathbb{Q}(\sqrt{2})$ will result in one more expansion :

$$\begin{aligned}(P_4, Q_4) &= (X, X^0(0 + X(-\sqrt{2} + Y))) \\ &= (X, -\sqrt{2}X + XY).\end{aligned}$$

The first null coefficient of (P_1, Q_1) comes from the exceptional polygon $[(0, 0), (0, 7)]$. The second one corresponds to the fictitious edge of $\mathcal{GN}(F)$ introduced at the first recursive call. This may seem inefficient, but it has no impact on the complexity and clarifies arguments in Section 4.

3.3 Polygon trees

To a function call **RNPuiseux**(L, F) (see Section 3.2), we associate a labelled rooted tree. By definition, the *depth* of a vertex v is the number of edges on the path from the root to v . In particular, the root vertex has depth 0. The tree is constructed recursively from the root vertex as follow (see Figure 2). Even depth vertices correspond to function calls.

- A vertex v of even depth l is labelled with the polygon $\mathcal{P}$, that is $\mathcal{EN}(H)$ for the root vertex ($l = 0$), and $\mathcal{GN}(H)$ for recursive calls ($l > 0$).
- To each Δ of $\mathcal{P}$ corresponds an edge from v to a depth $l + 1$ vertex. Label the edge with Δ (represented by its endpoint).
- A child (depth $l + 1$ vertex) is labelled with the corresponding integer partition $[\phi_\Delta]$ (see the end of Section 1).
- To each choice of root ξ of ϕ_Δ made by the algorithm corresponds an edge from a depth $l + 1$ vertex to a depth $l + 2$ vertex. The edge is labelled with the pair (k, f) , where k is the multiplicity of ξ of and $f = [L(\xi) : L]$.
- Then, we proceed recursively : A depth $l + 2$ vertex is the root vertex of the tree corresponding to the function call **RNPuiseux**($L(\xi), H'$) where H' is the polynomial H' obtained for a choice of edge Δ and a choice of root ξ .

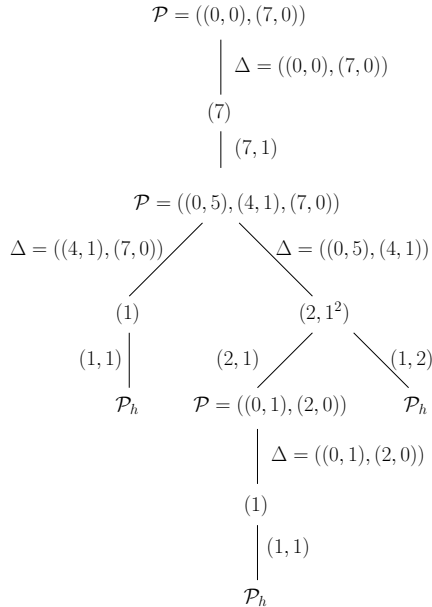


Figure 2: Polygon tree $\mathcal{RT}(\mathbb{Q}, F)$ for Example 4.

The leaves are even depth vertices labelled with polygons that have only one side $\mathcal{P}_h = [(0, 1), (1, 0)]$. Note that the roots ξ are not part of the tree. Since the squarefree factorization is a subproduct of the factorization over L , the labelled tree can be obtained at no significant cost. If l is the depth of the function call tree generated by **RNPuiseux**(L, F), then the labelled tree constructed has depth $2l$.

For a function call **CNPuiseux**(F), we define a similar tree, but in this case, an edge from a partition to a polygon is only labelled with a multiplicity k because the ground field is $\bar{L}$ and all field extension have degree 1.

DEFINITION 12. We denote by $\mathcal{RT}(L, F)$ (resp. $\mathcal{T}(F)$) a tree associated to the function call **RNPuiseux**(L, F) (resp. **CNPuiseux**(F)). In both cases, the tree is called the polygon tree associated to the function call.

It turns out that $\mathcal{T}(F)$ is precisely the symbolic information required to achieve our goal.

PROPOSITION 2. The tree $\mathcal{T}(F)$ can easily be obtained from $\mathcal{RT}(L, F)$ as follow : duplicate f times each edge labelled (k, f) (together with the subtree rooted at this edge) and replace the tag (k, f) by the tag k .

4. GOOD REDUCTION

We consider a polynomial F with coefficients in an algebraic number field K and discuss how to choose a prime number p so that the computation of rational Puiseux expansions modulo p provides enough information to guide floating point computations of Puiseux series, namely $\mathcal{T}(F)$.

We denote by $\mathfrak{o}$ the ring of algebraic integers of K , $\mathfrak{p}$ a prime ideal of $\mathfrak{o}$ and $v_{\mathfrak{p}}$ the corresponding valuation of K . Finally, we define : $\mathfrak{o}_{\mathfrak{p}} = \{\alpha \in K \mid v_{\mathfrak{p}}(\alpha) \geq 0\}$.

Let L be the finite extension generated over K by the Puiseux series coefficients of F . Note that L also contains the coefficients of rational Puiseux expansions computed by **RNPuiseux**. If $\mathfrak{O}$ stands for the ring of algebraic integers of L and $\mathfrak{P}$ for a prime ideal of $\mathfrak{O}$, we introduce $\mathfrak{O}_{\mathfrak{P}} = \{\alpha \in L \mid v_{\mathfrak{P}}(\alpha) \geq 0\}$.

In the sequel, $\mathfrak{P}$ will always denote a prime ideal of $\mathfrak{O}$ dividing $\mathfrak{p}$. The reduction modulo $\mathfrak{P}$ of $\alpha \in \mathfrak{O}_{\mathfrak{P}}$ is represented by $\bar{\alpha}$. We extend this notation to polynomials and fractional power series with coefficients in $\mathfrak{O}_{\mathfrak{P}}$. If $\alpha \in \mathfrak{o}_{\mathfrak{p}}$, since $\mathfrak{P}$ divides $\mathfrak{p}$, reduction modulo $\mathfrak{P}$ and $\mathfrak{p}$ coincide and we shall use the same notation $\bar{\alpha}$.

4.1 Modular reduction of Puiseux series

Our reduction strategy is based on the following definition :

DEFINITION 13. Let p be a prime number and $\mathfrak{p}$ a prime ideal of $\mathfrak{o}$ dividing p . We say that F has local (at $X = 0$) good reduction at $\mathfrak{p}$ if : $F \in \mathfrak{o}_{\mathfrak{p}}[X, Y]$, $p > d = \deg_Y(F)$ and $v_{\mathfrak{p}}(tc(\Delta_F)) = 0$.

Note that if F has local good reduction at $\mathfrak{p}$ and $\mathfrak{P}$ divides $\mathfrak{p}$, then $v_{\mathfrak{P}}(tc(\Delta_F)) = 0$, and so has F at $\mathfrak{P}$. We shall use this fact freely in the sequel.

REMARK 2. Applying our local criterion to all places of $K[X]$, we obtain that $[\Delta_F]$ must be equal to $[\Delta_{\bar{F}}]$ (preservation of the squarefree factorization). This test has been used by the second author as a genus preservation condition

(good reduction, in a classical sense) in his implementation of Trager's algorithm for the integration of algebraic functions [15], publicly available since Maple V.5. This condition was derived from proofs in ([9], Section III.6), using elementary considerations. This test was also brought to the attention of the Computer Algebra community by Trager (unpublished document), as a consequence of a more sophisticated theorem by Fulton [10].

A fundamental result for the reduction strategy is the following consequence of a theorem by Dwork and Robba [8] :

THEOREM 3. *If F has local good reduction at $\mathfrak{p}$, then the Puiseux series coefficients of F above 0 are in $\mathfrak{O}_{\mathfrak{p}}$.*

We emphasize that this result holds for *any* $\mathfrak{P}$ dividing $\mathfrak{p}$.

EXAMPLE 5. *Consider the case $F(X, Y) = Y^2 - X^3(p + X)$ with $p > 2$. Puiseux series above 0 are :*

$$\begin{aligned} S_{1j}(X) &= (-1)^j \sqrt{p} X^{3/2} \left(1 + \frac{X}{p}\right)^{1/2} \\ &= (-1)^j \sqrt{p} X^{3/2} \left(1 + \frac{X}{2p} - \frac{X^2}{8p^2} + \dots\right). \end{aligned}$$

They are obviously not reducible modulo p , but the discriminant criteria of the theorem detects this deficiency.

THEOREM 4. *Let $\{S_i\}_{1 \leq i \leq s}$ be a set of representatives for the cycles of F above 0. Assume that F has local good reduction at $\mathfrak{p}$. Then, $\{\overline{S}_i\}_{1 \leq i \leq s}$ is a set of representatives for the cycles of $\overline{F}$ above 0.*

However, annihilation modulo $\mathfrak{P}$ of Puiseux series coefficients is not totally controlled by our good reduction criterion. If F is irreducible in $\overline{K}[[X]][Y]$, all non-characteristic coefficients may vanish modulo $\mathfrak{P}$, as shown by Proposition 1 (consider for instance the minimal polynomial over $\mathbb{Q}(X)$ of $S(X) = pX + X^{3/2}$). If F is not irreducible, our criterion will also detect cancellation of coefficients that “separate” cycles. This property is contained in Theorem 5.

4.2 Modular reduction of polygon trees

If $F \in \mathfrak{o}_{\mathfrak{p}}[X, Y]$ and $p > d$, algorithms of Section 3 can be applied to the reduction $\overline{F}$ of F modulo $\mathfrak{p}$, so that the notations $\mathcal{T}(\overline{F})$ and $\mathcal{RT}(\mathbb{F}_{p^t}, \overline{F})$ make sense. The computed expansions have coefficients in a finite extension of $\mathbb{F}_p$.

The following result is crucial. It allows to obtain by means of modular computations the symbolic information required by the numerical algorithm in [12] :

THEOREM 5. *If F has local good reduction at $\mathfrak{p}$, then :*

$$\mathcal{T}(F) = \mathcal{T}(\overline{F}).$$

The correspondence between $\mathcal{T}(F)$ and $\mathcal{T}(\overline{F})$ cannot be stated so simply if classical polygons are used instead of generic ones : Non-characteristic coefficients of Puiseux series may vanish upon modular reduction, yielding polygon modifications. Moreover, if the exceptional polygon is replaced by the generic polygon, the good reduction criterion does not detect the cancellation of $F(0, 0)$, as shown by the example $F(X, Y) = (Y + p + X)(Y + 1 + X)$. But the criterion detects a change of root multiplicities. This example justifies the introduction of $\mathcal{EN}(F)$.

4.3 Choosing a good prime

This part is devoted to the choice of a prime ideal $\mathfrak{p}$ such that F has local good reduction at $\mathfrak{p}$.

Assume that $K = \mathbb{Q}(\gamma)$ and let M_γ be the minimal polynomial of γ over $\mathbb{Q}$. The elements of K are represented as polynomials in γ of degree less than $w = [K : \mathbb{Q}]$ with coefficients in $\mathbb{Q}$. Up to a change of variable in M_γ and the coefficients of F , we suppose that γ belongs to $\mathfrak{o}$, namely $M_\gamma \in \mathbb{Z}[T]$.

DEFINITION 14. *Let P be a multivariate polynomial of $K[\underline{X}]$. There exists a unique pair (H, c) with $H \in \mathbb{Z}[T, \underline{X}]$, $c \in \mathbb{N}$, $\deg_T(H) < w$ and $P(\underline{X}) = H(\gamma, \underline{X})/c$, where c is minimal. The polynomial H is called the numerator of P and is denoted $\text{num}(P)$. The integer c is called the denominator of H and is written $\text{denom}(P)$. We define the size of P as follow : $ht(P) = \max\{\log_2 c, \log_2 \|R\|_\infty\}$.*

Denoting $F_n = \text{num}(F)$ and $b = \text{denom}(F)$, we have $F(X, Y) = \frac{F_n(\gamma, X, Y)}{b}$. We are left with the problem of finding a prime number p and a prime ideal $\mathfrak{p}$ of $\mathfrak{o}$ dividing p such that :

(C₁) $p > d$.

(C₂) p does not divide b .

(C₃) We can determine an explicit representation of a prime ideal $\mathfrak{p}$ of $\mathfrak{o}$ dividing p , so that a morphism $\mathfrak{o} \rightarrow \mathfrak{o}/\mathfrak{p} \cong \mathbb{F}_{p^t}$ can be effectively computed.

(C₄) $tc(\Delta_F) \not\equiv 0$ modulo $\mathfrak{p}$.

Condition (C₁) and (C₂) are easily checked out. We deal with condition (C₃) in a standard fashion. Let $\overline{M}$ be any irreducible factor of M_γ in $\mathbb{F}_p[T]$ and M be a lifting of $\overline{M}$ in $\mathbb{Z}[T]$. It is well-known that if p is a prime number not dividing the index $e_\gamma = [\mathfrak{o} : \mathbb{Z}(\gamma)]$ and if $\overline{M}$ is any irreducible factor of M_γ in $\mathbb{F}_p[T]$, then the ideal $\mathfrak{p} = (p, M(\gamma))$ of $\mathfrak{o}$ is prime [3]. Hence, elements of $\mathfrak{o}$ can be reduced by means of the morphism $\mathfrak{o} \rightarrow \mathfrak{o}/\mathfrak{p} \cong \mathbb{F}_p[T]/(\overline{M}) \cong \mathbb{F}_{p^t}$, where $t = \deg \overline{M}$. Computing e_γ is a non-trivial task, and so is the computation of generators of prime ideals dividing p when p divides e_γ . If e_γ is unknown, it is sufficient to choose p so that it does not divide Δ_{M_γ} , since e_γ divides Δ_{M_γ} .

In practice, $\overline{M}$ is chosen amongst the factors of $\overline{M_\gamma}$ of smallest degree. Moreover, it is worth trying a few primes p in order to reduce t , the case $t = 1$ being the most favorable.

As for (C₄), deterministic and randomized strategies are studied in the next subsections. In order to simplify the analysis, we replace condition (C₄) by the following stronger condition :

(C'₄) $\text{Norm}_{K/\mathbb{Q}}(tc(\Delta_F)) \not\equiv 0$ modulo p .

If (C₁) to (C'₄) are verified, then for all prime ideals $\mathfrak{p}$ dividing p , F has local good reduction at $\mathfrak{p}$. In practice, though, we do not recommend to use (C'₄). Finally, we introduce the notation $N_F = b \cdot |\text{Norm}_{K/\mathbb{Q}}(tc(\Delta_F))| \cdot |\Delta_{M_\gamma}|$. Conditions (C₁) to (C'₄) are equivalent to :

(C₅) $p > d$ and $N_F \not\equiv 0$ modulo p .

4.3.1 Deterministic strategy

We determine a bound B such that, for all prime numbers $p > B$, condition (C₅) is satisfied. We first give two lemmas :

LEMMA 1. *The discriminant $\Delta_{F_n} \in \mathbb{Z}[X, T]$ of F_n with respect to Y satisfies :*

$$\|\Delta_{F_n}\|_\infty \leq (2d - 1)! d^d [(w + 1)(n + 1)]^{2d-2} \|F_n\|_\infty^{2d-1}.$$

We denote by $R_F(T)$ the numerator of $tc(\Delta_F)$. Note that $\text{denom}(tc(\Delta_F))$ is a power of b dividing b^{2d-1} .

LEMMA 2. *Define :*

$$\begin{aligned} B_0 &= \|\Delta_{F_n}\|_\infty (\|M_\gamma\|_\infty + 1)^{(w-1)(2d-1)-w+1}, \\ B_1 &= (w+1)^{(2w-1)/2} \|M_\gamma\|_\infty^{w-1} B_0^w, \\ B_2 &= w^w (w+1)^{(2w-1)/2} \|M_\gamma\|_\infty^{2w-1}. \end{aligned}$$

Then, $|\text{Norm}_{K/\mathbb{Q}}(R_F(\gamma))| \leq B_1$ and $e_\gamma \leq |\Delta_{M_\gamma}| \leq B_2$.

PROPOSITION 3. *Set $B = \max\{b, d+1, B_1, B_2\}$. Then, for all $p > B$, condition (C_5) is verified. Moreover, there exists a prime $p > B$ with size $ht(p)$ in :*

$$O(wd(w \, ht(M_\gamma) + ht(F) + \log(wnd))).$$

4.3.2 Probabilistic strategies

We begin with a ‘‘Monte-Carlo’’ method, best described by the following algorithm. We need two auxiliary procedures : The function call **RandomPrime**(A, C) returns a random prime number in the real interval $[A, C]$. We assume that the primes returned are uniformly distributed in the set of primes belonging to $[A, C]$ (see [14], Section 7.5). The function **Nextprime** gives the smallest prime larger than the argument.

MCGoodPrime(d, B', ϵ)

Input : d : The degree in Y of the polynomial F .
 B' : A real number such that prime factors of N_F are less or equal to B' .
 ϵ : A real number with $0 < \epsilon \leq 1$.

Output : A prime number p satisfying (C_5) with probability at least $1 - \epsilon$.

Begin

If $B' < 100$ **then** **Return** **NextPrime**(B') **End**
 $K \leftarrow 6 \ln B' / (\epsilon \ln \ln B') + 2d / \ln d$
 $C \leftarrow \max\{2d, K(\ln K)^2\}$
Return **RandomPrime**($d+1, C$)

End.

PROPOSITION 4. *MCGoodPrime*(d, B', ϵ) *returns a prime p satisfying $ht(p) \in O(\log \log B' + \log d + \log \epsilon^{-1})$. In particular, if $B' = \max\{b, B_1, B_2\}$ (see Lemma 2), then : $ht(p) \in O(\log(dw \log n) + \log ht(F) + \log ht(M_\gamma) + \log \epsilon^{-1})$. The probability that p does not satisfy (C_5) is less than ϵ .*

Finally, we consider a ‘‘Las Vegas’’ flavoured method :

LVGoodPrime(F, M_γ)

Input : F : A polynomial as in Section 1.
 M_γ : A monic irreducible polynomial in $\mathbb{Z}[T]$.

Output : A prime number p satisfying (C_5) .

Begin

$d \leftarrow \deg_Y(F)$
 $N_F \leftarrow \text{denom}(F) \cdot |\text{Norm}_{K/\mathbb{Q}}(R_F(T))| \cdot |\text{Disc}(M_\gamma)|$
 $B' \leftarrow \max\{\text{denom}(F), |\text{Norm}_{K/\mathbb{Q}}(R_F(T))|, |\text{Disc}(M_\gamma)|\}$
Repeat
 $p \leftarrow \text{MCGoodPrime}(d, B', 1/2)$
until p **does not divide** N_F **End**
Return p

End.

PROPOSITION 5. *LVGoodPrime*(F, M_γ) *returns a prime p satisfying $ht(p) \in O(\log(dw \log n) + \log ht(F) + \log ht(M_\gamma))$ and (C_5) with an average number of iterations less than 2.*

The computation of B' and N_F may have a significant cost. In our monodromy context [12], though, we need to determine Δ_F anyway.

5. COMPLEXITY OVER A FINITE FIELD

In this section, L denotes a finite field and F belongs to $L[X, Y]$. Otherwise, we keep the notations and assumptions of Section 1. We denote by $p > d$ the characteristic of L . We also define $t_0 = [L : \mathbb{F}_p]$. This section is devoted to the proof of the following theorem :

THEOREM 6. *Assuming that FFT-based polynomial multiplication over finite fields is used, RNPuiseux can compute singular parts of a system of rational Puiseux expansions above 0 of F in $O^*(d^3 n^2 + d^2 n t_0 \log p)$ field operations in L .*

As usual, the notation O^* hides logarithmic factors.

REMARK 3. *This result improves the bound of [7], which is in $O(d^6 n^2)$ field operations. Our estimates include factorization cost, while Duval relies on the D5 system to avoid factorizations. The gain comes from :*

- truncation of powers of X in the course of the algorithm (see Proposition 6),
- reducing transformations to shifts of univariate polynomials, for which fast methods are available (Proposition 7),
- a bound for δ_F (see below and Proposition 9).

We first introduce notations and make some assumptions :

- $\{R_i\}_{1 \leq i \leq \rho}$ with $R_i(T) = (X_i(T), Y_i(T))$ stands for singular parts of a system of rational Puiseux expansions,
- (r_i, e_i, f_i) , $1 \leq i \leq \rho$ are respectively the regularity index, the ramification index and the coefficient field degree over L of R_i .
- For each rational Puiseux expansion R_i , we can deduce $e_i f_i$ Puiseux series denoted $S_{ijk}(X)$, $1 \leq k \leq e_i$, $1 \leq j \leq f_i$.
- We define $\delta_F = \sum_{i=1}^{\rho} f_i r_i$.
- L_t denotes an extension of degree t of L .
- $M(N)$ will denote the number of field operations in L_t needed to compute the product of two polynomials in $L_t[Z]$ of degree no larger than N . We recall that $M(N) \in O(N^2)$ for classical arithmetic and $M(N) \in O(N \log N \log \log N) \subset O^*(N)$ if FFT-based multiplication is used.
- A field operation in L_t can be done using $O(M(t) \log t)$ field operations in L .

We refer the reader to [17] for assertions regarding the complexity of operations over finite fields.

REMARK 4. *It is worth noting that δ_F represents essentially the number of elements of L necessary to represent the Y_i . Indeed, each Y_i has at most $r_i + 1$ nonzero coefficients and each of those may be represented by at most f_i elements of L . Assume that a dense representation is used for the truncated power series Y_i (for instance, a vector of $r_i + 1$ elements of L_{f_i}) and assume in turn that the coefficients of Y_i are represented by vectors of f_i elements of L . Then, the size of the output is $\delta_F + \sum_i f_i$, which is bounded from below by δ_F and bounded from above by $\delta_F + d$.*

We split the proof into several results.

PROPOSITION 6. *Systems of rational Puiseux expansions for F and $\tilde{F}^{\delta_F}$ above 0 have the same singular parts (up to trivial changes of the parameter T). Moreover, singular parts of rational Puiseux expansions of F can be computed by applying RNPuiseux to $\tilde{F}^{\delta_F}$ and truncating polynomials H modulo X^{δ_F+1} at each stage of the algorithm.*

PROPOSITION 7. *The substitutions needed to compute the singular parts of a system of rational Puiseux expansions of F requires $O(\delta_F^2 M(d)^2 \frac{\log d}{d})$ field operations in L .*

PROPOSITION 8. *All factorizations of characteristic polynomials required by RNPuiseux can be computed with an expected number of $O(\delta_F \log d [M(d^2) + t_0 \log p M(d) \log d])$ field operations in L .*

It is interesting to bound first the number of operations in L in terms of the output size, namely δ_F .

THEOREM 7. *The number of operations in L required to compute singular parts of rational Puiseux expansions of F above 0 is in $O(\delta_F M(d) [\delta_F \frac{M(d)}{d} + M(d) + t_0 \log p])$.*

PROPOSITION 9. $\delta_F \leq v_X(\Delta_F) \leq n(2d - 2)$

Theorem 6 is now a trivial consequence of the last two results.

6. BIT-COMPLEXITY

Let F be a polynomial of $K[X, Y]$, where K is an algebraic number field represented as in Section 4.3. We recall that $[K : \mathbb{Q}] = w$. We study the bit-complexity of the computation of $\mathcal{T}(F)$. We estimate only word operations generated by arithmetic operations in various coefficient fields. Assuming some care is taken in the implementation, (for instance, access to coefficients of polynomial should be achieved in constant time), results below should give a realistic upper bound for the behaviour of an actual computer program.

Our bounds for randomized algorithms do not include the cost of generating prime numbers, nor the cost of computing bounds given by our formula.

We assume that elements of $\mathbb{F}_p$ are represented by nonnegative integers. In order to simplify expressions, we assume that fast arithmetic is used for integer arithmetic as well as polynomial arithmetic over finite fields.

THEOREM 8. *Given ϵ with $0 < \epsilon \leq 1$, there exists a probabilistic Monte Carlo algorithm that computes $\mathcal{T}(F)$ with an expected number of $O(d^3 n^2 w^2 \log^2 \epsilon^{-1} [ht(M_\gamma) + ht(F)])$ word operations and probability of error less than ϵ .*

7. CONCLUSION

This paper summarizes the results we have obtained in [13] regarding the symbolic part of our program towards a fast and reliable method to compute Puiseux series with floating point coefficients. In particular, the criterion ensuring preservation of polygon trees is essential. Along the path, we have derived improved complexity bounds for computations over finite fields. Although not optimal, these bounds are quite reasonable, i.e. quadratic in the output size, up to a factor d . Bit-complexity estimates for the Monte-Carlo version of the first step of our symbolic-numeric method confirm that the coefficient growth of pure symbolic Newton-Puiseux algorithm is avoided. Complexity bounds for the Las Vegas and deterministic version can be obtained similarly. Applying our reduction criterion for all places of $K[X]$ provides analogous bit-complexity bounds for the computation of the genus of an algebraic curve defined over an algebraic number field.

We are actively working on the numerical part of the algorithm, and in particular on error control, as well as on an improved implementation of both parts.

Acknowledgments

The authors would like to thank Mark van Hoeij for stimulating conversations and the referees for carefully proofreading the paper. We are also grateful to an anonymous referee of [12] for valuable suggestions.

8. REFERENCES

- [1] E. Brieskorn and H. Knörrer. *Plane Algebraic Curves*. Birkhäuser, 1986.
- [2] C. Chevalley. *Introduction to the Theory of Algebraic Functions of One Variable*, volume 6 of *Mathematical Surveys*. AMS, 1951.
- [3] H. Cohen. *A Course in Computational Algebraic Number Theory*. Springer-Verlag, 1993.
- [4] B. Deconinck and M. S. Patterson. Computing the Abel Map. *preprint*, 2007.
- [5] B. Deconinck and M. van Hoeij. Computing Riemann matrices of algebraic curves. *Phys. D*, 152/153:28–46, 2001.
- [6] D. Duval. *Diverses questions relatives au calcul formel avec des nombres algébriques*. PhD thesis, Université de Grenoble, 1987. Thèse d'Etat.
- [7] D. Duval. Rational Puiseux Expansions. *Compositio Mathematica*, 70:119–154, 1989.
- [8] B. Dwork and P. Robba. On natural radii of p -adic convergence. *Trans. Amer. Math. Soc.*, 256:199–213, 1979.
- [9] M. Eichler. *Introduction to the Theory of Algebraic Numbers and Functions*. Academic Press, 1966.
- [10] W. Fulton. Hurwitz Schemes and Irreducibility of Moduli of Algebraic Curves. *Annals of Mathematics*, 90:542–575, 1969.
- [11] H. T. Kung and J. F. Traub. All algebraic functions can be computed fast. *J. ACM*, 25(2):245–260, 1978.
- [12] A. Poteaux. Computing monodromy groups defined by plane algebraic curves. In *Proceedings of the 2007 International Workshop on Symbolic-numeric Computation*, pages 36–45, New-York, 2007. ACM.
- [13] A. Poteaux and M. Rybowicz. Towards a Symbolic-Numeric Method to Compute Puiseux Series: The Modular Part, 2008. <http://arxiv.org/abs/0803.3027>.
- [14] V. Shoup. *A Computational Introduction to Number Theory*. Cambridge University Press, 2005.
- [15] B. M. Trager. *Integration of Algebraic Functions*. PhD thesis, Department of EECS MIT, 1984.
- [16] M. van Hoeij. An Algorithm for Computing an Integral Basis in an Algebraic Function Field. *Journal of Symbolic Computation*, 18:353–363, 1994.
- [17] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, Cambridge, 1999.
- [18] P. G. Walsh. A Polynomial-time Complexity Bound for the Computation of the Singular Part of an Algebraic Function. *Mathematics of Computation*, 69:1167–1182, 2000.
- [19] O. Zariski. *Le problème des modules pour les branches planes*. Hermann, Paris, 1981.

Multi-modular Algorithm for Computing the Splitting Field of a Polynomial

Guénaél Renault
INRIA, Paris-Rocquencourt, SALSA Project
UPMC, Univ. Paris 06, LIP6
CNRS, UMR 7606, LIP6
UFR Ingénierie 919, LIP6 Passy-Kennedy,
Case 169, 4, Place Jussieu, F-75252 Paris
guenael.renault@lip6.fr

Kazuhiro Yokoyama
Rikkyo University
3-34-1 Nishi Ikebukuro, Toshima-ku
Tokyo 171-8501, Japan
yokoyama@rkmath.rikkyo.ac.jp

ABSTRACT

Let f be a univariate monic integral polynomial of degree n and let $(\alpha_1, \dots, \alpha_n)$ be an n -tuple of its roots in an algebraic closure $\mathbb{Q}$ of $\mathbb{Q}$. Obtaining an algebraic representation of the splitting field $\mathbb{Q}(\alpha_1, \dots, \alpha_n)$ of f is a question of first importance in effective Galois theory. For instance, it allows us to manipulate symbolically the roots of f . In this paper, we propose a new method based on multi-modular strategy. Actually, we provide algorithms for this task which return a triangular set encoding the *splitting ideal* of f . We examine the ability/practicality of the method by experiments on a real computer and study its complexity.

Categories and Subject Descriptors

I.1 [Computing Methodologies]: Symbolic and algebraic manipulations

General Terms

Algorithms, Theory

Keywords

Galois theory, splitting field

1. INTRODUCTION

In [18] the authors proposed an approach for computing the splitting field of a monic integral polynomial f . This approach is based on *indeterminate coefficients* strategy and Hensel lifting. It takes as input the action of the Galois group of f over approximation of roots of f in a p -adic number field $\mathbb{Q}_p$ (or one of its extensions). To compute the Galois group G_f of f over $\mathbb{Q}$, the approach of p -adic approximation is very practical and efficient (see [21, 10, 9]).

In the approach of [18], the authors did not use all the data obtained during Galois group computation. Also, on

some particular examples, the method was not so efficient. By experimentations, we discovered that using these data is a very low time consuming. Thus, in order to use a large part of these data and obtain a compromise for this approach we propose here a multi-modular approach for computing the splitting field of f . Moreover, the recent version 2.13 of the *computer algebra system* MAGMA [5] provides a new implementation (by Fieker and Klüners) of the Galois group computation based on p -adic approximations where it becomes easy to access the data computed during this procedure. Also, since the new computer architectures are now based on multi-core processors it is important to study new algorithms which can benefit from these new features. For all these reasons, a multi-modular strategy has to be studied.

The key of the multi-modular strategy proposed here comes from the following assertion: From data obtained during the computation of the permutation action of G_f over approximate roots of f modulo a prime p_1 , we can easily obtain the action of the same permutation representation over approximate roots of f modulo another prime p_2 (see Section 3).

From this action over approximate roots of f modulo different primes we compute approximations, modulo the same primes, of the Gröbner basis $\mathcal{G}$ of the splitting ideal $\mathcal{M}$, that is the ideal of all the algebraic relations of the roots of f . Then we reconstruct it by *Chinese Remainder Theorem* (See Section 4). Thus, the splitting field of f is given by $\mathbb{Q}[x_1, \dots, x_n]/\mathcal{M}$; let us remark that it is easy to perform arithmetic operations in this algebra. Moreover, in general, expressions by primitive elements tend to suffer "expression swell", that is, huge coefficients appear and those harm the efficiency. So, for our purpose, simple extension does not seem suited.

In order to compute the approximate projections of $\mathcal{M}$ in different p -adic fields, we use the knowledge of certain algebraic structures, the action of G_f over the p -adic approximation of roots of f and a theoretical form of $\mathcal{G}$ given by the corresponding *computation scheme*, a very useful object introduced in [18] (see also [17]). The computation scheme gives sparse forms with indeterminate coefficients for the polynomials in $\mathcal{G}$ and techniques to avoid some computations (see Section 3.3). In Section 3 we show how to interpolate these sparse form by adaptation of the formulae given in [6] and modular computations. From these theoretical forms we deduce, in the same section, the best bounds in our knowledge for the coefficients of a basis of $\mathcal{M}$.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

In Section 4 we give basic discussion on efficiency of multi-modular strategy in a general form, and present concrete algorithms for our subject corresponding to this multi-modular strategy with effective tests for *correctness*. Then, based on the algorithms, we give certain results about the multi-modular strategy's theoretical efficiency. Section 5 is devoted to the experiments, by which the practicality of the multi-modular strategy is examined.

2. PRELIMINARIES

We provide necessary notions and summarize some results of [21] and [18].

2.1 Splitting Field and Galois Group over $\mathbb{Q}$

Let $f(x)$ be a monic square-free integral polynomial of degree n and $\underline{\alpha}$ the set of all its roots in an algebraic closure $\bar{\mathbb{Q}}$ of $\mathbb{Q}$. The splitting field K_f of f is the extension field $\mathbb{Q}(\underline{\alpha})$ obtained by adjoining $\underline{\alpha}$ to $\mathbb{Q}$. The group G_f of $\mathbb{Q}$ -automorphisms of K_f acts faithfully on $\underline{\alpha}$, thus one can consider the permutation representation G_f of this group. Fixing a numbering of the roots $\underline{\alpha} = \{\alpha_1, \dots, \alpha_n\}$ of f , G_f is viewed as a subgroup of S_n . The group G_f is called the Galois group of f .

To express K_f symbolically, we consider the epimorphism $\phi : \mathbb{Q}[x_1, \dots, x_n] \ni x_i \mapsto \alpha_i \in K_f$ of $\mathbb{Q}$ -algebras. For simplicity, we write $X = \{x_1, \dots, x_n\}$. Then K_f is represented by the residue class ring $\mathcal{A}$ of the polynomial ring $\mathbb{Q}[X]$ factored by the kernel $\mathcal{M}$ of ϕ . We call $\mathcal{M}$ the *splitting ideal of f associated with the assignment of the roots $\alpha_1, \dots, \alpha_n$* . In this setting, computing K_f means to compute a *Gröbner basis* $\mathcal{G}$ of $\mathcal{M}$ (see [4]). Now we fix the lexicographic order $<$ on terms with $x_1 < \dots < x_n$, then the reduced Gröbner basis of $\mathcal{M}$ coincides with the generating set $\{g_1, g_2, \dots, g_n\}$ obtained by *successive extensions*, that is, for each i ,

1. g_i is a polynomial in $x_1, \dots, x_i$ and monic with respect to x_i , and
2. $\mathbb{Q}(\alpha_1, \dots, \alpha_i) \cong \mathbb{Q}[x_1, \dots, x_i] / \langle g_1, \dots, g_i \rangle$,

where $\langle F \rangle$ denotes the ideal generated by an element or a set F . This implies that g_i is an irreducible factor of $f(x_i)$ over $\mathbb{Q}[x_1, \dots, x_{i-1}] / \langle g_1, \dots, g_{i-1} \rangle$ such that $g_i(\alpha_1, \dots, \alpha_i) = 0$. Thus this reduced Gröbner basis can be obtained by “*algebraic factoring methods*” (see [3]) and is said to be a *triangular basis* (see [12, 6]). For a Gröbner basis $\mathcal{G} \subset \mathbb{Q}[X]$ and a polynomial P , let $\text{NF}(P, \mathcal{G})$ denote the normal form of P in $\mathbb{Q}[X]$ with respect to $\mathcal{G}$ (see [4]).

The group S_n acts naturally on $\mathbb{Q}[X]$ with $x_i^\sigma = x_{i\sigma}$ for $1 \leq i \leq n$ and $\sigma \in S_n$. Thus G_f is the $\mathbb{Q}$ -automorphisms group of $\mathcal{A}$ denoted by $\text{Aut}_{\mathbb{Q}}(\mathcal{A})$ (see [3, 1]). We use the following notation for groups: For a group G acting on a set $\mathcal{S}$, the stabilizer in G of an element or a subset A of $\mathcal{S}$ is denoted by $\text{Stab}_G(A)$, i.e. $\text{Stab}_G(A) = \{\sigma \in G : A^\sigma = A\}$. If G is the full symmetric group on $\mathcal{S}$, we simply write $\text{Stab}(A)$ for $\text{Stab}_G(A)$. We denote by $\text{Stab}_G([a_1, \dots, a_k])$ the point-wise stabilizer of a subset $A = \{a_1, \dots, a_k\}$ of $\mathcal{S}$, i.e. $\text{Stab}_G([a_1, \dots, a_k]) = \{\sigma \in G \mid a_i^\sigma = a_i, \forall i \in \{1, \dots, k\}\}$. The set of right cosets of H in G is denoted by $H \backslash G$ and the set of all representatives of $H \backslash G$ by $H \backslash \backslash G$.

Definition 1. We call the ideal generated by the polynomials $t_1 + a_1, \dots, t_n + (-1)^{n-1} a_n$, where t_i is the i -th elementary symmetric function on X and $f(x) = x^n + a_1 x^{n-1} + \dots + a_n$, the *universal splitting ideal* of f and denote it by

$\mathcal{M}_0$. We call the residue class ring $\mathbb{Q}[X] / \mathcal{M}_0$ the *universal splitting ring of f over $\mathbb{Q}$* and denote it by $\mathcal{A}_0$.

With respect to the fixed order $<$, the reduced Gröbner basis of $\mathcal{M}_0$ is composed of the n *Cauchy's modules* of f (see [19]) and it is called the *standard generating set*. Since S_n stabilizes $\mathcal{M}_0$, S_n also acts faithfully on $\mathcal{A}_0$, i.e. $S_n \subset \text{Aut}_{\mathbb{Q}}(\mathcal{A}_0)$. We have the following theorem (see [16, 2, 21] for details and other references).

THEOREM 2.1. *There is a one-to-one correspondence between the set of all primitive idempotents of $\mathcal{A}_0$ and the set of all prime divisors of $\mathcal{M}_0$. Let m be the primitive idempotent corresponding to the fixed prime divisor $\mathcal{M}$. Then, $G_f = \text{Stab}(\mathcal{M}) = \text{Stab}(m)$ and $\mathcal{M}^\sigma = \{g \in \mathbb{Q}[X] \mid gm^\sigma = 0 \text{ in } \mathcal{A}_0\}$. Moreover, we have $\mathcal{M}_0 = \bigcap_{\sigma \in G_f \backslash \backslash S_n} \mathcal{M}^\sigma$ and $\mathcal{A}_0 = \bigoplus_{\sigma \in G_f \backslash \backslash S_n} m^\sigma \mathcal{A}_0 = \bigoplus_{\sigma \in G_f \backslash \backslash S_n} \mathbb{Q}[X] / \mathcal{M}^\sigma$.*

2.2 Splitting Field and Modular Computation

Now we consider the relation between the splitting ring over $\mathbb{Q}$ and that over a p -adic field $\mathbb{Q}_p$. The n -tuple $\underline{\alpha} = \{\alpha_1, \dots, \alpha_n\}$ and the splitting ideal $\mathcal{M}$ associated with the assignment x_i to α_i are fixed. The primitive idempotent of $\mathcal{A}_0$ corresponding to $\mathcal{M}$ is denoted by m . For a prime integer p , we denote by $\mathbf{Z}_p^0$ (resp. $\mathbf{Z}_p$) the localization of $\mathbf{Z}$ at p (resp. the completion of $\mathbf{Z}_p^0$). We denote by π_p the projection from $\mathbf{Z}_p[X]$ to $\mathbb{F}_p[X]$ (the natural extension of the projection from $\mathbf{Z}$ to $\mathbb{F}_p$). From now on, we will consider prime numbers p satisfying the following property:

$$\mathbb{P} : \quad \pi_p(f) \text{ is square-free.}$$

Let $\bar{\mathcal{M}}_0^p$ denote the ideal $\pi_p(\mathcal{M}_0 \cap \mathbf{Z}_p^0[X])$ in $\mathbb{F}_p[X]$ and $\mathcal{G}_0$ denote the standard generating set of $\mathcal{M}_0$. By construction, the Cauchy's modules of f are polynomials with integral coefficients and monic in their greatest variable. Thus, the set $\pi_p(\mathcal{G}_0)$ is a Gröbner basis of $\bar{\mathcal{M}}_0^p$. Moreover, $\mathcal{G}_0$ is a Gröbner basis of the universal splitting ideal $\mathbb{Q}_p \otimes_{\mathbb{Q}} \mathcal{M}_0$ of f as a polynomial with coefficients in $\mathbb{Q}_p$ and that of $\mathbf{Z}_p[X] \otimes_{\mathbf{Z}_p^0} (\mathcal{M}_0 \cap \mathbf{Z}_p^0[X])$ over $\mathbf{Z}_p$. The ideal $\mathbb{Q}_p \otimes_{\mathbb{Q}} \mathcal{M}_0$ is denoted by $\mathcal{M}_0^{(p, \infty)}$. We denote $\mathbb{F}_p[X] / \bar{\mathcal{M}}_0^p$ by $\bar{\mathcal{A}}_0^p$ and $\mathbb{Q}_p[X] / \mathcal{M}_0^{(p, \infty)}$ by $\mathcal{A}_0^{(p, \infty)}$. We have the following result (see [21, 18]).

THEOREM 2.2. *We have the following assertions:*

1. *The projection π_p gives a one-to-one correspondence between the set of all primitive idempotents of $\mathcal{A}_0^{(p, \infty)}$ and that of $\bar{\mathcal{A}}_0^p$. Moreover, for each pair $(\bar{m}^{(p)}, m^{(p, \infty)})$ of corresponding primitive idempotents, $\text{Stab}(\bar{m}^{(p)}) = \text{Stab}(m^{(p, \infty)})$.*
2. *The idempotent m of $\mathcal{A}_0$ is also an idempotent of $\mathcal{A}_0^{(p, \infty)}$. Let $\bar{m}^{(p)}$ be a component of $\pi_p(m)$ and $m^{(p, \infty)}$ the primitive idempotent of $\mathcal{A}_0^{(p, \infty)}$ corresponding to $\bar{m}^{(p)}$. Then $\text{Stab}(m)$ contains $\text{Stab}(\bar{m}^{(p)}) (= \text{Stab}(m^{(p, \infty)}))$ and $\text{Stab}(\pi_p(m)) = \text{Stab}(m)$. Moreover, if we denote $\text{Stab}(\bar{m}^{(p)}) \backslash \backslash \text{Stab}(m)$ by $\mathcal{S}$ then $\pi_p(m) = \sum_{\sigma \in \mathcal{S}} \bar{m}^\sigma$ and $m = \sum_{\sigma \in \mathcal{S}} m^{(p, \infty)\sigma}$.*

Now we fix a component $\bar{m}^{(p)}$ of $\pi_p(m)$ and its corresponding idempotent $m^{(p, \infty)}$ of $\mathcal{A}_0^{(p, \infty)}$. Let $\bar{\mathcal{M}}^p$ be the maximal ideal of $\mathbb{F}_p[X]$ corresponding to $\bar{m}^{(p)}$ and $\mathcal{M}^{(p, \infty)}$ the maximal ideal of $\mathbb{Q}_p[X]$ corresponding to $m^{(p, \infty)}$. Moreover, let $\mathcal{G}^{(p, \infty)}$ and $\bar{\mathcal{G}}^{(p)}$ be the reduced Gröbner basis of $\mathcal{M}^{(p, \infty)}$ and that of $\bar{\mathcal{M}}^p$ respectively.

Definition 2. Let $\mathcal{G}^{(p,\infty)} = \{g_1^{(p,\infty)}, \dots, g_n^{(p,\infty)}\}$. For a positive integer k , we call the polynomials set $\{g_1^{(p,\infty)} \bmod p^{k+1}, \dots, g_n^{(p,\infty)} \bmod p^{k+1}\}$ the k -th approximation to the basis $\mathcal{G}^{(p,\infty)}$ and denote it by $\mathcal{G}^{(p,k)}$. Note that $\mathcal{G}^{(p,0)} = \bar{\mathcal{G}}^{(p)}$.

Approximations of the roots of f

The Gröbner basis $\bar{\mathcal{G}}^{(p)}$ can be lifted to $\mathcal{G}^{(p,\infty)}$ by Hensel construction based on *quadratic iteration*.

THEOREM 2.3. [21, 18] *The reduced Gröbner basis $\mathcal{G}^{(p,\infty)}$ of the ideal $\mathcal{M}^{(p,\infty)}$ with respect to $\prec$ is contained in $\mathbf{Z}_p[X]$, and $\bar{\mathcal{G}}^{(p)}$ is lifted uniquely to $\mathcal{G}^{(p,\infty)}$ by Hensel construction.*

REMARK 3. From $\bar{\mathcal{G}}^{(p)}$, we can construct the approximate Gröbner basis $\mathcal{G}^{(p,k)}$ for any integer k . As soon as we have $\mathcal{G}^{(p,k)}$, we can compute with the roots of f in $\mathbb{Z}/p^{k+1}\mathbb{Z}$ by computing normal forms modulo this basis. Thus, in the sequel, the expression approximations of the roots of f modulo p^{p+1} will mean that we have such a Gröbner basis $\mathcal{G}^{(p,k)}$.

REMARK 4. As to the Gröbner basis $\mathcal{G}$, the denominators of its elements are related to the discriminant $d(f)$ of f . (See Section 3.4.) For each prime p satisfying the property $\mathbb{P}$, the square-freeness of $\pi_p(f)$ implies $\pi_p(d(f)) \neq 0$, and thus, it follows that $\pi_p(\mathcal{G})$ is well-defined, that is, p does not divide any of the denominators.

Now we will study the construction of $\mathcal{G}$ by *Chinese Remainder Theorem*.

3. MODULAR CONSTRUCTION OF $\mathcal{G}$

In this section, we fix a *splitting ideal* $\mathcal{M}$ of f , the corresponding idempotent m and its stabilizer, the Galois group G_f represented as a sub-group of S_n . We denote by $Z_{\mathbb{K}}(I)$ the algebraic variety over a field $\mathbb{K}$ associated to an ideal I in a polynomial ring. Let $E = \{e_1 < \dots < e_s\}$ be a subset of $\{1, \dots, n\}$ and $\underline{\gamma} = (\gamma_1, \dots, \gamma_n)$ an element of $Z_{\bar{\mathbb{Q}}}(\mathcal{M})$. We denote by $\underline{\gamma}(E)$ the projection of $\underline{\gamma}$ on the indexes given by E (i.e. $(\gamma_{e_1}, \dots, \gamma_{e_s})$) and $Z_{\bar{\mathbb{Q}}}(\mathcal{M})(E) = \{\underline{\gamma}(E) : \underline{\gamma} \in Z_{\bar{\mathbb{Q}}}(\mathcal{M})\}$.

3.1 Approximation of $\mathcal{G}$

Let $\mathcal{G} = \{g_1, \dots, g_n\}$ be the Gröbner basis of $\mathcal{M}$, we will describe here how to compute the coefficients of polynomials g_i 's by indeterminate coefficient strategy and multi-modular computation.

More precisely, by the knowledge of a special subset E_i of $\{1, \dots, n\}$, we can deduce an equation which defines the polynomial g_i (see Section 3.3):

$$g_i(\underline{\gamma}) = 0 \text{ for every } \underline{\gamma} \in Z_{\bar{\mathbb{Q}}}(\mathcal{M})(E_i). \quad (3.1)$$

We can replace the variety $Z_{\bar{\mathbb{Q}}}(\mathcal{M})(E_i)$ by $Z_{\bar{\mathbb{Q}}_p}(\mathbb{Q}_p \otimes_{\mathbb{Q}} \mathcal{M})(E_i)$, where p is a prime satisfying $\mathbb{P}$. Then, by using approximations of roots, we obtain the same equation but for the approximation of g_i . To do that, we need to recall some results (see [21]).

Let p be a prime integer satisfying $\mathbb{P}$, $\bar{m}^{(p)}$ a component of $\pi_p(m)$, $\bar{\mathcal{M}}^p$ its corresponding maximal ideal of $\mathbb{F}_p[X]$ and $\mathcal{M}^{(p,\infty)}$ its corresponding maximal ideal of $\mathbb{Q}_p[X]$ which is a divisor of $\mathbb{Q}_p[X] \otimes_{\mathbb{Q}} \mathcal{M}$.

PROPOSITION 3.1. *Let $\mathcal{S} = \text{Stab}(\bar{m}^{(p)}) \setminus \text{Stab}(m)$. Then $\mathbb{Q}_p \otimes_{\mathbb{Q}} \mathcal{M} = \cap_{\sigma \in \mathcal{S}} (\mathcal{M}^{(p,\infty)})^{\sigma}$, and $\pi_p(\mathcal{M} \cap \mathbf{Z}_p^0) = \cap_{\sigma \in \mathcal{S}} (\bar{\mathcal{M}})^{\sigma}$.*

By Proposition 3.1, we can reduce the equation (3.1) to the following.

$$\text{NF}(g_i, (\mathcal{G}^{(p,\infty)})^{\sigma}) = 0 \text{ for every } \sigma \in G_{E_i} \setminus G_f, \quad (3.2)$$

where $G_f = \text{Stab}(m)$ and G_{E_i} denotes $\text{Stab}([\alpha_{e_{i,1}}, \dots, \alpha_{e_{i,t}}])$ for $E_i = \{e_{i,1}, \dots, e_{i,t}\}$. Because $\{(\mathcal{G}^{(p,\infty)})^{\sigma} : \sigma \in G_{E_i} \setminus G_f\}$ provides all elements of $Z_{\bar{\mathbb{Q}}_p}(\mathbb{Q}_p \otimes_{\mathbb{Q}} \mathcal{M})(E_i)$. Moreover, replacing $\mathcal{G}^{(p,\infty)}$ with $\mathcal{G}^{(p,k)}$ (see Remark 3), we have the following equation that approximation $g_i \bmod p^{k+1}$ must satisfy.

$$\text{NF}(g_i, (\mathcal{G}^{(p,k)})^{\sigma}) \equiv 0 \pmod{p^{k+1}} \forall \sigma \in G_{E_i} \setminus G_f. \quad (3.3)$$

REMARK 5. *The components of $\pi_p(m)$ are conjugate to each other by the action of $G_f = \text{Stab}(m)$. From this fact, it follows that any choice of $\bar{m}^{(p)}$ from the components of $\pi_p(m)$ still give the same Gröbner basis $\mathcal{G} = \{g_1, \dots, g_n\}$.*

In [18] the authors presented a linear system resolution to compute $g_i \bmod p^{k+1}$. Here, we will present in a further section how to do this by interpolation. Before that, we present how to compute an approximation of g_i modulo an integer M by *Chinese Remainder Theorem*.

3.2 Chinese Remainder Construction

Recall that m is the idempotent of $\mathcal{A}_0$ corresponding to the fixed splitting ideal $\mathcal{M}$ corresponding to the specific roots ordering. We now consider primes $p_1, \dots, p_i$ satisfying the property $\mathbb{P}$ and a component $\bar{m}^{(p_i)}$ of $\pi_{p_i}(m)$ for each p_i . Let $\mathcal{G}^{(p_1,k_1)}, \dots, \mathcal{G}^{(p_i,k_i)}$ the approximate Gröbner bases corresponding to these components.

As seen in Section 3.1, we can approximate the polynomials g_i modulo each $p_i^{k_i}$. Thus, by *Chinese Remainder Theorem* we lift them in the ring $\mathbb{Z}/M\mathbb{Z}$ where $M = \prod_{j=1}^i p_j^{k_j}$. From this computation we obtain the projection of $\mathcal{G}$ modulo M , that is the set of polynomials $\{g_1 \bmod M, \dots, g_n \bmod M\}$. But, in all this computation we assume that the idempotent m is fixed. In practice, we can not assume this hypothesis. Thus we need a general method to assure that each component $\bar{m}^{(p_j)}$ will correspond to the same idempotent m . (See Remark 5.)

To do that, we will use data produced during the computation of the Galois group G_f done modulo $p_1^{k_1}$. Then we reorder the roots modulo primes $p_2, \dots, p_i$ by following criteria obtained from the data. From the computation of G_f [21, 10], we obtain a finite sequence $\{(I_i, A_i) : i = 1, \dots, t\}$ of *invariants* and their *integer evaluation* modulo $p_1^{k_1}$ such that $\text{NF}(I_i - A_i, \mathcal{G}^{(p_1,k_1)}) \equiv 0 \pmod{p_1^{k_1+1}}$ for every i . Since $p_1^{k_1}$ exceeds the computed theoretical bound so that the corresponding relative Lagrange resolvent has A_i as its simple integral root. This implies $(I_i - A_i)m = 0$ for every i and $\text{NF}(g_i, \mathcal{G}^{(p_1,k_1)}) \equiv 0 \pmod{p_1^{k_1+1}}$ for every g_i . Conversely, tracing the determination process of G_f with different modulus q^k , if q^k exceeds the bound, we have the same result. As $p_1^{k_1}$ already exceeds the bound, we have

THEOREM 3.2. *Let q be a prime satisfying the property $\mathbb{P}$, $\mathcal{G}^{(q,k)}$ be the k -th approximation of a Gröbner basis of a maximal divisor of $\mathbb{Q}_q \otimes_{\mathbb{Q}} \mathcal{M}_0$ and $\bar{m}^{(q)}$ its corresponding primitive idempotent of $\bar{\mathcal{M}}_0^q$. If $q^k > p_1^{k_1}$ and $\text{NF}(I_i - A_i, \mathcal{G}^{(q,k)}) \equiv 0 \pmod{q^{k+1}}$ for every i , then $\text{NF}(g_i, \mathcal{G}^{(q,k)}) \equiv 0 \pmod{q^{k+1}}$ for every g_i , that is, $\bar{m}^{(q)}$ is a component of $\pi_q(m)$.*

This method can be seen as a modular Galois group computation guided by the knowledge of the exact branch taken during the descent from S_n to G_f in the permutations subgroups tree of degree n . Efficient implementations of Galois group computation use some techniques to cut this descending branch and allow to begin from a subgroup of S_n (see [10]). To be more efficient, we plan to adapt our method with these techniques.

3.3 Computation Scheme and i -relations

In this section, we recall the definition and give some new results about *computation scheme* and *i -relation* (see [18]).

In [18, Section 3] the authors present a framework for the computation of the Gröbner basis $\mathcal{G} = \{g_1, \dots, g_n\}$ with indeterminate coefficients strategy. In this framework, we attach to a particular permutation representation G_f a set of *good* theoretical form for polynomials of $\mathcal{G}$ and *techniques* which allow us to avoid computations for some g_i . This is what we call *computation scheme* since this guides the algorithm for computing $\mathcal{G}$.

In particular, we associate to each polynomial g_i an integers set $E_i = \{e_1 < \dots < e_s = i\}$ which describes a triangular set $T_i = \{g_1^*, \dots, g_s^* = g_i\}$ where $g_k^* \in \mathbb{Q}[x_{e_1}, \dots, x_{e_k}]$ and $g_k^*(x_{e_k}, \alpha_{e_{k-1}}, \dots, \alpha_{e_1})$ is a minimal polynomial of the $\mathbb{K}$ -extension $\mathbb{K}(\alpha_{e_k})$ where $\mathbb{K} = \mathbb{Q}(\alpha_{e_{k-1}}, \dots, \alpha_{e_1})$; we will denote by $d(E_i)_k$ the degree of this extension. In [18] the theoretical form g_s^* is used to compute g_i by indeterminate coefficients strategy. The number of coefficients to compute is deduced from E_i (or equivalently by T_i) and is denoted by $d(E_i)$:

$$d(E_i) := \prod_{k=1}^s \deg_{x_{e_k}} g_k^*,$$

and this quantity is called *the degree of E_i* .

There may be a lot of different sets E_i which all correspond to the polynomial g_i but, the smaller $d(E_i)$ is, the more efficient our algorithm will be. For example, we can choose the trivial set $E_i = \{1, 2, \dots, i-1, i\}$ which has maximal degree but in almost all cases we can find a better set E_i corresponding to g_i . This is why these sets E_i are important in our algorithm and we call them *i -relations*.

The *computation scheme* introduced in [18] provides also some techniques to avoid the computation of some g_i 's. Thus, to this framework we attach the set $\mathcal{I}$ of integers corresponding to the polynomials we have to compute. The total number of coefficients to compute in $\mathcal{G}$ is the sum of the degrees of i -relation with i in $\mathcal{I}$ and we denote it $c(G_f)$. To compute the polynomials with index in $\mathcal{I}$ modulo a power of a prime, the strategy used was based on *indeterminate coefficients* followed by a *linear algebra* step. Here we want to replace the second step by an *interpolation* step. This is what we present in the next section.

3.4 Lagrange Formulae and i -relations

In [6], Lagrange formulae are presented for general triangular sets. These formulae can be used to compute the Gröbner basis $\mathcal{G}$, this is what is done in [13]. In this case, the total number of coefficients to compute will be of the order of the size of the Galois group G_f which may be very large. Thus, to overcome this problem we introduce Lagrange formulae for i -relations in order to use the *computation scheme*.

Let $E_i = \{e_1 < \dots < e_s = i\}$ be an i -relation and T_i its associated triangular basis as defined in Section 3.3. The affine variety $Z_{\overline{\mathbb{Q}}}(T_i)$ is equiprojectable, thus one can apply

the Lagrange formulae given in [6] but, since here we are in a very special context, we will restate the construction by using the permutation representation of the given Galois group G_f .

Let σ be a given permutation in S_n . (Here we write $\sigma(a)$ for a^σ for simplifying formulas.) We denote by $\mathcal{O}(j, \sigma)$ the orbit of $\sigma(e_j)$ under the action of the point-wise stabilizer $\text{Stab}_{G_f}[\sigma(e_1), \sigma(e_2), \dots, \sigma(e_{j-1})]$ defined by $\{\tau \in G_f \mid \tau(\sigma(e_i)) = \sigma(e_i), \forall i \in \{1, \dots, j-1\}\}$. By using the map $i \mapsto \alpha_i$, the set $\mathcal{O}(j, \sigma)$ corresponds to the orbit of the element $\sigma(\alpha_{e_j})$ over the field $\mathbb{Q}(\alpha_{\sigma(e_1)}, \alpha_{\sigma(e_2)}, \dots, \alpha_{\sigma(e_{j-1})})$.

From this orbit we can interpret the formula given in [6] in our specific case.

THEOREM 3.3. *Let $E_i = \{e_1 < \dots < e_s = i\}$ be an i -relation. The corresponding polynomial g_i verify $g_i = \mathcal{L}_i$ with*

$$\mathcal{L}_i = \sum_{\sigma \in \text{Trans}} \left(\left(\prod_{j=1}^{s-1} \prod_{\substack{e \in \mathcal{O}(j, \sigma) \\ e \neq \sigma(e_j)}} \frac{x_{e_j} - \alpha_e}{\alpha_{\sigma(e_j)} - \alpha_e} \right) \prod_{e \in \mathcal{O}(s, \sigma)} x_i - \alpha_e \right)$$

where Trans is the transversal $\text{Stab}_{G_f}([e_1, \dots, e_s]) \backslash \backslash G_f$

3.5 Bound for the Coefficients of g_s^*

In the formula $\mathcal{L}_i$ given in Theorem 3.3 the denominators can be canceled by multiplying with a sufficiently large power of $d(f)$ the discriminant of the polynomial f . The multiplication by $d(f)$ can cancel two denominators of the form $\prod_{\substack{e \in \mathcal{O}(j, \sigma) \\ e \neq \sigma(e_j)}} \frac{1}{\alpha_{\sigma(e_j)} - \alpha_e}$. Thus, all the denominators can

be canceled by multiplying $\mathcal{L}_i$ with $D_i = d(f)^{\lceil \frac{s}{2} \rceil}$.

The polynomial $D_i \mathcal{L}_i$ has integral coefficients, we will now investigate a bound over the coefficient c corresponding to the multi-degree $(k_1, \dots, k_s)$. We denote by d_j the degree in x_j of $\mathcal{L}_i$. We note $d(E_i) = \prod_{k=1}^s d_k = |\text{Stab}_{G_f}([e_1, \dots, e_s])|$. Let δ be a bound over the differences of roots $|\alpha_i - \alpha_j|$ and ν a bound over the absolute values of the roots $|\alpha_i|$. Here we will modify the proof given in [13] to deal with the case of an i -relation:

1. After cancellation of the denominators by multiplying with D_i , it remains, in the numerator, a product of $n(n-1)\lceil \frac{s}{2} \rceil - d_1 - \dots - d_s + s$ elements of the form $(\alpha_j - \alpha_i)$. This product will be distributed on all the coefficients of g_i and is bounded by $\mathbb{B} = \delta^{n(n-1)\lceil \frac{s}{2} \rceil - d_1 - \dots - d_s + s}$.
2. The indeterminate x_{e_i} of degree k_i in $\mathcal{L}_i$ comes from a product of $d_i - 1$ elements of the form $(x_{e_i} - \alpha_j)$. Thus, its absolute value can be bounded by the well known binomial quantity $\binom{d_i-1}{k_i} \nu^{d_i-1-k_i}$.

Hence, by summing all these products over the transversal (see Theorem 3.3), we obtain the following bound for the absolute value of c :

$$d(E_i) \binom{d_1-1}{k_1} \nu^{d_1-1-k_1} \dots \binom{d_s}{k_s} \nu^{d_s-k_s} \mathbb{B}.$$

4. MULTI-MODULAR STRATEGY

In order to attain efficient computation of splitting fields, we can make good use of more sophisticated modular computation technique, “multi-modular” one. Here we show details on our technique and its variants for improvements.

4.1 Basic Discussion on Modular Techniques

There are several strategies on applying modular techniques for splitting field computation. Among those, a *multi-modular strategy* described below shall be effective and efficient under the following assumption which seems natural phenomena for our problem.

Assumption: The computed theoretical bound, say B_T , on the coefficients of the Gröbner basis $\mathcal{G}$ is much larger than the *real bound* B_R , that is, the maximal absolute value of numerators and denominators of coefficients of $\mathcal{G}$. Also B_T is much larger than the bound B_G used for the Galois group determination.

Under the assumption, it is quite natural to use some heuristic bound B_H much smaller than B_T . Our computation can be one instance of the following model:

COMPUTATIONAL MODEL WITH MODULAR COMPUTATION
Here, the target which we want to compute is some mathematical object over the rational number field $\mathbb{Q}$.

Step 1. Candidate Computation: **Proc_CAND**

Step 1-1. Modular Image Computation

We set the modulus q , and then compute the modular image of the target modulo q .

Step 1-2. Conversion

By rational reconstruction, we have a candidate of the target.

Step 2. Correctness Check: **Proc_CHECK**

We check whether a candidate is correct or not by some efficiently computable test. If the test is OK, we have the correct result.

In our case, **Proc_CHECK** can be executed by *ideal inclusion test*, which shall correspond to *trial division* for polynomial factorization, as pointed out in [18]. Also, we may use another modular technique for **Proc_CHECK**. Further discussion will be given later.

When we use $2B_T^2$ for the modulus, the computed candidate is always correct and **Proc_CHECK** is not necessary. (We note that for rational reconstruction, the modulus should be twice of square of the bound.) On the other hand, when we use some heuristic modulus q , we have to execute recursive computation to reach the correct answer. When **Proc_CHECK** fails, we can apply several strategies in Step 1:

S1: Replace a larger modulus q' and execute Step 1-1.

S2: Take another modulus q' prime to q , execute Step 1-1 with q' and combine the result with the old one by Chinese remainder theorem. Then the modulus in **Proc_CHECK** becomes $q \times q'$.

S3: Lift up the candidate to a larger modulus q' by Hensel construction.

Here, we call the strategy S2 *multi-modular strategy*, and the strategy S3 *p-adic strategy*. Apparently, the strategy S1 is not efficient compared with other two.

To find the most practical one among strategies in the above, we examine those total times of computation. Let $T_P(q)$ be the time of Step 1-1 of **Proc_CAND**, where q is the modulus used, $T_R(q)$ the time of Step 1-2 and T_C the time of **Proc_CHECK**. Here, we dare to omit the effect of $c(G_f)$ on those in order to make our argument clear. In Section 4.3, we will give further discussion taking $c(G_f)$ into account.

If we use $2B_T^2$ for the modulus, **Proc_CHECK** is not necessary, and thus the total time $T_0 = T_P(2B_T^2) + T_R(2B_T^2)$. If

we use some heuristic modulus q , then we should repeat the computation till the modulus exceeds $2B_R^2$. Suppose that we reach to it by s times recursion, where $q_1, \dots, q_s$ are moduli used. Then we have the following total times:

S2: $T_2 = \sum_{i=1}^s (T_P(q_i) + T_{CRT}(q_1 \cdots q_{i-1}, q_i) + T_R(\prod_{j=1}^i q_j) + T_C)$, where $q_1 \cdots q_{s-1} < 2B_R^2 < q_1 \cdots q_s$ and $T_{CRT}(q, q')$ denotes the time for Chinese remainder theorem for two moduli q to q' .

S3: $T_3 = T_P(q_1) + \sum_{i=2}^s (T_H(q_i, q_{i-1}) + T_R(q_i) + T_C)$, where $q_{s-1} < 2B_R^2 < q_s$ and $T_H(q', q)$ denotes the time for Hensel lifting from the modulus q to q' .

Getting precise estimation of T_C is very difficult, when we apply *ideal inclusion test*. Thus, we may also apply additional modular technique to have efficient realization of **Proc_CHECK**. The basic procedure is the following:

MODULAR CHECK: **Proc_MODCHECK**

Once we have a candidate $\mathcal{C}$ constructed by using modulo q , we check if it is still valid modulo another q' . If so, we can show that $\mathcal{C}$ is still a candidate modulo $q \times q'$. Otherwise, we compute the modular image of a candidate modulo q' by Step 1-1 of **Proc_CAND** and apply Chinese remainder theorem to get the modular image of new candidate $\mathcal{C}'$ modulo $q \times q'$.

Now we denote by $T_{MC}(q)$ the time for modular check modulo q , not including any candidate construction. In our case, we suppose that $T_{MC}(q)$ is much smaller than $T_P(q)$ and $T_H(q', q'')$ with $q = q'/q''$. (See Section 4.3 for details.) Then, we have two types of usage of MODULAR CHECK:

U1: We can reduce the number of **Proc_CHECK** by repeating **Proc_MODCHECK** until we have a *stable* result. Then, it is highly supposed that the computed candidate is correct. With this practical assumption, the size of the total modulus is supposed the same order as that of $2B_R^2$, and so that of B_R . Then we have $T_2 = O(\sum_{i=1}^s (T_P(q_i) + T_{CRT}(q_1 \cdots q_{i-1}, q_i) + T_R(\prod_{j=1}^i q_j) + T_{MC}(q_i)) + T_C)$, where $\prod_{i=1}^s q_i = O(B_R)$, and $T_3 = O(T_P(q_1) + \sum_{i=2}^s (T_H(q_i, q_{i-1}) + T_R(q_i) + T_{MC}(q_i)) + T_C)$, where $q_s = O(B_R)$.

U2: We repeat **Proc_MODCHECK** till the total modulus reaches the theoretical bound. (So, the p-adic strategy is not suited for this approach.) This may sound somehow contradictory to our strategy. But, it is still able to give a practical solution as $T_{MC}(q)$ is much smaller than $T_P(q)$. In this case, we have $T_2 = O(\sum_{i=1}^s (T_P(q_i) + T_{CRT}(q_1 \cdots q_{i-1}, q_i) + T_R(\prod_{j=1}^i q_j)) + \sum_{i=1}^t T_{MC}(q_i))$, where $\prod_{i=1}^s q_i = O(B_R)$ and $\prod_{i=1}^t q_i = O(B_T)$.

Omitting the correctness check in U1, the total efficiency can be much improved. In this case, the result is not proven to be correct, but it will be with a high probability.

We will show certain practicality and theoretical efficiency of the multi-modular strategy, as it can use the both usages of *modular check*. In Section 4.2, we will give details of algorithms based on multi-modular strategy, and in Section 4.3, based on algorithms given in Section 4.2, we will discuss those efficiency, including estimation on T_C .

4.2 Algorithms

In the case of the computation of the splitting field of the polynomial f , the target will be the Gröbner basis $\mathcal{G}$ of the splitting ideal corresponding to the symmetric representation G of the Galois group of f . All our first inputs came from the computation of G_f by modular algorithm

(see [21],[9, 10]) modulo $p_1^{k_1}$ a power of a prime satisfying property $\mathbb{P}$. From this computation we obtain a finite sequence $\{(I_i, A_i) : i = 1, \dots, t\}$ of *invariants* and their *integer evaluation* modulo $p_1^{k_1}$ as in Theorem 3.2. Because, we will use *multi-modular strategy* S2 for computing $\mathcal{G}$ we may need to reorder the roots of f modulo $p_2^{k_2}$, where p_2 is a prime satisfying $\mathbb{P}$ different from p_1 . To do this we use the sequence $\{(I_i, A_i)\}$ as in Theorem 3.2, this is described in the following function:

Function: GoodOrdering $((p_2, k_2))$

We assume $p_2^{k_2} \geq p_1^{k_1}$ (else we exit with an error).
Let $\hat{\alpha}$ be the roots of f modulo $p_2^{k_2}$.
Let t be the length of the sequence $\{(I_i, A_i)\}$.
for $i = 1$ to t **do**
 Let (G, H) be the groups corresponding to I_i .
 for $\sigma \in G \setminus H$ **do**
 if $I_i(\hat{\alpha}^\sigma) = A_i$ **then**
 $\hat{\alpha} := \hat{\alpha}^\sigma$
 break
 end if
 end for
end for
return $\hat{\alpha}$

As presented in Section 4.1 we now give an algorithm to compute a candidate $\mathcal{G}_{\text{cand}} = \{\tilde{g}_1, \dots, \tilde{g}_n\}$ for the target $\mathcal{G}$ by using a heuristic bound B_T . This first step is described by the algorithm **Proc_CAND** that takes as input an approximation of the roots of $f \bmod p_1^{k_1}$ as a Gröbner basis $\mathcal{G}^{(p_1, k_1)}$ of the splitting ideal of $f \bmod p_1^{k_1}$, the Galois group G_f corresponding to the order of these roots and a set $\mathcal{P}$ of couples (p, k) where p 's are different primes (different from p_1) satisfying property $\mathbb{P}$ and k 's are integers such that $p_1^{k_1} \prod_{(p,k) \in \mathcal{P}} p^k \geq B_H$. To the group G_f is associated a *computation scheme* which is represented by the set $\mathcal{I}$ of indexes of the g_i to compute in $\mathcal{G}$ and the corresponding i -relation E_i . Before giving this first algorithm, we give a specific *Chinese Remaining Theorem* procedure which will be used in the sequel:

Procedure: SpecCRT $(\{g_i \bmod M, i \in \mathcal{I}\}, (p, k))$

Compute and reorder the roots of $f \bmod p^k$.
Interpolate the g_i 's mod p^k corresponding to E_i with $i \in \mathcal{I}$.
Let M be the modulus $M \times p^k$.
By CRT compute all the $g_i \bmod M$ ($i \in \mathcal{I}$).

Now, we can describe the first algorithm corresponding to the first step **Proc_CAND**:

Algorithm 1: Proc_CAND $(\mathcal{G}^{(p_1, k_1)}, G_f, \mathcal{P})$

Interpolate all the $g_i \bmod p_1^{k_1}$ corresponding to E_i ($i \in \mathcal{I}$).
Let M be the modulus $p_1^{k_1}$.
for $(p, k) \in \mathcal{P}$ **do**
 SpecCRT $(\{g_i \bmod M, i \in \mathcal{I}\}, (p, k))$.
 Let M be the modulus $M \times p^k$.
end for
S1:
try to convert all g_i 's mod M to rational polynomials h_i 's.
if all the conversions above succeed **then**
 The polynomial $\tilde{g}_i$ is h_i .
else
 Find a new couple (p, k) not in $\mathcal{P}$.
 SpecCRT $(\{g_i \bmod M, i \in \mathcal{I}\}, (p, k))$.
 Let M be the modulus $M \times p^k$.
 Goto step **S1** and add (p, k) in $\mathcal{P}$.
end if
For each $\tilde{g}_j$ with $j \notin \mathcal{I}$, apply a technique over one $\tilde{g}_i$ with $j < i$ to obtain $\tilde{g}_j$.

Return $\mathcal{G}_{\text{cand}}, G_f, \mathcal{I}, \mathcal{P}$.

Since we concentrate here on multi-modular strategy for the computation of $\mathcal{G}$, we give an algorithm corresponding to **Proc_CHECK** based on the strategy S2. Recall that this algorithm is based on the *ideal inclusion test* given in [18]:

Algorithm 2: Proc_CHECK_S2 $(\{\tilde{g}_1, \dots, \tilde{g}_n\}, G_f, \mathcal{I}, \mathcal{P})$

S1:
if The equality $\text{NF}(\tilde{g}_i, c_i(f)) = 0$ is true for all $i \in \mathcal{I}$ **then**
 The Gröbner basis $\mathcal{G}$ is $\{\tilde{g}_1, \dots, \tilde{g}_n\}$.
else
 Let M be the product $\prod_{(p,k) \in \mathcal{P}} p^k$.
 S2: Find a new couple (p, k) not in $\mathcal{P}$.
 SpecCRT $(\{g_i \bmod M, i \in \mathcal{I}\}, (p, k))$.
 Apply techniques to obtain the $\tilde{g}_j$'s with $j \notin \mathcal{I}$.
 Let M be the modulus $M \times p^k$.
 Convert all the $\tilde{g}_i$'s mod M to rational polynomials.
 if conversions above pass **then** goto **S1** **else** goto **S2**.
end if
Return $\mathcal{G}$.

As we said in Section 4.1, we can take advantage of the multi-modular strategy in **Proc_CHECK** too. Here we give the implementation of **Proc_MODCHECK** with a generic stopping condition which can be stated in function of the chosen U1 or U2 version.

Algorithm 3: Proc_MODCHECK $(\{\tilde{g}_1, \dots, \tilde{g}_n\}, G_f, \mathcal{I}, \mathcal{P})$

Let M be the product $\prod_{(p,k) \in \mathcal{P}} p^k$.
S1:
if The specified condition **COND** is not satisfied **then**
 Find a new couple (p, k) not in $\mathcal{P}$.
 Compute and reorder the roots $\hat{\alpha}$ of $f \bmod p^k$.
 if there exists $i \in \mathcal{I}$ such that $\tilde{g}_i(\hat{\alpha}) \neq 0 \bmod p^k$ **then**
 SpecCRT $(\{g_i \bmod M, i \in \mathcal{I}\}, (p, k))$.
 Apply techniques to obtain the $\tilde{g}_j$'s with $j \notin \mathcal{I}$.
 end if
 Let M be the modulus $M \times p^k$.
 Goto **S1**.
end if
 Convert all the $\tilde{g}_i$'s mod M to rational polynomials g_i 's.
S2: Return $\{g_1, \dots, g_n\}$.

If condition **COND** is chosen to be the one corresponding to the version U1 then the output of **Proc_MODCHECK** is not proven but we can use it as an input of **Proc_CHECK**. Otherwise, the condition **COND** will correspond to version U2 and in this case the output will be proven.

4.3 Discussion on Efficiency

Here we discuss the efficiency of the proposed “multi-modular strategy” along by basic estimation on concrete procedures given in the previous subsection. We use the facts given in [18]. By actual experiments on real computers, the authors found that sometimes the correctness check dominated the total efficiency, even though the modulus used there was the same order of the real bound. To resolve this problem, we introduce the usage U2 and show that it can give good estimation on the total efficiency.

To estimate the total efficiency, we have to give concrete representation of the functions $T_P, T_R, T_{CRT}, T_H, T_{MC}$ and T_C . To distinguish the time for Step 1-1 of **Proc_CAND** by linear system solving [18] and that by Lagrange interpolation in Section 3.4, we denote by T_{PL} the time by linear system solving and by T_{PI} that by interpolation. Also, we denote by $\mathbb{M}(q)$ the unit cost of integer arithmetics of size q . Then, $\mathbb{M}(q) = O(q^2)$ for usual multiplication technique, and $\mathbb{M}(q) = O(q^{1+\epsilon})$ for fast multiplication technique. Also by

[18], we have $T_C = O(\log(n)c(G_f)^2\mathbb{M}(\log(B')))$ and B' is the largest integer appeared in the normal form computation. (Finding a good estimation on B' is a difficult problem.)

For the strategy S2, it had better to use Lagrange interpolation for `Proc_CAND` and $T_{PI}(q) = O(c(G_f)^2\mathbb{M}(\log(q)) + T_{GO})$, where T_{GO} is the time for `GoodOrdering`. For the two quantities $T_{CRT}(A, B)$ and T_R , we can apply fast extended GCD computation technique for Chinese remainder Theorem and rational reconstruction (see [15]), and thus

$T_{CRT}(A, B) = O(c(G_f)\mathbb{M}(\log(C))\log\log(C))$, where $C = \max\{A, B\}$ and $T_R(q) = O(c(G_f)\mathbb{M}(\log(q))\log\log(q))$.

As to the modular check of correctness, we divide each candidate $\tilde{g}_i$ by $q = p^k$ and substitute the p-adic approximation of roots of f , and so $T_{MC}(q) = O(c(G_f)(\mathbb{M}(\log(q)) + \mathbb{M}(\log(D))))$, where D is the maximal absolute value of numerators and denominators of coefficients of $\tilde{g}_i$'s. As $D < B_R$, we have $T_{MC}(q) = O(c(G_f)(\mathbb{M}(\log(B_R))))$.

For the strategy S3, by the estimation in [18], we have $T_{PL}(q) = O((c(G_f)^\omega\mathbb{M}(\log(q))),$ and $\sum_{i=2}^s T_H(q_i, q_{i-1}) = O(n^2c(G_f)^2\mathbb{M}(\log(q_s)))$ where ω represents a feasible matrix multiplication exponent and $2 \leq \omega \leq 3$ (see [8]).

Thus, accumulating all representations, the total cost can be estimated in *slightly rough but simple form*, where we are assuming that $\log(B_T)$ is much larger than $\log(B_R)$ and the size of modulus q used is close to the size of the real bound B_R , that is, the number of correctness checks in U1 and that of CRT construction in U2 can be *bounded* (one or very small).

PROPOSITION 4.1. *Assume that $\log(B_T)$ is much larger than $\log(B_R)$ and the size of modulus q used and that of B_R are the same order.*

(1) *When we use the modular check U1, we have*

$$T_2 = O(c(G_f)^2\mathbb{M}(\log(B_R)) + c(G_f)\mathbb{M}(\log(B_R))\log\log(B_R) + T_{GO} + T_C),$$

$$T_3 = O(c(G_f)^\omega\mathbb{M}(\log(q_1)) + n^2c(G_f)^2\mathbb{M}(\log(B_R)) + T_C),$$

where $T_C = O(\log(n)c(G_f)^2\mathbb{M}(\log(B')))$ and B' is the largest integer appeared in the normal form computation.

(2) *When we use the modular check U2, we have*

$$T_2 = O(c(G_f)^2\mathbb{M}(\log(B_R)) + c(G_f)\mathbb{M}(\log(B_R))\log\log(B_R) + c(G_f)\mathbb{M}(\log(B_T)) + T_{GO}\frac{\log(B_T)}{\log(B_R)}).$$

Taking account of the estimation on B_T in Section 3.5, when `GoodOrdering` can be efficiently done like as T_{GO} is $O(c(G_f)\mathbb{M}(\log(B_R)))$, the strategy S2 with the modular check U2 can give the most efficient computation.

5. EXPERIMENTS AND REMARKS

We have implemented the algorithms of Section 4.2 with the *computer algebra system* MAGMA (version 2.13) in the case of an irreducible monic integral polynomial. This version of MAGMA provides a lot of functions for using the byproduct of the modular computation of the Galois group. All these functions are very efficient and easy to use but we need a little more, thus we have rewritten a large part of our sub-functions in order to be more efficient.

Reordering the roots: In the case of the normalizer of G_f in S_n is reduced to G_f , it is sometimes better to recompute the Galois group G_2 of f using another prime p_2 and we

finally reorder the roots by applying the permutation which conjugates G_f and G_2 , as we can use the efficient computation of the Galois group. This is specially effective when the descending process in the subgroups tree encounter two succeeding groups with large index. This method could be generalized by considering the normalizer of G_f and will study in a future implementation.

Choice of the primes p : Because of the limit imposed by the Tchebotarev's density theorem it may be hard to find primes which split completely in the stem field defined by the polynomial f . On the other hand, the costs of the p-adic arithmetic increase according to the order of the Galois group of f modulo p . Thus, we choose primes satisfying the property $\mathbb{P}$ such that the Galois group of f modulo theses primes has an order of at most 2.

Heuristic bound B_H : We choose, as an heuristic, to begin the computation `Proc_CAND` modulo the fifth power of the product of one, two or three minimal primes satisfying the condition state before.

Comments on the experiments: We try our implementation on several polynomials given by [11] from degree 6 to 9 and some polynomials of greater degree (not more than 13) corresponding to interesting computation schemes. By using this heuristic bound, `Proc_CAND` already computes in all the cases the final result, thus the remaining computation is the check procedure without any reconstruction. For groups G with small order or $c(G)$ (say under 500) our new implantation will not give a really better efficiency than the one presented in [18] which was already very efficient. The only gain is given by the use of the interpolation in place of the linear system resolution. Moreover, if the group is very small in the symmetric group of same degree, the reordering procedure may be time consuming. Thus, in this case it is preferable to use mono-modular algorithm and check the result by normal forms computations (`Proc_CHECK_S2`). In the case of groups G with high order and small $c(G)$, our experiments show that by using interpolation, the gain of efficiency of this new implementation is comparable to the gain of theoretical complexity. In these cases, the experimental cost of the reordering of the roots and the reconstruction of the polynomials by Chinese remainder theorem are very small in comparison to the other parts of the computation. Concerning the check procedure, in these cases, the modular strategy with version U2 is always better than the procedure `Proc_CHECK_S2`. This experiments allow us to think that this new method should be use for the computation of splitting fields with high absolute degree, the remaining bottleneck is the efficient calculation of a computation scheme (we have tabulated them up to degree 10).

Remarks on parallel computations: The computer algebra system MAGMA do not integrate any possibility for parallel computation, thus we have implemented all our algorithms with a sequential style. But, during our experiments we reported that all the parts of the computation that can be shared on different cores have the same time consuming in general. Thus, implementing these algorithms in a parallel system will have a real impact on the efficiency. For example, in the case of `Proc_CAND` the computations modulo two different primes has almost the same time consuming in general (this is the case when the word sizes of these primes are the same and they have the same decomposition property modulo f).

We give here some specific examples and comments about them. We give, for each example, the name of the group G in Butler and McKay's nomenclature, the order of G and the integer $c(G)$ (as the sum of the i -relations degrees). The column **Primes** shows the timings of computing primes with good properties. The column **Lin.** shows timing **Proc_CAND** using linear systems solving as in [18], **Int.** those when we use interpolation, **S2** the timings for check procedure **Proc_CHECK_S2**, **U2** the timing of **Proc_MODCHECK** based on version **U2** and **Ord.** the timing of reordering the roots. The total timings can be obtained by summing the column corresponding to strategies used during the computation. The measurements were made on a personal computer with a 3.0GHz Intel Xeon 64bits (all the timings are given in seconds).

Group	$ G $	$c(G)$	Primes	Lin.	Int.	S2	U2	Ord.
10T39	3840	10	2.13	0.04	0.05	0.01	0.01	1.01
10T36	1920	960 + 10	0.81	141.2	60.5	55.4	2.1	1.3
9T32	1512	2×1512	0.57	540.8	65 + 67	412.1	4.1	0.5
9T29	648	18 + 648	0.67	40.1	9.3	1.22	1.5	1.0
9T25	324	27	0.08	6.54	7.1	0.2	0.5	2.4
8T48	1344	336	1.25	9.89	4.23	15.34	1.1	0.8
7T6	2520	2520	1.5	785.3	76 + 158	8.2	2.1	2.5

The first line shows a very special case where the total timing is dominated by the research of primes with good property. These cases appear when the group is very large in comparison with its size $c(G)$ as here. On the last line, we show the two different timings for computing modulo two different primes. In this case, the first one corresponds to a prime which splits completely the polynomial f in contrary to the second one. This is a general fact, for a fixed word size, using a splitting prime will give better timings (in this case the first one is 10037 the second is 53). The remaining examples show that even the times between the different strategies are comparable, using multi-modular and interpolation approaches are in general better. We did not compare these timings with the ones given in [13] because of the difference of architecture, but if we project them on the same computer our implementation would be more efficient with a factor from 10 to 1000 (this big difference may come from the fact that *computation schemes* are not used in [13]). Also, we did not integrate in the last table the timings for computing $\mathcal{G}$ in MAGMA (version 2.14) with the new function **GaloisSplittingIdeal** (with parameter **Roots** set to false) since we can only obtain two of them by waiting not more than 300 seconds.

6. CONCLUSION

We have presented a new method based on a multi-modular strategy for the computation of the splitting field of a polynomial f . This new method is a good compromise for the one presented in [18] since it gives better results in the case where the later was inefficient. Also, the experiments show that this multi-modular method is a good candidate for a parallel implementation on a multi-core architecture.

The reordering roots function we give here is a general one. We plan to integrate efficient techniques of Galois groups computation (like the ones presented in [10]) in order to reduce the length of the descending branch in the subgroups tree.

We hope that such a multi-modular method could be generalized to the problem of computing algebraic or integral relations between the roots of a polynomial like in [7].

7. ACKNOWLEDGMENT

The authors would like to thank the referees for their valuable remarks.

8. REFERENCES

- [1] I. ABDELJAOUD, S. ORANGE, G. RENAULT, AND A. VALIBOUZE. Computation of the decomposition group of a triangular ideal. *AAECC Journ.*, 15(3-4):279–294, 2004.
- [2] J.-M. ARNAUDIÈS AND A. VALIBOUZE. Lagrange resolvents. *J. Pure Appl. Algebra*, 117/118:23–40, 1997. Algorithms for algebra (Eindhoven, 1996).
- [3] ANAI, H., NORO, M., AND YOKOYAMA, K. Computation of the splitting fields and the Galois groups of polynomials. In *Algorithms in algebraic geometry and applications*, vol. 143 of *Progr. Math.* Birkhäuser, Basel, 1996, 29–50.
- [4] BECKER, T., AND WEISPFENNING, V. *Gröbner bases*, vol. 141 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1993.
- [5] BOSMA, W., CANNON, J., AND PLAYOUST, C. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24, 3-4 (1997), 235–265.
- [6] DAHAN, X., AND SCHOST, É. Sharp estimates for triangular sets. In *ISSAC '04: Proc. of the 2004 International Symposium on Symbolic and Algebraic Computation* (New York, 2004), ACM Press, pp. 103–110.
- [7] FIEKER, C. AND GRAAF, W. Integral linear dependencies of algebraic numbers and algebraic Lie algebras. *LMS J. Comput. Math.*, 10 (2007), 271–287.
- [8] J. VON ZUR GATHEN AND J. GERHARD. *Modern computer algebra*. Cambridge University Press, Cambridge, second edition, 2003.
- [9] GEISLER, K. *Berechnung von Galoisgruppen über Zahl- und Funktionenkörpern*. PhD thesis, Univ. Berlin, 2003.
- [10] GEISLER, K. AND KLÜNERS, J. Galois group computation for rational polynomials. *J. Symbolic Comput.*, 30(6):653–674, 2000.
- [11] KLÜNERS, J., AND MALLE, G. A database for field extensions of the rationals. *LMS J. Comput. Math.* 4 (2001), 182–196 (electronic).
- [12] LAZARD, D. Solving zero-dimensional algebraic systems. *J. Symbolic Comput.* 13, 2 (1992), 117–131.
- [13] LEDERER, M., Explicit constructions in splitting fields of polynomials. *Riv. Mat. Univ. Parma* (7), 3* (2004), 233–244.
- [14] MCKAY, J., AND STAUDUHAR, R. Finding relations among the roots of an irreducible polynomial. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation* (New York, 1997), ACM, pp. 75–77.
- [15] PAN, V. AND WANG, X. Acceleration of Euclidean Algorithm and Rational Number Reconstruction. *SIAM J. Comput.* Vol.32, No.2, pp.548–556, 2003.
- [16] M. POHST AND H. ZASSENHAUS. *Algorithmic Algebraic Number Theory*. Cambridge Univ. Press, Cambridge, 1989.
- [17] RENAULT, G. Computation of the splitting field of a dihedral polynomial. In *Proc. of the 2006 International Symposium on Symbolic and Algebraic Computation* (New York, 2006). ACM Press, pp. 290–297.
- [18] RENAULT, G., AND YOKOYAMA, K. A modular method for computing the splitting field of a polynomial. In *Proc. of the 7th Algorithmic Number Theory Symposium* ANTS-VII, Berlin, Germany, 2006, LNCS 4076, Springer, pp. 124–140.
- [19] N. RENNERT AND A. VALIBOUZE. Calcul de résolvantes avec les modules de Cauchy. *Exp. Math.*, 8(4):351–366, 1999.
- [20] TCHEBOTAREV, N. *Gründzüge des Galois'schen Theorie*. P. Noordhoff, 1950.
- [21] YOKOYAMA, K. A modular method for computing the Galois groups of polynomials. *J. Pure Appl. Algebra* 117/118 (1997), 617–636.

Linearizability of 1:-3 Resonant System with Homogeneous Cubic Nonlinearities

Zhibek Kadyrsizova
al-Farabi Kazakh National University
Faculty of Mechanics and Mathematics,
Masanchi 39/47, Almaty 050012, Kazakhstan
kzhibek@mail.ru

Valery G. Romanovski
CAMTP – Center for Applied Mathematics and
Theoretical Physics, University of Maribor
Krekova 2, SI-2000 Maribor, Slovenia
valery.romanovsky@uni-mb.si

ABSTRACT

We study the systems of differential equations of the form $\dot{x} = x + p(x, y)$, $\dot{y} = -3y + q(x, y)$, where p and q are homogeneous polynomials of degree three (either of which may be zero). The necessary and sufficient coefficient conditions for linearization of such systems are obtained.

Categories and Subject Descriptors

G.m [Mathematics of Computing]: Miscellaneous; I.1 [Symbolic and Algebraic Manipulation]: Miscellaneous

General Terms

Theory, Algorithms

Keywords

ordinary differential equations, the center and linearizability problems, normal forms, polynomial ideals

1. INTRODUCTION

We consider a polynomial system of differential equations of the form

$$\begin{aligned} \frac{dx}{dt} &= px - \sum_{k+j=2}^n a_{kj} x^{k+1} y^j = P(x, y), \\ \frac{dy}{dt} &= -(qy - \sum_{k+j=2}^n b_{jk} x^j y^{k+1}) = Q(x, y), \end{aligned} \quad (1)$$

where x, y, a_{kj}, b_{jk} are complex variables.

In this paper we study the so-called problem of linearizability, for system (1), that is, the problem how to decide if a polynomial system of the form (1) can be transformed to the linear system

$$\dot{z}_1 = pz_1, \quad \dot{z}_2 = -qz_2 \quad (2)$$

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

by means of a change of the phase variables

$$\begin{aligned} z_1 &= x + \sum_{m+j=2}^{\infty} u_{m-1,j}^{(1)}(a, b) x^m y^j, \\ z_2 &= y + \sum_{m+j=2}^{\infty} u_{m,j-1}^{(2)}(a, b) x^m y^j, \end{aligned} \quad (3)$$

where (a, b) denotes $2n+2$ -tuple of the coefficients of system (1), that is, $(a, b) = (a_{10}, \dots, a_{-1,n}, b_{n,-1}, \dots, b_{01})$. If such transformation exists we say that the system is *linearizable*, it also means that the normal form of the system is its linear part.

Taking derivatives with respect to t in both parts of each of the equalities in (3) and then equating coefficients of the terms $x^{q_1+1}y^{q_2}$, $x^{q_1}y^{q_2+1}$ on each side of the equalities we obtain the recurrence formulae

$$(pq_1 - qq_2)u_{q_1q_2}^{(1)} = \sum_{s_1+s_2=0}^{q_1+q_2-1} [(s_1+q)u_{s_1s_2}^{(1)}a_{q_1-s_1,q_2-s_2} - s_2u_{s_1s_2}^{(1)}b_{q_1-s_1,q_2-s_2}], \quad (4)$$

$$(pq_1 - qq_2)u_{q_1q_2}^{(2)} = \sum_{s_1+s_2=0}^{q_1+q_2-1} [s_1u_{s_1s_2}^{(2)}a_{q_1-s_1,q_2-s_2} - (s_2+p)u_{s_1s_2}^{(2)}b_{q_1-s_1,q_2-s_2}], \quad (5)$$

where we initialize the calculations by setting $u_{00}^{(1)} = u_{00}^{(2)} = 1$. Thus, the coefficients $u_{q_1q_2}^{(1)}, u_{q_1q_2}^{(2)}$ of the transformation (3) can be computed step by step using the formulae (4), (5). In the case $pq_1 = qq_2$ the coefficients $u_{q_1q_2}^{(1)}, u_{q_1q_2}^{(2)}$ can be chosen arbitrarily, we set them equal to zero. Clearly the system is linearizable if and only if the quantities in the right-hand side of (4) and (5) are equal to zero for all (q_1, q_2) such that $pq_1 = qq_2$ or, equivalently, $q_1 = kq, q_2 = kp$ ($k = 1, 2, \dots$). The converse is known to also be true (the choices of the coefficients $u_{q_1,q_2}^{(1)}$ and $u_{q_1,q_2}^{(2)}$ do not matter). As a matter of definition, in such case we denote the polynomials in the right-hand side of (4) by $i_{qk,pk}$ and in the right-hand side of (5) by $-j_{qk,pq}$ and call them *q-th linearizability quantities*. We see that the system (1) with the given coefficients (a^*, b^*) is linearizable if and only if $i_{qk,pk}(a^*, b^*) = j_{qk,pk}(a^*, b^*) = 0$ for all $k \in \mathbb{N}$.

Thus, the linearizability problem for (1) is reduced to calculating the variety $\mathbf{V}(I)$ of the ideal $I = \langle i_{qk,pk}, j_{qk,pq} : k \in \mathbb{N} \rangle$ generated by the infinite set of polynomials $i_{qk,pk}, j_{qk,pq}$ ($k \in \mathbb{N}$); the variety $\mathbf{V}(I)$ is called the *linearizability variety*

of system (1). By the Hilbert Basis Theorem there exists an integer $N \geq 1$ such that $I = \langle i_{q,p}, j_{q,p}, \dots, i_{qN,pN}, j_{qN,pN} \rangle$, but no regular method to find such N is known. So we take the following strategy: find an integer $N_0 \geq 1$ such that

$$\mathbf{V}(\langle i_1, j_1, \dots, i_{N_0}, j_{N_0} \rangle) = \mathbf{V}(\langle i_1, j_1, \dots, i_{N_0+1}, j_{N_0+1} \rangle),$$

then compute the irreducible decomposition of the variety $\mathbf{V}(\langle i_1, j_1, \dots, i_{N_0}, j_{N_0} \rangle)$, and finally, using appropriate methods, show that all systems from each component of the decomposition are indeed linearizable.

In recent years many studies has been devoted to the investigation of the linearizability problem for different sub-families of the cubic system (that is, the system (1) where the right hand sides are polynomials of degree three), see e.g. [2, 3, 7, 10] and references therein. In this paper we obtain necessary and sufficient conditions for linearizability of the eight-parameter cubic system

$$\begin{aligned} \dot{x} &= x - a_{20}x^3 - a_{11}x^2y - a_{02}xy^2 - a_{-13}y^3, \\ \dot{y} &= -3y + b_{3,-1}x^3 + b_{20}x^2y + b_{11}xy^2 + b_{02}y^3. \end{aligned} \quad (6)$$

2. PRELIMINARIES

In this section we briefly mention some results which will be applied to prove linearizability of system (1). For more details the reader can consult, for instance, [2, 3].

A polynomial $f(x, y)$ defines an invariant curve $f(x, y) = 0$ if it satisfies the equation

$$\frac{\partial f}{\partial x}P + \frac{\partial f}{\partial y}Q = Kf, \quad (7)$$

with K being a polynomial. The polynomials $f(x, y)$ and $K(x, y)$ satisfying (7) are called the algebraic invariant curve and the cofactor, respectively. By definition a Darboux linearization of the system (1) is a change of variables

$$z_1 = H_1(x, y), \quad z_2 = H_2(x, y) \quad (8)$$

which transforms the system to the linear system (2), and such that at least one of the functions H_1, H_2 is of the form $H = f_0^{\alpha_0} f_1^{\alpha_1} \dots f_k^{\alpha_k}$, α_j 's being complex numbers, where the $f_i(x, y)$ are invariant algebraic curves of the system (1) defined by $f_i(x, y) = 0$. A simple computation shows that if there are algebraic invariant curves $f_1, f_2, \dots, f_k$ with the cofactors $K_1, K_2, \dots, K_k$ satisfying

$$\sum_{i=1}^k \alpha_i K_i = 0, \quad (9)$$

then $H = f_1^{\alpha_1} \dots f_k^{\alpha_k}$, is a first integral of the system (1).

A similar idea leads to the Darboux linearization. Namely, if there are $\alpha_0, \alpha_1, \dots, \alpha_k$ such that

$$K_0 + \sum_{i=1}^k \alpha_i K_i = p, \quad (10)$$

$f_0 = x + h.o.t$ and $f_i(0,0) \neq 0$ for $1 \leq i \leq k$ then by means of the substitution $z_1 = f_0 f_1^{\alpha_1} \dots f_k^{\alpha_k}$ the first equation of system (1) is transformed to the linear equation $\dot{z}_1 = pz_1$, and if

$$K_0 + \sum_{i=1}^k \alpha_i K_i = -q, \quad (11)$$

$f_0 = y + h.o.t$ and $f_i(0,0) \neq 0$ for $1 \leq i \leq k$ then the second equation of the system (1) is linearized by the change $z_2 = f_0 f_1^{\alpha_1} \dots f_k^{\alpha_k}$. If the system (1) is such that only one of the conditions (10), (11) is satisfied, say (11), but the system (1) has a Lyapunov first integral $\Psi(x, y)$ of the form

$$\Psi(x, y) = x^q y^p + \sum_{\substack{l+j=p+q+1 \\ l, j \geq 0}}^{\infty} v_{l,j} x^l y^j, \quad (12)$$

then (1) is linearizable by the change

$$z_1 = \left(\frac{\Psi(x, y)}{H_2(x, y)^p} \right)^{1/q}, \quad z_2 = H_2(x, y), \quad (13)$$

and, correspondingly, if (10) holds, then the linearizing transformation is given by

$$z_1 = H_1(x, y), \quad z_2 = \left(\frac{\Psi(x, y)}{H_1(x, y)^q} \right)^{1/p}, \quad (14)$$

as can be verified by a straightforward calculation.

3. THE LINEARIZABILITY CONDITIONS

For system (6) using (4) and (5) we computed the first six pairs of linearizability quantities $i_{3,1}, j_{3,1}, i_{6,2}, j_{6,2}, \dots, i_{18,6}, j_{18,6}$. The polynomials are too long, so we do not present them here, however the interested reader can easily compute them using any available computer algebra system. Then, using the radical membership test [4] we check that the variety of the ideal $I_6 = \langle i_{3,1}, j_{3,1}, \dots, i_{18,6}, j_{18,6} \rangle$ is the same as the variety of $I_5 = \langle i_{3,1}, j_{3,1}, \dots, i_{15,5}, j_{15,5} \rangle$. Thus, we guess that the linearizability variety of system (6) is equal to the variety of the ideal I_5 . To prove this we first look for the irreducible decomposition of the $\mathbf{V}(I_5)$. To do so, we use the routine *minAssChar* [5] of Singular [8], which finds the minimal associated primes of a polynomial ideal by means of the characteristic sets method [11]. Since the ideal is large and the calculations are very laborious in order to be able to carry out them we split our system (6) into three systems considering separately the cases:

(α) $a_{-13} = b_{3,-1} = 1$, (β) $b_{3,-1} = 0$, (γ) $a_{-13} = 0$, $b_{3,-1} = 1$.

We do not lose generality because by a linear transformation of the phase plane any system (6) can be brought to the form where one of conditions (α), (β) or (γ) is fulfilled. The results of the decompositions are the conditions presented in the following three theorems. From the reasoning given above they are the necessary conditions for the linearizability of (6). In the proofs of the following theorems we show that they are also the sufficient conditions for linearizability of system (6).

THEOREM 1. *System (6) with $b_{3,-1} = 1$, $a_{-1,3} = 1$ is linearizable if and only if one of the following conditions holds:*

- 1) $a_{02} = 5a_{11} + 3b_{11} = a_{20} + b_{20} = 25b_{02}^2 - 64b_{11} = b_{11}b_{02} + 4b_{20} = 5b_{20}b_{02} + 16 = b_{11}^2 - 5 = 4b_{20}b_{11} + 5b_{02} = 5b_{20}^2 - 4b_{11} = 0$;
- 2) $b_{20} = a_{02} - b_{02} = 3a_{11} + b_{11} = 5b_{02}^2 - 4b_{11} = 10b_{11}b_{02} + 27a_{20} = 5a_{20}b_{02} - 8 = 5b_{11}^2 + 27 = a_{20}b_{11} - 2b_{02} = 27a_{20}^2 + 16b_{11} = 0$;
- 3) $6a_{02} - b_{02} = 39a_{11} + b_{11} = 3a_{20} - 5b_{20} = 65b_{02}^2 - 256b_{11} = 5b_{11}b_{02} - 104b_{20} = 5b_{20}b_{02} - 32 = 5b_{11}^2 - 169 = 8b_{20}b_{11} - 13b_{02} = 13b_{20}^2 - 4b_{11} = 0$.

PROOF. Case 1). In this case we have $a_{20} = \frac{16}{5b_{02}}$, $a_{11} = -\frac{15b_{02}^2}{64}$, $a_{02} = 0$, $b_{20} = -\frac{16}{5b_{02}}$, $b_{11} = \frac{25b_{02}^2}{64}$ and b_{02} satisfies the equation $125b_{02}^4 - 4096 = 0$ (note that if $b_{02} = 0$ then system 1) has no solutions). Therefore $b_{02} = \pm \frac{8}{5^{3/4}}$ or $b_{02} = \pm \frac{8i}{5^{3/4}}$.

In the case $b_{02} = \frac{8}{5^{3/4}}$ system (6) takes the form

$$\begin{aligned}\dot{x} &= x - \frac{2}{5^{1/4}}x^3 + \frac{3}{5^{1/2}}x^2y - y^3, \\ \dot{y} &= -3y - \frac{2}{5^{1/4}}x^2y + 5^{1/2}xy^2 - \frac{8}{5^{3/4}}y^3 + x^3.\end{aligned}\quad (15)$$

To get rid of the radicals using the substitution

$$\tilde{x} = rx, \quad \tilde{y} = sy \quad (16)$$

with $r = -5^{3/8}$, $s = -5^{1/8}$ we transform (15) to the system

$$\begin{aligned}\dot{x} &= x - \frac{2}{5}x^3 + \frac{3}{5}x^2y - y^3, \\ \dot{y} &= -3y - \frac{2}{5}x^2y + xy^2 + \frac{8}{5}y^3 + \frac{1}{5}x^3.\end{aligned}\quad (17)$$

If $b_{02} = -\frac{8}{5^{3/4}}$ the system is transformed to (17) by substitution (16) with $r = 5^{3/8}i$, $s = -5^{1/8}i$ and if $b_{02} = \frac{8i}{5^{3/4}}$ and $b_{02} = -\frac{8i}{5^{3/4}}$ we apply (16) with $r = -(-1)^{3/4}5^{3/8}$, $s = -(-1)^{1/4}5^{1/8}$ and $r = -(-1)^{1/4}5^{3/8}$, $s = -(-1)^{3/4}5^{1/8}$, respectively.

To search for invariant lines we use the method of undetermined coefficients: we insert

$$f(x, y) = \sum_{i+j=0}^k f_{ij}x^i y^j$$

(a bound for the degree k is not known, so we try consecutively $k = 1, 2, \dots$ until we find sufficient number of the curves to construct a linearizing transformation or until the computer can complete the calculations) and

$$K(x, y) = \sum_{i+j=0}^2 K_{ij}x^i y^j$$

into (7) and collect terms. This yields a system of algebraic equations in the variables f_{ij}, K_{ij} . Solutions to the system give invariant algebraic curves and cofactors of (17). Using this approach for system (17) we found three invariant curves

$$\begin{aligned}f_1 &= x - \frac{3}{10}x^2y - \frac{3}{10}xy^2 - \frac{1}{10}x^3 - \frac{1}{10}y^3, \\ f_2 &= y - \frac{1}{10}x^2y - \frac{1}{10}xy^2 - \frac{1}{30}x^3 - \frac{1}{30}y^3, \\ f_3 &= 1 - \frac{2}{5}xy - \frac{1}{5}x^2 - \frac{1}{5}y^2,\end{aligned}$$

with respective cofactors $k_1 = 1 - \frac{3}{5}x^2 + \frac{6}{5}xy + \frac{9}{5}y^2$, $k_2 = -3 - \frac{3}{5}x^2 + \frac{6}{5}xy + \frac{9}{5}y^2$, $k_3 = -\frac{2}{5}x^2 + \frac{4}{5}xy + \frac{6}{5}y^2$.

Thus, the linearizing transformation for system (17) is

$$z_1 = f_1 f_3^{-3/2}, \quad z_2 = f_2 f_3^{-3/2}.$$

Case 2). We can proceed as in the previous case, that is, solve system 2) of the statement of the theorem and look for linearizations of the corresponding systems. Then we have to study four systems, as above. However a simpler way is to show that system 2) and the system $b_{02} - a_{02} =$

$3a_{11} + b_{11} = 8b_{11}^2 + 27a_{20}b_{02} = 4a_{-13}b_{11} - 5b_{02}^2 = a_{20}b_{11} - 2b_{3,-1}b_{02} = 8a_{-13}b_{3,-1} - 5a_{20}b_{02} = 27a_{20}a_{-13} + 10b_{11}b_{02} = 27a_{20}^2 + 16b_{3,-1}b_{11} = 0$ with $a_{-13} = b_{3,-1} = 1$ have the same set of solutions. In particular, it will be true if the Groebner bases [1] of the corresponding ideals coincide. Computing we see that they indeed coincide. It is shown in [9] that the system defined by these conditions can be transformed to the system

$$\dot{x} = x + \frac{8}{3}x^3 - x^2y - xy^2 + \frac{5}{12}y^3, \quad \dot{y} = -3y + 4x^3 - 3xy^2 + y^3. \quad (18)$$

As it is found in [9] there are three invariant algebraic curves

$$f_1(x, y) = 1 + 4x^2 + 4xy - \frac{1}{3}y^2 - \frac{4}{3}x^4 + \frac{8}{9}x^3y + \frac{2}{3}x^2y^2 - \frac{2}{3}xy^3 + \frac{5}{36}y^4,$$

$$f_2(x, y) = 12y - (2x - y)^3,$$

$$f_3(x, y) = x - \frac{1}{3}x^3 + \frac{1}{2}x^2y - \frac{1}{4}xy^2 + \frac{1}{24}y^3$$

with the cofactors $k_1(x, y) = 8x^2 - 8xy + 2y^2$, $k_2(x, y) = -3 + 2x^2 - 2xy + \frac{1}{2}y^2$, and $k_3(x, y) = 1 + 2x^2 - 2xy + \frac{1}{2}y^2$, respectively. Therefore, the first equation is linearizable by the substitution $z_1 = f_3 f_1^{-1/4}$ and the second one by the substitution $z_2 = \frac{1}{12}f_2 f_1^{-1/4}$.

Case 3). In this case we have $a_{02} = \frac{b_{02}}{6}$, $b_{11} = \frac{65b_{02}^2}{256}$, $a_{11} = -\frac{65b_{02}^2}{9984}$, $b_{20} = \frac{325b_{02}^3}{26624}$, $a_{20} = \frac{1625b_{02}^4}{79872}$ and b_{02} satisfies the equation $125b_{02}^4 - 65536 = 0$. So we have $b_{02} = \pm \frac{16}{5^{3/4}}$, $b_{02} = \pm \frac{16}{5^{3/4}}i$. In the case $b_{02} = \frac{16}{5^{3/4}}$ the system takes the form

$$\begin{aligned}\dot{x} &= x - \frac{2}{3}5^{3/4}x^3 + \frac{5^{1/2}}{15}x^2y - \frac{8}{15}5^{1/4}xy^2 - y^3 \\ \dot{y} &= -3y + \frac{2}{5^{1/4}}x^2y + \frac{13}{5^{1/2}}xy^2 + \frac{16}{5^{3/4}}y^3 + x^3.\end{aligned}$$

The substitution (16) with $r = 5^{3/8}$, $s = 5^{1/8}$ brings the system to the form

$$\begin{aligned}\dot{x} &= x - \frac{2}{3}x^3 + \frac{1}{15}x^2y - \frac{8}{15}xy^2 - y^3, \\ \dot{y} &= -3y + \frac{2}{5}x^2y + \frac{13}{5}xy^2 + \frac{16}{5}y^3 + \frac{1}{5}x^3.\end{aligned}\quad (19)$$

If $b_{02} = -\frac{16}{5^{3/4}}$ the system is transformed to (19) by (16) with $r = 5^{3/8}i$, $s = -5^{1/8}i$ and if $b_{02} = \frac{16}{5^{3/4}}i$ and $b_{02} = -\frac{16}{5^{3/4}}i$ by the substitutions with $r = -(-1)^{3/4}5^{3/8}$, $s = -(-1)^{1/4}5^{1/8}$ and $r = -(-1)^{1/4}5^{3/8}$, $s = -(-1)^{3/4}5^{1/8}$, respectively.

For system (19) we found five invariant curves

$$f_1 = x - \frac{11}{10}x^2y - \frac{19}{30}xy^2 - \frac{3}{10}x^3 - \frac{1}{10}y^3,$$

$$f_2 = y - \frac{3}{10}x^2y - \frac{9}{10}xy^2 - \frac{1}{30}x^3 - \frac{9}{10}y^3,$$

$$f_3 = 1 + \left(\frac{3}{5}\right)^{1/2}x + \left(\frac{1}{15}\right)^{1/2}y,$$

$$f_4 = 1 - \left(\frac{3}{5}\right)^{1/2}x - \left(\frac{1}{15}\right)^{1/2}y,$$

$$f_5 = 1 - 16/15x^2 + 8/25x^4 - 32/15xy + 64/5x^3y -$$

$$16/25y^2 + 464/225x^2y^2 + 256/225xy^3 + 8/45y^4.$$

Then the linearizing transformations for the first and the second equations of the system are $z_1 = f_1 f_3^{-1/6} f_4^{-1/6} f_5^{-1/2}$ and $z_2 = f_2 f_3^{5/2} f_4^{5/2} f_5^{-3/2}$, respectively. $\square$

Note that in the considered case (α) we found it helpful to first compute a Groebner basis of I_5 using the package *FGb* of [6] (we failed to compute it with *groebner* of Singular), and then apply *minAssChar* of Singular.

For most systems studied in the following two theorems we are able to find a linearization or to prove its existence only for one of the equations of the system. However each system which we consider below is a subcase of a system considered in [9], where integrals of the form (12) have been constructed (or at least it has been proven that they exist). So, if below the linearization of only one equation is proven, it is assumed that the other equation is linearized by a substitution of the form (13) or (14), where the integral $\Psi(x, y)$ exists by the results of [9]. In particular, if either differential equation in (6) is already linear then system (6) is linearizable. This covers the degenerate situations $b_{02} = 0$ in Case 1) of Theorem 2, $a_{20} = 0$ in Cases 2) and 6) of Theorem 2, $b_{20} = 0$ in Cases 1), 4), 5), and 6) of Theorem 3, and $a_{11} = 0$ in Case 3) of Theorem 3.

THEOREM 2. *System (6) with $b_{3,-1} = 0$ is linearizable if and only if one of the following conditions holds:*

- 1) $b_{11} = b_{20} = a_{-13} = a_{11} = 0$,
- 2) $b_{02} = b_{11} = a_{-13} = a_{02} = a_{11} = 0$,
- 3) $b_{02} = b_{11} = a_{02} = a_{11} = a_{20} - 3b_{20} = 0$,
- 4) $b_{02} = b_{11} = a_{02} = a_{11} = a_{20} + 3b_{20} = 0$,
- 5) $b_{02} = b_{11} = a_{02} = a_{11} = 5a_{20} - 3b_{20} = 0$,
- 6) $b_{11} = a_{13} = a_{02} = a_{11} = 2a_{20} - b_{20} = 0$,
- 7) $b_{20} = a_{20} = 0$,
- 8) $a_{-13} = a_{02} + b_{02} = a_{11} + b_{11} = a_{20} + b_{20} = 0$,
- 9) $a_{11} = a_{20} - b_{20} = 0$.

PROOF. Case 1). In this case the system has the form

$$\begin{aligned} \dot{x} &= x - a_{20} x^3 - a_{02} x y^2 = P(x, y), \\ \dot{y} &= -3y + b_{02} y^3 = Q(x, y). \end{aligned} \quad (20)$$

From the form of system (20) we see immediately that $f_1 = x$ and $f_2 = y$ are invariant lines, and their cofactors are $K_1 = P/x = 1 - a_{20} x^2 - a_{02} y^2$ and $K_2 = Q/y = (-3 + b_{02} y^2)$. We look for linearization of the first equation. We find that system (20) has two more invariant lines which do not pass through the origin:

$$f_3 = 1 + \left(\frac{b_{02}}{3}\right)^{1/2} y, \quad f_4 = 1 - \left(\frac{b_{02}}{3}\right)^{1/2} y,$$

with the cofactors

$$K_3 = -(3b_{02})^{1/2} y + b_{02} y^2, \quad K_4 = (3b_{02})^{1/2} y + b_{02} y^2.$$

We tried to solve equation (10), which in this case is $K_1 + \alpha_3 K_3 + \alpha_4 K_4 = 1$, however it had no solutions. We look for solutions of equation (11), which in this case is $K_2 + \alpha_3 K_3 + \alpha_4 K_4 = -3$ and find $\alpha_3 = \alpha_4 = -\frac{1}{2}$. Thus the linearization of the second equation is given by

$$z_2 = y f_3^{-1/2} f_4^{-1/2} = \frac{3y}{\sqrt{9 - 3b_{02} y^2}}.$$

Case 2). In this case we have 2 invariant lines: $f_1 = 1 + a_{20}^{1/2} x$, $f_2 = 1 - a_{20}^{1/2} x$. The linearization is

$$z_1 = \frac{x}{(f_1 f_2)^{1/2}}, \quad z_2 = y(f_1 f_2)^{\frac{b_{20}}{2a_{20}}}.$$

Case 3). In this case the system is:

$$\begin{aligned} \dot{x} &= x - 3b_{20} x^3 - a_{-13} y^3 = P(x, y), \\ \dot{y} &= -3y + b_{20} x^2 y = Q(x, y). \end{aligned} \quad (21)$$

We look for a linearization of the second equation in the form

$$z_2(x, y) = \sum_{k=1}^{\infty} f_k(x) y^k. \quad (22)$$

It exists if

$$\frac{\partial z_2}{\partial x} P + \frac{\partial z_2}{\partial y} Q = -3z_2, \quad (23)$$

that is, if the functions f_k satisfy the linear ordinary differential equations

$$f'_k(x) (x - 3b_{20} x^3) + (b_{20} k x^2 - 3k + 3) f_k(x) - a_{-13} f'_{k-3}(x), \quad (24)$$

where we initialize by setting $f_j(x) \equiv 0$ for $j \leq 0$. Computing we obtain $f_1(x) = (3b_{20} x^2 - 1)^{1/6}$, $f_2(x) = 0$, and $f_3(x) = 0$. We denote by p_k arbitrary polynomial of degree k in x . It is easy to see by induction that the functions f_k can be chosen in the form

$$f_{3k+1} = \frac{p_{5k}}{(3b_{20} x^2 - 1)^{4k-1/6}}, \quad f_{3k+2} = f_{3k+3} = 0, \quad (25)$$

for every $k \geq 0$, note that (25) yields

$$f'_{3k+1-3} = \frac{p_{5k-4}}{(3b_{20} x^2 - 1)^{4k-19/6}}.$$

Indeed, the solution of (24) is

$$f_k(x) = \frac{x^{3k-3}}{(3b_{20} x^2 - 1)^{4k/3-3/2}} \times \int \frac{(3b_{20} x^2 - 1)^{4k/3-3/2}}{x^{3k-3}} \frac{a_{-13} f'_{k-3}}{x(3b_{20} x^2 - 1)} dx. \quad (26)$$

When $k = 3s$ or $k = 3s + 2$ we obtain from (26) (setting the constant of integration to zero), $f_{3s} = 0$ and $f_{3s+2} = 0$, respectively. When $k = 3s + 1$ (26) gives

$$\begin{aligned} f_{3s+1}(x) &= \frac{x^{9s}}{(3b_{20} x^2 - 1)^{4s-1/6}} \times \\ &\int \frac{(3b_{20} x^2 - 1)^{4s-1/6}}{x^{9s+1}} \frac{p_{5s-4}}{(3b_{20} x^2 - 1)^{4s-13/6}} dx \\ &= \frac{x^{9s}}{(3b_{20} x^2 - 1)^{4s-1/6}} \int \frac{a_0}{x^{9s+1}} + \dots + \frac{a_{5s}}{x^{4s+1}} dx. \end{aligned} \quad (27)$$

Choosing the constant of integration to be zero we obtain after the integrations a function of the form (25). Thus, the second equation of (21) is linearizable by a substitution of the form (22).

Case 4). In this case the system is:

$$\begin{aligned} \dot{x} &= x + 3b_{20} x^3 - a_{-13} y^3 = P(x, y), \\ \dot{y} &= -3y + b_{20} x^2 y = Q(x, y). \end{aligned}$$

We look for a linearization of the second equation in the form (22). It exists if P and Q satisfy (23), that is, if the functions f_k satisfy the linear ordinary differential equations

$$f'_k(x) (x + 3b_{20} x^3) + (b_{20} k x^2 - 3k + 3) f_k(x) - a_{-13} f'_{k-3}(x), \quad (28)$$

with $f_j(x) \equiv 0$ for $j \leq 0$. Computing we obtain $f_1(x) = (3b_{20}x^2 + 1)^{-1/6}$, $f_2(x) = 0$, and $f_3(x) = 0$. As above let p_k be an arbitrary polynomial of degree k in x . By induction, similarly as above, we check that the functions f_k can be chosen in the form

$$f_{3k+1} = \frac{p_{7k}}{(3b_{20}x^2 + 1)^{5k+1/6}}, \quad f_{3k+2} = f_{3k+3} = 0 \quad (29)$$

for every $k \geq 0$.

Case 5). In this case the system is:

$$\dot{x} = -\frac{3b_{20}x^3}{5} + x - a_{-13}y^3, \quad \dot{y} = -3y + b_{20}x^2y.$$

As above, it is not difficult to show that there is a linearization (22), where the functions f_k satisfy the linear ordinary differential equations

$$(x - \frac{3}{5}b_{20}x^3)f'_k(x) + (kb_{20}x^2 - 3k + 3)f_k(x) - a_{13}f'_{k-3} = 0,$$

and have the form

$$f_{3k+1} = \frac{p_k}{(3b_{20}x^2 - 5)^{2k-5/6}}, \quad f_{3k+2} = f_{3k+3} = 0.$$

That yields the existence of a linearizing substitution (22).

Case 6). Like in case 2) the first equation is linearizable by $z_1 = \frac{x}{(f_1f_2)^{1/2}}$, where $f_1 = 1 + a_{20}^{1/2}x$, $f_2 = 1 - a_{20}^{1/2}x$.

Case 7). In this case the system is:

$$\dot{x} = x - a_{11}x^2y - a_{02}xy^2 - a_{-13}y^3, \quad \dot{y} = -3y + b_{11}xy^2 + b_{02}y^3.$$

The second equation is linearizable by a substitution (22), where f_k satisfy the differential equations

$$xf'_k(x) - (3k - 3)f_k(x) + G_k(x) = 0, \quad (30)$$

where

$$G_k(x) = -a_{11}x^2f'_{k-1}(x) - a_{02}xf'_{k-2}(x) - a_{-13}f'_{k-3}(x) + (k-1)b_{11}xf_{k-1}(x) + (k-2)b_{02}f_{k-2}(x).$$

It is easy to see that we can choose the solutions of (30) being polynomials of degree $k-1$.

Case 8). In this case we have the invariant curve $f_1 = 1 + a_{11}xy - a_{20}x^2 + \frac{1}{3}a_{02}y^2$, which allows to construct the linearization

$$z_1 = xf_1^{-1/2}, \quad z_2 = yf_1^{-1/2}.$$

Case 9). The system for this case is:

$$\dot{x} = x - a_{20}x^3 - a_{02}xy^2 - a_{-13}y^3, \\ \dot{y} = -3y + a_{20}x^2y + b_{11}xy^2 + b_{02}y^3.$$

We look for a linearization of the second equation in the form (22) with f_k satisfying the differential equations

$$(x - a_{20}x^3)f'_k(x) + (a_{20}kx^2 - 3k + 3)f_k(x) + G_k(x) = 0, \quad (31)$$

where

$$G_k(x) = -a_{02}xf'_{k-2}(x) - a_{-13}f'_{k-3}(x) + (k-1)b_{11}xf_{k-1}(x) + (k-2)b_{02}f_{k-2}(x).$$

In this case one can check that the solutions to (31) can be chosen in the form

$$f_k = \frac{p_{k-1}}{(a_{20}x^2 - 1)^{k-3/2}},$$

where, as usual, p_k are some polynomials in x of degree k . $\square$

The next theorem treats the remaining case (γ).

THEOREM 3. *System (6) with $b_{3,-1} = 1$, $a_{-1,3} = 0$ is linearizable if and only if one of the following conditions holds:*

- 1) $b_{11} = 3a_{20} - 7b_{20} = 3a_{02} + 5b_{02} = 4b_{20}^2 - 3a_{11} = 2a_{11}b_{20} + a_{02} + b_{02} = 9a_{11}^2 - 4b_{20}b_{02} = 0$;
- 2) $b_{11} = a_{11} = b_{02} = a_{02} = 0$;
- 3) $b_{20} = 3a_{11} + b_{11} = a_{20} = b_{02} = a_{02} = 0$;
- 4) $4a_{11} + b_{11} = a_{20} - 5b_{20} = a_{02} - b_{02} = 9b_{20}b_{11} - 4b_{02} = 48b_{20}^2 + b_{11} = 3b_{11}^2 + 64b_{20}b_{02} = 0$;
- 5) $4a_{11} + 3b_{11} = 5a_{20} + 7b_{20} = a_{02} - b_{02} = 9b_{20}b_{11} + 20b_{02} = 48b_{20}^2 - 25b_{11} = 15b_{11}^2 + 64b_{20}b_{02} = 0$;
- 6) $a_{11} = a_{20} - b_{20} = b_{02} = a_{02} = 0$.

PROOF. Case 1). In this case the system has the form

$$\dot{x} = x - \frac{7}{3}b_{20}x^3 - \frac{4}{3}b_{20}^2x^2y + \frac{20}{3}b_{20}^3xy^2, \\ \dot{y} = -3y + b_{20}x^2y + 4b_{20}^3y^3 + x^3.$$

In addition to $f_0 = x$ it has four invariant curves

$$f_1 = 1 + (3b_{20})^{1/2}x - \frac{2}{3^{1/2}}b_{20}^{3/2}y, \\ f_2 = 1 - (3b_{20})^{1/2}x + \frac{2}{3^{1/2}}b_{20}^{3/2}y, \\ f_3 = 1 + \left(\frac{b_{20}}{3}\right)^{1/2}x + \frac{2}{3^{1/2}}b_{20}^{3/2}y, \\ f_4 = 1 - \left(\frac{b_{20}}{3}\right)^{1/2}x - \frac{2}{3^{1/2}}b_{20}^{3/2}y.$$

So, the linearization for the first equation is

$$z_1 = xf_1^{-1/3}f_2^{-1/3}f_3^{-1/2}f_4^{-1/2}.$$

Case 2). The first equation is linearizable by the substitution

$$z_1 = \frac{x}{(1 - a_{20}x^2)^{1/2}}.$$

Case 3). We have the invariant curve $f_1 = 1 + 2a_{11}xy - \frac{1}{2}a_{11}x^4$, which yields the linearization $z_1 = xf_1^{-1/4}$.

Case 4). The system admits two invariant curves

$$f_1 = 1 + 48b_{20}^2xy - 8b_{20}x^2, \\ f_2 = 1 + 12b_{20}^2xy + b_{20}x^2 + 36b_{20}^3y^2.$$

So, the linearization for the first equation is

$$z_1 = xf_1^{-1/4}f_2^{1/2}.$$

Case 5). In this case we have the invariant curves

$$f_1 = 1 - \frac{24}{25}b_{20}^2xy + \frac{4}{5}b_{20}x^2, \\ f_2 = 1 - \frac{12}{25}b_{20}^2xy + \frac{1}{5}b_{20}x^2 + \frac{36}{125}b_{20}^3y^2.$$

Then, the first equation is linearizable by the substitution

$$z_1 = xf_1^{-1}f_2^{1/2}.$$

Case 6). The corresponding system has the invariant curves $f_1 = 1 - b_{20}^{1/2}x$, $f_2 = 1 + 2b_{20}^{1/2}x + b_{20}x^2$. So, the linearization for the first equation is

$$z_1 = xf_1^{-1/2}f_2^{-1/4}.$$

The proof is completed. $\square$

4. ACKNOWLEDGMENTS

The second author acknowledges the support of the Slovenian Research Agency, of the Nova Kreditna Banka Maribor, of TELEKOM Slovenije, and of the Transnational Access Programme at RISC-Linz of the European Commission Framework 6 Programme for Integrated Infrastructures Initiatives under the project SCIENCE (Contract No. 026133). The authors also thank the referees for their valuable remarks that helped to improve the presentation of the work.

5. REFERENCES

- [1] B. Buchberger. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems (An Algorithmical Criterion for the Solvability of Algebraic Systems of Equations). *Aequationes mathematicae*, 4(3):374–383, 1970. (English translation in: B. Buchberger, F. Winkler (eds.), *Groebner Bases and Applications*, *London Mathematical Society Lecture Note Series*, 251: 535–545, Cambridge University Press, 1998).
- [2] C. Christopher, P. Mardešić, and C. Rousseau. Normalizable, integrable and linearizable saddle points for complex quadratic systems in $\mathbb{C}^2$. *J. Dynam. Control Systems*, 9:311–363, 2003.
- [3] C. Christopher and C. Rousseau. Nondegenerate linearizable centres of complex planar quadratic and symmetric cubic systems in $\mathbb{C}^2$. *Publicacions Matemàtiques*, 45(1):95–123, 2001.
- [4] D. Cox, J. Little, and D. O’Shea. *Ideals, Varieties, and Algorithms, Second Edition*. Springer–Verlag, New York, 1997.
- [5] W. Decker, G. Pfister, and H. A. Schönemann. *Singular 2.0 library for computing the primary decomposition and radical of ideals* *primdec.lib*, 2001.
- [6] J.-C. Faugère. <http://fgbrs.lip6.fr/jcf/Software/FGb/index.html>.
- [7] J. Giné and M. Grau. Linearizability and integrability of vector fields via commutation. *J. Math. Anal. Appl.*, 319:326–332, 2006.
- [8] G.-M. Greuel, G. Pfister, and H. Schönemann. *Singular 3.0. A Computer Algebra System for Polynomial Computations*. Centre for Computer Algebra, University of Kaiserslautern, 2005.
- [9] Z. Hu, V. G. Romanovski, and D. S. Shafer. 1 : −3 resonant centers on $\mathbb{C}^2$, with homogeneous cubic nonlinearities. *Computers and Mathematics with Applications (accepted)*, 2008.
- [10] V. G. Romanovski and M. Robnik. The center and isochronicity problems for some cubic systems. *J. Phys. A: Math. Gen.*, 34:10267–10292, 2001.
- [11] D. Wang. *Elimination Methods*. Springer–Verlag, New York, 2001.

Integro-Differential Polynomials and Operators

Markus Rosenkranz and Georg Regensburger^{*}
Johann Radon Institute for Computational and Applied Mathematics (RICAM)
Austrian Academy of Sciences
Altenbergerstraße 69
A-4040 Linz, Austria
{markus.rosenkranz,georg.regensburger}@oeaw.ac.at

ABSTRACT

We propose two algebraic structures for treating integral operators in conjunction with derivations: The algebra of integro-differential polynomials describes nonlinear integral and differential operators together with initial values. The algebra of integro-differential operators can be used to solve boundary problems for linear ordinary differential equations. In both cases, we describe canonical/normal forms with algorithmic simplifiers.

Categories and Subject Descriptors

I.1.1 [Symbolic and Algebraic Manipulation]: Expressions and Their Representation—*simplification of expressions*; I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms—*algebraic algorithms*

General Terms

Theory, Algorithms

Keywords

Integral operators, integro-differential algebras, noncommutative Gröbner bases, Green’s operators, linear boundary value problems

1. INTRODUCTION

While differential operators are studied extensively in symbolic computation, this cannot be asserted about *integral operators*. In the former case, one uses two fundamental structures for transferring analysis to algebra: “differential operators” and “differential polynomials”; both of these can act on suitable function spaces (the former linearly and the latter nonlinearly). In this paper, we propose two analogous algebraic structures for treating integral operators (along

with differential operators): “integro-differential operators” and “integro-differential polynomials”; both of these are extensions of the corresponding differential structures. In Section 2, we review some notions about integro-differential algebras needed for these constructions.

The *integro-differential polynomials* are introduced here for the first time. Their construction is explained in Section 3, the computational approach in Section 4. While modeling nonlinear integral operators, the most important use of the integro-differential polynomial ring $\mathcal{F}\{u\}$ over a given integro-differential algebra $\mathcal{F}$ is probably to describe extensions of integro-differential algebras in a constructive fashion. In practice, one can start with the integro-differential algebra $\mathcal{F}_0$ of exponential polynomials, adjoin a solution of differential equations (with initial values) by passing to a quotient $\mathcal{F}_1$ of $\mathcal{F}_0\{u\}$, and iterate this procedure.

The notion of *integro-differential operators* has been introduced in [23], where it is used for multiplying and factoring BVPs (= linear boundary value problems for ordinary differential equations). In fact, one of the main applications of integro-differential operators is that they describe the differential equation, boundary conditions and solution operator (Green’s operator) of a BVP in uniform language. In [23] we have constructed a monoid on BVPs isomorphic to the compositional structure of their Green’s operators, studied in [21] from an abstract viewpoint. In this paper, we will review their construction and main properties in Section 5 and focus on computational aspects in Section 6.

For both integro-differential polynomials and operators, the crucial instrument for an algorithmic treatment is of course the usage of standard representatives, but they arise in fairly different contexts: In the former case, where we prefer to speak of *canonical forms*, we employ tools from universal algebra to build a canonical simplifier for the appropriate polynomial concept. In the latter case, where we shall use word *normal forms*, our approach is to construct a confluent rewrite system (equivalently: a noncommutative Gröbner basis).

2. INTEGRO-DIFFERENTIAL ALGEBRAS

Our starting point is a commutative *differential algebra* $(\mathcal{F}, \partial)$ over a field K , so $\partial: \mathcal{F} \rightarrow \mathcal{F}$ is a K -linear map satisfying the Leibniz rule

$$\partial(fg) = f\partial(g) + g\partial(f). \quad (1)$$

For convenience, we may assume $K \leq \mathcal{F}$, and we write f' as a shorthand for $\partial(f)$. Furthermore, we will assume that K has characteristic zero and $\mathbb{Q} \leq K$, hence $\mathcal{F}$ is what is

^{*}This work was supported by the Austrian Science Fund (FWF) under the SFB grant F1322.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC’08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

sometimes called a Ritt algebra [14, p. 12]. The algebra of differential operators over $\mathcal{F}$ is denoted by $\mathcal{F}[\partial]$ as in [27].

For inhomogeneous differential equations $Tu = f$ with $T \in \mathcal{F}[\partial]$, the solution operators (mapping $f \in \mathcal{F}$ to $u \in \mathcal{F}$) are integral operators. The simplest equation is $u' = f$, and its solution operators $\int$ are exactly the *sections* (i.e. K -linear right inverses) of the differential operator ∂ so that

$$\partial \int = 1. \quad (2)$$

Note that derivations need not have sections (for example in the algebra of univariate differential polynomials, the indeterminate cannot be a derivative).

The *characterization* of sections follows from Linear Algebra, see [19, p. 17] or [21]: Every section $\int: \mathcal{F} \rightarrow \mathcal{F}$ of the derivation $\partial: \mathcal{F} \rightarrow \mathcal{F}$ corresponds to a unique projector $P: \mathcal{F} \rightarrow \mathcal{F}$ with

$$P = 1 - \int \partial \quad (3)$$

and to a unique direct sum decomposition $\mathcal{F} = \mathcal{C} \dot{+} \mathcal{I}$ of K -vector spaces with

$$\mathcal{C} = \text{Ker}(\partial) = \text{Im}(P) \quad \text{and} \quad \mathcal{I} = \text{Im}(\int) = \text{Ker}(P).$$

Moreover, if $\int$ is any fixed section of ∂ , every projector P with $\text{Im}(P) = \text{Ker}(\partial)$ induces a section $(1 - P)\int$, and every section of ∂ arises uniquely in this way.

We refer to the elements of $\mathcal{I} = \text{Im}(\int)$ as *initialized* (with respect to $\int$), while those of $\mathcal{C} = \text{Ker}(\partial)$ are usually called the *constants* (with respect to ∂).

As a standard example, we take $\mathcal{F} = C^\infty[a, b]$ where differentiability in the endpoints is understood in the sense of one-sided derivatives. The initialized functions are those that can be written as $F(x) = \int_a^x f(\xi) d\xi$ for $f \in C^\infty[a, b]$ and an initialization point $\alpha \in [a, b]$; hence F is the unique antiderivative of f that fulfills the initial condition $F(\alpha) = 0$.

For solving inhomogeneous differential equations of higher order, one must expect to iterate the section $\int$. While this would in general lead to nested integrals, we know from the classical C^∞ setting that the Green's operator can always be expressed via the Green's function [26] by a single integration. To capture this behavior, we need an identity for resolving nested integrals (eventually leading to the rewrite rule for $\int f \int$ in Table 1). Such an identity is given by the so-called *Baxter axiom* (of weight zero), asserting

$$\int f \cdot \int g = \int f \int g + \int g \int f \quad (4)$$

for all $f, g \in \mathcal{F}$. Note that we apply the following convention in this paper: An integral like $\int f \int g$ should be interpreted as $\int (f \int g)$, unless we use $\cdot$ as on the left-hand side above.

Obviously (4) is an algebraic version of integration by parts, written in a way that does not involve the derivation. (For the integro-differential polynomials, the role of the Baxter axiom is more subtle: From the left to right, it “flattens” products of nested integrals; in the other direction, it is used for “integrating out” coefficient functions—see Section 4.) A weight-zero *Baxter algebra* $(\mathcal{F}, \int)$ is then a K -algebra $\mathcal{F}$ with a K -linear operation $\int$ fulfilling the Baxter axiom (4); we refer to [11, 2, 24] for more details.

What we shall actually use is the *differential Baxter axiom*, which requires

$$\int f g = f \int g - \int f' g \quad (5)$$

for all $f, g \in \mathcal{F}$. Note that this is what most people do when they actually apply integration by parts. Variant (4) follows

immediately by substituting $\int f$ for f in (5), and often both versions are equivalent (see after Definition 11).

We can also characterize what makes the differential Baxter axiom stronger than the pure one: A section $\int$ of ∂ fulfills the differential Baxter axiom (5) iff it fulfills the pure Baxter axiom (4) and the *homogeneity condition*

$$\int c f = c \int f \quad (6)$$

for all $c \in \mathcal{C}$ and $f \in \mathcal{F}$. In fact, (6) implies that $\int: \mathcal{F} \rightarrow \mathcal{F}$ is $\mathcal{C}$ -linear and not only K -linear.

We refer to [23] for the proof of the equivalence and for an example of a differential algebra with a section that satisfies the pure Baxter axiom but not its differential form. To exclude such cases we will insist that *integral operators* must satisfy the differential Baxter axiom.

DEFINITION 1. Let $\mathcal{F}$ be a differential algebra over a field K . A section $\int$ of ∂ is called an integral if it satisfies the differential Baxter axiom (5). In this case, we call $(\mathcal{F}, \partial, \int)$ an integro-differential algebra.

As an example, take $\mathcal{F} = C^\infty[a, b]$ with its usual derivation ∂ and integral operators

$$\int^*: f \mapsto \int_a^x f(\xi) d\xi \quad \text{and} \quad \int_*: f \mapsto \int_x^b f(\xi) d\xi.$$

Then both $(\mathcal{F}, \partial, \int^*)$ and $(\mathcal{F}, \partial, \int_*)$ are integro-differential algebras. By contrast, the operator

$$f \mapsto \int_a^b \int_\tau^x f(\xi) d\xi d\tau,$$

is just a section for ∂ , but not an integral.

In the above example, the projectors $P^*: f \mapsto f(a)$ and $P_*: f \mapsto f(b)$ corresponding to the respective integral operators $\int^*$ and $\int_*$ are *multiplicative* (see (7) below), whereas the projector $\int_a^b$ for the third operator is not. This is true in general—we can characterize integrals by their projectors or images as detailed in [23]: A section $\int: \mathcal{F} \rightarrow \mathcal{F}$ of the derivation $\partial: \mathcal{F} \rightarrow \mathcal{F}$ is an integral iff $\mathcal{I} = \text{Im}(\int)$ is an ideal of $\mathcal{F}$ iff $P = 1 - \int \partial$ is multiplicative, meaning

$$P(fg) = P(f)P(g) \quad (7)$$

for all $f, g \in \mathcal{F}$. Using the homogeneity condition (6), this implies also

$$\int f g' = f g - \int f' g - P(f)P(g) \quad (8)$$

as an equivalent formulation (corresponding to the rewrite rule for $\int f \partial$ in Table 1) of the differential Baxter axiom (5).

Similar structures are introduced under the name *differential Rota-Baxter algebras* in the recent article [12]. A crucial difference is that they only require the section axiom (2) for connecting derivation and integral, but not the differential Baxter axiom (5). They construct free objects in more general categories where the algebras are over unital commutative rings rather than fields, they may be noncommutative, and the weight can be an arbitrary scalar.

3. THE ALGEBRA OF INTEGRO-DIFFERENTIAL POLYNOMIALS

In this section, we introduce the algebra of *integro-differential polynomials* obtained by adjoining one indeterminate function to an integro-differential algebra. This is a special

case of the general construction of polynomials in universal algebra. See for example [1] for the basic notions in universal algebra that we use in the following and [7, 13, 16] for details on polynomials in universal algebra.

The idea of the construction is as follows. Let $\mathcal{V}$ be a variety defined by a set E of identities or “laws” over a signature Σ . Let A be a fixed “coefficient domain” from the variety $\mathcal{V}$, and let X be a set of “variables” or “indeterminates”. Then all terms in the signature Σ with constants (henceforth called “coefficients”) in A and variables in X represent the same polynomial if their equality can be derived in finitely many steps from the identities in E and the operations in A . The set of all such terms $\mathcal{T}_\Sigma(A \cup X)$ modulo this congruence $\equiv$ is an algebra in $\mathcal{V}$, called the *polynomial algebra* (for $\mathcal{V}$) in X over A , denoted by $A_{\mathcal{V}}[X]$.

The polynomial algebra $A_{\mathcal{V}}[X]$ contains A as a subalgebra, and $A \cup X$ is a generating set. As in the case of polynomials for commutative rings, we have the *substitution homomorphism* in general polynomial algebras. Let B be an algebra in $\mathcal{V}$. Then given a homomorphism $\varphi_1: A \rightarrow B$ and a map $\varphi_2: X \rightarrow B$, there exists a unique homomorphism

$$\varphi: A_{\mathcal{V}}[X] \rightarrow B$$

such that $\varphi(a) = \varphi_1(a)$ for all $a \in A$ and $\varphi(x) = \varphi_2(x)$ for all $x \in X$.

In order to compute with polynomials one can use an effective canonical simplifier [7], that is, a computable map

$$\sigma: \mathcal{T}_\Sigma(A \cup X) \rightarrow \mathcal{T}_\Sigma(A \cup X)$$

such that

$$\sigma(T) \equiv T \quad \text{and} \quad S \equiv T \Rightarrow \sigma(S) = \sigma(T)$$

for all terms $S, T \in \mathcal{T}_\Sigma(A \cup X)$. The representatives in $\mathcal{R} := \text{Im}(\sigma)$ are called *canonical forms*. Canonical simplifiers correspond uniquely to so-called systems of canonical forms, i.e. a set of terms

$$\mathcal{R} \subseteq \mathcal{T}_\Sigma(A \cup X)$$

such that for every $T \in \mathcal{T}_\Sigma(A \cup X)$ one can compute a canonical form $R \in \mathcal{R}$ with $T \equiv R$ and such that $R \neq \tilde{R} \Rightarrow R \not\equiv \tilde{R}$ for $R, \tilde{R} \in \mathcal{R}$. In other words, for every polynomial in $A_{\mathcal{V}}[X]$ represented by a term T one can compute a term $R \in \mathcal{R}$ representing the same polynomial, with different terms in $\mathcal{R}$ representing different polynomials, see [16, p. 23].

As a well-known example take the *polynomial ring* $R[x]$ in one indeterminate x over a commutative ring R . The set of all terms of the form $a_n x^n + \dots + a_0$ with coefficients $a_i \in R$ and $a_n \neq 0$ together with 0 is a system of canonical forms for $R[x]$. One usually defines the polynomial ring directly in terms of these canonical forms. Polynomials for groups, bounded lattices and Boolean algebras are discussed in [16] along with systems of canonical forms.

Let us now consider the variety $\mathcal{V}$ of integro-differential algebras. Its *signature* Σ contains (besides the ring operations): the derivation ∂ , the integral $\int$, the family of unary “scalar multiplications” $(\cdot)_\lambda)_{\lambda \in K}$; for convenience we also include the projection P . The *identities* E are (besides those of a K -algebra and the K -linearity of the operators $\partial, \int, P$): the Leibniz rule (1), the section axiom (2), the definition of the projection (3), and the differential Baxter axiom (5).

DEFINITION 2. Let $\mathcal{F}$ be an integro-differential algebra. Then $\mathcal{F}_{\mathcal{V}}[u]$ is called the algebra of integro-differential polynomials in u over $\mathcal{F}$ and denoted by $\mathcal{F}\{u\}$ in analogy to differential polynomials.

We will also use the following *identities* following from E and describing the basic interactions between the operations in $\mathcal{F}$: the pure Baxter axiom (4), the multiplicativity of the projection (7), the identities

$$P^2 = P, \quad \partial P = 0, \quad P \int = 0, \quad \int P(f)g = P(f) \int g, \quad (9)$$

and the variant (8) of the differential Baxter axiom connecting all three operations. Moreover, we use also the shuffle identity [25, 20] obtained from iterating the Baxter axiom

$$\int f_1 \int \dots \int f_m \cdot \int g_1 \int \dots \int g_n = \sum \int h_1 \int \dots \int h_{m+n}, \quad (10)$$

where the sum ranges over all shuffles of $(f_1, \dots, f_m)$ and $(g_1, \dots, g_n)$. By construction of the polynomial algebra, all these identities hold also for $\mathcal{F}\{u\}$.

We will use f, g for denoting coefficients in $\mathcal{F}$ and V for terms in $\mathcal{T}_\Sigma(\mathcal{F} \cup \{u\})$. As for differential polynomials, we write u_n for the n th derivative of u . We use the *multi-index notation*

$$u^\beta = \prod_{i=0}^{\infty} u_i^{\beta_i}$$

for a sequence β in $\mathbb{N}$ with only finitely many nonzero entries. The order of a differential monomial u^β is the highest derivative appearing in u^β or $-\infty$ if $\beta = 0$. Moreover, we write $V(0)$ for $P(V)$ and

$$u(0)^\alpha = \prod_{i=0}^{\infty} u_i(0)^{\alpha_i}.$$

for a multi-index α .

4. CANONICAL FORMS FOR INTEGRO-DIFFERENTIAL POLYNOMIALS

Our goal is to find a system of canonical forms for integro-differential polynomials. As a first step, we describe a system of terms that is sufficient for *representing every polynomial*, but not in a unique (canonical) way.

LEMMA 3. Every polynomial in $\mathcal{F}\{u\}$ can be represented by a finite sum of terms of the form

$$f u(0)^\alpha u^\beta \int f_1 u^{\gamma_1} \int \dots \int f_n u^{\gamma_n}, \quad (11)$$

where each multi-index as well as n may be zero.

PROOF. By induction on the structure of terms, using the identities of integro-differential algebras and the above mentioned consequences (except the differential variants of the Baxter axiom). $\square$

Note that for terms only involving the derivation, (11) gives already the usual canonical form for differential polynomials. With the aid of Lemma 3, we can now determine the *constants* in $\mathcal{F}\{u\}$.

PROPOSITION 4. Every constant in $\mathcal{F}\{u\}$ can be represented as a finite sum $\sum_\alpha c_\alpha u(0)^\alpha$ with constants c_α in $\mathcal{F}$.

PROOF. By the identity $\int \partial = 1 - P$, a term V represents a constant in $\mathcal{F}\{u\}$ iff $P(V) \equiv V$. Since V is congruent to a finite sum of terms of the form (11) and since $\text{Im}(P) = \mathcal{C}$, the identities for P imply that V is congruent to a finite sum of terms of the form $c_\alpha u(0)^\alpha$. $\square$

It is immediately clear that terms of the form (11) cannot be canonical forms for general integro-differential polynomials since for example $\int fgu$ and $\lambda^{-1} \int f \lambda gu$ with $\lambda \in K$ represent the same polynomial. This can be solved by *choosing a basis* $\mathcal{B}$ for $\mathcal{F}$ containing 1.

A second problem for canonical forms comes from the fact that we can integrate certain differential polynomials using *integration by parts* (8). For example, the terms $\int f u'$ and $f u - \int f' u - f(0)u(0)$ represent the same polynomial. More generally, we have the following identity.

LEMMA 5. *We have*

$$\int V u_k^{\beta_k} u_{k+1} \equiv \frac{1}{\beta_k + 1} \left(V u_k^{\beta_k + 1} - \int V' u_k^{\beta_k + 1} - V(0) u_k(0)^{\beta_k + 1} \right) \quad (12)$$

where $k, \beta_k \geq 0$.

PROOF. Using (8) and the Leibniz rule, we see that

$$\begin{aligned} \int V u_k^{\beta_k} u_{k+1} &= \int (V u_k^{\beta_k})(u_k)' \\ &\equiv V u_k^{\beta_k + 1} - \int V' u_k^{\beta_k + 1} - \beta_k \int V u_k^{\beta_k} u_{k+1} - V(0) u_k(0)^{\beta_k + 1}, \end{aligned}$$

and the equation follows. $\square$

In particular, if $V = f u_0^{\beta_0} \dots u_{k-1}^{\beta_{k-1}}$, then V' and hence also the right-hand side of (12) contains only differential monomials with order at most k . So if the highest derivative in the differential monomial u^β of order $k+1$ appears linearly, the term $\int f u^\beta$ is congruent to a sum of terms involving only differential monomials of order at most k . This motivates the following *classification of differential monomials*; confer also [4, 10].

DEFINITION 6. *A monomial (11), with u^β having order k , is said to have depth n and order k . It is called quasiconstant if $\beta = 0$, quasilinear if $k > 0$ and the highest derivative appears linearly; otherwise it is called functional.*

DEFINITION 7. *We write $\mathcal{R}$ for the set of all K -linear combinations of terms of the form*

$$b u(0)^\alpha u^\beta \int b_1 u^{\gamma_1} \int \dots \int b_n u^{\gamma_n}, \quad (13)$$

where $b, b_1, \dots, b_n \in \mathcal{B}$, the multi-indices α, β as well as n may be zero, and $u^{\gamma_1}, \dots, u^{\gamma_n}$ are functional.

As we will see, $\mathcal{R}$ forms a *system of canonical forms* for $\mathcal{F}\{u\}$. The easier part of this claim is that every polynomial has such a representation.

PROPOSITION 8. *Every polynomial in $\mathcal{F}\{u\}$ can be represented by a term in $\mathcal{R}$.*

PROOF. Using basis expansions and the K -linearity of the integral, we can represent with Lemma 3 every polynomial in $\mathcal{F}\{u\}$ as a K -linear combination of terms of the form

$$b u(0)^\alpha u^\beta \int b_1 u^{\gamma_1} \int \dots \int b_n u^{\gamma_n}, \quad (14)$$

where the multi-indices and n can also be zero.

With basis expansions and the identity

$$\int f \int V \equiv \int f \cdot \int V - \int V \int f,$$

coming from the pure Baxter axiom (4), we can achieve that every multi-index γ_k in (14) is nonzero (induction on depth). Using Lemma 5, one sees that a term $\int b_1 u^{\gamma_1}$ is congruent to a sum of terms involving only integral terms with functional differential monomials (induction on order). Finally one shows (induction on depth and order) that this also holds for terms of the form

$$\int b_1 u^{\gamma_1} \int \dots \int b_n u^{\gamma_n}.$$

The proposition then follows by basis expansions and the K -linearity of the integral. $\square$

It remains to show that each term in $\mathcal{R}$ represents a different polynomial. To this end, let $\langle \mathcal{R} \rangle$ be the *free vector space* over the set of terms (13). In order to distinguish the basis vectors of $\langle \mathcal{R} \rangle$ from the corresponding terms in $\mathcal{R}$, we denote them by

$$\langle b u(0)^\alpha u^\beta \int b_1 u^{\gamma_1} \int \dots \int b_n u^{\gamma_n} \rangle. \quad (15)$$

If $b, b_1, \dots, b_n$ are no basis vectors, (15) is to be understood as an abbreviation for the corresponding basis expansion. We equip the free vector space $\langle \mathcal{R} \rangle$ with the structure of an integro-differential algebra. The operations are defined on the basis vectors mimicking the corresponding operations in $\mathcal{T}_\Sigma(\mathcal{F} \cup \{u\})$, and reducing to congruent terms in $\mathcal{R}$.

The *multiplication* in $\langle \mathcal{R} \rangle$ is introduced in stages. Let J and $\tilde{J}$ range over pure integral terms $\int b_1 u^{\gamma_1} \int \dots \int b_n u^{\gamma_n}$, including 1 for $n = 0$. The product of a term $\langle b u(0)^\alpha u^\beta \rangle$ with a general term $\langle \tilde{b} u(0)^{\tilde{\alpha}} u^{\tilde{\beta}} \tilde{J} \rangle$ is defined as

$$\langle \tilde{b} \tilde{b} u(0)^{\alpha + \tilde{\alpha}} u^{\beta + \tilde{\beta}} \tilde{J} \rangle.$$

Corresponding to the shuffle identity (10), we define the product $\langle \int b u^\gamma J \rangle \langle \int \tilde{b} u^{\tilde{\gamma}} \tilde{J} \rangle$ of pure integrals recursively as

$$\langle \int b u^\gamma \rangle \star \langle J \rangle \langle \int \tilde{b} u^{\tilde{\gamma}} \tilde{J} \rangle + \langle \int \tilde{b} u^{\tilde{\gamma}} \tilde{J} \rangle \star \langle J \rangle \langle \int b u^\gamma J \rangle,$$

where $\star$ denotes the operation of nesting integrals (multiplication binds stronger than $\star$); the base case is given by the neutral element 1. With this product, the pure integral terms form a subalgebra isomorphic to the shuffle algebra so that $\cdot$ is associative and commutative. Finally, the product of two general basis vectors $\langle b u(0)^\alpha u^\beta J \rangle$ and $\langle \tilde{b} u(0)^{\tilde{\alpha}} u^{\tilde{\beta}} \tilde{J} \rangle$ is given by multiplying $\langle b u(0)^\alpha u^\beta \rangle \langle \tilde{b} u(0)^{\tilde{\alpha}} u^{\tilde{\beta}} \rangle$ with $\langle J \rangle \langle \tilde{J} \rangle$.

The *derivation* of basis vector is defined through the Leibniz rule, using also the identities $\partial P = 0$, $\partial \int = 1$ and basis expansions.

The *integral* of a basis vector is defined recursively (first by depth and then by order), based on the classification of Definition 6. In the quasiconstant case, we define

$$\int \langle b u(0)^\alpha J \rangle = \langle \int b \rangle \langle u(0)^\alpha J \rangle - \int \langle J' \rangle \int \langle b u(0)^\alpha \rangle,$$

where J' is J with the integral removed (zero for $J = 1$). For a quasilinear basis vector

$$\langle b u(0)^\alpha V u_k^{\beta_k} u_{k+1} J \rangle \quad \text{with} \quad V = u_0^{\beta_0} \dots u_{k-1}^{\beta_{k-1}},$$

we set $s = \beta_k + 1$ and define the integral by

$$\begin{aligned} s \int \langle b u(0)^\alpha V u_k^{\beta_k} u_{k+1} J \rangle \\ = \langle b u(0)^\alpha V u_k^s J \rangle - \langle u(0)^\alpha \rangle \int \langle b V J' \rangle \langle u_k^s \rangle - \langle b V u^\alpha u_k^s J \rangle(0); \end{aligned}$$

the third summand is absent unless $J = 1$.

In the functional case, we use

$$\int \langle bu(0)^\alpha u^\beta J \rangle = \langle u(0)^\alpha \int bu^\beta J \rangle,$$

as a definition for the integral.

For showing that $\langle \mathcal{R} \rangle$ is an *integro-differential algebra*, we have to verify the axioms: First of all we see that it is a commutative K -algebra by our previous remark about the shuffle product. The Leibniz rule and the section axiom follow immediately from the definition. The only difficult task is to prove the differential Baxter axiom. An easy calculation shows that

$$\int \langle u(0)^\alpha \rangle \langle R \rangle = \langle u(0)^\alpha \rangle \int \langle R \rangle.$$

Proposition 4 then implies that $\int$ is homogeneous over the constants in $\langle \mathcal{R} \rangle$. By the observation before (6), it suffices therefore to verify the pure Baxter axiom. The proof is lengthy (using inductions over depth and order, with case distinctions according to the definition of the integral) and will be presented in a subsequent publication.

PROPOSITION 9. *With the operations defined as above, $\langle \mathcal{R} \rangle$ is an integro-differential algebra.*

The integro-differential algebra $\langle \mathcal{R} \rangle$ provides the key for showing that all terms in $\mathcal{R}$ represent different polynomials of $\mathcal{F}\{u\}$.

THEOREM 10. *The terms in $\mathcal{R}$ constitute a system of canonical forms for $\mathcal{F}\{u\}$, provided that basis expansion in $\mathcal{F}$ is computable.*

PROOF. Since $\langle \mathcal{R} \rangle$ is an integro-differential algebra, there exists a unique substitution homomorphism

$$\varphi: \mathcal{F}\{u\} \rightarrow \langle \mathcal{R} \rangle$$

such that $\varphi(f) = \langle f \rangle$ for all $f \in \mathcal{F}$ and $\varphi(u) = \langle u \rangle$. Let

$$\pi: \mathcal{R} \rightarrow \mathcal{F}\{u\}$$

denote the restriction of the canonical epimorphism associated with $\equiv$. Then $\varphi \circ \pi$ is injective since it maps $R \in \mathcal{R}$ to $\langle R \rangle \in \langle \mathcal{R} \rangle$ and surjective by Proposition 8. We conclude that π is also bijective, so $\mathcal{R}$ is indeed a system of canonical forms. $\square$

5. THE ALGEBRA OF INTEGRO-DIFFERENTIAL OPERATORS

As explained in the Introduction, one important application of integro-differential polynomials is the *adjunction* of new elements to an initially given integro-differential algebra $\mathcal{F}$; this issue will be broached in a future paper. If $\mathcal{F}$ is ordinary (see Definition 11 below), we can thus ensure that a given homogeneous differential equation $Tu = 0$ with monic $T \in \mathcal{F}[\partial]$ is dimensionally adequate, meaning $\dim_K \text{Ker}(T) = \deg T$. This is the prerequisite for finding the Green's operator of the corresponding inhomogeneous equation $Tu = f$; see [23] for a detailed description of the solution method. Its groundwork consists of adding and multiplying integro-differential operators, and this is what we shall consider here.

Before giving the construction of integro-differential operators, we will explicitly restrict ourselves to *ordinary differential equations* in the following sense. Note that in the following definition our terminology deviates from [15, p. 58], where it only refers to having a single derivation.

DEFINITION 11. *A differential algebra $\mathcal{F}$ over a field K is called ordinary if $\dim_K \text{Ker}(\partial) = 1$. An integro-differential algebra $(\mathcal{F}, \partial, \int)$ is called ordinary if $(\mathcal{F}, \partial)$ is ordinary.*

As a consequence, the solution space of a homogeneous differential equation $Tu = 0$ with monic $T \in \mathcal{F}[\partial]$ is now finite-dimensional, so we can indeed enforce *dimensional adequacy* by adjunction. (The notion of saturated integro-differential algebra [23] postulates dimensional adequacy for every monic $T \in \mathcal{F}[\partial]$.)

Clearly we have $K = \mathcal{C}$ in an ordinary differential algebra $\mathcal{F}$, which is thus an algebra over its own field of constants. But then a section is automatically homogeneous over $\mathcal{C}$, so the pure Baxter axiom (4) and its differential version (5) coincide. Moreover, one knows from Linear Algebra that a *projector* P onto a one-dimensional subspace $[w]$ of a K -vector space V can be written as $P(v) = \varphi(v)w$ with a functional φ that can be made unique by the normalization $\varphi(w) = 1$. If V is a K -algebra, a projector onto $K = [1]$ is canonically described by the functional φ with $\varphi(1) = 1$. This holds in particular in an ordinary differential algebra, where the projectors (3) corresponding to sections of the derivation can be regarded as normalized functionals.

In an ordinary integro-differential algebra $\mathcal{F}$, the normalized functional corresponding to the integral $\int$ is moreover multiplicative, as explained at the end of Section 2. We call this multiplicative functional

$$\mathbf{e} = 1 - \int \partial \quad (16)$$

its *evaluation*. The terminology stems from the standard example $\mathcal{F} = C^\infty[a, b]$, where $\mathbf{e}$ is a point evaluation (see below Definition 1). The multiplicative functionals on an algebra are known as its *characters* (note that all characters are normalized). We write $\mathcal{M}(\mathcal{F})$ for the vector space of all characters on an ordinary integro-differential algebra $\mathcal{F}$, including the evaluation $\mathbf{e}$ as a distinguished character.

Let $\mathcal{F}$ be a *fixed ordinary integro-differential algebra* over a field K with evaluation $\mathbf{e}$. The variables f, g are used for elements of $\mathcal{F}$, the variables φ, ψ for elements of $\mathcal{M}(\mathcal{F})$. We introduce now an algebra of operators on $\mathcal{F}$ using rewrite systems [1] in the spirit of [3].

fg	$\rightarrow$	$f \cdot g$	∂f	$\rightarrow$	$\partial \cdot f + f \partial$
$\varphi\psi$	$\rightarrow$	ψ	$\partial\varphi$	$\rightarrow$	0
φf	$\rightarrow$	$(\varphi \cdot f) \varphi$	$\partial\int$	$\rightarrow$	1
$\int f \int$	$\rightarrow$	$(\int \cdot f) \int - \int (\int \cdot f)$			
$\int f \partial$	$\rightarrow$	$f - \int (\partial \cdot f) - (\mathbf{e} \cdot f) \mathbf{e}$			
$\int f \varphi$	$\rightarrow$	$(\int \cdot f) \varphi$			

Table 1: Rewrite System for $\mathcal{F}[\partial, \int]$

DEFINITION 12. *The integro-differential operators $\mathcal{F}[\partial, \int]$ are defined as the K -algebra generated by the symbols ∂ and $\int$, the “functions” $f \in \mathcal{F}$ and the multiplicative “functionals” $\varphi \in \mathcal{M}(\mathcal{F})$, modulo the rewrite system of Table 1.*

In the rules of Table 1, we use the notation $U \cdot f$ for the *action* of U on an element $f \in \mathcal{F}$, where U is an element of the free algebra in the above generators. It is an easy matter

to check that the rewrite rules of Table 1 are fulfilled in $\mathcal{F}$, so we may lift $\cdot$ to an action of $\mathcal{F}[\partial, \int]$ on $\mathcal{F}$. In particular, $f \cdot g$ now denotes the product in $\mathcal{F}$.

We remark that Table 1 is to be understood as including *implicit rules* for $\int\int$, $\int\partial$ and $\int\varphi$ by substituting $f = 1$ in the rules for $\int f\int$, $\int f\partial$ and $\int f\varphi$, respectively. Moreover, one obtains the *derived rule* $\mathbb{E}\int = 0$ from the definition of the evaluation $\mathbb{E}$. Note also that $\mathcal{F}[\partial] \subseteq \mathcal{F}[\partial, \int]$, with the same induced action on $\mathcal{F}$.

THEOREM 13. *The rewrite system for $\mathcal{F}[\partial, \int]$ in Table 1 is convergent.*

In other words, the polynomials given by the difference between the left-hand and right-hand sides of Table 1 form a two-sided noncommutative *Gröbner basis*. The proof is given in [23]. For the theory of Gröbner bases, we refer to [5, 6], for its noncommutative extension to [17, 18].

6. NORMAL FORMS FOR INTEGRO-DIFFERENTIAL OPERATORS

Having a convergent rewrite system, every integro-differential operator has a *unique normal form* [1, p. 12]. To compute such normal forms we also need a canonical simplifier on the free algebra generated by ∂ and $\int$, the functions $f \in \mathcal{F}$ and the functionals $\varphi \in \mathcal{M}(\mathcal{F})$; one possibility is by basis expansion in $\mathcal{F}$. Here we summarize the description of the normal forms on $\mathcal{F}[\partial, \int]$ obtained in [23].

We first consider operators in the right ideal

$$\mathcal{S}(\mathcal{F}) = \mathcal{M}(\mathcal{F}) \mathcal{F}[\partial, \int],$$

which we call *Stieltjes boundary conditions* over $\mathcal{F}$ or “boundary conditions” for short. Every such boundary condition has the normal form

$$\sum_{\varphi \in \mathcal{M}(\mathcal{F})} \left(\sum_{i \in \mathbb{N}} a_{\varphi, i} \varphi \partial^i + \varphi \int f_{\varphi} \right)$$

with $a_{\varphi, i} \in K$ and $f_{\varphi} \in \mathcal{F}$ almost all zero. We write $\mathcal{F}[\mathbb{E}]$ for the left $\mathcal{F}$ -submodule generated by $\mathcal{S}(\mathcal{F})$ and call them *Stieltjes boundary operators* or “boundary operators” for short.

With the rule for ∂f of Table 1 it is clear that the *differential operators* $\mathcal{F}[\partial] \subset \mathcal{F}[\partial, \int]$ have their usual normal forms. Analogously, we write $\mathcal{F}[\int] \subset \mathcal{F}[\partial, \int]$ for the subalgebra of *integral operators*, generated by the functions and $\int$ modulo the rule for $\int f\int$ of Table 1; one sees immediately that that their normal forms are linear combinations of $f\int g$ with $f, g \in \mathcal{F}$.

THEOREM 14. *Up to ordering the summands, every normal form of $\mathcal{F}[\partial, \int]$ with respect to the rewrite system of Table 1 can be written uniquely as a sum $T + G + B$ with $T \in \mathcal{F}[\partial]$ and $G \in \mathcal{F}[\int]$ and $B \in \mathcal{F}[\mathbb{E}]$.*

We can use integro-differential operators for specifying and solving boundary problems. Since space is limited, we can only state the main results here; for details and complete proofs, we must again refer to [23]. We formulate the *boundary problem* for a monic differential operator $T \in \mathcal{F}[\partial]$ with $\deg T = n$ and Stieltjes boundary conditions $\beta_1, \dots, \beta_n \in \mathcal{S}(\mathcal{F})$ as follows.

Given a forcing function $f \in \mathcal{F}$, find $u \in \mathcal{F}$ such that

$$\begin{aligned} Tu &= f, \\ \beta_1 u &= \dots = \beta_n u = 0. \end{aligned} \tag{17}$$

We call the boundary problem regular if there is a unique $u \in \mathcal{F}$ for every $f \in \mathcal{F}$; this implies in particular that $\beta_1, \dots, \beta_n$ are linearly independent over K .

The first step in solving (17) is to consider the corresponding *initial value problem* based on a character $\eta \in \mathcal{M}(\mathcal{F})$, where one replaces the boundary conditions $\beta_1, \dots, \beta_n$ by $\eta, \eta\partial, \dots, \eta\partial^{n-1}$. Note that one may in particular choose $\eta = \mathbb{E}$, evaluating in the initialization point. The main idea of solving initial value problems is of course an adaption of the familiar variation-of-constants formula (see for example in [9, p. 74] for systems and [9, p. 87] for scalar differential equations).

PROPOSITION 15. *Let $T \in \mathcal{F}[\partial]$ be a monic differential operator with $\deg T = n$ such that $Tu = 0$ has a fundamental system of solutions $u_1, \dots, u_n \in \mathcal{F}$. If W is its Wronskian matrix and $d = \det W$ is invertible in $\mathcal{F}$, the initial value problem $Tu = f$ based on $\eta \in \mathcal{M}(\mathcal{F})$ has the unique solution*

$$u = \sum_{i=1}^n u_i (1 - \eta) \int d^{-1} d_i f \tag{18}$$

for every forcing function $f \in \mathcal{F}$. Here $d_i = \det W_i$, where W_i is the matrix obtained from W by replacing the i th column by the n th unit vector.

PROOF. We can use the usual technique of reformulating $Tu = f$ as a system of linear first-order differential equations with companion matrix $A \in \mathcal{F}^{n \times n}$. The integral operator

$$\mathfrak{f} = (1 - \eta) \int$$

is a section of ∂ with corresponding projector $1 - \mathfrak{f}\partial = \eta$. Since η is multiplicative, we know from Section 2 that $\mathfrak{f}$ is an integral. We extend the action of the operators $\mathfrak{f}, \partial, \eta$ componentwise to $\mathcal{F}^n$. Setting now

$$\hat{u} = W \mathfrak{f} W^{-1} \hat{f}$$

with $\hat{f} = (0, \dots, 0, f)^T \in \mathcal{F}^n$, one may readily check that $\hat{u} \in \mathcal{F}^n$ is a solution of the first-order system $\hat{u}' = A\hat{u} + \hat{f}$ with initial condition $\eta\hat{u} = 0$. Writing u for the first component of $\hat{u}$, we have a solution of the initial value problem $Tu = f$ based on $\eta \in \mathcal{M}(\mathcal{F})$. Using Cramer’s rule to compute the n th column of W^{-1} , we see that

$$W^{-1} \hat{f} = d^{-1} f (d_1, \dots, d_n)^T,$$

and (18) follows since the first row of W is $(u_1, \dots, u_n)$.

For proving uniqueness, assume $Tu = 0$ along with the initial conditions $\eta u = \dots = \eta u^{(n-1)} = 0$. Let

$$u = c_1 u_1 + \dots + c_n u_n$$

with coefficients in K . Then the initial conditions yield $\eta(Wc) = 0$ with $c = (c_1, \dots, c_n)^T \in K^n$. But $\eta(Wc) = \eta(W)c$ because η is linear, and $\det \eta(W) = \eta(\det W)$ because it is moreover multiplicative. Since $\det W \in \mathcal{F}$ is invertible, this implies that $\eta(W) \in K^{n \times n}$ is regular, so $c = \eta(W)^{-1} 0 = 0$ and $u = 0$. $\square$

The above proposition hinges on two conditions: The first has already been discussed and can be satisfied by adjunction. The second condition needs an *invertible Wronskian* d .

This could also be enforced by a suitable localization of $\mathcal{F}$, as for Picard-Vessiot rings [27, p. 12]. But in many applications, this condition will come out naturally: The Wronskian d is always an exponential over $\mathcal{F}$ since it satisfies the differential equation $d' = ad$, where a is the trace of the companion matrix A .

Since every integro-differential algebra $\mathcal{F}$ comes with the evaluation $\eta = \mathbf{E}$ as a distinguished character, we can speak of the initial value problem associated with a monic $T \in \mathcal{F}[\partial]$. Then the map $T^\diamond : \mathcal{F} \rightarrow \mathcal{F}$ described by the assignment $f \mapsto u$ in (18) simplifies to

$$T^\diamond = \sum_{i=1}^n u_i \int d^{-1} d_i. \quad (19)$$

We call $T^\diamond \in \mathcal{F}[\partial, \int]$ the *fundamental right inverse* of T . Note that (19) can be further simplified if T has constant coefficients; see [22].

The next step in solving (17) is to compute the *projector* onto $\text{Ker}(T) = [u_1, \dots, u_n]$ along

$$[\beta_1, \dots, \beta_n]^\perp = \{u \in \mathcal{F} \mid \beta_1 u = \dots = \beta_n u = 0\},$$

which can be achieved as follows: Change from the basis $\beta_1, \dots, \beta_n$ of $[\beta_1, \dots, \beta_n]$ to a new basis $\tilde{\beta}_1, \dots, \tilde{\beta}_n$ over K biorthogonal to $u_1, \dots, u_n$ in the sense that $\tilde{\beta}_i(u_j) = \delta_{ij}$. Then the projector can be determined as

$$P = u_1 \tilde{\beta}_1 + \dots + u_n \tilde{\beta}_n \in \mathcal{F}[\mathbf{E}].$$

See [23] for further details.

We can now put everything together for determining the *Green's operator* $G: f \mapsto u$ of (17). The point is that $T^\diamond$ solves the initial value problem, while $1 - P$ “translates” the initial conditions $\mathbf{E}, \mathbf{E}\partial, \dots, \mathbf{E}\partial^{n-1}$ to the required boundary conditions $\beta_1, \dots, \beta_n$.

THEOREM 16. *Let $T \in \mathcal{F}[\partial]$ be monic with $\deg T = n$ and $\beta_1, \dots, \beta_n \in \mathcal{S}(\mathcal{F})$ such that the boundary problem (17) is regular. If the conditions of Proposition 15 are satisfied, the Green's operator of (17) is given by*

$$G = (1 - P) T^\diamond,$$

where P is the projector onto $\text{Ker}(T)$ along $[\beta_1, \dots, \beta_n]^\perp$.

PROOF. Let $u_1, \dots, u_n$ be a fundamental system for T . We have $TG = TT^\diamond - PT^\diamond = 1 - 0$ since P annihilates $u_1, \dots, u_n$. Thus $u = Gf$ satisfies the differential equation $Tu = f$ of (17).

For ensuring the boundary conditions of (17), we prove $\beta_i G = 0$ for $i = 1, \dots, n$. But we have even $\beta_i(1 - P) = 0$ because $1 - P$ projects onto $[\beta_1, \dots, \beta_n]^\perp$. $\square$

In analysis, the Green's operator G is usually written as an integral operator with the bivariate *Green's function* as its kernel. As remarked in Section 2, this is the effect of the Baxter axiom. Hence the abstract version of a Green's function is the Green's operator $G \in \mathcal{F}[\partial, \int]$ written in its normal form. In the classical $C^\infty[a, b]$ setting, there is indeed a straight-forward correspondence between normal forms and Green's functions [22].

7. CONCLUSION

We have presented two algorithmic tools for studying integration from an *algebraic operator perspective*. The integro-differential polynomials, introduced for the first time in this

paper, enjoy a rich structure that deserves further analysis. Specifically, their quotient algebras are relevant in view of adjunctions (see at the beginning of Section 5). Unlike the integro-differential polynomials, the integro-differential operators are an algebraic model of linear operators, based on the (noncommutative) compositional structure. Their normal forms are much easier to describe since one can fall back on Gröbner basis methods. We can benefit from both in the study of differential equations—particularly when considered with boundary conditions.

Acknowledgments

We would like to thank our project leaders *Bruno Buchberger* and *Heinz W. Engl* for their continuous support, critical comments and helpful suggestions.

8. REFERENCES

- [1] F. Baader and T. Nipkow. *Term rewriting and all that*. Cambridge University Press, Cambridge, 1998.
- [2] G. Baxter. An analytic problem whose solution follows from a simple algebraic identity. *Pacific J. Math.*, 10:731–742, 1960.
- [3] G. M. Bergman. The diamond lemma for ring theory. *Adv. in Math.*, 29(2):178–218, 1978.
- [4] A. H. Bilge. A REDUCE program for the integration of differential polynomials. *Comput. Phys. Comm.*, 71(3):263–268, 1992.
- [5] B. Buchberger. *An algorithm for finding the bases elements of the residue class ring modulo a zero dimensional polynomial ideal (German)*. PhD thesis, Univ. of Innsbruck, 1965. English translation published in *J. Symbolic Comput.*, 41(3-4):475–511, 2006.
- [6] B. Buchberger. Introduction to Gröbner bases. 1998. In [8], pp. 3–31.
- [7] B. Buchberger and R. Loos. Algebraic simplification. In *Computer algebra*, pages 11–43. Springer, Vienna, 1983.
- [8] B. Buchberger and F. Winkler, editors. *Gröbner bases and applications*, volume 251 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, Cambridge, 1998. Papers from the Conference on 33 Years of Gröbner Bases held at the University of Linz, Linz, February 2–4, 1998.
- [9] E. A. Coddington and N. Levinson. *Theory of ordinary differential equations*. McGraw-Hill Book Company, Inc., New York-Toronto-London, 1955.
- [10] I. M. Gelfand and L. A. Dikiĭ. Fractional powers of operators, and Hamiltonian systems. *Funkcional. Anal. i Priložen.*, 10(4):13–29, 1976. English translation: *Functional Anal. Appl.* 10 (1976), no. 4, 259–273 (1977).
- [11] L. Guo. Baxter algebras and differential algebras. In *Differential algebra and related topics (Newark, NJ, 2000)*, pages 281–305. World Sci. Publ., River Edge, NJ, 2002.
- [12] L. Guo and W. Keigher. On differential Rota-Baxter algebras. *J. Pure Appl. Algebra*, 212(3):522–540, 2008.
- [13] H. Hule. Polynome über universalen Algebren. *Monatsh. Math.*, 73:329–340, 1969.

- [14] I. Kaplansky. *An introduction to differential algebra*. Publ. Inst. Math. Univ. Nancago, No. 5. Hermann, Paris, 1957.
- [15] E. Kolchin. *Differential algebra and algebraic groups*, volume 54 of *Pure and Applied Mathematics*. Academic Press, New York-London, 1973.
- [16] H. Lausch and W. Nöbauer. *Algebra of polynomials*, volume 5 of *North-Holland Mathematical Library*. North-Holland Publishing Co., Amsterdam, 1973.
- [17] F. Mora. Groebner bases for non-commutative polynomial rings. In *AAECC-3: Proceedings of the 3rd International Conference on Algebraic Algorithms and Error-Correcting Codes*, pages 353–362, London, UK, 1986. Springer-Verlag.
- [18] T. Mora. An introduction to commutative and noncommutative Gröbner bases. *Theoret. Comput. Sci.*, 134(1):131–173, 1994. Second International Colloquium on Words, Languages and Combinatorics (Kyoto, 1992).
- [19] M. Z. Nashed and G. F. Votruba. A unified operator theory of generalized inverses. In M. Z. Nashed, editor, *Generalized inverses and applications (Proc. Sem., Math. Res. Center, Univ. Wisconsin, Madison, Wis., 1973)*, pages 1–109. Academic Press, New York, 1976.
- [20] R. Ree. Lie elements and an algebra associated with shuffles. *Ann. of Math. (2)*, 68:210–220, 1958.
- [21] G. Regensburger and M. Rosenkranz. An algebraic foundation for factoring linear boundary problems. *Ann. Mat. Pura Appl. (4)*, 2008. DOI:10.1007/s10231-008-0068-3.
- [22] M. Rosenkranz. A new symbolic method for solving linear two-point boundary value problems on the level of operators. *J. Symbolic Comput.*, 39(2):171–199, 2005.
- [23] M. Rosenkranz and G. Regensburger. Solving and factoring boundary problems for linear ordinary differential equations in differential algebras. *J. Symbolic Comput.*, 2007. DOI:10.1016/j.jsc.2007.11.007.
- [24] G.-C. Rota. Baxter algebras and combinatorial identities (I, II). *Bull. Amer. Math. Soc.*, 75:325–334, 1969.
- [25] G.-C. Rota. Ten mathematics problems I will never solve. *Mitt. Dtsch. Math.-Ver.*, (2):45–52, 1998.
- [26] I. Stakgold. *Green's functions and boundary value problems*. John Wiley & Sons, New York, 1979.
- [27] M. van der Put and M. F. Singer. *Galois theory of linear differential equations*, volume 328 of *Grundlehren der Mathematischen Wissenschaften*. Springer, Berlin, 2003.

Power Series Composition and Change of Basis

Alin Bostan
Algorithms Project
INRIA Rocquencourt
France
Alin.Bostan@inria.fr

Bruno Salvy
Algorithms Project
INRIA Rocquencourt
France
Bruno.Salvy@inria.fr

Éric Schost
ORCCA and CSD
University of Western Ontario
London, ON, Canada
eschost@uwo.ca

ABSTRACT

Efficient algorithms are known for many operations on truncated power series (multiplication, powering, exponential, ...). Composition is a more complex task. We isolate a large class of power series for which composition can be performed efficiently. We deduce fast algorithms for converting polynomials between various bases, including Euler, Bernoulli, Fibonacci, and the orthogonal Laguerre, Hermite, Jacobi, Krawtchouk, Meixner and Meixner-Pollaczek.

Categories and Subject Descriptors:

I.1.2 [Computing Methodologies]: Symbolic and Algebraic Manipulation – *Algebraic Algorithms*

General Terms: Algorithms, Theory

Keywords: Fast algorithms, transposed algorithms, basis conversion, orthogonal polynomials.

1. INTRODUCTION

Through the Fast Fourier Transform, fast polynomial multiplication has been the key to devising efficient algorithms for polynomials and power series. Using techniques such as Newton iteration or divide-and-conquer, many problems have received satisfactory solutions: polynomial evaluation and interpolation, power series exponentiation, logarithm, ... can be performed in quasi-linear time.

In this article, we discuss two questions for which such fast algorithms are not known: power series composition and change of basis for polynomials. We isolate special cases, including most common families of orthogonal polynomials, for which our algorithms reach quasi-optimal complexity.

Composition. Given a power series g with coefficients in a field $\mathbb{K}$, we first consider the map of evaluation at g

$$\text{Eval}_{m,n}(\cdot, g) : A \in \mathbb{K}[x]_m \mapsto A(g) \bmod x^n \in \mathbb{K}[x]_n.$$

Here, $\mathbb{K}[x]_m$ is the m -dimensional $\mathbb{K}$ -vector space of polynomials of degree less than m . We note Eval_n for $\text{Eval}_{n,n}$.

To study this problem, as usual, we denote by M a *multiplication time* function, such that polynomials of degree

less than n can be multiplied in $M(n)$ operations in $\mathbb{K}$. We impose the usual super-linearity conditions of [17, Chap. 8]. Using Fast Fourier Transform algorithms, $M(n)$ can be taken in $O(n \log(n))$ over fields with suitable roots of unity, and $O(n \log(n) \log \log(n))$ in general [31, 14].

If $g(0) = 0$, the best known algorithm, due to Brent and Kung, uses $O(\sqrt{n \log n} M(n))$ operations in $\mathbb{K}$ [11]; in small characteristic, a quasi-linear algorithm is known [5]. There are however special cases of power series g with faster algorithms: evaluation at $g = \lambda x$ takes linear time; evaluation at $g = x^k$ requires *no* arithmetic operation. A non-trivial example is $g = x + a$, which takes time $O(M(n))$ when the base field has characteristic zero or large enough [1]. Brent and Kung [11] also showed how to obtain a cost in $O(M(n) \log(n))$ when g is a polynomial; this was extended by van der Hoeven [22] to the case where g is algebraic over $\mathbb{K}(x)$. In §2, we prove that evaluation at $g = \exp(x) - 1$ and at $g = \log(1 + x)$ can also be performed in $O(M(n) \log(n))$ operations over fields of characteristic zero or larger than n .

Using associativity of composition and the linearity of the map $\text{Eval}_{m,n}$, we show in §2 how to use these special cases as building blocks, to obtain fast evaluation algorithms for a large class of power series. This idea was first used by Pan [28], who applied it to functions of the form $(ax + b)/(cx + d)$. Our extensions cover further examples such as $2x/(1 + x)^2$ or $(1 - \sqrt{1 - x^2})/x$, for which we improve the previously known costs.

Bivariate problems. Our results on the cost of evaluation (and of the transposed operation) are applied in §3 to special cases of a more general composition, reminiscent of umbral operations [30]. Given a *bivariate* power series $\mathbf{F} = \sum_{j \geq 0} \xi_j(x) t^j$, we consider the linear map

$$\text{Eval}_n(\cdot, \mathbf{F}, t) : (a_0, \dots, a_{n-1}) \mapsto \sum_{j < n} \xi_j(x) a_j \bmod x^n.$$

For instance, with

$$\mathbf{F} = \frac{1}{1 - tg(x)} = \sum_{j \geq 0} g(x)^j t^j,$$

this is the map $\text{Eval}_n(\cdot, g)$ seen before. For general $\mathbf{F}$, the conversion takes quadratic time (one needs n^2 coefficients for $\mathbf{F}$). Hence, better algorithms can only been found for structured cases; in §3, we isolate a large family of bivariate series $\mathbf{F}$ for which we can provide such fast algorithms. This approach follows Frumkin's [16], which was specific to Legendre polynomials.

Change of basis. Our framework captures in particular the generating series of many classical polynomial families,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

for which it yields at once conversion algorithms between the monomial and polynomial bases, in both directions.

Thus, we obtain in §4 change of basis algorithms of cost only $O(M(n))$ for all of Jacobi, Laguerre and Hermite orthogonal polynomials, as well as Euler, Bernoulli, and Mott polynomials (see Table 3). These algorithms are derived in a uniform manner from our composition algorithms; they improve upon the existing results, of cost $O(M(n) \log(n))$ or $O(M(n) \log^2(n))$ at best (see below for historical comments).

We also obtain $O(M(n) \log(n))$ conversion algorithms for a large class of Sheffer sequences [30, Chap. 2], including actuarial polynomials, Poisson-Charlier polynomials and Meixner polynomials (see Table 4).

Transposition. A key aspect of our results is their heavy use of transposed algorithms. Introduced under this name by Kaltofen and Shoup, the *transposition principle* is an algorithmic theorem with the following content: given an algorithm that performs an $r \times s$ matrix-vector product $b \mapsto Mb$, one can deduce an algorithm with the same complexity, up to $O(r+s)$ operations, and that performs the transposed matrix-vector product $c \mapsto M^t c$. In other words, this relates the cost of computing a $\mathbb{K}$ -linear map $f: V \rightarrow W$ to that of computing the transposed map $f^t: W^* \rightarrow V^*$.

For the transposition principle to apply, some restrictions must be imposed on the computational model: we require that only linear operations in the coefficients of b are performed (all our algorithms satisfy this assumption). See [12] for a precise statement, Kaltofen’s “open problem” [23] for further comments and [7] for a systematic review of some classical algorithms from this viewpoint.

To make the design of transposed algorithms transparent, we choose as much as possible to describe our algorithms in a “functional” manner. Most of our questions boil down to computing linear maps $\mathbb{K}[x]_m \rightarrow \mathbb{K}[x]_n$; expressing algorithms as a factorization of these maps into simpler ones makes their transposition straightforward. In particular, this leads us to systematically indicate the dimensions of the source (and often target) space as a subscript.

Previous work. The question of efficient change of basis has naturally attracted a lot of attention, so that fast algorithms are already known in many cases.

Gerhard [18] provides $O(M(n) \log(n))$ conversion algorithms between the falling factorial basis and the monomial basis: we recover this as a special case. The general case of Newton interpolation is discussed in [6, p. 67] and developed in [9]. The algorithms have cost $O(M(n) \log(n))$ as well.

More generally, if (P_i) is a sequence of polynomials satisfying a recurrence relation of fixed order (such as an orthogonal family), the conversion from (P_i) to the monomial basis (x^i) can also be computed in $O(M(n) \log(n))$ operations: an algorithm is given in [29], and an algorithm for the transpose problem is in [15]. Both operate on real or complex arguments, but the ideas extend to more general situations. Alternative algorithms, based on structured matrices techniques, are given in [21]. They perform conversions in both directions in cost $O(M(n) \log^2(n))$.

The overlap with our results is only partial: not all families satisfying a fixed order recurrence relation fit in our framework; conversely, our method applies to families which do not necessarily satisfy such recurrences (the work-in-progress [8] specifically addresses conversion algorithms for orthogonal polynomials).

Besides, special algorithms are known for converting be-

tween particular families, such as Chebyshev, Legendre and Bézier [27, 4], with however a quadratic cost. Floating-point algorithms are known as well, of cost $O(n)$ for conversion from Legendre to Chebyshev bases [2] and $O(n \log(n))$ for conversions between Gegenbauer bases [25], but the results are approximate. Approximate conversions for the Hermite basis are discussed in [26], with cost $O(M(n) \log(n))$.

Note on the base field. For the sake of simplicity, in all that follows, the base field is supposed to have characteristic 0. All results actually hold more generally, for fields whose characteristic is sufficiently large with respect to the target precision of the computation. However, completely explicit estimates would make our statements cumbersome.

2. COMPOSITION

Associativity of composition can be read both ways: in the identity $A(f \circ g) = A(f) \circ g$, f is either composed on the left of g or on the right of A . In this section, we discuss the consequences of this remark. We first isolate a class of operators f for which both left and right composition can be computed fast. Most results are known; we introduce two new ones, regarding exponentials and logarithms. Using these as building blocks, we then define *composition sequences*, which enable us to obtain more complex functions by iterated compositions. We finally discuss the cost of the map Eval_n and of its inverse for such functions, showing how to reduce it to $O(M(n))$ or $O(M(n) \log(n))$.

2.1 Basic Subroutines

We now describe a few basic subroutines that are the building blocks in the rest of this article.

Left operations on power series. In Table 1, we list basic composition operators, defined on various subsets of $\mathbb{K}[[x]]$. Explicitly, any such operator $\circ$ is defined on a *domain* $\text{dom}(\circ)$, given in the third column. Its action on a power series $g \in \text{dom}(\circ)$ is given in the second column, and the cost of computing $\circ(g) \bmod x^n$ is given in the last column.

Operator	Action	Domain	Cost
A_a (add)	$a + g$	$\mathbb{K}[[x]]$	1
M_λ (mul)	λg	$\mathbb{K}[[x]]$	n
P_k (power)	g^k	$\mathbb{K}[[x]]$	$O(\log k + M(n))$
$R_{k,\alpha,r}$ (root)	$g^{1/k}$	$\alpha^k x^{rk} (1 + x\mathbb{K}[[x]])$	$O(M(n))$
Inv (inverse)	$1/g$	$\mathbb{K}^* + x\mathbb{K}[[x]]$	$O(M(n))$
E (exp.)	$\exp(g) - 1$	$x\mathbb{K}[[x]]$	$O(M(n))$
L (log.)	$\log(1 + g)$	$x\mathbb{K}[[x]]$	$O(M(n))$

Table 1: Basic Operations on Power Series

Some comments are in order. For addition and multiplication, we take $a \in \mathbb{K}$ and λ in $\mathbb{K}^*$. To lift indeterminacies, the value of $R_{k,\alpha,r}(g)$ is defined as the unique power series with leading term αx^r whose k th power is g ; observe that to compute $R_{k,\alpha,r}(g) \bmod x^n$, we need g modulo $x^{n+r(k-1)}$ as input. Finally, we choose to subtract 1 to the exponential so as to make it the inverse of the logarithm. All complexity results are known; they are obtained by Newton iteration [10].

Right operations on polynomials. In Table 2, we describe a few basic linear maps on $\mathbb{K}[x]_m$ (observe that the dimension m of the source is mentioned as a subscript). Their action on a polynomial

$$A(x) = a_0 + \cdots + a_{m-1} x^{m-1} \in \mathbb{K}[x]_m$$

Name	Notation	Action	Cost
Powering	$\text{Power}_{m,k}$	$A(x^k)$	0
Reversal	Rev_m	$x^{m-1}A(1/x)$	0
Mod	$\text{mod}_{m,n}$	$A \bmod x^n$	0
Scale	$\text{Scale}_{\lambda,m}$	$A(\lambda x)$	$O(m)$
Diagonal	$\Delta_m(\cdot, s_i)$	$\sum a_i s_i x^i$	m
Multiply	$\text{Mul}_{m,n}(\cdot, P)$	$AP \bmod x^n$	$M(\max(n, m))$
Shift	$\text{Shift}_{a,m}$	$A(x+a)$	$M(m) + O(m)$

Table 2: Basic Operations on Polynomials

is described in the third column. In the case of powering, it is assumed that $k \in \mathbb{N}_{>0}$. Here and in what follows, we freely identify $\mathbb{K}[x]_m$ and $\mathbb{K}^m$, through the isomorphism

$$\sum_{i < m} a_i x^i \in \mathbb{K}[x]_m \leftrightarrow (a_0, \dots, a_{m-1}) \in \mathbb{K}^m.$$

All of the cost estimates are straightforward, except for the shift, which, in characteristic 0, can be deduced from the other ones by the factorization [1]:

$$\text{Shift}_{a,m} = \Delta_m(\text{Rev}_m(\text{Mul}_{m,m}(\text{Rev}_m(\Delta_m(\cdot, i!)), P)), 1/i!),$$

where P is the polynomial $\sum_{i=0}^{n-1} a^i x^i / i!$. We continue with some equally simple operators, whose description however requires some more detail. For $k \in \mathbb{N}_{>0}$, any polynomial A in $\mathbb{K}[x]$ can be uniquely written as

$$A(x) = A_{0/k}(x^k) + A_{1/k}(x^k)x + \dots + A_{k-1/k}(x^k)x^{k-1}.$$

Inspecting degrees, one sees that if A is in $\mathbb{K}[x]_m$, then $A_{i/k}$ is in $\mathbb{K}[x]_{m_i}$, with

$$m_i = \lfloor m/k \rfloor + \begin{cases} 1 & \text{if } i \leq m \bmod k, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

This leads us to define the map $\text{Split}_{m,k}$:

$$A \in \mathbb{K}[x]_m \mapsto (A_{0/k}, \dots, A_{k-1/k}) \in \mathbb{K}[x]_{m_0} \times \dots \times \mathbb{K}[x]_{m_{k-1}}.$$

It uses no arithmetic operation. We also use linear combination with polynomial coefficients. Given polynomials $G_0, \dots, G_{k-1}$ in $\mathbb{K}[x]_m$, we denote by

$$\text{Comb}_m(\cdot, G_0, \dots, G_{k-1}) : \mathbb{K}[x]_m^k \rightarrow \mathbb{K}[x]_m$$

the map sending $(A_0, \dots, A_{k-1}) \in \mathbb{K}[x]_m^k$ to

$$A_0 G_0 + \dots + A_{k-1} G_{k-1} \bmod x^m \in \mathbb{K}[x]_m.$$

It can be computed in $O(kM(m))$ operations. Finally, we extend our set of subroutines on polynomials with the following new results on the evaluation at $\exp(x) - 1$ and $\log(1+x)$.

PROPOSITION 1. *The maps*

$$\text{Exp}_{m,n} : A \in \mathbb{K}[x]_m \mapsto A(\exp(x) - 1) \bmod x^n \in \mathbb{K}[x]_n,$$

$$\text{Log}_{m,n} : A \in \mathbb{K}[x]_m \mapsto A(\log(1+x)) \bmod x^n \in \mathbb{K}[x]_n$$

can be computed in $O(M(n) \log(n))$ arithmetic operations.

PROOF. We start by truncating A modulo x^n , since

$$\text{Exp}_{m,n}(A) = \text{Exp}_{m,n}(A \bmod x^n).$$

After shifting by -1 , we are left with the question of evaluating a polynomial in $\mathbb{K}[x]_n$ at $\sum_{i < n} x^i / i!$. Writing its matrix shows that this map factors as $\Delta_n(\text{MultiEval}_n^t(\cdot), 1/i!)$, where MultiEval_n is the map

$$A \in \mathbb{K}[x]_n \mapsto (A(0), \dots, A(n-1)) \in \mathbb{K}^n.$$

To summarize, we have obtained that

$$\text{Exp}_{m,n}(A) = \Delta_n(\text{MultiEval}_n^t(\text{Shift}_{-1,n}(\text{mod}_{m,n}(A))), 1/i!).$$

Using fast transposed evaluation [13, 7], $\text{Exp}_{m,n}(A)$ can thus be computed in $O(M(n) \log(n))$ operations. Inverting these computations leads to the factorization

$$\text{Log}_{m,n}(A) = \text{Shift}_{1,n}(\text{Interp}_n^t(\Delta_n(\text{mod}_{m,n}(A), i!))),$$

where Interp_n is interpolation at $0, \dots, n-1$. Using algorithms for transpose interpolation [24, 7], this operation can be done in time $O(M(n) \log(n))$. $\square$

2.2 Associativity Rules

For each basic power series operation in Table 1, we now express $\text{Eval}_{m,n}(A, \circ(g))$ in terms of simpler operations; we call these descriptions *associativity rules*. We write them in a formal manner: this formalism is the key to automatically design complex composition algorithms, and makes it straightforward to obtain *transposed* associativity rules, required in the next section. Most of these rules are straightforward; care has to be taken regarding truncation, though.

Scaling, Shift and Powering.

$$\text{Eval}_{m,n}(A, M_\lambda(g)) = \text{Eval}_{m,n}(\text{Scale}_{\lambda,m}(A), g), \quad (A_1)$$

$$\text{Eval}_{m,n}(A, A_a(g)) = \text{Eval}_{m,n}(\text{Shift}_{a,m}(A), g), \quad (A_2)$$

$$\text{Eval}_{m,n}(A, P_k(g)) = \text{Eval}_{k(m-1)+1,n}(\text{Power}_{m,k}(A), g). \quad (A_3)$$

Inversion. From $A(1/g) = (\text{Rev}_m(A))(g)/g^{m-1}$ and writing $h = g^{1-m} \bmod x^n$, we get

$$\text{Eval}_{m,n}(A, \text{Inv}(g)) = \text{Mul}_{n,n}(\text{Eval}_{m,n}(\text{Rev}_m(A), g), h), \quad (A_4)$$

Root taking. For g and h in $\mathbb{K}[[x]]$, if $g = h^k$, one has $A(h) = A_{0/k}(g) + A_{1/k}(g)h + \dots + A_{k-1/k}(g)h^{k-1}$. We deduce the following rule, where the indices m_i are defined in Equation (1).

$$h_i = h^i \bmod x^n \text{ for } 0 \leq i < k$$

$$A_0, \dots, A_{k-1} = \text{Split}_{m,k}(A) \quad (A_5)$$

$$B_i = \text{Eval}_{m_i,n}(A_i, g) \text{ for } 0 \leq i < k$$

$$\text{Eval}_{m,n}(A, R_{k,\alpha,r}(g)) = \text{Comb}_n(B_0, \dots, B_{k-1}, 1, \dots, h_{k-1}).$$

Exponential and Logarithm.

$$\text{Eval}_{m,n}(A, E(g)) = \text{Eval}_n(\text{Exp}_{m,n}(A), g), \quad (A_6)$$

$$\text{Eval}_{m,n}(A, L(g)) = \text{Eval}_n(\text{Log}_{m,n}(A), g). \quad (A_7)$$

2.3 Composition sequences

We now describe more complex evaluations schemes, obtained by composing the former basic ones.

DEFINITION 1. Let $\mathbf{O}$ be the set of actions from Table 1. A sequence $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_L)$ with entries in $\mathbf{O}$ is defined at a series $g \in \mathbb{K}[[x]]$ if g is in $\text{dom}(\mathbf{o}_1)$, and for $i \leq L$, $\mathbf{o}_{i-1}(\dots \mathbf{o}_1(g))$ is in $\text{dom}(\mathbf{o}_i)$. It is a composition sequence if it is defined at x ; in this case, $\mathbf{o}$ computes the power series $g_1, \dots, g_L$, with $g_0 = x$ and $g_i = \mathbf{o}_i(g_{i-1})$; it outputs g_L .

Examples. As mentioned in [28], the rational series $g = (ax+b)/(cx+d) \in \mathbb{K}[[x]]$, with $cd \neq 0$, decomposes as

$$\frac{ax+b}{cx+d} = \frac{e}{cx+d} + f \text{ with } e = b - \frac{ad}{c} \text{ and } f = \frac{a}{c}.$$

This shows that g is output by the composition sequence $(M_c, A_d, \text{Inv}, M_e, A_f)$. A more complex example is

$$g = \frac{2x}{(1+x)^2} = \frac{1}{2} \left(1 - \left(1 - \frac{2}{1+x} \right)^2 \right),$$

which shows that g is output by the composition sequence

$$(A_1, \text{Inv}, M_{-2}, A_1, P_2, M_{-1}, A_1, M_{1/2}).$$

Finally, consider $g = \log((1+x)/(1-x))$. Using

$$g = \log \left(1 + \left(-2 - \frac{2}{x-1} \right) \right),$$

we get the composition sequence $(A_{-1}, \text{Inv}, M_{-2}, A_{-2}, L)$.

Computing the associated power series. Our main algorithm requires truncations of the series $g_1, \dots, g_L$ associated to a composition sequence. The next lemma discusses the cost of their computation. In all complexity estimates, *the composition sequence $\mathbf{o}$ is fixed*; hence, our estimates hide a dependency in $\mathbf{o}$ in their constant factors.

LEMMA 1. *If $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_L)$ is a composition sequence that computes power series $g_1, \dots, g_L$, one can compute all $g_i \bmod x^n$ in time $O(M(n))$.*

PROOF. All operators in $\mathbf{O}$ preserve the precision, except for root-taking, since the operator $R_{k,\alpha,r}$ loses $r(k-1)$ terms of precision. For $i \leq L$, define $\varepsilon_i = r(k-1)$ if $\mathbf{o}_i$ has the form $R_{k,\alpha,r}$, $\varepsilon_i = 0$ otherwise, and define $n_L = n$ and inductively $n_{i-1} = n_i + \varepsilon_i$. Starting the computations with $g_0 = x$, we iteratively compute $g_i \bmod x^{n_i}$ from $g_{i-1} \bmod x^{n_{i-1}}$.

Inspecting the list of possible cases, one sees that computing g_i always takes time $O(M(n_{i-1}))$. For powering, this estimate is valid because we disregard the dependency in $\mathbf{o}$: otherwise, terms of the form $\log(k)$ would appear. For the same reason, $O(M(n_{i-1}))$ is in $O(M(n))$, as is the total cost, obtained by summing over all i . $\square$

Composition using composition sequences. We now study the cost of computing the map $\text{Eval}_n(\cdot, g)$, assuming that $g \in \mathbb{K}[[x]]$ is output by a composition sequence $\mathbf{o}$. The cost depends on the operations in $\mathbf{o}$. To keep simple expressions, we distinguish two cases: if $\mathbf{o}$ contains no operation $\mathbf{E}$ or $\mathbf{L}$, we let $T_{\mathbf{o}}(n) = M(n)$; otherwise, $T_{\mathbf{o}}(n) = M(n) \log(n)$.

THEOREM 1 (COMPOSITION). *Let $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_L)$ be a composition sequence that outputs a series $g \in \mathbb{K}[[x]]$. Given $\mathbf{o}$, one can compute the map $\text{Eval}_n(\cdot, g)$ in time $O(T_{\mathbf{o}}(n))$.*

PROOF. We follow the algorithm of Figure 1. The main function first computes the sequence $G = g_1, \dots, g_L$ modulo x^n , using a subroutine $\text{ComputeG}(\mathbf{o}, n)$ that follows Lemma 1. The cost $O(M(n))$ of this operation is in $O(T_{\mathbf{o}}(n))$. Then, we call the auxiliary Eval_{aux} function.

On input $A, m, n, \ell, \mathbf{o}, G$, this latter function computes $\text{Eval}_{m,n}(A, g_\ell)$. This is done recursively, applying the appropriate associativity rule (A_1) to (A_7) ; the pseudo-code uses a C-like `switch` construct to find the matching case. Even if the initial polynomial A is in $\mathbb{K}[x]_n$, this may not be the case for the arguments passed to the next calls; hence the need for the extra parameter m . For root-taking, the subroutine FindDegrees computes the quantities m_i of Eq. (1).

Since we write the complexity as a function of n , the cost analysis is simple: even if several recursive calls are generated (k for k th root-taking), their total number is still $O(1)$.

```

Evalaux( $A, m, n, \ell, \mathbf{o}, G$ )
if  $\ell = 0$  return  $A \bmod x^n$ 
 $\ell' = \ell - 1$ 
switch( $\mathbf{o}_\ell$ )
case( $M_\lambda$ ):    $B = \text{Scale}_{\lambda,m}(A)$ 
              return Evalaux( $B, m, n, \ell', \mathbf{o}, G$ )
case( $A_a$ ):    $B = \text{Shift}_{a,m}(A)$ 
              return Evalaux( $B, m, n, \ell', \mathbf{o}, G$ )
case( $P_k$ ):    $B = \text{Power}_{m,k}(A)$ 
              return Evalaux( $B, km - k + 1, n, \ell', \mathbf{o}, G$ )
case( $\text{Inv}$ ):    $B = \text{Rev}_m(A)$ 
               $C = \text{Eval}_{\text{aux}}(B, m, n, \ell', \mathbf{o}, G)$ 
              return  $\text{Mul}_{n,n}(C, g_{\ell'}^{1-m} \bmod x^n)$ 
case( $R_{k,\alpha,r}$ ):  $m_0, \dots, m_{k-1} = \text{FindDegrees}(m, k)$ 
               $h_0 = 1$ 
              for  $i = 1, \dots, k-1$  do
                 $h_i = h h_{i-1} \bmod x^n$ 
               $A_0, \dots, A_{k-1} = \text{Split}_{m,k}(A)$ 
              for  $i = 0, \dots, k-1$  do
                 $B_i = \text{Eval}_{\text{aux}}(A_i, m_i, n, \ell', \mathbf{o}, G)$ 
              return  $\text{Comb}_n(B_0, \dots, B_{k-1}, h_0, \dots, h_{k-1})$ 
case( $\mathbf{E}$ ):      $B = \text{Exp}_{m,n}(A)$ 
              return Evalaux( $B, n, n, \ell', \mathbf{o}, G$ )
case( $\mathbf{L}$ ):      $B = \text{Log}_{m,n}(A)$ 
              return Evalaux( $B, n, n, \ell', \mathbf{o}, G$ )

```

```

Eval( $A, n, \mathbf{o}$ )
 $G = \text{ComputeG}(\mathbf{o}, n)$ 
return Evalaux( $A, n, n, L, \mathbf{o}, G$ )

```

Figure 1: Algorithm Eval.

Similarly, the degree of the argument A passed through the recursive calls may grow, but only like $O(n)$.

Two kinds of operations contribute to the cost: precomputations of $g_{\ell-1}^{1-m} \bmod x^n$ (for Inv) or of $1, g_\ell, \dots, g_\ell^{k-1} \bmod x^n$ for $R_{k,\alpha,r}$, and linear operations on A : shifting, scaling, multiplication ... The former take $O(M(n))$, since the exponents involved are in $O(n)$. The latter operations take $O(M(n))$ if no Exp or Log operation is performed, and $O(M(n) \log(n))$ otherwise. This concludes the proof. $\square$

2.4 Inverse map

The map $\text{Eval}_n(\cdot, g)$ is invertible if and only if $g'(0) \neq 0$ (hereafter, g' is the derivative of g). We discuss here the computation of the inverse map.

THEOREM 2 (INVERSE). *Let $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_L)$ be a composition sequence that outputs $g \in \mathbb{K}[[x]]$ with $g'(0) \neq 0$. One can compute the map $\text{Eval}_n^{-1}(\cdot, g)$ in time $O(T_{\mathbf{o}}(n))$.*

PROOF. If h is the power series $h = \sum_{i \geq i_0} h_i x^i$, with $h_{i_0} \neq 0$, $\text{val}(h) = i_0$ is the *valuation* of h , $\text{lc}(h) = h_{i_0}$ its *leading coefficient* and $\text{lt}(h) = h_{i_0} x^{i_0}$ its *leading term*. We also introduce an equivalence relation on power series: $g \sim h$ if $g(0) = h(0)$ and $\text{lt}(g - h(0)) = \text{lt}(h - h(0))$. The proof of the next lemma is immediate by case inspection.

LEMMA 2. *For $\mathbf{o}$ in $\mathbf{O}$, if $h \sim g$ and g is in $\text{dom}(\mathbf{o})$, then h is in $\text{dom}(\mathbf{o})$ and $\mathbf{o}(h) \sim \mathbf{o}(g)$.*

Series tangent to the identity. We prove the proposition in two steps. For series of the form $g = x \bmod x^2$, it suffices

to “reverse” step-by-step the computation sequence for g . The following lemma is crucial.

LEMMA 3. *Let g be in $\mathbb{K}[[x]]$, with $g = x \bmod x^2$, and let $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_L)$ be a sequence defined at g . Then $\mathbf{o}$ is a composition sequence.*

PROOF. We have to prove that $\mathbf{o}$ is defined at x , i.e., that all of $\mathbf{o}_1(x), \mathbf{o}_2(\mathbf{o}_1(x)), \dots$ are well-defined. This follows by applying the previous lemma inductively. $\square$

We can now work on the inversion property proper. Let thus $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_L)$ be a computation sequence, that computes $g_1, \dots, g_L$ and outputs $g = g_L$, with $g = x \bmod x^2$. We define operations $\tilde{\mathbf{o}}_1, \dots, \tilde{\mathbf{o}}_L$ through the following table (note that we reverse the order of the operations):

operation	$\mathbf{o}_i$	$\tilde{\mathbf{o}}_{L+1-i}$
Add	$\mathbf{A}_a$	$\mathbf{A}_{-a}$
Mul	$\mathbf{M}_\lambda$	$\mathbf{M}_{1/\lambda}$
Powering	$\mathbf{P}_k$	$\mathbf{R}_{k, \text{lc}(g_{i-1}), \text{val}(g_{i-1})}$
Root	$\mathbf{R}_{k, \alpha, r}$	$\mathbf{P}_k$
Inverse	Inv	Inv
Exp.	$\mathbf{E}$	$\mathbf{L}$
Log.	$\mathbf{L}$	$\mathbf{E}$

LEMMA 4. *The sequence $\tilde{\mathbf{o}} = (\tilde{\mathbf{o}}_1, \dots, \tilde{\mathbf{o}}_L)$ is a composition sequence and outputs a series $\tilde{g}$ such that $\tilde{g}(g) = x$.*

PROOF. One sees by induction that for all i , $\tilde{\mathbf{o}}_{i-1}(\dots \tilde{\mathbf{o}}_1(g))$ is in $\text{dom}(\tilde{\mathbf{o}}_i)$ and $\tilde{\mathbf{o}}_i(\dots \tilde{\mathbf{o}}_1(g)) = g_{L-i}$. This shows that the sequence $\tilde{\mathbf{o}}$ is defined at g and that $\tilde{\mathbf{o}}_L(\dots \tilde{\mathbf{o}}_1(g)) = x$. From Lemma 3, we deduce that $\tilde{\mathbf{o}}$ is defined at x . Letting $\tilde{g}$ be the output of $\tilde{\mathbf{o}}$, the previous equality gives $\tilde{g}(g) = x$, which concludes the proof. $\square$

Since $\mathbf{T}_{\mathbf{o}} = \mathbf{T}_{\tilde{\mathbf{o}}}$, and in view of Theorem 1, the next lemma concludes the proof of Theorem 2 in the current case.

LEMMA 5. *With g and $\tilde{g}$ as above, the map $\text{Eval}_n(\cdot, \tilde{g})$ is the inverse of $\text{Eval}_n(\cdot, g)$.*

PROOF. Let F be in $\mathbb{K}[x]_n$ and let $G = \text{Eval}_n(F, g)$, so that $F(g) = G + H$, with $\text{val}(H) \geq n$. Evaluating at $\tilde{g}$, we get $F = G(\tilde{g}) + H(\tilde{g}) = G(\tilde{g}) \bmod x^n$, since $\text{val}(\tilde{g}) = 1$. $\square$

General case. Lemma 5 fails when $\text{val}(g) = 0$. We can however reduce the general case to that where $\text{val}(g) = 1$. Let us write $g = g_0 + g_1x + \dots$, with $g_1 \neq 0$, and define $\tilde{g} = (g - g_0)/g_1$, so that $\tilde{g} = x \bmod x^2$. If $\mathbf{o}$ is a composition sequence for g , then $\tilde{\mathbf{o}} = (\mathbf{o}, \mathbf{A}_{-g_0}, \mathbf{M}_{1/g_1})$ is a composition sequence for $\tilde{g}$, and we have $\mathbf{T}_{\tilde{\mathbf{o}}} = \mathbf{T}_{\mathbf{o}}$. Thus, by the previous point, we can use this composition sequence to compute the map $\text{Eval}_n^{-1}(\cdot, \tilde{g})$ in time $O(\mathbf{T}_{\mathbf{o}}(n))$. From the equality

$$\text{Eval}_n(A, g) = \text{Eval}_n(\text{Scale}_{g_1, n}(\text{Shift}_{g_0, n}(A)), \tilde{g}),$$

we deduce

$$\text{Eval}_n^{-1}(A, g) = \text{Shift}_{-g_0, n}(\text{Scale}_{1/g_1, n}(\text{Eval}_n^{-1}(A, \tilde{g}))).$$

Since scaling and shifting induce only an extra $O(\mathbf{M}(n))$ arithmetic operations, this finishes the proof of Theorem 2.

3. CHANGE OF BASIS

This section applies our results on composition to *change of basis* algorithms, between the monomial basis (x^i) and various families of polynomials (P_i), with $\deg(P_i) = i$, for which we reach quasi-linear complexity. As an intermediate step, we present a bivariate evaluation algorithm.

3.1 Main Theorem

Let $\mathbf{F} \in \mathbb{K}[[x, t]]$ be the bivariate power series

$$\mathbf{F} = \sum_{i,j \geq 0} F_{i,j} x^i t^j = \sum_{j \geq 0} \xi_j(x) t^j.$$

Associated with $\mathbf{F}$, we consider the map

$$\text{Eval}_n(\cdot, \mathbf{F}, t) : (a_0, \dots, a_{n-1}) \mapsto \sum_{j < n} \xi_j(x) a_j \bmod x^n.$$

The matrix of this map is $[F_{i,j}]_{i,j < n}$. The following theorem shows that for a large class of series $\mathbf{F}$, the operation $\text{Eval}_n(\cdot, \mathbf{F}, t)$ and its inverse can be performed efficiently. The proof relies on a transposition argument, given in §3.3.

THEOREM 3 (MAIN THEOREM). *Let $f, g, h, u, v \in \mathbb{K}[[z]]$ be such that*

- g and h are given by composition sequences $\mathbf{o}_g$ and $\mathbf{o}_h$;
- f, u and v can be computed modulo z^n in time $\mathbf{T}(n)$;
- $g(0)h(0) = 0$ and $g'(0), h'(0), u(0), v(0)$ are non-zero;
- all coefficients of f are non-zero.

Then the series $\mathbf{F}(x, t) = u(x)v(t)f(g(x)h(t))$ is well-defined. Besides, one can compute the map $\text{Eval}_n(\cdot, \mathbf{F}, t)$ and its inverse in time $O(\mathbf{T}(n) + \mathbf{T}_{\mathbf{o}_g}(n) + \mathbf{T}_{\mathbf{o}_h}(n))$.

PROOF. Write $f = \sum_{k \geq 0} f_k z^k$,

$$g(x)^k = \sum_{i \geq 0} g_{k,i} x^i \quad \text{and} \quad h(t)^k = \sum_{j \geq 0} h_{k,j} t^j.$$

Since $g(0)h(0) = 0$, we have that either $h_{k,j} = 0$ for $k > j$, or $g_{k,i} = 0$ for $k > i$. Thus, the coefficient $F_{i,j}^*$ of $\mathbf{F}^*$ is well-defined and

$$F_{i,j}^* = \sum_{k \leq n} f_k g_{k,i} h_{k,j}.$$

These coefficients are those of a product of three matrices, the middle one being diagonal; we deduce the factorization

$$\text{Eval}_n(\cdot, \mathbf{F}^*, t) = \text{Eval}_n(\cdot, g) \circ \Delta_n(\cdot, f_i) \circ \text{Eval}_n^t(\cdot, h).$$

The assumptions on f, g and h further imply that the map $\text{Eval}_n(\cdot, \mathbf{F}^*, t)$ is invertible, of inverse

$$\text{Eval}_n^{-1}(\cdot, \mathbf{F}^*, t) = \text{Eval}_n^{-t}(\cdot, h) \circ \Delta_n(\cdot, f_i^{-1}) \circ \text{Eval}_n^{-1}(\cdot, g).$$

By Theorems 1 and 2, as well as Theorem 4 stated below, $\text{Eval}_n(\cdot, \mathbf{F}^*, t)$ and its inverse can thus be evaluated in time $O(\mathbf{T}(n) + \mathbf{T}_{\mathbf{o}_g}(n) + \mathbf{T}_{\mathbf{o}_h}(n))$. Now, from the identity $\mathbf{F} = u(x)v(t)\mathbf{F}^*$, we deduce that

$$\text{Eval}_n(\cdot, \mathbf{F}, t) = \text{Mul}_{n,n}(\cdot, u) \circ \text{Eval}_n(\cdot, \mathbf{F}^*, t) \circ \text{Mul}_{n,n}^t(\cdot, v).$$

Our assumptions on u and v make this map invertible, and

$$\text{Eval}_n^{-1}(\cdot, \mathbf{F}, t) = \text{Mul}_{n,n}^t(\cdot, b) \circ \text{Eval}_n^{-1}(\cdot, \mathbf{F}^*, t) \circ \text{Mul}_{n,n}(\cdot, a),$$

with $a(x) = 1/u \bmod x^n$ and $b(t) = 1/v \bmod t^n$. The extra costs induced by the computation of u, v , their inverses, and the truncated products fit in $O(\mathbf{T}(n) + \mathbf{M}(n))$. $\square$

3.2 Change of Basis

To conclude, we consider polynomials $(P_i)_{i \geq 0}$ in $\mathbb{K}[x]$, with $\deg(P_i) = i$, with generating series defined in terms of series u, v, f, g, h as in Theorem 3 by

$$\mathbf{P} = \sum_{i \geq 0} P_i(x) t^i = u(x)v(t)f(g(x)h(t)).$$

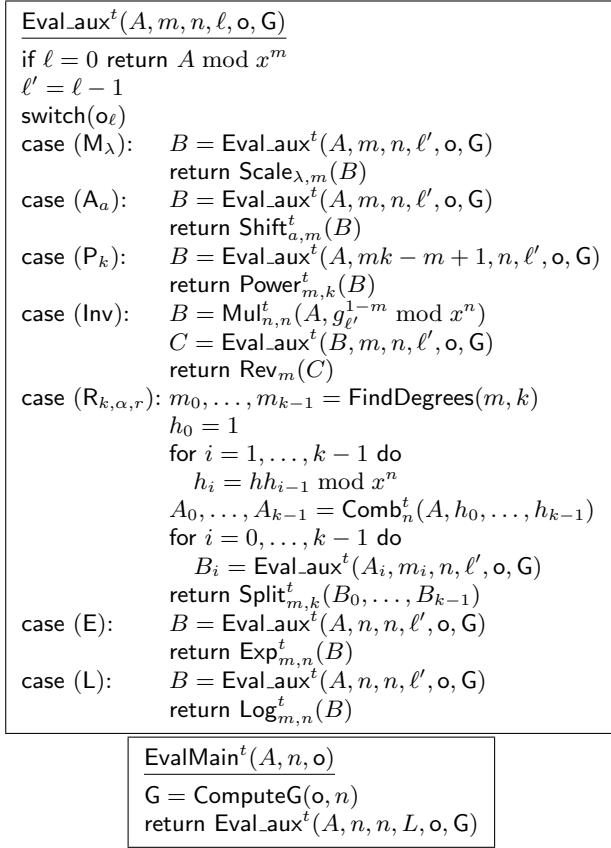


Figure 2: Algorithm Eval^t .

COROLLARY 1. *Under the above assumptions, one can perform the change of basis from $(x^i)_{i \geq 0}$ to $(P_i)_{i \geq 0}$, and conversely, in time $O(T(n) + T_{\mathbf{o}_g}(n) + T_{\mathbf{o}_h}(n))$.*

A surprisingly large amount of classical polynomials fits into this framework (see next section). An important special case is provided by Sheffer sequences [30, Chap. 2], whose exponential generating function has the form

$$\sum_{i \geq 0} \frac{P_i(x)}{i!} t^i = v(t) e^{xh(t)}.$$

Examples include the actuarial, Laguerre, Meixner and Poisson-Charlier polynomials, and the Bernoulli polynomials of the second kind (see Tables 3 and 4). In this case, if h is output by the composition sequence $\mathbf{o}$ and $v(t)$ can be computed modulo t^n in time $T(n)$, one can perform the change of basis from $(x^i)_{i \geq 0}$ to $(P_i)_{i \geq 0}$, and conversely, in time $O(T(n) + T_{\mathbf{o}}(n))$.

3.3 Transposed evaluation

The following completes the proof of Theorem 3.

THEOREM 4 (TRANSPPOSITION). *Let $\mathbf{o} = (\mathbf{o}_1, \dots, \mathbf{o}_L)$ be a composition sequence that outputs $g \in \mathbb{K}[[x]]$. Given $\mathbf{o}$, one can compute the map $\text{Eval}_n^t(\cdot, g)$ in time $O(T_{\mathbf{o}}(n))$.*

PROOF. This result follows directly from the transposition principle. However, we give an explicit construction of the transposed map $\text{Eval}_n^t(\cdot, g)$ in Figure 2. Non-linear precomputations are left unchanged. The terminal case $\ell = 0$

is dealt with by noting that the transpose of $\text{mod}_{m, n}$ is $\text{mod}_{n, m}$. To conclude, it suffices to give transposed associativity rules for our basic operators. The formal approach we use to write our algorithms pays off now, as it makes this transposition process automatic.

Recall that our algorithms deal with polynomials. The dual of $\mathbb{K}[x]_m$ can be identified with $\mathbb{K}[x]_m$ itself: to a $\mathbb{K}$ -linear form ℓ over $\mathbb{K}[x]_m$, one associates $\sum_{i < m} \ell(x^i) x^i$. Hence, transposed versions of algorithms acting on polynomials are seen to act on polynomials as well. Remark also that diagonal operators are their own transpose.

Multiplication. In [7], following [19], details of the transposed versions of plain, Karatsuba and FFT multiplications are given, with a cost matching that of the direct product. Without relying on such techniques, by writing down the multiplication matrix, one sees that $\text{Mul}_{n, m}^t(\cdot, P)$ is

$$A \in \mathbb{K}[x]_m \mapsto (A \text{Rev}_{d+1}(P) \bmod x^{n+d}) \text{div } x^d \in \mathbb{K}[x]_n,$$

if P has degree d . Using standard multiplication algorithms, this formulation leads to slower algorithms than those of [7]. However, in our usage cases, n , m and d are of the same order of magnitude, and only a constant factor is lost.

Scale. The operator $\text{Scale}_{\lambda, n}$ is diagonal; through transposition, the associativity rule becomes:

$$\text{Eval}_{m, n}^t(A, M_\lambda(g)) = \text{Scale}_{\lambda, m}(\text{Eval}_{m, n}^t(A, g)). \quad (A_1^t)$$

Shift. The transposed map Rev_n^t of the reversal operator coincides with Rev_n itself, since this operator is symmetric. By transposing the identity for Shift, we deduce

$$\text{Shift}_{a, n}^t(A) = \Delta_n(\text{Rev}_n(\text{Mul}_{n, n}^t(\text{Rev}_n(\Delta_n(A, 1/i!)), P)), i!).$$

This algorithm for the transpose operation, though not described as such, was already given in [18]. This yields:

$$\text{Eval}_{m, n}^t(A, A_a(g)) = \text{Shift}_{a, m}^t(\text{Eval}_{m, n}^t(A, g)). \quad (A_2^t)$$

Powering. The dual map $\text{Power}_{n, k}^t$ maps $A \in \mathbb{K}[x]_{k(n-1)+1}$ to $A_{0/k} \in \mathbb{K}[x]_n$ (with the notation of §2.1). We deduce:

$$\text{Eval}_{m, n}^t(A, P_k(g)) = \text{Power}_{m, k}^t(\text{Eval}_{k(m-1)+1, n}^t(A, g)). \quad (A_3^t)$$

Inversion. The transposed version of the rule for Inv is

$$\text{Eval}_{m, n}^t(A, \text{Inv}(g)) = \text{Rev}_m(\text{Eval}_{m, n}^t(\text{Mul}_{n, n}^t(A, g^{1-m}), g)). \quad (A_4^t)$$

Root taking. Considering its matrix, one sees that $\text{Split}_{m, k}^t$ maps $(A_0, \dots, A_{k-1}) \in \mathbb{K}[x]_{m_0} \times \dots \times \mathbb{K}[x]_{m_{k-1}}$ to

$$A_0(x^k) + A_1(x^k)x + \dots + A_{k-1}(x^k)x^{k-1} \in \mathbb{K}[x]_m.$$

Besides, since the map **Comb** is the direct sum of the maps

$$\text{Mul}_{n, n}(\cdot, G_i) : \mathbb{K}[x]_n \rightarrow \mathbb{K}[x]_n,$$

its transpose $\text{Comb}_n^t(\cdot, G_0, \dots, G_{k-1})$ sends $A \in \mathbb{K}[x]_n$ to

$$(\text{Mul}_{n, n}^t(A, G_i))_{0 \leq i \leq k-1} \in \mathbb{K}[x]_n^k.$$

Putting this together gives the transposed associativity rule

$$\begin{aligned} h_i &= h^i \bmod x^n \text{ for } 0 \leq i < k \\ A_0, \dots, A_{k-1} &= \text{Comb}_n^t(A, h_0, \dots, h_{k-1}) \\ B_i &= \text{Eval}_{m_i, n}^t(A_i, g) \text{ for } 0 \leq i < k \\ \text{Eval}_{m, n}^t(A, R_{k, \alpha, r}(g)) &= \text{Split}_{m, k}^t(B_0, \dots, B_{k-1}) \end{aligned} \quad (A_5^t)$$

Exponential and Logarithm. From the proof of Proposition 1, we deduce the transposed map of $\text{Exp}_{m,n}$, $\text{Log}_{m,n}$ and their associativity rules

$$\begin{aligned}\text{Exp}_{m,n}^t(A) &= \text{mod}_{n,m}(\text{Shift}_{-1,n}^t(\text{MultiEval}(\Delta_n(A, 1/i!))), \\ \text{Eval}_{m,n}^t(A, E(g)) &= \text{Exp}_{m,n}^t(\text{Eval}_{n,n}^t(A, g)); \quad (A_6^t) \\ \text{Log}_{m,n}^t(A) &= \text{mod}_{n,m}(\Delta_n(\text{Interp}_n(\text{Shift}_{1,n}^t(A)), i!)), \\ \text{Eval}_{m,n}^t(A, L(g)) &= \text{Log}_{m,n}^t(\text{Eval}_{n,n}^t(A, g)). \quad (A_7^t)\end{aligned}$$

4. APPLICATIONS

Many generating functions of classical families of polynomials fit into our framework. To obtain conversion algorithms, it is sufficient to find suitable composition sequences. Table 3 lists families of polynomials for which conversions can be done in time $O(M(n))$ with our method (see e.g. [30, 3] for more on these classical families). In Table 4, a similar list is given, leading to conversions of cost $O(M(n) \log n)$; most of these entries are actually Sheffer sequences. Many other families can be obtained as special cases (e.g., Gegenbauer, Legendre, Chebyshev, Mittag-Leffler, etc).

The entry marked by $(\star)$ is from [18]; the entries marked by $(\star\star)$ are orthogonal polynomials, for which one conversion (from the orthogonal to the monomial basis) is already mentioned with the same complexity in [15, 29].

In all cases, the pre-multiplier $u(x)v(t)$ depends on t only and can be computed at precision n in time $O(M(n))$; all our functions f can be expanded at precision n in time $O(n)$. Regarding the functions $g(x)$ and $h(t)$, most entries are easy to check; the only explanations needed concern some series $h(t)$. Rational functions are covered by the first example of §2.3; the second example of §2.3 deals with Jacobi polynomials and Spread polynomials; the last example of §2.3 shows how to handle functions with logarithms. For Fibonacci polynomials, the function $h(t) = t/(1 - t^2)$ satisfies

$$(2h)^2 = \left(\frac{1+t^2}{1-t^2}\right)^2 - 1.$$

From this, we deduce the sequence for h :

$$(P_2, M_{-1}, A_1, \text{Inv}, M_2, A_{-1}, P_2, A_{-1}, R_{2,2,1}, M_{1/2}).$$

For Mott polynomials the series $h(t) = (1 - \sqrt{1 - t^2})/t$ can be rewritten as

$$h = \sqrt{\frac{2}{1 + \sqrt{1 - t^2}}} - 1.$$

This yields the composition sequence

$$(P_2, M_{-1}, A_1, R_{2,1,0}, A_1, \text{Inv}, M_2, A_{-1}, R_{2,1,0}).$$

5. EXPERIMENTS

We implemented the algorithms for change of basis using NTL [32]; the experiments are done for coefficients defined modulo a 40 bit prime, using the `ZZ_p` NTL class (our algorithms still work for degrees small with respect to the characteristic). All timings reported here are obtained on a Pentium M, 1.73 Ghz, with 1 GB memory.

Our implementation follows directly the presentation of the former sections. We use the transposed multiplication implementation of [7]. The Newton iteration for inverse is built-in in NTL; we use the standard Newton iteration for

square root [10]. Exponentials are computed using the algorithm of [20]. Powers are computed through exponential and logarithm [10], except when the arguments are binomials, when faster formulas for binomial series are used. For evaluation and interpolation at $0, \dots, n-1$, and their transposes, we use the implementation of [7].

We use the Jacobi and Mittag-Leffler orthogonal polynomials (a special case of Meixner polynomials, with $\beta = 0$ and $c = -1$), with the composition sequences of §2.3. Our algorithm has cost $O(M(n))$ for the former and $O(M(n) \log(n))$ for the latter. We compare this to the naive approach of quadratic cost in Figure 3 and 4, respectively. Timings are given for the conversion from orthogonal to monomial bases; those for the inverse conversion are similar.

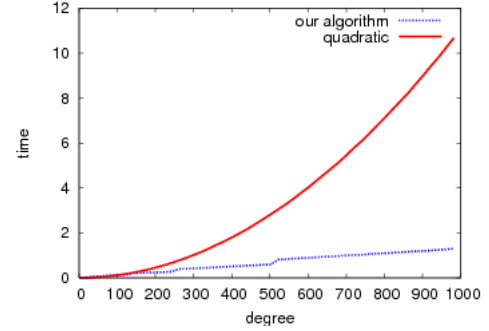


Figure 3: Jacobi polynomials.

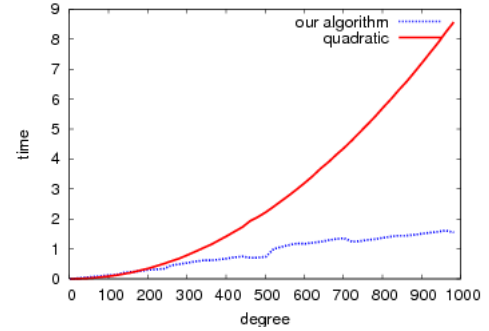


Figure 4: Mittag-Leffler polynomials.

Our algorithm performs better than the quadratic one. The crossover points lie between 100 and 200; this large value is due to the constant hidden in our big-Oh estimates: in both cases, there is a contribution of about $20M(n)$, plus an additional $M(n) \log(n)$ for Mittag-Leffler.

6. DISCUSSION

This article provides a flexible framework for generating new families of conversion algorithms: it suffices to add new composition operators to Table 1 and provide the corresponding associativity rules. Still, several questions need further investigation. Several of the composition sequences we use are non-trivial: this raises in particular the questions of characterizing what functions can be computed by a composition sequence, and of determining such sequences algorithmically. Besides, the costs of our algorithms are measured only in terms of arithmetic operations; the questions of numerical stability (for floating-point computations) or of coefficient size (when working over $\mathbb{Q}$) require further work.

Acknowledgments. We thank ANR Gecko, the joint Inria-Microsoft Research Lab and NSERC for financial support.

polynomial	generating series	$u(x)v(t)$	$f(z)$	$g(x)$	$h(t)$
Laguerre L_n^α	$\sum_{n \geq 0} L_n^\alpha(x) t^n$	$(1-t)^{-1-\alpha}$	$\exp(z)$	$-x$	$t(1-t)^{-1}$
Hermite H_n	$\sum_{n \geq 0} \frac{1}{n!} H_n(x) t^n$	$\exp(-t^2)$	$\exp(z)$	$2x$	t
Jacobi $P_n^{(\alpha, \beta)}$	$\sum_{n \geq 0} \frac{(\alpha+\beta+1)_n}{(\beta+1)_n} P_n^{(\alpha, \beta)}(x) t^n$	$(1+t)^{-\alpha-\beta-1}$	${}_2F_1(\frac{\alpha+\beta+1}{2}, \frac{\alpha+\beta+2}{2}; \beta+1; z)$	$1+x$	$2t(1+t)^{-2}$
Fibonacci F_n	$\sum_{n \geq 0} F_n(x) t^n$	$(1-t^2)^{-1}$	$(1-z)^{-1}$	x	$t(1-t^2)^{-1}$
Euler E_n^α	$\sum_{n \geq 0} \frac{1}{n!} E_n^\alpha(x) t^n$	$2^\alpha(e^t+1)^{-\alpha}$	$\exp(z)$	x	t
Bernoulli B_n^α	$\sum_{n \geq 0} \frac{1}{n!} B_n^\alpha(x) t^n$	$t^\alpha(e^t-1)^{-\alpha}$	$\exp(z)$	x	t
Mott M_n	$\sum_{n \geq 0} \frac{1}{n!} M_n(x) t^n$	1	$\exp(z)$	$-x$	$(1-\sqrt{1-t^2})/t$
Spread S_n	$\sum_{n \geq 0} S_n(x) t^n$	$(1+t)(1-t)^{-1}$	$z(1+4z)^{-1}$	x	$t(1-t)^{-2}$
Bessel p_n	$\sum_{n \geq 0} \frac{1}{n!} p_n(x) t^n$	1	$\exp(z)$	x	$1-\sqrt{1-2t}$

Table 3: Polynomials with conversion in $O(M(n))$

polynomial		generating series	$u(x)v(t)$	$f(z)$	$g(x)$	$h(t)$
Falling factorial $(x)_n$	(*)	$\sum_{n \geq 0} \frac{1}{n!} (x)_n t^n$	1	$\exp(z)$	x	$\log(1+t)$
Bell ϕ_n		$\sum_{n \geq 0} \frac{1}{n!} \phi_n(x) t^n$	1	$\exp(z)$	x	$\exp(t) - 1$
Bernoulli, 2nd kind b_n		$\sum_{n \geq 0} \frac{1}{n!} b_n(x) t^n$	$t/\log(1+t)$	$\exp(z)$	x	$\log(1+t)$
Poisson-Charlier $c_n(x; a)$		$\sum_{n \geq 0} \frac{1}{n!} c_n(x; a) t^n$	$\exp(-t)$	$\exp(z)$	x	$\log(1+t/a)$
Actuarial $a_n^{(\beta)}$		$\sum_{n \geq 0} \frac{1}{n!} a_n^{(\beta)}(x) t^n$	$\exp(\beta t)$	$\exp(z)$	$-x$	$\exp(t) - 1$
Narumi $N_n^{(a)}$		$\sum_{n \geq 0} \frac{1}{n!} N_n^{(a)}(x) t^n$	$t^a \log(1+t)^{-a}$	$\exp(z)$	x	$\log(1+t)$
Peters $P_n^{(\lambda, \mu)}$		$\sum_{n \geq 0} \frac{1}{n!} P_n^{(\lambda, \mu)}(x) t^n$	$(1+(1+t)^\lambda)^{-\mu}$	$\exp(z)$	x	$\log(1+t)$
Meixner-Pollaczek $P_n^{(\lambda)}(x; \phi)$	(**)	$\sum_{n \geq 0} P_n^{(\lambda)}(x; \phi) t^n$	$(1+t^2-2t \cos \phi)^{-\lambda}$	$\exp(z)$	ix	$\log(\frac{1-te^{i\phi}}{1-te^{-i\phi}})$
Meixner $m_n(x; \beta, c)$	(**)	$\sum_{n \geq 0} \frac{(\beta)_n}{n!} m_n(x; \beta, c) t^n$	$(1-t)^{-\beta}$	$\exp(z)$	x	$\log(\frac{1-t/c}{1-t})$
Krawtchouk $K_n(x; p, N)$	(**)	$\sum_{n \geq 0} \binom{N}{n} K_n(x; p, N) t^n$	$(1+t)^N$	$\exp(z)$	x	$\log(\frac{p-(1-p)t}{p(1+t)})$

Table 4: Polynomials with conversion in $O(M(n) \log(n))$

7. REFERENCES

- [1] A. V. Aho, K. Steiglitz, and J. D. Ullman. Evaluating polynomials at fixed sets of points. *SIAM J. Comp.*, 4(4):533–539, 1975.
- [2] B. K. Alpert and V. Rokhlin. A fast algorithm for the evaluation of Legendre expansions. *SIAM J. Sci. Statist. Comp.*, 12(1):158–179, 1991.
- [3] G. Andrews, R. Askey, and R. Roy. *Special functions*. Cambridge University Press, 1999.
- [4] R. Barrio and J. Peña. Basis conversions among univariate polynomial representations. *C. R. Math. Acad. Sci. Paris*, 339(4):293–298, 2004.
- [5] D. J. Bernstein. Composing power series over a finite ring in essentially linear time. *J. Symb. Comp.*, 26(3):339–341, 1998.
- [6] D. Bini and V. Y. Pan. *Polynomial and matrix computations. Vol. 1*. Birkhäuser Boston Inc., 1994.
- [7] A. Bostan, G. Lecerf, and É. Schost. Tellegen’s principle into practice. In *ISSAC’03*, pages 37–44. ACM, 2003.
- [8] A. Bostan, B. Salvy, and É. Schost. Fast algorithms for orthogonal polynomials. In preparation.
- [9] A. Bostan and É. Schost. Polynomial evaluation and interpolation on special sets of points. *J. Complexity*, 21(4):420–446, 2005.
- [10] R. P. Brent. Multiple-precision zero-finding methods and the complexity of elementary function evaluation. In *Analytic Computational Complexity*, pages 151–176. Acad. Press, 1975.
- [11] R. P. Brent and H. T. Kung. Fast algorithms for manipulating formal power series. *J. ACM*, 25(4):581–595, 1978.
- [12] P. Bürgisser, M. Clausen, and A. Shokrollahi. *Algebraic complexity theory*, volume 315 of *GMW*. Springer-Verlag, 1997.
- [13] J. Canny, E. Kaltofen, and Y. Lakshman. Solving systems of non-linear polynomial equations faster. In *ISSAC’89*, pages 121–128. ACM, 1989.
- [14] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Inform.*, 28(7):693–701, 1991.
- [15] J. R. Driscoll, J. D. M. Healy, and D. N. Rockmore. Fast discrete polynomial transforms with applications to data analysis for distance transitive graphs. *SIAM J. Comp.*, 26(4):1066–1099, 1997.
- [16] M. Frumkin. A fast algorithm for expansion over spherical harmonics. *Appl. Algebra Engrg. Comm. Comp.*, 6(6):333–343, 1995.
- [17] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, 1999.
- [18] J. Gerhard. Modular algorithms for polynomial basis conversion and greatest factorial factorization. In *RWCA’00*, pages 125–141, 2000.
- [19] G. Hanrot, M. Quercia, and P. Zimmermann. The Middle Product Algorithm, I. *Appl. Algebra Engrg. Comm. Comp.*, 14(6):415–438, 2004.
- [20] G. Hanrot and P. Zimmermann. Newton iteration revisited. <http://www.loria.fr/~zimmerma/papers>, 2002.
- [21] G. Heinig. Fast and superfast algorithms for Hankel-like matrices related to orthogonal polynomials. In *NAA’00*, volume 1988 of *LNCS*, pages 361–380. Springer-Verlag, 2001.
- [22] J. van der Hoeven. Relax, but don’t be too lazy. *J. Symb. Comput.*, 34(6):479–542, 2002.
- [23] E. Kaltofen. Challenges of symbolic computation: my favorite open problems. *J. Symb. Comp.*, 29(6):891–919, 2000.
- [24] E. Kaltofen and Y. Lakshman. Improved sparse multivariate polynomial interpolation algorithms. In *ISSAC’88*, volume 358 of *LNCS*, pages 467–474. Springer Verlag, 1989.
- [25] J. Keiner. Computing with expansions in Gegenbauer polynomials. Preprint AMR07/10, U. New South Wales, 2007.
- [26] G. Leibon, D. Rockmore, and G. Chirikjian. A fast Hermite transform with applications to protein structure determination. In *SNC’07*, pages 117–124, New York, NY, USA, 2007. ACM.
- [27] Y.-M. Li and X.-Y. Zhang. Basis conversion among Bézier, Tchebyshev and Legendre. *Comput. Aided Geom. Design*, 15(6):637–642, 1998.
- [28] V. Y. Pan. New fast algorithms for polynomial interpolation and evaluation on the Chebyshev node set. *Computers and Mathematics with Applications*, 35(3):125–129, 1998.
- [29] D. Potts, G. Steidl, and M. Tasche. Fast algorithms for discrete polynomial transforms. *Math. Comp.*, 67(224):1577–1590, 1998.
- [30] S. Roman. *The umbral calculus*. Dover publications, 2005.
- [31] A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7:281–292, 1971.
- [32] V. Shoup. A new polynomial factorization algorithm and its implementation. *J. Symb. Comp.*, 20(4):363–397, 1995.

Loewy Decomposition of Third-Order Linear PDE's in the Plane

D. Grigoriev
CNRS, IRMAR,
Université de Rennes, Beaulieu,
35042, Rennes, France
dmitry.grigoryev@univ-rennes1.fr

F. Schwarz
Fraunhofer Gesellschaft
Institut SCAI
53754 Sankt Augustin
Germany
Fritz.Schwarz@scai.fhg.de

ABSTRACT

Loewy's decomposition of a linear ordinary differential operator as the product of largest completely reducible components is generalized to partial differential operators of order three in two variables. This is made possible by considering the problem in the ring of partial differential operators where both left intersections and right divisors of left ideals are not necessarily principal. Listings of possible decomposition types are given. Many of them are illustrated by worked out examples. Algorithmic questions and questions of uniqueness are discussed in the Summary.

Categories and Subject Descriptors

I.1 [Symbolic and Algebraic Manipulation]:
Applications

General Terms

Algorithms

Keywords

D-module, Loewy decomposition, Janet basis

1. INTRODUCTION

About one hundred years ago Loewy proved the fundamental result that any ordinary differential operator may be represented uniquely as the product of largest completely reducible operators, i.e. operators that are the left intersection of irreducible operators of lower order [10]; see also Chapter 2 of the book [12]. This decomposition provides a detailed understanding of the structure of the solution space of the corresponding differential equation. Therefore it would be highly desirable to generalize it to partial differential operators as well. Amazingly this has never really been done since Loewy's original work. In this article Loewy decompositions of third-order operators in two variables are considered in detail; the possible limitations are also discussed.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC '08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

In the subsequent section some basic notations from differential algebra are introduced; details may be found in the book by Kolchin [9] or the articles by Buium and Cassidy [2] or Quadrat [11]. The main part of the article is organized according to leading derivatives of the respective operators.

2. BASIC DIFFERENTIAL ALGEBRA

In this section some basic terminology from differential algebra is introduced. Rings of differential operators $\mathcal{D} \equiv \mathcal{F}[\partial_x, \partial_y]$ or $\mathcal{D} \equiv \mathbb{Q}(x, y)[\partial_x, \partial_y]$ are considered; $\mathcal{F}$ is a universal differential field. $\mathcal{F}$ or $\mathbb{Q}(x, y)$ are called the *base field*. Let the left ideal I , or simply ideal I , be generated by elements $l_i \in \mathcal{D}$, $i = 1, \dots, p$. Then one writes $I = \langle l_1, \dots, l_p \rangle$. As a rule, the l_i are assumed to form a Janet basis in the term order *glex* with $x > y$. If z is some differential indeterminate, $l_i z = 0$, $i = 1, \dots, p$, is the corresponding system of pde's. If only the leading terms of the generators of an ideal are of interest the non-leading terms are omitted; it is denoted by $\langle \dots \rangle_{LT}$. Let $I \subseteq \mathcal{F}[\partial_x, \partial_y]$ be an ideal and H_I its Hilbert-Kolchin polynomial ([9], page 130; [2], page 602). The degree $\deg(H_I)$ of H_I is called the *differential type* of I . Its leading coefficient $lc(H_I)$ is called the *typical differential dimension* of I . The pair $(\deg(H_I), lc(H_I))$ has been baptized the *gauge* of the ideal I [7].

Let I and J be two ideals. Important constructions are the greatest common right divisor $Gcrd(I, J)$ or the sum ideal; and the least common left multiple $Lclm(I, J)$ or the left intersection. In [7] it has been shown how they are computed algorithmically. Finally the *relative syzygy module* $Syz(I, J)$ of I and $J = \langle g_1, \dots, g_q \rangle$ is generated by

$$\{h \equiv (h_1, \dots, h_q) \in \mathcal{D}^q \mid h_1 g_1 + \dots + h_q g_q \in I\}.$$

Define two ordinary differential operators by

$$\begin{aligned} D_x^m &\equiv d_x^m + a_1 d_x^{m-1} + \dots + a_{m-1} d_x + a_m, \\ D_y^n &\equiv d_y^n + b_1 d_y^{n-1} + \dots + b_{n-1} d_y + b_n; \end{aligned}$$

m and n are natural numbers not less than 2. Several ideals generated by an operator of order three and one of these operators will occur later in this article. A special notation is introduced for them as shown in the table below. These ideals have an important meaning as divisors in the decompositions to be discussed in later chapters. Due to its close relation to the iteration scheme introduced by Laplace, see [6], vol. II, Chapter V, it is suggested to call them *Laplace divisors*. The pair of upper indices of the ideals in Table 1 denotes the gauge of the respective ideal.

Notation	Leading term ideal
$\mathcal{J}_4^{(1,1)}(m)$	$\langle \partial_{xyy}, \partial_x^m \rangle_{LT}$
$\mathcal{J}_5^{(1,1)}(n)$	$\langle \partial_{xxy}, \partial_y^n \rangle_{LT}$
$\mathcal{J}_3^{(1,2)}(m)$	$\langle \partial_{xxy}, \partial_x^m \rangle_{LT}$
$\mathcal{J}_4^{(1,2)}(n)$	$\langle \partial_{xyy}, \partial_y^n \rangle_{LT}$
$\mathcal{J}_5^{(1,2)}$	$\langle \partial_{xxx}, \partial_{xxy} \rangle_{LT}$
$\mathcal{J}_6^{(1,2)}$	$\langle \partial_{xxy}, \partial_{xyy} \rangle_{LT}$

Table 1. Some ideals of gauge (1,1) and (1,2) and order of derivatives not higher than 3. The ideals involving the parameter m or n will occur later on as Laplace divisors.

It turns out that the intersection ideals generated by two first-order operators are of fundamental importance for understanding the different decomposition types discussed later in this article. They are described in the following two theorems. The first theorem has been proved in [7].

THEOREM 2.1. *Let the ideals $I_i = \langle \partial_x + a_i \partial_y + b_i \rangle$ for $i = 1, 2$ with $I_1 \neq I_2$ be given. Both ideals have gauge (1,1). There are three different cases for their intersection $I_1 \cap I_2$, all are of gauge (1,2).*

i) *Separable case $a_1 \neq a_2$. If $\left(\frac{b_1 - b_2}{a_1 - a_2} \right)_x = \left(\frac{a_1 b_2 - a_2 b_1}{a_1 - a_2} \right)_y$ there holds $I_1 \cap I_2 = \langle \partial_{xx} \rangle_{LT}$ and*

$$I_1 + I_2 = \left\langle \partial_x + \frac{a_1 b_2 - a_2 b_1}{a_1 - a_2}, \partial_y + \frac{b_1 - b_2}{a_1 - a_2} \right\rangle.$$

ii) *Double root $a_1 = a_2 = a$, $b_1 \neq b_2$. There holds $I_1 \cap I_2 = \langle \partial_{xx} \rangle_{LT}$ and $I_1 + I_2 = \langle 1 \rangle$.*

iii) *If the preceding two cases do not apply there holds $I_1 \cap I_2 = \langle \partial_{xxx}, \partial_{xxy} \rangle_{LT}$ and $I_1 + I_2 = \langle 1 \rangle$.*

If a decomposition comprises an operator with leading derivative ∂_y , the following result is required.

THEOREM 2.2. *Let the ideals $I_1 = \langle \partial_x + a_1 \partial_y + b_1 \rangle$ and $I_2 = \langle \partial_y + b_2 \rangle$ be given. There are two different cases for their intersection $I_1 \cap I_2$.*

i) *If $(b_1 - a_1 b_2)_y = b_{2,x}$ there holds $I_1 \cap I_2 = \langle \partial_{xy} \rangle_{LT}$ and $I_1 + I_2 = \langle \partial_x + b_1 - a_1 b_2, \partial_y + b_2 \rangle$.*

ii) *If the preceding case does not apply there holds $I_1 \cap I_2 = \langle \partial_{xxy}, \partial_{xyy} \rangle_{LT}$ and $I_1 + I_2 = \langle 1 \rangle$.*

The proof involves a slight modification of the proof of Theorem 2.1, it involves essentially Janet basis calculations. The special case that n first-order equations originate from the factorization of an operator of order n is treated next.

LEMMA 2.3. *Let L be a partial differential operator in x and y with leading term ∂_x^n , and let $l_i \equiv \partial_x + a_i \partial_y + b_i$, $i = 1, \dots, n$, $a_i \neq a_j$ for $i \neq j$, be n right divisors of L . Then the intersection ideal generated by the l_i is principal and is generated by L .*

Factoring an ordinary differential operator involves solving Riccati equations $y' + ay^2 + by + c = 0$, factoring pde's in the plane may require to solve equations of the form

$$z_x + az_y + bz^2 + cz + d = 0;$$

a, b, c, d are from the base field. They are called *partial Riccati equations* and have been discussed in detail in [7].

3. PDE'S WITH LEADING DERIVATIVE ∂_{xxx}

This case is also interesting for historical reasons because it was the first third order partial differential operator for which factorizations were considered [1]. Especially the operator discussed in Blumberg's thesis, see Example 3.7 below, attracted a lot of interest.

PROPOSITION 3.1. *Let the third order linear partial differential operator*

$$D_{xxx} \equiv \partial_{xxx} + A_1 \partial_{xxy} + A_2 \partial_{xyy} + A_3 \partial_{yyy} + A_4 \partial_{xx} + A_5 \partial_{xy} + A_6 \partial_{yy} + A_7 \partial_x + A_8 \partial_y + A_9 \quad (1)$$

be given with $A_k \in \mathbb{Q}(x, y)$ for all k . Its first order right factors $\partial_x + a \partial_y + b$ are essentially determined by the roots a_1, a_2 and a_3 of $a^3 - A_1 a^2 + A_2 a - A_3 = 0$. The following alternatives may occur.

i) *If a_1, a_2 and a_3 are three pairwise different rational solutions and the corresponding b_i are determined by (8), each pair a_i, b_i satisfying (9) and (10) yields a factor $l_i = \partial_x + a_i \partial_y + b_i$. If there are three factors, the operator is completely reducible and there holds $D_{xxx} = \text{Lclm}(l_1, l_2, l_3)$. If there are two factors, their intersection may not be principal. If only a single pair satisfies conditions (9) and (10), or there is only a single rational solution which satisfies them, there is only a single factor.*

ii) *If $a_1 = a_2$ is a twofold rational solution that does not satisfy (8), and $a_3 \neq a_1$ is a simple rational solution, a single right factor $\partial_x + a_3 \partial_y + b_3$ exists if a_3 , and b_3 as determined by (8) satisfy (9) and (10).*

iii) *If $a_1 = a_2$ is a twofold rational solution satisfying (8), $a_3 \neq a_1$ is a simple rational solution, and a_3 and b_3 as determined by (8) do not satisfy (9) and (10), the following subcases may occur.*

a) *There is a right factor of the form $l(\Phi) \equiv \partial_x + a_1 \partial_y + R(x, y, \Phi(\phi))$ whereas $R(x, y, \Phi(\phi))$ is the general solution of (9) satisfying (10); $\phi(x, y)$ is the first integral of $\frac{dy}{dx} = a_1(x, y)$, Φ is an undetermined function; $l(\Phi)$ leads to a second order right factor $\text{Lclm}(l_1(\Phi_1), l_2(\Phi_2))$ with $\Phi_1 \neq \Phi_2$ specializations of Φ .*

b) *There are two right factors $l_i \equiv \partial_x + a_1 \partial_y + r_i(x, y)$, $i = 1, 2$, if (9) has rational solutions $r_i(x, y)$ satisfying (10), or if it has a general rational solution as in case a), but constraint (10) singles out the special solutions $r_1(x, y)$ and $r_2(x, y)$; l_1 and l_2 generate a second order factor $\text{Lclm}(l_1, l_2)$.*

c) *There is a single right factor $l \equiv \partial_x + a_1 \partial_y + r(x, y)$ if (9) has the single rational solution $r(x, y)$ satisfying (10), or if it has two rational solutions or a general rational solution, but (10) singles out the single solution $r(x, y)$.*

iv) *If $a_1 = a_2$ is a twofold rational solution satisfying (8), $a_3 \neq a_1$ is a simple rational solution, and a_3 and b_3 as determined by (8) satisfy (9) and (10), the same distinctions occur as in the preceding case iii), supplemented by an additional factor $\partial_x + a_3 \partial_y + b_3$.*

v) If $a_1 = a_2 = a_3 = \frac{1}{3}A_1$ is a threefold rational solution, there holds

$$A_1^2 A_4 - 3A_1 A_5 + 9A_6 = 0. \quad (2)$$

The following subcases may occur.

a) If the coefficient of b in

$$\begin{aligned} (A_{1,x} + \frac{1}{3}A_1 A_{1,y} + \frac{2}{3}A_1 A_4 - A_5)b = \\ \frac{1}{3}A_{1,xx} + \frac{2}{9}A_1 A_{1,xy} + \frac{2}{27}A_1^2 A_{1,yy} - \frac{2}{9}A_{1,x} A_{1,y} \\ + \frac{1}{3}A_4 A_{1,x} - \frac{1}{27}A_1 A_{1,y}^2 - \frac{1}{9}A_1 A_4 A_{1,y} \\ + \frac{1}{3}A_5 A_{1,y} + \frac{1}{3}A_1 A_7 - A_8 = 0 \end{aligned} \quad (3)$$

does not vanish, b may be determined uniquely from it. A factor exists if the constraint

$$\begin{aligned} b_{xx} + \frac{2}{3}A_1 b_{xy} + \frac{1}{9}A_1^2 b_{yy} - 3bb_x + A_4 b_x - A_1 b b_y \\ - (\frac{2}{3}A_{1,x} + \frac{2}{9}A_1 A_{1,y} + \frac{1}{3}A_1 A_4 - A_5)b_y \\ + b^3 - A_4 b^2 + A_7 b - A_9 = 0. \end{aligned} \quad (4)$$

is satisfied.

b) If the coefficient of b in (3) vanishes, there holds

$$\begin{aligned} A_{1,x} + \frac{1}{3}A_1 A_{1,y} + \frac{2}{3}A_1 A_4 - A_5 = 0, \\ A_{1,xx} + \frac{2}{3}A_1 A_{1,xy} + \frac{2}{9}A_1^2 A_{1,yy} + A_4 A_{1,x} + \frac{1}{3}A_{1,x} A_{1,y} \\ + \frac{2}{9}A_1 A_{1,y}^2 + \frac{1}{3}A_1 A_4 A_{1,y} + \frac{1}{3}A_1 A_7 - 3A_8 = 0 \end{aligned} \quad (5)$$

and b is determined by

$$\begin{aligned} b_{xx} + \frac{2}{3}A_1 b_{xy} + \frac{1}{9}A_1^2 b_{yy} - 3bb_x + A_4 b_x - A_1 b b_y \\ + \frac{1}{3}(A_{1,x} + \frac{1}{3}A_1 A_{1,y} + A_1 A_4)b_y \\ + b^3 - A_4 b^2 + A_7 b - A_9 = 0. \end{aligned} \quad (6)$$

PROOF. Dividing the operator (1) by $\partial_x + a\partial_y + b$, the requirement that this division be exact leads to the following set of equations between the coefficients.

$$a^3 - A_1 a^2 + A_2 a - A_3 = 0, \quad (7)$$

$$\begin{aligned} (A_1 - 3a)a_x + (3a^2 - 3A_1 a + 2A_2)a_y \\ - A_4 a^2 + A_5 a + (3a^2 - 2A_1 a + A_2)b = A_6, \end{aligned} \quad (8)$$

$$\begin{aligned} (A_1 - 3a)b_x + (3a^2 - 3A_1 a + 2A_2)b_y - (A_1 - 3a)b^2 \\ + (A_5 - 2A_4 a - 2A_1 a_y + 3a a_y - 3a_x)b \\ + a_{xx} + (A_1 - a)a_{xy} + (a^2 - A_1 a + A_2)a_{yy} \\ - 2a_x a_y + A_4 a_x - (A_1 - a)a_y^2 - (A_4 a - A_5)a_y + A_7 a - A_8 = 0, \end{aligned} \quad (9)$$

$$\begin{aligned} b_{xx} + (A_1 - a)b_{xy} + (a^2 - A_1 a + A_2)b_{yy} \\ - (2a_x + (A_1 - a)a_y + A_4 a - A_5)b_y \\ + (A_4 - 3b)b_x + (3a - 2A_1)bb_y + b^3 - A_4 b^2 + A_7 b - A_9 = 0. \end{aligned} \quad (10)$$

The algebraic equation (7) determines a . The following discussion is subdivided by the various cases.

Case i). Assume at first that it has three simple roots a_1, a_2 and a_3 . None of them may be rational, there may be a single rational solution, or all three roots are rational. For any of these roots the coefficient of b in (8) does not vanish because it is the derivative of (7) w.r.t. a . Therefore for each a_i , equation (8) determines the corresponding value b_i . For those values of a_i and b_i which satisfy the constraints (9) and (10), a right factor $l_i \equiv \partial_x + a_i \partial_y + b_i$ exists. If there are three right factors, by Lemma 2.3 D_{xxx} is completely reducible and there holds $D_{xxx} = \text{Lclm}(l_1, l_2, l_3)$.

Case ii). Assume now there is a twofold rational solution $a_1 = a_2 \neq a_3$ of equation (7). The double root $a_1 = \frac{1}{3}(A_1 - \sqrt{A_1^2 - 3A_2})$ is one of the roots of the equation $p \equiv 3a^2 - 2A_1 a + A_2 = 0$ of (7). The coefficient of b in (8) vanishes for $a = a_1$ because it is equal to p ; (8) becomes a constraint for a_1 . Assume that it is not obeyed. For the root a_3 , the coefficient b_3 follows from (8). The existence of a right factor involving a_3 and b_3 depends on whether they satisfy (9) and (10). If this is true there is a single right factor $\partial_x + a_3 \partial_y + b_3$.

Case iii). If the double root a_1 does satisfy the constraint following from (8), the corresponding value $b = b_1$ has to be determined from the partial Riccati equation (9) with $a = a_1$. Those values of a_1 and b_1 satisfying (10) have to be singled out. If a_3 and b_3 do not satisfy (9) and (10), the only possible factor is the one involving a_1 and b_1 . A simple calculation shows that the terms of (9) involving derivatives of b simplify to $b_x + a_1 b_y$. According to Lemma 5.3 of [7], the general solution for b_1 may contain an undetermined function Φ of $\phi(x, y)$, the first integral of $\frac{dy}{dx} = a_1(x, y)$. The condition (10) may be satisfied without constraints for Φ , it may restrict it to a certain form, or it may completely eliminate the corresponding solution. Similar arguments apply if there is a single rational solution for b_1 , or if there are two of them. Whenever two factors are obtained, by case ii) of Theorem 2.1 they generate a principal ideal.

Case iv). If again a_1 satisfies the constraint following from (8), and a_3 and b_3 satisfy (9) and (10) now, the same alternatives as in case iii) occur with an additional factor $\partial_x + a_3 \partial_y + b_3$.

Case v). Finally assume there is a threefold solution $a_1 = a_2 = a_3 = \frac{1}{3}A_1$. Then $A_2 = \frac{1}{3}A_1^2$ and $A_3 = \frac{1}{27}A_1^3$. The coefficient of b in (8) vanishes again, it becomes the constraint (2). The coefficients of b_x, b_y and b^2 in (9) vanish.

If $A_{1,x} + \frac{1}{3}A_1 A_{1,y} + \frac{2}{3}A_1 A_4 - A_5 \neq 0$, b is determined by (3). In order for a factor to exist, in addition (4) must be satisfied. This is subcase a). In the exceptional case that the coefficient of b in (3) vanishes, it becomes a constraint and b has to be determined from (6). This is subcase b). $\square$

In order to obtain a complete answer comprising all possible factorizations of the operator (1), second order right factors have to be taken into account as well. They are considered next.

PROPOSITION 3.2. Let the third order partial differential operator (1) be given with $A_k \in \mathbb{Q}(x, y)$ for all k . Its second order right factors

$$\partial_{xx} + a\partial_{xy} + b\partial_{yy} + c\partial_x + d\partial_y + f \quad (11)$$

are determined by the roots a_1, a_2 and a_3 of

$$a^3 - 2A_1 a^2 + (A_1^2 + A_2)a - A_1 A_2 + A_3 = 0.$$

The following alternatives may occur.

i) If a_1, a_2 and a_3 are three pairwise different rational solutions, for each a_i the corresponding values of b, c, d and f follow from equations (20) ... (22). Those values which satisfy constraints (23) yield a factor.

ii) If $a_1 = a_2 \neq a_3$ is a twofold solution and $b_1 = a_1^2 - A_1 a_1 + A_2$, there must hold

$$(A_1 - a_1)[a_{1,x} + (A_1 - a_1)a_{1,y} + A_1 a_1 - A_5] = b_{1,x} + (A_1 - a_1)b_{1,y} + A_4 b_1 - A_6. \quad (12)$$

The coefficient c has to be determined from the partial Riccati equation

$$\begin{aligned} c_x - (a - A_1)c_y - c^2 + \frac{3}{3a - 2A_1}[(a - \frac{1}{3}A_1)_x \\ - (a - A_1)(a - \frac{1}{3}A_1)_y + \frac{4}{3}(a - \frac{1}{2}A_1)A_4 - \frac{1}{3}A_5]c \\ + \frac{1}{3a - 2A_1}[a_{xx} - 2(a - A_1)a_{xy} \\ + (a^2 - 2A_1a + A_1^2)a_{yy} - a_y(a - A_1)_x \\ + (a - A_1)(a_y^2 - A_{1,y}a_y - 2A_4a_y - A_{4,y}a + A_{5,y} + A_7) \\ + A_{4,x}a + 2A_4a_x - A_{5,x} - A_4A_5 + A_8 + A_4^2a] = 0. \end{aligned} \quad (13)$$

If it has a rational solution, d and f follow from the first equation of (21) and (22) respectively. The values satisfying the constraint

$$\begin{aligned} c_{xx} - 2(a - A_1)c_{xy} + (a - A_1)^2c_{yy} \\ + [(a - A_1)_x - \frac{1}{2}(a^2 - A_1a - A_1^2)_y + (a - A_1)(2A_4 - 3c)]c_y \\ + [(a - A_1)(3c - A_4)_y + A_{4,x} + A_4^2 + A_7]c \\ + A_{7,x} - (a - A_1)A_{7,y} + A_4A_7 - A_9 = 0. \end{aligned} \quad (14)$$

lead to a factor.

iii) If $a_1 = a_2 = a_3 = \frac{2}{3}A_1$ is a threefold solution and $b_1 = \frac{1}{9}A_1^2$, c is uniquely determined by

$$\begin{aligned} (A_{1,x} + \frac{1}{3}A_1A_{1,y} + \frac{2}{3}A_1A_4 - A_5)c = \frac{2}{3}A_{1,xx} + \frac{4}{9}A_1A_{1,xy} \\ + \frac{2}{27}A_1^2A_{1,yy} + \frac{2}{9}A_{1,x}A_{1,y} + \frac{4}{3}A_4A_{1,x} + \frac{2}{27}A_1A_{1,y}^2 \\ + \frac{4}{9}A_1A_4A_{1,y} + \frac{2}{9}A_1^2A_{4,y} + \frac{2}{3}A_1(A_{4,x} + A_4^2) \\ - \frac{1}{3}A_1(A_{5,y} + A_7) - A_4A_5 - A_{5,x} + A_8 \end{aligned} \quad (15)$$

if its coefficient does not vanish. The remaining coefficients follow from

$$\begin{aligned} d - \frac{1}{3}A_1c + \frac{2}{3}A_{1,x} + \frac{2}{9}A_1A_{1,y} + \frac{2}{3}A_1A_4 - A_5 = 0, \\ f + c_x + \frac{1}{3}A_1c_y - c^2 + A_4c - A_7 = 0. \end{aligned} \quad (16)$$

In addition the two constraints

$$A_1^2A_4 - 3A_1A_5 + 9A_6 = 0, \quad (17)$$

$$\begin{aligned} c_{xx} + \frac{2}{3}A_1c_{xy} + \frac{1}{9}A_1^2c_{yy} + (2A_4 - 3c)c_x \\ + (\frac{1}{3}A_{1,x} + \frac{1}{9}A_1A_{1,y} + \frac{2}{3}A_1A_4 - A_1c)c_y \\ + c^3 - 2A_4c^2 + (\frac{1}{3}A_1A_{4,y} + A_{4,x} + A_4^2 + A_7)c \\ - \frac{1}{3}A_1A_{7,y} - A_4A_7 - A_{7,x} + A_9 = 0 \end{aligned} \quad (18)$$

must be satisfied. If the coefficient of c in (15) vanishes, it becomes a constraint and c has to be determined from (18).

PROOF. Dividing the operator (1) by (11), the requirement that this division be exact leads to the following set of equations for the coefficients.

$$a^3 - 2A_1a^2 + (A_1^2 + A_2)a - A_1A_2 + A_3 = 0, \quad (19)$$

$$b - a^2 + A_1a - A_2 = 0, \quad (20)$$

$$\begin{aligned} (A_1 - 2a)c + d + a_x + (A_1 - a)a_y + A_4a - A_5 = 0, \\ (a^2 - A_1a + A_2)c - (A_1 - a)d - b_x \\ - (A_1 - a)b_y - A_4b + A_6 = 0, \end{aligned} \quad (21)$$

$$f + c_x + (A_1 - a)c_y - c^2 + A_4c - A_7 = 0, \quad (22)$$

$$\begin{aligned} (A_1 - a)f - d_x - (A_1 - a)d_y - (A_4 - c)d + A_8 = 0, \\ f_x + (A_1 - a)f_y + (A_4 - c)f - A_9 = 0. \end{aligned} \quad (23)$$

The algebraic equation (19) determines always a . The following discussion is subdivided by the various cases of the above theorem.

Case i) Assume it has three simple roots $a_1 \neq a_2 \neq a_3$. The corresponding values b_i may be determined from (20). The coefficient determinant of c and d in (21) is $a^2 - \frac{4}{3}A_1a + \frac{1}{3}(A_1^2 + A_2)$, it is the derivative of (19) w.r.t. a that does not vanish for simple roots. Therefore for any simple root a_i , the corresponding values c_i and d_i may be determined from (21). Finally f_i follows from (22). These values have to satisfy the constraints (23) in order to determine a factor.

Case ii) Assume now there is a twofold rational solution $a_1 = a_2 \neq a_3$ of (19). The double root $a_1 = \frac{2}{3}A_1 + \frac{1}{3}\sqrt{A_1^2 - 3A_2}$ is one of the roots of the equation $3a^2 - 4A_1a + A_1^2 + A_2 = 0$ of (19). In order to exclude a triple root there must hold $A_1^2 - 3A_2 \neq 0$; b_1 follows from (20). Because the coefficient determinant of c and d in (21) vanishes for $a = a_1$, one has to proceed as follows. Eliminating d from the first equation (21) and substituting it into the second one yields the constraint (12). Eliminating d from the first equation (21), f from (22) and substituting them into the first equation of (23), the partial Riccati equation (13) for c is obtained. If it has a rational solution c_1 , the coefficient d_1 may be obtained from the first equation (21) and f_1 from (22). Substituting these values into the second equation of (23), the constraint (14) follows. It may exclude a factor, or it may constrain the undetermined elements in the solution of (13).

Case iii) Finally assume there is a threefold solution $a_1 = a_2 = a_3 = \frac{2}{3}A_1$ with the corresponding value $b_1 = \frac{1}{9}A_1^2$. Then $A_2 = \frac{1}{3}A_1^2$ and $A_3 = \frac{1}{27}A_1^3$. The coefficient determinant for c and d in (21) vanishes. By elimination equation (15) is obtained for c if $A_{1,x} + \frac{1}{3}A_1A_{1,y} + \frac{2}{3}A_1A_4 - A_5 \neq 0$, and (16) for the remaining coefficients d and e . In order for a factor to exist, the two constraints (17) and (18) must be satisfied. In the exceptional case that the coefficient of c in (15) vanishes, it becomes a constraint; c has to be determined from (18). $\square$

The next corollary summarizes to what extent the factors of Propositions 3.1 and 3.2 may be obtained algorithmically.

COROLLARY 3.3. Any first-order factor of (1) corresponding to a simple root of (7), or any second-order factor corresponding to a simple root of (19), may be determined algorithmically. In order to determine any such factor corresponding to a twofold root of (7) or (19) requires to find rational solutions of a partial Riccati equation. Finally, a factor corresponding to a threefold root may be determined algorithmically if $A_{1,x} + \frac{1}{3}A_1A_{1,y} + \frac{2}{3}A_1A_4 \neq A_5$, otherwise a quasilinear second-order pde has to be solved for which a solution algorithm is not known.

PROOF. Testing the conditions of case *i*) of both Proposition 3.1 and Proposition 3.2 requires only differentiations and arithmetic in the base field. Case *ii*), *iii*) and *iv*) of Proposition 3.1 dealing with a twofold root of (7) require to determine rational solutions of the partial Riccati equation (8) for the coefficient a . To this end a general first-order ode must be solved as discussed in Appendix B of [7]. The same argument applies to case *ii*) of Proposition 3.2. In case *v*) of Proposition 3.1, b follows from the linear algebraic equation (3) if its coefficient does not vanish; this leads to the above condition. If it does vanish, b has to be determined from equation (6). Similar arguments apply for case *iii*) of Proposition 3.2 and equation (18) for the coefficient c . $\square$

After the possible factorizations of an operator (1) have been determined, a listing of its completely reducible components may be set up as follows.

THEOREM 3.4. Any differential operator

$$L \equiv \partial_{xxx} + A_1\partial_{xxy} + A_2\partial_{xyy} + A_3\partial_{yyy} + A_4\partial_{xx} + A_5\partial_{xy} + A_6\partial_{yy} + A_7\partial_x + A_8\partial_y + A_9 \quad (24)$$

decomposes into completely reducible components corresponding to one of the types $\mathcal{L}_{xxx}^{(k)}$, $k = 1, \dots, 5$, defined as follows; L_{xxx} , L_{xx} and L_x are completely reducible operators with leading derivatives ∂_{xxx} , ∂_{xx} or ∂_x respectively; $\mathcal{J}_5^{(1,2)}$ is the ideal $\langle \partial_{xxx}, \partial_{xxy} \rangle_{LT}$ defined in Table 1. Upper indices distinguish different copies within a type definition.

$$\mathcal{L}_{xxx}^{(1)} : L_{xxx}, \mathcal{L}_{xxx}^{(2)} : L_{xx}L_x, \mathcal{L}_{xxx}^{(3)} : L_xL_{xx}, \mathcal{L}_{xxx}^{(4)} : L_x^{(1)}L_x^{(2)}L_x^{(3)},$$

$$\mathcal{L}_{xxx}^{(5)} : Syz(\langle L, \mathcal{J}_5^{(1,2)} \rangle \mathcal{J}_5^{(1,2)}).$$

The decomposition $\mathcal{L}_{xxx}^{(1)}$ is completely reducible.

PROOF. It is based on Propositions 3.1 and 3.2. In the separable case *i*) of Proposition 3.1 there may be three factors with a principal intersection, this yields type $\mathcal{L}_{xxx}^{(1)}$. There may be two factors which do not have a principal intersection. If they are divided out and the decompositions of the respective second-order factor are taken into account (see Section 4 of [7]), type $\mathcal{L}_{xxx}^{(5)}$ is obtained. If case *i*) allows only a single factor, or if case *ii*), or subcase *c*) of case *iii*), or case *v*) applies, and again the decompositions of the corresponding second order left factor are taken into account, decomposition types $\mathcal{L}_{xxx}^{(2)}$ or $\mathcal{L}_{xxx}^{(4)}$ follow. For case *iii*), the principality of the intersection of the right factors leads to type $\mathcal{L}_{xxx}^{(3)}$. In case *iv*) the decompositions of case *iii*) are extended by an additional first order factor. According to Lemma 2.3, combined with the factors already obtained they generate a principal intersection covered by $\mathcal{L}_{xxx}^{(1)}$. By definition, an irreducible operator (24) corresponds to a type $\mathcal{L}_{xxx}^{(1)}$ decomposition. $\square$

COROLLARY 3.5. Let l_x , l_{xx} and l_{xxx} denote irreducible operators with leading term ∂_x , ∂_{xx} or ∂_{xxx} . An additional upper index distinguishes different copies of the respective operator. Φ is an undetermined function of a single argument. The types $\mathcal{L}_{xx}^{(i,j)}$ defined in Table 2 are refinements of the types $\mathcal{L}_{xxx}^{(i)}$, $i = 1, \dots, 5$ defined in Theorem 3.4.

This is an immediate consequence of the proof given for the preceding theorem if the various factorization alternatives of Proposition 3.1 and Proposition 3.2 are not merged into completely reducible components.

EXAMPLE 3.6. Consider the operator

$$L \equiv \partial_{xxx} + (y+2)\partial_{xxy} + (y+1)\partial_{xyy} + (1 - \frac{1}{y})\partial_{xx} + (y+2 - \frac{1}{y})\partial_{xy} + (y+1)\partial_{yy} - \frac{1}{y}\partial_x - \frac{1}{y}\partial_y.$$

Equation (7) reads $a^3 - (y+2)a^2 + (y+1)a = 0$ with the three roots $a_1 = 0$, $a_2 = 1$ and $a_3 = y+1$, i. e. case *i*) of Proposition 3.1. The corresponding values of b are $b_1 = 1$, $b_2 = b_3 = 0$. It turns out that all pairs a_i , b_i satisfy conditions (9) and (10). Consequently there are three right factors $l_1 = \partial_x + 1$, $l_2 = \partial_x + \partial_y$ and $l_3 = \partial_x + (y+1)\partial_y$. There holds $L = Lclm(l_1, l_2, l_3)$, i. e. the decomposition type is $\mathcal{L}_{xxx}^{(1,2)}$. The general solution of $Lz = 0$ is

$$z = F(y)e^{-x} + G(x-y) + H[(y+1)e^{-x}]$$

where F , G and H are undetermined functions.

EXAMPLE 3.7. (Blumberg 1912) In his dissertation Blumberg [1] considered the third order operator

$$L \equiv \partial_{xxx} + x\partial_{xxy} + 2\partial_{xx} + 2(x+1)\partial_{xy} + \partial_x + (x+2)\partial_y \quad (25)$$

generating a principal ideal of gauge (1, 3). He gave its factorizations

$$L = \begin{cases} (\partial_{xx} + x\partial_{xy} + \partial_x + (x+2)\partial_y)(\partial_x + 1) \\ (\partial_{xx} + 2\partial_x + 1)(\partial_x + x\partial_y). \end{cases} \quad (26)$$

with two completely reducible second order left factors.

This result may be obtained by Proposition 3.1 as follows. Equation (7) is $a^3 - xa^2 = a^2(a-x) = 0$ with the double root $a_1 = 0$, and the simple root $a_2 = x$. The latter yields $b_2 = 0$. Because these values satisfy (9) and (10), the factor $l_2 = \partial_x + x\partial_y$ is obtained, i. e. case *iv*) of Theorem 3.1 applies. For the double root $a_1 = 0$, from (9) the Riccati equation

$$b_x - b^2 + \left(2 + \frac{2}{x}\right)b - 1 - \frac{2}{x} = 0$$

is obtained with the general solution $b = 1 + \frac{1}{x} - \frac{1}{x + \Phi(y)}$. Substitution into (10) yields $\Phi = 0$, i.e. $b = 1$ and the factor $l_1 = \partial_x + 1$.

The second order left factor in the first line at the right hand side of (26) is absolutely irreducible, whereas the second order factor in the second line is the left intersection of two first order factors, i. e. (26) may be further decomposed into irreducibles as

$$L = \begin{cases} (\partial_{xx} + x\partial_{xy} + \partial_x + (x+2)\partial_y)(\partial_x + 1) \\ Lclm(\partial_x + 1, \partial_x + 1 - \frac{1}{x})(\partial_x + x\partial_y). \end{cases} \quad (27)$$

$\mathcal{L}_{xxx}^{(1)}$	$\mathcal{L}_{xxx}^{(1,1)} : l_{xxx}, \mathcal{L}_{xxx}^{(1,2)} : Lclm(l_x^{(1)}, l_x^{(2)}, l_x^{(3)}), \mathcal{L}_{xxx}^{(1,3)} : Lclm(l_{xx}, l_x).$
$\mathcal{L}_{xxx}^{(2)}$	$\mathcal{L}_{xxx}^{(2,1)} : l_{xx}l_x, \mathcal{L}_{xxx}^{(2,2)} : Lclm(l_x^{(1)}, l_x^{(2)})l_x^{(3)}, \mathcal{L}_{xxx}^{(2,3)} : Lclm(l_x^{(1)}(\Phi))l_x^{(2)}.$
$\mathcal{L}_{xxx}^{(3)}$	$\mathcal{L}_{xxx}^{(3,1)} : l_xl_{xx}, \mathcal{L}_{xxx}^{(3,2)} : l_x^{(1)}Lclm(l_x^{(2)}, l_x^{(3)}), \mathcal{L}_{xxx}^{(3,3)} : l_x^{(1)}Lclm(l_x^{(2)}(\Phi)).$
$\mathcal{L}_{xxx}^{(4)}$	$\mathcal{L}_{xxx}^{(4,1)} : l_x^{(1)}l_x^{(2)}l_x^{(3)}.$
$\mathcal{L}_{xxx}^{(5)}$	$\mathcal{L}_{xxx}^{(5)} : Syz(\langle L \rangle, \mathcal{J}_5^{(1,2)})Lclm(\langle (\partial_{xx} + \dots)(\partial_x + \dots) \rangle, \langle (\partial_{xy} + \dots)(\partial_x + \dots) \rangle).$

Table 2. The decomposition types of Corollary 3.5

The intersection ideal of l_1 and l_2 is not principal, by Theorem 2.1 it is

$$\begin{aligned}
Lclm(l_1, l_2) &= \langle L_1 \equiv \partial_{xxx} - x^2 \partial_{xyy} \\
&\quad + 3\partial_{xx} + (2x+3)\partial_{xy} - x^2 \partial_{yy} + 2\partial_x + (2x+3)\partial_y, \\
L_2 &\equiv \partial_{xxy} + x\partial_{xyy} - \frac{1}{x}\partial_{xx} - \frac{1}{x}\partial_{xy} + x\partial_{yy} \\
&\quad - \frac{1}{x}\partial_x - (1 + \frac{1}{x})\partial_y \rangle
\end{aligned} \tag{28}$$

with gauge (1, 2), therefore the decomposition (26) is of type $\mathcal{L}_{xxx}^{(5)}$. Both generators L_1 and L_2 have the divisors l_1 and l_2 . There holds $L \in Lclm(l_1, l_2)$, explicitly $L = L_1 + xL_2$.

As a consequence the system of equations $L_1z = 0$ and $L_2z = 0$ has the two solutions $F(y - \frac{1}{2}x^2)$ and $G(y)e^{-x}$; they correspond to $l_2z = 0$ and $l_1z = 0$ respectively.

In order to obtain the remaining part of the solution of Blumberg's equation $Lz = 0$ of gauge (1, 1) the relative syzygies module

$$\begin{aligned}
Syz(\langle L \rangle, \langle L_1, L_2 \rangle) \\
&= \langle (1, x), (-\partial_y + \frac{1}{x}, \partial_x - x\partial_y + 2 + \frac{1}{x}) \rangle \\
&= \langle (1, x), (0, \partial_x + 1 + \frac{1}{x}) \rangle
\end{aligned}$$

is constructed. Introducing the new differential indeterminates z_1 and z_2 , the equations $z_1 + xz_2 = 0$ and $z_{2,x} + (1 + \frac{1}{x})z_2 = 0$ are obtained with the solution $z_1 = -H(y)e^{-x}$ and $z_2 = H(y)\frac{1}{x}e^{-x}$, H an undetermined function of y . The desired solution is a special solution of the inhomogeneous system $L_1z = -H(y)e^{-x}$, $L_2z = H(y)\frac{1}{x}e^{-x}$. The solution of the corresponding homogeneous system is already known, therefore the general solution of $Lz = 0$ may be written in terms of integrations as

$$z = F(y - \frac{1}{2}x^2) + G(y)e^{-x} + \int H(\bar{y} + \frac{1}{2}x^2)e^{-x}dx \Big|_{\bar{y}=y-\frac{1}{2}x^2} \tag{29}$$

where $\bar{y} = y - \frac{1}{2}x^2$.

EXAMPLE 3.8. Consider the operator

$$\begin{aligned}
L &\equiv \partial_{xxx} + (x+y-1)\partial_{xxy} - (x+y)\partial_{xyy} - (x-y-1)\partial_{xx} \\
&\quad - (x-y-1)\partial_{xx} - (x^2+xy-x+1)\partial_{xy} - (x+y)\partial_{yy} \\
&\quad - (xy+x-y+1)\partial_x - (x^2+xy+y)\partial_y - xy - 1.
\end{aligned}$$

Equation (7) reads $a^3 - (x+y+1)a^2 - (x+y)a = 0$ with roots $a_1 = 0$, $a_2 = -1$ and $a_3 = x+y$, i. e. case *i*) of Proposition 3.1 applies. Equation (8) yields the corresponding values

$$b_1 = 1, \quad b_2 = -x \quad \text{and} \quad b_3 = -\frac{2x^2 + xy - x - y^2 - y + 2}{4x + 5y - 3}.$$

Only a_1 and b_1 satisfy the constraints (2) and (4). Consequently there is a single first order factor $l_1 \equiv \partial_x + 1$. Pursuant to case *i*) of Proposition 3.2 a second order factor does not exist. Dividing L by l_1 yields the operator

$$\begin{aligned}
&\partial_{xx} + (x+1)\partial_{xy} + x\partial_{yy} + (y+1)\partial_x + (xy+1)\partial_y + x + y \\
&= (\partial_x + x\partial_y + 1)(\partial_x + \partial_y + y)
\end{aligned}$$

Consequently the operator L has a type $\mathcal{L}_{xxx}^{(4,1)}$ decomposition

$$L = (\partial_x + x\partial_y + 1)(\partial_x + \partial_y + y)(\partial_x + 1).$$

EXAMPLE 3.9. Consider the operator

$$\begin{aligned}
L &\equiv \partial_{xxx} + (y+1)\partial_{xxy} + (x+y+1)\partial_{xx} + (xy+x+y^2+2y+1)\partial_{xy} \\
&\quad + (x+y)\partial_x + (xy+x+y^2+y)\partial_y.
\end{aligned}$$

Equation (7) reads $a^2(a-y-1) = 0$ with double root $a_1 = a_2 = 0$ and single root $a_3 = y+1$. It turns out that case *iv*) applies with $a_1 = 0$, $b_1 = 1$ and $a_3 = y+1$, $b_3 = 0$. The corresponding first order factors yield the divisor as the principal intersection

$$Lclm(\partial_x + 1, \partial_x + (y+1)\partial_y) = \partial_{xx} + (y+1)\partial_{xy} + \partial_x + (y+1)\partial_y.$$

Consequently L has the decomposition

$$(\partial_x + x + y)Lclm(\partial_x + 1, \partial_x + (y+1)\partial_y)$$

of type $\mathcal{L}_{xxx}^{(3,2)}$.

4. PDE'S WITH LEADING DERIVATIVE ∂_{xxy}

If an equation does not contain a derivative ∂_{xxx} but only ∂_{yyy} , permuting x and y leads to an equation of the form (1) such that the above theorem may be applied. If there is neither a term ∂_{xxx} or ∂_{yyy} , the general third order operator

$$\begin{aligned}
D_{xxy} &\equiv \partial_{xxy} + A_1\partial_{xyy} + A_2\partial_{xx} + A_3\partial_{xy} + A_4\partial_{yy} \\
&\quad + A_5\partial_x + A_6\partial_y + A_7
\end{aligned} \tag{30}$$

is obtained. Its possible decompositions are described in this subsection. Like in the previous case, first- and second-order factors are considered separately.

PROPOSITION 4.1. *Let the third order partial differential operator (30) be given with $A_k \in \mathbb{Q}(x, y)$ for all k . The following first order right factors may occur.*

- i)* If $A_1 \neq 0$, $a_1 = 0$, $a_2 = A_1$, and b is determined from $a_x - (3a - 2A_1)a_y - A_2a^2 + A_3a - (2a - A_1)b = A_4$, a first order right factor $\partial_x + a_i\partial_y + b_i$ exists if a_i and b_i satisfy the constraints

$$\begin{aligned}
&b_x - (3a - 2A_1)b_y - b^2 - (2a_y + 2A_2a - A_3)b + a_{xy} - (a - A_1)a_{yy} \\
&\quad + A_2a_x - a_y^2 - (A_2a - A_3)a_y + A_5a = A_6,
\end{aligned}$$

$$b_{xy} + (A_1 - a)b_{yy} + A_2b_x - (2b + a_y + A_2a - A_3)b_y - A_2b^2 + A_5b = A_7.$$

ii) If $a = A_1 = A_4 = 0$, the factor $\partial_x + b$ exists with $b = \frac{A_{6,y} + A_2A_6 - A_7}{A_{3,y} + A_2A_3 - A_5}$ if $A_{3,y} + A_2A_3 - A_5 \neq 0$ and $b_x - b^2 + A_3b = A_6$.

iii) If $a = A_1 = A_4 = 0$ and the two constraints

$$A_{6,y} + A_2A_6 - A_7 = 0, \quad A_{3,y} + A_2A_3 - A_5 = 0$$

are satisfied, the following two subcases may occur.

a) There is a right factor of the form $\partial_x + R(x, y, \Phi(y))$, where R is the general rational solution of

$$b_x - b^2 + A_3b - A_6 = 0 \quad (31)$$

involving an undetermined function $\Phi(y)$.

b) There is a single factor, or there are two factors $\partial_x + r_i(x, y)$ where $r_i(x, y)$ are special rational solutions of (31).

iv) A factor $\partial_y + A_2$ exists if the following two constraints are satisfied.

$$\begin{aligned} A_5 &= A_2A_3 + 2A_{2,x} + A_{2,y}A_1 - A_2^2A_1, \\ A_7 &= A_2A_6 + A_{2,y}A_4 - A_4A_2^2 + A_3A_{2,x} \\ &\quad + A_{2,xx} + A_{2,xy}A_1 - 2A_{2,x}A_2A_1. \end{aligned}$$

PROPOSITION 4.2. Let the third order partial differential operator (30) be given with $A_k \in \mathbb{Q}(x, y)$ for all k . The following second order right factors may occur.

i) A factor with leading derivative ∂_{xx} does not contain a derivative ∂_{yy} . A factor $\partial_{xx} + a_1\partial_{xy} + a_2\partial_x + a_3\partial_y + a_4$ exists if the two constraints

$$\begin{aligned} A_{1,yy} + 2A_{1,y}A_2 + A_{2,y}A_1 - A_{3,y} \\ + A_2^2A_1 - A_2A_3 + A_5 &= 0, \\ A_{4,yy} + 2A_{4,y}A_2 + A_{4,y}A_2 - A_{6,y} \\ + A_2^2A_4 - A_2A_6 + A_7 &= 0 \end{aligned}$$

are satisfied. Then $a_1 - A_1 = 0$, $a_2 = A_3 - A_1A_2 - A_{1,y}$, $a_3 = A_4$ and $a_4 = A_6 - A_{5,y} - A_2A_4$.

ii) A factor $\partial_{xy} + a_1\partial_{yy} + a_2\partial_x + a_3\partial_y + a_4$ may exist if one of the four alternatives applies.

a) If the conditions $A_1 = A_4 = 0$, $A_{2,x} + \frac{1}{2}A_2A_3 - \frac{1}{2}A_5 = 0$ and $A_{2,xx} + A_{2,x}A_3 + \frac{1}{2}A_2A_6 - \frac{1}{2}A_7 = 0$ are satisfied, the coefficients of the factor may be determined from $a_1 = 0$, $a_2 = A_2$,

$$a_{3,x} - a_3^2 + A_3a_3 - A_6 = 0 \text{ and } a_4 = A_2a_3 + 2A_{2,x}.$$

b) If the conditions $A_1 = A_4 = 0$ and $A_{2,x} + \frac{1}{2}A_2A_3 - \frac{1}{2}A_5 \neq 0$ are satisfied, define $P \equiv A_{2,x} + A_2A_3 - A_5$ and $Q \equiv 2A_{2,x} + A_2A_3 - A_5$. Then $a_1 = 0$, $a_2 = A_2$, $a_3 = \frac{1}{Q}(P_x + A_3P - A_2A_6 + A_7)$ and $a_4 = A_2(a_3 - A_3) - A_{2,x} + A_5$. The following condition must hold. $a_{3,x} - a_3^2 + A_3a_3 - A_6 = 0$.

c) If $A_1 \neq 0$ there is a factor with coefficients $a_1 = A_1$, $a_2 = A_2$,

$$a_3 = A_3 + \frac{A_{1,x}}{A_1} - \frac{A_4}{A_1}, \quad a_4 = \left(\frac{A_{1,x}}{A_1} - \frac{A_4}{A_1}\right)A_2 - A_{2,x} + A_5$$

if the following constraints are satisfied.

$$a_{3,x} - a_3^2 + A_3a_3 - A_6 = 0, \quad a_{4,x} - a_3a_4 + A_3a_4 - A_7 = 0.$$

d) If $A_1 \neq 0$ there is a factor with coefficients $a_1 = 0$, $a_2 = A_2$, $a_3 = \frac{A_4}{A_1}$ and

$$a_4 = \frac{A_2A_4}{A_1} - A_{2,x} - A_{2,y}A_1 + A_2^2A_1 - A_2A_3 + A_5$$

if the following conditions are satisfied

$$a_{3,x} + a_{3,y}A_1 - A_1A_2a_3 + A_3a_3 + A_1a_4 - A_6 = 0,$$

$$a_{4,x} + a_{4,y}A_1 - A_1A_2a_4 + (A_3 - a_3)a_4 - A_7 = 0.$$

The proof of the two preceding propositions are similar to those in the previous section and are therefore omitted.

In order to solve concrete problems it is important to know to what extent the factorizations described in this section may be determined algorithmically. The answer to this question is given in the following corollary.

COROLLARY 4.3. Any principal divisor of an operator (30) may be determined algorithmically. The same is true for a possible Laplace divisor of a given order.

PROOF. Testing the conditions (34) of case i) of Proposition 5.1 requires only differentiations and arithmetic in the base field. The same is true for case ii), subcase b). In subcase a) of case ii) the rational solutions of an ordinary Riccati equation have to be determined which is algorithmically possible. In [8] the algorithmic construction of Laplace divisors for fixed values of m or n has been shown. $\square$

THEOREM 4.4. Any differential operator

$$\begin{aligned} D_{xxy} &\equiv \partial_{xxy} + A_1\partial_{xyy} \\ &\quad + A_2\partial_{xx} + A_3\partial_{xy} + A_4\partial_{yy} + A_5\partial_x + A_6\partial_y + A_7. \end{aligned} \quad (32)$$

decomposes uniquely into largest completely reducible components corresponding to one of the types $\mathcal{L}_{xxy}^{(k)}$, $k = 1, \dots, 11$, defined as follows; L_{xxy} , L_{xy} , L_{xx} , L_x and L_y are completely reducible operators with leading terms ∂_{xxy} , ∂_{xy} , ∂_{xx} , ∂_x or ∂_y respectively.

$$\mathcal{L}_{xxy}^{(1)} : L_{xxy}, \quad \mathcal{L}_{xxy}^{(2)} : L_{xy}L_x, \quad \mathcal{L}_{xxy}^{(3)} : L_{xx}L_y,$$

$$\mathcal{L}_{xxy}^{(4)} : L_xL_{xy}, \quad \mathcal{L}_{xxy}^{(5)} : L_yL_{xx},$$

$$\mathcal{L}_{xxy}^{(6)} : L_x^{(1)}L_x^{(2)}L_y, \quad \mathcal{L}_{xxy}^{(7)} : L_x^{(1)}L_yL_x^{(2)}, \quad \mathcal{L}_{xxy}^{(8)} : L_yL_x^{(1)}L_x^{(2)},$$

$$\mathcal{L}_{xxy}^{(9)} : \text{Syz}(\langle \partial_{xxy} \rangle, \mathcal{J}_4^{(1,1)}(m))\mathcal{J}_4^{(1,1)}(m),$$

$$\mathcal{L}_{xxy}^{(10)} : \text{Syz}(\langle L \rangle, \mathcal{J}_4^{(1,2)}(n))\mathcal{J}_4^{(1,2)}(n),$$

$$\mathcal{L}_{xxy}^{(11)} : \text{Syz}(\langle L \rangle, \mathcal{J}_6^{(1,2)})\mathcal{J}_6^{(1,2)}.$$

If the completely reducible components are split into irreducible ones, the following refined decomposition scheme is obtained.

COROLLARY 4.5. Let l_x , l_y , l_{xx} , l_{xy} and l_{xxy} denote irreducible operators with leading derivatives as determined by the respective subindex. An additional upper index distinguishes different copies of the respective operator; Φ is an undetermined function of a single argument. The types $\mathcal{L}_{xxy}^{(i,j)}$ defined in Table 3 are refinements of the types $\mathcal{L}_{xxy}^{(i)}$, $i = 1, \dots, 10$ of Theorem 5.4.

This is an immediate consequence of the proof given for the preceding theorem if the various factorization alternatives of Proposition 3.1 and Proposition 3.2 are not merged into completely reducible components.

$\mathcal{L}_{xxy}^{(1)}$	$\mathcal{L}_{xxy}^{(1,1)} : l_{xyy}, \mathcal{L}_{xxy}^{(1,2)} : Lclm(l_x, l_y^{(1)}, l_y^{(2)}), \mathcal{L}_{xxy}^{(1,3)} : Lclm(l_{yy}, l_x),$ $\mathcal{L}_{xxy}^{(1,4)} : Lclm(l_{xy}, l_y), \mathcal{L}_{xxy}^{(1,5)} : Lclm(\mathcal{J}_4^{(1,1)}(m), \mathcal{J}_4^{(1,2)}(n)).$
$\mathcal{L}_{xxy}^{(2)}$	$\mathcal{L}_{xxy}^{(2,1)} : l_{yy}l_x, \mathcal{L}_{xxy}^{(2,2)} : Lclm(l_y^{(1)}, l_y^{(2)})l_x, \mathcal{L}_{xxy}^{(2,3)} : Lclm(l_y(\Phi))l_x.$
$\mathcal{L}_{xxy}^{(3)}$	$\mathcal{L}_{xxy}^{(3,1)} : l_{xy}l_y, \mathcal{L}_{xxy}^{(3,2)} : Lclm(l_x, l_y^{(1)})l_y^{(2)}, \mathcal{L}_{xxy}^{(3,3)} : Lclm(\mathcal{J}_2^{(1,1)}(m), \mathcal{J}_3^{(1,1)}(n))l_y.$
$\mathcal{L}_{xxy}^{(4)}$	$\mathcal{L}_{xxy}^{(4,1)} : l_xl_{yy}, \mathcal{L}_{xxy}^{(4,2)} : l_xLclm(l_y^{(1)}, l_y^{(2)}), \mathcal{L}_{xxy}^{(4,3)} : l_xLclm(l_y(\Phi)).$
$\mathcal{L}_{xxy}^{(5)}$	$\mathcal{L}_{xxy}^{(5,1)} : l_y l_{xy}, \mathcal{L}_{xxy}^{(5,2)} : l_y^{(1)} Lclm(l_x, l_y^{(1)}).$
$\mathcal{L}_{xxy}^{(6,7,8)}$	$\mathcal{L}_{xxy}^{(6,1)} : l_x l_y^{(1)} l_y^{(2)}, \mathcal{L}_{xxy}^{(7,1)} : l_y^{(1)} l_y^{(2)} l_x, \mathcal{L}_{xxy}^{(8,1)} : l_y^{(1)} l_x l_y^{(2)}.$
$\mathcal{L}_{xxy}^{(9)}$	$\mathcal{L}_{xxy}^{(9,1)} : Syz(\langle D_{xyy} \rangle, \langle D_{xyy}, D_x^m \rangle) \langle D_{xyy}, (\partial_x^{m-1} + \dots)(\partial_x + A_2) \rangle,$
$\mathcal{L}_{xxy}^{(10)}$	$\mathcal{L}_{xxy}^{(10,1)} : Syz(\langle D_{xyy} \rangle, \langle D_{xyy}, D_y^n \rangle) \langle D_{xyy}, (\partial_y^{n-1} + \dots)(\partial_y + A_2) \rangle.$
$\mathcal{L}_{xxy}^{(11)}$	$\mathcal{L}_{xxy}^{(11,1)} : Syz(\langle \partial_{xyy} \rangle, \mathcal{J}_6^{(1,2)}) Lclm((\partial_{xy} + \dots)(\partial_x + \dots), (\partial_{xy} + \dots)(\partial_y + \dots)).$

Table 3. The decomposition types of Corollary 4.5

EXAMPLE 4.6. Consider the operator

$$L \equiv \partial_{xxy} + \partial_{xyy} + y\partial_{xx} + (x - y - 1)\partial_{xy} - \partial_{yy} + (xy + x + 1)\partial_x - (x - y)\partial_y + xy + 1.$$

By case *i*) of Proposition 4.1 the two factors $l_1 \equiv \partial_x - 1$ and $l_2 \equiv \partial_x + \partial_y - y$ exist. They have a principal left intersection $\partial_{xx} + \partial_{xy} - (y + 1) - \partial_y + y$; consequently by Theorem 4.4, L has the type $\mathcal{L}_{xxy}^{(5)}$ decomposition

$$L = (\partial_y + x)Lclm(\partial_x - 1, \partial_x + \partial_y - y).$$

The two first-order right factors contribute $F(y)\exp(-x)$ and $G(x - y)\exp(\frac{1}{2}x^2 - xy)$ to the solution.

5. PDE'S WITH LEADING DERIVATIVE ∂_{xyy}

If an equation contains a single mixed derivative of order three it may be assumed to be ∂_{xyy} . The corresponding operator is

$$D_{xyy} \equiv \partial_{xyy} + A_1\partial_{xx} + A_2\partial_{xy} + A_3\partial_{yy} + A_4\partial_x + A_5\partial_y + A_6. \quad (33)$$

Its possible decompositions are discussed in this subsection. First- and second-order factors are considered one after another. Proofs are again omitted.

PROPOSITION 5.1. *Let the third order partial differential operator (33) be given with $A_k \in \mathbb{Q}(x, y)$ for all k . The following first order right factors may occur.*

- i) A factor with leading derivative ∂_x exists if the two constraints*

$$\begin{aligned} A_5 &= 2A_{3,y} + A_2A_3, \\ A_6 &= A_{3,yy} + A_{3,y}A_2 + A_{3,x}A_1 + A_3(A_4 - A_1A_3) \end{aligned} \quad (34)$$

are satisfied; then the factor is $\partial_x + A_3$.

- ii) For a factor $\partial_y + c$ to exist there must hold $A_1 = 0$. Define*

$$P \equiv A_{4,x} + A_3A_4 - A_6 \quad \text{and} \quad Q \equiv A_{2,x} + A_2A_3 - A_5. \quad (35)$$

The following two subcases may occur.

- a) If $P = 0$ and $Q = 0$, a factor $\partial_y + c$ exists if c is a solution of $c_y - c^2 + A_2c - A_4 = 0$.*

- b) If $Q \neq 0$ and $P_yQ - P(Q_y + P) + A_2PQ - A_4Q^2 = 0$ there is the factor $\partial_y + \frac{A_{4,x} + A_3A_4 - A_6}{A_{2,x} + A_2A_3 - A_5}$.*

A complete listing of all possible Loewy decompositions has to include the divisors of order 2; they are considered next.

PROPOSITION 5.2. *Let the third order partial differential operator (33) be given with $A_k \in \mathbb{Q}(x, y)$ for all k . The following second order right factors may occur.*

- i) A factor $\partial_{yy} + a_1\partial_x + a_2\partial_y + a_3$ exists if there holds*

$$A_5 = A_{2,x} + A_2A_3,$$

$$A_6 = A_{4,x} - A_{3,x}A_1 - 2A_3A_{1,x} - A_{1,xx} + A_3(A_4 - A_1A_3).$$

Then $a_1 = A_1$, $a_2 = A_2$ and $a_3 = A_4 - A_1A_3 - A_{1,x}$.

- ii) A factor with leading derivative ∂_{xy} does not contain a term with ∂_{yy} . A factor $\partial_{xy} + a_1\partial_x + a_2\partial_y + a_3$ exists if one of the following two subcases applies.*

- a) There holds $A_1 = 0$, $A_5 - 2A_{3,y} - A_2A_3 = 0$ and $A_6 + A_{3,yy} + A_{2,y}A_3 - A_{5,y} - A_3A_4 = 0$. The coefficient a_1 may be determined from the Riccati equation*

$$a_{1,y} - a_1^2 + A_2a_1 - A_4 = 0. \quad (36)$$

The remaining coefficients are $a_2 = A_3$ and $a_3 = A_3a_1 + A_{3,y}$.

- b) There holds $A_1 = 0$ and $A_5 \neq 2A_{3,y} + A_2A_3$. Then $a_2 = A_3$, $a_3 = A_3a_1 - A_{3,y} - A_2A_3 + A_5$,*

$$a_1 = A_2 + \frac{A_{3,yy} + A_{2,y}A_3 - A_{5,y} + A_6 - A_3A_4}{2A_{3,y} + A_2A_3 - A_5}.$$

Substituting a_1 into $a_{1,y} - a_1^2 + A_2a_1 - A_4 = 0$ yields a constraint for the coefficients $A_1, \dots, A_5$.

Similar as for operators (30), the decompositions of (33) may be determined algorithmically to a large extent as is shown next.

COROLLARY 5.3. *Any principal divisor of an operator (33) may be determined algorithmically. The same is true for a possible Laplace divisor of a given order.*

$\mathcal{L}_{xyy}^{(1)}$	$\mathcal{L}_{xyy}^{(1,1)} : l_{xyy}, \mathcal{L}_{xyy}^{(1,2)} : Lclm(l_x, l_y^{(1)}, l_y^{(2)}), \mathcal{L}_{xyy}^{(1,3)} : Lclm(l_{yy}, l_x),$ $\mathcal{L}_{xyy}^{(1,4)} : Lclm(l_{xy}, l_y), \mathcal{L}_{xyy}^{(1,5)} : Lclm(\mathcal{J}_4^{(1,1)}(m), \mathcal{J}_4^{(1,2)}(n)).$
$\mathcal{L}_{xyy}^{(2)}$	$\mathcal{L}_{xyy}^{(2,1)} : l_{yy}l_x, \mathcal{L}_{xyy}^{(2,2)} : Lclm(l_y^{(1)}, l_y^{(2)})l_x, \mathcal{L}_{xyy}^{(2,3)} : Lclm(l_y(\Phi))l_x.$
$\mathcal{L}_{xyy}^{(3)}$	$\mathcal{L}_{xyy}^{(3,1)} : l_{xy}l_y, \mathcal{L}_{xyy}^{(3,2)} : Lclm(l_x, l_y^{(1)})l_y^{(2)}, \mathcal{L}_{xyy}^{(3,3)} : Lclm(\mathcal{J}_2^{(1,1)}(m), \mathcal{J}_3^{(1,1)}(n))l_y.$
$\mathcal{L}_{xyy}^{(4)}$	$\mathcal{L}_{xyy}^{(4,1)} : l_xl_{yy}, \mathcal{L}_{xyy}^{(4,2)} : l_xLclm(l_y^{(1)}, l_y^{(2)}), \mathcal{L}_{xyy}^{(4,3)} : l_xLclm(l_y(\Phi)).$
$\mathcal{L}_{xyy}^{(5)}$	$\mathcal{L}_{xyy}^{(5,1)} : l_y l_{xy}, \mathcal{L}_{xyy}^{(5,2)} : l_y^{(1)} Lclm(l_x, l_y^{(1)}).$
$\mathcal{L}_{xyy}^{(6,7,8)}$	$\mathcal{L}_{xyy}^{(6,1)} : l_x l_y^{(1)} l_y^{(2)}, \mathcal{L}_{xyy}^{(7,1)} : l_y^{(1)} l_y^{(2)} l_x, \mathcal{L}_{xyy}^{(8,1)} : l_y^{(1)} l_x l_y^{(2)}.$
$\mathcal{L}_{xyy}^{(9)}$	$\mathcal{L}_{xyy}^{(9,1)} : Syz(\langle D_{xyy} \rangle, \langle D_{xyy}, D_x^m \rangle) \langle D_{xyy}, (\partial_{x^{m-1}} + \dots)(\partial_x + A_2) \rangle,$
$\mathcal{L}_{xyy}^{(10)}$	$\mathcal{L}_{xyy}^{(10,1)} : Syz(\langle D_{xyy} \rangle, \langle D_{xyy}, D_y^n \rangle) \langle D_{xyy}, (\partial_{y^{n-1}} + \dots)(\partial_y + A_2) \rangle.$
$\mathcal{L}_{xyy}^{(11)}$	$\mathcal{L}_{xyy}^{(11,1)} : Syz(\langle \partial_{xyy} \rangle, \mathcal{J}_6^{(1,2)}) Lclm((\partial_{xy} + \dots)(\partial_x + \dots), (\partial_{xy} + \dots)(\partial_y + \dots)).$

Table 4. The decomposition types of Corollary 5.5

The proof is almost identical to that of Corollary 4.3 and is therefore omitted.

A full listing of maximal completely reducible components may be obtained from the preceding results. Any operator (33) decomposes uniquely according to one of the following decomposition schemes.

THEOREM 5.4. *Any differential operator*

$$L \equiv \partial_{xyy} + A_1 \partial_{xx} + A_2 \partial_{xy} + A_3 \partial_{yy} + A_4 \partial_x + A_5 \quad (37)$$

decomposes uniquely into largest completely reducible components corresponding to one of the types $\mathcal{L}_{xyy}^{(k)}$, $k = 1, \dots, 11$, defined as follows; L_{xyy} , L_{xy} , L_{xx} , L_x and L_y are completely reducible operators with leading derivatives ∂_{xyy} , ∂_{xy} , ∂_{xx} , ∂_x or ∂_y respectively.

$$\begin{aligned} \mathcal{L}_{xyy}^{(1)} : L_{xyy}, \quad \mathcal{L}_{xyy}^{(2)} : L_{yy}L_x, \quad \mathcal{L}_{xyy}^{(3)} : L_{xy}L_y, \\ \mathcal{L}_{xyy}^{(4)} : L_xL_{yy}, \quad \mathcal{L}_{xyy}^{(5)} : L_yL_{xy}, \\ \mathcal{L}_{xyy}^{(6)} : L_xL_y^{(1)}L_y^{(2)}, \quad \mathcal{L}_{xyy}^{(7)} : L_y^{(1)}L_xL_y^{(2)}, \quad \mathcal{L}_{xyy}^{(8)} : L_y^{(1)}L_y^{(2)}L_x, \\ \mathcal{L}_{xyy}^{(9)} : Syz(\langle \partial_{xyy} \rangle, \mathcal{J}_4^{(1,1)}(m))\mathcal{J}_4^{(1,1)}(m) \\ \mathcal{L}_{xyy}^{(10)} : Syz(\langle L \rangle, \mathcal{J}_4^{(1,2)}(n))\mathcal{J}_4^{(1,2)}(n). \\ \mathcal{L}_{xyy}^{(11)} : Syz(\langle \partial_{xyy} \rangle, \mathcal{J}_6^{(1,2)})\mathcal{J}_6^{(1,2)}. \end{aligned}$$

The decomposition into irreducible components leads to the following refined decomposition scheme.

COROLLARY 5.5. *Let l_x , l_y , l_{xx} , l_{xy} and l_{xyy} denote irreducible operators with leading derivatives as determined by the respective subindex. An additional upper index distinguishes different copies of the respective operator; Φ is an undetermined function of a single argument. The types $\mathcal{L}_{xyy}^{(i,j)}$ defined in Table 4 are refinements of the types $\mathcal{L}_{xyy}^{(i)}$, $i = 1, \dots, 10$ defined in Theorem 5.4.*

This is an immediate consequence of the proof given for the preceding theorem if the various factorization alternatives of Proposition 3.1 and Proposition 3.2 are not merged into completely reducible components.

EXAMPLE 5.6. Consider the operator

$$\begin{aligned} L \equiv \partial_{xyy} + 2\left(x + \frac{1}{y}\right)\partial_{xy} + y\partial_{yy} \\ + x\left(x + \frac{2}{y}\right)\partial_x + 2(xy + 2)\partial_y + x^2 + 4x + \frac{2}{y}. \end{aligned}$$

Its coefficients satisfy conditions (34) of Proposition 5.1; consequently there is a factor $\partial_y + x$. Furthermore there holds $P = 0$ and $Q = 0$ for P and Q defined by (35). The Riccati equation for c is

$$c_y - c^2 + 2\left(x + \frac{1}{y}\right)c - x\left(x + \frac{2}{y}\right) = 0$$

with the rational solutions $c = x$ and $c = x + \frac{1}{y}$; they yield the factors $\partial_y + x$ and $\partial_y + x + \frac{1}{y}$ with the type $\mathcal{L}_{xyy}^{(1,2)}$ decomposition $L = Lclm(\partial_x + y, \partial_y + x, \partial_y + x + \frac{1}{y})$. The general solution of $Lz = 0$ is obtained by three integrations as

$$z = [F(x) + G(x)\frac{1}{y} + H(y)] \exp(-xy).$$

F , G and H are undetermined functions.

EXAMPLE 5.7. Consider the operator

$$L \equiv \partial_{xyy} + \left(\frac{1}{x} - 2\right)\partial_{xy} - \partial_{yy} - \frac{1}{x}\partial_x - \left(\frac{1}{x^2} + \frac{1}{x} - 2\right)\partial_y + \frac{2}{x^2} + \frac{2}{x}.$$

There holds $A_1 = 0$ and $P = Q = 0$, i. e. case ii), subcase a) of Proposition 5.1 applies. The equation

$$c_y - c^2 + \left(\frac{1}{x} - 2\right)c + \frac{1}{x} = 0$$

has the general solution $c = \frac{y \exp(\frac{1}{2}y^2 + y) - C}{\exp(\frac{1}{2}y^2 + y) + C}$ where C

is a constant. For $C = 0$ and $C \rightarrow \infty$ the two rational solutions $c = y$ and $c = -1$ are obtained. They yield the factors $\partial_y + y$ and $\partial_y - 1$ and the type $\mathcal{L}_{xyy}^{(4,2)}$ decomposition

$$\begin{aligned} L = (\partial_x - x)Lclm(\partial_y + y, \partial_y + 1) = \\ (\partial_x - x)\left(\partial_{yy} + \frac{y^2 - 2}{y + 1}\partial_y - \frac{y^2 + y - 1}{y + 1}\right). \end{aligned}$$

The general solution of $Lz = 0$ is

$$\begin{aligned} z = F(x) \exp\left(-\frac{1}{2}y^2\right) + G(x) \exp(-y) + \exp\left(\frac{1}{2}x^2 - y\right) \\ \times \int H(y) e^y \frac{dy}{y + 1} - \exp\left(\frac{1}{2}x^2 - \frac{1}{2}y^2\right) \int H(y) \exp\left(\frac{1}{2}y^2\right) \frac{dy}{y + 1} \end{aligned}$$

F , G and H are undetermined functions.

EXAMPLE 5.8. Consider the operator

$$L \equiv \partial_{xyy} + \left(x + 1 + \frac{1}{y}\right)\partial_{xy} + y\partial_{yy} \\ + \left(x + \frac{1}{y} - \frac{1}{y^2}\right)\partial_x + (xy + y + 3)\partial_y + xy + x + 2.$$

By case *i*) and case *ii*), subcase *b*) of Proposition 5.1 the type $\mathcal{L}_{xyy}^{(5,2)}$ decomposition

$$L = (\partial_y + 1)Lclm\left(\partial_x + y, \partial_y + x + \frac{1}{y}\right) \\ = (\partial_y + 1)\left(\partial_{xy} + \left(x + \frac{1}{y}\right)\partial_x + y\partial_y + xy + 2\right)$$

is obtained. The two arguments of the $Lclm$ yield the solutions $z_1 = F(y) \exp(-xy)$ and $z_2 = G(x) \frac{1}{y} \exp(-xy)$; F and G are undetermined functions. L may be factorized as $L = (\partial_y + 1)(\partial_y + x + \frac{1}{y})(\partial_x + y)$ from which the third solution

$$z_3 = \exp\left(-\left(x + 1\right)y\right) \frac{1}{y} \int H(x) \frac{xy - y - 1}{(x - 1)^2} \exp(xy) dx$$

follows; H is again an undetermined function. Finally the general solution of $Lz = 0$ is $z = z_1 + z_2 + z_3$.

EXAMPLE 5.9. Consider the operator

$$L \equiv \partial_{xyy} - 2x\partial_{xy} + y\partial_{yy} + x^2\partial_x - 2(xy - 1)\partial_y + x(xy - 2).$$

By case *i*) and case *ii*), subcase *b*) of Proposition 5.1 the factorizations

$$L = \begin{cases} (\partial_{yy} - 2x\partial_y + x^2)(l_1 \equiv \partial_x + y) \\ (\partial_{xy} - x\partial_x + y\partial_y - xy + 3)(l_2 \equiv \partial_y - x). \end{cases}$$

are obtained. The intersection of l_1 and l_2 is

$$Lclm(l_1, l_2) = \langle \partial_{xyy} - 2x\partial_{xy} + y\partial_{yy} + x^2\partial_x - 2(xy - 1)\partial_y + x(xy - 2), \\ \partial_{xxy} - x\partial_{xx} + 2y\partial_{xy} - 2(xy + 1)\partial_x + y^2\partial_y - y(xy + 2) \rangle.$$

It is not principal, i.e. L has decomposition type $\mathcal{L}_{xyy}^{(11)}$. The two equations $l_1z = 0$ and $l_2z = 0$ yield the solutions $z_1 = F(y) \exp(-xy)$ and $z_2 = G(x) \exp(xy)$. A third solution may be obtained from the factorization $\partial_{yy} - 2x\partial_y + x^2 = (\partial_y - x)(\partial_y - x)$. It yields

$$z_3 = y \exp(-xy) \int xH(x) \exp(2xy) dx.$$

The general solution of $Lz = 0$ is $z = z_1 + z_2 + z_3$.

6. SUMMARY AND CONCLUSION

In this article for the first time Loewy decompositions for a full class of partial differential equations of differential type different from zero are considered. It became clear that Theorems 2.1 and 2.2 are fundamental for a full understanding of how the various decompositions arise. Furthermore it has been shown by numerous examples that obtaining such a decomposition is essentially synonymous with finding its solutions in closed form. However, two basic questions remain to be discussed.

In the first place this is the uniqueness of a decomposition as it is the case for ordinary operators. To this end, one probably has to make the problem more specific by the requirement that the admitted divisors have differential type 1, as it is true for the given operators.

Secondly there is the question to what extent these decompositions may be obtained algorithmically. For ordinary operators it is sufficient to determine rational solutions of ordinary Riccati equations. It has been shown that for decomposing linear pde's of the plane in general rational solutions of partial Riccati equations are required. This problem boils down to finding rational first integrals of a general quasilinear first-order ode; for this problem see [4] and [3]. Up to this point the discussion concerns principal divisors. Deciding the non-existence of a Laplace divisor requires an upper bound for its order. It may not be possible to find such a bound, i.e. this problem may not be decidable. Finally there remain the exceptional cases mentioned at the end of Corollary 3.3.

Obviously there are many possible extensions of the results presented in this article. To mention just a few, there should be a close connection between the type of solution and the decomposition type as it is true for ordinary differential equation, Section 2.1 of [12]. For applications decompositions in more than two independent variables would be highly desirable. Dealing with more than a single dependent variables would involve modules over the respective rings of differential operators.

7. REFERENCES

- [1] H. Blumberg, *Über algebraische Eigenschaften von linearen homogenen Differentialausdrücken*, Inaugural-Dissertation, Göttingen, 1912.
- [2] A. Buium, Ph. Cassidy, *Differential Algebraic Geometry and Differential Algebraic Groups: From Algebraic Differential Equations to Diophantine Geometry*, in: Selected Works of Ellis Kolchin, AMS Press, 1999; H. Bass, A. Buium, Ph. Cassidy, Eds.
- [3] G. Chen, Y. Ma, *Algorithmic Reduction and Rational General Solutions of First-Order Algebraic Differential Equations*, in: Differential Equations with Symbolic Computation, D. Wang and Z. Zhen, eds, Birkhäuser Verlag, Basel, 2005.
- [4] A. Eremenko, *Rational Solutions of First-Order Differential Equations*, Ann. Acad. Scient. Fennicae Math. **23**, 181-190 (1998).
- [5] A. R. Forsyth, *Theory of Differential Equations*, vol. I,...,VI, Cambridge, At the University Press (1906).
- [6] E. Goursat, *Leçon sur l'intégration des équation aux dérivées partielles*, I and II, A. Hermann, Paris 1898.
- [7] D. Grigoriev, F. Schwarz, *Factoring and solving linear partial differential equations*, Computing **73**, 179-197 (2004).
- [8] D. Grigoriev, F. Schwarz, *Generalized Loewy Decomposition of D-Modules*, Proceedings of the ISSAC'05, 163-170, ACM Press, 2005, Manuel Kauers, ed.
- [9] E. Kolchin, *Differential Algebra and Algebraic Groups*, Academic Press, 1973.
- [10] A. Loewy, *Über vollständig reduzible lineare homogene Differentialgleichungen*, Mathematische Annalen, **56**, 89-117 (1906).
- [11] A. Quadrat, *An introduction to the algebraic theory of linear systems of partial differential equations*, rapport INRIA, 2008.
- [12] F. Schwarz, *Algorithmic Lie Theory for Solving Ordinary Differential Equations*, Chapman & Hall/CRC 2007.

The Nearest Polynomial with a Zero in a Given Domain from a Geometrical Viewpoint

Hiroshi Sekigawa

NTT Communication Science Laboratories
Nippon Telegraph and Telephone Corporation
3-1 Morinosato-Wakamiya, Atsugi-shi,
Kanagawa 243-0198, Japan
sekigawa@theory.brl.ntt.co.jp

ABSTRACT

For a real univariate polynomial f and a closed complex domain D , whose boundary C is a simple curve parameterized by a univariate piecewise rational function, a rigorous method is given for finding a real univariate polynomial $\tilde{f}$ such that $\tilde{f}$ has a zero in D and $\|f - \tilde{f}\|_\infty$ is minimal. First, it is proved that the minimum distance between f and polynomials having a zero at $\alpha \in \mathbb{C}$ is a piecewise rational function of the real and imaginary parts of α . Thus, on C , the minimum distance is a piecewise rational function of a parameter obtained through the parameterization of C . Therefore, by using the property that $\tilde{f}$ has a zero on C and computing the minimum distance on C , $\tilde{f}$ can be constructed.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms—Algebraic algorithms

General Terms

Algorithms, Theory

Keywords

Polynomial, zero, perturbation, l^∞ -norm, Davenport-Schinzel sequence

1. INTRODUCTION

Problems on the locations of the zeros of a polynomial are very interesting and important in both theory and practice. For example, in control theory, the locations of the zeros of polynomials affect the stability and performance of systems. In practical examples, the coefficients of polynomials can contain errors, because they are obtained through measurement or can be specified with only finite precision. As a result, control theory has considered problems on how perturbations of such coefficients affect the locations of zeros

and the properties of systems. Kharitonov's theorem [10] and the edge theorem [2] are landmark results in the vast literature on such problems.

There are many studies on the locations of the zeros of a polynomial in research on symbolic-numeric algorithms [5, 6, 7, 8, 11, 14, 15, 16, 17, 19]. For a given complex polynomial f and a domain D , reference [6] proposes a method for finding a nearest polynomial to f in the l^2 -norm with a zero in D . For a given real polynomial g and a complex number α , reference [5] provides an explicit formula for a nearest polynomial to g in the l^2 -norm with a zero at α . Reference [9] gives an explicit formula for the distance between g and a nearest polynomial in the l^∞ -norm.

In this paper, we consider the following problem.

Given a real univariate polynomial f and a closed domain $D \subseteq \mathbb{C}$, find a nearest real univariate polynomial $\tilde{f}$ such that $\tilde{f}$ has a zero in D and the l^∞ -norm of $f - \tilde{f}$ is minimal.

This problem is closely related to problems treated in references [12, 13]. (See also a textbook, reference [3].) Here, we reduce the problem of searching in a subset of a given domain [12, 13] to the solution of algebraic equations.

Although we previously examined this problem [15] and proposed a method for finding $\tilde{f}$, the number of equations to be solved is of exponential order in the size of the input. In contrast, in the method described in this paper, the number of equations and their degrees are of polynomial order in the size of the input.

The rest of this paper is organized as follows. Section 2 explains the theorems supporting the proposed method, whose details and computational complexity are given in Section 3. Section 4 shows an example of applying the method, and Section 5 concludes the paper by mentioning directions for future work.

2. THEORETICAL BACKGROUND

2.1 Existence of a Nearest Polynomial

Suppose that we are given real univariate polynomials $f(x)$, $e_1(x)$, $\dots$, $e_n(x)$ and a closed set $D \subset \mathbb{C}$ satisfying the following conditions.

Condition 1.

1. The number of e_j is finite and $\text{span}\{e_1(x), \dots, e_n(x)\} \cong \mathbb{R}^n$; that is, $\{e_1(x), \dots, e_n(x)\}$ is a basis of the real vector space $\text{span}\{e_1(x), \dots, e_n(x)\}$.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

2. The polynomial f does not have a zero in D .
3. Let

$$F = \left\{ f(x) + \sum_{j=1}^n c_j e_j(x) \mid c_j \in \mathbb{R} \right\}.$$

The degrees of the polynomials in F are constant when D is not bounded.

Then, consider the following problem.

Problem 1. Find $\tilde{f} = f + \sum_{j=1}^n a_j e_j \in F$ such that $\tilde{f}$ has a zero in D and $\|f - \tilde{f}\|_\infty = \max\{|a_1|, \dots, |a_n|\}$ is minimal.

Remark 1. Finding a nearest polynomial in a weighted l^∞ -norm is reduced to Problem 1. Let $w_j > 0$ for $j = 1, \dots, n$. Since the equations $\tilde{f} = f + \sum_{j=1}^n a_j e_j = f + \sum_{j=1}^n w_j a_j e_j / w_j$ hold, the weighted l^∞ -norm of $f - \tilde{f}$ with respect to the basis $\{e_1(x), \dots, e_n(x)\}$ and the weights w_j —that is, $\max_j \{w_j |a_j|\}$ —is equal to the l^∞ -norm of $f - \tilde{f}$ with respect to the basis $\{e_1(x)/w_1, \dots, e_n(x)/w_n\}$.

Clearly, there is no solution to Problem 1 if there is no polynomial in F having a zero in D .

Example 1. Let $F = \{(x+1) + c(-x+1) \mid c \in \mathbb{R}\}$, and let $D \subset \mathbb{C}$ be a closed set that is disjoint from $\mathbb{R}$. Then, there is no polynomial in F having a zero in D .

Let $g(x) = (x+1) + b(-x+1)$ ($b \in \mathbb{R}$). If $b = 1$, then $g(x) = 2$ and $g(x)$ has no zero. Otherwise, $g(x)$ is of degree one, and the zero of $g(x)$ is a real number.

There is a solution to Problem 1 if there is a polynomial in F having a zero in D .

THEOREM 1. Define F , f , and D as above. If there is a polynomial $g \in F$ having a zero in D , then there is a polynomial $\tilde{f} \in F$ such that $\tilde{f}$ has a zero in D and $\|f - \tilde{f}\|_\infty$ is minimal.

PROOF. For the case when D is bounded, see the proof of Theorem 1 in [15].

When D is not bounded, the leading coefficients of polynomials in F are equal, from the assumption that the degrees of the polynomials in F are constant. To find a nearest polynomial, it is sufficient to investigate polynomials h satisfying $\|f - h\|_\infty \leq \|f - g\|_\infty$. There exists a positive number R such that all of the zeros of these polynomials belong to a closed disk with center 0 and radius R . Therefore, we can replace D by the closed bounded set $D \cap \{z \in \mathbb{C} \mid |z| \leq R\}$. $\square$

Remark 2. If (1) the number of e_j is not finite, or if (2) D is not bounded and the degrees of the polynomials in F are not constant, then Theorem 1 does not hold.

1. If the number of e_j is not finite, then we have the following counterexample. Let $f(x) = x$, $e_j(x) = x^j$ ($1 \leq j$), and let

$$F = \left\{ f + \sum_{j=2}^{\infty} c_j e_j \mid c_j \in \mathbb{R}, \#\{c_j \mid c_j \neq 0\} < \infty \right\}.$$

That is, F is the set of all real univariate polynomials without constant terms. Let $D = \{1\}$. Then, the polynomial

$$f_m(x) = x - \frac{1}{m}(x^2 + \dots + x^{m+1})$$

belongs to F , and $f(1) \neq 0$, $f_m(1) = 0$, and $\|f - f_m\|_\infty = 1/m$ hold. Therefore, there is no solution to Problem 1.

2. If D is not bounded and the degrees of the polynomials in F are not constant, then we have the following counterexample. Let $f(x) = 1$, $F = \{f(x) + cx \mid c \in \mathbb{R}\}$, and $D = \{z \in \mathbb{C} \mid \operatorname{Re} z \leq 0\}$, the left half-plane in $\mathbb{C}$. Then, the polynomial

$$f_\epsilon(x) = 1 + \epsilon x \quad (0 < \epsilon)$$

belongs to F , and $f_\epsilon(x)$ has a zero at $-1/\epsilon \in D$. Since $\|f - f_\epsilon\|_\infty = \epsilon$ and $f(x)$ has no zero in D , there is no solution to Problem 1.

Remark 3. In this situation, $\deg(g)$ might not be constant for $g \in F$. For example, when there is a number j such that $\deg(e_j) > \deg(f)$, $\deg(g)$ is not constant.

If we require $\deg(g)$ to be constant, then there might be no solution to Problem 1. Let $f(x) = 1 + x + x^2$, $F = \{f(x) + c_1 + c_2x + c_3x^2 \mid c_j \in \mathbb{R}\}$, and $D = \{1/2\}$. If we require the degree $\tilde{f} \in F$ to be 2, then there is no polynomial $\tilde{f}$ such that $\|f - \tilde{f}\|_\infty$ is minimal. For a proof of this point, see page 77 in [14].

2.2 Main Theorems

Hereafter, we further assume that D satisfies the following condition.

Condition 2. $D \subset \mathbb{C}$ is a closed domain whose boundary C is a simple curve.

We denote the open disk centered at $z \in \mathbb{C}$ with radius r as $B(z; r)$, and the set of interior points of $X \subset \mathbb{C}$, that is,

$$\{z \in X \mid \exists r > 0 \text{ such that } B(z; r) \subset X\},$$

as X° .

The following proposition shows that a nonzero polynomial that attains a minimum has no zero in D° .

PROPOSITION 1. Define f , e_j ($j = 1, \dots, n$), and D as above. We further assume that f has no zero in D , and that the nonzero polynomial

$$g(x) = f(x) + \sum_{j=1}^n b_j e_j(x) \quad (b_j \in \mathbb{R})$$

has a zero in D° . Then, there exists a nonzero polynomial

$$\tilde{g}(x) = f(x) + \sum_{j=1}^n \tilde{b}_j e_j(x) \quad (\tilde{b}_j \in \mathbb{R})$$

such that $\tilde{g}$ has a zero in D° and $\|f - \tilde{g}\|_\infty < \|f - g\|_\infty$ holds.

PROOF. See the proof of Lemma 1 in [15]. $\square$

The following is a sufficient condition for the existence of a polynomial in F having a zero in D , which is equivalent to the existence of a nearest polynomial.

THEOREM 2. Define f , e_j ($j = 1, \dots, n$), and D as above. When $n \geq 2$, there is a polynomial in F having a zero in D .

PROOF. First, we prove the case when $n = 2$. Since both e_1 and e_2 are not constant, by renumbering the indexes, if necessary, we assume that e_1 is not constant. Therefore, there is a nonempty open set $U \subset D$ such that $0 \notin e_1(U)$. Then, there is $\alpha \in U$ such that $e_2(\alpha)/e_1(\alpha) \notin \mathbb{R}$, since the image of U under the holomorphic function e_2/e_1 in U is open. Hence, $e_1(\alpha)$ and $e_2(\alpha)$ are linearly independent over $\mathbb{R}$. Therefore, there exist $c_1, c_2 \in \mathbb{R}$ such that $-f(\alpha) = c_1 e_1(\alpha) + c_2 e_2(\alpha)$; that is, $f(\alpha) + c_1 e_1(\alpha) + c_2 e_2(\alpha) = 0$.

When $n > 2$, set

$$G = f(x) + \text{span}\{e_1(x), e_2(x)\}.$$

Then, G is a subset of F , and there is a polynomial in G having a zero in D . Therefore, the statement of the theorem holds. $\square$

Remark 4. When $n = 1$ and f and e_1 are linearly dependent over $\mathbb{R}$, the zero polynomial is the unique polynomial having a zero in D . This statement follows from the assumption that f has no zero in D and $e_1 = af$ for some $a \in \mathbb{R}$.

When $n = 1$ and f and e_1 are linearly independent over $\mathbb{R}$, the following are sufficient conditions for the existence of a polynomial in F having a zero in D .

- e_1 has a zero $\alpha \in D^\circ$.
- The image of C under e_1/f intersects $\mathbb{R}$ at a point other than 0.

When the first condition holds, take an open set $U \subset D$ such that $\alpha \in U$. Then, there is $\beta \in U$ such that $e_1(\beta)/f(\beta)$ is a nonzero real number c , since the image of U under e_1/f is an open set containing 0. The polynomial $f - e_1/c$ has a zero $\beta \in D$.

When D is bounded, the second condition is a necessary and sufficient condition.

When D is not bounded and the degrees of the polynomials in F are constant, we can decide the existence of a polynomial in F having a zero in D by using the computation method described in the next section.

LEMMA 1. Define f and e_j as above, and let

$$F_\epsilon(x) = \left\{ f(x) + \sum_{j=1}^n b_j e_j(x) \mid b_j \in \mathbb{R}, |b_j| \leq \epsilon \right\},$$

where $0 \leq \epsilon$. Then, the following is a necessary and sufficient condition for the existence of $\tilde{f} \in F_\epsilon(x)$ such that $\tilde{f}$ has a zero at $\alpha \in \mathbb{C}$.

1. When $0, e_1(\alpha), \dots, e_n(\alpha)$ lie on a straight line, the condition is that $f(\alpha)$ lies on the straight line and the following inequality holds.

$$|f(\alpha)| \leq \epsilon \sum_{j=1}^n |e_j(\alpha)|.$$

2. When $0, e_1(\alpha), \dots, e_n(\alpha)$ do not lie on a straight line, the condition is that the following inequalities hold for $j = 1, \dots, n$.

$$|A(e_j(\alpha), f(\alpha))| \leq \epsilon \sum_{k=1}^n |A(e_j(\alpha), e_k(\alpha))|,$$

where

$$A(u, v) = \begin{vmatrix} \text{Re } u & \text{Re } v \\ \text{Im } u & \text{Im } v \end{vmatrix}.$$

Remark 5. $A(u, v) = 0$ if and only if $0, u$, and v lie on a straight line. This is equivalent to u and v being linearly dependent over $\mathbb{R}$.

PROOF. There exists $g \in F_\epsilon(x)$ such that α is a zero of g if and only if 0 belongs to the set $F_\epsilon(\alpha) = \{h(\alpha) \mid h(x) \in F_\epsilon(x)\}$. We can write $F_\epsilon(\alpha)$ as $f(\alpha) + E_\epsilon(\alpha)$, where

$$E_\epsilon(\alpha) = \left\{ \sum_{j=1}^n b_j e_j(\alpha) \mid b_j \in \mathbb{R}, |b_j| \leq \epsilon \right\}.$$

Thus, 0 belongs to the set $F_\epsilon(\alpha)$ if and only if $-f(\alpha)$ belongs to the set $E_\epsilon(\alpha)$. This is equivalent to $f(\alpha) \in E_\epsilon(\alpha)$, since $E_\epsilon(\alpha)$ is symmetric about the origin.

Next, we prove that $E_\epsilon(\alpha)$ is a convex polygon. Let

$$\tilde{e}_j(x) = \begin{cases} -e_j(x), & \text{if } \text{Re } e_j(\alpha) < 0 \text{ or} \\ & \text{if } \text{Re } e_j(\alpha) = 0 \text{ and } \text{Im } e_j(\alpha) < 0, \\ e_j(x), & \text{otherwise.} \end{cases}$$

Then, $E_\epsilon(\alpha)$ can be written as $-\epsilon \sum_{j=1}^n \tilde{e}_j(\alpha) + V_\epsilon(\alpha)$, where

$$V_\epsilon(\alpha) = \left\{ \sum_{j=1}^n 2t_j \epsilon \tilde{e}_j(\alpha) \mid 0 \leq t_j \leq 1 \ (j = 1, 2, \dots, n) \right\}.$$

The set $V_\epsilon(\alpha)$ is a convex polygon (see Theorem 4 in [16]), and thus, so is $E_\epsilon(\alpha)$.

We can find the vertices of $V_\epsilon(\alpha)$ as follows. First, sort $\tilde{e}_j(\alpha) \neq 0$ as the arguments in increasing order. (We take $-\pi < \arg \tilde{e}_j(\alpha) \leq \pi$.) If two or more $\tilde{e}_j(\alpha)$, say $\tilde{e}_j(\alpha), \tilde{e}_k(\alpha), \tilde{e}_l(\alpha)$, have the same argument, then replace $\tilde{e}_j(\alpha), \tilde{e}_k(\alpha), \tilde{e}_l(\alpha)$ with $\tilde{e}_j(\alpha) + \tilde{e}_k(\alpha) + \tilde{e}_l(\alpha)$, and write the results of sorting as $p_1, p_2, \dots, p_{\tilde{n}}$. Then, the vertices of the convex polygon are $v_0, v_1, \dots, v_{2\tilde{n}-1}$, in counterclockwise order, where

$$v_j = \begin{cases} \sum_{k=1}^j 2\epsilon p_k & (j = 0, 1, \dots, \tilde{n} - 1), \\ \sum_{k=j-\tilde{n}+1}^{\tilde{n}} 2\epsilon p_k & (j = \tilde{n}, \dots, 2\tilde{n} - 1). \end{cases}$$

Therefore, the convex polygon $E_\epsilon(\alpha)$ has the vertices $\epsilon w_0, \epsilon w_1, \dots, \epsilon w_{2\tilde{n}-1}$, in counterclockwise order, where

$$w_j = \begin{cases} \sum_{k=1}^j p_k - \sum_{k=j+1}^{\tilde{n}} p_k & (j = 0, 1, \dots, \tilde{n} - 1), \\ -w_{j-\tilde{n}} & (j = \tilde{n}, \dots, 2\tilde{n} - 1). \end{cases}$$

Figure 1 shows an example of $e_j(\alpha), \tilde{e}_j(\alpha)$, and the corresponding convex polygons $E_\epsilon(\alpha)$ for $\epsilon = 0.5$ and 1 when $n = 3$.

Now, we prove Statement 1. When $0, e_1(\alpha), \dots, e_n(\alpha)$ lie on a straight line, the convex polygon $E_\epsilon(\alpha)$ degenerates to a line segment, and a necessary and sufficient condition for the existence of a nearest polynomial is that $f(\alpha)$ lies on the line segment. Suppose that there is a nearest polynomial. Then, $\sum_{j=1}^n \tilde{e}_j(\alpha) \neq 0$ holds, and the two endpoints of the line segment $E_\epsilon(\alpha)$ are $\pm \epsilon \sum_{j=1}^n \tilde{e}_j(\alpha)$. Thus, $E_\epsilon(\alpha)$ can be described as the set of points z belonging to the straight line passing through the two points $\pm \sum_{j=1}^n \tilde{e}_j(\alpha)$ and satisfying

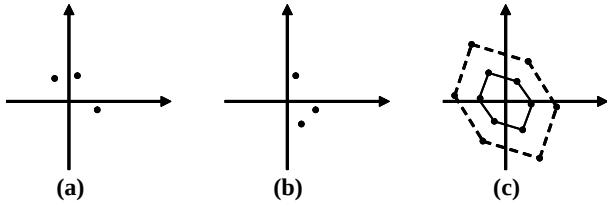


Figure 1: (a) $e_j(\alpha)$ ($j = 1, 2, 3$). (b) $\tilde{e}_j(\alpha)$ ($j = 1, 2, 3$). (c) Convex polygons $E_\epsilon(\alpha)$ for $\epsilon = 0.5$ (solid line) and $\epsilon = 1$ (dashed line).

the inequality $|z| \leq \epsilon \sum_{j=1}^n |\tilde{e}_j(\alpha)|$. Then, the first statement follows from the fact that $|\tilde{e}_j(\alpha)| = |e_j(\alpha)|$ ($j = 1, \dots, n$).

Next, we prove Statement 2. When the convex polygon $E_\epsilon(\alpha)$ does not degenerate to a line segment, for every $\tilde{e}_j(\alpha) \neq 0$ there exist exactly two edges parallel to $\tilde{e}_j(\alpha)$, and for every edge there exists at least one $\tilde{e}_j(\alpha)$ parallel to the edge. For $\tilde{e}_j(\alpha) \neq 0$, take the vertex ϵw_k such that the edge whose endpoints are ϵw_k and ϵw_{k+1} is parallel to $\tilde{e}_j(\alpha)$, and such that k is minimal, and write k as $\nu(j)$. Then, the equations defining the two straight lines containing the two edges parallel to $\tilde{e}_j(\alpha)$ are

$$A(\tilde{e}_j(\alpha), x - \epsilon w_{\nu(j)}) = 0, \quad A(\tilde{e}_j(\alpha), x + \epsilon w_{\nu(j)}) = 0. \quad (1)$$

Equations (1) are equivalent to

$$A(\tilde{e}_j(\alpha), x) = \pm \epsilon A(\tilde{e}_j(\alpha), w_{\nu(j)}).$$

Since 0 belongs to the convex polygon $E_\epsilon(\alpha)$, it can be described as the set of points z satisfying the inequalities

$$|A(\tilde{e}_j(\alpha), z)| \leq \epsilon |A(\tilde{e}_j(\alpha), w_{\nu(j)})| \quad (j = 1, \dots, n).$$

Thus, there exists $\tilde{f} \in F_\epsilon(x)$ such that $\tilde{f}$ has a zero at α if and only if the following inequalities hold.

$$|A(\tilde{e}_j(\alpha), f(\alpha))| \leq \epsilon |A(\tilde{e}_j(\alpha), w_{\nu(j)})| \quad (j = 1, \dots, n).$$

Note that the following equality holds.

$$w_{\nu(j)} = \sum_{k \in L(j)} \tilde{e}_k(\alpha) - \sum_{k \in R(j)} \tilde{e}_k(\alpha),$$

where $L(j) = \{k \mid A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha)) < 0\}$ and $R(j) = \{k \mid A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha)) > 0\}$. Thus, we have

$$\begin{aligned} & A(\tilde{e}_j(\alpha), w_{\nu(j)}) \\ &= \sum_{k \in L(j)} A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha)) - \sum_{k \in R(j)} A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha)) \\ &= - \sum_{k \in L(j)} |A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha))| - \sum_{k \in R(j)} |A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha))| \\ &= - \sum_{k=1}^n |A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha))|. \end{aligned}$$

Note that $A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha)) = 0$ holds for $k \notin L(j) \cup R(j)$. Therefore, the inequality

$$|A(\tilde{e}_j(\alpha), f(\alpha))| \leq \epsilon \sum_{k=1}^n |A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha))|$$

holds. Finally, by noting that

$$\begin{aligned} |A(\tilde{e}_j(\alpha), f(\alpha))| &= |A(e_j(\alpha), f(\alpha))|, \\ |A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha))| &= |A(e_j(\alpha), e_k(\alpha))|, \end{aligned}$$

the proof is complete. $\square$

The next theorem follows from Lemma 1. To avoid square roots, we modify the formula for the degenerate case in Lemma 1.

THEOREM 3. Define f , e_j , and F as above.

1. When $A(e_j(\alpha), e_k(\alpha)) = 0$ holds for $1 \leq j < k \leq n$, a necessary and sufficient condition for the existence of a nearest polynomial is that $A(e_j(\alpha), f(\alpha)) = 0$ holds for $j = 1, \dots, n$ and there is j such that $e_j(\alpha) \neq 0$. The minimal distance is

$$\frac{A(f(\alpha), if(\alpha))}{\sum_{j=1}^n |A(e_j(\alpha), if(\alpha))|}.$$

2. When $A(e_j(\alpha), e_k(\alpha)) \neq 0$ holds for some $j < k$, a nearest polynomial exists, and the minimal distance is

$$\max_{j=1, \dots, n} \left\{ \frac{|A(e_j(\alpha), f(\alpha))|}{\sum_{k=1}^n |A(e_j(\alpha), e_k(\alpha))|} \right\},$$

where the elements whose denominators are 0 are omitted.

PROOF. Since the second statement immediately follows from Lemma 1, we prove only the first statement here.

When $0, e_1(\alpha), \dots, e_n(\alpha)$ lie on a straight line and the condition for the existence of a nearest polynomial holds, we can write $e_j(\alpha) = a_j f(\alpha)$ ($a_j \in \mathbb{R}$) for $j = 1, \dots, n$. Then, the first condition in Lemma 1, that $|f(\alpha)| \leq \epsilon \sum_{j=1}^n |e_j(\alpha)|$, can be written as $|f(\alpha)| \leq \epsilon |f(\alpha)| \sum_{j=1}^n |a_j|$. Since $f(\alpha) \neq 0$ holds, the minimum ϵ is $1/\sum_{j=1}^n |a_j|$.

On the other hand,

$$A(e_j(\alpha), if(\alpha)) = A(a_j f(\alpha), if(\alpha)) = a_j A(f(\alpha), if(\alpha)).$$

Noting that $A(f(\alpha), if(\alpha)) = |f(\alpha)|^2 > 0$, we have

$$\frac{A(f(\alpha), if(\alpha))}{\sum_{j=1}^n |A(e_j(\alpha), if(\alpha))|} = \frac{1}{\sum_{j=1}^n |a_j|},$$

which proves the first statement. $\square$

When D consists of one point, methods for computing the minimum distance and finding a nearest polynomial in the l^2 -norm are as described in [5]. In [9], a formula for the minimum distance in the l^∞ -norm is given, with a proof different from the above proof. Here, we provide this proof because we use it to construct a nearest polynomial.

3. COMPUTATION METHODS

In this section, we further assume the following condition.

Condition 3. $C = \cup_{\mu=1}^M C_\mu$ ($M < \infty$) and $C_\mu \cap C_\nu = \emptyset$ ($\mu \neq \nu$), where each C_μ is parameterized by $\gamma_\mu \in \mathbb{C}(s)$ as

$$\gamma_\mu : I_\mu \longrightarrow C_\mu.$$

Here, $I_\mu \subset \mathbb{R}$ is of type $[a, b]$, $[a, \infty)$, $(-\infty, b]$, or $\mathbb{R}$.

For a polynomial $g(x) \in \mathbb{C}[x]$, we denote by $\bar{g}(x)$ the polynomial whose coefficients are the complex conjugates of the coefficients of $g(x)$. For a rational function $h(x) = h_1(x)/h_2(x)$ ($h_1(x), h_2(x) \in \mathbb{C}[x]$), we denote by $\bar{h}(x)$ the rational function $\bar{h}_1(x)/\bar{h}_2(x)$. For $g \in \mathbb{R}[x]$, set

$$\begin{aligned} g_{\mu,r}(s) &= \frac{g(\gamma_\mu(s)) + g(\bar{\gamma}_\mu(s))}{2}, \\ g_{\mu,i}(s) &= \frac{g(\gamma_\mu(s)) - g(\bar{\gamma}_\mu(s))}{2i}. \end{aligned}$$

Then, $g_{\mu,r}(s), g_{\mu,i}(s) \in \mathbb{R}(s)$, and for $a \in \mathbb{R}$, we have

$$g_{\mu,r}(a) = \operatorname{Re} g(\gamma_\mu(a)), \quad g_{\mu,i}(a) = \operatorname{Im} g(\gamma_\mu(a)).$$

Therefore, we write $g_{\mu,r}(s)$ and $g_{\mu,i}(s)$ as $\operatorname{Re} g(\gamma_\mu(s))$ and $\operatorname{Im} g(\gamma_\mu(s))$, respectively.

From Proposition 1, it is sufficient to investigate zeros on C . We show here that we can decide whether there is a nearest polynomial $\tilde{f}$, we can compute the minimum distance and a zero of f on C , and we can construct $\tilde{f}$ when it exists.

In the following, we do not assume that numbers should belong to some proper subset of $\mathbb{C}$. In actual computation, however, we should restrict them to $\mathbb{Q}(i)$ or algebraic numbers.

3.1 Computation of the Minimum Distance

Let $\Phi_\mu(s)$ be the minimal distance between f and the polynomials having $\gamma_\mu(s) \in C_\mu$ as a zero. To compute $\Phi_\mu(s)$, we define several subsets of I_μ .

- Δ_μ is defined as follows. When $n = 1$, it is defined as I_μ . When $n \geq 2$, Δ_μ is defined as $\{\sigma \in I_\mu \mid A(e_j(\gamma_\mu(\sigma)), e_k(\gamma_\mu(\sigma))) = 0 \ (1 \leq j < k \leq n)\}$.
- $\tilde{\Delta}_\mu \subset \Delta_\mu$ is defined as $\{\sigma \in \Delta_\mu \mid A(e_j(\gamma(\sigma)), f(\gamma(\sigma))) = 0 \ (j = 1, \dots, n)\} \setminus \{\sigma \in I_\mu \mid e_j(\gamma_\mu(\sigma)) = 0 \ (j = 1, \dots, n)\}$.
- $\tilde{Z}_\mu \subset \tilde{\Delta}_\mu$ is defined as $\bigcup_{j=1}^n \{\sigma \in \tilde{\Delta}_\mu \mid e_j(\gamma_\mu(\sigma)) = 0\}$.
- J_μ is defined as $I_\mu \setminus \Delta_\mu$.
- $Z_\mu \subset J_\mu$ is defined as $\bigcup_{1 \leq j < k \leq n} \{\sigma \in J_\mu \mid A(e_j(\gamma_\mu(\sigma)), e_k(\gamma_\mu(\sigma))) = 0\} \cup \bigcup_{j=1}^n \{\sigma \in J_\mu \mid A(e_j(\gamma_\mu(\sigma)), f(\gamma_\mu(\sigma))) = 0\}$.

Here, when constructing the union, we omit sets such that

$$A(e_j(\gamma_\mu(s)), e_k(\gamma_\mu(s))), \quad A(e_j(\gamma_\mu(s)), f(\gamma_\mu(s))),$$

are identically equal to 0.

Using the above sets, we can represent $\Phi_\mu(s)$ as follows.

$$\Phi_\mu(s) = \begin{cases} \frac{A(f(\gamma_\mu(s)), if(\gamma_\mu(s)))}{\sum_{j=1}^n |A(e_j(\gamma_\mu(s)), if(\gamma_\mu(s)))|}, & s \in \tilde{\Delta}_\mu, \\ \max\{\varphi_{\mu,1}(s), \dots, \varphi_{\mu,n}(s)\}, & s \in J_\mu, \end{cases}$$

where we take the pointwise maximums and define

$$\varphi_{\mu,j}(s) = \frac{|A(e_j(\gamma_\mu(s)), f(\gamma_\mu(s)))|}{\sum_{k=1}^n |A(e_j(\gamma_\mu(s)), e_k(\gamma_\mu(s)))|}.$$

Note that $\Phi_\mu(s)$ is not defined for $s \in \Delta_\mu \setminus \tilde{\Delta}_\mu$.

$\tilde{Z}_\mu$ is a finite set, and $\tilde{\Delta}_\mu \setminus \tilde{Z}_\mu$ is the union of a finite number of intervals, each of which can degenerate to a point. $A(e_j(\gamma_\mu(s)), if(\gamma_\mu(s))) \neq 0$ in each interval of $\tilde{\Delta}_\mu \setminus \tilde{Z}_\mu$, because $A(e_j(\gamma_\mu(s)), if(\gamma_\mu(s))) = 0$ if and only if $e_j(\gamma_\mu(s)) = 0$ when $s \in \tilde{\Delta}_\mu$. Therefore, $\Phi_\mu(s)$ is a rational function on each interval of $\tilde{\Delta}_\mu \setminus \tilde{Z}_\mu$.

$\varphi_{\mu,j}(s)$ is a rational function of s in $J_\mu \setminus Z_\mu$, because the signatures of

$$A(e_j(\gamma_\mu(s)), e_k(\gamma_\mu(s))), \quad A(e_j(\gamma_\mu(s)), f(\gamma_\mu(s)))$$

do not change in each interval of $J_\mu \setminus Z_\mu$. Since Z_μ is a finite set, $J_\mu \setminus Z_\mu$ is the union of a finite number of intervals.

We write $J_\mu \setminus Z_\mu$ as the union of a finite number of intervals $J_{\mu,j}$ and apply the following algorithm for $\{\varphi_{\mu,1}, \dots, \varphi_{\mu,n}\}$ and each $J_{\mu,j}$. Then, we obtain $\Phi_\mu(s)$ as a form of piecewise rational function on $J_\mu \setminus Z_\mu$.

ALGORITHM 1 (COMPUTATION OF $\max\{f_1, \dots, f_n\}$).

Input: A set of rational functions $S = \{f_1, \dots, f_n\}$ and a real interval I such that there are no zeros of the denominators of f_j in I .

Output: $\max\{f_1, \dots, f_n\}$ in the form of $\{(I_j, f_{\theta(j)}) \mid 1 \leq j \leq p\}$, where $I = \bigcup_{j=1}^p I_j$, $I_j \cap I_k = \emptyset$ ($j \neq k$), and $\max\{f_1, \dots, f_n\} = f_{\theta(j)}$ in I_j .

1. If $S = \{f_1\}$, then terminate with the output $\{(I, f_1)\}$.

Otherwise, divide S into two subsets

$$S_1 = \{f_1, \dots, f_{\lceil n/2 \rceil}\}, \quad S_2 = \{f_{\lceil n/2 \rceil+1}, \dots, f_n\}.$$

2. Apply this algorithm to (S_1, I) and (S_2, I) , and obtain $\tilde{f}_1 = \max S_1$ and $\tilde{f}_2 = \max S_2$ as $\{(I_{1,j}, f_{\theta_1(j)}) \mid 1 \leq j \leq q\}$ and $\{(I_{2,j}, f_{\theta_2(j)}) \mid 1 \leq j \leq r\}$, respectively.

3. Compute $\max\{\tilde{f}_1, \tilde{f}_2\}$, and obtain $\{(I_j, f_{\theta(j)}) \mid 1 \leq j \leq p\}$.

Terminate with the output $\{(I_j, f_{\theta(j)}) \mid 1 \leq j \leq p\}$.

In Step 3, take intervals $I_{1,j} \cap I_{2,k} \neq \emptyset$. There are at most $q + r - 1$ such intervals, and to obtain $\max\{f_1, \dots, f_n\}$, we compute $\max\{\tilde{f}_1, \tilde{f}_2\}$ in each nonempty interval. Let the infimum and supremum of a nonempty interval $I_{1,j} \cap I_{2,k}$ be a and b , respectively. (a might be $-\infty$, and b might be ∞ .) Let $\sigma_1 < \dots < \sigma_\zeta$ be the distinct roots of $f_{\theta_1(j)}(s) = f_{\theta_2(k)}(s)$ in the interval (a, b) . Then, to obtain $\max\{\tilde{f}_1, \tilde{f}_2\}$ in $I_{1,j} \cap I_{2,k}$, we compare the values of $f_{\theta_1(j)}(\tau_l)$ and $f_{\theta_2(k)}(\tau_l)$ for $l = 1, \dots, \zeta + 1$, where

$$a < \tau_1 < \sigma_1 < \tau_2 < \dots < \tau_\zeta < \sigma_\zeta < \tau_{\zeta+1} < b.$$

To obtain the minimum distance, we use the following algorithm. Here, if $X = \emptyset$, we regard $\min_{s \in X} \{\Phi(s)\}$ as ∞ , and if $\Phi(s)$ is not defined at $s = \sigma$, we regard $\Phi(\sigma)$ as ∞ .

ALGORITHM 2.

Input: $f, \{e_1, \dots, e_n\}, \{\gamma_1, \dots, \gamma_M\}$, and $\{I_1, \dots, I_M\}$, as defined above.

Output: The minimum distance between f and a nearest polynomial.

1. Carry out the following procedure for $\mu = 1, \dots, M$.

If I_μ consists of one point σ , then $m_\mu = \Phi_\mu(\sigma)$.

Otherwise, carry out (a) and (b).

- (a) Construct Δ_μ , $\tilde{\Delta}_\mu$, $\tilde{Z}_\mu$, and Z_μ .
Write $\tilde{\Delta}_\mu \setminus \tilde{Z}_\mu$ and $J_\mu \setminus Z_\mu$ as the unions of intervals, and construct Φ_μ by using Algorithm 1.
- (b) Compute the following.

$$\begin{aligned} m_{\mu,1} &= \min_{\sigma \in \tilde{Z}_\mu} \{ \Phi_\mu(\sigma) \}, \\ m_{\mu,2} &= \min_{\sigma \in Z_\mu} \{ \Phi_\mu(\sigma) \}, \\ m_{\mu,3} &= \min_{\sigma \in \{ \inf I_\mu, \sup I_\mu \} \cap I_\mu} \{ \Phi_\mu(\sigma) \}, \\ m_{\mu,4} &= \min_{\sigma \in \tilde{\Delta}_\mu \setminus \tilde{Z}_\mu} \{ \Phi_\mu(\sigma) \}, \\ m_{\mu,5} &= \min_{\sigma \in J_\mu \setminus Z_\mu} \{ \Phi_\mu(\sigma) \}, \\ m_\mu &= \min \{ m_{\mu,1}, \dots, m_{\mu,5} \}. \end{aligned}$$

2. Terminate with the output $\min \{ m_1, \dots, m_M \}$.

Note that $\tilde{Z}_\mu$, Z_μ , and $\{ \inf I_\mu, \sup I_\mu \} \cap I_\mu$ are finite sets. Since we obtain Φ_μ as a piecewise rational function on $\tilde{\Delta}_\mu \setminus \tilde{Z}_\mu$ and $J_\mu \setminus Z_\mu$, we can compute $m_{\mu,4}$ and $m_{\mu,5}$. If the output is ∞ , there is no polynomial of the form $f + \sum_{j=1}^n a_j e_j$ having a zero in D .

3.2 Computational Complexity

To analyze the computational complexity of Algorithm 1, we introduce Davenport-Schinzel sequences [4].

Definition 1. Let $\Sigma_n = \{ a_1, a_2, \dots, a_n \}$, and define $L_{n,s}$ as the set of strings over the alphabet Σ_n that do not contain any $a_i a_i$ as a substring and do not contain any ξ_{ij}^s ($i \neq j$) as a subsequence, where ξ_{ij}^s is an alternating string of length $s+2$, defined as follows: $\xi_{ij}^1 = a_i a_j a_i$, $\xi_{ij}^{2p} = \xi_{ij}^{2p-1} a_j$, $\xi_{ij}^{2p+1} = \xi_{ij}^{2p} a_i$ ($1 \leq p$). Then, $\lambda(n, s)$ is defined as $\max \{ |\sigma| \mid \sigma \in L_{n,s} \}$.

A sequence belonging to $L_{n,s}$ is said to be an (n, s) Davenport-Schinzel sequence.

The following lemma shows the relation between Algorithm 1 and Davenport-Schinzel sequences.

Lemma 2. Let $f_1, \dots, f_n$ be real-valued functions of t , each of which is continuous. If no two distinct functions f_i and f_j intersect more than s times, then

$$h(t) = \max \{ f_1(t), \dots, f_n(t) \}$$

is made up of no more than $\lambda(n, s)$ pieces, and this bound is the best possible.

Proof. See [1]. (This lemma is essentially Lemma 2.4 in [1].) $\square$

Let δ be the maximum of the degrees of the numerators and denominators of $f_1, \dots, f_n$ in Algorithm 1. Then, f_j and f_k ($j \neq k$) intersect at most 2δ times in I . Thus, to construct $\max \{ \tilde{f}_1, \tilde{f}_2 \}$ in Step 3 of Algorithm 1, the number of equations $f_j = f_k$ to be solved is at most

$$\lambda \left(\left\lceil \frac{n}{2} \right\rceil, 2\delta \right) + \lambda \left(\left\lfloor \frac{n}{2} \right\rfloor, 2\delta \right) - 1.$$

Therefore, to construct $\max \{ f_1, \dots, f_n \}$, the number of equations $f_j = f_k$ to be solved is at most

$$\sum_{j=1}^{\lceil \log_2 n \rceil} 2^j \lambda(2^{\lceil \log_2 n \rceil - j}, 2\delta). \quad (2)$$

To evaluate (2), we use the following lemma.

Lemma 3. For any positive integer m , the inequality $m\lambda(n, s) \leq \lambda(mn, s)$ holds.

Proof. Let $\sigma_1, \dots, \sigma_m$ be (n, s) Davenport-Schinzel sequences over the alphabets $\Sigma_n, \Sigma_{2n} \setminus \Sigma_n, \dots, \Sigma_{mn} \setminus \Sigma_{(m-1)n}$, respectively. Then, the sequence $\sigma_1 \dots \sigma_m$ is an (mn, s) Davenport-Schinzel sequence over Σ_{mn} . $\square$

Using Lemma 3, we obtain

$$\begin{aligned} \sum_{j=1}^{\lceil \log_2 n \rceil} 2^j \lambda(2^{\lceil \log_2 n \rceil - j}, 2\delta) &\leq 2^{\lceil \log_2 n \rceil} \lambda(2^{\lceil \log_2 n \rceil - 1}, 2\delta) \\ &< 2^{\lceil \log_2 n \rceil} \lambda(n, 2\delta). \end{aligned}$$

$\lambda(n, s)$ is of polynomial order in n and s . A bound

$$\lambda(n, s) \leq sn(n-1) + 1$$

is provided in [4], and in fact, $\lambda(n, s)$ is almost linear in n . See [18] for the bounds.

Next, we analyze the computational complexity of Algorithm 2. For a polynomial $p(x) \in \mathbb{R}[x]$ of degree d and a rational function $r(x) \in \mathbb{R}(x)$, whose numerator and denominator have degrees of at most m , the degrees of the numerator and denominator of $p(r(x))$ are at most md . Thus, the degrees of the numerators and denominators of $\text{Rep}(r(x))$ and $\text{Imp}(r(x))$ are at most $2md$.

Let $d = \max \{ \deg(f), \deg(e_1), \dots, \deg(e_n) \}$, and let m be the maximum degree of the numerators and denominators of $\gamma_1(s), \dots, \gamma_M(s)$. Then, the degrees of the numerators and denominators of $f(\gamma_\mu(s))$ and $e_j(\gamma_\mu(s))$ are at most md , and the degrees of the numerators and denominators of

$$\begin{aligned} &A(e_j(\gamma_\mu(s)), e_k(\gamma_\mu(s))), \quad A(e_j(\gamma_\mu(s)), f(\gamma_\mu(s))), \\ &A(f(\gamma_\mu(s)), if(\gamma_\mu(s))), \end{aligned}$$

are at most $8md$.

For each μ ($\mu = 1, \dots, M$), the number of equations to be solved and their degrees are as follows.

- To construct Δ_μ , $\tilde{\Delta}_\mu$, $\tilde{Z}_\mu$, and Z_μ , the number of equations to be solved is one each. The degrees of the equations for Δ_μ , $\tilde{\Delta}_\mu$, $\tilde{Z}_\mu$, and Z_μ are at most

$$8md, \quad 8md, \quad mnd, \quad 4mn(n+1)d,$$

respectively.

- To construct Φ_μ in $J_\mu \setminus Z_\mu$, the number of equations to be solved and the degree of the equations are at most

$$(8mn(n+1)d + 16md + 2) \lceil \log_2 n \rceil \lambda(n, 16m(n+1)d),$$

respectively.

Note that $J_\mu \setminus Z_\mu$ is the union of at most $4mn(n+1)d + 8md + 1$ intervals. To construct Φ_μ on each interval, the number of equations to be solved is at most $2 \lceil \log_2 n \rceil \lambda(n, 16m(n+1)d)$.

- To compute $m_{\mu,1}$, $m_{\mu,2}$, and $m_{\mu,3}$, the numbers of values to be compared are at most

$$mnd, \quad 4mn(n+1)d, \quad 2,$$

respectively.

- $\tilde{\Delta}_\mu \setminus \tilde{Z}_\mu$ is the union of at most $8md$ intervals, each of which can degenerate to a point. To compute $m_{\mu,4}$ on a nondegenerate interval in $\tilde{\Delta}_\mu \setminus \tilde{Z}_\mu$, we solve $\Phi'_\mu(s) = 0$. The degree of the numerator of $\Phi'_\mu(s)$ is at most

$$16m(n+1)d - 1,$$

since the degrees of the numerators and denominators of $\Phi_\mu(s)$ are at most $8m(n+1)d$.

- To compute $m_{\mu,5}$, we solve $\Phi'_\mu(s) = 0$ in each interval in $J_\mu \setminus Z_\mu$. The degree of the numerator of $\Phi'_\mu(s)$ is at most

$$16m(n+1)d - 1,$$

since the degrees of the numerators and denominators of $\Phi_\mu(s)$ are at most $8m(n+1)d$.

From the above statements, we conclude that the number of equations to be solved and the degrees are of polynomial order in m, n, d , and M .

3.3 Construction of a Nearest Polynomial

Suppose that a nearest polynomial exists. Let ϵ be the distance between f and a nearest polynomial having $\alpha \in C$ as a zero. Hereafter, we use the notations given in the proof of Lemma 1.

When the convex polygon $E_\epsilon(\alpha)$ degenerates to a line segment, there is a nearest polynomial $\tilde{f} = f + \sum_{j=1}^n a_j e_j$ such that

$$a_j = \begin{cases} -\epsilon, & \text{if } A(e_j(\alpha), if(\alpha)) \geq 0, \\ \epsilon, & \text{otherwise.} \end{cases}$$

When the convex polygon $E_\epsilon(\alpha)$ does not degenerate to a line segment, let j be the smallest number such that the equality in Lemma 1 holds. If $A(\sum_{k=1}^n \tilde{e}_k(\alpha), f(\alpha)) \leq 0$, then set $\tau = 1$; otherwise, set $\tau = -1$. Then, $\tau f(\alpha)$ lies on the edge whose endpoints are $\epsilon w_{\nu(j)}$ and $\epsilon w_{\nu(j)+1}$ ($0 \leq j \leq \tilde{n} - 1$). Therefore, there is t ($0 \leq t \leq 1$) such that

$$\tau f(\alpha) = (1-t)\epsilon w_{\nu(j)} + t\epsilon w_{\nu(j)+1}$$

holds. Thus,

$$\begin{aligned} \tau f(\alpha) &= (1-t)\epsilon \left(\sum_{k=1}^{\nu(j)} p_k - \sum_{k=\nu(j)+1}^{\tilde{n}} p_k \right) \\ &\quad + t\epsilon \left(\sum_{k=1}^{\nu(j)+1} p_k - \sum_{k=\nu(j)+2}^{\tilde{n}} p_k \right) \\ &= \epsilon \sum_{k=1}^{\nu(j)} p_k - \epsilon \sum_{k=\nu(j)+2}^{\tilde{n}} p_k + \epsilon(2t-1)p_{\nu(j)} \\ &= \epsilon \sum_{k \in L(j)} \tilde{e}_k(\alpha) + \epsilon \sum_{k \in R(j)} \tilde{e}_k(\alpha) + \epsilon(2t-1)p_j. \end{aligned}$$

Let $P(j) = \{k \mid A(\tilde{e}_j(\alpha), \tilde{e}_k(\alpha)) = 0\}$. The polynomial

$$\begin{aligned} g(x) &= f(x) + \sum_{k \in L(j)} \tau \epsilon \tilde{e}_k(x) + \sum_{k \in R(j)} \tau \epsilon \tilde{e}_k(x) \\ &\quad + \tau \epsilon (2t-1) \sum_{k \in P(j)} \tilde{e}_k(x) \end{aligned}$$

has α as a zero. Noting that $0 \leq t \leq 1$, we have

$$\sum_{k \in P(j)} (-\epsilon \tilde{e}_k(\alpha)) \leq \epsilon(2t-1) \sum_{k \in P(j)} \tilde{e}_k(\alpha) \leq \sum_{k \in P(j)} \epsilon \tilde{e}_k(\alpha).$$

Therefore, we can find $Q \subset P(j)$ and $l \in Q$ such that

$$\begin{aligned} &\sum_{k \in Q} (-\epsilon \tilde{e}_k(\alpha)) + \sum_{k \in P(j) \setminus Q} \epsilon \tilde{e}_k(\alpha) \\ &\leq \epsilon(2t-1) \sum_{k \in P(j)} \tilde{e}_k(\alpha) \\ &\leq \sum_{k \in Q \setminus \{l\}} (-\epsilon \tilde{e}_k(\alpha)) + \sum_{k \in (P(j) \setminus Q) \cup \{l\}} \epsilon \tilde{e}_k(\alpha). \end{aligned}$$

Set

$$\begin{aligned} \tilde{t} &= \frac{1}{\tilde{e}_l(\alpha)} \left((2t-1) \sum_{k \in P(j)} \tilde{e}_k(\alpha) \right. \\ &\quad \left. + \sum_{k \in Q \setminus \{l\}} \tilde{e}_k(\alpha) - \sum_{k \in P(j) \setminus Q} \tilde{e}_k(\alpha) \right), \end{aligned}$$

and

$$a_k = \begin{cases} \tau \tilde{t} \epsilon, & \text{if } k = l, \\ \tau \epsilon, & \text{if } k \neq l \text{ and } \tilde{e}_k(\alpha) = e_k(\alpha), \\ -\tau \epsilon, & \text{otherwise.} \end{cases}$$

Then, $\tilde{f}(x) = f(x) + \sum_{k=1}^n a_k e_k(x)$ is a nearest polynomial such that $\tilde{f}(\alpha) = 0$ and $|a_k| = \epsilon$ for $k \neq l$.

4. EXAMPLE

To illustrate the flow of the algorithms, we give the following example.

Example 2. Let $f(x) = x^2 + 2$, and let D be the closed unit disk whose boundary C is the union of $C_1 = \{1\}$ and $C_2 = C \setminus \{1\}$. C_1 and C_2 are parameterized by

$$\begin{aligned} \gamma_1 : I_1 &\rightarrow C_1, \quad s \mapsto s, \\ \gamma_2 : I_2 &\rightarrow C_2, \quad s \mapsto \frac{s-i}{s+i}, \end{aligned}$$

where $I_1 = \{1\}$ and $I_2 = \mathbb{R}$. Find a nearest polynomial

$$\tilde{f} = f + a_1 e_1 + a_2 e_2 + a_3 e_3 \quad (a_1, a_2, a_3 \in \mathbb{R})$$

having a zero in D , where $e_1 = x^2$, $e_2 = x$, and $e_3 = 1$.

First, we show the computation for $\mu = 1$. Since $f(1) = 3$, $e_1(1) = e_2(1) = e_3(1) = 1$ lie on a straight line (the real axis), and $e_1(1) \neq 0$, there is a nearest polynomial having 1 as a zero. The minimal distance m_1 is 1.

Next, we show the computation for $\mu = 2$.

$$\begin{aligned} f(\gamma_2(s)) &= \frac{3s^2 + 2is - 3}{s^2 + 2is - 1} \\ &= \frac{3s^4 - 2s^2 + 3}{s^4 + 2s^2 + 1} + i \frac{-4s^3 + 4s}{s^4 + 2s^2 + 1}, \\ e_1(\gamma_2(s)) &= \frac{s^2 - 2is - 1}{s^2 + 2is - 1} \\ &= \frac{s^4 - 6s^2 + 1}{s^4 + 2s^2 + 1} + i \frac{-4s^3 + 4s}{s^4 + 2s^2 + 1}, \\ e_2(\gamma_2(s)) &= \frac{s-i}{s+i} = \frac{s^2-1}{s^2+1} + i \frac{-2s}{s^2+1}, \\ e_3(\gamma_2(s)) &= 1. \end{aligned}$$

Therefore,

$$\begin{aligned}
A(e_1(\gamma_2(s)), e_2(\gamma_2(s))) &= \frac{2s}{s^2 + 1}, \\
A(e_1(\gamma_2(s)), e_3(\gamma_2(s))) &= \frac{4s^3 - 4s}{s^4 + 2s^2 + 1}, \\
A(e_2(\gamma_2(s)), e_3(\gamma_2(s))) &= \frac{2s}{s^2 + 1}, \\
A(e_1(\gamma_2(s)), f(\gamma_2(s))) &= \frac{8s^3 - 8s}{s^4 + 2s^2 + 1}, \\
A(e_2(\gamma_2(s)), f(\gamma_2(s))) &= \frac{2s}{s^2 + 1}, \\
A(e_3(\gamma_2(s)), f(\gamma_2(s))) &= \frac{-4s^3 + 4s}{s^4 + 2s^2 + 1}.
\end{aligned}$$

Thus, we have

$$\begin{aligned}
\Delta_2 &= \tilde{\Delta}_2 = \{0\}, & \tilde{Z}_2 &= \emptyset, & Z_2 &= \{-1, 1\}, \\
\Delta_2 \setminus \tilde{\Delta}_2 &= \emptyset, & J_2 &= (-\infty, 0) \cup (0, \infty), \\
J_2 \setminus Z_2 &= (-\infty, -1) \cup (-1, 0) \cup (0, 1) \cup (1, \infty).
\end{aligned}$$

In $\tilde{\Delta}_2$ and Z_2 , we have $\Phi_2(0) = 1$ and $\Phi_2(-1) = \Phi_2(1) = 1/2$.

In $J_2 \setminus Z_2$, we have

$$\Phi_2(s) = \begin{cases} \frac{4s^2 - 4}{3s^2 - 1}, & s \in (-\infty, -\sqrt{7/5}] \cup (\sqrt{7/5}, \infty], \\ \frac{1}{2}, & s \in [-\sqrt{7/5}, -1) \cup (-1, -\sqrt{5/7}] \\ & \cup [\sqrt{5/7}, 1) \cup (1, \sqrt{7/5}], \\ \frac{4s^2 - 4}{s^2 - 3}, & s \in (-\sqrt{5/7}, 0) \cup (0, \sqrt{5/7}). \end{cases}$$

Since $\{\inf I_2, \sup I_2\} \cap I_2 = \emptyset$, we obtain

$$m_{2,1} = m_{2,3} = \infty, \quad m_{2,2} = m_{2,5} = \frac{1}{2}, \quad m_{2,4} = 1.$$

Therefore, $m_2 = \min\{m_{2,1}, m_{2,2}, m_{2,3}, m_{2,4}, m_{2,5}\} = 1/2$ and $\min\{m_1, m_2\} = 1/2$; that is, the minimal distance is $1/2$. A nearest polynomial is $(3x^2 + 3)/2$, having zeros at $\pm i$.

5. CONCLUSION

For a given real univariate polynomial f and a prescribed closed complex domain D whose boundary is a simple curve, we have proposed a rigorous method for finding a real univariate polynomial $\tilde{f}$ such that $\tilde{f}$ has a zero in D and $\|f - \tilde{f}\|_\infty$ is minimal.

Improving the efficiency of this method through numeric computation is one of our future research directions. Another direction is to consider the case of complex coefficients.

6. REFERENCES

- [1] M. J. Atallah. Some dynamic computational geometry problems. *Computers and Mathematics with Applications*, 11(12):1171–1181, 1985.
- [2] A. C. Bartlett, C. V. Hollot, and H. Lin. Root location of an entire polytope of polynomials: It suffices to check the edges. *Mathematics of Controls, Signals and Systems*, 1:61–71, 1988.
- [3] S. P. Bhattacharyya, H. Chapellat, and L. H. Keel. *Robust Control, The Parametric Approach*. Prentice-Hall, 1995.

- [4] H. Davenport and A. Schinzel. A combinatorial problem concerned with differential equations. *American Journal of Mathematics*, 87(3):684–694, 1965.
- [5] S. Graillat. A note on a nearest polynomial with a given root. *ACM SIGSAM Bulletin*, 39(2):53–60, 2005.
- [6] M. A. Hitz and E. Kaltofen. Efficient algorithms for computing the nearest polynomial with constrained roots. In *Proc. 1998 International Symposium on Symbolic and Algebraic Computation (ISSAC98)*, pages 236–243, 1998.
- [7] M. A. Hitz, E. Kaltofen, and Y. N. Lakshman. Efficient algorithms for computing the nearest polynomial with a real root and related problems. In *Proc. 1999 International Symposium on Symbolic and Algebraic Computation (ISSAC99)*, pages 205–212, 1999.
- [8] E. Kaltofen. Efficient algorithms for computing the nearest polynomial with parametrically constrained roots and factors. Lecture at the Workshop on Symbolic and Numerical Scientific Computation (SNSC’99), 1999.
- [9] M. Karow. Geometry of spectral value sets. PhD thesis, Universität Bremen, 2003.
- [10] V. L. Kharitonov. Asymptotic stability of an equilibrium position of a family of systems of linear differential equations. *Differentsial’nye Uravneniya*, 14(11):2086–2088, 1978.
- [11] R. G. Mosier. Root neighborhoods of a polynomial. *Math. Comp.*, 47(175):265–273, 1986.
- [12] L. Qiu and E. J. Davison. A simple procedure for the exact stability robustness computation of polynomials with affine coefficient perturbations. *Systems and Control Letters*, 13:413–420, 1989.
- [13] A. Rantzer. Stability conditions for polytopes of polynomials. *IEEE Trans. Auto. Control*, 37(1):79–89, 1992.
- [14] N. Rezvani and R. C. Corless. The nearest polynomial with a given zero, revisited. *ACM SIGSAM Bulletin*, 39(3):73–79, 2005.
- [15] H. Sekigawa. The nearest polynomial with a zero in a given domain. In *Proc. 2007 International Workshop on Symbolic-Numeric Computation (SNC’07)*, pages 190–196, 2007.
- [16] H. Sekigawa and K. Shirayanagi. Locating real multiple zeros of a real interval polynomial. In *Proc. 2006 International Symposium on Symbolic and Algebraic Computation (ISSAC2006)*, pages 310–317, 2006.
- [17] H. Sekigawa and K. Shirayanagi. On the location of zeros of an interval polynomial. In D. Wang and L. Zhi, editors, *Symbolic-Numeric Computation*, pages 167–184. Birkhäuser, 2007.
- [18] M. Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and Their Geometric Applications*. Cambridge University Press, 1995.
- [19] H. J. Stetter. The nearest polynomial with a given zero, and similar problems. *ACM SIGSAM Bulletin*, 33(4):2–4, 1999.

Moving Frames for Laplace Invariants

Ekaterina Shemyakova
Research Institute for Symbolic Computation
J.Kepler University
Altenbergerstr. 69, A-4040 Linz, Austria
kath@risc.uni-linz.ac.at

Elizabeth L. Mansfield
Institute of Mathematics, Statistics and Actuarial
Science
The University of Kent, Canterbury, CT2 7NZ
E.L.Mansfield@kent.ac.uk

ABSTRACT

The development of symbolic methods for the factorization and integration of linear PDEs, many of the methods being generalizations of the Laplace transformations method, requires the finding of complete generating sets of invariants for the corresponding linear operators and their systems with respect to the gauge transformations $L \rightarrow g(x, y)^{-1} \circ L \circ g(x, y)$. Within the theory of Laplace-like methods, there is no uniform approach to this problem, though some individual invariants for hyperbolic bivariate operators, and complete generating sets of invariants for second- and third-order hyperbolic bivariate ones have been obtained.

Here we demonstrate a systematic and much more efficient approach to the same problem by application of moving-frame methods. We give explicit formulae for complete generating sets of invariants for second- and third-order bivariate linear operators, hyperbolic and non-hyperbolic, and also demonstrate the approach for pairs of operators appearing in Darboux transformations.

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: AlgorithmsAlgebraic algorithms

General Terms

Algorithms, Theory

Keywords

Partial Differential Operators, moving frames, invariants, gauge transformations

1. INTRODUCTION

Nowadays, constructive factorization algorithms are greatly in demand, being used in recent algorithms for the exact solution of Linear Partial Differential Equations (LPDEs). For example, they are used in the numerous generalizations and

modifications of the 18th-century Laplace-Transformations Method, in the Loewy decomposition method, and in other methods (see for example [2, 3, 4, 10, 21, 20]). When seeking factorizations, it is useful to consider transformations of Linear Partial Differential Operators (LPDOs), in particular gauge transformations, which are transformations of the form $L \rightarrow g(x, y)^{-1} \circ L \circ g(x, y)$. Such transformations do not change the property that a particular LPDO has a factorization, and the corresponding classification problem needs the discovery of invariants. The invariants of interest are algebraic expressions formed using the coefficients of the LPDO and their derivatives. Sums, products and derivatives of such invariants are again invariants, implying the existence of infinitely many invariants. Therefore, it is preferable to have a generating set of invariants, meaning that any invariant can be expressed in terms of these basic ones.

The venerable Laplace-Transformations Method [7] displays the connection between factorization, integration, and invariant finding. It applies to the second-order hyperbolic LPDO

$$L = D_x \circ D_y + a(x, y)D_x + b(x, y)D_y + c(x, y). \quad (1)$$

Further below we omit the designation of the dependence on x and y in the coefficients. Euler and Laplace introduced the notion of the Laplace invariants while searching for exact methods to integrate the above equation. The integration method was based on the idea that the corresponding equation $L(z) = 0$ could be solved in quadratures provided L had a factorization. However, for many functions a , b and c , the operator does not have a factorization, and so Laplace considered two incomplete factorizations of (1):

$$L = (D_x + b) \circ (D_y + a) + h = (D_y + a) \circ (D_x + b) + k,$$

where $h = c - a_x - ab$ and $k = c - b_y - ab$. Laplace proved that the functions h and k are invariants of (1) with respect to gauge transformations, and consequently they are called the *Laplace invariants*. In addition there is no invariant that is not a function of h and k and their derivatives, and so $\{h, k\}$ is a generating set of invariants.

Laplace's problem equation (1) is second order, and from the end of the 18th century until recently nothing was known about invariants (of LPDOs) for orders higher than two. An attempt was made in [11] to find invariants for hyperbolic bivariate third-order operators having a symbol with constant coefficients. Although four invariants were determined, they were not sufficient to form a generating set. Relaxing the condition of constant coefficients and extending the order,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

[21] mentioned, for hyperbolic bivariate arbitrary-order operators, a way to determine some invariants. To improve on these results, new ways to obtain invariants for hyperbolic bivariate arbitrary order operators with symbols containing nonconstant coefficients by means of generalized Laplace methods were introduced [17, 18]. Thus, for third-order operators of the form

$$L = (p(x, y)D_x + q(x, y)D_y)D_xD_y + \sum_{i+j=0}^2 a_{ij}(x, y)D_x^iD_y^j, \quad (2)$$

with p and q not zero, which is a normalized form of a hyperbolic operator on the plane, a complete generating set of invariants was determined. This completed the results in [11] and extending them to the nonconstant case. The results were obtained by means of some generalizations of Laplace methods, and are capable of bringing us a number of invariants for hyperbolic operators of arbitrary order as well. However, such an approach will require large computations, and is quite inefficient. Also questions on the automatic extension of a partial set of invariants in order to obtain a complete generating set of invariants remain open. The area of applications for a generating set of invariants is broad. In particular, the generating set of invariants for the operators (2) allows us to obtain a complete description of properties such as the existence of a factorization of a certain type, or of the existence of any factorization for equivalent classes of (2) [19].

Generalizations of the Laplace transformations method, factorization problems of LPDOs – all these topics – have been developed independently from, and without reference to, the methods of moving frames. The idea of moving frames is associated with Cartan [5], but in fact was used earlier, see [1], for studying geometric properties of submanifolds and their invariants under the action of a transformation group. In the 1970's several researchers made an attempt to provide Cartan's intuitive method with a strong theoretic foundation. More recently, Fels and Olver formulated a new, constructive approach to equivariant moving frame theory for the *finite-dimensional* group actions – the regularized moving frame method (see the series of works starting with [8, 9]). These methods happen to be very useful and have been applied in various areas of mathematics. In particular, in differential equations the theory has been applied to the computation of symmetry groups and the classification of second-order hyperbolic bivariate LPDEs, in the form $u_{xy} + a(x, y)u_x + b(x, y)u_y + c(x, y)u = 0$. For references, refer to [13]. In [12], the regularized moving frame method was applied to construct new methods analogous to differential-elimination algorithms for systems given in terms of the differential invariants of a Lie group action, for which the classical differential-elimination methods are not effective. Substantial recent contributions of Olver and Pohjanpelto [14, 15, 16] and of the same authors with Cheh [6], explore the infinite-dimensional case, and pave the way for computer algebra applications.

In the present paper, the infinite-dimensional case is under consideration (as the gauge transformation group is infinite-dimensional). We apply the (regularized) moving frame methods to find complete generating sets of invariants for second- and third-order bivariate hyperbolic and non-hyperbolic LPDOs, and some pairs of those. For these cases we construct moving frames by means of the Cartan normalization

procedure associated with a choice of local cross-section to the pseudo-group orbits. Then the moving frames induce an invariantization process, which being applied to the standard jet coordinates provides us with a complete local system of normalized differential invariants.

Thus, in the present paper we show an elegant way to obtain a generating set of invariants for the case of second- and third-orders bivariate hyperbolic operators. These generating sets by themselves do not constitute a new result, as they were already found by means of the generalized Laplace methods, but then our approach allows us to go further, and obtain generating sets of invariants for second- and third-order non-hyperbolic LPDOs, which could not be handled by the generalized Laplace methods. Looking forward to the applications we considered also the pairs of the operators, that appear in the Darboux (1, 1)-transformations, and (2, 2)-transformations [7].

The paper is organized as follows. In Section 2 preliminary definitions and notations are given. In Section 3, a way to obtain a generating set of invariants for third-order bivariate hyperbolic operators by means of generalized Laplace methods is shown. In Section 4 and Section 5 we completely solve the problem of finding invariants and their generating sets for second- and third-order LPDOs, hyperbolic and non-hyperbolic, by applying regularized moving frame methods. In Section 6 we demonstrate the approach for obtaining a complete generating sets of invariants on the example of the pairs appearing in the Darboux (1, 1)- and (2, 2)-transformations. Applications to factorizations will be in a later paper.

2. DEFINITIONS AND NOTATIONS

Consider a field K of characteristic zero with commuting derivations ∂_x, ∂_y acting on it. Consider the ring of linear differential operators $K[D] = K[D_x, D_y]$, where D_x, D_y correspond to the derivations ∂_x, ∂_y , respectively. In $K[D]$ the variables D_x, D_y commute with each other, but not with elements of K . For $a \in K$ we have the relation $D_i a = a D_i + \partial_i(a)$. Any operator $L \in K[D]$ is of the form $L = \sum_{i+j=0}^d a_{ij} D_x^i D_y^j$, where $a_{ij} \in K$. The polynomial $\text{Sym}_L = \sum_{i+j=d} a_{ij} X^i Y^j$ in formal variables X, Y is called the (principal) *symbol* of L . An operator $L \in K[D]$ is said to be *hyperbolic* if its symbol is completely factorable (all factors are of first order) and each factor has multiplicity one. In the present paper we assume that the symbols of the operators are completely factorable.

Let K^* denote the set of invertible elements in K . For $L \in K[D]$ and every $g \in K^*$ consider the gauge transformation $L \rightarrow g^{-1} \circ L \circ g$. Then an algebraic differential expression I in coefficients of L is *invariant* under the gauge transformations (we consider only these in the present paper) if it is unaltered by these transformations. Trivial examples of invariants are the coefficients of the symbol of the operator. A generating set of invariants is a set using which all possible differential invariants can be expressed.

Let $u = u(x, y)$, then for the mixed partial derivatives of u we will use the following conventional notations: u_I , where $I \in A \times B$ meaning $u_I = \frac{\partial u}{\partial^a x \partial^b y}$ for some $a \in A$ and $b \in B$.

3. GENERALIZED LAPLACE METHOD FOR HYPERBOLIC LPDOS

Any hyperbolic bivariate third-order LPDO in some system of coordinates has the form (2). To ease the notation, later on in the paper, we do not designate the dependance on x and y of the coefficients of (2). Note that the normalized form of such operators is slightly simpler than above, namely, one can put without loss of generality $p = 1$. We introduce the parameter p to make all the reasoning symmetric with respect to x and y . The family of operators (2) admits gauge transformations, and being the coefficients of the symbol, p, q are invariants.

In [18] 12 invariants have been computed. However, all these 12 invariants can be expressed in terms of just four ones (I_1, I_2, I_3, I_4 in (3)), and no other invariant (independent from these four ones) can be obtained. On the other hand one must find at least one extra invariant in order to form a generating system of invariants, because none of the invariants yet obtained depends on the free coefficient a_{00} of (2).

By means of mathematical intuition and results from running a custom MAPLE package for linear partial differential operators with parametric coefficients, the authors of [18] succeeded in finding a fifth independent invariant. They proved that the enlarged set of 5 independent invariants is a generating set of invariants.

THEOREM 3.1. [18] *The following form a generating set of invariants for operators of the form (2):*

$$\left. \begin{aligned} I_p &= p, \\ I_q &= q, \\ I_1 &= 2q^2a_{20} - qa_{11}p + 2a_{02}p^2, \\ I_2 &= a_{02}p^2q_y - qp^2a_{02y} + q^2a_{20x}p - q^2a_{20}p_x, \\ I_3 &= a_{10}p^2 + (2q_y p - 3qp_y)a_{20} + a_{20}^2q - a_{11y}p^2 \\ &\quad + a_{11}p_y p + qp a_{20y} - a_{11}a_{20}p, \\ I_4 &= a_{01}q^2 + (2qp_x - 3pq_x)a_{02} + a_{02}^2p - a_{11x}q^2 \\ &\quad + a_{11}qq_x + qp a_{02x} - a_{02}a_{11}q, \\ I_5 &= a_{00}p^3q + J_5, \end{aligned} \right\} \quad (3)$$

where $J_5 = 2a_{02}p^3a_{20x} - 2q^2a_{20}^2p_x - a_{02}a_{10}p^3 + \frac{1}{2}a_{11x}p_y p^2q + \frac{1}{2}a_{11y}p_x p^2q + (\frac{1}{2}p_{xy}p^2q - p_x p_y p q)a_{11} + a_{11}p q a_{20}p_x - \frac{1}{2}a_{11xy}p^3q + (qq_x p^2 - q^2 p_x p)a_{20y} - a_{11}p^2 q a_{20x} + (qp^2 q_y - pq^2 p_y)a_{20x} + 2q^2 a_{20}a_{20x}p + q(q_{xy}p^2 - qp_{xy}p + 4qp_x p_y - 2p_x q_y p - 2q_x p p_y)a_{20} - a_{01}a_{20}p^2q - 2a_{02}p^2a_{20}p_x + a_{20}a_{11}a_{02}p^2$.

A constructive proof of the theorem was given, that is assuming that the values of seven invariants are equal for two operators L_1 and L_2 of the form (2), a function $g \in K$ that connects them (i.e. $g^{-1}L_1g = L_2$) was determined.

For LPDOs of arbitrary order the idea of obstacles to factorizations, which is the generalized Laplace method, is capable of bringing us a number of invariants for hyperbolic operators of arbitrary order. However for LPDOs of arbitrary order such an approach will require large computations, and be quite inefficient. Already in the case of a third-order operator the generalized Laplace method brought 12 invariants, and only 4 of those were independent from each other. As the order of the operators increases, such inefficiency will get worse. Also one has to find a way to make the enlarging and completing of a generating set of invariants automatic, but at the moment such an algorithm is not known even for the case of order three. One idea might be to analyze the structure of invariants for operators of order three (four, if

we will find them) and try to guess the form for invariants for arbitrary order operators.

4. MOVING FRAMES FOR LPDOS OF ORDER 2

Consider, first, hyperbolic LPDOs. Second-order bivariate hyperbolic LPDOs have the normalized form (1).

THEOREM 4.1. *The invariants $I_y^b = b_y - a_x$ and $I^c = c - ab - a_x$ form together a generating set of invariants for operators of the form (1).*

REMARK 4.2. *The statement by itself is not a new result, since such a complete system of invariants for hyperbolic second-order bivariate operators has been known for many years [7]. The new thing here is the moving frame approach which we demonstrate on this simplest possible case.*

PROOF. Consider the transform formulae of the gauge transformation action $L \rightarrow f^{-1}Lf$, with $f = f(x, y) \in K^*$, which is considering as $f = \exp(g(x, y))$ to avoid fractions:

$$\left\{ \begin{array}{l} \tilde{x} = x, \\ \tilde{y} = y, \end{array} \right. \quad \& \quad \left\{ \begin{array}{l} \tilde{a} = a + g_y, \\ \tilde{b} = b + g_x, \\ \tilde{c} = c + ag_x + bg_y \\ \quad + g_{xy} + g_x g_y. \end{array} \right. \quad (4)$$

The prolonged action in the extended space M with local variables $(x, y, a, b, c, a_x, b_x, c_x, a_y, \dots)$ is free, because already the conditions $\tilde{a} = a, \tilde{b} = b$ imply that g should be a constant, while we consider the gauge transformations up to a constant factor. In a neighborhood of some generic point (x_0, y_0) we have

$$\begin{aligned} g(x, y) &= g(x_0, y_0) + g_x(x_0, y_0)(x - x_0) \\ &\quad + g_y(x_0, y_0)(y - y_0) + \frac{1}{2}g_{xx}(x_0, y_0)(x - x_0)^2 + \dots \end{aligned} \quad (5)$$

and without loss of generality one can assume $g(x_0, y_0) = 0$. Now $g_J(x_0, y_0)$, $J \in \mathbb{N}_0^2$ are independent parameters of the prolonged action. Further below we omit the designation of the dependence on x_0 and y_0 .

We start the Cartan normalization procedure by choosing normalization equations $\tilde{a} = 0$ and $\tilde{b} = 0$, which can be solved for the parameters g_x and g_y . To construct the moving frame cross-section, we need to obtain equations for all the independent parameters. Consider the following formulae of the first prolongation:

$$\left\{ \begin{array}{l} \tilde{a}_x = a_x + (g_y)_x, \\ \tilde{a}_y = a_y + (g_y)_y, \\ \tilde{b}_x = b_x + (g_x)_x. \end{array} \right.$$

The normalization equations $\tilde{a}_x = 0, \tilde{a}_y = 0$, and $\tilde{b}_x = 0$ allow us to fix the values of the parameters g_{xy}, g_{yy} , and g_{xx} , correspondingly. Considering further prolongations of the action, and choosing the normalization equations

$$\tilde{a}_J = 0, \forall J \in \mathbb{N}_0^2,$$

we get equations for every action parameter $(g_y)_J$, while the normalization equations

$$\tilde{b}_I = 0, \quad \forall I \in \mathbb{N}_0 \times \{0\} \quad (6)$$

provide us with the equations for all $(g_x)_I$. Thus, all the parameters have been determined, and we can explicitly construct a moving frame cross-section. The frame (or moving

frame) consists of the values of $g_x, g_y, \dots$ in terms of a_J and b_J .

To compute the invariantization ι of a function or a differential operator, one writes out explicitly how it transforms under the action, and replace the parameters by their values given by the frame. Invariantizing the coordinate functions on M leads us to the differential invariants

$$I_J^{a_{ij}} = \iota((a_{ij})_J) = \widetilde{(a_{ij})_J}|_{\text{frame}} .$$

Those invariants that correspond to the coordinate functions of M appearing in the chosen normalization equations are constants, and called *phantom differential invariants*, while the remaining form a complete system of differential invariants. Indeed, the moving frame provides us with a normalized form for each LPDO of the specified symbol type under the action of the gauge group. This is illustrated in Figure 1.

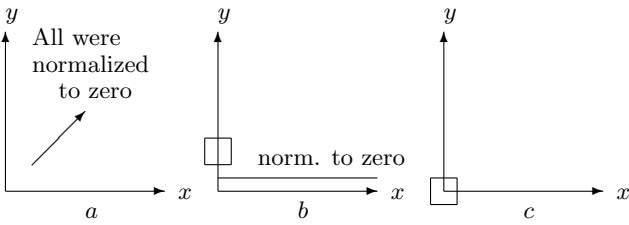


Figure 1: Three copies of $\mathbb{N}^2$, one for each independent variable a, b , and c showing which derivatives have been normalized. The $\square$ indicates a generating invariant.

To see that a complete system is obtained from the restrictions of $\tilde{b}_y$ and $\tilde{c}$ on the frame it is necessary to check that all the remaining invariantized conditions can be obtained by differentiating I_y^b and I^c , and then use the recurrence formulae given by Olver and Pohjanpelto [16]. Since the normalization equations imply $g_x = -b$, $g_y = -a$, $g_{xy} = -a_x$, we compute

$$\begin{aligned} I_y^b &= \tilde{b}_y|_{\text{frame}} = b_y - a_x , \\ I^c &= \tilde{c}|_{\text{frame}} = c + a(-b) + b(-a) - a_x + ab = c - ab - a_x . \end{aligned}$$

□

REMARK 4.3. Another complete generating system for the operators of the form (1) consists of the Laplace invariants $\{h, k\}$, and can be expressed in terms of the new ones as

$$h = I^c , \quad k = I^c - I_y^b .$$

Similar reasonings and considerations imply a complete generating set of invariants for the case of non-hyperbolic second order operators, which in appropriate variables has the normalized form

$$D_{xx} + a_{10}(x, y)D_x + a_{01}(x, y)D_y + a_{00}(x, y) . \quad (7)$$

Below we omit the designation of the dependence of the coefficients on x and y .

THEOREM 4.4. The action of the gauge group on operators of the form (1) has a_{01} as one invariant. For equivalence classes with $a_{01} \neq 0$, a generating system of invariants

is

$$\begin{aligned} I^{a_{01}} &= a_{01} , \\ I_x^{a_{00}} &= a_{00x} - a_{10}a_{10x}/2 - a_{01}a_{10y}/2 - a_{10xx}/2 \\ &\quad + (a_{01x}a_{10}^2/4 - a_{01x}a_{00} + a_{01x}a_{10x}/2)/a_{01} . \end{aligned}$$

The equivalence classes with $a_{01} = 0$ has a generating system of invariants consisting of the one invariant

$$I^{a_{00}} = a_{00} - a_{10}^2/4 - a_{10x}/2 .$$

PROOF. The gauge transformations act on the coefficients of the operators (7) as follows:

$$\begin{cases} \widetilde{a_{00}} &= a_{00} + a_{10}g_x + a_{01}g_y + g_{xx} + g_x^2 , \\ \widetilde{a_{10}} &= a_{10} + 2g_x , \\ \widetilde{a_{01}} &= a_{01} . \end{cases}$$

The moving frame expressions for parameters $(g_x)_J$, $J \in \mathbb{N}_0^2$ can be found from the normalization equations

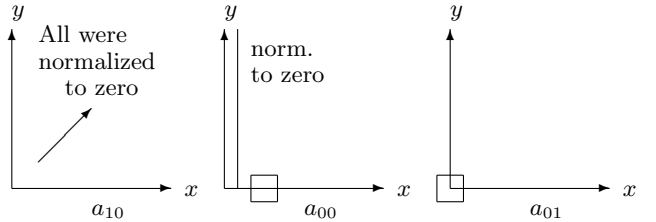
$$\widetilde{a_{10J}} = 0 , \quad J \in \mathbb{N}_0^2 . \quad (8)$$

Then one substitutes into the remaining transformation formulae. Next note that the group parameters $(g_y)_I$, where $I \in \{0\} \times \mathbb{N}_0$ occur in the transformation formulae multiplied by a_{01} and its derivatives. Therefore, we are forced to consider cases.

The first case is $a_{01} \neq 0$. Then we can obtain the moving frame expressions for the remaining parameters as above, from the normalizing equations

$$\widetilde{a_{00I}} = 0 , \quad I \in \{0\} \times \mathbb{N}_0 . \quad (9)$$

Thus, we can write down the moving frame explicitly, and the diagram



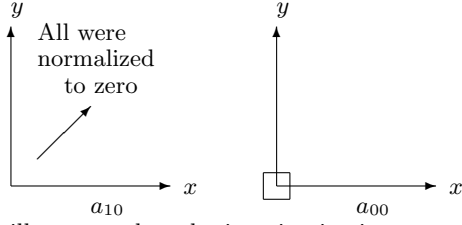
illustrates that the invariantization process gives us a complete generating system of invariants $\{\tilde{a}_{01}|_{\text{frame}}, \tilde{a}_{00x}|_{\text{frame}}\}$. Again, we rely on results of Olver and Pohjanpelto that the remaining invariantized coordinates can be obtained using derivatives of these two using the recurrence formulae.

Namely, the normalization equations (8) imply the following expressions for parameters:

$$\begin{aligned} g_x &= -a_{10}/2 , \quad g_{xx} = -a_{10x}/2 , \quad g_{xy} = -a_{10y}/2 , \\ g_{xxx} &= -a_{10xx}/2 , \quad g_{xxy} = -a_{10xy}/2 . \end{aligned} \quad (10)$$

Where from (9) for $I = (0, 0)$, we have $g_y = -(a_{00} + a_{10}g_x + g_{xx} + g_x^2)/a_{01} = (-a_{00} + a_{10}^2/4 + a_{10x}/2)/a_{01}$, $g_{yy} = -(a_{00y} + a_{10y}g_x + a_{10}g_{xy} + a_{01y}g_y + g_{xxy} + 2g_xg_{xy})/a_{01} = (-4a_{00y}a_{01} + 2a_{10}a_{10y}a_{01} + 4a_{01y}a_{00} - a_{01y}a_{10}^2 - 2a_{01y}a_{10x} + 2a_{10xy}a_{01})/(4a_{01}^2)$. Then we can compute $\tilde{a}_{01}|_{\text{frame}} = I^{a_{01}}$ and $\tilde{a}_{00x}|_{\text{frame}} = I_x^{a_{00}}$.

The second case is $a_{01} = 0$. Then no $(g_y)_I$, where $I \in \{0\} \times \mathbb{N}_0$ occurs in the transform formulae, and, therefore, the moving frame expressions for these parameters can be chosen arbitrarily. The diagram is now



and illustrates that the invariantization process gives us a complete generating system of invariants which consists of the only invariant $\tilde{a}_{00}|_{\text{frame}}$. Employing the moving frame expressions (10), which are the same for the two cases $a_{01} = 0$ and $a_{01} \neq 0$, we have

$$I^{a_{00}} = \tilde{a}_{00}|_{\text{frame}} = a_{00} + a_{10}g_x + g_x^2 = a_{00} - a_{10}^2/4 - a_{10}x/2.$$

□

REMARK 4.5. *Non-hyperbolic operators are particularly difficult to treat with generalized Laplace transformation methods.*

5. MOVING FRAME METHODS FOR LPDOS OF ORDER 3

First, consider the case of hyperbolic operators.

REMARK 5.1. *In all the proofs below we omit repetitions, that is the description in details of those steps that are analogous to ones in the previously considered cases.*

THEOREM 5.2 (Sym = $XY(pX + qY)$). *Given non-zero $p, q \in K$, operators of the form (2) have the following complete generating set of invariants:*

$$\begin{aligned} I_y^{a_{02}} &= a_{02y} - \frac{1}{q}q_y a_{02} - q \frac{1}{p}a_{20x} + q \frac{1}{p^2}p_x a_{20}, \\ I^{a_{11}} &= a_{11} - 2p \frac{1}{q}a_{02} - 2q \frac{1}{p}a_{20}, \\ I^{a_{10}} &= a_{10} - \frac{1}{p}a_{11}a_{20} - 2a_{20x} + \left(\frac{2p_x}{p} + \frac{qp_y}{p^2}\right)a_{20} \\ &\quad - \frac{q}{p}a_{20y} + \frac{q}{p^2}a_{20}^2, \\ I^{a_{01}} &= a_{01} - a_{11}a_{02} \frac{1}{q} - \frac{p}{q}a_{02x} + \frac{pq_x}{q^2}a_{02} + \frac{p}{q^2}a_{02}^2 \\ &\quad - \frac{2q}{p}a_{20x} + \frac{2qp_x}{p^2}a_{20}, \\ I^{a_{00}} &= a_{00} - \frac{q}{p}a_{20xy} - a_{20xx} + \frac{qp_x}{p^2}a_{20y} \\ &\quad - \frac{1}{q}a_{10}a_{02} - \frac{1}{p}a_{01}a_{20} - \frac{1}{p}a_{11}a_{20x} + \\ &\quad + \left(\frac{qp_{xy}}{p^2} - \frac{2p_x^2}{p^2} + \frac{p_{xx}}{p} - \frac{2qp_y p_x}{p^3}\right)a_{20} - \frac{2qp_x}{p^3}a_{20}^2 \\ &\quad + \left(\frac{2p_x}{p} + \frac{qp_y}{p^2}\right)a_{20x} + \frac{p_x}{p^2}a_{11}a_{20} + \frac{1}{pq}a_{11}a_{20}a_{02} \\ &\quad + \frac{2}{q}a_{02}a_{20x} - \frac{2p_x}{pq}a_{02}a_{20} + \frac{2q}{p^2}a_{20}a_{20x}. \end{aligned}$$

PROOF. Consider the gauge transformation group acting on the operators of the family of hyperbolic bivariate third-order operators (2) (in the normalized form). The transform formulae corresponding to the gauge transformations with

$f = \exp(g(x, y))$ are as follows: $\tilde{x} = x$, $\tilde{y} = y$, $\tilde{p} = p$, $\tilde{q} = q$, and

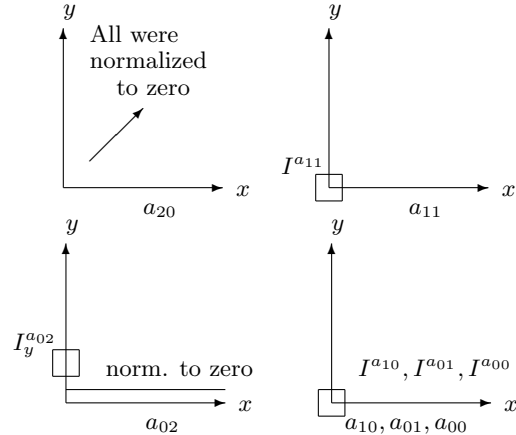
$$\begin{cases} \tilde{a}_{20} = a_{20} + pg_y, & \tilde{a}_{11} = a_{11} + 2pg_x + 2qg_y, \\ \tilde{a}_{02} = a_{02} + qg_x, \\ \tilde{a}_{10} = a_{10} + 2a_{20}g_x + a_{11}g_y + 2pg_{xy} + 2pg_yg_x + qg_{yy} \\ \quad + qg_y^2, \\ \tilde{a}_{01} = a_{01} + a_{11}g_x + 2a_{02}g_y + pg_{xx} + pg_x^2 + 2qg_{xy} \\ \quad + 2qg_yg_x, \\ \tilde{a}_{00} = a_{00} + a_{10}g_x + a_{01}g_y + a_{20}g_{xx} + a_{20}g_x^2 + a_{11}g_{xy} \\ \quad + a_{02}g_{yy} + a_{02}g_y^2 + pg_{xy} + 2pg_{xy}g_x + pg_yg_{xx} \\ \quad + pg_yg_x^2 + a_{11}g_yg_x + qg_xg_{yy} + qg_xg_y^2 \\ \quad + qg_{xyy} + 2qg_yg_{xy}. \end{cases}$$

Reasoning as in Section 4, we construct explicitly a moving frame by the normalization procedure. Thus, we choose normalization equations so that they will be convenient for future computations, and provide equations for all parameters g_J , $J \in \mathbb{N}_0^2$ of the prolonged action.

Since $p \neq 0$ and $q \neq 0$, the formulae for $\tilde{a}_{20}$ and $\tilde{a}_{02}$ are analogous to those for $\tilde{a}$ and $\tilde{b}$ appearing in the proof of Theorem 4.1, and the normalization and invariantization procedures can be done also analogously. Namely, the normalization equations

$$\tilde{a}_{20,J} = 0, J \in \mathbb{N}_0^2, \quad \tilde{a}_{02,I} = 0, \forall I \in \mathbb{N}_0 \times \{0\} \quad (11)$$

imply the equations for all the parameters g_J . Then the diagram



illustrates that the restrictions of the coefficients $\tilde{a}_{02y}$, $\tilde{a}_{11}$, $\tilde{a}_{10}$, $\tilde{a}_{01}$, and $\tilde{a}_{00}$ on the frame form a generating system of (differential) invariants. To compute them, we find the expressions of the group parameters from the normalization equations (11): $g_y = -a_{20}/p$, $g_x = -a_{02}/q$, $g_{xy} = p_x a_{20}/p^2 - a_{20x}/p$, $g_{yy} = p_y a_{20}/p^2 - a_{20y}/p$, $g_{xx} = q_x a_{02}/q^2 - a_{02x}/q$, $g_{xyy} = -a_{20xy}/p + p_{xy} a_{20}/p^2 + p_y a_{20x}/p^2 - 2p_x p_y a_{20}/p^3 + p_x a_{20y}/p^2$, $g_{xxy} = -a_{20xx}/p + p_{xx} a_{20}/p + 2p_x a_{20x}/p^2 - 2p_x^2 a_{20}/p^3$, and substituting these expressions into the transform at a formulae for $\tilde{a}_{02y}$, $\tilde{a}_{11}$, $\tilde{a}_{10}$, $\tilde{a}_{01}$, and $\tilde{a}_{00}$, we complete the proof of the theorem, once again invoking the recurrence formulae of Olver and Pohjanpelto [15]. □

REMARK 5.3. *This theorem demonstrates the approach, but the statement of the theorem is not of particular value by itself, because another complete generating system for hyperbolic bivariate LPDOs of order three has been determined before (see Theorem 3.1). These previously obtained generators can be written in terms of those obtained by the moving*

frame method by the Fels-Olver-Thomas replacement theorem.

Assuming that such symbols of considered LPDOs are completely factorable, there is such a system of coordinates such that the symbol of a non-hyperbolic LPDO of order three has form either X^3 or X^2Y . No complete generating system of invariants has been known for such operators before.

THEOREM 5.4 (Sym = X^3). *Consider operators*

$$L = D_x^3 + \sum_{i+j=0}^2 a_{ij} D_x^i D_y^j, \quad a_{ij} = a_{ij}(x, y). \quad (12)$$

Then a_{02} and a_{11} are invariants. Depending on the equality or non-equality to zero of those, we consider three types of equivalent classes of such operators.

1. For classes with the property $a_{02} \neq 0$, then the invariants a_{02} and a_{11} together with three invariants

$$\left. \begin{aligned} I^{a_{10}} &= a_{10} - \frac{a_{01}a_{11}}{2a_{02}} - \frac{a_{20}^2}{3} + \frac{a_{11}^2 a_{20}}{6a_{02}} - a_{20x}, \\ I_x^{a_{01}} &= a_{01x} + \frac{a_{02x}(a_{11}a_{20} - 3a_{01})}{3a_{02}} - \frac{(a_{11}a_{20})_x - 2a_{02}a_{20y}}{3}, \\ I^{a_{00}} &= a_{00} + a_{11}a_{20}(a_{01} - \frac{1}{6}a_{11}a_{20})/(6a_{02}) \\ &\quad - \frac{1}{3}a_{10}a_{20} - \frac{1}{4}a_{01}^2 a_{02} + \frac{2}{27}a_{20}^3 \\ &\quad - \frac{a_{01y}}{2} + \frac{a_{11y}a_{20} - a_{11}a_{20y}}{6} - \frac{a_{20xx}}{3} \\ &\quad + a_{02y}(\frac{1}{21}a_{01} - \frac{1}{61}a_{11}a_{20})/a_{02}. \end{aligned} \right\} \quad (13)$$

form a generating set of invariants.

2. For classes with the two properties $a_{02} = 0$ and $a_{11} \neq 0$, the invariant a_{11} together with the three invariants

$$\left. \begin{aligned} I_x^{a_{10}} &= a_{10x} - \frac{2a_{20}a_{20x} + a_{11}a_{20y}}{3} - a_{20xx} \\ &\quad + a_{11x}(\frac{1}{3}a_{20}^2 - a_{10} + a_{20x})/a_{11}, \\ I^{a_{01}} &= a_{01} - \frac{1}{3}a_{11}a_{20}, \\ I^{a_{00}} &= a_{00} + a_{01}(a_{20x} - a_{10})/a_{11} - \frac{a_{20}^3}{27} - \\ &\quad \frac{a_{11}a_{20y} + a_{20}a_{20x} + a_{20xx} + a_{11}a_{01}a_{20}}{3}. \end{aligned} \right\} \quad (14)$$

form a generating set of invariants.

3. For classes with the two properties $a_{02} = 0$ and $a_{11} = 0$, a_{01} is an invariant. If $a_{01} \neq 0$, then three invariants

$$\left. \begin{aligned} I^{a_{10}} &= a_{10} - \frac{1}{3}a_{20}^2 - a_{20x}, \\ I^{a_{01}} &= a_{01}, \\ I_x^{a_{00}} &= a_{00x} - \frac{1}{3}(a_{10}a_{20} + a_{20xx})_x \\ &\quad + \frac{a_{01x}(a_{10}a_{20} + a_{20xx} - \frac{2}{9}a_{20}^3 - 3a_{00})}{3a_{01}} \\ &\quad - \frac{1}{3}a_{01}a_{20y} + \frac{2}{9}a_{20}^2 a_{20x}. \end{aligned} \right\} \quad (15)$$

form a generating set of invariants. If $a_{01} = 0$, then two invariants

$$\left. \begin{aligned} I^{a_{10}} &= a_{10} - \frac{1}{3}a_{20}^2 - a_{20x}, \\ I_x^{a_{00}} &= a_{00} - \frac{1}{3}a_{10}a_{20} + \frac{2}{27}a_{20}^3 - \frac{1}{3}a_{20xx}. \end{aligned} \right\} \quad (16)$$

form a generating set of invariants.

PROOF. The gauge transformations imply the following action on the coefficients of (12) and independent variables:

$\tilde{x} = x$, $\tilde{y} = y$, and

$$\left\{ \begin{aligned} \tilde{a}_{11} &= a_{11}, \quad \tilde{a}_{02} = a_{02}, \\ \tilde{a}_{10} &= a_{10} + 2a_{20}g_x + a_{11}g_y + 3g_{xx} + 3g_x^2, \\ \tilde{a}_{00} &= a_{00} + a_{10}g_x + a_{01}g_y + a_{20}(g_{xx} + g_x^2) + \\ &\quad a_{11}(g_{xy} + g_yg_x) + a_{02}(g_{yy} + g_y^2) + \\ &\quad g_{xxx} + 3g_{xx}g_x + g_x^3, \\ \tilde{a}_{20} &= a_{20} + 3g_x, \\ \tilde{a}_{01} &= a_{01} + a_{11}g_x + 2a_{02}g_y. \end{aligned} \right.$$

One readily sees two invariants $I^{a_{02}} = a_{02}$ and $I^{a_{11}} = a_{11}$. Then taking normalization equations

$$\tilde{a}_{20J} = 0, \quad J \in \mathbf{N}_0^2,$$

we can find expressions for all the parameters g_{xJ} . In order to determine those for the remaining parameters, we have to treat a situation similar to that we have in the proof of Theorem 4.4. Namely, none of the extended action transform formulae contains parameters $(g_y)I$, $I \in \{0\} \times \mathbf{N}_0$ without a multiplying coefficient. We treat this difficulty by considering cases. For example, the normalization equations

$$\tilde{a}_{01y \dots y} = 0$$

bring us equations for all $(g_y)I$, provided $a_{02} \neq 0$. In this case, we can have a generating set of invariants as the restriction of $a_{02}, a_{11}, a_{10}, a_{01x}, a_{00}$ on the frame. Thus, we have (13).

Suppose now that $a_{02} = 0$, then the action can be defined as $\tilde{x} = x$, $\tilde{y} = y$, and

$$\left\{ \begin{aligned} \tilde{a}_{11} &= a_{11}, \quad \tilde{a}_{02} = 0, \quad \tilde{a}_{01} = a_{01} + a_{11}g_x, \\ \tilde{a}_{00} &= a_{00} + a_{10}g_x + a_{01}g_y + a_{20}(g_{xx} + g_x^2) + \\ &\quad a_{11}(g_{xy} + g_yg_x) + g_{xxx} + 3g_{xx}g_x + g_x^3, \\ \tilde{a}_{20} &= a_{20} + 3g_x, \\ \tilde{a}_{10} &= a_{10} + 2a_{20}g_x + a_{11}g_y + 3g_{xx} + 3g_x^2. \end{aligned} \right.$$

Once again none of the transform equations contain g_y without a coefficient, so we consider cases. For example, normalization equations

$$(\tilde{a}_{10y})I = 0, \quad I \in \{0\} \times \mathbf{N}_0$$

define the action of all the parameters g_I provided $a_{11} \neq 0$. Thus, we can explicitly define a moving frame, and then the invariantization process outputs a generating system of invariants consisting of the following invariants: $I^{a_{11}} = a_{11}$, and the invariants (14).

Suppose now that $a_{02} = 0$ and $a_{11} = 0$. Then the action is $\tilde{x} = x$, $\tilde{y} = y$, and

$$\left\{ \begin{aligned} \tilde{a}_{11} &= 0, \quad \tilde{a}_{02} = 0, \quad \tilde{a}_{01} = a_{01}, \\ \tilde{a}_{10} &= a_{10} + 2a_{20}g_x + 3g_{xx} + 3g_x^2, \quad \tilde{a}_{20} = a_{20} + 3g_x, \\ \tilde{a}_{00} &= a_{00} + a_{10}g_x + a_{01}g_y + a_{20}(g_{xx} + g_x^2) \\ &\quad + g_{xxx} + 3g_{xx}g_x + g_x^3. \end{aligned} \right.$$

Once again none of the transform equations contain g_y without a parametric multiple, so we consider cases. Normalization equations

$$(\tilde{a}_{00y})I = 0, \quad I \in \{0\} \times \mathbf{N}_0$$

define all the parameters g_I provided $a_{01} \neq 0$. Thus, we can explicitly define a moving frame, and have a complete generating system of invariants consisting of the invariants $I_x^{a_{00}} = a_{00x}|_{\text{frame}}$, $I^{a_{01}} = a_{01}|_{\text{frame}}$, and $I^{a_{10}} = a_{10}|_{\text{frame}}$, and (15) are the explicit expressions of them.

Suppose now that $a_{02} = 0$ and $a_{11} = 0$, and $a_{01} = 0$. Then the action is $\tilde{x} = x$, $\tilde{y} = y$, and

$$\begin{cases} \tilde{a}_{11} = 0, & \tilde{a}_{02} = 0, & \tilde{a}_{01} = 0, & \tilde{a}_{20} = a_{20} + 3g_x, \\ \tilde{a}_{10} = a_{10} + 2a_{20}g_x + 3g_{xx} + 3g_x^2, \\ \tilde{a}_{00} = a_{00} + a_{10}g_x + a_{20}(g_{xx} + g_x^2) \\ \quad + g_{xxx} + 3g_{xx}g_x + g_x^3, \end{cases}$$

Here no $(g_y)_I$, where $I \in \{0\} \times \mathbb{N}_0$ occurs in the transforms formulae, and, therefore, the moving frame expressions for these parameters can be chosen arbitrarily. Thus, we defined a moving frame explicitly, and have a complete generating system of invariants consisting of the invariants $I^{a_{00}} = a_{00}|_{\text{frame}}$ and $I^{a_{10}} = a_{10}|_{\text{frame}}$, and (16) are the explicit expressions of them. $\square$

THEOREM 5.5 (Sym = X^2Y). *For the class of LPDOs*

$$L = D_x^2 D_y + \sum_{i+j=0}^2 a_{ij} D_x^i D_y^j, \quad a_{ij} = a_{ij}(x, y) \quad (17)$$

the following form a complete generating set of invariants.

$$\left. \begin{aligned} I^{a_{02}} &= a_{02}, \\ I_y^{a_{11}} &= a_{11y} - 2a_{20x}, \\ I^{a_{10}} &= a_{10} - a_{20}a_{11} - 2a_{20x}, \\ I^{a_{01}} &= a_{01} - \frac{1}{4}a_{11}^2 - 2a_{02}a_{20} - \frac{1}{2}a_{11x}, \\ I^{a_{00}} &= a_{00} - \frac{1}{2}a_{10}a_{11} - a_{01}a_{20} + \frac{1}{2}a_{20}a_{11}^2 \\ &\quad - a_{02}a_{20y} + a_{02}a_{20}^2 - a_{20xx}. \end{aligned} \right\} \quad (18)$$

PROOF. The gauge transformations with $f = f(x, y) = \exp(g(x, y)) \in K^*$ acting on the coefficients of (17) and independent variables can be defined as $\tilde{x} = x$, $\tilde{y} = y$, and

$$\left\{ \begin{aligned} \tilde{a}_{20} &= a_{20} + g_y, \\ \tilde{a}_{11} &= a_{11} + 2g_x, \\ \tilde{a}_{02} &= a_{02}, \\ \tilde{a}_{10} &= a_{10} + 2a_{20}g_x + a_{11}g_y + 2g_{xy} + 2g_yg_x, \\ \tilde{a}_{01} &= a_{01} + a_{11}g_x + 2a_{02}g_y + g_{xx} + g_x^2, \\ \tilde{a}_{00} &= a_{00} + a_{10}g_x + a_{01}g_y + a_{20}g_{xx} + a_{02}g_x^2 \\ &\quad + a_{11}g_{xy} + a_{11}g_yg_x + a_{02}g_{yy} + a_{02}g_y^2 \\ &\quad + g_{xxx} + 2g_{xy}g_x + g_yg_{xx} + g_yg_x^2. \end{aligned} \right.$$

One can readily see that a_{02} is an invariant, and that we can apply without any additional tricks, normalization procedures choosing the normalization equations

$$\tilde{a}_{20J} = 0, \quad J \in \mathbb{N}_0^2, \quad (\tilde{a}_{11})_I = 0, \quad I \in \mathbb{N}_0 \times \{0\},$$

and then invariantization procedure which results in the generating set of invariants (18). $\square$

6. INVARIANTS OF PAIRS OF OPERATORS

In the sections above generating system of invariants are determined for different classes of operators. Often one also needs a generating system of invariants for a pair of operators, that is when the both operators are under the gauge transformations with the same function. Naturally, invariants of each of the operators are invariants of the pair. However, there can be also invariants that involve coefficients of the both operators, that Olver calls “joint invariants”.

In the present section we determine a complete generating system of invariants for (1, 1)- and (2, 2)-transformations of Darboux [7]. Recall that given an operator $L = D_{xy} + aD_x + bD_y + c$, where all the coefficients are functions of x and y ,

then the operator $L' = D_{xy} + a'D_x + b'D_y + c'$ is said to be the result of a (m, n) -transformation of L if there exist

$$M = \sum_{i=1}^m p_i D_x^i + \sum_{j=1}^n q_j D_y^j + r,$$

and $M' = \sum_{i=1}^m p'_i D_x^i + \sum_{j=1}^n q'_j D_y^j + r'$, where all the coefficients are functions of x and y , such that

$$M \circ L = L' \circ M'. \quad (19)$$

Note that the property of the existence of such transformations is invariant under the gauge transformations. Indeed, if $L \rightarrow L'$, then $g^{-1}Lg \rightarrow g^{-1}L'g$. To describe all or some (m, n) -transformations one can consider transformations of a pair (M, L) into some pair (M', L') , such that (19) holds. To describe such transformations invariantly, we need to determine a complete generating system for the pair of operators M and L .

Consider the pairs involved in the (1, 1)-transformations. Being the simplest case of the (m, n) -transformation, (1, 1)-transformations generalize the classical Laplace transformations. The following statement can be proved.

PROPOSITION 6.1. *For the pair of operators $L = D_{xy} + aD_x + bD_y + c$ and $M = pD_x + qD_y + r$, where all the coefficients are functions of x and y , a generating system of invariants is $I_y^b = b_y - a_x$, $I^c = c - ab - a_x$, $I^p = p$, $I^q = q$, $I^r = r - bp - aq$.*

REMARK 6.2. *Note that invariant I_r depends on the coefficients of the both operators of a pair.*

PROOF. The action of the gauge transformations with $f = \exp(g(x, y))$ is

$$\left\{ \begin{aligned} \tilde{x} &= x, \\ \tilde{y} &= y, \end{aligned} \right. \quad \& \quad \left\{ \begin{aligned} \tilde{a} &= a + g_y, \\ \tilde{b} &= b + g_x, \\ \tilde{c} &= c + ag_x + bg_y + g_{xy} + g_xg_y, \\ \tilde{p} &= p, \quad \tilde{q} = q, \\ \tilde{r} &= r + pg_x + qg_y. \end{aligned} \right.$$

Choosing the normalization equations

$$\tilde{a}_J = 0, \quad \forall J \in \mathbb{N}_0^2, \quad \tilde{b}_I = 0, \quad \forall I \in \mathbb{N}_0 \times \{0\} \quad (20)$$

we get equations for every action parameter g_J . Thus, we can explicitly construct a moving frame. The invariantization process results in the five invariants of the statement of the proposition. $\square$

Consider pairs of LPDOs appearing in the (2, 2)-transformations of Darboux.

PROPOSITION 6.3. *For the pair of operators $L = D_{xy} + aD_x + bD_y + c$ and $M = p_2D_{xx} + p_1D_x + q_2D_{yy} + q_1D_y + r$, where all the coefficients are functions of x and y , a complete generating system of invariants is*

$$\left. \begin{aligned} I_y^b &= b_y - a_x, \\ I^c &= c - ab - a_x, \\ I^{p_2} &= p_2, \\ I^{q_2} &= q_2, \\ I^{p_1} &= p_1 - 2p_2b, \\ I^{q_1} &= q_1 - 2q_2a, \\ I^r &= r - p_1b - q_1a - p_2b_x + p_2b^2 - q_2a_y + q_2a^2. \end{aligned} \right\}$$

REMARK 6.4. *Note that invariants I^{p_1} , I^{q_1} , I_r depend on the coefficients of the both operators of a pair.*

PROOF. The action of the gauge transformations with $f = \exp(g(x, y))$ on the coefficients of the operator M is

$$\begin{cases} \tilde{q}_2 &= q_2, \\ \tilde{p}_1 &= p_1 + 2p_2g_x, \\ \tilde{q}_1 &= q_1 + 2q_2g_y, \\ \tilde{r} &= r + p_1g_x + q_1g_y + p_2g_{xx} + p_2g_x^2 + q_2g_{yy} + q_2g_y^2. \end{cases}$$

The transform action formulae for the coefficients of the operator L are the very same as in the previous proposition, and we have the same normalization equations (20). Then the invariantization process results in the six generating invariants

$$\tilde{b}_y|_{\text{frame}}, \tilde{c}|_{\text{frame}}, \tilde{p}_2|_{\text{frame}}, \tilde{q}_2|_{\text{frame}}, \tilde{p}_1|_{\text{frame}}, \tilde{q}_1|_{\text{frame}}, \tilde{r}|_{\text{frame}}.$$

□

REMARK 6.5. Note that for all the (m, n) -transformations of operators $L = D_{xy} + aD_x + bD_y + c$, where m and n are not necessarily 1 or 2, we can introduce the same moving frame.

7. CONCLUSIONS

We have demonstrated an elegant, efficient and systematic way for obtaining a generating set of invariants for bivariate LPDOs. Explicit formulae of the invariants of generating systems are given for operators of orders 2, 3, hyperbolic and non-hyperbolic. Also for the pairs of operators appearing in the (i, i) -transformations of Darboux, $i = 1, 2$ complete generating set of invariants were obtained.

8. ACKNOWLEDGMENTS

The first author acknowledges support for this work from FWF (Austrian national science fund) under the projects SFB F013/F1304 and thanks her doctoral supervisor Professor Franz Winkler for his continued support and encouragement. The second author would like to thank Professor Franz Winkler and RISC, Linz for their hospitality while the research for this paper was initiated, as well as acknowledge EPSRC UK (Project EP/E001823/1).

9. REFERENCES

- [1] M. Akivis and B. Rosenfeld. Elie cartan (1869-1951). *American Mathematical Society. Translations. Math. Monographs.*, 123, 1993.
- [2] I. Anderson and M. Juras. Generalized Laplace invariants and the method of Darboux. *Duke J. Math.*, 89:351–375, 1997.
- [3] I. Anderson and N. Kamran. The variational bicomplex for hyperbolic second-order scalar partial differential equations in the plane. *Duke J. Math.*, 87:265–319, 1997.
- [4] C. Athorne. A $z \times r$ toda system. *Phys. Lett. A.*, 206:162–166, 1995.
- [5] E. Cartan. *Oeuvres Completes, II*, volume 2, chapter Sur la structure des groupes infinis de transformations, pages 571–714. Gauthier-Villars, Paris, 1953.
- [6] J. Cheh, P. Olver, and J. Pohjanpelto. Maurer–Cartan equations for Lie symmetry pseudo-groups of differential equations. *J. Math. Phys.*, 46, 2005. 023504.
- [7] G. Darboux. *Leçons sur la théorie générale des surfaces et les applications géométriques du calcul infinitésimal*, volume 2. Gauthier-Villars, 1889.
- [8] M. Fels and P. J. Olver. Moving coframes. I. A practical algorithm. *Acta Appl. Math.*, 51(2):161–213, 1998.
- [9] M. Fels and P. J. Olver. Moving coframes. II. Regularization and theoretical foundations. *Acta Appl. Math.*, 55(2):127–208, 1999.
- [10] D. Grigoriev and F. Schwarz. Generalized loewy-decomposition of d-modules. In *ISSAC '05: Proceedings of the 2005 international symposium on Symbolic and algebraic computation*, pages 163–170, New York, NY, USA, 2005. ACM.
- [11] E. Kartashova. A hierarchy of generalized invariants for linear partial differential operators. *TMPH (Journal of Theoretical and Mathematical Physics)*, 147(3):839–846, 2006.
- [12] E. Mansfield. Algorithms for symmetric differential systems. *Found. Comput. Math.*, 1:335–383, 2001.
- [13] O. Morozov. Contact-equivalence problem for linear hyperbolic equations. *Journal of Mathematical Sciences*, 135:2680–2694(15), 2006.
- [14] P. Olver and J. Pohjanpelto. Maurer–Cartan forms and the structure of Lie pseudo-groups. *Selecta Math.*, 11:99–126, 2005.
- [15] P. Olver and J. Pohjanpelto. Moving frames for Lie pseudo-groups. *Canadian J. Math.*, 2007. to appear.
- [16] P. Olver and J. Pohjanpelto. Pseudo-groups, moving frames, and differential invariants. In M. Eastwood and J. W. Miller, editors, *Symmetries and Overdetermined Systems of Partial Differential Equations*, volume 144 of *IMA Volumes in Mathematics and Its Applications*, pages 127–149. Springer-Verlag, New York, 2007.
- [17] E. Shemyakova. A full system of invariants for third-order linear partial differential operators. *Lecture Notes in Comput. Sci.*, 4120:360–369, 2006.
- [18] E. Shemyakova and F. Winkler. A full system of invariants for third-order linear partial differential operators in general form. *Lecture Notes in Comput. Sci.*, 4770:360–369, 2007.
- [19] E. Shemyakova and F. Winkler. Invariant necessary and sufficient conditions of factorizations of hyperbolic bivariate third-order linear partial differential operators. *submitted*, 2007.
- [20] S. Tsarev. Factorization of linear partial differential operators and darboux’ method for integrating nonlinear partial differential equations. *Theo. Math. Phys.*, 122:121–133, 2000.
- [21] S. Tsarev. Generalized laplace transformations and integration of hyperbolic systems of linear partial differential equations. In *ISSAC '05: Proceedings of the 2005 international symposium on Symbolic and algebraic computation*, pages 325–331, New York, NY, USA, 2005. ACM Press.

Real Root Isolation for Exp-Log Functions

Adam Strzeboński
Wolfram Research Inc.
100 Trade Centre Drive
Champaign, IL 61820, U.S.A.
adams@wolfram.com

ABSTRACT

We present a real root isolation procedure for univariate functions obtained by composition and rational operations from exp, log, and real constants. We discuss implementation of the procedure and give empirical results. The procedure requires the ability to determine signs of exp-log functions at simple roots of other exp-log functions. The currently known method to do this depends on Schanuel's conjecture [6].

Categories and Subject Descriptors

I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms—Algebraic algorithms; G.4 [Mathematical Software]: Algorithm design and analysis

General Terms

Algorithms, Experimentation, Reliability, Theory

Keywords

exp-log functions, real root isolation, solving equations

1. INTRODUCTION

DEFINITION 1.1. *The set of exp-log functions is the smallest set of partial functions $\mathbb{R} \rightarrow \mathbb{R}$ containing exp, log, the identity function and the constant functions, closed under addition, multiplication and composition of functions.*

The domain $D(f)$ of an exp-log function f is determined as follows:

1. the domain of exp, the identity function and the constant functions is $\mathbb{R}$ and the domain of log is $\mathbb{R}_+$,
2. $D(f + g) = D(fg) = D(f) \cap D(g)$,
3. $D(f(g)) = g^{-1}(D(f))$.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC '08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

In particular, $D(f)$ is an open set and f is C^∞ in $D(f)$.

REMARK 1.2. *The multiplicative inverse function*

$$\text{inv} : \mathbb{R} \setminus \{0\} \ni x \rightarrow 1/x = x \exp(-\log(x^2)) \in \mathbb{R}$$

and the integer base radical functions

$$\text{rad}_n : \mathbb{R}_+ \ni x \rightarrow \sqrt[n]{x} = \exp(\log(x)/n) \in \mathbb{R}$$

for $n \in \mathbb{Z}_+$, are exp-log functions.

DEFINITION 1.3. *Let $f : \mathbb{R} \supseteq D(f) \rightarrow \mathbb{R}$ and let $a \in D(f)$. a is a root of f iff $f(a) = 0$ and for some $\varepsilon > 0$ if $a - \varepsilon < x < a \vee a < x < a + \varepsilon$ then $x \in D(f) \wedge f(x) \neq 0$.*

The domain of an exp-log function consists of a finite number of open, possibly unbounded, intervals and an exp-log function has a finite number of real roots. This property is well known, since it follows from the fact that the model-theoretic structure $\mathbb{R}_{\text{exp}} := (\mathbb{R}, <, 0, 1, +, \cdot, \exp)$ is o-minimal [2, 10, 8]. The algorithm given in this paper computes domains and isolating intervals for real roots of exp-log functions. The proof of correctness of the algorithm does not use the o-minimality of $\mathbb{R}_{\text{exp}}$ and hence it provides a new proof of the property.

EXAMPLE 1.4. $f : \mathbb{R} \setminus \{a\} \ni x \rightarrow x - a - \sqrt{(x - a)^2} \in \mathbb{R}$ is an exp-log function which is zero at infinitely many points, but has no roots. Note that the continuous extension of f to $\mathbb{R}$ is not an exp-log function, since it is not differentiable at a .

DEFINITION 1.5. *The set of exp-log expressions with coefficients in a computable field $K \subseteq \mathbb{R}$ is defined recursively as follows:*

1. elements of K and the variable x are exp-log expressions,
2. if f and g are exp-log expressions, so are $f + g$, $f \cdot g$ and $\frac{f}{g}$,
3. if f is an exp-log expression, so are $\exp(f)$ and $\log(f)$.

Each exp-log expression represents an exp-log function, however the same function may be represented by many different expressions. In the following, when we refer to the domain, point values and roots of an exp-log expression, we mean the domain, point values and roots of the corresponding exp-log function.

DEFINITION 1.6. Two exp-log expressions are equivalent if they represent the same exp-log function. An exp-log function (expression) g extends an exp-log function (expression) f if $D(f) \subseteq D(g)$ and $\forall_{x \in D(f)} f(x) = g(x)$.

EXAMPLE 1.7. $\exp(x) - \exp(-x)$ is equivalent to $\frac{\exp(2x) - 1}{\exp(x)}$. $\log(x^2)$ extends $2\log(x)$.

For an exp-log expression f , f' denotes the expression obtained from f by applying the rules of differentiation with respect to x . The function represented by f' extends the derivative of the function represented by f .

DEFINITION 1.8. An exp-log root constant is an expression $\text{Root}(f, a, b)$, where f is an exp-log expression, $a < b$ are rational numbers, $[a, b] \subseteq D(f)$, $f(a)f(b) < 0$ and f' has a constant nonzero sign on (a, b) . An exp-log root constant $\text{Root}(f, a, b)$ represents the only root of f in (a, b) . In the following we identify exp-log root expressions with the roots they represent.

In this paper we present a procedure which given an exp-log function finds its domain, roots and intervals on which the function is identically zero. The returned roots and interval endpoints are elements of K or exp-log root constants.

The assumption that K is a computable field implies that there is an algorithm for computation of signs of elements of K . However, the root isolation procedure requires the ability to determine signs of constants obtained by evaluation of exp-log functions at arbitrary exp-log root constants. At present, there is a known method to do this only when K is contained in the field of real elementary constants ([6], see section 3). The proof that the method terminates relies on Schanuel's conjecture [3]. When K is not contained in the field of elementary constants, the root isolation procedure may fail if a call to the sign determination heuristic it uses fails.

In the following sections we introduce the notion of semi Fourier sequence, give an algorithm computing semi Fourier sequences for exp-log functions, discuss the issue of sign testing, give a description of the main root finding procedure and finally discuss our implementation of the procedure and report some empirical results.

2. SEMI FOURIER SEQUENCES

DEFINITION 2.1. A sequence $f_1, \dots, f_k : I \rightarrow \mathbb{R}$ of differentiable functions defined in an open interval $I \subseteq \mathbb{R}$ is a weak Fourier sequence if for all $1 \leq i \leq k-1$ and all $x \in I$

$$\text{sign}(f_{i+1}(x)) = \text{sign}(f'_i(x)) \quad (1)$$

and

$$f_k(x) \neq 0$$

PROPOSITION 2.2. Let $f_1, \dots, f_k : (a, b) \rightarrow \mathbb{R}$, where $a, b \in \mathbb{R} \cup \{-\infty, \infty\}$, be a weak Fourier sequence. Then, for any $1 \leq i \leq k$,

1. f_i has at most $k-i$ distinct roots in (a, b) ,
2. $\forall_{a \leq x < b} \exists_{x < c < b} \forall_{x < y \leq c} f(y) \neq 0$,
3. $\forall_{a < x \leq b} \exists_{a < d < x} \forall_{d \leq y < x} f(y) \neq 0$.

PROOF. The proposition follows by induction from Rolle's theorem. $\square$

NOTATION 2.3. Let $f_1, \dots, f_k : (a, b) \rightarrow \mathbb{R}$ be a weak Fourier sequence. Then, for any $1 \leq i \leq j \leq k$,

1. $\text{sign}(f_i(x^+)) := \text{sign}(f_i(c))$, where $a \leq x < b$ and $\forall_{x < y \leq c} f_i(y) \neq 0$,
2. $\text{sign}(f_i(x^-)) := \text{sign}(f_i(d))$, where $a < x \leq b$ and $\forall_{d \leq y < b} f_i(y) \neq 0$,
3. $\text{sgc}_{i,j}(x)$, for $a < x < b$, is number of sign changes in the sequence

$$f_i(x), \dots, f_j(x)$$

with terms equal to zero removed,

4. $\text{sgc}_{i,j}(x^+)$, for $a \leq x < b$, is number of sign changes in the sequence

$$f_i(x^+), \dots, f_j(x^+)$$

5. $\text{sgc}_{i,j}(x^-)$, for $a < x \leq b$, is number of sign changes in the sequence

$$f_i(x^-), \dots, f_j(x^-)$$

6. $\text{sgc}(x) := \text{sgc}_{1,k}(x)$, $\text{sgc}(x^+) := \text{sgc}_{1,k}(x^+)$, $\text{sgc}(x^-) := \text{sgc}_{1,k}(x^-)$.

THEOREM 2.4. Let $I \subseteq \mathbb{R}$ be an open interval and let $f_1, \dots, f_k : I \rightarrow \mathbb{R}$ be a weak Fourier sequence. Then for any $x \in I$

$$\text{sgc}(x^+) = \text{sgc}(x)$$

and

$$\text{sgc}(x^-) = \text{sgc}(x) + r + 2s$$

where $r, s \in \mathbb{Z}_+ \cup \{0\}$ and

$$f_1(x) = \dots = f_r(x) = 0, f_{r+1}(x) \neq 0$$

Moreover, $s = 0$ unless there is $t > r+1$ such that $f_t(x) = 0$.

PROOF. Note that if

$$i_0 = 1 < i_1 < \dots < i_m < i_{m+1} = k$$

then for any $y \in I$

$$\text{sgc}(y) = \sum_{l=0}^m \text{sgc}_{i_l, i_{l+1}}(y) \quad (2)$$

If $f_l(x) \neq 0$ for all $i \leq l \leq j$, then clearly $\text{sgc}_{i,j}$ is constant in a neighbourhood of x .

Now suppose that

$$f_p(x) = \dots = f_q(x) = 0, f_{q+1}(x) \neq 0$$

Conditions (1) imply that there is a neighbourhood U of x in which none of f_i , $p \leq i \leq q+1$, has zeroes other than x , and we have one of the following four sign combinations uniquely determined by the signs of f_{q+1} in U .

	$y < x$	x	$y > x$	$y < x$	x	$y > x$
f_p	-	0	+	+	0	-
f_{p+1}	+	0	+	-	0	-
f_{p+2}	-	0	+	+	0	-
$\dots$						
f_q	-	0	+	+	0	-
f_{q+1}	+	+	+	-	-	-

	$y < x$	x	$y > x$	$y < x$	x	$y > x$
f_p	+	0	+	-	0	-
f_{p+1}	-	0	+	+	0	-
f_{p+2}	+	0	+	-	0	-
$\dots$						
f_q	-	0	+	+	0	-
f_{q+1}	+	+	+	-	-	-

If $p = 1$, then, for $y \in U$, $sgc_{1,q+1}(y) = q$ if $y < x$ and $sgc_{1,q+1}(y) = 0$ if $y \geq x$.

If $p > 1$, and $f_{p-1}(x) \neq 0$, then there is a neighbourhood $V \subseteq U$ of x , such that $f_{p-1}(y) \neq 0$ for $y \in V$. If $y \in V$, and we have one of the first two sign combinations, then $q - p$ is even and, depending on the sign of f_{p-1} in V , either

$$sgc_{p-1,q+1}(y) = \begin{cases} q - p + 2, & \text{for } y < x \\ 0, & \text{for } y \geq x \end{cases}$$

or

$$sgc_{p-1,q+1}(y) = \begin{cases} q - p + 1, & \text{for } y < x \\ 1, & \text{for } y \geq x \end{cases}$$

If $y \in V$, and we have one of the last two sign combinations, then $q - p$ is odd and, depending on the sign of f_{p-1} in V , either

$$sgc_{p-1,q+1}(y) = \begin{cases} q - p + 2, & \text{for } y < x \\ 1, & \text{for } y \geq x \end{cases}$$

or

$$sgc_{p-1,q+1}(y) = \begin{cases} q - p + 1, & \text{for } y < x \\ 0, & \text{for } y \geq x \end{cases}$$

The discussion above combined with equation (2) proves the theorem. $\square$

Note that, unlike for polynomial derivative sequences, there is no relation between r and the multiplicity of the root of f_1 at x .

DEFINITION 2.5. A sequence $\frac{g_1}{h_1}, \dots, \frac{g_m}{h_m} : U \rightarrow \mathbb{R}$ of functions defined in an open set $U \subseteq \mathbb{R}$ is a semi Fourier sequence for f if $U = D(f)$ and for all $x \in U$

1. h_i is continuous and $h_i(x) \neq 0$ for all $1 \leq i \leq m$,
2. $\text{sign}(\frac{g_1(x)}{h_1(x)}) = \text{sign}(f(x))$,
3. g_i is differentiable and $\text{sign}(\frac{g_{i+1}(x)}{h_{i+1}(x)}) = \text{sign}(g'_i(x))$ for all $1 \leq i \leq m - 1$,
4. g_m is constant in each interval contained in U .

COROLLARY 2.6. Let $\frac{g_1}{h_1}, \dots, \frac{g_m}{h_m}$ be a semi Fourier sequence for f and let $a < b$ be such that $(a, b) \subseteq U$. Functions $h_1, \dots, h_m$ have constant nonzero signs $s_1, \dots, s_m$ in (a, b) . For $1 \leq i \leq m$ put

$$f_i := s_1 \cdot \dots \cdot s_i \cdot g_i$$

so that $\text{sign}(f_{i+1}(x)) = \text{sign}(f'_i(x))$ for all $a < x < b$. Let k be the largest index for which f_k is not identically zero on (a, b) or 0 if such index does not exist. If $k > 0$, let $1 \leq p \leq k$. Then

1. $f_{k+1}, \dots, f_m$ are identically zero in I ,
2. if $k > 0$, f_k is nonzero and constant in I ,

3. $\text{sign}(f(x)) = \text{sign}(f_1(x))$ for all $x \in I$,
4. $sgc_{p,k}(a^+) \geq sgc_{p,k}(b^-)$,
5. if $sgc_{p,k}(a^+) = sgc_{p,k}(b^-)$, then f_p has a constant nonzero sign in (a, b) ,
6. if $sgc_{p,k}(a^+) = sgc_{p,k}(b^-) + 1$, then f_p has exactly one simple root in (a, b) .

PROOF. The first three statements are obvious. By Theorem 2.4 applied to $f_p, \dots, f_k$, the function $sgc_{p,k}$ is decreasing and right-continuous, which proves (4). To prove (5), note that the value of $sgc_{p,k}$ decreases by at least one when passing through a root of f_p . Finally, keeping the notation of Theorem 2.4, the value of $sgc_{p,k}$ decreases by one when passing through x iff $r = 1$ and $s = 0$. This proves (6), since $r = 1$ exactly when x is a simple root of f_p . $\square$

The following algorithm finds a semi Fourier sequence for an arbitrary exp-log expression. The construction is similar to that of semi Budan sequences in [5], except that here we keep track of factors removed in simplification steps.

Let us introduce terminology and subalgorithms used in the description of the algorithm.

DEFINITION 2.7. The rank of an exp-log expression is defined recursively as follows:

1. Elements of K and the variable x have rank 0,
2. $\text{rank}(f + g) = \text{rank}(fg) = \text{rank}(\frac{f}{g}) := \max(\text{rank}(f), \text{rank}(g))$,
3. $\text{rank}(\exp(f)) = \text{rank}(\log(f)) = \text{rank}(f) + 1$.

The log-content of an exp-log expression f is defined as follows:

1. If $\text{rank}(f) > 0$, $\text{logcont}(f)$ is the number of distinct subexpressions $\log(p_i)$ of f with $\text{rank}(p_i) = \text{rank}(f) - 1$,
2. If $\text{rank}(f) = 0$, $\text{logcont}(f) := 0$.

A log-type expression is an exp-log expression f with

$$\text{logcont}(f) > 0$$

Let χ be a choice algorithm which for any finite set of expressions deterministically picks one of them. The leading logarithm of a log-type expression f is the result of applying χ to the set of subexpressions of f of the form $\log(p)$ with $\text{rank}(p) = \text{rank}(f) - 1$.

An exp-type expression is an exp-log expression f with

$$\text{rank}(f) > 0 \wedge \text{logcont}(f) = 0$$

A poly-type expression is an exp-log expression with rank 0.

An exp-log expression g is normal if one of the following three conditions holds:

1. $g = a_n \log(p)^n + \dots + a_1 \log(p) + a_0$, where g is a log-type expression, $\log(p)$ is the leading logarithm of g , $n > 0$ is the degree of g , the leading coefficient $a_n \neq 0$ is normal and $a_0, \dots, a_n$ do not contain $\log(p)$,

2. $g = a_n + a_{n-1} \exp(f_{n-1}) + \dots + a_0 \exp(f_0)$, where g is an exp-type expression,

$$\text{rank}(f_0) = \dots = \text{rank}(f_{n-1}) = \text{rank}(g) - 1$$

$n > 0$ is the degree of g , the leading coefficient $a_n \neq 0$ is normal and $\text{rank}(a_0), \dots, \text{rank}(a_n) < \text{rank}(g)$,

3. $g = a_n x^n + \dots + a_1 x + a_0$, where $n \geq 0$ is the degree of g , the leading coefficient $a_n \neq 0$ and $a_0, \dots, a_n \in K$.

The set $TLVS(f)$ of top level variable subexpressions of an exp-log expression f is defined recursively as follows:

1. If $c \in K$, $TLVS(c) := \emptyset$,
2. $TLVS(x) := \{x\}$,
3. $TLVS(f+g) = TLVS(f \cdot g) := TLVS(f) \cup TLVS(g)$,
4. $TLVS(\frac{f}{g}) := TLVS(f) \cup \{g\}$,
5. $TLVS(\exp(f)) := \{\exp(f)\}$,
6. $TLVS(\log(f)) := \{\log(f)\}$.

ALGORITHM 2.8. (*NormalFraction*)

Input: An exp-log expression f .

Output: An exp-log expression $\frac{g}{h}$, such that

- $\frac{g}{h}$ extends f ,
 - g is normal,
 - h is a product of expressions that appear as factors of denominators of subexpressions of f and of an expression $\exp(m_1 p_1 + \dots + m_k p_k)$, where $\exp(p_1), \dots, \exp(p_k)$ appear as subexpressions in f and $m_1, \dots, m_k \in \mathbb{Z}$.
1. Let f_{rat} be the expression obtained from f by replacing all elements of $TLVS(f)$ with new symbolic variables $\{y_1, \dots, y_l\}$, so that identical subexpressions are replaced with the same variables. f_{rat} is a rational expression in $\{x, y_1, \dots, y_l\}$ such that all denominators of fractions in f_{rat} are variables.
 2. Let $\frac{g_{\text{rat}}}{h_{\text{rat}}}$ be the representation of f_{rat} as a two-level fraction such that g_{rat} and h_{rat} are relatively prime polynomials and h_{rat} is a product of variables replacing denominators of fraction subexpressions of f .
 3. Replace variables $\{y_1, \dots, y_l\}$ in g_{rat} and h_{rat} with the corresponding subexpressions of f , obtaining exp-log expressions g_1 and h_1 such that $\frac{g_1}{h_1}$ extends f .
 4. If g_1 is a log-type expression, g_1 is a polynomial in its leading logarithm $\log(p)$.

(a) Collect terms of g_1 with respect to $\log(p)$ to obtain $a_{n,1} \log(p)^n + \dots + a_{1,1} \log(p) + a_{0,1}$.

(b) Compute $\frac{a_n}{h_2} := \text{NormalFraction}(a_{n,1})$.

(c) While $a_n = 0$ and $n > 0$, set $n := n - 1$ and compute

$$\frac{a_n}{h_2} := \text{NormalFraction}(a_{n,1})$$

(d) If $n > 0$, put $g := a_n \log(p)^n + \dots + a_{1,1} h_2 \log(p) + a_{0,1} h_2$, else put $g := a_0$.

(e) Put $h := h_1 h_2$ and return $\frac{g}{h}$.

5. If g_1 is an exp-type expression, g_1 is a polynomial in all exponentials of rank equal to $\text{rank}(g_1)$.

(a) Collect terms of g_1 with respect to exponentials of rank equal to $\text{rank}(g_1)$ to obtain

$$a_{n,1} + a_{n-1,1} \exp(f_{n-1}) + \dots + a_{0,1} \exp(f_0)$$

(b) Put $f_n := 0$ and compute

$$\frac{a_n}{h_2} := \text{NormalFraction}(a_{n,1})$$

(c) While $a_n = 0$ and $n > 0$, set $n := n - 1$ and compute

$$\frac{a_n}{h_2} := \text{NormalFraction}(a_{n,1})$$

(d) If $n > 0$, put $g := a_n + a_{n-1,1} h_2 \exp(f_{n-1} - f_n) + \dots + a_{0,1} h_2 \exp(f_0 - f_n)$, else put $g := a_0$.

(e) Put $h := h_1 h_2 \exp(-f_n)$ and return $\frac{g}{h}$.

6. g_1 is a poly-type expression. Collect terms of g_1 with respect to x obtaining $g = a_n x^n + \dots + a_1 x + a_0$, where $n \geq 0$ is the degree of g , the leading coefficient $a_n \neq 0$ and $a_0, \dots, a_n \in K$.

7. Return $\frac{g}{h_1}$.

Note that in all recursive calls the input to *NormalFraction* has either a lower rank or the same rank and a lower log-content than f , hence the recursion terminates. It is easy too see that the output of *NormalFraction* has the required properties.

DEFINITION 2.9. The signature of a normal exp-log expression g is the expression $\text{sig}(g) := (r, l, n, s)$, where $r = \text{rank}(g)$, $l = \text{logcont}(g)$, n is the degree of g and, if $r > 0$, s is the signature of the leading coefficient of g , otherwise $s = 0$. Since the leading coefficient of g is a proper subexpression of g , the signature is well defined.

LEMMA 2.10. Define the relation $\succeq$ on the set S of all signatures of normal exp-log expressions by

$$(r_1, l_1, n_1, s_1) \succeq (r_2, l_2, n_2, s_2)$$

iff

$$(r_1 > r_2 \vee r_1 = r_2 \wedge (l_1 > l_2 \vee l_1 = l_2 \wedge (n_1 > n_2 \vee n_1 = n_2 \wedge s_1 \succeq s_2)))$$

where $0 \succeq 0$ is defined to be true, and let $s_1 \succ s_2$ denote $s_1 \succeq s_2 \wedge s_1 \neq s_2$. Then

1. $\succeq$ is a linear order,
2. if $(r, l, n, s) \in S$ and $r > 0$ then $(r, l, n, s) \succ s$,
3. all strictly descending $\succ$ -chains are finite.

PROOF. (1) is obvious. To show (2), let (r, l, n, s) be the signature of g . Then s is the signature of the leading coefficient a_n of g . If g is a log-type expression, then either $\text{rank}(a_n) < r$ or $\text{rank}(a_n) = r$ and the log-content of a_n is less than l . If g is an exp-type expression then $\text{rank}(a_n) < r$, which completes the proof of (2).

Suppose there exists an infinite strictly descending chain of signatures

$$(r_1, l_1, n_1, s_1) \succ (r_2, l_2, n_2, s_2) \succ \dots$$

Since $r_1 \geq r_2 \geq \dots$ is an infinite descending chain of non-negative integers, there exists r and N_1 such that $r_k = r$ for all $k \geq N_1$. Then $l_{N_1} \geq l_{N_1+1} \geq \dots$ is an infinite descending chain of non-negative integers, and so there exists l and $N_2 \geq N_1$ such that $l_k = l$ for all $k \geq N_2$. Similarly, $n_{N_2} \geq n_{N_2+1} \geq \dots$ is an infinite descending chain of non-negative integers, and so there exists n and $N_3 \geq N_2$ such that $n_k = n$ for all $k \geq N_3$. Let us call (r, l, n) the stable triple of the infinite descending chain. Let X be the set of stable triples of all strictly descending infinite chains of signatures. Put $r_0 := \min\{r' : \exists l', n' (r', l', n') \in X\}$, $l_0 := \min\{l' : \exists n' (r_0, l', n') \in X\}$, $n_0 := \min\{n' : (r_0, l_0, n') \in X\}$. Since $(r_0, l_0, n_0) \in X$, there exists an infinite strictly descending chain of signatures

$$(r_0, l_0, n_0, s_{0,1}) \succ (r_0, l_0, n_0, s_{0,2}) \succ \dots$$

Then the chain $s_{0,1} \succ s_{0,2} \succ \dots$ must also be strictly descending. Let (r', l', n') be its stable triple. Then there exists a normal exp-log expression g such that $(r_0, l_0, n_0, s_{0,k})$ is the signature of g and $s_{0,k} = (r', l', n', s')$ is the signature of the leading coefficient of g . By the proof of (2), either $r' < r_0$ or $r' = r_0 \wedge l' < l_0$. Since $(r', l', n') \in X$, this contradicts the choice of (r_0, l_0, n_0) . $\square$

LEMMA 2.11. *Let $g \notin K$ be a normal exp-log expression. There exists an algorithm `NormalDerivative` which finds a normal exp-log expression $ND(g)$ and an exp-log expression h such that $\frac{ND(g)}{h}$ extends g' and $\text{sig}(ND(g)) \prec \text{sig}(g)$.*

PROOF. By Lemma 2.10 (3) we can proceed by induction on signature of g . If $\text{rank}(g) = 0$,

$$g = a_n x^n + \dots + a_1 x + a_0$$

and

$$ND(g) := g' = na_n x^{n-1} + \dots + a_1$$

is a normal expression and

$$\text{sig}(ND(g)) = (0, 0, n-1, 0) \prec (0, 0, n, 0) = \text{sig}(g)$$

Now assume that $\text{rank}(g) > 0$ and for any normal exp-log expression f with $\text{sig}(f) \prec \text{sig}(g)$ the lemma is true. If g is a log-type expression

$$g = a_n \log(p)^n + \dots + a_1 \log(p) + a_0$$

we have

$$g' = (a_n)' \log(p)^n + \frac{na_n p' + p(a_{n-1})'}{p} \log(p)^{n-1} + \dots + (a_0)'$$

By the induction hypothesis, $\frac{ND(a_n)}{h_n}$ extends $(a_n)'$ and

$$s_1 = \text{sig}(ND(a_n)) \prec s = \text{sig}(g)$$

If $ND(a_n) \neq 0$, set

$$ND(g) := ND(a_n) \log(p)^n + h_n \frac{na_n p' + p(a_{n-1})'}{p} \log(p)^{n-1} + \dots + h_n (a_0)'$$

Then $\frac{ND(g)}{h_n}$ extends g' and

$$\text{sig}(ND(g)) = (r, l, n, s_1) \prec (r, l, n, s) = \text{sig}(g)$$

If $ND(a_n) = 0$, set $k := n$ and $b_k := 0$. While $b_k = 0$ and $k > 0$ decrement k and compute

$$\frac{b_k}{h_k} := \text{NormalFraction}\left(\frac{(k+1)a_{k+1}p' + p(a_k)'}{p}\right)$$

If $k > 0$, set

$$ND(g) := b_k \log(p)^k + h_k \frac{ka_k p' + p(a_{k-1})'}{p} \log(p)^{k-1} + \dots + h_k (a_0)'$$

and put $s_2 := \text{sig}(b_k)$. Then $\frac{ND(g)}{h_k}$ extends g' and

$$\text{sig}(ND(g)) = (r, l, k, s_2) \prec (r, l, n, s) = \text{sig}(g)$$

If $k = 0$, set $ND(g) := b_0$. Then $\frac{ND(g)}{h_0}$ extends g' and either the log-content of b_0 is less than l or the rank of b_0 is less than r , and hence $\text{sig}(ND(g)) \prec \text{sig}(g)$.

Suppose now that g is an exp-type normal expression.

$$g = a_n + a_{n-1} \exp(f_{n-1}) + \dots + a_0 \exp(f_0)$$

Then g' is equal to

$$(a_n)' + (a_{n-1}(f_{n-1})' + (a_{n-1})' \exp(f_{n-1}) + \dots + (a_0(f_0)' + (a_0)' \exp(f_0))$$

By the induction hypothesis, $\frac{ND(a_n)}{h_n}$ extends $(a_n)'$ and

$$s_1 = \text{sig}(ND(a_n)) \prec s = \text{sig}(g)$$

If $ND(a_n) \neq 0$, set $ND(g)$ to be

$$ND(a_n) + h_n(a_{n-1}(f_{n-1})' + (a_{n-1})' \exp(f_{n-1}) + \dots + h_n(a_0(f_0)' + (a_0)' \exp(f_0))$$

Then $\frac{ND(g)}{h_n}$ extends g' and

$$\text{sig}(ND(g)) = (r, 0, n, s_1) \prec (r, 0, n, s) = \text{sig}(g)$$

If $ND(a_n) = 0$, set $k := n$ and $b_k := 0$. While $b_k = 0$ and $k > 0$ decrement k and compute

$$\frac{b_k}{h_k} = \text{NormalFraction}(a_k(f_k)' + (a_k)')$$

If $k > 0$, set

$$ND(g) := b_k \exp(f_k) + \dots + h_k(a_0(f_0)' + (a_0)' \exp(f_0))$$

and put $s_2 := \text{sig}(b_k)$. Then $\frac{ND(g)}{h_k}$ extends g' and

$$\text{sig}(ND(g)) = (r, 0, k, s_2) \prec (r, 0, n, s) = \text{sig}(g)$$

If $k = 0$, set $ND(g) := b_0$. Then $\frac{ND(g)}{h_0 \exp(-f_0)}$ extends g' . Since $\text{rank}(b_0) < r$, $\text{sig}(ND(g)) \prec \text{sig}(g)$. $\square$

The following algorithm constructs a semi Fourier sequence for an arbitrary exp-log expression.

ALGORITHM 2.12. (*SemiFourierSequence*)

Input: An exp-log expression f .

Output: Normal exp-log expressions $g_1, \dots, g_m$ and exp-log expressions $h_1, \dots, h_m$ such that $\frac{g_1}{h_1}, \dots, \frac{g_m}{h_m}$ is a semi Fourier sequence for f .

1. Set $\frac{g_1}{h_1} := \text{NormalFraction}(f)$, $m := 1$.

2. If $g_m = 0$, return $g_1, \dots, g_{m-1}$ and $h_1, \dots, h_{m-1}$.

3. If $g_m \in K$, return $g_1, \dots, g_m$ and $h_1, \dots, h_m$.
4. Set $\frac{g_{m+1}}{h_{m+1}} := \text{NormalDerivative}(g_m)$, increment m and go to (2).

By Lemma 2.11 and Lemma 2.10 (3), the algorithm terminates and gives a semi Fourier sequence for f .

CLAIM 2.13. *Expressions $h_1, \dots, h_m$ returned by SemiFourierSequence are products of exponentials and expressions that appear in f as factors of denominators or arguments of log.*

PROOF. *SemiFourierSequence* constructs new expressions by using *NormalFraction*, taking the derivative, multiplying by a denominator or dividing by an exponential. Hence an expression that appears as a factor of a denominator in a newly constructed expression either appeared as a factor of a denominator in a previously constructed expression, or is an exponential, or appeared in a previously constructed expression as a factor of an argument of log. To prove the claim it suffices to note that any logarithm that appears in a newly constructed expression must have appeared in a previously constructed expression. $\square$

REMARK 2.14. *Semi Fourier sequences of exp-log functions are not unique. The sequence returned by SemiFourierSequence depends on the choice algorithm used to determine leading logarithms and on the choices made in the NormalFraction algorithm.*

Any factors of $h_1, \dots, h_m$ that are known to be positive, for instance exponentials, can be removed and the sequence remains a semi Fourier sequence for f .

If g_m is a sparse polynomial

$$g_m = a_{n_k} x^{n_k} + \dots + a_{n_1} x^{n_1} + a_{n_0}$$

step (4) can be changed to

$$\begin{aligned} g_{m+1} &:= n_k a_{n_k} x^{n_k - n_1} + \dots + n_1 a_{n_1} \\ h_{m+1} &:= \frac{1}{x^{n_1 - 1}} \end{aligned}$$

The resulting sequence is shorter and is a semi Fourier sequence for the restriction of f to $D(f) \setminus \{0\}$.

3. SIGN TESTING

The root isolation algorithm requires the ability to compute the sign of $f(r)$ for any an exp-log expression f and exp-log root constant r . The key part of determining the sign of $f(r)$ is the ability to recognize when $f(r) = 0$. If $f(r) \neq 0$, its sign can be computed by decreasing the size of the isolating interval I of r until interval arithmetic evaluation of $f(I)$ yields an interval which does not contain zero.

DEFINITION 3.1. *An elementary constant is a coordinate of a point $\alpha \in \mathbb{C}^n$ such that $f_1(\alpha) = \dots = f_n(\alpha) = 0$, where f_i is either a polynomial in $(x_1, \dots, x_n)$ or a function of the form $x_j - \exp(x_k)$, and the Jacobian $Jac(f_1, \dots, f_n)$ is nonzero at α .*

The procedure given in [6] recognizes when an elementary constant is zero. The proof that the procedure terminates relies on Schanuel's conjecture [3]. If K is contained in the field of real elementary constants, Lemma 3.2 below shows that $f(r)$ is an elementary constant and hence the procedure

given in [6] can be used to decide whether $f(r) = 0$. When $f(r)$ is an exp-log expression with coefficients in $\mathbb{Q}$, more efficient methods may be used [4, 7].

LEMMA 3.2. *If f is an exp-log expression with coefficients in the field of real elementary constants and $r = \text{Root}(g, a, b)$ is an exp-log root constant, then $f(r)$ is an elementary constant.*

PROOF. For some $k \geq 2$, let $x = (x_1, \dots, x_k)$, let $\alpha \in \mathbb{R}^k$ and

$$F : \mathbb{R}^k \ni x \rightarrow (f_1(x), \dots, f_k(x)) \in \mathbb{R}^k$$

be such that $f_1(x), \dots, f_k(x)$ are multivariate exp-log expressions, $F(\alpha) = 0$ and $Jac(F) \neq 0$. Multivariate exp-log expressions in variables $x_1, \dots, x_k$ with coefficients in K are defined as follows:

1. elements of K and variables $x_1, \dots, x_k$ are multivariate exp-log expressions,
2. if f and g are multivariate exp-log expressions, so are $f + g$, $f \cdot g$ and $\frac{f}{g}$,
3. if f is a multivariate exp-log expression, so are $\exp(f)$ and $\log(f)$.

Let $\bar{x} = (x_1, \dots, x_k, x_{k+1})$. Define

$$\Phi_h(F) : \mathbb{R}^{k+1} \ni \bar{x} \rightarrow (f_{1,h}(\bar{x}), \dots, f_{k+1,h}(\bar{x})) \in \mathbb{R}^{k+1}$$

and

$$\Psi(F) : \mathbb{R}^{k+1} \ni \bar{x} \rightarrow (f_{1,\log}(\bar{x}), \dots, f_{k+1,\log}(\bar{x})) \in \mathbb{R}^{k+1}$$

as follows. For $1 \leq i \leq k$, let $f_{i,h}(\bar{x})$ be $f_i(x)$ with each instance of the subexpression $h(x)$ replaced by x_{k+1} , so that

$$f_i(x) = f_{i,h}(x_1, \dots, x_k, h(x))$$

let $f_{i,\log}(\bar{x})$ be $f_i(x)$ with each instance of $\log(x_k)$ replaced by x_{k+1} , so that

$$f_i(x) = f_{i,\log}(x_1, \dots, x_k, \log(x_k))$$

let $f_{k+1,h}(\bar{x}) := x_{k+1} - h(x)$ and let

$$f_{k+1,\log}(\bar{x}) := x_k - \exp(x_{k+1})$$

Put

$$\begin{aligned} \beta &:= (\alpha_1, \dots, \alpha_k, h(\alpha)) \\ \gamma &:= (\alpha_1, \dots, \alpha_k, \log(\alpha_k)) \end{aligned}$$

Then $\Phi_h(F)(\beta) = 0$, $\Psi(F)(\gamma) = 0$,

$$Jac(\Phi_h(F))_{x_{k+1}=h(x)} = Jac(F)$$

and

$$Jac(\Psi(F))_{x_{k+1}=\log(x_k)} = -x_k Jac(F)$$

Hence $Jac(\Phi_h(F))_{\bar{x}=\beta} \neq 0$ and, if F contains $\log(x_k)$ as a subexpression, $\alpha_k \neq 0$ and so $Jac(\Psi(F))_{\bar{x}=\gamma} \neq 0$.

Put $x := (x_1, x_2)$, $f_1(x) := x_1 - f(x_2)$, $f_2(x) := g(x_2)$, $F = (f_1, f_2)$ and $\alpha := (f(r), r)$. Then $F(\alpha) = 0$ and

$$Jac(F)_{x=\alpha} = g'(x_2)_{x=\alpha} = g'(r) \neq 0$$

Starting with F , by successive application of operations Φ_h and Ψ to eliminate subexpressions of the form $\exp(h)$ and $\log(h)$, we can obtain

$$\bar{F} : \mathbb{R}^n \ni \bar{x} \rightarrow (\bar{f}_1(\bar{x}), \dots, \bar{f}_n(\bar{x})) \in \mathbb{R}^n$$

and $\bar{\alpha} = (f(r), r, \alpha_3, \dots, \alpha_n) \in \mathbb{R}^n$, such that $\bar{F}(\bar{\alpha}) = 0$, $\bar{f}_i$ is either a polynomial in $\bar{x} = (x_1, \dots, x_n)$ or a function of the form $x_j - \exp(x_k)$ and the Jacobian $Jac(\bar{F})$ is nonzero at $\bar{\alpha}$. This proves that $f(r)$ is an elementary constant. $\square$

4. ROOT ISOLATION

This section presents the main root isolation algorithm for exp-log functions. Let us first describe subalgorithms used by the main algorithm. The first algorithm finds signs near endpoints of an interval for a weak Fourier sequence of exp-log functions.

ALGORITHM 4.1. (*LimitSigns*)

Input: $a < b$, a weak Fourier sequence $f_1, \dots, f_k : (a, b) \rightarrow \mathbb{R}$ of exp-log functions, a sequence $l_1, \dots, l_{k-1}$ of signs and a direction $dir \in \{\text{left}, \text{right}\}$. a and b are elements of K , exp-log root constants, $-\infty$, or ∞ . If $dir = \text{left}$, $l_j = \text{sign}(\lim_{x \rightarrow a^+} f_j(x))$, else $l_j = \text{sign}(\lim_{x \rightarrow b^-} f_j(x))$, for $1 \leq j \leq k-1$.

Output: Sequence $s_1, \dots, s_k$ of nonzero signs such that, if $dir = \text{left}$, $s_j = \text{sign}(f_j(a^+))$, else $s_j = \text{sign}(f_j(b^-))$, for $1 \leq j \leq k$.

1. Since f_k is nonzero in (a, b) , $\text{sign}(f_k(x))$ is constant in (a, b) . Pick a rational number $a < x_0 < b$ and set $s_k := \text{sign}(f_k(x_0))$.
2. For $j := k-1$ to 1 do
 - (a) If $l_j \neq 0$, set $s_j = l_j$.
 - (b) If $l_j = 0$ and $dir = \text{left}$, if $s_{j+1} > 0$, set $s_j := 1$, else set $s_j := -1$.
 - (c) If $l_j = 0$ and $dir = \text{right}$, if $s_{j+1} > 0$, set $s_j := -1$, else set $s_j := 1$.
3. Return $s_1, \dots, s_k$.

The next algorithm allows to reduce the problem of finding roots of an exp-log function f in an arbitrary open interval $I \subseteq D(f)$ to finding roots in an open interval with rational endpoints that belong to $D(f)$.

ALGORITHM 4.2. (*RationalEndpoint*)

Input: $a < b$, a weak Fourier sequence $f_1, \dots, f_k : (a, b) \rightarrow \mathbb{R}$ of exp-log functions, a sequence $s_1, \dots, s_k$ of nonzero signs and a direction $dir \in \{\text{left}, \text{right}\}$. a and b are elements of K , exp-log root constants, $-\infty$, or ∞ . For $1 \leq j \leq k$, if $dir = \text{left}$, $s_j = \text{sign}(f_j(a^+))$, else $s_j = \text{sign}(f_j(b^-))$.

Output: A rational number $a < c < b$ and a sequence $sc_1, \dots, sc_k$ of nonzero signs, such that $\text{sign}(f_j(c)) = sc_j$, for $1 \leq j \leq k$, and there are no roots of f_1 in U , where if $dir = \text{left}$, $U = (a, c]$, else $U = [c, b)$.

1. Pick a rational number $a < c < b$ and compute $sc_j := \text{sign}(f_j(c))$. Let n be the number of sign changes in the sequence $s_1, \dots, s_k$.
2. While $\text{sgc}_{1,k}(c) \neq n$ or $sc_j = 0$ for some $1 \leq j \leq k$:
 - (a) If $dir = \text{left}$, if $a = -\infty$, set $d := \min(-1, 2c)$, else pick a rational number d such that $d - a < \frac{c-a}{2}$.
 - (b) If $dir = \text{right}$, if $b = \infty$, set $d := \max(1, 2d)$, else pick a rational number d such that $b - d < \frac{b-c}{2}$.

(c) Set $c := d$ and compute $sc_j := \text{sign}(f_j(c))$.

3. Return c and $sc_1, \dots, sc_k$. By part (5) of Corollary 2.6, c has the required properties.

The following algorithm finds an exp-log root constant representing an isolated root of an exp-log function.

ALGORITHM 4.3. (*FindIsolatedRoot*)

Input: An exp-log function f and a bounded interval (a, b) such that

$$\text{sign}(f(a)f(b)) = -1$$

and f has exactly one simple root in (a, b) .

Output: The root of f in (a, b) represented as an exp-log root constant or an element of K .

1. Set $c := a$ and $d := b$.
2. If the result of interval arithmetic evaluation of f' at (c, d) does not contain 0, return $\text{Root}(f, c, d)$.
3. Set $e := \frac{c+d}{2}$. If $f(e) = 0$, return e .
4. If $\text{sign}(f(c)f(e)) = -1$, set $d := e$, else set $c := e$. Goto (2).

Instead of interval bisection it is more efficient in practice to obtain a smaller isolating interval using a numeric root finding method with a faster convergence. Our implementation uses Brent's method.

The next subalgorithm computes the zero set of an exp-log function with a given semi Fourier sequence in an open interval contained in the domain of the function.

ALGORITHM 4.4. (*RootsInInterval*)

Input: An exp-log function f , a semi Fourier sequence

$$\frac{g_1}{h_1}, \dots, \frac{g_m}{h_m}$$

for f and $a < b$ such that $(a, b) \subseteq D(f)$. a and b are elements of K , exp-log root constants, $-\infty$, or ∞ .

Output: A list $r_1, \dots, r_t$ of exp-log root constants or elements of K representing all roots of f in (a, b) or IdZero if f is identically zero in (a, b) .

1. Put $I := (a, b)$ and let $f_1, \dots, f_m$ be as in Corollary 2.6.
2. Find a rational number $a < e < b$. Let k be the largest $1 \leq i \leq m$ such that $f_i(e) \neq 0$ or 0 if such i does not exist. By part (1) of Corollary 2.6, $f_{k+1}, \dots, f_m$ are identically zero in I .
3. If $k = 0$, f is identically zero in I . Return IdZero .
4. If $k = 1$, f is nonzero and constant in I . Return $\emptyset$.
5. Compute

$$\begin{aligned} la_j &:= \text{sign}(\lim_{x \rightarrow a^+} f_j(x)) \\ lb_j &:= \text{sign}(\lim_{x \rightarrow b^-} f_j(x)) \end{aligned}$$

for $1 \leq j \leq k-1$, using the algorithm given in [1] to compute limits and the procedure described in the previous section to compute signs.

6. Use *LimitSigns* to compute $sa_1, \dots, sa_k$ and $sb_1, \dots, sb_k$ such that $sa_j = \text{sign}(f_j(a^+))$, else $sb_j = \text{sign}(f_j(b^-))$, for $1 \leq j \leq k$. Let $sgca$ and $sgcb$ be the numbers of sign changes in these sequences.
7. If $sgca = sgcb$, f has no roots in I , by part (5) of Corollary 2.6. Return $\emptyset$.
8. Use *RationalEndpoint* to find $c, sc_1, \dots, sc_k, d$ and $sd_1, \dots, sd_k$ such that
$$\text{sign}(f_j(c)) = sc_j \wedge \text{sign}(f_j(d)) = sd_j$$
for $1 \leq j \leq k$, and there are no roots of f in $(a, c] \cup [d, b)$.
9. Set $roots = \emptyset$ and $stack := \{(c, d, k)\}$. At any point of the computation $stack$ contains triples (c, d, k) such that $c < d$ are rational numbers, f_k has a constant nonzero sign in (c, d) and $sgc_{1,k}(c^+) - sgc_{1,k}(d^-) > 0$.
10. While $stack \neq \emptyset$ do:

(a) Set

$$(c, d, k) := \text{First}(stack) \\ stack := \text{Rest}(stack)$$

(b) If $sgc_{1,k}(c^+) - sgc_{1,k}(d^-) = 1$, f has exactly one simple root in (c, d) . Let r be the output of *FindIsolatedRoot* applied to f and (c, d) . Append r to $roots$ and continue the loop.

(c) By the construction of $stack$,

$$sgc_{1,k}(c^+) - sgc_{1,k}(d^-) > 1$$

Let l be the smallest element of

$$\{i : 2 \leq i \leq k \wedge sgc_{i,k}(c^+) - sgc_{i,k}(d^-) = 1\}$$

(d) By Theorem 2.4, f_l has exactly one simple root in (c, d) . Let r be the output of *FindIsolatedRoot* applied to f_l and (c, d) .

(e) If $f(r) = 0$, append r to $roots$.

(f) Compute $sr_j := \text{sign}(f_j(r))$, for $1 \leq j \leq l - 1$. Since r is the only root of f_l in (c, d) , f_l has constant signs in (c, r) and in (r, d) .

(g) Use *LimitSigns* to compute $sr_j^- := \text{sign}(f_j(r^-))$ and $sr_j^+ := \text{sign}(f_j(r^+))$, for $1 \leq j \leq l$.

(h) Use *RationalEndpoint* to find $u < r < v$, $su_j := \text{sign}(f_j(u))$, $sv_j := \text{sign}(f_j(v))$, for $1 \leq j \leq l$, such that there are no roots of f in $[u, r) \cup (r, v]$.

(i) If $sgc_{1,l}(c^+) - sgc_{1,l}(u^-) > 0$, add (c, u, l) to $stack$.

(j) If $sgc_{1,l}(v^+) - sgc_{1,l}(d^-) > 0$, add (v, d, l) to $stack$.

11. Return $roots$.

The algorithm given in [1] can be used to compute limits at numbers represented by exp-log root constants, since we can compute signs of exp-log functions at such numbers. By Proposition 2.2, the loop in step (10) is executed at most $\frac{(m-1)m}{2}$ times. Steps (10e) through (10h) can often be significantly simplified when $f(r) \neq 0$. If $r = \text{Root}(f_l, \alpha, \beta)$ and the result of interval arithmetic evaluation of f at $[\alpha, \beta]$ does not contain zero, then $f(r) \neq 0$ and one can use $u = \alpha$

and $v = \beta$. Otherwise, if $sgc_{1,l+1}(\alpha) = sgc_{1,l+1}(\beta)$, then also $f(r) \neq 0$ and one can use $u = \alpha$ and $v = \beta$. Finally, if the zero testing algorithm has shown that $f(r) \neq 0$ by finding an interval I with rational endpoints, such that $r \in I$ and the result of interval arithmetic evaluation of f at I does not contain zero, then the endpoints of I can be used as u and v .

Let us now describe the main root isolation algorithm for exp-log functions.

ALGORITHM 4.5. (*ExpLogRootIsolation*)

Input: An exp-log expression f .

Output: Lists $(a_1, b_1), \dots, (a_l, b_l)$, $(c_1, d_1), \dots, (c_m, d_m)$ and $r_1, \dots, r_n$ such that a_i, b_i, c_j, d_j and r_k are elements of K , exp-log root constants, $-\infty$, or ∞ ,

$$D(f) = (a_1, b_1) \cup \dots \cup (a_l, b_l)$$

and

$$f^{-1}(0) = (c_1, d_1) \cup \dots \cup (c_m, d_m) \cup \{r_1, \dots, r_n\}$$

1. Use *SemiFourierSequence* to compute normal exp-log expressions $g_1, \dots, g_t$ and exp-log expressions $h_1, \dots, h_t$ such that $\frac{g_1}{h_1}, \dots, \frac{g_t}{h_t}$ is a semi Fourier sequence for f .

2. Let $H := \{h_1, \dots, h_t\}$, let L be the set of all expressions that appear in f as arguments of log, and let Den be the set of all expressions that appear in f as a denominator. Use *ExpLogRootIsolation* recursively to compute the domains and zero sets of all factors of elements of H , L and Den . Then $D(f)$ is the set where all elements of H and Den are defined and nonzero and all elements of L are defined and positive. Let $(a_1, b_1), \dots, (a_l, b_l)$ be the set of components of $D(f)$.

3. Set $rootlist = \emptyset$ and $idzero = \emptyset$.

4. For $1 \leq i \leq l$ do:

(a) Use *RootsInInterval* to compute the zero set of f in (a_i, b_i) .

(b) If the output is *IdZero*, add (a_i, b_i) to $idzero$.

(c) If the output is a list of roots, add the roots to $rootlist$.

5. Return $(a_1, b_1), \dots, (a_l, b_l)$, $idzero$ and $rootlist$.

Note that all inputs to recursive *ExpLogRootIsolation* calls in step (2) appear in f as proper subexpressions, hence the recursion terminates.

EXAMPLE 4.6. Find the domain and the zero set of $f(x) = \exp(x)/(1+x) - x^2 \log(x)$.

SemiFourierSequence gives

$$\begin{aligned}\frac{g_1(x)}{h_1(x)} &= \frac{(-x^3 - x^2) \log(x) + \exp(x)}{1+x} \\ \frac{g_2(x)}{h_2(x)} &= \frac{(-3x^2 - 2x) \log(x) - x^2 - x + \exp(x)}{1} \\ \frac{g_3(x)}{h_3(x)} &= \frac{(-6x - 2) \log(x) - 5x - 3 + \exp(x)}{1} \\ \frac{g_4(x)}{h_4(x)} &= \frac{-6 \log(x) - 11 - 2/x + \exp(x)}{1} \\ \frac{g_5(x)}{h_5(x)} &= \frac{-6x + 2 + x^2 \exp(x)}{x^2} \\ \frac{g_6(x)}{h_6(x)} &= \frac{-6 + (x^2 + 2x) \exp(x)}{1} \\ \frac{g_7(x)}{h_7(x)} &= \frac{x^2 + 4x + 2}{1} \\ \frac{g_8(x)}{h_8(x)} &= \frac{2x + 4}{1} \\ \frac{g_9(x)}{h_9(x)} &= \frac{2}{1}\end{aligned}$$

Nonconstant factors of elements of H , L and Den are $x + 1$ and x . Their roots are $\{-1, 0\}$. $D(f)$ consists of those among the intervals $(-\infty, -1)$, $(-1, 0)$ and $(0, \infty)$ for which all elements of $L = \{x\}$ are positive. Hence $D(f) = (0, \infty)$.

RootsInInterval is called to find roots of f in $(0, \infty)$. For $1 \leq i \leq 9$, h_i is positive in $(0, \infty)$, hence $f_i = g_i$. Since $f_9 = 2$ is a nonzero constant, $k = 9$. Sequences $\text{sign}(f_j(0^+))$ and $\text{sign}(f_j(\infty^-))$, for $1 \leq i \leq 9$, are $(1, 1, 1, -1, 1, -1, 1, 1, 1)$ and $(1, 1, 1, 1, 1, 1, 1, 1, 1)$. RationalEndpoint finds $c = \frac{1}{4}$ and $d = 7$, with sign sequences the same as at 0 and ∞ . The initial value of stack is $\{(\frac{1}{4}, 7, 9)\}$.

In the first iteration of the loop in step (10), $(c, d, k) = (\frac{1}{4}, 7, 9)$. $\text{sgc}_{1,k}(c^+) - \text{sgc}_{1,k}(d^-) = 4$ and $l = 6$. FindIsolatedRoot applied to f_6 and (c, d) gives

$$r_1 = \text{Root}(-6 + (x^2 + 2x) \exp(x), \alpha_1, \beta_1)$$

with the rational numbers $\alpha_1 = \frac{8603293572629645}{9858382060577563}$ and $\beta_1 = \frac{15177907175903278}{17392130880674017}$ bounding r sharing first 20 decimal digits

$$0.87268818755089033264 \dots$$

The sign sequences of $f_1, \dots, f_7$ evaluated at α_1 and at β_1 are $(1, 1, -1, -1, -1, -1, 1)$ and $(1, 1, -1, -1, -1, 1, 1)$. Both sequences contain two sign changes, hence there are no roots of f in (α_1, β_1) , in particular $f(r) \neq 0$. The numbers of sign changes in the sequence $f_1, \dots, f_6$ evaluated at $\frac{1}{4}$, α_1 , β_1 and 7 are 3, 1, 2 and 0, hence triples $(\frac{1}{4}, \alpha_1, 6)$ and $(\beta_1, 7, 6)$ are added to stack.

In the second iteration, $(c, d, k) = (\frac{1}{4}, \alpha_1, 6)$. $\text{sgc}_{1,k}(c^+) - \text{sgc}_{1,k}(d^-) = 2$ and $l = 5$. FindIsolatedRoot applied to f_5 and (c, d) gives

$$r = \text{Root}(-6x + 2 + x^2 \exp(x), \alpha_2, \beta_2)$$

with the rational numbers $\alpha_2 = \frac{4081802980088075}{11170713760239559}$ and $\beta_2 = \frac{12947598123789739}{35433829910219993}$ bounding r sharing first 20 decimal digits

$$0.36540216388111439918 \dots$$

Interval arithmetic evaluation of f at (α_2, β_2) gives an interval not containing zero, hence there are no roots of f in (α_2, β_2) , in particular $f(r) \neq 0$. The sign sequences of

$f_1, \dots, f_5$ evaluated at $\frac{1}{4}$, α_2 , β_2 and α_1 are $(1, 1, 1, -1, 1)$, $(1, 1, 1, -1, 1)$, $(1, 1, 1, -1, -1)$ and $(1, 1, -1, -1, -1)$, hence there are no roots of f in $(\frac{1}{4}, \alpha_1)$.

In the next iteration, $(c, d, k) = (\beta_1, 7, 6)$. $\text{sgc}_{1,k}(c^+) - \text{sgc}_{1,k}(d^-) = 2$ and $l = 3$. FindIsolatedRoot applied to f_3 and (c, d) gives

$$r = \text{Root}((-6x - 2) \log(x) - 5x - 3 + \exp(x), \alpha_3, \beta_3)$$

with the rational numbers $\alpha_3 = \frac{45724519493580038}{11106478695297865}$ and $\beta_3 = \frac{20152338824691207}{489499975850651}$ bounding r sharing first 20 decimal digits

$$4.1169231714223128647 \dots$$

The sign sequences of f_1, f_2, f_3, f_4 evaluated at α_3 and at β_3 are $(-1, -1, -1, 1)$ and $(-1, -1, 1, 1)$. Both sequences contain one sign change, hence there are no roots of f in (α_3, β_3) , in particular $f(r) \neq 0$. The sign sequences of f_1, f_2, f_3 evaluated at β_1 , α_3 , β_3 and 7 are

$$(1, 1, -1), (-1, -1, -1), (-1, -1, 1), (1, 1, 1)$$

hence there is one simple root of f in each of (β_1, α_3) and $(\beta_3, 7)$. FindIsolatedRoot finds the roots to be

$$\begin{aligned}r_1 &= \text{Root}((-x^3 - x^2) \log(x) + \exp(x), \alpha_4, \beta_4) \\ r_2 &= \text{Root}((-x^3 - x^2) \log(x) + \exp(x), \alpha_5, \beta_5)\end{aligned}$$

with $\alpha_4 = \frac{34023785792187461}{17946115613634908}$, $\beta_4 = \frac{20854319682786641}{10999776284652230}$, $\alpha_5 = \frac{100543306819}{16076243394}$, $\beta_5 = \frac{56998088969}{9113636504}$ and 20 digit approximations of r_1 and r_2 equal to

$$1.8958858019580143857 \dots$$

$$6.2541543042651945557 \dots$$

5. IMPLEMENTATION AND EXPERIMENTAL RESULTS

We have implemented *ExpLogRootIsolation* as a part of the *Mathematica* system. The implementation has been done partly in the C source code of *Mathematica* and partly in the *Mathematica* programming language. Our implementation does not use the zero testing algorithm defined in [6]. Instead we use a zero testing heuristic which combines interval arithmetic evaluation up to a pre-selected interval size with symbolic simplification for expressions for which the interval arithmetic evaluation did not yield a nonzero sign. If the heuristic fails to determine the sign of a constant, the constant is recorded and the algorithm proceeds as if the constant was zero. The constants for which the heuristic failed are returned with the answer. For the examples given in this section the heuristic was always able to successfully determine the signs.

The experiments have been run on a laptop computer with a 1.8 GHz Pentium M processor and 1.74 GB of RAM assigned to the Linux virtual machine. The columns in tables give the total time in seconds (TotT), the time in seconds for computing the semi Fourier sequence (SFT), the total time in seconds for all recursive calls to *ExpLogRootIsolation* (RecT), the number of distinct real roots (Rts), the number of connected components of the domain (Cmp), the number of functions in the semi Fourier sequence (SFL), the total number of iterations in point (10) in all calls to *RootsInInterval* from the main call to *ExpLogRootIsolation* (Iter).

Table 1: Example 5.1.

No.	TotT	SFT	RecT	Rts	Cmp	SFL	Iter
1	0.292	0.084	< 0.001	1	2	14	8
2	16.9	0.720	< 0.001	2	3	109	74
3	0.200	0.004	< 0.001	2	1	9	4
4	0.748	0.120	< 0.001	1	1	23	9
5	0.348	0.020	0.008	1	2	5	1
6	2.94	0.168	0.088	2	1	38	19
7	2401	0.976	0.036	3	2	246	245
8	0.448	0.016	< 0.001	2	2	13	8

EXAMPLE 5.1. The following examples have been derived from examples in [1] by adding constants, chosen so that each function has at least one real root.

1. $e^x(e^{1/x-e^{-x}} - e^{1/x}) + 5$
2. $e^{\frac{e^x-e^{-x}}{1-1/x}} - e^{e^x} - 10^8$
3. $\frac{e^{e^x+e^{-x}}}{e^{e^x}} - 10^{67}$
4. $\frac{e^{e^x}}{e^{e^x-e^{-x}}} - 10^5$
5. $(3^x + 5^x)^{1/x} - 10$
6. $\frac{x}{\log(x \log(x) \log(2)/\log(x))} - 10$
7. $\frac{\exp(4xe^{-x}/(e^{-x}+e^{\frac{-2x^2}{x+1}}))-e^x}{e^{4x}} - \frac{1}{10}$
8. $\frac{\exp(\frac{xe^{-x}}{e^{-x}+e^{-2x^2/(x+1)}})}{e^x} - \frac{7}{10}$

EXAMPLE 5.2. The following examples are based on named exp-log functions and constants found in [9].

1. Smarandache constant
 $127^x - 113^x - 1$
2. Half-maximum points of the Gaussian function
 $\frac{e^{\frac{-x^2}{2}} - \frac{1}{2}}{\sqrt{2\pi}}$
3. Fixed point of inverse hyperbolic cotangent
 $\operatorname{arccoth}(x) - x$
4. Inflection points of Einstein's E_1 function
 $\frac{\partial^2}{\partial x^2} \frac{x^2 e^x}{(e^x - 1)^2}$
5. Intersection points of Einstein's E_4 function with a hyperbola
 $\frac{x}{e^x - 1} - \log(1 - e^{-x}) - \frac{1}{x}$
6. Laplace limit constant
 $\frac{xe^{\sqrt{x^2+1}}}{1 + \sqrt{x^2+1}} - 1$

Table 2: Example 5.2.

No.	TotT	SFT	RecT	Rts	Cmp	SFL	Iter
1	0.028	< 0.001	< 0.001	1	1	3	1
2	0.032	0.004	< 0.001	2	1	2	2
3	0.032	< 0.001	< 0.001	2	2	2	2
4	0.072	0.008	0.008	2	2	9	2
5	0.212	0.016	< 0.001	2	1	10	4
6	0.240	0.024	0.028	1	1	16	2
7	0.052	0.004	0.012	1	2	4	1
8	0.168	0.012	< 0.001	2	2	9	5

7. Maximum of Planck's radiation function

$$\frac{\partial}{\partial x} \frac{15}{\pi^4 x^5 (e^{1/x} - 1)}$$

8. Inflection points of Planck's radiation function

$$\frac{\partial^2}{\partial x^2} \frac{15}{\pi^4 x^5 (e^{1/x} - 1)}$$

EXAMPLE 5.3. The last series contains examples constructed to have specific properties.

1. High degree sparse polynomial

$$x^{1000000} - 2x^{777777} + 3x^{123456} - 4x^{54321} + 5x^{6789} - 6x^{555} + 7x^{100} - 8x^{21} + 9x - 10$$

2. Algebraic function involving a high degree radical

$$2x^{\frac{123451}{67890}} - x^2 + 4\sqrt{x} - 4x - \frac{9}{8}$$

3. Function involving non-rational real powers

$$x^\pi - x^{x^{\sqrt{2}}} - \sqrt{3}x + \sqrt[3]{2}$$

4. Function with a quadruple root at 1 and two simple roots

$$e^{x-1} - \log(e^{(x-1)^2} + x + 3) - \frac{51}{250}(x-1)^3 - \frac{8}{25}(x-1)^2 - \frac{4}{5}(x-1) + \log(5) - 1$$

5. Function with two roots that share the first 388 decimal digits

$$(e^{e^x} - x^2 - 1)^2 - 10^{-777}$$

6. Functions with many real roots. T_n and U_n are degree n Chebyshev polynomials of the first and the second kind.

(a)

$$e^{T_7(x)} - \log(U_7(x) + 10)$$

(b)

$$e^{T_{10}(x)} - \log(U_{10}(x) + 10)$$

(c)

$$e^{T_{10}(x)} - 9\log(U_{10}(x) + 10) - x^2 + \frac{389}{20}$$

Table 3: Example 5.3.

No.	TotT	SFT	RecT	Rts	Cmp	SFL	Iter
1	6.72	< 0.001	< 0.001	4	1	10	19
2	0.18	0.004	0.004	4	1	5	8
3	0.644	0.144	< 0.001	3	1	12	9
4	1.04	0.028	0.032	3	1	31	29
5	5.56	0.068	< 0.001	2	1	35	19
6a	3.28	0.036	0.008	8	1	61	56
6b	5.81	0.060	0.008	10	1	66	90
6c	28.2	0.076	0.004	16	1	77	112

6. ACKNOWLEDGMENTS

The author would like to thank the reviewers to their useful comments and suggestions.

7. REFERENCES

- [1] D. Gruntz. *On computing limits in a symbolic manipulation system*. PhD thesis, ETH, 1996.
- [2] A. G. Hovanskii. On a class of systems of transcendental equations. *Soviet Math. Dokl.*, 22:762–765, 1980.
- [3] S. Lang. *Introduction to transcendental numbers*. Addison-Wesley, 1966.
- [4] C. Li, S. Pion, and C. K. Yap. Recent progress in exact geometric computation. *J. of Logic and Algebraic Programming*, 64:85–111, 2005.
- [5] D. Richardson. A solution of the identity problem for integral exponential functions. *Z. Math. Logik Grundlagen Math.*, 15:333–340, 1969.
- [6] D. Richardson. How to recognize zero. *J. Symb. Comput.*, 24:627–645, 1997.
- [7] D. Richardson. Zero tests for constants in simple scientific computation. *Mathematics in Computer Science*, 1:21–37, 2007.
- [8] L. van den Dries, A. Macintyre, and D. Marker. The elementary theory of restricted analytic fields with exponentiation. *Ann. Math.*, 140:183–205, 1994.
- [9] E. Weisstein. Mathworld. <http://mathworld.wolfram.com>.
- [10] A. J. Wilkie. Model completeness results for expansions of the ordered field of real numbers by restricted pfaffian functions and the exponential function. *J. Amer. Math. Soc.*, 9:1051–1094, 1996.

Gröbner Bases for Public Key Cryptography*

Massimo Caboara, Fabrizio Caruso, Carlo Traverso
Dipartimento di Matematica, Università di Pisa
{caboara,caruso,traverso}@dm.unipi.it

ABSTRACT

Up to now, any attempt to use Gröbner bases in the design of public key cryptosystems has failed, as anticipated by a classical paper of B. Barkee *et al.*; we show why, and show that the only residual hope is to use binomial ideals, i.e. lattices.

We propose two lattice-based cryptosystems that will show the usefulness of multivariate polynomial algebra and Gröbner bases in the construction of public key cryptosystems. The first one tries to revive two cryptosystems Polly Cracker and GGH, that have been considered broken, through a hybrid; the second one improves a cryptosystem (NTRU) that only has heuristic and challenged evidence of security, providing evidence that the extension cannot be broken with some of the standard lattice tools that can be used to break some reduced form of NTRU.

Because of the bounds on length, we only sketch the construction of these two cryptosystems, and leave many details of the construction of private and public keys, of the proofs and of the security considerations to forthcoming technical papers.

Categories and Subject Descriptors

I.1.2 [Algorithms]: Algebraic Algorithms; E.3 [Data Encryption]: Public key cryptosystems

General Terms

Algorithms

1. INTRODUCTION: BARKEE WAS RIGHT.

In 1994 an article appeared in the Journal of Symbolic Computation [2], with a remarkable title, and an even more remarkable set of authors:

*Supported by MURST under the PRIN project “Agebra commutativa, computazionale e combinatorica”. The research has originated from the activities of the “Special semester on Gröbner bases”, held in Linz at RICAM in 2006.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC’08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Boo Barkee, Deh Cac Can, Julia Ecks, Theo Moriarty, R. F. Ree: “*Why you cannot even hope to use Gröbner Bases in Public Key Cryptography: an open letter to a scientist who failed and a challenge to those who have not yet failed*” (partially supported by Spectre)

The title of the paper is a bit misleading; it does not discuss the undoubtable possibilities of Gröbner bases in cryptanalysis, but only the use of Gröbner bases in the construction of a public key cryptosystem. More precisely, it just shows how any attempt to exploit in a public key cryptosystem the worst-case complexity of Gröbner bases is deemed to fail, since computing a Gröbner basis and finding a normal form are quite different problems; decyphering cannot be more complex than encyphering, in a dense complexity model, hence the worst case complexity cannot be used to prevent attacks to the cryptogram based on linear algebra and partial Gröbner basis computation.

The cryptosystem considered (or rather a cryptosystem scheme) is the following:

1. The private key consists in a (multivariate) polynomial ring $R = k[X]$, a term-ordering, and an ideal I given through a Gröbner basis G with respect to the term-ordering, and a set M of power products of R that are in normal form mod I .
2. The public key consists in R , M and a finite subset $F \subseteq I$. The term-ordering is not public.
3. Messages are linear combinations with coefficients in k of elements of M .
4. A cryptogram c is obtained by adding to a message m an element of the ideal generated by F , i.e.
$$c = m + \sum \phi_i f_i, \phi_i \in R, f_i \in F.$$
5. Decyphering is done by computing the normal form
$$c \Rightarrow_G m.$$

Cryptosystems of this type should be called Barkee cryptosystems, but are rather known as Polly Cracker [6], that originally identified a special subclass independently defined.

These are the concluding remarks of Barkee:

“The high complexity of Gröbner bases is in fact strictly related with the existence of polynomials in an ideal whose minimal degree representation in terms of a given basis is doubly exponential in the degree of the basis elements. Since such polynomials cannot be used as encoded messages, a cryptographic scheme applying the complexity of Gröbner basis to an ideal membership problem is bound to fail.

“Is our reader able to find a scheme which overcomes this difficulty?”

*“In particular our reader could think (perhaps with some reason) that a **sparse** scheme could work. We believe (perhaps without reason) that sparsity will make the scheme easier to crack. We would be glad to test our belief on specific sparse schemes.”*

It turned out that many other scientists have failed in the subsequent years. A recent survey of Franziska Löw ben Bezalel, Miss Mary Grace Marple, Theo Moriarty, Ludovic Poirot and Cabdulqadir Taher ElGarweyne [13] has shown that the extreme sparsity is the common weakness of all the Polly Cracker cryptosystems. We detail here the argument, that is only sketched in [13], and generalizes methods of [11]. The argument is also reminiscent of the “symbolic reduction” phase of the F4 algorithm [5].

Remark that in the computation of the cryptogram $c = m + \sum \phi_i f_i$, representing polynomials as sum of monomials and applying the distributive law, a finite set S (of moderate size) of monomials appears. If we can bound this *support of the computation*, we can reconstruct for each ϕ_i a set of monomials containing their support. The ϕ_i can then be reconstructed by linear algebra. Our aim is hence to find a small set $S' \supseteq S$.

We assume first that every polynomial of F , as well as any other polynomial “easily” obtained from F has at least three monomials.

Because of the sparsity, i.e. the extremely low probability that a monomial appears in the support of a polynomial, whenever we add a monomial multiple $\mu = x^\alpha f_i$ of f_i to an already constructed polynomial f , it is extremely unlikely that more than one monomial of f and one of μ cancel; the exceptions should be part of the design, and in this case the design itself has to be public, hence it is likely to introduce new types of attacks.

As a result, the new polynomial contains more monomials than f . This restricts the number of monomials in S (and of steps in the sum $\sum \phi_i f_i$).

We hence build S' starting from the support of c and enlarge it until we can reconstruct the ϕ_i .

In the cryptogram c , and hence in S' , there will be pairs of monomials that also appear in some $x^\alpha f_i$; we add to S' the support of $x^\alpha f_i$, and we continue until we succeed in decyphering via linear algebra, or until the procedure fails to enlarge S' (or the resources are exhausted).

This reasoning breaks if F contains monomials or binomials. In this case, a reduction may replace one monomial with another, or even with nothing; encyphering, one can perform long chains of reductions without exponential growth. An eavesdropper can only assume that every operation has one element in common with the others, but this does not help: every reconstruction has multiple possibilities, and the possibility of long simplification chains entail an exponential reconstruction. But another idea is possible.

If F contains monomials or binomials, or if monomials or binomials can be easily obtained from F , and moreover the Gröbner basis of the ideal J generated by these “easy” monomials and binomials can be computed, instead of using $k[X]$ and its monomials, we can use $k[X]/J$ and its standard monomials, and see everything inside it. The attack is hence easier.

The only residual possibility is therefore that F itself is composed of monomials and binomials, i.e. it is a 2-nomial ideal (a 2-nomial is $aX^\alpha + bX^\beta$), and the Gröbner basis is difficult to compute.

Since moreover the constants a and b can be seen as additional variables, we can restrict the discussion to binomials, i.e. polynomials being the difference of two power products $X^\alpha - X^\beta$, and we can even assume that in $k[X]/I$ the variables are invertible, since this can be handled e.g. adding one variable T and one polynomial $1 - T \prod X_i$.

These assumptions allow to use alternative ideal representations that do not restrict seriously the class of examples.

2. LATTICE POLLY CRACKER

2.1 Binomial ideals and lattices

Let I be a binomial ideal. We can associate with each binomial $b = X^\alpha - X^\beta \in I$ the vector $\mathbf{v}_b = \alpha - \beta \in \mathbf{Z}^n$. Conversely, given an element $\mathbf{v} \in \mathbf{Z}^n$ define $\alpha_i = \max(0, v_i)$, $\beta_i = -\min(0, v_i)$, and the binomial $b_v = X^\alpha - X^\beta$. Clearly, $\mathbf{v}_{b_v} = \mathbf{v}$, while $b = b_{\mathbf{v}_b} b'$, where b' is the GCD of the two monomials of b .

From a binomial ideal I we obtain a lattice, and from a lattice we obtain a binomial ideal; going from a binomial ideal I to the corresponding lattice and back, we obtain the saturation $I :^* \prod X_i = k[X, X^{-1}]I \cap k[X]$, that is a binomial ideal. We call a binomial ideal saturated with respect to the variables a *Lattice Ideal*.

A lattice usually is defined as a free Abelian group with a scalar product. We are rather interested in full-rank integer lattices, i.e. sublattices of $\mathbf{Z}^n$ of rank n , and the scalar product is not used explicitly, hence may be changed depending of the problem. The changes of coordinates in $\mathbf{Z}^n$ have to be unimodular, but not necessarily orthogonal.

Given a set of generators of an integer lattice, i.e. a set of elements of $\mathbf{Z}^n$, we have a corresponding ideal generated by the binomials corresponding to the vectors. This ideal might not be variable-saturated, but if the generating set contains a vector α such that all its components are positive, then the ideal contains $X^\alpha - 1$, and this shows that every variable is invertible in the quotient, hence the ideal is variable-saturated, and corresponds to the ideal associated to the lattice.

A term ordering defines a total ordering on the lattice, and a set of elements of the lattice is a Gröbner basis if the corresponding set of binomials is a Gröbner basis.

A Polly Cracker system based on a lattice ideal has some peculiarities. Since the normal form of a monomial modulo a binomial ideal is a monomial with the same coefficient, encyphering a polynomial is equivalent to encyphering its monomials (no interaction can happen between the monomials).

Because of this, a Polly Cracker based on a lattice ideal can be seen without any reference to polynomials and ideals, hence we will speak of “Lattice Polly Cracker”.

2.2 Quotient of lattices and normal forms

Given a lattice $L \subseteq \mathbf{Z}^n$ of rank n one wants to represent the elements of $\mathbf{Z}^n/L$ in a canonical form. There are three families of such normal forms:

Minkowski normal form with respect to a basis $B = \{\mathbf{b}_i\}$ of the lattice. For a vector $\mathbf{v} \in \mathbf{Z}^n$, with $\mathbf{v} = \sum a_i \mathbf{b}_i$, the normal form is given by $\mu_B(\mathbf{v}) = \mathbf{v} - \sum \bar{a}_i \mathbf{b}_i$, where $\bar{a}_i = [a_i]$, i.e. a_i rounded to the nearest integer. Instead of rounding, floor or ceiling can be used, giving variants of the normal form.

Voronoi normal form with respect to a metric. It associates $\mathbf{v}$ to the vector $\beta(\mathbf{v}) = \mathbf{v} - \mathbf{v}'$, where $\mathbf{v}'$ is the lattice vector closest to $\mathbf{v}$ with respect to the metric (a sparring method has to be designed when the closest vector is not unique).

Gröbner normal form with respect to a product-compatible total ordering τ on $\mathbf{Z}^n$. The ordering has to have 0 as minimum for $\mathbf{N}^n$ and is usually computed through a Gröbner basis G of L . It is denoted by $\gamma_\tau(\mathbf{v})$ or $\gamma_G(\mathbf{v})$, or simply $\gamma(\mathbf{v})$, and is the τ -smallest positive element equivalent to $\mathbf{v}$.

It is sometimes useful to consider shifted Gröbner normal forms, $\gamma_{G,\alpha}(\mathbf{v}) = \gamma_G(\mathbf{v} + \alpha) - \alpha$.

The Minkowski and l_2 -Voronoi normal forms coincide if the basis is orthogonal with respect to the scalar product. If a basis is “almost orthogonal”, the Minkowski and Voronoi normal forms may coincide too, on all elements or on a controlled subset; the Minkowski normal form is easy to compute if one knows the basis. The Gröbner normal form is (relatively) easy to compute if one knows the ordering and the Gröbner basis of the lattice with respect to the ordering. The problem is that the Gröbner basis is often very large, and its size may depend substantially on the ordering.

An element of $\mathbf{Z}^n$ is said to be *standard* if it coincides with its normal form. The cardinality of each standard set is of course the same, being the cardinality of the quotient, and it is equal to the absolute value of the determinant of a matrix whose columns are a basis of the lattice.

2.3 Lattice cryptosystems, GGH

Many cryptosystems based on lattices have been defined [17], but most have been broken, and/or are impractical, they are hence a good company for Polly Cracker systems; their attractive feature is efficiency, something that Polly Cracker can hardly claim. We will consider two such systems, the Goldreich-Goldwasser-Halevi (GGH) cryptosystem [8] in this section, and the NTRU cryptosystem [10] in the next one.

Lattice cryptosystems try to exploit the NP-hardness of the SVP (shortest vector problem) and CVP (closest vector problem) for lattices. SVP and CVP for the l_1 norm can be easily reduced to the computation of a Gröbner basis (SVP, approximated up to a factor of 2) or to the computation of the Gröbner normal form (CVP), with respect of a term-ordering compatible with the metric,¹ hence it is cryptographically justified to try and use the Gröbner normal form in a lattice, in a way similar to GGH.

¹Any binomial corresponds to a vector that has a l_1 norm that is the sum of the degree of the head and of the tail. Since the ordering is degree-compatible, the degree of the binomial is equal to the degree of the head, and is at least one half of the l_1 norm, hence the smallest polynomial has at most twice the l_1 norm of the l_1 -smallest vector in the lattice.

The normal form of an element α is the smallest positive vector equivalent to it. To remove the positivity assumption, add to every variable x_i a variable y_i , and equations $x_i y_i - 1$; the normal form with respect to this ideal identifies the shortest vector equivalent to α , hence $\alpha - \gamma(\alpha)$ is the closest vector.

This implies, of course, that computing the Gröbner normal form for a degree-compatible ordering in a full-rank integer lattice is NP-hard.

The GGH cryptosystem can be summarized as follows:

- Choose V as $\{s, -s\}^n$ for a suitable $s \in \mathbf{N}$.
- Choose L through a basis B such that the elements of V are in Minkowski normal form; B is the private key.
- Choose B' another basis of L as public key. Heuristics are given to ensure that the elements of V cannot be reconstructed from $\mu_{B'}$.
- A message is an element $\mathbf{m} \in L$, and a corresponding cryptogram is an element $\mathbf{c} \in \mathbf{m} + V$, obtained obscuring $\mathbf{m}$ through a random element $\mathbf{v} \in V$.

A cryptogram $\mathbf{c} = \mathbf{m} + \mathbf{v}$ is decyphered computing its normal form $\mu_B(\mathbf{c}) = \mathbf{v}$, and computing $\mathbf{l} = \mathbf{c} - \mu_B(\mathbf{c})$.

Any basis B'' can be used to decypher, provided that most elements of V are in Minkowski normal form, e.g. if B'' is “almost” orthogonal. The conjectured security of the cryptosystem relies on the difficulty of finding a basis that is almost orthogonal from a suitably generated B' .

The positions of $\mathbf{v}$ and $\mathbf{m}$ can be exchanged: take $\mathbf{v}$ as message, and add a random obscuring element in L . Clearly, retrieving $\mathbf{v}$ one can find $\mathbf{l}$ and conversely, so the two variants are equivalent. See [15] for more details.

This remark however means that, when the message is given by $\mathbf{v} \in V$, if $\mathbf{m} + \mathbf{v}$ is sent, $\mu_{B'}(\mathbf{v}) = \mu_{B'}(\mathbf{m} + \mathbf{v})$ is easily computed. This shows that the randomization effect of $\mathbf{m}$ is completely illusory: the cryptosystem is as (in)secure as the one in which the enciphering is done as $\mathbf{c} = \mu_{B'}(\mathbf{m})$, without any randomization. See also [15].

Our description is not how the system was presented in [8], where it is stated that decyphering is done through the Voronoi normal form, but since B and V are chosen in such a way that they are the same, the two descriptions are equivalent.

GGH has been broken [16] using the fact that all the elements of V are congruent mod $2s\mathbf{Z}^n$; this can be fixed taking a different V , but overall the choice of parameters is so fragile that it was impossible to fix it maintaining the practicability of the cryptosystem.

The variant presented in [15] specifies for B' the Hermite Normal Form (HNF), hence reducing considerably the length of the public key and the complexity of enciphering. Since the HNF can be computed in polynomial time, this variant is as secure as the original form is.

2.4 GB-GGH aka LPC

Using the Gröbner normal form instead of the Minkowski normal form for decyphering, and a different set V , we define a cryptosystem that is both a variation of GGH and a *bona fide* Binomial Barkee cryptosystem, i.e. a Lattice Polly Cracker. This could be called either GB-GGH, BinBarkee or LPC (the last being our choice).

Assume that we have a lattice L , a Gröbner basis G of the lattice with respect to a term ordering τ , and a set V of elements in Gröbner normal form. Let B be a basis of the lattice. We define the LPC cryptosystem as follows:

- The private key is G, τ, L ;
- The set of messages is V ;
- The public key is B .
- A message $\mathbf{m}$ is encyphered as $\mathbf{c} = \mu_B(\mathbf{m})$;
- $\mathbf{c}$ is decyphered as $\gamma_G(\mathbf{c})$ if $\gamma_G(\mathbf{c}) \in V$, otherwise it is decyphered as ERROR.

One of the main difficulties in defining a LPC instance is that to decypher one has to compute a Gröbner basis; in computing it, a term ordering has to be chosen, and the complexity varies wildly depending on the ordering: we will see later that a Lex Gröbner basis of a full-rank lattice of dimension n has n elements and can be computed in polynomial time, while the Gröbner basis with respect of a degree-compatible ordering of a generic full-rank lattice apparently has exponentially many elements. Remark that the doubly-exponential lower bound [14] can only happen for lattices that are not of full rank.

The definition of V rules out the possibility of using a Lex basis (for suitable choices of V the elements are not in Lex Gröbner normal form)

This means that the lattice L should not be generic, and the ordering should be tailored to V and L ; but the public presentation B of the lattice should efficiently conceal the information on the structure of L that could otherwise allow the computation of a Gröbner basis suitable for decyphering by an eavesdropper.

Remark that, while a Gröbner basis and γ_G require to fix a coordinate system to represent polynomials, μ_B is independent of the coordinates; publishing V however requires a coordinate system, not necessarily the same that is used to define G .

We will start discussing the case in which the coordinates are the same, will show that in this case the cryptosystem is insecure, and finally propose a setting in which a change of coordinates makes the system more secure. The conclusion is that an attack to retrieve the change of coordinates might break the cryptosystem, and we study how to make this attack hard to complete.

The set V should be chosen to avoid to disclose information on the staircase of G and on the private coordinates. The natural choice is to define V as the set of monomials in which the degree of each variable is limited by a uniform bound a , i.e. the set $S_a = \{0, 1, \dots, a\}^n$, or a subset, to avoid the possibility of some attacks. V of course has to be defined in the public coordinates, but has to be used in the private coordinates. This is the main difficulty in the definition of the cryptosystem.

We need G to have the property that every element of V is in γ_g normal form. If there is no change of coordinates, this means that there should be no element of G whose leading term has all the variables with degree $\leq a$. If a change of coordinates is performed under some restrictions we can still safely bound V . We define the *maximal inner width* $MIW(G)$ and *minimal outer width* $MOW(G)$; $MIW(G) = \min_{g \in G} (\max_i \deg_{x_i} Lt(g))$ and $MOW(G) = \max_{g \in G} (\max_i \deg_{x_i} Lt(g))$ respectively. Large MIW allows to use G in encyphering, small MOW allows to use G for signatures.

LPC with a trapdoor lattice as described below is a Polly Cracker cryptosystem that appears to be resistant to the known message attacks. The question is whether we can build a system that resists also to the attacks to the trapdoor information. This requires a change of coordinates, that might in turn require the use of a shifted Gröbner normal form.

We remark that any Gröbner basis G' with respect to any term ordering and any coordinate system can be used for decyphering, as long as V is composed of elements in $\gamma_{G'}$ normal form, e.g. as long as $MIW(G')$ is sufficiently large.

2.5 Signing through LPC

The decyphering procedure for a lattice ideal can be used to design a signature scheme, following the usual scheme of proving the signer identity showing that he can decypher a random message generated by a hash key. See also [8, 18].

There is a small asymmetry between decyphering and signing, caused by the need of concealing the shape of the staircase of the lattice ideal.

Let us recapitulate the situation. The public key allows to compute the Minkowski normal form $\mu(x)$ of any element x , and the private key allows to find the Gröbner normal form $\gamma(x)$ of any element x .

For encyphering, one has to specify a subset $M \subseteq \mathbf{Z}^n$ such that $x \in M \Rightarrow \gamma(\mu(x)) = x$. This means that M has to be a subset of the elements in Gröbner normal form.

For signatures, one has to commit to decypher (i.e. find an equivalent element in a prescribed region) any element x ; one has to choose a subset $S \subseteq \mathbf{Z}^n$ where the signature might lie, hence S has to be a superset of the elements in Gröbner normal form.

One cannot take $M = S = Can(I)$, the set of standard monomials of the private Gröbner basis, since this would reveal informations essential to the security.

One can instead take for M (resp. S) a subset (a superset) of power products in which the degree of every variable is upper (resp. lower) bounded by the MIW (resp. MOW).

We concentrate the discussion on encyphering, and only a few details on signature will be given.

2.6 Trapdoor lattice ideals

In a lattice Polly Cracker, (as in any other cryptosystem) there are two factors to consider: how to protect the private key, and how to protect the message.

The private key, that is a Gröbner basis used to decypher a message, should be of moderate size, and easily computable. The knowledge of the lattice is not sufficient to produce the Gröbner basis: one has to choose the ordering, and start from a suitable set of generators (a basis of the lattice) and of coordinates.

Because of the lattice description, standard lattice tools can be used, notably the Lenstra-Lenstra-Lovász lattice reduction algorithm [12] and Schnorr's modifications BKZ [19, 20]. These tools, when the dimension is small, can bring a basis of a lattice to a standard form, that is often convenient for some attacks, possibly after choosing a suitable scalar product.

This implies that, to hope to be secure, the dimension of the lattice, hence the number of variables in the associated polynomial ring, has to be large, probably more than 100. Gröbner bases of binomial ideals in hundreds of variables are extremely large (at least exponential in the number of variables) unless a very special structure is present, in the ideal and/or in the ordering. This structure may make the Gröbner basis manageable, either in general or just for some term ordering. For example, if an ideal is a sum of ideals generated by polynomials in disjoint sets of variables, then the Gröbner basis will be the union of the Gröbner bases, hence could be manageable with any term ordering.

We will construct ideals such that the Gröbner basis is relatively small only in a very restricted set of term orderings and coordinates, and use this set as part of a trapdoor. A set of coordinates is the other part of the trapdoor.

While the protection given by a basis of the lattice is only marginal, the protection given by the ordering and the choice of coordinates is much stronger. The dependence of the Gröbner basis from these data is highly unstable, and small variations of the ordering can produce wild differences in the size of the Gröbner basis.

A lattice Polly Cracker protocol requires to choose a set of standard monomials, and not every Gröbner basis can ensure that a given set of monomials is standard.

We use a block term ordering,² with DegRevLex ordering inside of each block, and the key information is the partition in blocks. A different partition and/or relative ordering of the blocks produces a Gröbner basis that is too hard to compute, but, more importantly, it will usually be unsuitable to allow decyphering, having MIW too low. The blocks are only apparent in a specific set of coordinates, and, as we will show, can be detected with standard tools from the rest of the information in that set of coordinates. The change of coordinates is intended to lock this information.

2.7 Twisted product lattices

Given lattices $L_1 \subseteq \mathbf{Z}^{n_1}$, $L_2 \subseteq \mathbf{Z}^{n_2}$ of full rank, a $L \subseteq \mathbf{Z}^{n_1} \times \mathbf{Z}^{n_2}$ is called a *twisted product* of L_1 and L_2 if the projection of L into $\mathbf{Z}^{n_1}$ is L_1 and the intersection of L with $\mathbf{Z}^{n_2}$ is L_2 . The ordering in $\mathbf{Z}^{n_1} \times \mathbf{Z}^{n_2}$ is a product ordering if any element (v_1, v_2) is positive whenever v_1 is positive. We will show below how a Gröbner basis of a twisted product lattice can be computed.

A twisted product lattice has a basis being the columns of a block matrix $\begin{pmatrix} A_1 & 0 \\ C & A_2 \end{pmatrix}$; A_1, A_2 are bases of L_1 and L_2 respectively. This basis will be private, and the private coordinate system will be chosen compatibly with the product structure; the public coordinate system will hide this structure, as we will see later.

Every full rank lattice of dimension larger than 1 allows a decomposition as twisted product, (e.g. reducing a basis into Hermite normal form), but a random decomposition is unlikely to be useful for the purpose of decyphering messages.

We show that if A'_1, A'_2 are matrices that describe Gröbner bases of the lattices L_1, L_2 , then one can build a matrix $\begin{pmatrix} A'_1 & 0 \\ C' & A'_2 \end{pmatrix}$ describing a Gröbner basis of L .

In order to simplify the description, we assume that A_1 and A_2 contain a column whose entries are all positive. This is always possible, because L_1, L_2 are of full rank, the quotient is finite, hence the lattice contains a vector whose coordinates are all positive. In this case, the Gröbner basis of the lattice is the Gröbner basis of the ideal generated by the columns of the matrix.

Consider a matrix M such that $A_1 M = A'_1$. Let $C'' = CM$. Then $\begin{pmatrix} A'_1 \\ C'' \end{pmatrix}$ defines the same lattice as $\begin{pmatrix} A_1 \\ C \end{pmatrix}$.

Now we can find a matrix M' such that $C' = A_2 M' + C''$ has all entries that are non-positive. The matrix $\begin{pmatrix} A'_1 & 0 \\ C' & A'_2 \end{pmatrix}$ generates the same lattice of $\begin{pmatrix} A_1 & 0 \\ C & A_2 \end{pmatrix}$. We want to show that this matrix represents a (non reduced) Gröbner basis of the lattice.

²i.e. a partition of the variables is defined, and comparison between monomials in different elements of the partition is lexicographic

Since C' is non-positive, the leading terms of $\begin{pmatrix} A'_1 & 0 \\ C' & A'_2 \end{pmatrix}$

are the same as the leading terms of A'_1 , because the variables of the second block are infinitely small with respect to the variables of the first block; hence the multiplicity of the ideal generated by the leading terms of the binomials of the columns of A'_1 is equal to the determinant of A_1 (that is non zero). The same for A_2 . The argument in [21] shows that the binomials are a Gröbner basis: adding a non redundant element will decrease the multiplicity, that is bound to be equal to the determinant, hence cannot be further reduced.

Remark that the Gröbner basis that we constructed need not to be reduced. Reducing it is however simple, being equivalent to reducing the tail of every basis element.

Remark also that $MIW(L) = \min(MIW(L_1), MIW(L_2))$ and $MOW(L) = \max(MOW(L_1), MOW(L_2))$

All the arguments of course extend to twisted products of several lattices.

2.8 Attacks to twisted product lattices

We show how to disclose a twisted lattice structure if the coordinates are known up to the order (shuffled but not changed). The attack often allows to find a Gröbner basis sufficiently similar to the original.

Remark that to break the private key it is not necessary to recover exactly the twisted product structure, i.e. the original decomposition. Any basis of the lattice would be OK provided that we a) can compute the Gröbner basis of the lattice with respect to some ordering; b) the basis computed is able to decypher a message, i.e. all the elements of V , or at least a non irrelevant part of them, are in Gröbner normal form.

Indeed, any lattice in any coordinate system allows a twisted product decomposition: it is sufficient to compute the column Hermite normal form; an invertible matrix in Hermite normal form is a block decomposition in blocks of size 1. However, when in Hermite normal form, the determinant is the product of the diagonal elements. Quite often, many diagonal elements are 1, and this will certainly happen if the original matrix has a determinant that is a product of few primes, e.g. if the original square matrices have prime determinant. But this also happens if the matrices are random. Using the twisted lattice decomposition given by a Hermite normal form has usually $MIW = 1$, and is hence unsuitable.

We have thus discovered an important fact: since a matrix in Hermite normal form is in twisted lattice form, with blocks of length 1, the Lex Gröbner basis can be straightforwardly computed, bringing with column elementary operation every element off the diagonal to be negative.

In particular, assume that the lower right corner of the matrix is m , and let z be the corresponding variable; this means that $z^m - 1$ is an element of the reduced Gröbner basis of the lattice, and this means that z is an element of order m in the multiplicative group of $k[X]/J$, where J is the lattice ideal. This remark allows to compute the multiplicative order of any variable, just putting the variable as last one, and computing an Hermite normal form. This method can be extended to any vector, through a change of coordinates.

More in general, from a basis of a lattice, one can compute the determinant, and the order of any element will be a divisor of the determinant.

Assume that we know that a lattice is a twisted product $L_1 \times L_2$. Then the elements of the second block will have order that divides the determinant of the block, while the generic elements have order that divides the determinant of the lattice. In generic conditions, this will allow to detect the block decomposition.

When the blocks of a lattice are identified, the attacker is in the position of finding a Gröbner basis of the lattice, with respect to a block ordering corresponding to the twisted product decomposition, exactly in the same position of the designer of the private key.

Protecting the block decomposition making it unreadable from the orders of the variables seems difficult, hence we need to protect the coordinates in which we have the original decomposition. This means that we have to publish the lattice in a basis that is different from the basis in which the private computations are performed.

To fix the notation, let A be the block matrix, whose columns are a basis of the twisted lattice. Changing the basis, i.e. disclosing AR , R being a unimodular matrix, is not sufficient, we need to multiply it to the left too, by a unimodular matrix S , obtaining SAR . We will moreover need a shift vector $\mathbf{s}$. This vector should be such that if $\mathbf{m} \in V$ then $A\mathbf{m} + \mathbf{s}$ is in Gröbner normal form.

Consider now the enciphering of a message $\mathbf{m} \in V$; it is encyphered as $\mathbf{c} = \mathbf{m} + B\mathbf{y}$, $\mathbf{y}$ being a suitable vector with integer coordinates, so that $B\mathbf{y}$ is a lattice element. To recover $\mathbf{m}$ one computes the normal form of $S^{-1}\mathbf{c} + \mathbf{s} = S^{-1}\mathbf{m} + AR\mathbf{y}$; $AR\mathbf{y}$ is in the private lattice, and reduces to 0, hence if $S^{-1}\mathbf{m} + \mathbf{s}$ is in normal form, then $\mathbf{m}$ can be retrieved through $S(\gamma(S^{-1}\mathbf{c}) - \mathbf{s})$.

The highlighted condition shows that we cannot choose a generic unimodular S , we need to choose it carefully together with V and $\mathbf{s}$. In practice, it turns out that the construction of S is more difficult than the construction of the twisted lattice, since the security of the cryptosystem relies on S , hence one can choose S and V first, choose $\mathbf{s}$ such that $S^{-1}V + \mathbf{s}$ is in the positive quadrant, then build the private lattice in such a way that $S^{-1}V + \mathbf{s}$ is in Gröbner normal form.

The tuning of this construction might be delicate, since a matrix S that is too sparse allows attacks to the private key through the Hermite normal form, while a generic matrix S might allow attacks to the message through LLL and Minkowski normal form.

One still has to choose the matrix R . Following the advice of [15], we might represent the lattice in (column) Hermite normal form. However, we rather propose to represent it in a slightly different form, defined as the Hermite normal form, but in which the elements off the diagonal are always negative instead of positive, and their absolute value satisfies the same bound defining the Hermite normal form. In this form, the basis of the lattice is a reduced Gröbner basis for the Lex ordering, and with respect to this basis the Minkowski normal form and the Gröbner normal form coincide, as it is easy to check. The difference is purely æsthetic: we have a cryptosystem in which the secret key and the public key are Gröbner bases of the same lattice, and encyphering and decyphering are made with the same algorithm.

2.9 Twisted product lattices for Polly Cracker

The choice of lattices for the building blocks of Polly Cracker has to be tuned considering efficiency and security.

Up to now, we have strived for feasibility, making some reasonable assumptions and some experiments. These conclusions might be modified after closer security considerations.

The Gröbner basis of a lattice of full dimension in n variables with size of the random entries about 20 with DegRevLex ordering has size bounded (experimentally) by 3^n , and usually one is not far from the limit. The computation is usually unfeasible for $n > 12$; $n = 8$ instead is fairly rapid, and the Gröbner basis has usually 4000–5000 elements.

With these lattices a *MIW* of 20, and a *MOW* of 100 are easy to obtain.

Exhaustive search attacks can be performed to the highest and lowest block, so it is reasonable to protect them with a larger size, while for the intermediate blocks a size of 4 (that usually gives a Gröbner basis of size 30; sizes larger than 50 are unusual) seems instead reasonable. The size of the exponents is of course bounded by the *MOW*.

A total dimension of the lattice of 128 might be good, although as low as 32 and as high as 512 do not differ much in size of the resulting Gröbner basis, since the bulk of it is given by the 10000 elements of the cornerstones; adding 120 or 4000 elements for the middle blocks is not that much different. The main difference is given by the increase of the intermediate exponents of the Gröbner reduction, but this too can be tamed.

Of course, reducing a monomial by a basis of 10000 elements is not trivial. We have developed ad-hoc algorithms, making special use of the elements with pure power head, that are rather efficient; we can perform a reduction (i.e. a decyphering) in less than 1 second, with an unoptimized prototype, and we hope to improve it at least 100 times with a dedicated implementation. We don't have (yet) a reasonable complexity estimate on decyphering.

It seems possible that for signatures the size of the private key can be considerably reduced, since a partial Gröbner basis is sufficient for signing. A private key of n^2 bytes seems hence possible. The complexity of signing does not seem however to be radically improved with the partial basis.

More technical details on the choice and its reasons will be contained in a forthcoming technical paper.

3. IMPROVING NTRU

3.1 Introduction

In this section we examine another cryptosystem, NTRU [10] that is not defined by a lattice, but allows attacks through an associated lattice. NTRU is industrially used, but is considered at least with suspicion for a series of weaknesses. We will show that with the use of multivariate polynomial algebra, and of two Gröbner bases that are part of the private key, one can design an improvement that considerably strengthens the cryptosystem without seriously degrading the performance. In particular, the known lattice attacks to the private key are made impossible since now the key construction has a trapdoor protected by a private (instead of a public) lattice, and the attacks to the message are made more difficult since the message now can have larger norm.

3.2 Description of NTRU

NTRU [10] is not a lattice cryptosystem, it is a polynomial algebra cryptosystem allowing a lattice interpretation.

The reason is that if ϕ is a monic polynomial in $\mathbf{Z}[X]$ of degree n , then $A = \mathbf{Z}[X]/(\phi)$ is isomorphic (as $\mathbf{Z}$ -module) to $\mathbf{Z}^n$, and ideals of A can be seen naturally as sub lattices of $\mathbf{Z}^n$, invariant under multiplication by X .

In the case of NTRU, $\phi = X^n - 1$. This means that multiplication by X is a cyclic permutation of the canonical basis of $\mathbf{Z}^n$, and that an ideal is a lattice invariant by this action of the cyclic group.

The setup of NTRU is given by n , two elements q, p of A that are coprime (i.e. that they generate (1) in A ; to simplify the description, q is a prime number in the range 100-500 and p is 2 or 3, but different choices are possible).

A polynomial in $\mathbf{Z}[X]$ is called *small* if its coefficients are at most $p/2$ in absolute value, and its support is “small” (the bound depends on the setting and the context, it may be different for polynomials used differently); it is called *moderate* if its coefficients are smaller than $q/2$ in absolute value.

The private key is composed of two small polynomials f and g , both invertible mod q and with f also invertible mod p . The inverses are f_q^{-1} , f_p^{-1} respectively (the inverse of g is never used). The public key is $h = gf_q^{-1}$. A message is a small polynomial m , and is encyphered into a cryptogram $c = phr + m \bmod q$, where r is a random small polynomial. The parameters are chosen to ensure with high probability that $pgr + fm$ is moderate.

Decyphering is done as follows: compute $fc \equiv pgr + fm$ and represent it mod q as a moderate polynomial $\phi \in A$. Assuming that $prg + fm$ is moderate (and this is at least probabilistically granted by the size of the support) then $\phi \equiv fm \bmod p$, hence $f_p^{-1}\phi \equiv m \bmod p$, and because m is small it is recovered from its value mod p .

If $p = 2$ the only change needed is that the message m is a polynomial with coefficients in $\{0, 1\}$ but, before encyphering, the signs are randomly assigned.³

3.3 NTRU Lattice attacks

NTRU can be attacked interpreting its operations in a sublattice of $A \times A$, see [10, 4]. Consider the sublattice L of $A \times A$ generated by $(q, 0)$ and $(h, 1)$. This lattice has dimension $2n$ and determinant q^n , hence the shortest vector has norm comparable to $\sqrt{q}$, and because of the symmetry there is generically a space of dimension n of vectors of minimal length.

Since $fh \equiv g \bmod q$, the lattice contains (g, f) , that is (because of the smallness of f and g , required to make decyphering highly probable) an element of minimal norm in the lattice L . Lattice reduction algorithms are likely to find (g, f) , and indeed some early variants with smaller length have been broken with this method.

This attack, if successful, recovers either the private key, or an equivalent key; a similar attack can be attempted to recover a message, since (ph, r) is the lattice closest vector (under a suitable norm) to $(c, 0)$. In turn, it is likely that the closest vector problem can be solved through the shortest vector problem in the sublattice of $A \times A \times \mathbf{Z}$ generated by L and $(c, 0, 1)$. The shortest vector itself is assumed to be $(m, r, 1)$.

The attack does not work in practice for the suggested parameters, because, fine-tuning them, the shortest vector

³Thy need to be deterministically assigned if one wants to reconstruct r ; in our extension this is needed to avoid some chosen cyphertext attacks.

of L is not much shorter than the vectors in a large subspace, and the standard algorithms in that case either do not work or have high complexity.

One can remark that the security of the private key relies on having f and g of sufficiently large norm, the security of the message relies on having r and m sufficiently large, but if they are too large then $gr + fm$ is no longer moderate, hence the message cannot be decyphered. If one has to preserve decyphering, strengthening the key weakens the message, and conversely. The path to preserve both is narrow, and might require to increase n , hence message length and encyphering/decyphering costs.

3.4 GB-NTRU

Clearly, it is possible to use multivariate polynomials instead of univariate in NTRU. This gives no substantial advantage, but is essential for our extension. Because of the dimension of A it is unreasonable to use more than 4 variables, and reasonable examples with more than 2 variables are exceptional, so we limit the description to 2 variables.

The ring A will be $\mathbf{Z}[X, Y]/(X^{n_1} - 1, Y^{n_2} - 1)$. In our extension we will need $n_1 = n_2$, so to simplify notations we take directly $n = n_1 = n_2$. Everything else is unchanged, except that the lattice is invariant under a group that is not cyclic.

One might also consider polynomials different from $X^n - 1$, but, with the exception of $X^n + 1$, multiplication by X is not an isometry, and this would just make decyphering much more difficult.

The basic idea of our extension, that we call GB-NTRU, is to use a different lattice for encyphering and decyphering; more precisely, we define an ideal $I \supseteq (X^n - 1, Y^n - 1)$ and keep it secret. It will be used for key creation and decyphering, while the public key is indistinguishable from a (bivariate) NTRU public key, and encyphering is performed in the same way.

Disclosing the private ideal will allow the classic attacks to NTRU, hence it is vital that exhaustive search for ideals containing $(X^n - 1, Y^n - 1)$ in $\mathbf{Z}[X, Y]/q$ is unfeasible; and maximizing the number of such ideals just means that $x^n - 1$ splits in linear factors in $\mathbf{Z}/q$, i.e. n should divide $q - 1$.

This condition, combined with the condition that q be smaller than the length of messages (that will be used later) shows that an univariate GB-NTRU is inferior, as well as a bivariate GB-NTRU with $n_1 \neq n_2$.

3.4.1 Choosing the private ideal

Under the condition that q is prime and $n \mid (q - 1)$, any ideal of $\mathbf{Z}/q[X, Y]$ containing $(X^n - 1, Y^n - 1)$ corresponds to a subset S of the set of points (x, y) such that $x^n = y^n = 1$. A point (x, y) corresponds to the ideal $I_{(x, y)} = (X - x, Y - y)$ of the polynomials vanishing in (x, y) and S corresponds to $\bigcap_{(x, y) \in S} I_{(x, y)}$; moreover $(X^n - 1, Y^n - 1)$ itself corresponds to the set S_n of all solutions of $(X^n - 1, Y^n - 1)$. We will rather use a complementary notation, indicating $J_T = I_{S_n \setminus T}$.

The ideal $J_T/(q, X^n - 1, Y^n - 1)$ is a vector space of dimension equal to the cardinality of T , and a basis is composed of $(X^n - 1)(Y^n - 1)/(X - x)(Y - y)$ where $(x, y) \in T$.

3.4.2 Preparing the public key

The preparation of the public key is simple, but good working keys are not easy to find. If a key works well or

not has to be checked after the preparation, and depends on the sublattice of the ideal J_T .

We choose random small polynomials f and g in A that are invertible mod J_T . This means that f and g do not vanish on any point of $S_n \setminus T$. The size of the support of f and g is for now left undetermined, and will be discussed later.

We check that f is invertible mod p too (the case in which this assumption is relaxed will be discussed later).

Although we have discussed T before f and g , it is useful to choose f and g first, and T after them. The point is that f is invertible mod J_T if and only if any zero at a point of S_n is contained in T , so choosing T after f and g makes more likely that f and g have roots, and this might be useful.

We now build $f', g' \in A$, $f' \equiv f$ and $g' \equiv g \pmod{J_T}$; this means that we add to f, g a random element of J_T , and we impose that f', g' are invertible mod S_n . Now f', g' are no longer small, we compute a moderate polynomial $h \equiv f'^{-1}g' \pmod{q}$, and h will be the public key.

We have to check that the combination of the public key with the private key, consisting of (T, f, g) , allows decyphering.

Retrieving f', g' or f, g from h through a short element in the NTRU lattice is no longer possible, since (g', f') is not short, and (g, f) is not in the NTRU lattice: (g, f) is in a larger lattice that is private, i.e. J_T , and if T is sufficiently large, then exhaustive search is impossible.

One can always find a T' such that h can be factored as a product of small polynomials mod $J_{T'}$, but usually with T' too large to give a key allowing decyphering.

3.4.3 Encyphering

Encyphering is done like in NTRU: a message is a small polynomial m , r is a random small polynomial and m is encyphered as $c = phr + m$.

Now although we have the same encyphering of NTRU, it can be made more robust. We have remarked that the larger is the support of r and m , the more difficult it is to break the message through the NTRU lattice. But in NTRU one cannot increase the support without either weakening the key or making decyphering failures more likely, or even decyphering almost always impossible. In GB-NTRU instead the strength of the public key lies in the private lattice, and the support of f and g can be kept very small, just what is needed to protect the private lattice from exhaustive search attacks. This allows to increase the support of r and m in such a way that $(pgr, fm, 1)$ has norm larger than $\sqrt{q}$, hence is not a shortest vector in $(q, 0, 0), (h, 1, 0), (c, 0, 1)$. This means that the attacks to the message will be harder than in the original NTRU, hence the same level of security might be obtained with shorter lengths.

3.4.4 Decyphering

The receiver now has $c = ph + m$ and wants to find m . He computes $fc \equiv gpr + fm \pmod{J_T}$, hence $fc = gpr + fm + \alpha$, $\alpha \in J_T$. α is unknown to the receiver, since it depends on r , and $h' = gpr + fm + \alpha$ is not moderate (while $gpr + fm$ is—conjecturally—moderate). We try to guess α , conjecturing that α is the closest vector to h' of J_T (seen as lattice). If we have a “good” basis of J_T , and this can be checked at key creation, the closest vector can usually be determined through the Minkowski normal form (Babai Round-off algorithm). If J_T is not sufficiently good, or anyway the decyphering fails, (this means that $h'' = h' - \alpha$, the candidate $gpr + fm$, does

not satisfy the bounds) one can use the Babai closest plane algorithm CPA, [1], but this is usually much more expensive, and should be avoided if possible; hence only keys for which the need of using the CPA has experimentally low probability should be used.

From now on, everything proceeds like in NTRU: let h'' be the candidate computed above, reduce $h'' \pmod{p}$, multiply by f_p^{-1} ; the result will be m , unless some of the conjectures (the fact that α is the closest vector and that $gpr + fm$ is moderate) are false. But as in NTRU we can tune the parameters to ensure that the conjectures are true with high probability. At the end, it is advisable to reconstruct r and m , and check that they satisfy the specifications. This might identify decyphering errors, and allow to avoid chosen cyphertext attacks.⁴

3.5 When f_p is not invertible

In some case (for some choices of n and p) it is very difficult to ensure that a random polynomial is invertible mod p . This happens in particular when the splitting field of $X^n - 1$ has low degree over $\mathbf{Z}/p$.

In that case, identify an ideal $J' \subseteq A$ such that $fJ' \subseteq pA$ —one can take $pA : (f)$. Then f will be invertible mod J' , the decyphering computations will be mod J' instead of being mod p .

The decyphered polynomial is ambiguous, but restricting the support of m to a subset of monomials independent mod J' , the ambiguity is solved. In practice, this means forbid the few largest monomials, and can be integrated in the protocol to avoid disclosing informations on f .

3.6 Parameters for GB-NTRU

The need of having n prime, $n|q$ and a preference for $q < n^2$ to increase the message security considerably limits the set of acceptable n, q pairs.

We have experimented a few settings; we limited ourselves to two settings, that has proven to be reasonable, for extensive testing. We chose $q = 131$ or $q = 157$, $n = 13$, $p = 2$, $|T| = 8$, size of the support of f and g equal to 11 and of the support of r equal to 60. With these choices the density of successful keys is more than 1/1000. For each a large set of messages have been tested for robustness, and all have shown to resist the standard attacks.

While GB-NTRU does not improve NTRU with respect to provable security, it seems to allow shorter block lengths. Bivariate polynomial arithmetic in degree n has the same cost as univariate polynomial arithmetic in degree n^2 , hence the size of the public key and the cost of encyphering only depend on the block length, that is reduced. There is some loss of performance because the arithmetic with respect a prime modulus q is marginally more costly than arithmetic modulo 2^m , that is the suggested choice for q in NTRU, but this too is compensated by the shorter block length. Decyphering is however more costly, and we have not yet tested in deep the probability of decyphering failures, that are harder to analyze for GB-NTRU than for NTRU.

⁴This is a sketch of a chosen cyphertext attack, and how reconstructing the random element used in encoding will counter it: let c be a cyphertext, and let $\alpha = (X^n - 1)(Y^n - 1)/(X - a)(Y - b)$; then $(a, b) \in T \Leftrightarrow c$ and $c + \alpha$ are decyphered in the same way; this would allow to reconstruct the private ideal, testing all the points. If instead r is checked, $c + \alpha$ is recognized as invalid.

4. IMPLEMENTATION

We have implemented a prototype using CoCoA [3] (CoCoA-4 for the scripting language, CoCoALib-5 for the polynomial algebra), 4ti2 [7] for the toric Gröbner basis computation, NTL [20] for lattice computations. We plan to implement everything in C++.

Some full-size challenges, worked out examples, computation statistics, as well as an expanded version of the present paper and relevant code are (or will be) available at the page <http://posso.dm.unipi.it/crypto>.

5. CONCLUSIONS: BARKEE WAS WRONG. OR MAYBE NOT?

We have shown that Gröbner bases can be used to build public key cryptosystems by building two cryptosystems.

The first, a lattice cryptosystem of the Barkee family (AKA Polly Cracker) apparently resists to all known attacks to Polly Cracker and to lattices, although it might be subject to different types of attacks, and notwithstanding the size of the private key can be considered practical. Unfortunately the decyphering is rather slow, and we do not have yet a bound on the complexity.

In the second, of a completely different type, the Gröbner basis is surprisingly used only in the private key, and improves a currently used commercial cryptosystem.

Can we conclude that Barkee was wrong?

The second cryptosystem falsifies Barkee's title, not Barkee's argument, and the security of the first example has not yet undergone a serious public scrutiny to decide if it will be another failure and Barkee was right, or if this or a better incarnation will show that Barkee was wrong. So now we expect the reactions of Barkee's friends.⁵

6. REFERENCES

- [1] L. Babai. On Lovász' lattice reduction and the nearest lattice point problem *Combinatorica*, 6(1):1–13, 1986.
- [2] B. Barkee, D. C. Can, J. Ecks, T. Moriarty, and R. F. Ree. Why you cannot even hope to use Gröbner bases in Public Key Cryptography: an open letter to a scientist who failed and a challenge to those who have not yet failed. *J. Symb. Comput.*, 18(6):497–501, 1994.
- [3] CoCoATeam. Cocoa: a system for doing Computations in Commutative Algebra. Available at <http://cocoa.dima.unige.it>.
- [4] D. Coppersmith, A. Shamir. Lattice attacks on NTRU. In *Advances in Cryptology—EUROCRYPT '97, Lecture Notes in Comput. Sci.*, 1233:52–61, 1997.
- [5] J.-C. Faugere. A new efficient algorithm for computing Grobner bases (F_4). in *J. of Pure and Appl. Algebra*, 139(1):61–88, 1999.
- [6] M. Fellows and N. Koblitz. Combinatorial Cryptosystems Galore! In *Finite fields: theory, applications, and algorithms (Las Vegas, NV, 1993)*, volume 168 of *Contemp. Math.*, pages 51–61. Amer. Math. Soc., Providence, RI, 1994.
- [7] 4ti2 team. 4ti2—A software package for algebraic, geometric and combinatorial problems on linear spaces. Available at <http://www.4ti2.de>
- [8] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key Cryptosystems from Lattice Reduction Problems. In *Advances in cryptology—CRYPTO '97, Lecture Notes in Comput. Sci.*, 1294:112–131, 1997.
- [9] J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. H. Silverman and W. Whyte Digital Signatures Using the NTRU Lattice In *Topics in Cryptology—CT-RSA 2003, Lecture Notes in Comput. Sci.*, 2162:122–140. 2003.
- [10] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: a Ring-Based Public Key Cryptosystem. In *Algorithmic number theory, Lecture Notes in Comput. Sci.*, 1423:267–288. 1998. See also <http://www.ntru.com>.
- [11] D. Hofheinz and R. Steinwandt. A "differential" attack on Polly Cracker. *Int. J. Inf. Secur.*, 1:143–148, 2002.
- [12] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring Polynomials with Rational Coefficients. *Math. Ann.*, 261(4):515–534, 1982.
- [13] F. Levy-dit Vehel, M. Marinari, T. Mora, L. Perret, and C. Traverso. A Survey on Polly Cracker Systems. In *Gröbner Bases, Coding, and Cryptography*, RISC book series. Springer, Heidelberg, (to appear).
- [14] E.W. Mayr, A.R. Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Adv. Math.*, 46(3):305–329, 1982.
- [15] D. Micciancio. Improving Lattice Based Cryptosystems Using the Hermite Normal Form. In *CaLC '01: Revised Papers from the International Conference on Cryptography and Lattices*, pages 126–145, London, UK, 2001. Springer-Verlag.
- [16] P. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97. in *Advances in Cryptology—Crypto'99, Lecture Notes in Comput. Sci.*, 1666:288–304, 1999.
- [17] P. Q. Nguyen and J. Stern. The Two Faces of Lattices in Cryptology. In *Selected Areas in Cryptography, Lecture Notes in Comput. Sci.*, 2146:146–180, 2001.
- [18] T. Plantard, W. Susilo and K. T. Win. A Digital Signature Scheme Based on CVP_∞ . *Public Key Cryptography, PKC 2008, Lecture Notes in Comput. Sci.*, 4939:288–307, 2008
- [19] C.-P. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Math. Programming*, 66(2, Ser. A):181–199, 1994.
- [20] V. Shoup. NTL: A Library for doing Number Theory. Available at <http://www.shoup.net/ntl/>.
- [21] C. Traverso. Hilbert functions and the Buchberger algorithm. *J. Symbolic Comput.*, 22(4):355–376, 1996.

⁵Boo Barkee died several years ago, old and full of days. This is not surprising, since he was a dog, and his paper was written 14 years ago.

We profit of this footnote to dispel some false rumors on Boo Barkee and his coauthors. These rumors said that Boo Barkee was dead, and we can confirm, but also said that he was a Greek general.

This false information probably originates from his address in Ithaca, NY; people probably misread the state.

As far as Barkee's co-authors are concerned, Deh Cac Can is known to sign right to left, Julia Ecks was closely connected with Boo Barkee, Moriarty is still struggling with his archenemy, and the anonymity of R.F.Ree (Miskatonic University, Arkham, MS) should forever be preserved.

Computing the Multiplicity Structure from Geometric Involutive Form *

Xiaoli Wu and Lihong Zhi

Key Laboratory of Mathematics Mechanization
AMSS, Beijing 100190, China

xlwu@amss.ac.cn, lzhi@mmrc.iss.ac.cn
<http://www.mmrc.iss.ac.cn/~lzhi/>

ABSTRACT

We present a method based on symbolic-numeric reduction to geometric involutive form to compute the primary component and the differential operators for an isolated singular solution of a polynomial ideal. The singular solution can be exact or approximate. If the singular solution is known with limited accuracy, then we propose a new method to refine it to high accuracy.

Categories and Subject Descriptors: G.1.5 [Mathematics of Computing]: Roots of Nonlinear Equations; I.1.2 [Symbolic and Algebraic Manipulation]: Algebraic Algorithms

General Terms: Algorithms, Theory

Keywords: Involutive System, Numerical Linear Algebra, Differential Operator, Index, Multiplicity

1. INTRODUCTION

Consider an ideal I generated by a polynomial system $F = \{f_1, \dots, f_t\}$, where $f_i \in \mathbb{C}[x_1, \dots, x_s]$, $i = 1, \dots, t$. For a given isolated singular solution $\hat{\mathbf{x}} = (\hat{x}_1, \dots, \hat{x}_s)$ of F , suppose Q is the isolated primary component whose associate prime is $P = (x_1 - \hat{x}_1, \dots, x_s - \hat{x}_s)$, we use symbolic-numeric method based on the geometric jet theory of partial differential equations introduced in [36, 37, 43] to compute the index ρ and multiplicity μ , such that $Q = (I, P^\rho)$ and $\mu = \dim(\mathbb{C}[\mathbf{x}]/Q)$. The multiplication structure of the quotient ring $\mathbb{C}[\mathbf{x}]/Q$ is computed from the null space of the involutive form of Q . The differential operators are determined by computing the normal form of a polynomial with undetermined coefficients up to degree $\rho - 1$. If the singular solution is only known with limited accuracy, then the primary ideal Q has a cluster of solutions. A refined solution with higher accuracy can be obtained by averaging the eigenvalues of each multiplication matrix [5].

*This research was partially supported by NKBRPC (2004CB318000) and the Chinese National Natural Science Foundation under Grant 10401035.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISSAC'08, July 20–23, 2008, Hagenberg, Austria.

Copyright 2008 ACM 978-1-59593-904-3/08/07 ...\$5.00.

Inspired by recent works in [8, 9], we apply the involutive criterion to the truncated coefficient matrices formulated from the Taylor series expansions of polynomials in prolonged systems of F at $\hat{\mathbf{x}}$ to order k . The number of columns of these coefficient matrices is fixed by $\binom{k+s-1}{s}$. The differential operators can be obtained from the null space of the truncated coefficient matrix of the involutive system. Our algorithm for computing differential operators could be regarded as a primal version of the one given in [9].

Our method for computing index and multiplicity is also related to the one presented in [2]. The algorithm they presented for computing multiplicity is based on Bayer and Stillman's theory on regularity [3]. It was pointed out in [21] that the concept of involutivity of symbol is equivalent to its Mumford regularity. Our criterion for involutivity is similar to their stopping criterion for regularity. However we do not need homogenization procedure while the algorithm in [2] works for homogenous polynomials.

If a singular solution is only known with limited accuracy, by choosing a tolerance, we can compute the index, multiplicity and differential operators for this approximate singular solution. It is well known that numeric computations deeply depend on the choice of tolerance. In order to obtain accurate information about the multiplicity structure, we propose a method to improve the accuracy of the singular root.

Suppose $\hat{\mathbf{x}} = \hat{\mathbf{x}}_{\text{exact}} + \hat{\mathbf{x}}_{\text{error}}$ where $\hat{\mathbf{x}}_{\text{exact}}$ denotes the exact singular solution of F . We observe that a good approximation $\hat{\mathbf{y}}$ of $-\hat{\mathbf{x}}_{\text{error}}$ can be computed from the null vectors of the truncated coefficient matrix of the involutive system. The singular solution $\hat{\mathbf{x}} + \hat{\mathbf{y}}$ has higher accuracy compared with $\hat{\mathbf{x}}$. We can apply our procedure iteratively to $\hat{\mathbf{x}} + \hat{\mathbf{y}}$ with a smaller tolerance. A singular solution accurately to the full machine precision can usually be obtained in less than 3 iterations, as is shown in our experiments. It is still not clear how our refinement procedure related to the methods in [18, 19, 20, 32, 33]. The column dimension of the matrix we used for refining the approximate singular solution is $\binom{\rho+s}{s}$. Our algorithm for refining an approximate singular solution is not efficient when index ρ is big. We notice that the number of deflations of the algorithms in [19, 20] is not closely related to the index, their algorithms can be very efficient for singular solutions with large index.

All algorithms we present in this paper have been implemented in Maple 11. We give two examples to illustrate our methods along the paper. We also show the test results for a set of benchmark problems. All computations are done in Maple 11 with Digits := 14.

2. ISOLATED PRIMARY COMPONENT

2.1 Preliminaries

The following paragraphs give a brief outline of the notations and tools we use throughout this paper. We refer to [6, 41] for detailed introduction.

Definition 1. Let I be an ideal in the ring of polynomials over complex field denoted by $\mathbb{C}[\mathbf{x}] = \mathbb{C}[x_1, \dots, x_s]$. Let f, g be arbitrary elements in $\mathbb{C}[\mathbf{x}]$.

- I is prime if $fg \in I \implies f \in I$ or $g \in I$.
- I is primary if $fg \in I \implies f \in I$ or $g^m \in I$ for some positive integer m .
- I is radical if $f^m \in I \implies f \in I$.
- The radical of I is the set

$$\sqrt{I} = \{f \mid f^m \in I \text{ for some integer } m \geq 1\}.$$

It should be noted that $\sqrt{I}$ is an ideal. Every prime ideal is a radical ideal and the radical of a primary ideal is a prime ideal. An ideal is finitely generated if there exists a finite list of elements $f_1, f_2, \dots, f_t \in I$ such that every element in I can be written as a $\mathbb{C}[\mathbf{x}]$ -linear combination of $f_1, f_2, \dots, f_t$, and denoted by $I = (f_1, f_2, \dots, f_t)$.

Definition 2. If P and Q are ideals and have the property that (1) $fg \in Q$ and $f \notin Q$ implies $g \in P$, (2) $Q \subseteq P$, (3) $g \in P$ implies $g^\rho \in Q$ for some positive integer ρ , then Q is primary and P the prime ideal belonging to Q .

If Q is a primary ideal then $P = \sqrt{Q}$ is the prime ideal belonging to Q and Q is called P -primary.

Definition 3. Every polynomial ideal has an irredundant primary decomposition, i.e. $I = \bigcap_{i=1}^r Q_i$, where Q_i are primary, $Q_i \not\subseteq \bigcap_{j \neq i} Q_j$. We call Q_i a primary component (ideal) of I . Q_i is said to be isolated if no prime ideal belonging to Q_j , $j \neq i$, is divisible by a prime ideal belonging to Q_i .

Definition 4. ρ is called the index of a primary ideal Q if ρ is the minimal nonnegative integer such that $\sqrt{Q}^\rho \subseteq Q$.

Theorem 1. [41] Suppose the polynomial ideal I has an isolated primary component Q whose associated prime P is maximal, and ρ is the index of Q .

If $\sigma < \rho$, then

$$\dim(\mathbb{C}[\mathbf{x}]/(I, P^{\sigma-1})) < \dim(\mathbb{C}[\mathbf{x}]/(I, P^\sigma)). \quad (1)$$

If $\sigma \geq \rho$, then

$$Q = (I, P^\rho) = (I, P^\sigma). \quad (2)$$

Corollary 1. If a polynomial ideal I has an isolated primary component Q whose associated prime P is maximal, then the index ρ of Q is less than or equal to the multiplicity μ of Q .

Proof: The multiplicity μ of the isolated primary component Q is equal to the dimension of the quotient algebra $\mathbb{C}[\mathbf{x}]/(I, P^\rho)$. Since the dimension of $\mathbb{C}[\mathbf{x}]/(I, P^\sigma)$ increases strictly until $\sigma = \rho$, the multiplicity μ is bigger than or equal to the index ρ . $\square$

2.2 SNEPSolver

Consider a polynomial system $F = \{f_1, \dots, f_t\}$, where $f_i \in \mathbb{C}[x_1, \dots, x_s]$ is of degree d , $i = 1, \dots, t$ and $s \leq t$. The system can be written as

$$M_d^{(0)} \cdot [x_1^d, x_1^{d-1}x_2, \dots, x_s^2, x_1, \dots, x_s, 1]^T = [0, \dots, 0]^T$$

in terms of its coefficient matrix $M_d^{(0)}$. Here and hereafter, $[\dots]^T$ means the transposition. Further, $[\xi_1, \xi_2, \dots, \xi_s]$ is one of the solutions of the polynomial system, if and only if

$$[\xi_1^d, \xi_1^{d-1}\xi_2, \dots, \xi_s^2, \xi_1, \dots, \xi_s, 1]^T$$

is a null vector of the coefficient matrix $M_d^{(0)}$.

Since the number of monomials is usually much greater than the number of polynomials, the dimension of the null space can be large. Completion methods for polynomial ideals based on critical pairs [1, 10, 11, 17, 24, 25, 28, 29, 30, 31, 39, 40] aim to include additional polynomials belonging to the ideal generated by F , until the normal form is determined capable of deciding membership of the ideal. The method in [35, 36, 37, 43] focuses on direct methods to calculate and minimize these dimensions by using the criterion of involution for PDE R [15, 34]. Here R is equivalent to polynomial system F by the bijection

$$\phi : x_i \leftrightarrow \frac{\partial}{\partial x_i}, \quad 1 \leq i \leq s.$$

In the following, we briefly explain the symbolic-numeric elimination method in the language of polynomial algebra. We study the variety

$$V(F) = \left\{ [\mathbf{x}^d, \dots, 1] \in \mathbb{C}^{N_d} \mid M_d^{(0)} \cdot [\mathbf{x}^d, \dots, 1]^T = \mathbf{0} \right\},$$

where $N_d = \binom{d+s}{s}$, $\mathbf{x}^j$ denotes all monomials of total degree equal to j . All distinct monomials are regarded as independent variables and $V(F)$ is simply the null space of $M_d^{(0)}$.

A single prolongation of the system F is to multiply the polynomials in F by variables, so that the resulting augmented system has degree $d+1$. Successive prolongations of the system yield $F = F^{(0)}, F^{(1)}, F^{(2)}, \dots$, and a sequence of corresponding linear constant matrix systems:

$$M_d^{(0)} \cdot \mathbf{v}_d = \mathbf{0}, M_d^{(1)} \cdot \mathbf{v}_{d+1} = \mathbf{0}, M_d^{(2)} \cdot \mathbf{v}_{d+2} = \mathbf{0}, \dots$$

where $\mathbf{v}_i = [\mathbf{x}^i, \mathbf{x}^{i-1}, \dots, \mathbf{x}, 1]^T$.

A single geometric projection is defined as

$$\pi(F) = \left\{ [\mathbf{x}^{d-1}, \dots, 1] \in \mathbb{C}^{N_{d-1}} \mid \exists \mathbf{x}^d, M_d^{(0)} \cdot [\mathbf{x}^d, \dots, 1]^T = \mathbf{0} \right\}.$$

The projection operator π maps a point in $\mathbb{C}^{N_d}$ to one in $\mathbb{C}^{N_{d-1}}$ by eliminating the monomials of the highest degree d . A numeric projection operator $\hat{\pi}$ based on singular value decomposition (SVD) was purposed in [4, 36, 42]. We first find the singular value decomposition $M_d^{(0)} = U \cdot \Sigma \cdot V$. The approximate rank r of $M_d^{(0)}$ is the number of singular values bigger than a fixed tolerance. The tolerance is chosen according to the number of correct digits for the coefficients of the input polynomials. The dimension of F is defined as the dimension of the null space of $M_d^{(0)}$, so we have $\dim F = \dim \text{Nullspace}(M_d^{(0)}) = N_d - r$. Deleting the first r rows of V yields an approximate basis for the null space of $M_d^{(0)}$. To estimate $\dim \hat{\pi}(F)$, the components

of the approximate basis for the null space of $M_d^{(0)}$ corresponding to the monomials of the highest degree d are deleted. This projected basis yields an approximate spanning set for $\hat{\pi}(F)$. Application of the SVD to each of these approximate spanning sets yields the approximate dimensions of $\hat{\pi}(F)$, $\hat{\pi}^2(F)$, $\hat{\pi}^3(F)$, ..., which are required for the approximate involutive form test.

The symbol matrix of polynomials of degree d is simply the submatrix of the coefficient matrix $M_d^{(0)}$ corresponding to the monomials of the highest degree d . One of the most important requirements of involutive systems is that their symbols are involutive. The following criterion of involution for zero dimensional polynomial systems is given in [43].

Theorem 2. [43] *A zero dimensional polynomial system F is involutive at order m and projected order ℓ if and only if $\pi^\ell(F^{(m)})$ satisfies the projected elimination test:*

$$\dim \pi^\ell(F^{(m)}) = \dim \pi^{\ell+1}(F^{(m+1)}), \quad (3)$$

and the symbol involutive test:

$$\dim \pi^\ell(F^{(m)}) = \dim \pi^{\ell+1}(F^{(m)}). \quad (4)$$

The following algorithm given in [37, 43] solves zero dimensional polynomial systems based on the symbolic-numeric completion method.

Algorithm 1. *SNEPSolver*

Input: A zero dimensional ideal $I = (f_1, \dots, f_t)$ where the polynomials are in $\mathbb{C}[\mathbf{x}]$ of degree d and a tolerance τ .

Output: Dimension of the quotient ring $\mathbb{C}[\mathbf{x}]/I$ and its multiplication matrices $M_{x_1}, \dots, M_{x_s}$.

- Apply the symbolic-numeric completion method to $F = \{f_1, \dots, f_t\}$ with tolerance τ , we obtain the table of $\dim \hat{\pi}^\ell(F^{(m)})$.
- We seek the smallest m such that there exists an ℓ with $\hat{\pi}^\ell(F^{(m)})$ approximately involutive, i.e., satisfying the conditions (3, 4). If there are several such values for the given m , then choose the largest such ℓ .
- The number of solutions of polynomial system F is $d = \dim(\mathbb{C}[\mathbf{x}]/I) = \dim \hat{\pi}^\ell(F^{(m)})$.
- The multiplication matrices $M_{x_1}, \dots, M_{x_s}$ are formed from the null vectors of $\hat{\pi}^\ell(F^{(m)})$ and $\hat{\pi}^{\ell+1}(F^{(m)})$.

Remark 1. Instead of choosing monomials to form a normal set of size d , we compute the SVD of the approximate basis of the null space of $\hat{\pi}^{\ell+1}(F^{(m)})$. According to (4), the first d left singular vectors permit a stable representation of the other rows in the approximate basis of the null space of $\hat{\pi}^\ell(F^{(m)})$, a polynomial basis formed from these singular vectors leads to a stable representation of multiplicative structure of the quotient ring $\mathbb{C}[\mathbf{x}]/I$. The solutions of F can be obtained by computing eigenvalues and eigenvectors of the multiplication matrices [1, 5, 25].

2.3 Algorithm for Computing Isolated Primary Component

For a given isolated solution of the ideal $I = (f_1, \dots, f_t)$, suppose Q is the isolated primary component whose associate prime $P = (x_1 - \hat{x}_1, \dots, x_s - \hat{x}_s)$, we apply SNEPSolver to compute the index ρ , such that $Q = (I, P^\rho)$ and the multiplication structure of the quotient ring $\mathbb{C}[\mathbf{x}]/Q$.

Algorithm 2. *IsolatedPrimaryComponent*

Input: An isolated multiple solution $\hat{\mathbf{x}}$ of an ideal $I = (f_1, \dots, f_t)$, a tolerance τ .

Output: The multiplicity μ , the index ρ , and multiplication matrices $M_{x_1}, \dots, M_{x_s}$ of the quotient ring $\mathbb{C}[\mathbf{x}]/Q$ where $Q = (I, P^\rho)$.

- Form the prime ideal $P = (x_1 - \hat{x}_1, \dots, x_s - \hat{x}_s)$.
- Compute $d_k = \dim(\mathbb{C}[\mathbf{x}]/(I, P^k))$ as described above by SNEPSolver for the given tolerance τ until $d_k = d_{k-1}$, then set $\rho = k - 1$, $\mu = d_\rho$ and $Q = (I, P^\rho)$.
- Compute the multiplication matrices $M_{x_1}, \dots, M_{x_s}$ of $\mathbb{C}[\mathbf{x}]/Q$ by SNEPSolver.

Symbolic methods based on the uniqueness of the reduced Gröbner basis are given in [12, 16] to determine the index of Q . However, when the multiple zero is only known with finite precision, their methods are subject to numerical stability problem.

Remark 2. The set made up of these computed multiplication matrices $\{M_{x_1}, \dots, M_{x_s}\}$ is called numerical local ring in [8] for a given root $\hat{\mathbf{x}}$.

Since the ideal (I, P^k) is generated by polynomials

$$F_k = \{f_1, \dots, f_t, (x_1 - \hat{x}_1)^{\alpha_1} \dots (x_s - \hat{x}_s)^{\alpha_s}, \sum_{i=1}^s \alpha_i = k\}.$$

Without loss of generality, suppose $d \leq k$, we prolong all polynomials f_i to have degree k . Since all monomials of degree $k + j$ appear in the prolonged system $F_k^{(j)}$, the symbol matrices of $F_k^{(j)}$ always have full rank, i.e., the symbols of $F_k^{(j)}$ are involutive. The dimension of the prolonged system, denoted by $\dim F_k^{(j)} = \dim \text{Nullspace}(M_k^{(j)})$, decreases strictly until it is stabilized, where $M_k^{(j)}$ is the coefficient matrix of $F_k^{(j)}$. So we have the following simple criterion of involution for polynomial system F_k .

Theorem 3. *The zero dimensional polynomial system F_k is involutive at order m if and only if*

$$\dim F_k^{(m)} = \dim F_k^{(m+1)}. \quad (5)$$

Example 1. [32] *Consider an ideal I generated by the polynomials*

$$\{f_1 = x_1^2 + x_2 - 3, f_2 = x_1 + 0.125x_2^2 - 1.5\}. \quad (6)$$

The system has (1, 2) as a 3-fold solution.

Form the maximal ideal $P = (x_1 - 1, x_2 - 2)$.

- $k = 2$, we consider the system

$$F_2 = \{f_1, f_2, (x_1 - 1)^2, (x_1 - 1)(x_2 - 2), (x_2 - 2)^2\}.$$

Since $\dim F_2 = 1$, $\dim F_2^{(1)} = \dim F_2^{(2)} = 2$, we have $\dim(\mathbb{C}[\mathbf{x}]/(I, P^2)) = 2$.

Similarly, we have

- $k = 3$, $\dim F_3 = 1$, $\dim F_3^{(1)} = \dim F_3^{(2)} = 3$, we have $\dim(\mathbb{C}[\mathbf{x}]/(I, P^3)) = 3$.
- $k = 4$, $\dim F_4 = 1$, $\dim F_4^{(1)} = \dim F_4^{(2)} = 3$, we have $\dim(\mathbb{C}[\mathbf{x}]/(I, P^4)) = 3$.

Therefore, the index and multiplicity of the root $(1, 2)$ are: $\rho = 3$, $\mu = 3$. The multiplication matrices with respect to the normal set $\{x_1, x_2, 1\}$ are:

$$M_{x_1} = \begin{bmatrix} 0 & -1 & 3 \\ 6 & 3 & -10 \\ 1 & 0 & 0 \end{bmatrix}, \quad M_{x_2} = \begin{bmatrix} 6 & 3 & -10 \\ -8 & 0 & 12 \\ 0 & 1 & 0 \end{bmatrix} \quad (7)$$

The triple eigenvalues of M_{x_1} and M_{x_2} are 1 and 2 respectively.

If the singular solution $\hat{\mathbf{x}}$ is only known approximately, then the polynomial system F_k has a cluster of solutions. The Schur factorization of multiplication matrices M_{x_i} consists of only one block. As shown in [5], the average of the cluster eigenvalues of M_{x_i} computed by $\text{Trace}(M_{x_i})/\mu$ gives a refined value for $\hat{x}_i$. We can apply the procedure again for the refined singular solution and obtain singular solution with higher accuracy.

Example 1 (continued) Suppose we are given an approximate singular solution:

$$\hat{\mathbf{x}} = (1 + 2.5428 \times 10^{-4} + 2.4352 \times 10^{-4} i, \\ 2 + 8.4071 \times 10^{-4} + 3.6129 \times 10^{-4} i).$$

We choose a tolerance $\tau = 10^{-4}$ and apply Algorithm 2 to $\hat{\mathbf{x}}$ and the polynomial system (6). The dimensions computed for the given tolerance are the same as shown above. Therefore, we get the same index and multiplicity for the approximate singular solution with respect to the given tolerance. The refined root computed from the multiplication matrices is:

$$(1 + 9.5829 \times 10^{-8} - 1.2762 \times 10^{-7} i, \\ 2 - 2.6679 \times 10^{-6} + 3.5569 \times 10^{-7} i).$$

Then use this refined solution as an initial one and set $\tau = 10^{-6}$, run Algorithm 2 again, we obtain:

$$(1 - 1.0000 \times 10^{-15} + 2.5854 \times 10^{-14} i, 2 + 8.4457 \times 10^{-14}).$$

3. MODIFIED SNEPSOLVER FOR COMPUTING DIFFERENTIAL OPERATORS

To apprehend the structure of the dual space of an ideal at a multiple zero further, we calculate a basis for the dual space.

Let $D(\alpha) = D(\alpha_1, \dots, \alpha_s) : \mathbb{C}[\mathbf{x}] \rightarrow \mathbb{C}[\mathbf{x}]$ denote the differential operator defined by:

$$D(\alpha_1, \dots, \alpha_s) = \frac{1}{\alpha_1! \dots \alpha_s!} \partial x_1^{\alpha_1} \dots \partial x_s^{\alpha_s},$$

for non-negative integer array $\alpha = [\alpha_1, \dots, \alpha_s]$. We write $\mathfrak{D} = \{D(\alpha), |\alpha| \geq 0\}$ and denote by $\text{Span}_{\mathbb{C}}(\mathfrak{D})$ the $\mathbb{C}$ -vector space generated by $\mathfrak{D}$ and introduce a morphism on $\mathfrak{D}$ that acts as “integral”:

$$\sigma_{x_j}(D(\alpha)) = \begin{cases} D(\alpha_1, \dots, \alpha_j - 1, \dots, \alpha_s), & \text{if } \alpha_j > 0, \\ 0, & \text{otherwise.} \end{cases}$$

Definition 5. A subspace L of $\text{Span}_{\mathbb{C}}(\mathfrak{D})$ is said to be closed if

$$\sigma_{x_j}(L) \subseteq L, \quad j = 1, \dots, s.$$

Definition 6. Given a zero $\hat{\mathbf{x}} = (\hat{x}_1, \dots, \hat{x}_s)$ of an ideal I , we define the subspace of differential operators associated to I and $\hat{\mathbf{x}}$ as

$$\Delta_{\hat{\mathbf{x}}} := \{L \in \text{Span}_{\mathbb{C}}(\mathfrak{D}) \mid L(f)|_{\mathbf{x}=\hat{\mathbf{x}}} = 0, \forall f \in I\}. \quad (8)$$

Theorem 4. [7] Let M be the maximal ideal $(x_1, \dots, x_s)$ of $\mathbb{C}[\mathbf{x}]$. There is a bijective correspondence between M -primary ideals of $\mathbb{C}[\mathbf{x}]$ and closed subspaces of $\text{Span}_{\mathbb{C}}(\mathfrak{D})$:

$$\begin{array}{c} \{M\text{-primary ideals in } \mathbb{C}[\mathbf{x}]\} \\ \updownarrow \\ \{\text{closed subspaces of } \text{Span}_{\mathbb{C}}(\mathfrak{D})\}. \end{array}$$

Moreover, for a zero dimensional M -primary ideal of $\mathbb{C}[\mathbf{x}]$ whose multiplicity is μ , we have that $\dim_{\mathbb{C}}(\Delta_{\hat{\mathbf{x}}}) = \mu$.

3.1 Algorithm for Computing Differential Operators

The following algorithm computes the differential operators from the output of Algorithm 2. Moreover, the differential operators evaluated at the multiple zero are functionals which constitute a set of bases for the dual space of the ideal at the multiple zero.

Algorithm 3. *DifferentialOperatorsI*

Input: Multiplication matrices $M_{x_1}, \dots, M_{x_s}$, multiple zero $\hat{\mathbf{x}}$ and index ρ .

Output: $L = \{L_1, \dots, L_{\mu}\}$, a basis for the space $\Delta_{\hat{\mathbf{x}}}$.

- Write the Taylor expansion at $\hat{\mathbf{x}}$ of a polynomial $h \in \mathbb{C}[\mathbf{x}]$ up to degree $\rho - 1$ with coefficients $c_{\alpha} \in \mathbb{C}$:

$$T_{\rho-1}(h) = \sum_{\alpha \in \mathbb{N}^s, |\alpha| < \rho} c_{\alpha} (x_1 - \hat{x}_1)^{\alpha_1} \dots (x_s - \hat{x}_s)^{\alpha_s}.$$

- Compute the normal form of h from the multiplication matrices $M_{x_1}, \dots, M_{x_s}$, and expand it at $\hat{\mathbf{x}}$,

$$\text{NF}(h(x)) = \sum_{\beta} d_{\beta} (\mathbf{x} - \hat{\mathbf{x}})^{\beta}.$$

- Find scalars $a_{\alpha\beta} \in \mathbb{C}$ such that $d_{\beta} = \sum_{\alpha} a_{\alpha\beta} c_{\alpha}$. For each β such that $d_{\beta} \neq 0$, return the operator

$$L_{\beta} = \sum_{\alpha} a_{\alpha\beta} \frac{1}{\alpha_1! \dots \alpha_s!} \partial x_1^{\alpha_1} \dots \partial x_s^{\alpha_s} = \sum_{\alpha} a_{\alpha\beta} D(\alpha).$$

An alternative procedure for computing these differential operators based on Gröbner basis computation is given in [7]. Our algorithm can be applied to polynomial system with floating point coefficients since it computes the normal form stably from the multiplication matrices computed by

SNEPSolver. Furthermore, in [7], the degree of polynomial $h(\mathbf{x})$ is bounded by the multiplicity μ which is larger or equal to our degree bound ρ according to Corollary 1.

Example 1 (continued) We compute the differential operators at the exact solution (1, 2):

- Write the Taylor expansion of a polynomial at (1, 2) up to degree $\rho - 1 = 2$,

$$h(\mathbf{x}) = c_{0,0} + c_{1,0}(x_1 - 1) + c_{0,1}(x_2 - 2) + c_{2,0}(x_1 - 1)^2 + c_{1,1}(x_1 - 1)(x_2 - 2) + c_{0,2}(x_2 - 2)^2.$$

- From the multiplication matrices (7), we obtain the normal form of h by replacing x_1^2, x_1x_2, x_2^2 with

$$x_1^2 = -x_2 + 3, x_1x_2 = 6x_1 + 3x_2 - 10, x_2^2 = -8x_1 + 12.$$

The differential operators are:

$$\begin{cases} L_1 &= D(0, 0), \\ L_2 &= D(0, 1) - D(2, 0) + 2D(1, 1) - 4D(0, 2), \\ L_3 &= D(1, 0) - 2D(2, 0) + 4D(1, 1) - 8D(0, 2). \end{cases}$$

Corollary 2. Applying the differential operators output from Algorithm 3 to the polynomials in F , we obtain a new polynomial system

$$\{L_j(f_i) \mid L_j \in L, f_i \in F, 1 \leq j \leq \mu, 1 \leq i \leq t\}, \quad (9)$$

and $\hat{\mathbf{x}}$ becomes its isolated simple solution.

Proof: If $\hat{\mathbf{x}}$ is a singular solution of system (9), then there exists a non-trivial differential operator L_τ , its differential order $|\tau| \geq 1$. Take $L_\beta \in L$ that has the highest differential order in L . Then $L_\tau \circ L_\beta$ is a differential operator for F at $\hat{\mathbf{x}}$ which is not included in L . It is a contradiction. $\square$

Suppose the multiple solution $\hat{\mathbf{x}}$ is only known with limited accuracy. Deflation method demonstrated in [19, 20] adds new equations and variables to F to have $\hat{\mathbf{x}}$ become a simple solution, then use Newton iteration to refine it. The new polynomial system (9) also has $\hat{\mathbf{x}}$ as its simple root, however, it is not clear whether we can use the attained information to refine $\hat{\mathbf{x}}$.

3.2 Specialized SNEPSolver

We have applied Algorithms 2 and 3 to a set of examples shown in the Table 1. For some examples, the systems become too large after we add all monomials of high degrees. In this subsection, we propose a modified SNEPSolver. The matrices we used to verify the involutivity are much smaller than the ones used by SNEPSolver.

Suppose $\hat{\mathbf{x}} = (\hat{x}_1, \dots, \hat{x}_s)$ is an isolated zero of a set of multivariate polynomials $F = \{f_1, \dots, f_t\}$. Let $P = (x_1 - \hat{x}_1, \dots, x_s - \hat{x}_s)$ and I be an ideal having P -primary isolated component. Let

$$T_k(F) = \{T_k(f_1), \dots, T_k(f_t)\},$$

where $T_k(f_i) = \sum_{|\alpha| < k} f_{i,\alpha}(\mathbf{x} - \hat{\mathbf{x}})^\alpha$ denotes truncated Taylor series expansions of the polynomial f_i at $\hat{\mathbf{x}}$ to order k . The zero can be moved to the origin by changing of variables. For simplicity, we suppose $\hat{\mathbf{x}}$ is the origin.

The ideal (I, P^k) is generated by the polynomials $F_k = T_k(F) \cup P^k$. The lower submatrices of the symbol matrices of F_k and prolonged system $F_k^{(j)}$ are identity matrices corresponding to monomials in P^k and P^{k+j} . Therefore the

symbol matrices are of full column rank, the dimensions and null spaces of coefficient matrices of the systems F_k and $F_k^{(j)}$ can be computed from the coefficient matrices generated by the truncated systems $T_k(F)$ and $T_k(F^{(j)})$. So we can work with the coefficient matrices of truncated system $T_k(F)$ and truncated prolonged systems $T_k(F^{(j)})$ instead of the coefficient matrices of $F \cup P^k$ and their prolongations. For simplicity, we still use $M_k^{(j)}$ to denote the coefficient matrices of the system $T_k(F^{(j)})$. Let $d_k^{(j)} = \dim \text{Nullspace}(M_k^{(j)})$.

Since all polynomials are truncated by degree k , the coefficient matrix $M_k^{(j)}$ has only $\binom{k+s-1}{s}$ columns. Furthermore, the number of prolongations m has an upper bound: $m \leq \max(1, k - \min(\text{ldeg}(f_1), \dots, \text{ldeg}(f_t)))$, where $\text{ldeg}(f)$ denotes the lowest degree of f .

Example 2. [19] The following polynomial system has 15 regular solutions and three 4-fold multiple solutions:

$$\{f_1 = x_1^3 + x_2^2 + x_3^2 - 1, f_2 = x_1^2 + x_2^3 + x_3^2 - 1, f_3 = x_1^2 + x_2^2 + x_3^3 - 1\}$$

We pick the multiple root $\hat{\mathbf{x}} = (1, 0, 0)$. By changing of variables, we get a new system $\{g_1 = x_1^3 + 3x_1^2 + 3x_1 + x_2^2 + x_3^2, g_2 = x_1^2 + 2x_1 + x_2^3 + x_3^2, g_3 = x_1^2 + 2x_1 + x_2^2 + x_3^3\}$ has the 4-fold multiple solution $\hat{\mathbf{x}} = (0, 0, 0)$. Let $P = (x_1, x_2, x_3)$.

- $k = 2$. We have:

$$[T_2(g_1), T_2(g_2), T_2(g_3)]^T = M_2^{(0)} \cdot [x_1, x_2, x_3, 1]^T,$$

where

$$M_2^{(0)} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 \end{bmatrix}$$

and $d_2^{(0)} = 3$. The prolonged matrix $M_2^{(1)}$ is obtained by adding zero elements to $M_2^{(0)}$. Hence $d_2^{(1)} = 3$ and

$$d_2 = \dim(\mathbb{C}[\mathbf{x}]/(I, P^2)) = 3.$$

- $k = 3$. We have:

$$[T_3(g_1), T_3(g_2), T_3(g_3)]^T = M_3^{(0)} \cdot [x_1^2, \dots, x_3, 1]^T,$$

where

$$M_3^{(0)} = \begin{bmatrix} 3 & 0 & 0 & 1 & 0 & 1 & 3 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 \end{bmatrix}$$

and $d_3^{(0)} = 7$. After the first prolongation, we have:

$$[T_3(x_1g_1), T_3(x_1g_2), T_3(x_1g_3), \dots, T_3(g_3)]^T = M_3^{(1)} \cdot [x_1^2, x_1x_2, x_1x_3, x_2^2, x_2x_3, x_3^2, x_1, x_2, x_3, 1]^T,$$

where

$$M_3^{(1)} = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 & 1 & 3 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 \end{bmatrix}.$$

We have $d_3^{(1)} = 4$. The prolonged matrix $M_3^{(2)}$ is obtained by adding zeros to $M_3^{(1)}$, hence $d_3^{(2)} = 4$, and

$$d_3 = \dim(\mathbb{C}[\mathbf{x}]/(I, P^3)) = 4.$$

- $k = 4$. We compute $d_4^{(0)} = 17$, $d_4^{(1)} = 8$, and $d_4^{(2)} = d_4^{(3)} = 4$. Hence,

$$d_4 = \dim(\mathbb{C}[\mathbf{x}]/(I, P^4)) = 4.$$

Since $d_3 = d_4 = 4$, the multiplicity of $\hat{\mathbf{x}} = (0, 0, 0)$ is $\mu = 4$ and the index of the primary ideal is $\rho = 3$, and $Q = (I, P^3)$.

The null space of the matrix $M_3^{(1)}$ has dimension 4 and can be written as

$$N_3^{(1)} = [e_{10}, e_9, e_8, e_5],$$

where e_i is the i -th column of 10×10 identity matrix.

It is interesting to notice that the differential operators can be obtained by multiplying the differential operators of order less than 3 with the 4 null vectors in $N_3^{(1)}$:

$$\{D(0, 0, 0), D(0, 0, 1), D(0, 1, 0), D(0, 1, 1)\}.$$

It is not a coincidence that the differential operators can be obtained from null vectors of the coefficient matrix $M_3^{(1)}$.

Theorem 5. Let $Q = (I, P^\rho)$ be the isolated primary component of an ideal $I = (f_1, \dots, f_t)$ at the multiple solution $\hat{\mathbf{x}}$ and μ be the multiplicity of $\hat{\mathbf{x}}$. Suppose the system $F_\rho = T_\rho(F) \cup P^\rho$ is involutive at prolongation order m , the null space of the matrix $M_\rho^{(m)}$ is generated by vectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_\mu$. Let

$$\mathbf{L} = [D(\rho - 1, 0, \dots, 0), D(\rho - 2, 1, 0, \dots, 0), \dots, D(0, \dots, 0)]$$

denote the vector consists of all differential operator of order less than ρ . Then a basis for the space $\Delta_{\hat{\mathbf{x}}}$ can be computed as

$$L_j = \mathbf{L} \cdot \mathbf{v}_j, \quad \text{for } 1 \leq j \leq \mu.$$

Proof: Since the system F_ρ is involutive at prolongation order m , for any polynomial $f \in I$, the coefficient vector $\mathbf{f}$ of the truncated polynomial $T_\rho(f)$ can be expressed as $\mathbf{f} = \mathbf{c} \cdot M_\rho^{(m)}$, where $\mathbf{c}$ is a complex row vector. For $1 \leq j \leq \mu$, we have

$$L_j(f) |_{\mathbf{x}=\hat{\mathbf{x}}} = L_j(T_k(f)) |_{\mathbf{x}=\hat{\mathbf{x}}} = \mathbf{c} \cdot M_\rho^{(m)} \cdot \mathbf{v}_j = 0. \quad \square$$

Algorithm 4. *DifferentialOperatorsII*

Input: An isolated multiple solution $\hat{\mathbf{x}} = (\hat{x}_1, \dots, \hat{x}_s)$ of an ideal $I = (f_1, \dots, f_t)$ and a tolerance τ .

Output: The multiplicity μ , the index ρ of the primary component $Q = (I, P^\rho)$ and a set of differential operators $L = \{L_1, \dots, L_\mu\}$.

- Form the coefficient matrix $M_k^{(0)}$ by computing the truncated multivariate Taylor series expansions of $f_1, \dots, f_t$ at $\hat{\mathbf{x}}$ to order k . The prolonged matrix $M_k^{(j)}$ is computed by shifting $M_k^{(0)}$ accordingly.
- Compute $d_k^{(j)} = \dim \text{Nullspace}(M_k^{(j)})$ for the given τ . The prolongation is stopped until $d_k^{(m)} = d_k^{(m+1)} = d_k$.
- If $d_k = d_{k-1}$, then set $\rho = k - 1$ and $\mu = d_\rho$.

- Compute the null vectors of $M_\rho^{(m)}$ denoted by $\mathbf{v}_1, \dots, \mathbf{v}_\mu$. The differential operators are computed as $L_j = \mathbf{L} \cdot \mathbf{v}_j$, for j from 1 to μ .

Remark 3. If the zero is not at the origin, we can also compute the matrix $M_k^{(0)}$ by changing of variables $y_i = x_i - \hat{x}_i$ for $1 \leq i \leq s$ and compute the coefficients of polynomials $f_1(y_1 + \hat{x}_1, \dots, y_s + \hat{x}_s), \dots, f_t(y_1 + \hat{x}_1, \dots, y_s + \hat{x}_s)$ with respect to the variables $y_1, \dots, y_s$. However, we only need the coefficients of $y_1^{\alpha_1} \dots y_s^{\alpha_s}$ with total degree less than k . Hence it is more efficient and numerically stable to compute the Taylor expansions at $\hat{\mathbf{x}}$ to order k .

There is an impressive paper written by Dayton and Zeng [9]. They compute the differential operators from the dual space. We obtain the differential operators from studying the primal space directly.

3.3 Algorithm for Refining Approximate Singular Solution

Suppose we are only given an approximate root

$$\hat{\mathbf{x}} = \hat{\mathbf{x}}_{\text{exact}} + \hat{\mathbf{x}}_{\text{error}},$$

where $\hat{\mathbf{x}}_{\text{error}}$ denotes the error in the solution and $\hat{\mathbf{x}}_{\text{exact}}$ denotes the exact solution of the polynomial system $F = \{f_1, \dots, f_t\}$ with multiplicity μ and index ρ . The output of Algorithm 4 is a set of differential operators satisfying approximately,

$$L_j(f_i) |_{\mathbf{x}=\hat{\mathbf{x}}} \approx 0, \quad 1 \leq j \leq \mu, \quad 1 \leq i \leq t.$$

According to Remark 3, the matrix $M_k^{(0)}$ in Algorithm 4 is the coefficient matrix of truncated polynomials $T_k(G) = \{T_k(g_1), \dots, T_k(g_t)\}$ where

$$g_i = f_i(y_1 + \hat{x}_{1,\text{exact}} + \hat{x}_{1,\text{error}}, \dots, y_s + \hat{x}_{s,\text{exact}} + \hat{x}_{s,\text{error}})$$

obtained by changing of variables $y_i = x_i - \hat{x}_i$, for $1 \leq i \leq s$. The output of Algorithm 4 can be regarded equivalently as the differential operators of system $G = \{g_1, \dots, g_t\}$ at its approximate solution $\hat{\mathbf{y}} = 0$,

$$L_j(g_i) |_{\mathbf{y}=\hat{\mathbf{y}}} \approx 0, \quad 1 \leq j \leq \mu, \quad 1 \leq i \leq t.$$

Suppose Algorithm 4 computes the multiplicity μ and index ρ correctly, and the system $G_{\rho+1} = T_{\rho+1}(G) \cup P^{\rho+1}$, $P = (y_1, \dots, y_s)$ is involutive at prolongation order m for a given tolerance. The matrix $M_{\rho+1}^{(m)}$ corresponds to the coefficient matrix of the prolonged system $T_{\rho+1}(G^{(m)})$ by truncating the monomials $\mathbf{y}^\alpha$ for $\alpha \geq \rho + 1$. Similar to Algorithm 1, we can use the null vectors of the matrix $M_{\rho+1}^{(m)}$ to form the multiplication matrices and compute a solution $\hat{\mathbf{y}}$ which is usually a good approximation for $-\hat{\mathbf{x}}_{\text{error}}$. This is mainly due to the fact that

$$\hat{\mathbf{y}} = -\hat{\mathbf{x}}_{\text{error}} = (-\hat{x}_{1,\text{error}}, \dots, -\hat{x}_{s,\text{error}})$$

is an exact solution of the system G with the multiplicity μ and index ρ .

Since all computations are done modular the ideal $P^{\rho+1} = (y_1, \dots, y_s)^{\rho+1}$. If the given approximate solution $\hat{\mathbf{x}}$ is not far away from the exact solution $\hat{\mathbf{x}}_{\text{exact}}$, then we have $\hat{\mathbf{y}} \approx 0$, the terms $\hat{\mathbf{y}}^\alpha$ for $\alpha > \rho$ will be much closer to a zero vector. We are performing a generalized Newton iteration modular $P^{\rho+1}$. The profound theory still need to be investigated in future. We refer to [18] for interesting discussion on quadratic Newton iteration for systems with multiplicity.

Algorithm 5. *MultipleRootRefiner(MRR for short)*

Input: An isolated approximate singular solution $\hat{\mathbf{x}}$ of an ideal $I = (f_1, \dots, f_t)$ and a tolerance τ .

Output: Refined solution $\hat{\mathbf{x}}$, the multiplicity μ and index ρ of the primary component $Q = (I, P^\rho)$ and differential operators $L = \{L_1, \dots, L_\mu\}$ for the refined solution.

- For given approximate root $\hat{\mathbf{x}}$ and tolerance τ , applying Algorithm 4 to estimate the multiplicity μ and index ρ .
- Suppose the truncated system $G_{\rho+1}$ is involutive at prolongation order m , then form the multiplication matrices $M_{x_1}, \dots, M_{x_s}$ from the null vectors of the matrix $M_{\rho+1}^{(m)}$. An approximate solution $\hat{\mathbf{y}}$ is obtained by averaging the trace of each multiplication matrix.
- Set $\hat{\mathbf{x}} = \hat{\mathbf{x}} + \hat{\mathbf{y}}$ and run the first two steps for the refined solution. The tolerance is decreased according to the solution $\hat{\mathbf{y}}$.
- If $\hat{\mathbf{y}}$ converges to the origin, then we get the refined solution $\hat{\mathbf{x}}$ with high accuracy. We apply Algorithm 4 to compute the differential operators with respect to the refined solution. Otherwise, we decrease the tolerance and run the above steps again.

Example 2 (continued) Suppose we are given an approximate solution $\hat{\mathbf{x}} = (1.001, -0.002, -0.001 i)$. By changing of variables, we have a perturbed system

$$G = \{f_j(y_1 + 1.001, y_2 - 0.002, y_3 - 0.001 i), j = 1, 2, 3\}.$$

Applying Algorithm 4 to estimate multiplicity and index for a given tolerance $\tau = 10^{-2}$.

- The singular values of $M_3^{(1)}$ are: $\{6.2680, \dots, 0.2167, 4.6071 \times 10^{-3}, 1.7236 \times 10^{-5}, 5.5960 \times 10^{-7}, 3.5123 \times 10^{-9}\}$ and $d_3^{(1)} = 4$.
- The singular values of $M_3^{(2)}$ are: $\{6.2680, \dots, 0.2168, 5.8533 \times 10^{-3}, 2.2785 \times 10^{-5}, 8.1321 \times 10^{-6}, 6.4922 \times 10^{-9}\}$ and $d_3^{(2)} = 4$.
- The singular values of $M_4^{(2)}$ are: $\{7.2589, \dots, 0.07891, 2.0256 \times 10^{-5}, 5.2991 \times 10^{-8}, 7.4807 \times 10^{-9}, 7.4618 \times 10^{-12}\}$ and $d_4^{(2)} = 4$.

So that for the given tolerance $\tau = 10^{-2}$, the multiplicity is $\mu = 4$ and the index is $\rho = 3$. The multiplication matrices are formed from the null space of $M_4^{(2)}$. The solution computed by averaging the eigenvalues of the multiplication matrices is

$$\begin{aligned} \hat{\mathbf{y}} = & (-0.0009994 - 7.5315 \times 10^{-10} i, \\ & 0.002001 + 2.8002 \times 10^{-8} i, \\ & -1.4949 \times 10^{-6} + 0.0010000 i). \end{aligned}$$

Adding $\hat{\mathbf{y}}$ to $\hat{\mathbf{x}}$, we obtain the refined solution $\hat{\mathbf{x}}$. We apply Algorithm 5 twice to the new singular solution $\hat{\mathbf{x}}$ for tolerance 10^{-5} and 10^{-8} respectively, and get the refined solution:

$$\begin{aligned} \hat{\mathbf{x}} = & (1 + 7.0405 \times 10^{-18} - 7.8146 \times 10^{-19} i, \\ & 1.0307 \times 10^{-16} - 1.9293 \times 10^{-17} i, \\ & 1.5694 \times 10^{-16} + 7.9336 \times 10^{-17} i). \end{aligned}$$

4. EXPERIMENTS

The following experiments are done for Digits := 14 in Maple 11 under Windows. The systems DZ1 and DZ2 are quoted from [9]. The system D2 [8] is positive dimensional, but we can compute its isolated zero dimensional primary component at the origin. The other examples are cited from the PHCPack demos <http://www.math.uic.edu/~jan/>. The second column lists the singular solutions $\hat{\mathbf{x}}$, where $Z_2 = (-.7071, .4082, .5774, .2500, -.1443, -.4082)$. We use ρ and μ to represent the index and the multiplicity respectively. The fifth column lists the increase in the number of correct digits from the initial guess to the refined solutions by SNEPSolver. The empty box denotes that SNEPSolver fails after running out of memory. The sixth columns shows the increase in the number of correct digits of the approximate solutions obtained by Algorithm MultipleRootRefiner.

System	Zero	ρ	μ	SNEPSolver	MRR
cmbs1	(0, 0, 0)	5	11	5 → 14	3 → 11 → 15
cmbs2	(0, 0, 0)	4	8	5 → 15	3 → 13 → 15
mt191	(0, 1, 0)	3	4	5 → 10 → 15	4 → 9 → 15
LVZ	(0, 0, -1)	7	18		5 → 10 → 14
KSS	(1, 1, 1, 1, 1, 1)	5	16		5 → 11 → 14
Caprasse	$(2, -i\sqrt{3}, 2, i\sqrt{3})$	3	4	5 → 14	4 → 12 → 15
DZ1	(0, 0, 0, 0)	11	131	5 → 14	5 → 14
DZ2	(0, 0, -1)	8	16		4 → 7 → 14
tangents1	Z_2	4	4		3 → 10 → 16
D2	(0, 0, 0)	5	5	5 → 10 → 15	5 → 10 → 15
Ojika1	(1, 2)	3	3	5 → 7 → 14	3 → 6 → 18
Ojika2	(0, 1, 0)	2	2	5 → 10 → 15	5 → 10 → 14
Ojika3	(0, 0, 1)	3	4	5 → 9 → 14	4 → 8 → 15
Ojika4	(0, 0, 1)	3	3	5 → 10 → 14	3 → 7 → 15

Table 1: Algorithm Performance

5. CONCLUSION

The multiplicity structure of a singular solution has been studied extensively in [2, 7, 8, 9, 14, 22, 23, 25, 26, 27]. Various methods have also been proposed for computing the singular solutions to high accuracy [5, 18, 19, 20, 32, 33]. In this paper, we describe algorithms based on the geometric involutive form to completely describe the multiplicity structure of an isolated singular solution.

If the polynomial system and the singular solutions are known exactly, the tolerance is set to be zero. We compute the multiplicity, index and differential operators by exact linear algebra computation. If we are given an approximate isolated singular solution of an exact polynomial system, then we refine the singular solution to have high accuracy and obtain accurate multiplicity structure with respect to the refined solution. However, if the polynomials are only known with limited accuracy, the results we computed depend on a properly chosen tolerance. If there is no information about correct digits of the input data, we decide the tolerance by checking the biggest jump in the singular values of the coefficient matrix of the polynomial system F . It is interesting to investigate whether we can find a nearby polynomial system which has an isolated singular solution with given structure, such as multiplicity, index or differential operators.

Suppose the ideal $I = (f_1, \dots, f_t)$ is zero dimensional, then applying our method to each root, we can compute the primary decomposition of I . However, I need not to be zero dimensional. We only require that the primary component Q is isolated and zero dimensional. Furthermore, the algorithms we present can be generalized to compute an isolated primary component for a maximal ideal represented by polynomials, not only the ones generated by the singular solutions.

Acknowledgments

We greatly thank Greg Reid for fruitful discussions. Our thanks go also to Philippe Trébuchet for interesting discussions during the first Chinese-SALSA workshop.

6. REFERENCES

- [1] AUZINGER, W., AND STETTER, H. An elimination algorithm for the computation of all zeros of a system of multivariate polynomial equations. *Intern. Series in Numer. Math.* 86 (1988), 11–30.
- [2] BATES, D., PETERSON, C., AND SOMMESE, A. A numerical-symbolic algorithm for computing the multiplicity of a component of an algebraic set. *Journal of Complexity* 22 (2006), 475–489.
- [3] BAYER, D., AND STILLMAN, M. A criterion for detecting m -regularity. *Inventiones Mathematicae* 87, 1 (1987), 1–11.
- [4] BONASIA, J., LEMAIRE, F., REID, G., SCOTT, R., AND ZHI, L. Determination of approximate symmetries of differential equations. In *CRM Proceedings and Lecture Notes* (2004), pp. 233–249.
- [5] CORLESS, R., GIANNI, P., AND TRAGER, B. A reordered Schur factorization method for zero-dimensional polynomial systems with multiple roots. In *Proc. 1997 Internat. Symp. Symbolic Algebraic Comput. ISSAC'97* (New York, 1997), Küchlin, Ed., ACM Press, pp. 133–140.
- [6] COX, D., LITTLE, J., AND O'SHEA, D. *Ideals, Varieties, and Algorithms*. Springer-Verlag, New York, 1992. Undergraduate Texts in Mathematics.
- [7] DAMIANO, A., SABADINI, I., AND STRUPPA, D. Computational methods for the construction of a class of Noetherian operators. *Experiment. Math.* 16 (2007), 41–55.
- [8] DAYTON, B. Numerical local rings and local solutions of nonlinear systems. In *SNC'07 Proc. 2007 Internat. Workshop on Symbolic-Numeric Comput.* (New York, 2007), J. Verschelde and S. M. Watt, Eds., ACM Press, pp. 79–86.
- [9] DAYTON, B., AND ZENG, Z. Computing the multiplicity structure in solving polynomial systems. In Kauers [13], pp. 116–123.
- [10] FAUGÈRE, J. A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In *Proc. 2002 Internat. Symp. Symbolic Algebraic Comput. ISSAC'02* (New York, 2002), T. Mora, Ed., ACM Press, pp. 75–83.
- [11] GERDT, V., AND BLINKOV, Y. Involutive bases of polynomial ideals. *Mathematics and Computers in Simulation* 45 (1998), 519–541.
- [12] GIANNI, P., TRAGER, B., AND ZACHARIAS, G. Gröbner bases and primary decomposition of polynomial ideals. *J. Symbolic Comput.* 6 (1988), 149–167.
- [13] KAUSERS, M., Ed. *ISSAC '05 Proc. 2005 Internat. Symp. Symbolic Algebraic Comput.* (New York, 2005), ACM Press.
- [14] KOBAYASHI, H., SUZUKI, H., AND SAKAI, Y. Numerical calculation of the multiplicity of a solution to algebraic equations. *Math. Comp.* 67(221) (1998), 257–270.
- [15] KURANISHI, M. On E.Cartan's prolongation theorem of exterior differential systems. *Amer. J. Math.* 79 (1957), 1–47.
- [16] LAKSHMAN, Y. A single exponential bound on the complexity of computing Gröbner bases of zero dimensional ideals. *Effective Methods in Algebraic Geometry, Progress in Mathematics* (1994), 227–234.
- [17] LAZARD, D. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In *Proc. European Conf. Comput. Algebra. Lect. Notes in Comp. Sci.*, 162, Springer (1983), pp. 146–157.
- [18] LECERF, G. Quadratic Newton iteration for systems with multiplicity. *Foundations of Computational Mathematics* 2, 3 (2002), 247–293.
- [19] LEYKIN, A., VERSCHELDE, J., AND ZHAO, A. Newton's method with deflation for isolated singularities of polynomial systems. *Theoretical Computer Science* 359 (2006), 111–122.
- [20] LEYKIN, A., VERSCHELDE, J., AND ZHAO, A. Higher-order deflation for polynomial systems with isolated singular solutions. Manuscript, 19 pages, Jan 2007.
- [21] MALGRANGE, B. Cartan involutiveness = Mumford regularity. *Contemp. Math.* 331 (2003), 193–205.
- [22] MARINARI, M., MORA, T., AND MÖLLER, H. Gröbner duality and multiplicities in polynomial solving. In *Proc. 1995 Internat. Symp. Symbolic Algebraic Comput. (ISSAC'95)* (New York, 1995), A. Levelt, Ed., ACM Press, pp. 167–179.
- [23] MARINARI, M., MORA, T., AND MÖLLER, H. On multiplicities in polynomial system solving. *Trans. Amer. Math. Soc.* 348 (1996), 3283–3321.
- [24] MÖLLER, H., AND SAUER, T. H-bases for polynomial interpolation and system solving. *Advances Comput. Math.* 12 (2000), 23–35.
- [25] MÖLLER, H., AND STETTER, H. Multivariate polynomial equations with multiple zeros solved by matrix eigenproblems. *Numer. Math.* 70 (1995), 311–329.
- [26] MÖLLER, H., AND TENBERG, R. Multivariate polynomial system solving using intersections of eigenspaces. *J. Symbolic Comput.* 30 (2001), 1–19.
- [27] MOURRAIN, B. Isolated points, duality and residues. *J. of Pure and Applied Algebra* 117 & 118 (1996), 469–493.
- [28] MOURRAIN, B. A new criterion for normal form algorithms. In *AAECC* (Berlin, 1999), M. Fossorier, H. Imai, S. Lin, and A. Poli, Eds., vol. 1719, Springer, pp. 430–443.
- [29] MOURRAIN, B., AND TRÉBUCHET, P. Solving projective complete intersection faster. In *Proc. 2000 Internat. Symp. Symbolic Algebraic Comput. ISSAC'00* (New York, 2000), C. Traverso, Ed., ACM Press, pp. 430–443.
- [30] MOURRAIN, B., AND TRÉBUCHET, P. Algebraic methods for numerical solving. In *Proceedings of the 3rd International Workshop on Symbolic and Numeric Algorithms for Scientific Computing* (2002), pp. 42–47.
- [31] MOURRAIN, B., AND TRÉBUCHET, P. Generalized normal forms and polynomial system solving. In Kauers [13], pp. 253–260.
- [32] OJIKAWA, T. Modified deflation algorithm for the solution of singular problems. *J. Math. Anal. Appl.* 123 (1987), 199–221.
- [33] OJIKAWA, T., WATANABE, S., AND MITSUI, T. Deflation algorithm for the multiple roots of a system of nonlinear equations. *J. Math. Anal. Appl.* 96 (1983), 463–479.
- [34] POMMARET, J. *Systems of Partial Differential Equations and Lie Pseudogroups*. Gordon and Breach Science Publishers, 1978.
- [35] REID, G., SCOTT, R., WU, W., AND ZHI, L. Algebraic and geometric properties of nearby projectively involutive polynomial systems. Manuscript, 2005.
- [36] REID, G., TANG, J., AND ZHI, L. A complete symbolic-numeric linear method for camera pose determination. In *Proc. 2003 Internat. Symp. Symbolic Algebraic Comput. ISSAC'03* (New York, 2003), J. Sendra, Ed., ACM Press, pp. 215–223.
- [37] REID, G., AND ZHI, L. Solving polynomial systems via symbolic-numeric elimination method. *J. Symbolic Comput.* (2008). To appear.
- [38] SEILER, W. *Involution - The formal theory of differential equations and its applications in computer algebra and numerical analysis*. Habilitation thesis, Univ. of Mannheim, Germany, 2002.
- [39] STETTER, H. *Numerical Polynomial Algebra*. SIAM, Philadelphia, 2004.
- [40] TRÉBUCHET, P. *Vers une Résolution Stable et Rapide des Équations Algébriques*. Dissertation, Université Pierre et Marie Curie, France, 2002.
- [41] VAN DER WAERDEN, B. L. *Algebra*. Frederick Ungar Pub. Co., 1970.
- [42] WITTKOPF, A., AND REID, G. Fast differential elimination in C: The CDiffelim Environment. *Comp. Phys. Comm.* 139(2) (2001), 192–217.
- [43] ZHI, L., AND REID, G. Solving nonlinear polynomial system via symbolic-numeric elimination method. In *Proc. International conference on polynomial system solving* (2004), J. Faugère and F. Rouillier, Eds., pp. 50–53.

Exact Algorithms and Software in Optimization and Polyhedral Computation

Komei Fukuda
Institute for Operations Research and
Institute of Theoretical Computer Science
ETH Zurich, Zurich, Switzerland
fukuda@ifor.math.ethz.ch

ABSTRACT

In this tutorial, we present a new field of research on implementing exact algorithms in optimization and polyhedral computation in exact (arbitrary precision) arithmetic. Such algorithms including those to solve linear programming and convex hull problems have been implemented mostly with floating-point arithmetic. This new field of developing “exact software” in optimization and polyhedral computation is now at the second stage where new mathematical tools relying on optimization and polyhedral codes are being developed that have not been implemented before. This tutorial is not meant to cover all important developments but to look at some interesting facets.

Categories and Subject Descriptors

G.4 [Mathematics of Computing]: Mathematical Software—*Reliability and robustness*; I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms

General Terms

Algorithms, Reliability

Keywords

convex polyhedra, geometry, optimization, exact computation

1. INTRODUCTION

Most existing computer codes for solving optimization problems such as linear and quadratic programming work with floating-point arithmetic and thus do not guarantee the correctness of output. Those codes typically read input with imprecise numbers and output approximate solutions that are usually satisfactory for users who are not expecting mathematical correctness. Clearly these imprecise codes are not very useful for proving mathematical claims or for solving problems even approximately which have very large and small numbers in their input data. Another disadvantage of imprecise codes is that even if such codes might be correct almost always, say at the failure rate of 0.0001% for your problem instances, it might not be suitable for a computation that requires the solution of a large number of

optimization problems without any single failure. There are many such computations arising from computational geometry and algebra.

One natural way to implement exact algorithms mathematically correctly is to use exact (arbitrary precision) arithmetic such as rational exact arithmetic. Fortunately many efficient exact arithmetic libraries are now available [1, 2, 3, 4]. This tutorial is not meant to cover all important developments but to look at some interesting facets along this direction.

2. OPTIMIZATION

2.1 Linear Programming

Given a rational matrix $A \in \mathbb{Q}^{m \times d}$, two vectors $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^d$, a *linear programming problem* (LP) is to find a vector x maximizing $c^T x$ subject to $Ax \leq b$ and $x \geq 0$. The simplex method is a practically efficient method to solve LPs which is easily implementable with exact arithmetic. There are several exact implementations of the simplex method, e.g. [11, 6, 8, 15]. These implementations run with exact arithmetic and are extremely reliable. EXLP [15] is a standalone code and can exploit the sparsity of input matrix A . On the other hand, CDDLIB, LRSLIB and CGAL’s solver [11, 6, 8] are all callable libraries but do not take advantage of the sparsity.

2.2 Quadratic Programming

If one replaces the linear objective function $c^T x$ of the LP with a concave quadratic function, one obtains a convex quadratic programming problem (QP). Perhaps, [8] is a unique exact library to solve (dense) convex QPs.

3. POLYHEDRAL COMPUTATION

A (convex) *polyhedron* P in $\mathbb{R}^d$ is a set of form $P(A, b) := \{x \in \mathbb{R}^d : Ax \leq b\}$ for some matrix $A \in \mathbb{Q}^{m \times d}$ and some vector $b \in \mathbb{Q}^m$. The system $Ax \leq b$ is called an *H-representation* of P . A bounded polyhedron is called *polytope*. Every polyhedron has another representation, namely, the set of all extreme points (vertices) and extreme rays, known as a *V-representation*.

3.1 Representation Conversion

The representation conversion for a polyhedron is to find the second representation from a given (H or V)-representation. In fact, the two conversions are computationally equivalent. Two codes [11, 6] implementing two totally differ-

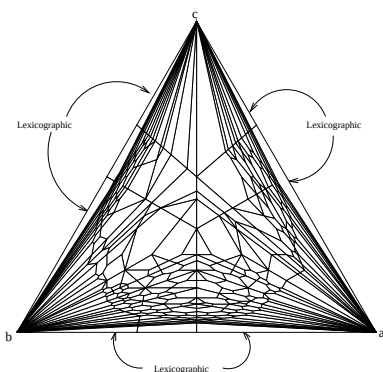


Figure 1: The Gröbner Fan of $\langle a^5 + b^3 + c^2 - 1, a^2 + b^2 + c - 1, a^6 + b^5 + c^3 - 1 \rangle$

ent algorithms have been used extensively, and for example, a polyhedral-manipulation system POLYMAKE [13] includes them as core engines. There are many other exact implementations available, see [10].

3.2 Minkowski Sums

The Minkowski sum $P + Q$ of two polytopes P and Q is a very useful operation in mathematics. An exact implementation MINKSUM [18] of the algorithm [9] relies on the exact LP solver in CDDLIB [11].

4. APPLICATIONS

4.1 Regular Triangulations

Given a finite set P of points in $\mathbb{R}^d$, the software TOPCOM [16] can generate all regular triangulations of P . Checking whether a given triangulation is regular or not can be formulated as an LP. TOPCOM calls the exact LP solver in CDDLIB [11] for this purpose.

4.2 Gröbner fans

An algorithm to compute the Gröbner fan of a polynomial ideal was recently proposed in [12], and an exact implementation is available [14]. Figure 1 is a graphical output of a Gröbner fan as a cut section of the positive orthant in $\mathbb{R}^3$ by a hyperplane. Like [18], the code calls the exact LP solver in CDDLIB [11] many times in order to remove redundant inequalities from a highly redundant linear inequality system.

4.3 Enumeration of Lattice Points

Given a polytope $P \in \mathbb{R}^d$, generating or counting all integer (lattice) points in P is a classical problem in mathematics. It is also closely related to solving integer programming which is to find a lattice point in P maximizing a linear objective function. There are a couple of implementations of Barvinok's algorithm available [17, 7]. Both rely on exact algebraic and polyhedral computation codes such as 4ti2 [5] and CDDLIB [11].

5. FUTURE PERSPECTIVE

The recent developments of exact implementations, typically in the form of callable libraries, of optimization and polyhedral computation algorithms have contributed to the

computing environment of mathematicians and scientists. In particular, we are seeing a totally new set of mathematical tools built on such libraries. It is expected that new exact tools and libraries will find users beyond the scientific community, because in application domains such as control theory and mechanical engineering, numerical instability caused by imprecision might result in meaningless computation.

6. REFERENCES

- [1] CGAL, Computational Geometry Algorithms Library. <http://www.cgal.org>.
- [2] The CORE library. <http://cs.nyu.edu/exact/>.
- [3] GMP, GNU's library for arbitrary precision arithmetic. <http://gmplib.org/>.
- [4] PARI/GP, a c-library for fast computation in number theory. <http://pari.math.u-bordeaux.fr/>.
- [5] 4ti2 team. 4ti2—a software package for algebraic, geometric and combinatorial problems on linear spaces. Available at www.4ti2.de.
- [6] D. Avis. *lrs Homepage*. <http://cgm.cs.mcgill.ca/~avis/C/lrs.html>.
- [7] J. de Loera, D. Haws, R. Hemmecke, P. Huggins, J. Tauzer, and R. Yoshida. *LattE*. University of California, Davis, 2005. <http://www.math.ucdavis.edu/~latte/>.
- [8] K. Fischer, B. Gärtner, S. Schönherr, and F. Wessendorp. CGAL's linear and quadratic programming solver, 2007. <http://www.cgal.org/Pkg/QPSolver>.
- [9] K. Fukuda. From the zonotope construction to the Minkowski addition of convex polytopes. *Journal of Symbolic Computation*, 38(4):1261–1272, 2004.
- [10] K. Fukuda. Polyhedral computation FAQ, 2004. <http://www.ifor.math.ethz.ch/~fukuda/fukuda.html>.
- [11] K. Fukuda. cdd, cddplus and cddlib homepage, 2007. http://www.ifor.math.ethz.ch/~fukuda/cdd_home/.
- [12] K. Fukuda, A. Jensen, and R. Thomas. Computing Gröbner fans. *Mathematics of Computation*, 76:2189–2212, 2007.
- [13] E. Gawrilow and M. Joswig. polymake: a framework for analyzing convex polytopes. In G. Kalai and G. M. Ziegler, editors, *Polytopes — Combinatorics and Computation*, pages 43–74. Birkhäuser, 2000. <http://www.math.tu-berlin.de/diskregeom/>.
- [14] A. Jensen. Gfan version 0.3: A user's manual, 2007. <http://www.math.tu-berlin.de/~jensen/software/gfan/gfan.html>.
- [15] M. Kiyomi. EXLP, an exact linear programming solver. <http://members.jcom.home.ne.jp/masashi777/exlp.html>.
- [16] J. Rambau. TOPCOM, a package for computing Triangulations of point configurations and oriented matroids. <http://www.rambau.wm.uni-bayreuth.de/>.
- [17] S. Verdoolaege. Barvinok: a library for counting the number of integer points in parametric and non-parametric polytopes, version 0.26, 2008. <http://www.kotnet.org/~skimo/barvinok/>.
- [18] C. Weibel. *Minksum version 1.4*. Mathematics Institute, EPF Lausanne, 2006. <http://roso.epfl.ch/cw/poly/public.php>.

Algorithmic Invariant Theory

Gregor Kemper
Technische Universität München
Zentrum Mathematik - M11
Boltzmannstr. 3, 85748 Garching
Germany
kemper@ma.tum.de

Categories and Subject Descriptors

I.1 [Symbolic and Algebraic Manipulation]: Algorithms

General Terms

Algorithms

1. EXTENDED ABSTRACT

Invariant theory can be put in a very general context: If “ $\sim$ ” is an equivalence relation on a set X , then an **invariant** is a function on X which is constant on every equivalence class. So invariants serve to parametrize equivalence classes. The goals of invariant theory are to find all invariants that meet some further restrictions (such as continuity or polynomiality), and to study to which extent these invariants separate equivalence classes. For example, the determinant of a square matrix is an invariant w.r.t. the equivalence relation given by similarity.

In the classical situation of invariant theory, the equivalence classes are given by the orbits of a group action. In fact, one considers the following setting: G is a linear algebraic group over an algebraically closed field K , and V is a finite-dimensional K -vector space with a linear G -action, given by a morphism $G \times V \rightarrow V$. In other words, we assume that the action can be described by polynomial functions. A natural extension is to substitute V by an affine K -variety X , which is then called a **G -variety**. The **invariant ring**

$$K[V]^G := \{f \in K[V] \mid f \circ \sigma = f \text{ for all } \sigma \in G\}$$

consists of all polynomials $f: V \rightarrow K$ which are constant on every G -orbit. Observe that the above example of the determinant of a square matrix falls into this situation, with $G = \text{GL}_n(K)$ acting on $V = K^{n \times n}$ by conjugation. The following questions are central in invariant theory:

- *Hilbert’s 14th Problem*: Is $K[V]^G$ finitely generated as a K -algebra?
- If so, find generators (algorithmically).

- Which G -orbits can be separated by invariants, i.e., for which $x, y \in V$ does there exist $f \in K[V]^G$ with $f(x) \neq f(y)$?

Invariant theory has gone a long way towards answering Hilbert’s 14th Problem. In fact, it is known by results of Hilbert, Nagata, Haboush and Popov that $K[X]^G$ is finitely generated for all G -varieties X if and only if G is reductive. However, it happens quite often that for a given non-reductive group G and a given linear representation V , the invariant ring $K[V]^G$ is finitely generated in spite of this result. So the problem to classify all pairs (G, V) or (G, X) such that the invariant ring is finitely generated is still open. Note that the class of reductive groups include all finite groups and all classical groups.

The algorithmic side of the problem has been trailing behind the theoretical progress, but has by now almost completely caught up. In fact, we do have an algorithm for computing generators of $K[X]^G$ in the case that G is reductive and X is a G -variety. Major stepping stones towards this algorithm were provided by Derksen’s algorithm, which solves the problem for linearly reductive groups, and by an algorithm for computing separating invariants, a topic that we will address shortly.

If G acts linearly on V , then the invariant ring $K[V]^G$ is a graded algebra. If it is finitely generated, there exists a number, written as $\beta(K[V]^G)$, which is the smallest d such that $K[V]^G$ can be generated by homogeneous invariants of degree $\leq d$. This number is interesting since any upper bound for $\beta(K[V]^G)$ leads to an algorithm for computing generators. If G is finite of order not a multiple of the characteristic $\text{char}(K)$ of K , then the famous Noether bound (which in this generality was not proved until 2000) tells us that

$$\beta(K[V]^G) \leq |G|.$$

This bound becomes (catastrophically) wrong in the *modular case*, i.e., when $\text{char}(K)$ divides $|G|$.

A fairly recent trend in invariant theory has been the study of *separating invariants*. By definition, a subset $S \subseteq K[V]^G$ is called **separating** if for all pairs of points $x, y \in V$ the existence of $f \in K[V]^G$ with $f(x) \neq f(y)$ implies the existence of $f \in S$ with $f(x) \neq f(y)$. In other words, S has the same capabilities of separating orbits as $K[V]^G$. Every generating subset of $K[V]^G$ is separating, so the concept of a separating subset is a *weakening* of the concept of a generating subset. But separating subsets have better properties than separating ones. For example,

- there always exists a finite separating subset (even if $K[V]^G$ is not finitely generated), and

- for G finite, there exists a separating set of homogeneous invariants of degree $\leq |G|$, even in the modular case. So Noether's bound always holds for separating invariants.

Separating invariants are also useful for algorithmic purposes: The author has found an algorithm for computing separating invariants in the case that G is reductive. Since we also have algorithms for extending a separating subset into a generating subset, we obtain the above-mentioned algorithm for computing $K[X]^G$ for G reductive.

The following problems are still open:

- Find an algorithm for computing a separating subset for G non-reductive.
- Find a test for finite generation of $K[V]^G$.
- Compute $K[V]^G$ as the ring of regular functions of a quasi-affine variety U :

$$K[V]^G = K[U].$$

(By a result of Nagata, such a U exists even if $K[V]^G$ is not finitely generated.)

- Implement the known algorithms (i.e., beyond Derksen's algorithm and the finite groups case).

Algebraic Algorithms and Coding Theory

Madhu Sudan
MIT Computer Science and Artificial Intelligence Laboratory
The Stata Center, Building 32
Cambridge, MA 02139
USA
madhu@mit.edu

Categories and Subject Descriptors

I.1 [Symbolic and Algebraic Manipulation]: Algorithms

General Terms

Algorithms

ABSTRACT

The associated talk surveys some recent developments in algorithmic coding theory that answer some fundamental questions with algebraic techniques.

1. INTRODUCTION

The theory of error-correcting codes has long seen codes with remarkable combinatorial performance emerging from the study of algebraic functions over finite fields (e.g., Reed-Solomon code [7], BCH codes [2, 5], algebraic-geometry codes [3, 9]). They have also provided the inspiration for, and benefited from, the development of algebraic algorithms (e.g., Berlekamp's algorithm for factoring univariate polynomials [1], Groebner basis based algorithms for decoding algebraic-geometry codes). This phenomenon has repeated itself in recent years with a resurgence of algorithms for problems in error-correction (list-decoding of Reed-Solomon codes [8, 4] and the recent results of Parvaresh-Vardy [6] and Guruswami-RudraGuRu), which have in turn inspired new (fast) algorithms for polynomial factorization (due to Chris Umans [10]).

In this survey we will introduce the basic algebraic codes and their decoding algorithms. The hope is to eventually describe the Guruswami-Rudra result which shows how to construct codes over a large alphabet of rate $1 - p - o(1)$ that correct (list-decode) p fraction of adversarially injected errors in polynomial time. Prior to this result no explicit construction of such codes (capable of correcting so many errors with even exponential time decoding algorithms) was known!

2. ACKNOWLEDGEMENT

This work is supported in part by NSF Award CCR 0514915.

3. REFERENCES

- [1] Elwyn Berlekamp. Factoring polynomials over large finite fields. *Mathematics of Computations*, 24:713–735, 1970.
- [2] R. C. Bose and D. K. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and Control*, 3:68–79, 1960.
- [3] V. D. Goppa. Codes associated with divisors. *Problems of Information Transmission*, 13(1):22–26, 1977.
- [4] Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Transactions on Information Theory*, 45:1757–1767, 1999.
- [5] A. Hocquenghem. Codes correcteurs d'erreurs. *Chiffres (Paris)*, 2:147–156, 1959.
- [6] Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In *FOCS*, pages 285–294. IEEE Computer Society, 2005.
- [7] Irving S. Reed and Gustav Solomon. Polynomial codes over certain finite fields. *J. SIAM*, 8:300–304, 1960.
- [8] Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.
- [9] Michael A. Tsfasman, Serge G. Vlăduț, and Thomas Zink. Modular curves, Shimura curves, and codes better than the Varshamov-Gilbert bound. *Math. Nachrichten*, 109:21–28, 1982.
- [10] Chris Umans. Fast polynomial factorization and modular composition in small characteristic. In *Proceedings of ACM STOC '08*, page to appear, 2008.

Author Index

Abramov, Sergei A.	1	Huang, Lei	181	Poteaux, Adrien	239
Achatz, Melanie	215	Janovitz-Freireich, Itnuit	125	Regensburger, Georg	261
Anai, Hirokazu	147	Jeffrey, David J.	189	Renault, Guénaél	247
Antritter, Felix	7	Kadyrsizova, Zhibek	255	Roche, Daniel S.	103
Aschenbrenner, Matthias	117	Kaltofen, Erich	141, 155	Romanovski, Valery G.	255
Barkatou, Moulay A.	15	Kanno, Masaaki	147	Rónyai, Lajos	125
Bostan, Alin	23, 269	Kauers, Manuel	133	Rosenkranz, Markus	261
Brickenstein, Michael	55	Kemper, Gregor	335	Rouillier, Fabrice	79
Broughton, Gary	15	Koepf, Wolfram	39	Ruatta, Olivier	47
Burr, Michael	87	Koiran, Pascal	141	Rybowicz, Marc	239
Caboara, Massimo	315	Le Roux, Nicolas	23	Salvy, Bruno	269
Caruso, Fabrizio	315	Lee, J.	197	Schost, Éric	269
Chen, Falai	31	Lemaire, François	207	Schwarz, Fritz	277
Choi, Sung Woo.	87	Levandovskyy, Viktor	173	Šebek, Michael	111
Chyzak, Frédéric	23	Lévine, Jean	7	Sekigawa, Hiroshi	287
Daouda, Diatta Niang	47	Leykin, Anton	165	Shemyakova, Ekaterina	295
De Loera, Jesús A.	197	Li, Bin	155	Stein, William	5
Debeerst, Ruben	39	Li, Hongbo	181	Strzeboński, Adam	303
Deng, Jiansong	31	Liang, Songxin	189	Sudan, Madhu	337
Dreyer, Alexander	55	Malkin, Peter N.	197	Szántó, Ágnes	125
Dumas, Jean-Guillaume	63	Mansfield, Elizabeth L.	3, 295	Traverso, Carlo	315
El Din, Mohab Safey	71, 79	Margulies, Susan	197	van Hoeij, Mark	39
Faugère, Jean-Charles	79	Maza, Marc Moreno	189, 207	Wang, Xuhui	31
Fukuda, Komei	333	McCallum, Scott	215	Weispfenning, Volker	215
Galehouse, Benjamin	87	Morales, Jorge Martín	173	Wu, Xiaoli	325
Gerdt, Vladimir P.	95	Moroz, Guillaume	79	Xie, Yuzhen	207
Giesbrecht, Mark	103	Mourrain, Bernard	47, 125	Yang, Zhengfeng	155
Grigoriev, Dima	277	Odehnal, Boris	223	Yap, Chee K.	87
Hara, Shinji	147	Pan, Wei	207	Yokoyama, Kazuhiro	147, 247
Henrion, Didier	111	Peternell, Martin	223	Zhi, Lihong	155, 325
Hillar, Christopher J.	117	Pflügel, Eckhard	15, 231	Zinin, Mikhail V.	95