Lecture Notes in Artificial Intelligence 5081

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Deepak Kapur (Ed.)

Computer Mathematics

8th Asian Symposium, ASCM 2007
Singapore, December 15-17, 2007
Revised and Invited Papers

Springer

Series Editors

Randy Goebel, University of Alberta, Edmonton, Canada
Jörg Siekmann, University of Saarland, Saarbrücken, Germany
Wolfgang Wahlster, DFKI and University of Saarland, Saarbrücken, Germany

Volume Editor

Deepak Kapur
University of New Mexico, Department of Computer Science
Albuquerque, NM 87131-0001, USA
E-mail: kapur@cs.unm.edu

Library of Congress Control Number: 2008935385

CR Subject Classification (1998): I.2.2, I.1-2, F.4.1, G.2, I.6

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-540-87826-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-87826-1 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2008
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12524266 06/3180 5 4 3 2 1 0

Preface

This volume contains the proceedings of the Eighth Asian Symposium on Computer Mathematics (ASCM 2007), which was held at the Grand Plaza Park Hotel City Hall, Singapore, December 15–17, 2007. Previous ASCM meetings were held in Beijing, China (1995), Kobe, Japan (1996), Lanzhou, China (1998), Chiang Mai, Thailand (2000), Matsuyama, Japan (2001), Beijing, China (2003), and Seoul, Korea (2005).

Amongst 65 submissions by authors from 20 mostly Asian countries, the Program Committee selected 23 regular papers and 13 posters for presentation at the symposium. The presentations and papers went through another round of reviewing after the symposium, and 22 regular papers and five short papers on posters were selected for the proceedings. The international Program Committee of ASCM 2007 had strong Asian participation, and the reviewing process was aided by numerous reviewers from around the world. I am very grateful to the Program Committee members and the reviewers for their work in evaluating the submissions before and after the conference.

In addition to contributed papers, ASCM 2007 had three invited talks—by Rida Farouki on computational geometry, by Xiaoyun Wang on cryptology, and by Georges Gonthier on a computer proof of the celebrated Four Color Theorem. I would like to thank the speakers for their excellent talks. A paper by Prof. Farouki and his coauthors is included in the proceedings. Prof. Wang's research activities and publications can be found at her home page `http://www.infosec.sdu.edu.cn/2person_wangxiaoyun2.htm`. Details about Dr. Gunthier's computerized proof of the four color theorem can be found by visiting his home page `http://research.microsoft.com/~gonthier`.

It is my hope that ASCM continues to provide a forum for participants, especially from Asia, to present original research, to learn about new developments, and to exchange ideas and views on doing mathematics with computers.

ASCM 2007 was organized by the School of Computing of the National University of Singapore, and supported by the Lee Foundation, Kim Seng Holdings, and the Institute of Systems Science, Beijing, China. I thank Eng-Wee Chionh, who served as the General Chair, and the staff of the School of Computing of the National University of Singapore. Finally, I am grateful to Stephan Falke for his help in preparing this volume.

June 2008 Deepak Kapur

Conference Organization

Program Chair

Deepak Kapur	University of New Mexico, USA

Program Committee

Manindra Agarwal	IIT Kanpur, India
Leonid Bokut	Sobolev Institute, Russia
Shang-Ching Chou	Wichita State University, USA
Falai Chen	University of Science and Technology of China, China
Guoting Chen	University of Lille I, France
Eng-Wee Chionh	National University of Singapore, Singapore
Andreas Dolzmann	University of Passau, Germany
Ding-Zhu Du	University of Texas at Dallas, USA
Xiao-Shan Gao	Chinese Academy of Sciences, China
Shuhong Gao	Clemson University, USA
Keith Geddes	University of Waterloo, Canada
Vladimir Gerdt	Joint Institute for Nuclear Research, Russia
Hoon Hong	North Carolina State University, USA
Jieh Hsiang	National Taiwan University, Taiwan
Tetsuo Ida	University of Tsukuba, Japan
Seok-Jin Kang	Seoul National University, Korea
Yonggu Kim	Chonnam National University, Korea
Wen-shin Lee	University of Antwerp, Belgium
Ziming Li	Chinese Academy of Sciences, China
Miroslaw Majewski	NYIT Abu Dhabi, United Arab Emirates
Matu-Tarow Noda	Ehime Campus Information Services, Japan
Tobias Nipkow	Technical University of Munich, Germany
Hoang Xuan Phu	Academy of Science and Technology, Vietnam
Raja Natarajan	Tata Institute of Fundamental Research, India
Meera Sitharam	University of Florida, USA
Lim Yohanes Stefanus	University of Indonesia, Indonesia
Nobuki Takayama	Kobe University, Japan
Toby Walsh	National ICT, Australia
Dongming Wang	Beihang University, China and CNRS, France
Chaoping Xing	National University of Singapore, Singapore
Lu Yang	East China Normal University, China
Kazuhiro Yokoyama	Rikkyo University, Japan
Jianmin Zheng	Nanyang Technological University, Singapore

Conference Chairs

Eng-Wee Chionh	National University of Singapore, Singapore
Huaxiong Wang	Nanyang Technological University, Singapore

Publicity Chair

Dingkang Wang	Chinese Academy of Sciences, China

Local Organizing Committee

Secretariat	Stefanie Ng, Judy Ng
Web, CMT	Zaini Bin Mohammad
Finance	Lay Khim Chng, Noraiszah Hamzah, Rachel Goh
CD, Abstracts	Alexia Leong
Web Registration	Philip Lim
Audio-visual	Bernard Tay, Mohamad Nazri Bin Sulaiman, Chin Ming Chow

External Reviewers

John Abbott
Alkiviadis Akritas
Amir Amiraslani
Hirokazu Anai
Saugata Basu
Anna Bigatti
Peter Borwein
François Boulier
Jacek Chrzaszcz
Xavier Dahan
Jiansong Deng
Rida Farouki
Mitsushi Fujimoto
Laureano González Vega
Benjamin Gregoire
Markus Hitz
Fangjian Huang
Dorothy Kucar
Alexander Boris Levin
Yongbin Li
Bao Liu
Igor Markov
Marc Moreno Maza
A. S. Vasudeva Murthy
Masayuki Noro
Wei Pan
Pavel Pech
John Perry
Gerhard Pfister
Krishna Sankaranarayana
Eric Schost
Wolfgang Schreiner
Naresh Sharma
G. Sivakumar
K. V. Subrahmanyam
Laurent Théry
Vlad Timofte
Michel Toulouse
Ping-Sing Tsai
Regina Tyshkevich
Luca Vigano
Dingkang Wang
Wenping Wang
Xingmao Wang
Joab Winkler
Min Wu
Yuzhen Xie
Pingkun Yan
Noson Yanofsky
Alberto Zanoni
Zhengbing Zeng

Table of Contents

Algorithms and Implementations

Numerical Methods and Applications

Elimination: Triangular Forms, Resultants, Equation Solving

Cryptology

Computational Logic

Computing the Minkowski Value of the Exponential Function over a Complex Disk

Hyeong In Choi[1], Rida T. Farouki[2], Chang Yong Han[3], and Hwan Pyo Moon[1]

[1] Department of Mathematics, Seoul National University,
Seoul 151–747, South Korea
[2] Department of Mechanical and Aeronautical Engineering,
University of California, Davis, CA 95616, USA
[3] School of Electronics and Information, Kyung Hee University,
Yongin–si, Gyeonggi–do 446–701, South Korea
hichoi@snu.ac.kr, farouki@ucdavis.edu, cyhan@khu.ac.kr, hpmoon@snu.ac.kr

Abstract. Basic concepts, results, and applications of the Minkowski geometric algebra of complex sets are briefly reviewed, and preliminary ideas on its extension to evaluating transcendental functions of complex sets are discussed. Specifically, the Minkowski value of the exponential function over a disk in the complex plane is considered, as the limit of partial–sum sets defined by the monomial or Horner evaluation schemes.

1 Introduction

The *Minkowski sum* and *Minkowski product* of complex–number sets[1] $\mathcal{A}$, $\mathcal{B}$ are defined by

$$\begin{aligned} \mathcal{A} \oplus \mathcal{B} &= \{\, \mathbf{a} + \mathbf{b} \mid \mathbf{a} \in \mathcal{A} \text{ and } \mathbf{b} \in \mathcal{B} \,\}\,, \\ \mathcal{A} \otimes \mathcal{B} &= \{\, \mathbf{a} \times \mathbf{b} \mid \mathbf{a} \in \mathcal{A} \text{ and } \mathbf{b} \in \mathcal{B} \,\}\,. \end{aligned} \tag{1}$$

For "simple" operand sets $\mathcal{A}$ and $\mathcal{B}$ — e.g., circular disks (see Figure 1) — these expressions admit exact boundary evaluation [14]. For more general complex sets, bounded by piecewise–analytic curves, algorithms are available [10,11,13] to approximate Minkowski sum and product boundaries to any specified precision. Minkowski sums and products are commutative and associative, but products do *not* distribute over sums: we have, instead, the subdistributive inclusion relation

$$(\mathcal{A} \oplus \mathcal{B}) \otimes \mathcal{C} \;\subseteq\; (\mathcal{A} \otimes \mathcal{C}) \oplus (\mathcal{B} \otimes \mathcal{C})\,. \tag{2}$$

The sum and product (1) are basic operations in the *Minkowski algebra of complex sets*, which is concerned [14] with complex–number sets generated by certain combinations of complex values that vary independently over given set operands. Specifying the negation and reciprocal of set $\mathcal{B}$ by

$$-\mathcal{B} = \{\, -\mathbf{b} \mid \mathbf{b} \in \mathcal{B} \,\} \qquad \text{and} \qquad \mathcal{B}^{-1} = \{\, \mathbf{b}^{-1} \mid \mathbf{b} \in \mathcal{B} \,\}$$

[1] Following prior use [13,14] we denote real values by italic characters, complex values by bold characters, and sets of complex values by upper–case calligraphic characters.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 1–21, 2008.
© Springer-Verlag Berlin Heidelberg 2008

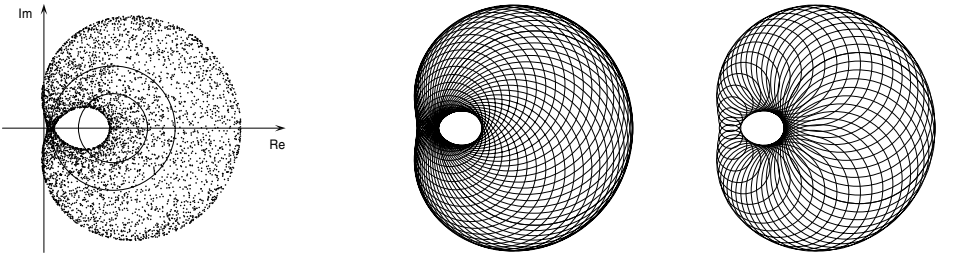


Fig. 1. Left: visualization of the Minkowski product of two circles as the region bounded by a *Cartesian oval* (a quartic algebraic curve) through a Monte Carlo experiment using products of randomly–sampled points on the circles. Right: Cartesian ovals realized as products of one circle with points of the other, and vice–versa. See [14] for more details, and [15] for a general theory accommodating the Minkowski product of $N > 2$ circles.

one can also introduce the "simple" Minkowski difference and division operations $\mathcal{A} \ominus \mathcal{B} = \mathcal{A} \oplus -\mathcal{B}$ and $\mathcal{A} \oslash \mathcal{B} = \mathcal{A} \otimes \mathcal{B}^{-1}$. Note, however, that $\ominus$ and $\oslash$ are *not* inverses of $\oplus$ and $\otimes$. Another type of Minkowski difference is defined [19,34] by

$$\mathcal{A} \ominus \mathcal{B} \;=\; (\,\mathcal{A}' \oplus -\mathcal{B}\,)'\,, \tag{3}$$

where $\mathcal{C}'$ denotes the *complement* of set $\mathcal{C}$. This difference (also known [19] as a Minkowski *decomposition*) does satisfy $(\mathcal{A} \oplus \mathcal{B}) \ominus \mathcal{B} = \mathcal{A}$ — although it is not always true that $(\mathcal{A} \ominus \mathcal{B}) \oplus \mathcal{B} = \mathcal{A}$. The Minkowski sum in (1) and the difference (3) can be interpreted, respectively, as set *unions* and *intersections*

$$\mathcal{A} \oplus \mathcal{B} \;=\; \bigcup_{\mathbf{b} \in \mathcal{B}} \mathcal{A} + \mathbf{b} \qquad \text{and} \qquad \mathcal{A} \ominus \mathcal{B} \;=\; \bigcap_{\mathbf{b} \in \mathcal{B}} \mathcal{A} - \mathbf{b}$$

of the translates $\mathcal{A} + \mathbf{b}$ and $\mathcal{A} - \mathbf{b}$ of set $\mathcal{A}$ by the points of $\mathcal{B}$ and $-\mathcal{B}$. This can be verified using de Morgan's laws

$$(\mathcal{A} \cup \mathcal{B})' \;=\; \mathcal{A}' \cap \mathcal{B}' \qquad \text{and} \qquad (\mathcal{A} \cap \mathcal{B})' \;=\; \mathcal{A}' \cup \mathcal{B}' \tag{4}$$

— i.e., set unions and intersections are exchanged under complementation.

Consider, for example, the case of circular disks. Taking $\mathcal{A} = D(\mathbf{c}_A, r_A)$ and $\mathcal{B} = D(\mathbf{c}_B, r_B)$ where $D(\mathbf{c}, r)$ denotes the disk with center $\mathbf{c}$ and radius r, we have $\mathcal{A} \oplus \mathcal{B} = D(\mathbf{c}_A + \mathbf{c}_B, r_A + r_B)$. Then $\mathcal{A} \ominus \mathcal{B} = \mathcal{A} \oplus -\mathcal{B}$ gives $D(\mathbf{c}_A - \mathbf{c}_B, r_A + r_B)$, so that $(\mathcal{A} \ominus \mathcal{B}) \oplus \mathcal{B} = D(\mathbf{c}_A, r_A + 2r_B) \neq \mathcal{A}$ if $r_B \neq 0$. On the other hand, the definition (3) gives $\mathcal{A} \ominus \mathcal{B} = D(\mathbf{c}_A - \mathbf{c}_B, r_A - r_B)$ when $r_A \geq r_B$, and the empty set $\emptyset$ when $r_A < r_B$. Hence, definition (3) yields $(\mathcal{A} \ominus \mathcal{B}) \oplus \mathcal{B} = \mathcal{A}$ when $r_A \geq r_B$, but $(\mathcal{A} \ominus \mathcal{B}) \oplus \mathcal{B} = \emptyset$ when $r_A < r_B$.

By analogy with (3), an alternative Minkowski division can be defined by

$$\mathcal{A} \oslash \mathcal{B} \;=\; (\,\mathcal{A}' \otimes \mathcal{B}^{-1}\,)'\,. \tag{5}$$

This satisfies $(\mathcal{A} \otimes \mathcal{B}) \oslash \mathcal{B} = \mathcal{A}$, but $(\mathcal{A} \oslash \mathcal{B}) \otimes \mathcal{B} = \mathcal{A}$ does not always hold. The Minkowski product in (1) and the division (5) can be interpreted, respectively, as set unions and intersections

$$\mathcal{A} \otimes \mathcal{B} = \bigcup_{\mathbf{b} \in \mathcal{B}} \mathcal{A}\,\mathbf{b} \qquad \text{and} \qquad \mathcal{A} \oslash \mathcal{B} = \bigcap_{\mathbf{b} \in \mathcal{B}} \mathcal{A}\,\mathbf{b}^{-1}$$

of the scalings/rotations $\mathcal{A}\,\mathbf{b}$ and $\mathcal{A}\,\mathbf{b}^{-1}$ of set $\mathcal{A}$ by the points of $\mathcal{B}$ and $\mathcal{B}^{-1}$ — again, one can invoke de Morgan's laws (4) to verify this.

The Minkowski algebra can be usefully extended in many ways. For example, replacing the binary operations $\mathbf{a}+\mathbf{b}$ and $\mathbf{a}\times\mathbf{b}$ by an analytic bivariate function $\mathbf{f}(\mathbf{a},\mathbf{b})$ we obtain the "implicitly–defined" set denoted [13] by $\mathcal{A}\,\textcircled{\mathbf{f}}\,\mathcal{B}$. Whereas the sum $\mathcal{A} \oplus \mathcal{B}$ and product $\mathcal{A} \otimes \mathcal{B}$ can be regarded as unions of *translations* and *scalings/rotations* of set $\mathcal{A}$ by the points of set $\mathcal{B}$ (or vice–versa), respectively, the implicitly–defined set $\mathcal{A}\,\textcircled{\mathbf{f}}\,\mathcal{B}$ can be regarded as a union of *conformal mappings* of one set, dependent upon the points of the other set [13].

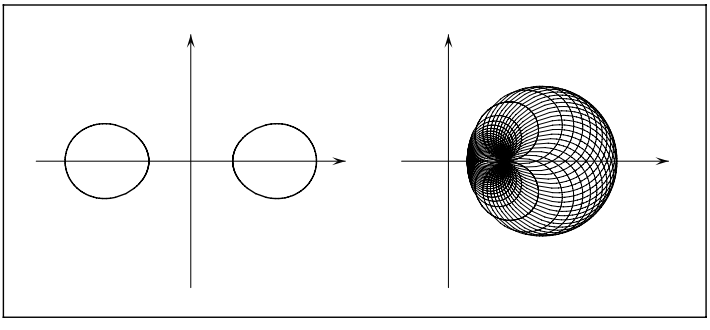


Fig. 2. A single loop of the *ovals of Cassini*, a quartic algebraic curve (left), specifies the Minkowski square root of a circular disk (right) that does not include the origin [5] — this result is also generalized in [5] to identify the n^{th} Minkowski roots of a disk

Unary set operations can also be introduced. For example, the n^{th} *Minkowski power* $\otimes^n\mathcal{A}$ and n^{th} *Minkowski root* $\otimes^{1/n}\mathcal{A}$ of a set $\mathcal{A}$ are specified by

$$\otimes^n\mathcal{A} = \{\, \mathbf{z}_1\mathbf{z}_2\cdots\mathbf{z}_n \mid \mathbf{z}_i \in \mathcal{A} \;\text{ for }\; i = 1,\ldots,n \,\},$$

$$\{\, \mathbf{z}_1\mathbf{z}_2\cdots\mathbf{z}_n \mid \mathbf{z}_i \in \otimes^{1/n}\mathcal{A} \;\text{ for }\; i = 1,\ldots,n \,\} = \mathcal{A},$$

and are also amenable to closed–form evaluations [5,15] if $\mathcal{A}$ is a circular disk — see Figure 2. Another important unary operation corresponds to evaluation of a function $f(\mathcal{X})$ with a set argument $\mathcal{X}$. This is, however, a more subtle problem than in the case of a scalar argument. Since Minkowski sums and products do not obey the distributive law, a *specific algorithm* for the evaluation (describing the exact order of its arithmetic operations) must be specified to uniquely define $f(\mathcal{X})$. The case of a polynomial f with a circular disk $\mathcal{X}$ as argument was studied

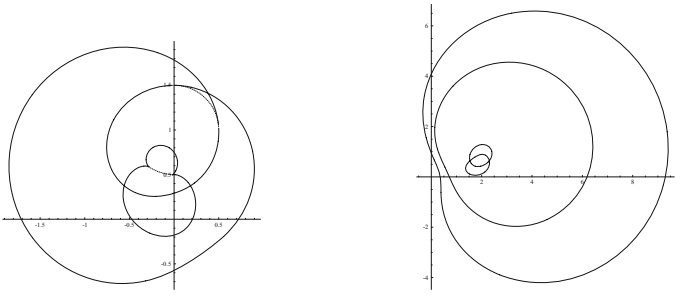


Fig. 3. Envelope curves (boundary supersets) for the Minkowski Horner values of two cubic polynomials over circular disks in the complex plane: see [6] for complete details

in [6], in the context of both the *monomial* and *Horner* evaluation algorithms — see Figure 3 for some examples.

The solution of elementary equations in the Minkowski algebra, involving an unknown set $\mathcal{X}$ and given "simple" coefficient sets $\mathcal{A}, \mathcal{B} \ldots$, was considered in [7]. For circular disks $\mathcal{A}$ and $\mathcal{B}$, sufficient–and–necessary conditions were specified for the existence of a solution $\mathcal{X}$ (see Figure 4) to the linear equation

$$\mathcal{A} \otimes \mathcal{X} \, = \, \mathcal{B} \, , \tag{6}$$

and the nature of the solution was determined. The generalization of (6) to the case where $\mathcal{X}$ is replaced by the n^{th} Minkowski power $\otimes^n \mathcal{X}$ was also treated, and the extension to linear systems in several unknown sets was discussed.

The Minkowski algebra of sets in $\mathbb{C}$ has diverse applications and connections. It may be regarded as the natural extension of (real) *interval arithmetic* [29,30] to complex–number sets. Its two–dimensional nature, however, endows it with a richer geometrical content. It offers an elegant "shape operator" language for

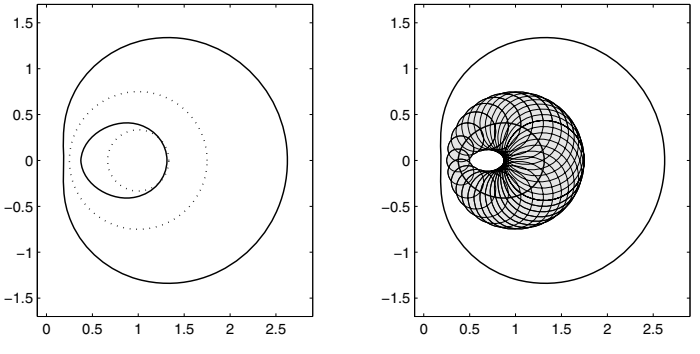


Fig. 4. Left: solution of the equation $\mathcal{A} \otimes \mathcal{X} = \mathcal{B}$ when $\mathcal{A}$, $\mathcal{B}$ are disks (shown dotted) — see [7]. The solution $\mathcal{X}$ is the region bounded by the inner loop of the Cartesian oval shown as a solid locus. Right: as $\mathbf{x}$ traces the inner loop of the Cartesian oval, the envelope of the one–parameter family of disks $\mathcal{A} \otimes \{\mathbf{x}\}$ yields the boundary of disk $\mathcal{B}$.

planar domains, with connections to direct and inverse wavefront constructions in geometric optics [3,4,14], stability analysis of dynamic systems with uncertain parameters [8,9,12], and image analysis and mathematical morphology [32,33]. The Minkowski sum is a basic operation [20,26,28] in Euclidean geometry, with a natural generalization to $\mathbb{R}^n$ for $n \geq 3$, and important applications in geometric design, computer graphics, path planning, and related fields [18,22,23,24,25,27].

We are interested here in extending the repertoire of Minkowski set operations to accommodate *transcendental* functions of complex sets — in particular, the *Minkowski exponential* of a circular disk $\mathcal{A}$ in the complex plane. This is defined by evaluating the Taylor series of the exponential function, up to the n–th order term, using the Minkowski monomial or Horner algorithms described in [6], and considering the proper limit as $n \to \infty$ of the infinite sequence of complex sets thus defined. The resulting complex sets, $\exp_m(\mathcal{A})$ and $\exp_h(\mathcal{A})$, differ because of the disparate ways in which complex values $\mathbf{z} \in \mathcal{A}$ are used in their generation: whereas the truncated series for $\exp_m(\mathcal{A})$ employs $\frac{1}{2}n(n+1)$ independent $\mathbf{z}$ values (each monomial term is evaluated independently), that for $\exp_h(\mathcal{A})$ incurs only n values of $\mathbf{z}$ in the "nested multiplication" process. Furthermore, $\exp_m(\mathcal{A})$ and $\exp_h(\mathcal{A})$ both differ from the simple image of $\mathcal{A}$ under the exponential map $\mathbf{z} \to \mathrm{e}^{\mathbf{z}}$, which we denote here by $\mathrm{e}^{\mathcal{A}}$.

2 Random–Coefficient Differential Equation

We illustrate here one possible application of the Minkowski exponential, in the set–valued solution to a differential equation whose coefficients exhibit random variations over prescribed sets. By "random variations" we do not mean that the coefficients are *stochastic functions* of the independent variable (e.g., time), but rather that they are regarded like *quantum variables*, i.e., each time they are invoked they exhibit independent random values, in accordance with a definite probability distribution over some prescribed domain.

Consider the complex function $\mathbf{z}(t) = x(t) + \mathrm{i}\, y(t)$ satisfying the linear first–order differential

$$\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}t} \;=\; \mathbf{k}\,\mathbf{z}\,, \tag{7}$$

where $\mathbf{k} = \lambda + \mathrm{i}\,\mu$ is a complex value. For initial conditions $\mathbf{z}(0) = \mathbf{z}_0 = x_0 + \mathrm{i}\, y_0$, the formal solution is

$$\mathbf{z}(t) \;=\; \mathbf{z}_0\, \mathrm{e}^{\mathbf{k}t}\,. \tag{8}$$

Equation (7) is equivalent to a system of coupled first–order equations

$$\begin{bmatrix} x' \\ y' \end{bmatrix} \;=\; \begin{bmatrix} \lambda & -\mu \\ \mu & \lambda \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

for the real functions $x(t)$ and $y(t)$, specified by a skew–symmetric matrix. With $A = \sqrt{x_0^2 + y_0^2}$ and $\cos\phi = x_0/A$, $\sin\phi = y_0/A$ these functions have the form

$$x(t) \;=\; A\,\mathrm{e}^{\lambda t}\cos(\phi + \mu t)\,, \qquad y(t) \;=\; A\,\mathrm{e}^{\lambda t}\sin(\phi + \mu t)\,.$$

Now in (8), we consider the value of the exponential with complex argument $\mathbf{k}t$ to be defined by

$$\mathrm{e}^{\mathbf{k}t} = \lim_{n\to\infty} \left[1 + \mathbf{k}t + \frac{(\mathbf{k}t)^2}{2!} + \cdots + \frac{(\mathbf{k}t)^n}{n!} \right], \tag{9}$$

where the partial Taylor series, up to the n–th order term, is to be evaluated by a particular algorithm.

We now consider the solutions to (7) when the parameter $\mathbf{k}$ is interpreted as a random variable, confined to a disk $\mathcal{A}$ in the complex plane. There are several possible models for such an interpretation. The simplest is to consider complex numbers $\mathbf{k} \in \mathcal{A}$ selected *a priori* that are regarded as deterministic, constant values during the integration of (7). Then we simply have a family of solutions of the form (8) with $\mathbf{k}$ varying over $\mathcal{A}$. A perhaps more–interesting model is to consider $\mathbf{k}$ exhibiting some kind of random variation with t, confined to the domain $\mathcal{A}$. This is in the spirit of the so–called *stochastic differential equations* [1,16,17,31] used to model price fluctuations in financial markets.

Our interest is in a third, and even more general, interpretation — namely $\mathbf{k}$ is interpreted as a kind of "quantum variable" that yields a random value[2] from within $\mathcal{A}$ whenever it is measured or used (quantum variables have inherently indeterminate values, and only the relative probabilities for the outcome of their measurements are known). Equation (7) then has a set–valued solution, that can be described in terms of the Minkowski exponential as

$$\{\mathbf{z}_0\} \otimes \exp_m(\mathcal{A}\, t) \qquad \text{or} \qquad \{\mathbf{z}_0\} \otimes \exp_h(\mathcal{A}\, t)\,,$$

according to whether the monomial or Horner algorithm is used in evaluating the partial Taylor series in (9).

3 Minkowski Exponential of a Real Interval

Before studying the Minkowski exponentials of complex sets, it is instructive to consider first the simpler case of real intervals. According to the usual rules of real interval arithmetic [29,30] we have

$$\begin{aligned}
&[\,a,b\,] + [\,c,d\,] = [\,a+c, b+d\,]\,,\\
&[\,a,b\,] - [\,c,d\,] = [\,a-d, b-c\,]\,,\\
&[\,a,b\,] \times [\,c,d\,] = [\,\min(ac,ad,bc,bd), \max(ac,ad,bc,bd)\,]\,,\\
&[\,a,b\,] \div [\,c,d\,] = [\,a,b\,] \times [\,1/d, 1/c\,]\,.
\end{aligned} \tag{10}$$

Division is usually restricted to intervals with $0 \notin [\,c,d\,]$. Sums and products of intervals are commutative and associative, but products do not distribute over sums — instead, we have the sub–distributive inclusion relation

$$[\,a,b\,] \times ([\,c,d\,] + [\,e,f\,]) \;\subseteq\; ([\,a,b\,] \times [\,c,d\,]) + ([\,a,b\,] \times [\,e,f\,])\,.$$

Note also that the interval operators $-$, $\div$ are *not* inverses to $+$, $\times$.

[2] As a default, we assume that $\mathbf{k}$ has a uniform probability distribution over $\mathcal{A}$ — but any probability density function could, in principle, be used.

Consider a non–degenerate interval $I = [\,a, b\,]$. To compute the Minkowski exponential of I, it is convenient to identify four distinct cases:

$$\begin{aligned} &\text{case (1)}: \ 0 \le a < b\,, \\ &\text{case (2)}: \ a < 0 < b \quad \text{and} \quad |a| < b\,, \\ &\text{case (3)}: \ a < 0 < b \quad \text{and} \quad |a| > b\,, \\ &\text{case (4)}: \ a < b \le 0\,. \end{aligned} \tag{11}$$

The exponential e^x of a real variable x is defined by the infinite series

$$\mathrm{e}^x \ = \ 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots\,,$$

convergent for all x. To uniquely define a value for the Minkowski exponential of a set argument, we interpret it as the limit of a sequence of sets obtained by evaluating the partial sums of the above infinite series *using a specific algorithm* — since, in general, the algebra of sets does not satisfy the distributive law. We consider two algorithms: *monomial* evaluation, in which the terms of the series are independently evaluated and then summed, and *Horner* evaluation, in which "nested multiplication" is used to evaluate the partial sums.

Consider the *monomial* form of the Minkowski exponential of I, denoted $\exp_m(I)$. We need to compute the Minkowski powers of I, defined by

$$\otimes^k I \ = \ \underbrace{[\,a, b\,] \times \cdots \times [\,a, b\,]}_{k \text{ times}},$$

with $\otimes^0 I = [\,1, 1\,]$. The following results can be easily verified by induction

$$\text{case (1)}: \quad \otimes^k I = [\,a^k\,,\, b^k\,] \quad \text{for } k = 1, 2, 3, 4, \ldots,$$

$$\text{case (2)}: \quad \otimes^k I = [\,a b^{k-1}\,,\, b^k\,] \quad \text{for } k = 1, 2, 3, 4, \ldots,$$

$$\text{case (3)}: \quad \otimes^k I = \begin{cases} [\,a^k\,,\, a^{k-1} b\,] & \text{for } k = 1, 3, \ldots \\ [\,a^{k-1} b\,,\, a^k\,] & \text{for } k = 2, 4, \ldots \end{cases},$$

$$\text{case (4)}: \quad \otimes^k I = \begin{cases} [\,a^k\,,\, b^k\,] & \text{for } k = 1, 3, \ldots \\ [\,b^k\,,\, a^k\,] & \text{for } k = 2, 4, \ldots \end{cases},$$

using the product rule in (10) with $[\,c, d\,] = [\,a, b\,]$. The Minkowski monomial value for the exponential of the real interval I, defined by

$$\exp_m(I) \ = \ \sum_{k=0}^{\infty} \frac{1}{k!} \otimes^k I\,,$$

is an (infinite) weighted sum of the Minkowski powers $\otimes^k I$. Using the addition rule from (10) and taking the appropriate limits, we deduce that

$$\text{case (1)}: \quad \exp_m(I) = [\, \mathrm{e}^a \,,\, \mathrm{e}^b \,] \,,$$

$$\text{case (2)}: \quad \exp_m(I) = [\, 1 + \frac{a}{b}\,(\mathrm{e}^b - 1) \,,\, \mathrm{e}^b \,] \,,$$

$$\text{case (3)}: \quad \exp_m(I) = [\, 1 + \sinh a + \frac{b}{a}\,(\cosh a - 1) \,,\, \frac{b}{a}\sinh a + \cosh a \,] \,,$$

$$\text{case (4)}: \quad \exp_m(I) = [\, \sinh a + \cosh b \,,\, \sinh b + \cosh a \,] \,.$$

These expressions agree in the three transitional cases $a = 0$, $|a| = |b|$, $b = 0$.

On the other hand, the *Horner* value $\exp_h(I)$ for the Minkowski exponential of the interval I is defined as the limit

$$\exp_h(I) \;=\; \lim_{k\to\infty} H_k \,,$$

where the intervals H_k are "partial Horner sums" generated by the recursion

$$H_r \;=\; (H_{r-1} \otimes I) \;\oplus\; \left\{ \frac{1}{(k-r)!} \right\} \quad \text{for} \quad r = 1, \ldots, k$$

with $H_0 = \{1/k!\}$. The k–th Horner sum can be equivalently expressed as

$$H_k \;=\; \left\{ \frac{t_1 \cdots t_k}{k!} + \frac{t_2 \cdots t_k}{(k-1)!} + \cdots + \frac{t_{k-1} t_k}{2!} + t_k + 1 \;\middle|\; t_1, \ldots, t_k \in I \right\} .$$

For case (1), the lower and upper bounds of H_k are evidently realized by choosing $t_1 = \cdots = t_k = a$ and $t_1 = \cdots = t_k = b$, respectively. In case (2), the upper bound is again obtained with $t_1 = \cdots = t_k = b$, and the lower bound is achieved with $t_1, \ldots, t_{k-1} = b$ and $t_k = a$. Hence, we have

$$\text{case (1)}: \quad H_k = \left[\sum_{r=0}^{k} \frac{a^r}{r!} \,,\, \sum_{r=0}^{k} \frac{b^r}{r!} \right] ,$$

$$\text{case (2)}: \quad H_k = \left[1 + \frac{a}{b} \sum_{r=1}^{k} \frac{b^r}{r!} \,,\, \sum_{r=0}^{k} \frac{b^r}{r!} \right] .$$

Case (3) is more subtle, since there is no simple universal rule that identifies the combinations of $t_1, \ldots, t_k$ values generating the lower and upper bounds of H_k — the choice depends on the relative magnitudes of a and b. For example, with $k = 7$ and $[\, a, b \,] = [\, -1.2, 0.7 \,]$ the lower bound is generated by $t_1, \ldots, t_6 = b$ and $t_7 = a$, and the upper bound by $t_1 = \cdots = t_7 = b$, as in case (2). However, for $k = 6$ and $[\, a, b \,] = [\, -5.2, 0.3 \,]$ the lower bound is obtained with $t_1, \ldots, t_6 = b, a, a, a, a, a$, and the upper bound with $t_1, \ldots, t_6 = b, b, a, a, a, a$. For the case $k = 7$ and

$[\,a,b\,] = [\,-3.6, 1.7\,]$ the upper and lower bounds are generated by the choices $t_1, \ldots, t_6 = b, b, b, b, a, a, a$ and b, b, b, b, b, a, a respectively. The same phenomenon is observed in case (4) — the numerical values of a, b must be known in order to identify the lower/upper bounds for each Horner sum H_k.

Cases (3) and (4) indicate that, in general, it is not possible to express each Horner sum H_k symbolically in terms of a and b without knowing their precise magnitudes. This complicates determining the Horner value for the Minkowski exponential of an interval, as compared to the monomial value discussed above. However, in the "simple" cases (1) and (2), taking the limit as $k \to \infty$ gives

$$\text{case (1)}: \quad \exp_h(I) = [\,\mathrm{e}^a\,,\,\mathrm{e}^b\,]\,,$$

$$\text{case (2)}: \quad \exp_h(I) = [\,1 + \frac{a}{b}\,(\mathrm{e}^b - 1)\,,\,\mathrm{e}^b\,]\,.$$

Comparing the Minkowski monomial and Horner values $\exp_m(I)$ and $\exp_h(I)$ with the image of the interval I under the exponential map, defined by

$$\mathrm{e}^I \;=\; \{\,\mathrm{e}^t \mid t \in I\,\} \;=\; [\,\mathrm{e}^a\,,\,\mathrm{e}^b\,]\,,$$

we have the following inclusion relations

$$\text{case (1)}: \quad \mathrm{e}^I = \exp_h(I) \;=\; \exp_m(I)\,,$$

$$\text{case (2)}: \quad \mathrm{e}^I \subset \exp_h(I) \;=\; \exp_m(I)\,.$$

Also, $\mathrm{e}^I \subset \exp_m(I)$ for the Minkowski monomial value in cases (3) and (4).

In computing the Minkowski sum or product of complex disks $\mathcal{A}$ and $\mathcal{B}$ with centers $\mathbf{a}$ and $\mathbf{b}$, it is convenient to transform the operands to certain "canonical" positions [14]. For the Minkowski sum this amounts to moving the disk centers to the origin by the translations $-\mathbf{a}$ and $-\mathbf{b}$, while for the Minkowski product one moves the centers to the point 1 of the real axis by the complex scalings $1/\mathbf{a}$ and $1/\mathbf{b}$. The Minkowski sum or product of the original operands is then obtained from that of the transformed operands by the translation $\mathbf{a} + \mathbf{b}$ or the scaling/rotation defined by multiplying with $\mathbf{ab}$, respectively.

In general, however, the Minkowski exponential of a set does not admit such transformations to and from canonical position. We show this in the case (1) of a real interval $I = [\,a,b\,]$ with center $c = \frac{1}{2}(a+b)$ and half–width $w = \frac{1}{2}(b-a)$ for the monomial Minkowski exponential. Translating by $-c$ gives the interval $\tilde{I} = [\,-w, +w\,]$ centered on the origin, and we might try to recover $\exp_m(I)$ from $\exp_m(\tilde{I})$ by multiplying with e^c, since the exponential function satisfies $\mathrm{e}^c\mathrm{e}^x = \mathrm{e}^{c+x}$. In case (1) we have $I = [\,a,b\,]$ with $0 \le a < b$, and $\exp_m(I) = [\,\mathrm{e}^a, \mathrm{e}^b\,]$. On the other hand, we find that $\exp_m(\tilde{I}) = [\,2 - \mathrm{e}^w, \mathrm{e}^w\,]$ and hence

$$\mathrm{e}^c \exp_m(\tilde{I}) \;=\; [\,2\,\mathrm{e}^{\frac{1}{2}(a+b)} - \mathrm{e}^b, \mathrm{e}^b\,]\,.$$

Clearly, $\mathrm{e}^c \exp_m(\tilde{I}) \neq \exp_m(I)$ since $a < b$ by assumption.

Similarly, for two real intervals $I_1 = [\, a_1, b_1 \,]$ and $I_2 = [\, a_2, b_2 \,]$ one can readily verify in case (1) that

$$\exp_m(I_1 + I_2) \,=\, \exp_m(I_1) \exp_m(I_2)\,, \quad \exp_h(I_1 + I_2) \,=\, \exp_h(I_1) \exp_h(I_2)\,,$$

but for cases (2)–(4) we have

$$\exp_m(I_1 + I_2) \,\neq\, \exp_m(I_1) \exp_m(I_2)\,, \quad \exp_h(I_1 + I_2) \,\neq\, \exp_h(I_1) \exp_h(I_2)\,.$$

4 Exponential Image of a Circular Disk

Consider the image $\mathrm{e}^{\mathcal{A}}$ of the complex–plane disk $\mathcal{A}$ with center $\mathbf{c}$ and radius r under the exponential map $\mathbf{z} \to \mathrm{e}^{\mathbf{z}}$. It will be of interest to compare $\mathrm{e}^{\mathcal{A}}$ with the Minkowski exponentials $\exp_m(\mathcal{A})$ and $\exp_h(\mathcal{A})$. Without loss of generality, we may use the canonical disk

$$\tilde{\mathcal{A}} \,=\, \mathcal{A} \ominus \{\mathbf{c}\}$$

with center at the origin and radius r. The images of $\mathcal{A}$ and $\tilde{\mathcal{A}}$ are related by

$$\mathrm{e}^{\mathcal{A}} \,=\, \{\mathrm{e}^{\mathbf{c}}\} \otimes \mathrm{e}^{\tilde{\mathcal{A}}}$$

— i.e., $\mathrm{e}^{\mathcal{A}}$ is a scaling/rotation of $\mathrm{e}^{\tilde{\mathcal{A}}}$ by the complex number $\mathrm{e}^{\mathbf{c}}$. The boundary of the disk with center 0 and radius r has the parameterization

$$\mathbf{z}(\theta) \,=\, r\,\mathrm{e}^{\mathrm{i}\,\theta} \,=\, r\cos\theta \,+\, \mathrm{i}\, r \sin\theta$$

for $0 \le \theta < 2\pi$, and under the exponential map this curve becomes

$$\mathrm{e}^{\mathbf{z}(\theta)} \,=\, \mathrm{e}^{r\cos\theta}\,[\,\cos(r\sin\theta) \,+\, \mathrm{i}\,\sin(r\sin\theta)\,]\,. \tag{12}$$

Since $\mathrm{Re}(\mathrm{e}^{\mathbf{z}(-\theta)}) = \mathrm{Re}(\mathrm{e}^{\mathbf{z}(\theta)})$ and $\mathrm{Im}(\mathrm{e}^{\mathbf{z}(-\theta)}) = -\,\mathrm{Im}(\mathrm{e}^{\mathbf{z}(\theta)})$, it is symmetric about the real axis. It has the two points

$$\mathrm{e}^{\mathbf{z}(0)} \,=\, \mathrm{e}^{r} \qquad \text{and} \qquad \mathrm{e}^{\mathbf{z}(\pi)} \,=\, \mathrm{e}^{-r}\,,$$

on the positive real axis. These are the only points on the real axis if $r < \pi$, but for $r \ge \pi$ there are additional real points — they correspond to the values

$$\theta \,=\, \pm\sin^{-1}\frac{k\pi}{r} \qquad \text{for } \; k = 1, \ldots, n$$

where $n = \lfloor r/\pi \rfloor$. Similarly, $\mathrm{e}^{\mathbf{z}(\theta)}$ has points on the imaginary axis only when $r > \frac{1}{2}\pi$ — they correspond to the values

$$\theta \,=\, \pm\sin^{-1}\frac{(k-\frac{1}{2})\pi}{r} \qquad \text{for } \; k = 1, \ldots, n-1\,.$$

Examples of these curves are shown in Figures 5 and 6. The number of nested loops increases with r, and the boundary of $\mathrm{e}^{\mathcal{A}}$ is defined by the outermost loop, which (when $r > \pi$) corresponds to restricting (12) to the domain

$$-\sin^{-1}\frac{\pi}{r} \,\le\, \theta \,\le\, +\sin^{-1}\frac{\pi}{r}\,.$$

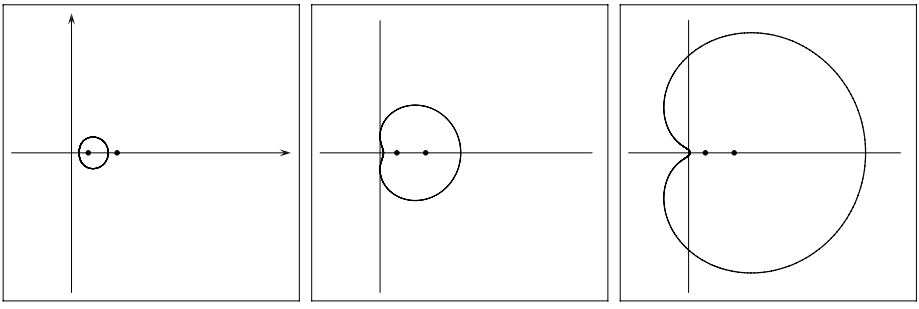


Fig. 5. The curve (12) with $r = \frac{1}{4}\pi$, $\frac{1}{2}\pi$ $\frac{3}{4}\pi$ (left to right). The dots indicate the points 1 and e on the real axis. For $r < \frac{1}{2}\pi$ the curve lies in the right half–plane; for $r = \frac{1}{2}\pi$ it is tangent to the imaginary axis; and for $r > \frac{1}{2}\pi$ it crosses into the left half–plane.

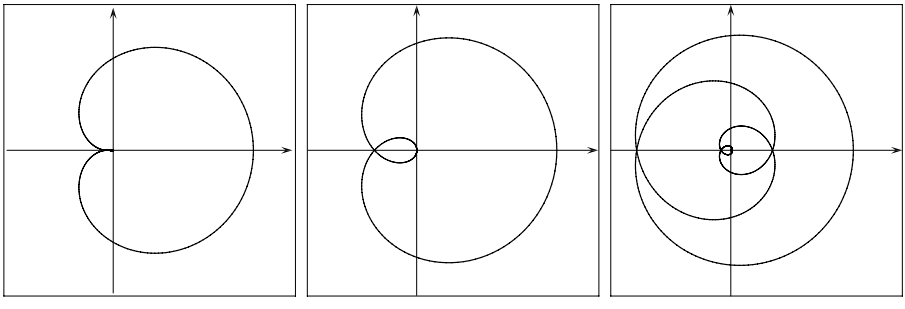


Fig. 6. The curve defined by (12) for $r = \pi$ (left), $r = \frac{3}{2}\pi$ (center), and $r = 6\pi$ (right). Note that, for clarity, these three plots employ different scales.

5 Monomial Minkowski Exponential

For a compact simply–connected domain $\mathcal{A}$ in the complex plane, the monomial Minkowski exponential may be formally expressed as

$$\exp_m(\mathcal{A}) \;=\; \bigoplus_{k=0}^{\infty} \frac{1}{k!} \otimes^k \mathcal{A}\,,$$

which can (in some sense) be interpreted as the limit of the partial sums

$$\mathcal{S}_n \;=\; \bigoplus_{k=0}^{n} \frac{1}{k!} \otimes^k \mathcal{A}\,. \tag{13}$$

A well–established means of defining the limit of an infinite sequence of sets is the Painlevé–Kuratowski convergence. For a given sequence of sets $\{\mathcal{C}_n\}$ in a Banach space X, the *Painlevé–Kuratowski limit* is defined by

$$\lim_{n\to\infty} \mathcal{C}_n \;=\; \{\, \mathbf{x} \in X \,|\, \lim_{n\to\infty} \mathit{dist}(\mathbf{x}, \mathcal{C}_n) = 0 \,\}\,,$$

$dist(\mathbf{x}, \mathcal{C}_n)$ being the distance from $\mathbf{x}$ to $\mathcal{C}_n$, measured in the metric induced by the norm of the Banach space X — see [2] and references therein for a detailed formulation of the Painlevé–Kuratowski convergence. A rigorous definition of the monomial Minkowski exponential of a complex set may be formulated in terms of the Painlevé–Kuratowski convergence as follows.

Definition 1. For a compact simply–connected set $\mathcal{A}$ in the complex plane, the *monomial Minkowski exponential* $\exp_m(\mathcal{A})$ is the Painlevé–Kuratowski limit of the sequence of sets $\{\mathcal{S}_n\}$ specified by the partial sums (13).

The following proposition describes methods for identifying points in the monomial Minkowski exponential of a complex set $\mathcal{A}$.

Proposition 2. *Let $\mathcal{A}$ be a compact simply–connected domain in the complex plane, let $\mathcal{M}_k = 1/k! \otimes^k \mathcal{A}$ be the k^{th} monomial term,*[3] *and let $\mathcal{S}_n$ be the partial sum (13). Then, for any complex number $\mathbf{x}$, the following are equivalent:*

(a) *The complex number $\mathbf{x}$ is contained in $\exp_m(\mathcal{A})$.*
(b) *A sequence of complex numbers $\{\mathbf{z}_k\}$ with $\mathbf{z}_k \in \mathcal{M}_k$ exists, such that*

$$\mathbf{x} \;=\; \sum_{k=0}^{\infty} \mathbf{z}_k \,.$$

(c) *A doubly–indexed sequence of complex numbers $\{\mathbf{w}_{jk}\}$ exists, such that $\mathbf{w}_{jk} \in \mathcal{A}$ for $1 \le j \le k < \infty$ and*

$$\mathbf{x} \;=\; \sum_{k=0}^{\infty} \left(\frac{1}{k!} \prod_{j=1}^{k} \mathbf{w}_{jk} \right) . \tag{14}$$

Proof: (a) $\Rightarrow$ (b) For a compact domain $\mathcal{A}$, every partial sum $\mathcal{S}_n$ is also compact, since it is a Minkowski sum of a finite number of scaled Minkowski powers of $\mathcal{A}$. Hence, from every $\mathcal{S}_n$ we can choose a point $\mathbf{s}_n$ for which the distance from $\mathbf{x}$ to $\mathcal{S}_n$ is realized — i.e.,

$$dist(\mathbf{x}, \mathcal{S}_n) \;=\; |\mathbf{x} - \mathbf{s}_n| \quad \text{for some } \mathbf{s}_n \in \mathcal{S}_n \,.$$

Since $\mathbf{x}$ is contained in the Painlevé–Kuratowski limit of $\{\mathcal{S}_n\}$, we have

$$\lim_{n\to\infty} |\mathbf{x} - \mathbf{s}_n| \;=\; \lim_{n\to\infty} dist(\mathbf{x}, \mathcal{S}_n) \;=\; 0 \,.$$

Thus, $\mathbf{x}$ is the limit point of the sequence $\{\mathbf{s}_n\}$. Now let $\mathbf{z}_k = \mathbf{s}_k - \mathbf{s}_{k-1}$ for $k \ge 1$ with $\mathbf{z}_0 = 1$. Then we have

$$\mathbf{s}_n \;=\; \sum_{k=0}^{n} \mathbf{z}_k \qquad \text{and} \qquad \sum_{k=0}^{\infty} \mathbf{z}_k \;=\; \lim_{n\to\infty} \mathbf{s}_n \;=\; \mathbf{x} \,.$$

[3] Note here that $\mathcal{M}_0 = \{1\}$. Also, the $k = 0$ term of the sum (14) has the value 1.

The fact that $\mathbf{z}_k \in \mathcal{M}_k$ follows immediately by noting that $\mathbf{s}_k = \mathbf{s}_{k-1} + \mathbf{z}_k$ and $\mathcal{S}_k = \mathcal{S}_{k-1} \oplus \mathcal{M}_k$.

(b) $\Rightarrow$ (c) Let $\{\mathbf{z}_k\}$ be the complex sequence for which condition (b) holds. Since $\mathbf{z}_k \in \mathcal{M}_k$, the k^{th} Minkowski power $\otimes^k\mathcal{A}$ contains $k!\,\mathbf{z}_k$. Hence, for each k, we can choose a k–tuple $(\mathbf{w}_{1k}, \cdots, \mathbf{w}_{kk})$ from $\mathcal{A}$ such that

$$\mathbf{z}_k \;=\; \frac{1}{k!} \prod_{j=1}^{k} \mathbf{w}_{jk}\,.$$

Condition (c) holds for such a choice of the doubly–indexed sequence $\{\mathbf{w}_{jk}\}$.

(c) $\Rightarrow$ (a) Let $\{\mathbf{w}_{jk}\}$ be the doubly–indexed sequence chosen from $\mathcal{A}$ for which condition (c) holds. Then the partial sum given by

$$\mathbf{s}_n \;=\; \sum_{k=0}^{n} \left(\frac{1}{k!} \prod_{j=1}^{k} \mathbf{w}_{jk} \right)$$

is contained in $\mathcal{S}_n$, and the point $\mathbf{x}$ can be expressed as

$$\mathbf{x} \;=\; \mathbf{s}_n \;+\; \sum_{k=n+1}^{\infty} \left(\frac{1}{k!} \prod_{j=1}^{k} \mathbf{w}_{jk} \right)\,.$$

We can estimate the distance from $\mathbf{x}$ to $\mathcal{S}_n$ as follows:

$$dist(\mathbf{x}, \mathcal{S}_n) \;\leq\; |\mathbf{x} - \mathbf{s}_n| \;\leq\; \sum_{k=n+1}^{\infty} \left| \frac{1}{k!} \prod_{j=1}^{k} \mathbf{w}_{jk} \right| \;\leq\; \sum_{k=n+1}^{\infty} \frac{C^k}{k!}\,,$$

the constant C being an upper bound on the norm of the elements in $\mathcal{A}$. The last expression in the above inequalities converges to 0 as $n \to \infty$. Hence, $\mathbf{x}$ is contained in the Painlevé–Kuratowski limit of $\{\mathcal{S}_n\}$, which is the monomial Minkowski exponential of $\mathcal{A}$. ∎

In order to elucidate the nature of the monomial Minkowski exponential of a given compact set $\mathcal{A}$, we focus on its boundary $\partial \exp_m(\mathcal{A})$. We note first that the monomial Minkowski exponential is a compact set whenever the given set $\mathcal{A}$ is compact. The fact that $\partial \exp_m(\mathcal{A})$ is a closed set follows directly from the definition of the Painlevé–Kuratowski limit, and its boundedness is easily deduced from condition (c) of Proposition 2. The following theorem gives a characterization of the boundary points of $\exp_m(\mathcal{A})$.

Theorem 3. *Suppose the complex number* $\mathbf{x}$ *lies on the boundary* $\partial \exp_m(\mathcal{A})$ *of the monomial Minkowski exponential of a compact set* $\mathcal{A}$ *in the complex plane. Then each point of the doubly–indexed sequence* $\{\mathbf{w}_{jk}\}$ *that generates* $\mathbf{x}$ *through (14) lies on the boundary* $\partial\mathcal{A}$ *of* $\mathcal{A}$*, provided that* $0 \notin \partial\mathcal{A}$*.*

Proof: Let $\{\mathbf{w}_{jk}\}$ be a doubly–indexed sequence generating the boundary point $\mathbf{x}$ by (14). When $0 \notin \partial\mathcal{A}$, the origin must lie in the interior or exterior of $\mathcal{A}$.

Assume first that 0 is in the *exterior* of $\mathcal{A}$. Now if an element $\mathbf{w}_{j_0k_0}$ of the sequence $\{\mathbf{w}_{jk}\}$ lies in the interior of $\mathcal{A}$, we can choose a positive real number ϵ such that the complex number $\mathbf{w}_{j_0k_0} + t\,\mathrm{e}^{\mathrm{i}\,\theta}$ is contained in $\mathcal{A}$ for all $t \in [\,0, \epsilon\,]$ and $\theta \in [\,0, 2\pi)$. Now let $\{\tilde{\mathbf{w}}_{jk}\}$ be the doubly–indexed sequence obtained from $\{\mathbf{w}_{jk}\}$ by replacing $\mathbf{w}_{j_0k_0}$ by $\mathbf{w}_{j_0k_0} + t\,\mathrm{e}^{\mathrm{i}\,\theta}$. Then the monomial Minkowski exponential $\exp_m(\mathcal{A})$ contains the complex number

$$\tilde{\mathbf{x}} \;=\; \sum_{k=0}^{\infty} \left(\frac{1}{k!} \prod_{j=1}^{k} \tilde{\mathbf{w}}_{jk} \right) . \tag{15}$$

This complex number $\tilde{\mathbf{x}}$ can be expressed as

$$\tilde{\mathbf{x}} \;=\; \mathbf{x} \,+\, t\,\mathrm{e}^{\mathrm{i}\,\theta} \frac{1}{k_0!} \prod_{\substack{1 \le j \le k_0 \\ j \neq j_0}} \mathbf{w}_{jk_0} \,.$$

Now the modulus

$$c \;=\; \frac{1}{k_0!} \prod_{\substack{1 \le j \le k_0 \\ j \neq j_0}} |\mathbf{w}_{jk_0}|$$

is a positive real number since $\mathcal{A}$ does not contain the origin. So $\tilde{\mathbf{x}}$ describes a circular disk centered at $\mathbf{x}$ of radius $c\epsilon$ as we vary t from 0 to ϵ for $\theta \in [\,0, 2\pi)$. But this contradicts the assumption that $\mathbf{x} \in \partial \exp_m(\mathcal{A})$.

We now assume that 0 is in the *interior* of $\mathcal{A}$. If none of the elements $\mathbf{w}_{jk}$ is zero, we can apply the same argument as in the preceding case. Otherwise, let k_0 be the smallest index such that

$$\prod_{j=1}^{k_0} \mathbf{w}_{jk_0} \;=\; 0 \,.$$

We choose a positive real number ϵ such that the disk centered at the origin of radius ϵ is contained in $\mathcal{A}$. Then we construct the doubly–indexed sequence $\{\tilde{\mathbf{w}}_{nk}\}$ from $\{\mathbf{w}_{jk}\}$ by replacing zeros in $\{\mathbf{w}_{1k_0}, \cdots, \mathbf{w}_{k_0k_0}\}$ with $t\,\mathrm{e}^{\mathrm{i}\,\theta}$ for $t \in [\,0, \epsilon\,]$ and $\theta \in [\,0, 2\pi)$. Then the complex numbers $\tilde{\mathbf{x}}$ given by (15) form a neighborhood of $\mathbf{x}$ as t varies in $[\,0, \epsilon\,]$. So we have a contradiction. ∎

Consider now the case where $\mathcal{A}$ is a circular disk in the complex plane with center $\mathbf{c}$ and radius r. In computing $\exp_m(\mathcal{A})$ it would be convenient if we could transform $\mathcal{A}$ to some "canonical" position. For the Minkowski sum of two disks, the canonical positions correspond to placing their centers at the origin through appropriate translations. For the Minkowski product of two disks, it is convenient to place their centers at the number 1 on the real axis through a complex scaling by the reciprocal of each center. For the Minkowski exponential of a disk, it seems natural to place the center at the origin rather than at the real number 1, and consider the translated disk

$$\tilde{\mathcal{A}} = \mathcal{A} - \mathbf{c} = \mathcal{A} \oplus \{-\mathbf{c}\} \,.$$

Unfortunately, there is no obvious relationship between $\exp_m(\mathcal{A})$ and $\exp_m(\tilde{\mathcal{A}})$ — in general, we have

$$\exp_m(\mathcal{A}) \;\neq\; \{\mathrm{e}^{\mathbf{c}}\} \otimes \exp_m(\tilde{\mathcal{A}})\,, \tag{16}$$

as already noted in Section 3 for the case of real intervals.

Remark 4. When $\mathcal{A}$ is a circular disk with center 0, computing $\exp_m(\mathcal{A})$ is a trivial task. Let $\mathcal{A}$ be the closed disk $D(0,r)$ with $r > 0$. The k–th Minkowski power $\otimes^k\mathcal{A}$ then becomes the circular disk $D(0,r^k)$, and the k–th monomial term $\mathcal{M}_k$ is $D(0,r^k/k!)$. Since the Minkowski sum of a finite number of circular disks centered at 0 is again a circular disk centered at 0, whose radius equals the sum of the individual radii, the partial sum $\mathcal{S}_n$ in (13) is given by

$$\mathcal{S}_n \;=\; \{1\} \oplus \bigoplus_{k=1}^{n} D\left(0, \frac{r^k}{k!}\right) \;=\; D\left(1, \sum_{k=1}^{n} \frac{r^k}{k!}\right)\,.$$

So $\{\mathcal{S}_n\}$ is a sequence of disks with common center 1 and strictly increasing radii. Hence, the monomial Minkowski exponential $\exp_m(\mathcal{A})$ is also a disk centered at 1, whose radius is the limit radius of $\mathcal{S}_n$. Hence, we conclude that

$$\exp_m(D(0,r)) \;=\; D(1, \mathrm{e}^r - 1)\,.$$

The relation (16) is a special instance of the more general inequality

$$\exp_m(\mathcal{A} \oplus \mathcal{B}) \;\neq\; \exp_m(\mathcal{A}) \otimes \exp_m(\mathcal{B})\,, \tag{17}$$

corresponding to a choice of the singleton set $\{\mathbf{c}\}$ for $\mathcal{B}$. It may be possible to derive inclusion relations between $\exp_m(\mathcal{A} \oplus \mathcal{B})$ and $\exp_m(\mathcal{A}) \otimes \exp_m(\mathcal{B})$. The latter expression corresponds to

$$\exp_m(\mathcal{A}) \otimes \exp_m(\mathcal{B}) \;=\; \left[\bigoplus_{k=0}^{\infty} \frac{1}{k!} \otimes^k \mathcal{A}\right] \otimes \left[\bigoplus_{k=0}^{\infty} \frac{1}{k!} \otimes^k \mathcal{B}\right]\,,$$

and by the definition of the monomial Minkowski exponential, we have

$$\exp_m(\mathcal{A} \oplus \mathcal{B}) \;=\; \bigoplus_{k=0}^{\infty} \frac{1}{k!} \otimes^k (\mathcal{A} \oplus \mathcal{B})\,.$$

Now using the sub–distributive law (2) one can show that

$$\otimes^2(\mathcal{A} \oplus \mathcal{B}) \;\subseteq\; (\otimes^2\mathcal{A}) \;\oplus\; 2(\mathcal{A} \otimes \mathcal{B}) \;\oplus\; (\otimes^2\mathcal{B})\,,$$

and more generally, for each $k \geq 2$,

$$\otimes^k(\mathcal{A} \oplus \mathcal{B}) \;\subseteq\; \bigoplus_{j=0}^{k} \binom{k}{j} (\otimes^{k-j}\mathcal{A}) \;\otimes\; (\otimes^j\mathcal{B})\,.$$

Substituting into the previous expression, we obtain

$$\exp_m(\mathcal{A} \oplus \mathcal{B}) \subseteq \bigoplus_{k=0}^{\infty} \frac{1}{k!} \left[\bigoplus_{j=0}^{k} \binom{k}{j} (\otimes^{k-j}\mathcal{A}) \otimes (\otimes^{j}\mathcal{B}) \right] .$$

Written out explicitly, the right–hand side is

$$\exp_m(\mathcal{A} \oplus \mathcal{B}) \subseteq \{1\} \oplus (\mathcal{A} \oplus \mathcal{B}) \oplus \frac{1}{2!}\left[(\otimes^2\mathcal{A}) \oplus 2(\mathcal{A} \otimes \mathcal{B}) \oplus (\otimes^2\mathcal{B}) \right]$$
$$\oplus \frac{1}{3!}\left[(\otimes^3\mathcal{A}) \oplus 3((\otimes^2\mathcal{A}) \otimes \mathcal{B}) \oplus 3(\mathcal{A} \otimes (\otimes^2\mathcal{B})) \oplus (\otimes^3\mathcal{B}) \right] \oplus \cdots$$

6 Convergence of Partial–Sum Approximations

In most cases, computing the exact boundary of $\exp_m(\mathcal{A})$ is a very difficult task — even for a simple domain $\mathcal{A}$. However, if we allow a small tolerance, the partial sum $\mathcal{S}_n$ can be a good approximation to $\exp_m(\mathcal{A})$ for sufficiently large n. For such approximations, we need to know how fast $\mathcal{S}_n$ converges to $\exp_m(\mathcal{A})$. The most common measure of the proximity of two sets is the *Hausdorff distance* [21]. The Hausdorff distance $d_H(X, Y)$ between two compact sets X and Y in a metric space is defined by

$$d_H(X, Y) = \max\{\sup_{\mathbf{x}\in X} \inf_{\mathbf{y}\in Y} dist(\mathbf{x}, \mathbf{y}), \sup_{\mathbf{y}\in Y} \inf_{\mathbf{x}\in X} dist(\mathbf{x}, \mathbf{y})\} ,$$

where $dist(\,,\,)$ is the distance in the metric space. The following proposition gives an estimate of the Hausdorff distance between $\exp_m(\mathcal{A})$ and the partial sum $\mathcal{S}_n$.

Proposition 5. *For a compact simply–connected domain $\mathcal{A}$ in the complex plane, let C be the maximum modulus of all points in $\mathcal{A}$. Then the Hausdorff distance between* $\exp_m(\mathcal{A})$ *and the partial sum $\mathcal{S}_n$ is bounded by*

$$d_H(\exp_m(\mathcal{A}), \mathcal{S}_n) < \frac{C^{n+1}}{(n+1)!} \, \mathrm{e}^{C} . \tag{18}$$

Proof: For any point $\mathbf{x}$ in $\exp_m(\mathcal{A})$, let $\{\mathbf{w}_{jk}\}$ be the doubly–indexed sequence of points in $\mathcal{A}$ such that

$$\mathbf{x} = \sum_{k=0}^{\infty} \left(\frac{1}{k!} \prod_{j=1}^{k} \mathbf{w}_{jk} \right) .$$

We separate the above summation into $\mathbf{s}_n$ and $\mathbf{r}_n$, where

$$\mathbf{s}_n = \sum_{k=0}^{n} \left(\frac{1}{k!} \prod_{j=1}^{k} \mathbf{w}_{jk} \right) , \qquad \mathbf{r}_n = \sum_{k=n+1}^{\infty} \left(\frac{1}{k!} \prod_{j=1}^{k} \mathbf{w}_{jk} \right) .$$

Then $\mathbf{s}_n$ is contained in $\mathcal{S}_n$. Thus, we have

$$\inf_{\mathbf{y}\in\mathcal{S}_n} dist(\mathbf{x},\mathbf{y}) \le dist(\mathbf{x},\mathbf{s}_n) \;=\; |\mathbf{r}_n|\,.$$

Since $\mathbf{x}$ is an arbitrary point in $\exp_m(\mathcal{A})$, we also have

$$\sup_{\mathbf{x}\in\exp_m(\mathcal{A})} \inf_{\mathbf{y}\in\mathcal{S}_n} dist(\mathbf{x},\mathbf{y}) \;\le\; |\mathbf{r}_n|\,.$$

On the other hand, for any point $\mathbf{s}_n$ in $\mathcal{S}_n$, we can construct a point $\mathbf{x}$ in $\exp_m(\mathcal{A})$ with $\mathbf{x} = \mathbf{s}_n + \mathbf{r}_n$ by choosing arbitrary numbers $\mathbf{w}_{jk}$ in $\mathcal{A}$ for $k \ge n+1$. Thus, we can derive

$$\sup_{\mathbf{y}\in\mathcal{S}_n} \inf_{\mathbf{x}\in\exp_m(\mathcal{A})} dist(\mathbf{x},\mathbf{y}) \;\le\; |\mathbf{r}_n|\,.$$

Therefore, the Hausdorff distance $d_H(\exp_m(\mathcal{A}),\mathcal{S}_n)$ is bounded by the modulus of the remainder term $\mathbf{r}_n$, which can be estimated as

$$|\mathbf{r}_n| \;\le\; \sum_{k=n+1}^{\infty} \left(\frac{1}{k!}C^k\right) \;\le\; \frac{C^{n+1}}{(n+1)!}\,\mathrm{e}^C\,.$$

This completes the proof. ∎

7 Monte Carlo Experiments

Since a closed–form derivation seems quite difficult, we consider here the use of Monte Carlo experiments for evaluating the monomial Minkowski exponential of a special circular disk. Let $\mathcal{A}$ be the circular disk of radius r centered at 1, which is the canonical position for the Minkowski product [14,15]. We further assume $\mathcal{A}$ does not contain the origin — i.e., $r < 1$. The maximum modulus of all points in $\mathcal{A}$ is therefore less than 2.

To compute the monomial Minkowski exponential $\exp_m(\mathcal{A})$ with a tolerance 10^{-3}, it suffices to evaluate the partial sum $\mathcal{S}_n$ with $n = 10$, since from (18) the Hausdorff distance between $\exp_m(\mathcal{A})$ and $\mathcal{S}_{10}$ is bounded by

$$d_H(\exp_m(\mathcal{A}),\mathcal{S}_{10}) \;<\; \frac{2^{11}}{11!}\,\mathrm{e}^2 \;\approx\; 3.79\times 10^{-4}\,.$$

Now to generate a sampling of points in the partial sum $\mathcal{S}_n$ by the Monte Carlo method, we need to evaluate the expression

$$1 + \sum_{k=1}^{n} \frac{1}{k!} \prod_{j=1}^{k} \mathbf{z}_{jk}$$

for many different sets of randomly–chosen points $\mathbf{z}_{jk} \in \mathcal{A}$. We are interested mainly in the boundary of $\mathcal{S}_n$, and a necessary condition for the above expression to yield a point on this boundary is that all the points $\mathbf{z}_{jk}$ are selected from the

boundary of $\mathcal{A}$. The plot on the left in Figure 7 shows the point cluster generated by the Monte Carlo method in this manner, for the partial sum $\mathcal{S}_{10}$ when $\mathcal{A}$ is the circular disk centered at 1 of radius 0.9. However, it is not easy to discern the boundary of $\mathcal{S}_{10}$ from this Monte Carlo result, since the point cluster is very sparse near the boundary. In order to gain a better impression of the boundary, we need a more structured method for selecting the points $\mathbf{z}_{jk}$ from $\partial\mathcal{A}$.

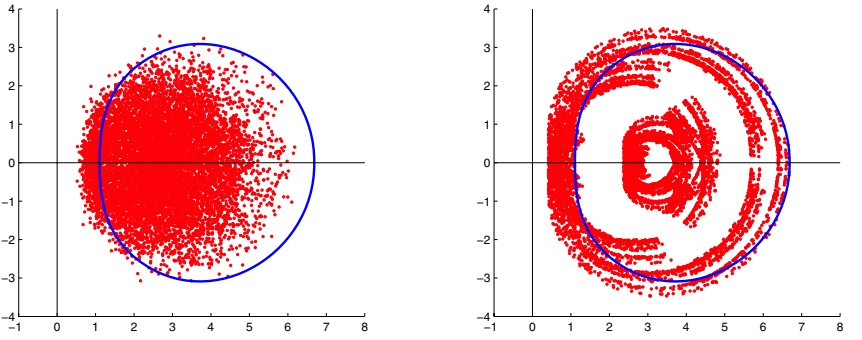


Fig. 7. Left: Unstructured Monte Carlo experiment for the partial sum $\mathcal{S}_{10}$ when $\mathcal{A}$ is the circular disk of radius 0.9 centered at 1. The closed curve is the boundary of the exponential image of $\mathcal{A}$. Right: Result of structured Monte Carlo simulation for $\mathcal{S}_{10}$.

The partial sum $\mathcal{S}_n$ is obtained by translating the successive Minkowski sum of n monomial terms from $\mathcal{M}_1$ to $\mathcal{M}_n$ by 1. Thus, any boundary point of $\mathcal{S}_n$ is generated by

$$1 + \sum_{k=1}^{n} \frac{1}{k!}\mathbf{w}_k \,, \tag{19}$$

where $\mathbf{w}_k$ is a boundary point of the k–th Minkowski power $\otimes^k\mathcal{A}$. It is known [15] that the boundary of the Minkowski power of a circular disk is a subset of the image of $\partial\mathcal{A}$ under the conformal map $\mathbf{z} \mapsto \mathbf{z}^k$. In other words, the boundary point $\mathbf{w}_k$ of $\otimes^k\mathcal{A}$ must be the k–th power of a single point on $\partial\mathcal{A}$. Therefore, a superset of $\partial(\otimes^k\mathcal{A})$ can be parameterized by

$$\mathbf{w}_k(\theta_k) \;=\; (1 + r\mathrm{e}^{\mathrm{i}\theta_k})^k \,.$$

From the above observation, a more restrictive necessary condition for $\mathbf{w}_k$ to generate a boundary point of $\partial\mathcal{S}_n$ can be formulated as follows. Since the partial sum $\mathcal{S}_n$ is essentially the Minkowski sum of n monomial terms, the boundary point is generated by the n–tuple of points $\{\mathbf{w}_1, \cdots, \mathbf{w}_n\}$ where the Gauss maps of $\partial(\otimes^k\mathcal{A})$ at $\mathbf{w}_k$ are matched. Hence, the outward normal vectors of the curves $\mathbf{w}_k(\theta)$, given by

$$-\,\mathrm{i}\,\mathbf{w}_k'(\theta_k) \;=\; k\,r\,\mathrm{e}^{\mathrm{i}\theta_k}(1 + r\mathrm{e}^{\mathrm{i}\theta_k})^{k-1} \,,$$

should have the same argument. Therefore, for a freely–chosen θ_1, the angles θ_k for $k \geq 2$ should satisfy

$$\theta_k + (k-1)\arg(1 + re^{i\theta_k}) \;=\; \theta_1 + 2m\pi$$

for some integer m. The above equation can be re–formulated as

$$\frac{r\sin\theta_k}{1 + r\cos\theta_k} \;=\; \tan\left(\frac{\theta_1 - \theta_k + 2m\pi}{k-1}\right).$$

By re–arranging, after multiplying both sides by $1 + r\cos\theta_k$, we obtain

$$r\left[\sin\theta_k - \cos\theta_k \tan\left(\frac{\theta_1 - \theta_k + 2m\pi}{k-1}\right)\right] \;=\; \tan\left(\frac{\theta_1 - \theta_k + 2m\pi}{k-1}\right).$$

This can be simplified by multiplying both sides with $\cos(\theta_1 - \theta_k + 2m\pi)/(k-1)$, yielding

$$-r\,\sin\left(\frac{\theta_1 - k\theta_k + 2m\pi}{k-1}\right) \;=\; \sin\left(\frac{\theta_1 - \theta_k + 2m\pi}{k-1}\right). \tag{20}$$

Although this equation does not admit an explicit closed–form solution for θ_k, in the context of the Monte Carlo experiment we can apply an iterative numerical procedure, such as the Newton–Raphson method. The overall scheme for the structured Monte Carlo method using equation (20) is as follows:

1. choose an arbitrary angle θ_1;
2. solve equation (20) numerically for $2 \leq k \leq n$;
3. generate a point on (the superset of) $\partial\mathcal{S}_n$ using equation (19).

The result of the structured Monte Carlo simulation is illustrated on the right in Fig. 7. The structured Monte Carlo simulation clearly gives a better impression of the monomial Minkowski exponential than the unstructured method.

8 Closure

Some preliminary results concerning the problem of computing the Minkowski exponential of a circular disk in the complex plane have been discussed. Two types of set exponentials were considered, based upon the monomial and Horner evaluation schemes. A basic difficulty, compared to computation of Minkowski sums, products, powers, and roots of circular disks, is the lack of a universal transformation of the circular disk operand to a "simple" canonical configuration. For the simpler case of the Minkowski exponentials of real intervals, complete closed–form expressions were derived in the monomial case, but the Horner case proved more difficult, and closed–form expressions were derived only for cases (1) and (2) of the four categories (11). For disks in the complex plane, Monte Carlo experiments were used to gain insight into basic features of the monomial Minkowski exponential, but the analysis and algorithmic boundary evaluation of Minkowski exponentials remain challenging open problems.

Acknowledgement

H. P. Moon was supported by the BK21 project of the Ministry of Education, Korea.

References

1. Arnold, L.: Stochastic Differential Equations: Theory and Applications. Wiley, New York (1974)
2. Cabot, A., Seeger, A.: Multivalued exponential analysis. Part I: Maclaurin exponentials. Set–Valued Analysis 14, 347–379 (2006)
3. Farouki, R.T., Chastang, J.–C.A.: Curves and surfaces in geometrical optics. In: Lyche, T., Schumaker, L.L. (eds.) Mathematical Methods in Computer Aided Geometric Design II, pp. 239–260. Academic Press, London (1992)
4. Farouki, R.T., Chastang, J.–C.A.: Exact equations of "simple" wavefronts. Optik 91, 109–121 (1992)
5. Farouki, R.T., Gu, W., Moon, H.P.: Minkowski roots of complex sets. In: Geometric Modeling and Processing 2000, pp. 287–300. IEEE Computer Society Press, Los Alamitos (2000)
6. Farouki, R.T., Han, C.Y.: Computation of Minkowski values of polynomials over complex sets. Numerical Algorithms 36, 13–29 (2004)
7. Farouki, R.T., Han, C.Y.: Solution of elementary equations in the Minkowski geometric algebra of complex sets. Advances in Computational Mathematics 22, 301–323 (2005)
8. Farouki, R.T., Han, C.Y.: Robust plotting of generalized lemniscates. Applied Numerical Mathematics 51, 257–272 (2005)
9. Farouki, R.T., Han, C.Y.: Root neighborhoods, generalized lemniscates, and robust stability of dynamic systems. Applicable Algebra in Engineering, Communication, and Computing 18, 169–189 (2007)
10. Farouki, R.T., Han, C.Y., Hass, J.: Boundary evaluation algorithms for Minkowski combinations of complex sets using topological analysis of implicit curves. Numerical Algorithms 40, 251–283 (2007)
11. Farouki, R.T., Hass, J.: Evaluating the boundary and covering degree of planar Minkowski sums and other geometrical convolutions. Journal of Computational and Applied Mathematics 209, 246–266 (2007)
12. Farouki, R.T., Moon, H.P.: Minkowski geometric algebra and the stability of characteristic polynomials. In: Hege, H.–C., Polthier, K. (eds.) Visualization in Mathematics 3, pp. 163–188. Springer, Heidelberg (2003)
13. Farouki, R.T., Moon, H.P., Ravani, B.: Algorithms for Minkowski products and implicitly–defined complex sets. Advances in Computational Mathematics 13, 199–229 (2000)
14. Farouki, R.T., Moon, H.P., Ravani, B.: Minkowski geometric algebra of complex sets. Geometriae Dedicata 85, 283–315 (2001)
15. Farouki, R.T., Pottmann, H.: Exact Minkowski products of N complex disks. Reliable Computing 8, 43–66 (2002)
16. Friedman, A.: Stochastic Differential Equations and Applications. Academic Press, New York (1975)
17. Gihman, I.I., Skorohod, A.V.: Stochastic Differential Equations (translated by K. Wickwire). Springer, Berlin (1972)

18. Ghosh, P.K.: A mathematical model for shape description using Minkowski operators. Computer Vision, Graphics, and Image Processing 44, 239–269 (1988)
19. Ghosh, P.K.: A unified computational framework for Minkowski operations. Computers & Graphics 17, 357–378 (1993)
20. Hadwiger, H.: Vorlesungen über Inhalt, Oberfläche, und Isoperimetrie. Springer, Berlin (1957)
21. Hausdorff, F.: Set Theory (translated by J. R. Aumann et al.), Chelsea, New York (1957)
22. Kaul, A.: Computing Minkowski sums, PhD Thesis, Columbia University (1993)
23. Kaul, A., Farouki, R.T.: Computing Minkowski sums of plane curves. International Journal of Computational Geometry and Applications 5, 413–432 (1995)
24. Kaul, A., Rossignac, J.R.: Solid interpolating deformations: Construction and animation of PIP. Computers and Graphics 16, 107–115 (1992)
25. Lozano–Pérez, T., Wesley, M.A.: An algorithm for planning collision–free paths among polyhedral obstacles. Communications of the ACM 22, 560–570 (1979)
26. Matheron, G.: Random Sets and Integral Geometry. Wiley, New York (1975)
27. Middleditch, A.E.: Applications of a vector sum operator. Computer Aided Design 20, 183–188 (1988)
28. Minkowski, H.: Volumen und Oberfläche. Mathematische Annalen 57, 447–495 (1903)
29. Moore, R.E.: Interval Analysis. Prentice–Hall, Englewood Cliffs (1966)
30. Moore, R.E.: Methods and Applications of Interval Analysis. SIAM, Philadelphia (1979)
31. Øksendal, B.K.: Stochastic Differential Equations: An Introduction with Applications. Springer, Berlin (1998)
32. Serra, J.: Image Analysis and Mathematical Morphology. Academic Press, London (1982)
33. Serra, J.: Introduction to mathematical morphology. Computer Vision, Graphics, and Image Processing 35, 283–305 (1986)
34. Tomiczková, S.: Minkowski Operations and Their Applications, PhD Thesis, Plzeň, Czech Republic (2006)

Unconstrained Parametric Minimization of a Polynomial: Approximate and Exact

S. Liang and D.J. Jeffrey

Department of Applied Mathematics,
The University of Western Ontario, London, Ont. N6A 5B7 Canada

Abstract. We consider a monic polynomial of even degree with symbolic coefficients. We give a method for obtaining an expression in the coefficients (regarded as parameters) that is a lower bound on the value of the polynomial, or in other words a lower bound on the minimum of the polynomial. The main advantage of accepting a bound on the minimum, in contrast to an expression for the exact minimum, is that the algebraic form of the result can be kept relatively simple. Any exact result for a minimum will necessarily require parametric representations of algebraic numbers, whereas the bounds given here are much simpler. In principle, the method given here could be used to find the exact minimum, but only for low degree polynomials is this feasible; we illustrate this for a quartic polynomial. As an application, we compute rectifying transformations for integrals of trigonometric functions. The transformations require the construction of polynomials that are positive definite.

1 Introduction

Let $n \in \mathbb{Z}$ be even, and let $P_n \in \mathbb{R}[a_0, \ldots, a_{n-1}][x]$ be monic in x, that is,

$$P_n(x) = x^n + \sum_{j=0}^{n-1} a_j x^j \ . \tag{1}$$

A function $L(a_j)$ of the coefficients is required that is a lower bound for $P_n(x)$, i.e., L must satisfy

$$(\forall x) P_n(x) \geq L(a_j) \ . \tag{2}$$

The problem definition does not require that the equality in (2) be realized. If that is also the case, then L is the minimum of P_n:

$$\min_{x \in R} P_n(x) = L_{\min}(a_j) \ . \tag{3}$$

Thus $L_{\min}$ obeys

$$(\forall x) P_n(x) \geq L_{\min}(a_j) \ , \qquad (\exists x) P_n(x) = L_{\min}(a_j) \ , \tag{4}$$

and $L_{\min} \geq L$, where L satisfies (2).

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 22–31, 2008.
© Springer-Verlag Berlin Heidelberg 2008

The problem described has connections to several areas of research, including parametric optimization, quantifier elimination and polynomial positive-definiteness. Much of the work on parametric optimization concerns topics such as the continuity of the optimum as a function of the parameters, or the performance of numerical methods; see, for example, [1, 2, 4]. The following problem was considered in [1].

$$\min\{\lambda^2 x^2 - 2\lambda(1-\lambda)x \mid x \geq 0\} \ .$$

The unique solution for the unconstrained problem is found for $\lambda \neq 0$ to be $-(1-\lambda)^2$, which is realised when x takes the value $\hat{x} = (1-\lambda)/\lambda$. The constrained problem has this solution only for $\lambda \in (0, 1)$ and ceases to be smooth at the end points. The unconstrained problem is covered in this paper, although the focus is on higher degree polynomials.

There has been a large amount of work on a related problem in quantifier elimination. For $n = 4$, Lazard [7] and Hong [5] have solved the following problem. Find a condition on the coefficients p, q, r that is equivalent to the statement

$$(\forall x) x^4 + px^2 + qx + r \geq 0 \ . \tag{5}$$

The solution they found is

$$\Big[[256r^3 - 128p^2r^2 + 144pq^2r + 16p^4r - 27q^4 - 4p^3q^2 \geq 0 \wedge 8pr - 9q^2 - 2p^3 \leq 0]$$

$$\bigvee [27q^2 + 8p^3 \geq 0 \wedge 8pr - 9q^2 - 2p^3 \geq 0]\Big] \bigwedge r \geq 0 \ . \tag{6}$$

It is clear that a solution of (3) gives a solution of this problem, since (5) is equivalent to the statement $r = -\min(x^4 + px^2 + qx \,|\, x)$. The question of positive definiteness is also related to the current problem. Ulrich and Watson [8] studied this problem for a quartic polynomial, except that they included the constraint $x \in R^+$, the positive real line.

Previous work has all been directed towards calculations of the minimum of a given polynomial. For $n = 2$, the minimum of a quadratic polynomial is a standard result.

$$\min_{x \in \mathbb{R}} P_2(x) = \min(x^2 + a_1 x + a_0) = a_0 - \tfrac{1}{4} a_1^2 \ , \tag{7}$$

and this is attained when $x = -\frac{1}{2} a_1$. For larger n, there is only the standard calculus approach, which uses the roots of the derivative. This, however, is only possible for numerical coefficients, because there is no way of knowing which root corresponds to the minimum. Floating-point approximations to the minimum are easily obtained.

If all of the coefficients of P_{2n} are purely numerical, rather than symbolic, then there are many ways to find the minimum. For example, MAPLE has the command `minimize` and the command `Optimize:-Minimize`. An example is

```
>minimize( x^4 - 5*x^2 + 4*x ,x);
 RootOf(2 _Z^3 - 5 _Z - 2,index=3 )^4
    - 5*RootOf(2 _Z^3 - 5 _Z - 2,index=3 )^2
    + 4 RootOf(2 _Z^3 - 5 _Z - 2,index=3 )
```

which can be simplified by MAPLE to

```
-(5/2) RootOf(2 _Z^3-5 _Z-2,index=3)^2
- 3 RootOf(2 _Z^3-5 _Z-2,index=3)
```

The second argument of `RootOf` selects, using an index, the appropriate root of the polynomial.

2 Algorithm for Lower Bound

We now describe a recursive algorithm. In principle, it could be used to find the minimum of a parametric polynomial, and indeed we show this below for a quartic polynomial, but the main intended use is for a simpler lower bound.

Consider a polynomial given by (1). We shall express the lower bound to P_n in terms of that for P_{n-2}. This recursive descent terminates at P_2, for which we have the result (7). The descent is based on the following obvious lemma.

Lemma 1. If $f(x)$ and $g(x)$ are two even-degree monic polynomials, then

$$\inf(f(x)+g(x)) \geq \inf f(x) + \inf g(x).$$

Proof: The equality holds when the minima of f and g are realized at the same critical point x. □

It is convenient at this point to acknowledge the evenness of the degree by changing notation to consider P_{2n}. We apply the lemma by using the standard transformation $x = y - a_{2n-1}/(2na_{2n})$ to remove the term in x^{2n-1} from $P_{2n}(x)$. Thus we have the depressed polynomial

$$P_{2n}(y) = a_{2n}y^{2n} + \sum_{j=0}^{2n-2} b_j y^j. \tag{8}$$

Now, we split P_{2n} into two even-degree polynomials with positive leading coefficients by introducing a parameter k_n satisfying $k_n > 0$ and $k_n > b_{2n-2}$.

$$P_{2n} = [a_{2n}y^{2n} + (b_{2n-2} - k_n)y^{2n-2}] + [k_n y^{2n-2} + \ldots] = P_{2n}^{(1)} + P_{2n}^{(2)} \ .$$

The minimum of $P_{2n}^{(1)}$ is

$$\inf(P_{2n}^{(1)}) = -\frac{(n-1)^{n-1}(k_n - b_{2n-2})^n}{n^n a_{2n}^{n-1}}$$

which is obtained at the critical points $y^2 = (n-1)(k_n - b_{2n-2})/(na_{2n})$.

Since $\deg P_{2n}^{(2)} = 2n-2 < 2n$, we can recursively compute the minimum and critical point of $P_{2n}^{(2)}$. Let the minimum and the corresponding critical point of $P_{2n}^{(2)}$ be $M(k_{n-1},\ldots,k_2), N(k_{n-1},\ldots,k_2)$ respectively. Then by Lemma 1, we have

$$\inf(P_{2n}) \geq -\frac{(n-1)^{n-1}(k_n - b_{2n-2})^n}{n^n a_{2n}^{n-1}} + M.$$

Therefore, a lower bound for P_{2n} is obtained after recursion in terms of parameters $k_n, k_{n-1},\ldots,k_2$. If it is possible to choose the k_i such that

$$\frac{(n-1)(k_n - b_{2n-2})}{n a_{2n}} = N(k_{i-1},\ldots,k_2)^2 \ , \tag{9}$$

at each recursive step, then an expression for the minimum would be obtained. However, our main aim is to find lower bounds in as simple a form as possible, hence we choose each k_i to satisfy the requirements in a simple way.

Since the k_i will appear in the denominators of expressions, it is not a good idea to allow a value that is too small. Otherwise there will be computational difficulties. A simple choice is $k_i = 1$, but this may not satisfy $1 > b_{2i-2}$. Therefore we have chosen to use

$$k_i = \max(1, 1 + b_{2i-2}) \ .$$

Table 1. A Maple procedure for computing a lower bound on the value of an even-degree monic polynomial

```
BoundPoly:=proc(p,var)
local m,n,a,b,c,redpoly,y,p1,p2,tp,par:
# Input: An even degree (parametric) polynomial p(var).
# Output: a lower bound.
m:=degree(p,var):
if m=0 or modp(m,2)<>0 then error("Bad input") end if;
  a:=coeff(p,var,m):
  b:=coeff(p,var,m-1):
  c:=coeff(p,var,m-2):
  if m=2 then
    (4*a*c-b^2)/(4*a):
  else
    n:=m/2:
    redpoly:=expand(subs(var=y-b/(m*a),p)):
    b:=coeff(redpoly,y,m-2):
    par:=max(1,b+1):
    p1:=a*y^m+(b-par)*y^(m-2):
    p2:=expand(redpoly-p1):
    tp:=(n-1)^(n-1)*(par-b)^n/(n^n*a^(n-1)):
    simplify(-tp+BoundPoly(p2,y)):
  end if
end proc:
```

This has the advantage that the simple value 1 will be selected whenever possible, and otherwise the more complicated value is used. Several other choices were tried, for example, $k_i = 1 + |b_{2i-2}|$. In either case, the results are much simpler if MAPLE is able to determine the sign of b_{2i-2}, otherwise many unsimplified expressions can appear in the output. The first choice gives the following algorithm, which is presented in MAPLE syntax in table 1.

3 Examples

Consider the polynomial

$$p = x^6 + x^4 - 2x^3 + x^2 - ax + 2 \ . \tag{10}$$

Applying the algorithm, we obtain

$$30299/17280 - (3/20)a - (1/5)a^2 \ . \tag{11}$$

Using a numerical routine, we can choose varying values of a and compute the numerical minimum and then plot this against the bound just obtained. This is shown in figure 1.

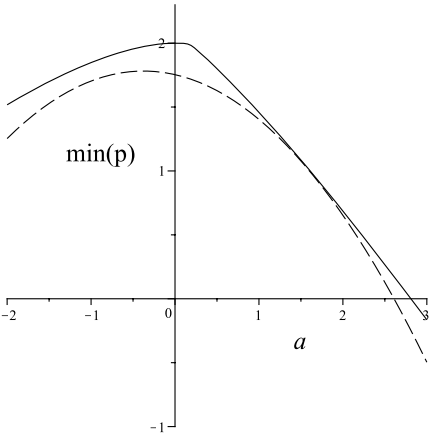


Fig. 1. The minimum of the polynomial $p(x)$ defined in (10) and the lower bound given in (11). The solid line is the exact minimum. Although very close, the two curves never touch.

For different values of a, this example shows both very close bounds and very poor ones. Thus for the case $a = 1.5$, the lower bound on the minimum is 1.078, whereas the true minimum is 1.085. In contrast, for large a, the exact minimum is asymptotically $-5(a/6)^{6/5}$, whereas the bound is $-a^2/5$, so the bound can

be arbitrarily bad in that case. However, as shown in the next section, in the intended application, there is no need for a close bound; any bound will be just as good.

A second example shows a different form of output. We assume the condition $a > 0$ and look for a lower bound on

$$p = x^6 + x^4 - 2x^3 + (1 + a)x^2 - x + 2 \ . \tag{12}$$

With the MAPLE assumption `assume(a,positive)`, we obtain the bound

$$\frac{24251 + 24628a}{3456(5 + 4a)} \ .$$

Notice that since $a > 0$, the denominator is never zero. We can quickly check the accuracy of this bound by trying a numerical comparison. Thus for $a = 10$, the bound takes the value $30059/17280 \approx 1.7395$, while the minimum value is actually 1.9771. For large, positive, a the minimum is asymptotically 2 and the bound is asymptotically $6157/3456 \approx 1.78$, so in this case the asymptotic behaviour is good.

4 The Minimum of a Quartic Polynomial

Although the main implementation aims for a simple lower bound, it has already been stated that the same approach can be used to find an minimum. We show that this is so, but also show the more complicated form of the result, by deriving an exact minimum for a quartic polynomial. As above, we need consider only a depressed quartic.

Theorem 1: If the coefficient $b_1 \neq 0$, the quartic polynomial

$$P_4(x) = x^4 + b_2x^2 + b_1x$$

has the minimum

$$\inf P_4 = b_2k_2 - \tfrac{3}{4}k_2^2 - \tfrac{1}{4}b_2^2 \ , \tag{13}$$

where

$$k_2 = s^{1/3} + \frac{b_2^2}{9s^{1/3}} + \frac{b_2}{3} \ , \tag{14}$$

$$s = \tfrac{1}{4}b_1^2 + \tfrac{1}{27}b_2^3 + \tfrac{1}{36}\sqrt{81b_1^4 + 24b_1^2b_2^3} \ . \tag{15}$$

Moreover, the minimum of P_4 is located at $x = x_{\mathrm{m}} = -\frac{1}{2}b_1/k_2$.

Proof: As above, the polynomial P_4 is split into two by introducing a parameter k_2 satisfying $k_2 > 0$ and $k_2 > b_2$.

$$P_4 = [y^4 + (b_2 - k_2)y^2] + [k_2y^2 + b_1y] = P_4^{(1)} + P_4^{(2)} \ .$$

The minimum of $P_4^{(1)}$ is

$$\inf(y^4 + (b_2 - k_2)y^2) = -\tfrac{1}{4}(k_2 - b_2)^2 \ ,$$

given the restrictions on k_2. The coordinate of this minimum obeys $y^2 = \frac{1}{2}(k_2 - b_2)$. The minimum of $P_4^{(2)}$ is $-b_1^2/2$, by (7), and therefore

$$\inf(P_4) \geq -b_1^2/(4k_2) - \tfrac{1}{4}(k_2 - b_2)^2 \ . \tag{16}$$

Equating the coordinates of the two infima gives an equation for the value of k_2 at which the lower bound equals the minimum of P_4.

$$\frac{k_2 - b_2}{2} = \left(\frac{-b_1}{2k_2}\right)^2 \ .$$

This is equivalent to the cubic

$$k_2^3 - b_2 k_2^2 - \tfrac{1}{2}b_1^2 = 0 \ , \tag{17}$$

which equation can also be obtained by maximizing the right side of (16) directly. It is straightforward to show that (17) has a unique positive solution, and furthermore it is always greater than b_2, as was assumed at the start of the derivation. Rewriting (17) in the form

$$\tfrac{1}{2}k_2^2 - \tfrac{1}{2}b_2 k_2 = b1^2/(4k_2) \ ,$$

allows the expression (16) to be transformed into the form given in the theorem statement. □

Since (17) has a unique positive solution, its solution takes the form (14) given in the theorem. For some values of the coefficients, the quantity s will be complex, but if $s^{1/3}$ is always evaluated as the principal value, then k_2 given by (14) is the real positive solution.

Theorem 2: For the case $b_1 = 0$, the quartic polynomial

$$P_4(x) = x^4 + b_2 x^2 \tag{18}$$

has the minimum

$$\inf P_4 = -\max(0, -b_2/2)^2 \ ,$$

at the points $x_{\mathrm{m}}^2 = \max(0, -b_2/2)$.

Proof: By differentiation. □

Implementation. The discussion here is in the same spirit as the discussion in [3]. The following issues must be addressed by the implementer, taking into account the facilities available in the particular CAS.

For a polynomial with numerical coefficients in the rational-number field Q, the infima can be algebraic numbers of degrees 1, 2 or 3. If the formulae (13) and (14) are used for substitution, the answer will always appear to be an algebraic

number of degree 3, and the simplification of such numbers into lower degree forms cannot be relied on in some systems. Therefore, if it is accepted that the system should return the simplest expressions possible, then the best strategy in this case is not to use (14), but instead to solve the cubic equation (17) directly. Even if simplicity is not an issue, roundoff error in the Cardano formula often results in a small nonzero imaginary part in k_2.

For symbolic coefficients, the main problem is the specialization problem [3]. Since Theorem 1 excludes $b_1 = 0$, it is important to see what would happen if the formulae (13) and (14) were returned to a user and later the user substituted coefficients giving $b_1 = 0$. Substituting $b_1 = 0$ into (14) gives

$$k_2 = \tfrac{1}{3}(b_2^3)^{1/3} + b_2^2/3(b_2^3)^{1/3} + b_2/3$$

For $b_2 > 0$ this gives $k_2 = b_2$, while for $b_2 < 0$ it simplifies to $k_2 = 0$. For $b_2 = 0$, the system should report a divide by zero error. Thus for $b_2 \neq 0$, (13) and (14) work even for $b_1 = 0$, although it should be noted that the position of the minimum, $-b_1/2k_2$, will give a divide by zero error for all $b_2 < 0$. It is important to remember in this discussion that the mathematical properties of (13) and (14). Thus, the fact that it is possible to obtain the correct result for $b_1 = b_2 = 0$ by taking limits is not relevant; what is relevant is how a CAS will manipulate the expressions.

An alternative implementation can use the fact that some CAS have functions for representing one root of an equation directly. In particular, Maple has the `RootOf` construction, but in order to specify the root uniquely, an interval must be supplied that contains it. The left side of (17) is $-\frac{1}{2}b_1^2$ for $k = 0, b_2$ and hence the interval can start at $\max(0, b_2)$. By direct calculation, the left side is positive at $|b_2| + b_1^2/6 + 1$. An advantage of this approach is the fact that $b_1 = b_2 = 0$ is no longer an exceptional case, at least for the value of the minimum: the position still requires separate treatment.

5 Application to Integration

Let $\psi, \phi \in \mathbb{R}[x, y]$ be polynomials over $\mathbb{R}$, the field of real numbers. A rational trigonometric function over $\mathbb{R}$ is a function of the form

$$T(\sin z, \cos z) = \frac{\psi(\sin z, \cos z)}{\phi(\sin z, \cos z)} \ . \tag{19}$$

The problem considered here is the integration of such a function with respect to a real variable, in other words, to evaluate $\int T(\sin x, \cos x)\ \mathrm{d}x$ with $x \in \mathbb{R}$. The particular point of interest lies in the continuity properties of the expression obtained for the integral. General discussions of the existence of discontinuities in expressions for integrals have been given by [6] and [10].

A simple example shows the difficulty to be faced. The integral below was evaluated as shown by all the common computer algebra programs (MAPLE, MATHEMATICA and others); notice that the integral depends on a symbolic parameter a.

$$U(x) = \frac{(a\cos^4 x + 3\sin^2 x\cos^2 x)}{\cos^6 x + (a\sin x\cos^2 x + \sin^3 x)^2} \ , \tag{20}$$

$$\int U(x)\ \mathrm{d}x = \arctan(a\tan x + \tan^3 x) \ . \tag{21}$$

It is a simple calculation to see that the integrand $U(x)$ is continuous at $x = \pi/2$, with $U(0) = 0$, but the expression for the integral is discontinuous at the same point, having a jump of π. We have

$$\lim_{x\uparrow\pi/2} \arctan(a\tan x + \tan^3 x) - \lim_{x\downarrow\pi/2} \arctan(a\tan x + \tan^3 x) = \pi \ .$$

The notion of a rectifying transformation was introduced in [6], and can be applied to this situation.

The general problem is to rectify expressions of the form $\arctan[P(u)]$, where $P \in \mathbb{R}[u]$, and without loss of generality is monic. Moreover, $u = \tan x$, where x is chosen according to the properties of the integrand. We note first the identity

$$\arctan x - \arctan y = \arctan\frac{x-y}{1+xy} + \begin{cases} \mathrm{sgn}(y)\pi \ , & \text{for } 1+xy<0 \ , \\ 0 \ , & \text{otherwise.} \end{cases} \tag{22}$$

We shall use this in a formal sense, dropping the piecewise constant. The two cases of P of even degree and P of odd degree are treated separately. For P of even degree, we transform as follows.

$$\arctan P(u) \to \arctan P(u) - \arctan(1/k) \to \arctan\frac{P-1/k}{1+P/k} = \arctan\frac{kP-1}{k+P} \ .$$

The first step simply adds a constant to the result of the integration. The second step uses formula (22), dropping the piecewise constant. The final expression will now be continuous provided

$$\forall u \in \mathbb{R}, \quad P(u) + k > 0 \ .$$

The problem, therefore, is to choose k so that this condition is satisfied. Notice that since $P(u)$ is even degree and monic, it will always be possible to satisfy the condition, and the problem is to find an expression for k. Also note that in the example, P contains a parameter, so a simple calculus exercise will not be sufficient to determine k.

For P of odd degree, we transform as follows.

$$\begin{aligned} \arctan(P(u)) &\to \arctan(P(u)) - \arctan u/k + \arctan u/k - \arctan u + x \\ &\to \arctan\frac{P(u)-u/k}{1+P(u)u/k} + \arctan\frac{u/k-u}{1+u^2/k} + x \ , \\ &= \arctan\frac{kP-u}{k+uP} + \arctan\frac{u-ku}{k+u^2} \ . \end{aligned} \tag{23}$$

The first step in the transformation uses the formal identity

$$\arctan u = \arctan(\tan x) \to x \ .$$

The second step combines the inverse tangents in pairs, again dropping the piecewise constants. This will be a continuous expression provided

$$\forall u \in \mathbb{R}, \quad k + uP(u) > 0 \ .$$

Since P has odd degree, uP has even degree, so again k exists. Our aim is therefore to obtain an expression for k in each case.

For the specific integral example given in (21), we have that $uP = u^4 + au^2$, and the above routine gives the lower bound $k = -1/4\,(\max(1, a+1) - a)^2$. This value can now be used in (23).

References

1. Bank, B., Guddat, J., Klatte, D., Kummer, B., Tammer, K.: Non-linear parametric optimization. Birkhäuser, Basel (1983)
2. Brosowski, B.: Parametric Optimization and Approximation. Birkhäuser, Basel (1985)
3. Corless, R.M., Jeffrey, D.J.: Well, it isn't quite that simple. SIGSAM Bulletin 26(3), 2–6 (1992)
4. Floudas, C.A., Pardalos, P.M.: Encyclopedia of Optimization. Kluwer Academic, Dordrecht (2001)
5. Hong, H.: Simple solution formula construction in cylindrical algebraic decomposition based quantifier elimination. In: Wang, P.S. (ed.) Proceedings of ISSAC 1992, pp. 177–188. ACM Press, New York (1992)
6. Jeffrey, D.J.: Integration to obtain expressions valid on domains of maximum extent. In: Bronstein, M. (ed.) Proceedings of ISSAC 1993, pp. 34–41. ACM Press, New York (1993)
7. Lazard, D.: Quantifier elimination: optimal solution for two classical problems. J. Symbolic Comp. 5, 261–266 (1988)
8. Ulrich, G., Watson, L.T.: Positivity conditions for quartic polynomials. SIAM J. Sci. Computing 15, 528–544 (1994)
9. Jeffrey, D.J., Rich, A.D.: The evaluation of trigonometric integrals avoiding spurious discontinuities. ACM TOMS 20, 124–135 (1994)
10. Jeffrey, D.J.: The importance of being continuous. Mathematics Magazine 67, 294–300 (1994)

The Nearest Real Polynomial with a Real Multiple Zero in a Given Real Interval

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
http://www.brl.ntt.co.jp/people/sekigawa/

Abstract. Given $f \in \mathbb{R}[x]$ and a closed real interval I, we provide a rigorous method for finding a nearest polynomial with a real multiple zero in I, that is, $\tilde{f} \in \mathbb{R}[x]$ such that $\tilde{f}$ has a multiple zero in I and $\|f - \tilde{f}\|_\infty$, the infinity norm of the vector of coefficients of $f - \tilde{f}$, is minimal. First, we prove that if a nearest polynomial $\tilde{f}$ exists, there is a nearest polynomial $\tilde{g} \in \mathbb{R}[x]$ such that the absolute value of every coefficient of $f - \tilde{g}$ is $\|f - \tilde{f}\|_\infty$ with at most one exceptional coefficient. Using this property, we construct $h \in \mathbb{R}[x]$ such that a zero of h is a real multiple zero $\alpha \in I$ of $\tilde{g}$. Furthermore, we give a rational function whose value at α is $\|f - \tilde{f}\|_\infty$.

1 Introduction

Problems on the locations of the zeros of a polynomial are very interesting and important both in theory and practice. In many practical examples, the coefficients of polynomials may contain errors because they are obtained through measurement or can be specified only with finite precision. As described in [1], problems as to how perturbations of such coefficients affect the location of zeros have been treated in control theory, and Kharitonov's theorem [2] and the edge theorem [3] are landmark results.

For real polynomials, it is natural to consider only real perturbations since in many practical examples, the errors are also real numbers. It is also natural to consider only real zeros for many applications. Hence, it is important to decide whether there exists a real multiple zero since it may become close real zeros or complex zeros through perturbations of coefficients, however small those perturbations may be. In particular, we consider the following type of problem.

> Given $f \in \mathbb{R}[x]$ and a closed real interval I, find a nearest polynomial $\tilde{f} \in \mathbb{R}[x]$ in a weighted l^∞-norm that has a multiple zero in I.

In this paper, the l^p-norm of a polynomial is the p-norm of the vector of coefficients of the polynomial with respect to a given basis.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 32–41, 2008.
© Springer-Verlag Berlin Heidelberg 2008

A real univariate polynomial $f(x) = \sum_{j=0}^{n} a_j x^j$ has a real multiple zero at a given point α if and only if $\sum_{j=0}^{n} a_j \alpha^j = \sum_{j=1}^{n} j a_j \alpha^{j-1} = 0$, and $f(x)$ has a complex zero at a given point β if and only if $\sum_{j=0}^{n} a_j \operatorname{Re} \beta^j = \sum_{j=0}^{n} a_j \operatorname{Im} \beta^j = 0$. Regarding a_j as variables, both systems of equations are of the form $\sum_{j=0}^{n} a_j \gamma_j = \sum_{j=0}^{n} a_j \delta_j = 0$ for given real numbers γ_j and δ_j. Thus, the decision methods as to whether $f(x)$ has a real multiple zero at α and as to whether $f(x)$ has a complex zero at β are closely related.

In control theory, the results in [4,5] are for finding the nearest polynomial having a complex zero in a prescribed complex domain by searching within a subset of the domain. (See also the textbook [6].)

In numerical analysis, finding the nearest polynomial to a given function under a number of conditions has been studied in approximation theory. (See the textbook [7], for example.) The Remez algorithm [8], which is an iterative algorithm, can be used for such problems when we use the uniform norm as the distance between two functions. For such optimization problems, theories on normed linear spaces, especially duality theorems, are useful [9].

The proposed method in this paper belongs to the categories of symbolic or symbolic-numeric algorithms. There are a lot of studies related to the above problem in these categories [10,1,11,12,13,14,15,16,17,18,19,20]. Theories on normed linear spaces and duality are also used, for example, [10,13,15,19]. The problem treated in [20] is closely related to the above problem. However, the authors of [20] use the l^2-norm while we use a weighted l^∞-norm. Given $f \in \mathbb{R}[x]$, a closed real interval I, and $\epsilon > 0$, we can decide whether there exists $g \in \mathbb{R}[x]$ such that g has a real multiple zero in I and $\|f - g\|_\infty \leq \epsilon$ using the method in [17]. However, we cannot compute the minimal ϵ such that there exists such a polynomial g. For a given complex (or real) polynomial and locus, reference [11] proposes a method to find the nearest polynomial in a weighted l^2-norm that has a zero in the locus. A similar problem for a real polynomial in a weighted l^∞-norm and a closed complex domain is treated in [16]. Although [16] treats complex zeros, because of the above reason, we can use a similar method to the one in [16] to solve the problem in this paper. Furthermore, we extend the results of [16] as follows. First, we prove that a nearest polynomial exists when the number of perturbed coefficients is more than one and I does not degenerate to a point. Second, we give a simpler explicit formula for the distance between f and a nearest polynomial $\tilde{f}$, together with a necessary and sufficient condition for the existence of $\tilde{f}$ when I consists of one point. We can obtain the latter result through arguments on convex polygons. From these arguments, we can naturally obtain the property that if a nearest polynomial exists, there is a nearest polynomial $\tilde{g}$ such that the absolute value of every coefficient of $f - \tilde{g}$ is $\|f - \tilde{g}\|_\infty$ with at most one exceptional coefficient.

The method in this paper is algebraic. We solve systems of algebraic equations to obtain a nearest polynomial $\tilde{f}$. We can represent exactly the coefficients of $\tilde{f}$, the multiple zeros of $\tilde{f}$ in I, and $\|f - \tilde{f}\|_\infty$ as algebraic numbers. However, to obtain their numerical values, iteration or search methods are more efficient.

The rest of this article is organized as follows. Section 2 describes the theorems that support the proposed methods, whose details are explained in Sect. 3. Section 4 concludes the paper by mentioning directions in which future work might proceed.

2 Theoretical Background

2.1 Existence of a Nearest Polynomial

In this paper, we assume the following conditions.

Condition 1

1. $f(x)$, $e_1(x)$, …, $e_n(x) \in \mathbb{R}[x]$ are given, the number of $e_i(x)$ is finite, and $e_1(x)\mathbb{R} + \cdots + e_n(x)\mathbb{R} \cong \mathbb{R}^n$.
2. A closed real interval I is given and $f(x)$ does not have a multiple zero in I.
3. The degrees of the polynomials in $F = \{ f(x) + \sum_{i=1}^{n} c_i e_i(x) \mid c_i \in \mathbb{R} \}$ are constant when I is not bounded.
4. A metric d on F is given such that the topology of F induced by d is the ordinal topology of $\mathbb{R}^n$. Here, we identify F with $\mathbb{R}^n$ through the map $F \ni f(x) + \sum_{i=1}^{n} c_i e_i(x) \mapsto (c_1, \ldots, c_n) \in \mathbb{R}^n$.

We consider the following problem.

Problem 1. Find $\tilde{f} \in F$ such that $\tilde{f}$ has a multiple zero in I and $d(f, \tilde{f})$ is minimal.

Clearly, there is no solution to Problem 1 if there is no polynomial in F having a multiple zero in I. If there is a polynomial in F having a multiple zero in I, there is a solution to Problem 1.

Theorem 1. *Define f, e_i, F, I and d as in Condition 1. If there is a polynomial $g \in F$ having a multiple zero in I, then there is a polynomial $\tilde{f} \in F$ such that $\tilde{f}$ has a multiple zero in I and $d(f, \tilde{f})$ is minimal.*

Proof. First, we prove the case where I is bounded.

Note that $f(x)+\sum_{i=1}^{n} t_i e_i(x)$ has a multiple zero at a if and only if the system of equations $f(a)+\sum_{i=1}^{n} t_i e_i(a) = f'(a)+\sum_{i=1}^{n} t_i e_i'(a) = 0$ hold. When a is real, the system of equations is equivalent to

$$\left(f(a) + \sum_{i=1}^{n} t_i e_i(a) \right)^2 + \left(f'(a) + \sum_{i=1}^{n} t_i e_i'(a) \right)^2 = 0. \tag{1}$$

Taking $G(a, t_1, \ldots, t_n)$ as a function from $\mathbb{R}^{n+1}$ to $\mathbb{R}$ defined by the left-hand side of (1), similar arguments to those in the proof for Theorem 1 in [16] hold, and we obtain the result.

Next, we prove the case where I is not bounded. To find a nearest polynomial, it is sufficient to investigate polynomials belong to $\tilde{F} = \{ h \in F \mid d(f, h) \leq$

$d(f,g)\}$. Since the leading coefficients of the polynomials in F are equal from the third condition of Condition 1, there exists $r > 0$ such that all of the zeros of the polynomials in $\tilde{F}$ belong to a closed disk with center 0 and radius r. Therefore, we can replace I by the closed bounded interval $I \cap [-r, r]$. □

Remark 1. The following examples show that Theorem 1 does not hold when the number of e_i is not finite, or I is not bounded and the degrees of the polynomials in F are not constant.

1. Let $f(x) = 1 - x$, $e_i(x) = x^i$ $(1 \leq i \in \mathbb{N})$,
$$F = \left\{ 1 - x + \sum_{i=1}^{\infty} c_i x^i \;\middle|\; c_i \in \mathbb{R},\ \#\{\, c_i \mid c_i \neq 0 \,\} < \infty \right\},$$
$I = \{\, 1 \,\}$, and $d(g,h) = \|g - h\|_\infty$. Then, for $2 \leq m \in \mathbb{N}$, the polynomial
$$f_m(x) = 1 - x - \frac{1}{m-1}(x^2 - x^3 + x^4 - \cdots - x^{2m-1})$$
belongs to F, $d(f, f_m) = 1/(m-1)$ holds, and f_m has a multiple zero at 1.
2. Let $f(x) = 1$, $e_1(x) = x$, $e_2(x) = x^2$, $I = \{\, x \in \mathbb{R} \mid 0 \leq x \,\}$, and $d(g,h) = \|g - h\|_\infty$. Then, the polynomial $f_\epsilon(x) = 1 - 2\epsilon x + \epsilon^2 x^2$ $(0 < \epsilon \leq 2)$ belongs to F, $d(f, f_\epsilon) = 2\epsilon$ holds, and $f_\epsilon(x)$ has a multiple zero at $1/\epsilon \in I$.

Remark 2. If we require $\deg(\tilde{f}) = \deg(f)$, there might be no solution to Problem 1. Let $f(x) = 1 + x + x^2$, $e_1(x) = 1$, $e_2(x) = x$, $e_3(x) = x^2$, $I = \{\, 1/2 \,\}$, and $d(g,h) = \|g - h\|_\infty$. Then, the zero polynomial is the only nearest polynomial. Any polynomial of degree two having $1/2$ as a multiple zero is of the form $c(1/4 - x + x^2)$, where $c \in \mathbb{R}$, $c \neq 0$. Since $d(f, c(1/4 - x + x^2)) = 1 + |c|$ holds, there is no polynomial $\tilde{f}$ such that $\deg(\tilde{f}) = 2$ and $d(f, \tilde{f})$ is minimal.

The following is a sufficient condition for the existence of a polynomial in F having a multiple zero in I.

Theorem 2. *Define f, e_i, and F as in Condition 1. If $2 \leq n$ and I does not degenerate to a point, there is a polynomial in F having a multiple zero in I.*

Proof. It is sufficient to prove the case where $n = 2$.

When the determinant $D(x) = \begin{vmatrix} e_1(x) & e_2(x) \\ e_1'(x) & e_2'(x) \end{vmatrix}$ is not identically equal to 0, there is $\alpha \in I$ such that $D(\alpha) \neq 0$. Therefore, $(e_1(\alpha), e_1'(\alpha))$ and $(e_2(\alpha), e_2'(\alpha))$ are linearly independent over $\mathbb{R}$. Hence, there exist c_1, $c_2 \in \mathbb{R}$ such that $(-f(\alpha), -f'(\alpha)) = c_1(e_1(\alpha), e_1'(\alpha)) + c_2(e_2(\alpha), e_2'(\alpha))$. Therefore, $f(x) + c_1 e_1(x) + c_2 e_2(x)$ has a multiple zero at $\alpha \in I$.

Next, we consider the case where $D(x)$ is identically equal to 0. Changing the indexes, if necessary, we can assume that e_1' is not the zero polynomial since e_1 or e_2 is not constant. Thus, we have $e_2/e_1 = e_2'/e_1'$ and set it as $\varphi(x)$. Then, we can write $e_2 = \varphi e_1$ and $e_2' = \varphi e_1'$. On the other hand, by taking the derivatives of the both sides of $e_2 = \varphi e_1$, we obtain $e_2' = \varphi' e_1 + \varphi e_1'$. Therefore, φ' is identically equal to zero and φ is constant, say, c. That is, $e_2 = c e_1$ $(c \in \mathbb{R})$ holds. This contradicts the first condition of Condition 1. □

2.2 Main Theorem

Hereafter, we use the l^∞-norm as the metric d. That is, for $g = f + \sum_{i=1}^n b_i e_i$ and $h = f + \sum_{i=1}^n c_i e_i$, define $d(g,h) = \max_i\{\,|b_i - c_i|\,\}$. If we want to use a weighted l^∞-norm $\max_i\{\,w_i|b_i - c_i|\,\}$ $(w_i > 0)$, we use $\{\,e_1/w_1, \ldots, e_n/w_n\,\}$ as a basis.

The following is the main theorem.

Theorem 3. *Define f, e_i, F, and I as in Condition 1.*

1. *If there is a polynomial in F having a multiple zero in I, then there is a nearest polynomial $\tilde{f}(x) = f(x) + \sum_{i=1}^n \tilde{a}_i e_i(x)$ having a multiple zero in I such that $|a_i - \tilde{a}_i| = \|f - \tilde{f}\|_\infty$ with at most one exceptional i.*
2. *When $I = \{\,\alpha\,\}$, the following holds.*
   (a) *If $(0,0)$, $(e_1(\alpha), e_1'(\alpha))$, $\ldots$, $(e_n(\alpha), e_n'(\alpha))$ lie on a straight line, then a necessary and sufficient condition for the existence of a nearest polynomial $\tilde{f}$ having a multiple zero in I is that $(e_i(\alpha), e_i'(\alpha)) \neq (0,0)$ holds for some i, and that $(f(\alpha), f'(\alpha))$ lies on the straight line. When a nearest polynomial $\tilde{f}$ exists, the following holds.*

$$\|f - \tilde{f}\|_\infty = \begin{cases} \dfrac{|f'(\alpha)|}{\sum_{i=1}^n |e_i'(\alpha)|}, & \text{if } e_1(\alpha) = \cdots = e_n(\alpha), \\ \dfrac{|f(\alpha)|}{\sum_{i=1}^n |e_i(\alpha)|}, & \text{otherwise.} \end{cases}$$

   (b) *If $(0,0)$, $(e_1(\alpha), e_1'(\alpha))$, $\ldots$, $(e_n(\alpha), e_n'(\alpha))$ do not lie on a straight line, then there is a nearest polynomial $\tilde{f}$ having a multiple zero in I, and*

$$\|f - \tilde{f}\|_\infty = \max_{i=1,\ldots,n} \left\{ \frac{|e_i(\alpha)f'(\alpha) - e_i'(\alpha)f(\alpha)|}{\sum_{j=1}^n |e_i(\alpha)e_j'(\alpha) - e_i'(\alpha)e_j(\alpha)|} \right\},$$

   where we omit the elements whose denominators are 0.

Proof. Let $F_\epsilon = \{\, g \in F \mid \|g - f\|_\infty \leq \epsilon \,\}$ $(0 < \epsilon)$. There exists $g \in F_\epsilon$ such that α is a real multiple zero of g if and only if $(0,0)$ belongs to $\boldsymbol{F}_\epsilon = \{\, (h(\alpha), h'(\alpha)) \mid h(x) \in F_\epsilon \,\}$. Let $\boldsymbol{f} = (f(\alpha), f'(\alpha))$ and $\hat{\boldsymbol{e}}_i = (\hat{e}_i(\alpha), \hat{e}_i'(\alpha))$, where

$$\hat{e}_i(x) = \begin{cases} -e_i(x), \text{ if } e_i(\alpha) < 0 \text{ or if } e_i(\alpha) = 0 \text{ and } e_i'(\alpha) < 0, \\ e_i(x), \text{ otherwise.} \end{cases}$$

We can write $\boldsymbol{F}_\epsilon$ as $\boldsymbol{f} + \boldsymbol{P}_\epsilon$, where $\boldsymbol{P}_\epsilon = \{\, \sum_{i=1}^n b_i \hat{\boldsymbol{e}}_i \mid b_i \in \mathbb{R},\ |b_i| \leq \epsilon \,\}$. Then, $(0,0)$ belongs to $\boldsymbol{F}_\epsilon$ if and only if $-\boldsymbol{f}$ belongs to the set $\boldsymbol{P}_\epsilon$. This is equivalent to $\boldsymbol{f} \in \boldsymbol{P}_\epsilon$, since $\boldsymbol{P}_\epsilon$ is symmetric about the origin.

Next, we prove that $\boldsymbol{P}_\epsilon$ is a convex polygon. Let $\boldsymbol{Q}_\epsilon$ be $\{\, \sum_{i=1}^n 2t_i\epsilon\hat{\boldsymbol{e}}_i \mid 0 \leq t_i \leq 1 \,\}$. Then, $\boldsymbol{P}_\epsilon$ can be written as $-\epsilon\sum_{i=1}^n \hat{\boldsymbol{e}}_i + \boldsymbol{Q}_\epsilon$. The set $\boldsymbol{Q}_\epsilon$ is a convex polygon (see Theorem 4 in [17]), and thus, so is $\boldsymbol{P}_\epsilon$.

We can find the vertices of $\boldsymbol{P}_\epsilon$ as follows. First, sort $\hat{\boldsymbol{e}}_i \neq (0,0)$ in increasing order of the slopes $\hat{e}_i'(\alpha)/\hat{e}_i(\alpha)$ (if $\hat{e}_i(\alpha) = 0$ then $\hat{e}_i'(\alpha)/\hat{e}_i(\alpha)$ is regarded as

∞). If two or more $\hat{\boldsymbol{e}}_i$, say $\hat{\boldsymbol{e}}_i$, $\hat{\boldsymbol{e}}_j$, $\hat{\boldsymbol{e}}_k$, have the same slope, replace $\hat{\boldsymbol{e}}_i$, $\hat{\boldsymbol{e}}_j$, $\hat{\boldsymbol{e}}_k$ with $\hat{\boldsymbol{e}}_i + \hat{\boldsymbol{e}}_j + \hat{\boldsymbol{e}}_k$, and write the sorted results as $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m$. Using Theorem 4 in [17] we can describe the vertices of $\boldsymbol{Q}_\epsilon$ in $\boldsymbol{s}_i$, and shifting the vertices with $-\epsilon \sum_{i=1}^{n} \hat{\boldsymbol{e}}_i$, we see that the vertices of $\boldsymbol{P}_\epsilon$ are, in counterclockwise order, $\epsilon \boldsymbol{p}_0$, $\ldots$, $\epsilon \boldsymbol{p}_{2m-1}$, where

$$\boldsymbol{p}_i = \begin{cases} \sum_{j=1}^{i} \boldsymbol{s}_j - \sum_{j=i+1}^{m} \boldsymbol{s}_j & (i = 0, \ \ldots, \ m-1), \\ -\boldsymbol{p}_{i-m} & (i = m, \ \ldots, \ 2m-1). \end{cases}$$

Figure 1 shows an example of $(e_i(\alpha), e_i'(\alpha))$, $\hat{\boldsymbol{e}}_i$, and the corresponding convex polygons $\boldsymbol{P}_\epsilon$ for $\epsilon = 0.5$ and 1 when $n = 3$.

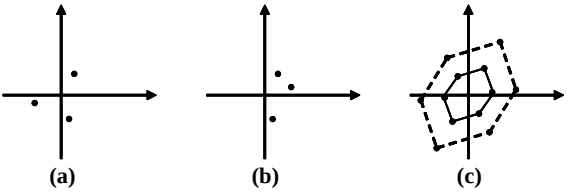


Fig. 1. (a) $(e_i(\alpha), e_i'(\alpha))$ $(i = 1, 2, 3)$. (b) $\hat{\boldsymbol{e}}_i$ $(i = 1, 2, 3)$. (c) Convex polygons $\boldsymbol{P}_\epsilon$ for $\epsilon = 0.5$ (solid line) and $\epsilon = 1$ (dashed line).

When $(0, 0)$, $(e_1(\alpha), e_1'(\alpha))$, $\ldots$, $(e_n(\alpha), e_n'(\alpha))$ lie on a straight line, $\boldsymbol{P}_\epsilon$ degenerates to a line segment, and the necessary and sufficient condition for the existence of a nearest polynomial is obvious. Suppose that there is a nearest polynomial. Then, $\sum_{i=1}^{n} \hat{\boldsymbol{e}}_i \neq 0$ holds and the two endpoints of the line segment are $\pm\epsilon \sum_{i=1}^{n} \hat{\boldsymbol{e}}_i$. Thus, the line segment can be described as the set of points $\boldsymbol{x}$ belonging to the straight line passing through the two points $\pm \sum_{i=1}^{n} \hat{\boldsymbol{e}}_i$ and satisfying the inequality $|\boldsymbol{x}| \leq \epsilon \sum_{i=1}^{n} |\hat{\boldsymbol{e}}_i|$. Therefore, there exists $\tilde{f} \in F_\epsilon$ such that $\tilde{f}$ has a multiple zero at α if and only if the inequality

$$|\boldsymbol{f}| \leq \epsilon \sum_{i=1}^{n} |\hat{\boldsymbol{e}}_i| \tag{2}$$

holds. Statement (a) of the second statement follows from (2).

When the convex polygon $\boldsymbol{P}_\epsilon$ does not degenerate to a line segment, for every $\hat{\boldsymbol{e}}_i \neq 0$ there exist exactly two edges parallel to $\hat{\boldsymbol{e}}_i$, and for every edge there exists at least one $\hat{\boldsymbol{e}}_i$ parallel to the edge. For $\hat{\boldsymbol{e}}_i \neq 0$, take the vertex $\epsilon \boldsymbol{p}_j$ such that the edge whose endpoints are $\epsilon \boldsymbol{p}_j$ and $\epsilon \boldsymbol{p}_{j+1}$ is parallel to $\hat{\boldsymbol{e}}_i$ and j is minimal, and denote j as $\nu(i)$. For two vectors $\boldsymbol{a} = (a_1, a_2)$ and $\boldsymbol{b} = (b_1, b_2)$, we denote $a_1 b_2 - a_2 b_1$ as $\det(\boldsymbol{a}, \boldsymbol{b})$. Then, the equations defining the two straight lines containing the two edges parallel to $\hat{\boldsymbol{e}}_i$ are $\det(\hat{\boldsymbol{e}}_i, \boldsymbol{x} \pm \epsilon \boldsymbol{p}_{\nu(i)}) = 0$, which are equivalent to $\det(\hat{\boldsymbol{e}}_j, \boldsymbol{x}) = \mp\epsilon \det(\hat{\boldsymbol{e}}_j, \boldsymbol{p}_{\nu(j)})$. Since $(0, 0)$ belongs to the convex polygon $\boldsymbol{P}_\epsilon$, it can be described as the set of points $\boldsymbol{x}$ satisfying the inequalities

$|\det(\hat{\boldsymbol{e}}_i, \boldsymbol{x})| \leq \epsilon |\det(\hat{\boldsymbol{e}}_i, \boldsymbol{p}_{\nu(i)})|$ $(i = 1, \ \ldots, \ n)$. Thus, there exists $\tilde{f} \in F_\epsilon$ such that $\tilde{f}$ has a multiple zero at α if and only if the following inequalities hold.

$$|\det(\hat{\boldsymbol{e}}_i, \boldsymbol{f})| \leq \epsilon |\det(\hat{\boldsymbol{e}}_i, \boldsymbol{p}_{\nu(i)})| \quad (i = 1, \ldots, \ n) \tag{3}$$

The left-hand side of (3) is equal to $|e_i(\alpha)f'(\alpha) - e_i'(\alpha)f(\alpha)|$. We compute the right-hand side of (3). Note that $\boldsymbol{p}_{\nu(i)} = \sum_{j \in L(i)} \hat{\boldsymbol{e}}_j - \sum_{j \in R(i)} \hat{\boldsymbol{e}}_j$ holds, where $L(i) = \{\, j \mid \det(\hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j) < 0 \,\}$ and $R(i) = \{\, j \mid \det(\hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j) > 0 \,\}$. Thus, we have

$$\begin{aligned}
\det(\hat{\boldsymbol{e}}_i, \boldsymbol{p}_{\nu(i)}) &= \sum_{j \in L(i)} \det(\hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j) - \sum_{j \in R(i)} \det(\hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j) \\
&= - \sum_{j \in L(i)} |\det(\hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j)| - \sum_{j \in R(i)} |\det(\hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j)| \\
&= - \sum_{j=1}^{n} |\det(\hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j)|.
\end{aligned}$$

Since $|\det(\hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j)| = |e_i(\alpha)e_j'(\alpha) - e_i'(\alpha)e_j(\alpha)|$ holds, statement (b) of the second statement holds.

Finally, it is sufficient to prove the case where $I = \{\, \alpha \,\}$ for the first statement. When $\boldsymbol{P}_\epsilon$ degenerates to a line segment, the equality in (2) holds for the minimum value of ϵ. Thus, the statement holds with no exceptional coefficient.

When convex polygon $\boldsymbol{P}_\epsilon$ does not degenerate to a line segment, $\boldsymbol{f}$ lies on an edge of $\boldsymbol{P}_\epsilon$ for the minimum value of ϵ. Thus, $\boldsymbol{f}$ is equal to $(1-t)\epsilon\boldsymbol{p}_i + t\epsilon\boldsymbol{p}_{i+1}$ for some i and $0 \leq t \leq 1$. (If $i = 2m-1$ then $i+1$ is regarded as 0.) Therefore, the statement holds. □

Remark 3. A duality theorem (see, for example, p. 119 in [9]) is used for the similar problem on finding a nearest real polynomial having a zero at a given complex number [10]. In that problem, the minimum distance is computed in Proposition 7.7.2 [21].

3 Computation Methods

In this section, we restrict numbers to real algebraic numbers and use exact computations unless mentioned otherwise. The computation method when I consists of one point is described in Theorem 3 and its proof. Hence, hereafter, we assume that I does not degenerate to a point.

The following two subsections explain the computation methods for obtaining candidate polynomials $\tilde{f}$ in two cases: the first case where $|a_i - \tilde{a}_i| = \|f - \tilde{f}\|_\infty$ for all i, and the second case where there is μ such that $|a_\mu - \tilde{a}_\mu| < \|f - \tilde{f}\|_\infty$. Note that only the first case occurs when $n = 1$. From Theorem 2, there is a nearest polynomial when $2 \leq n$. However, there might be no nearest polynomial when $n = 1$.

3.1 First Case

We can write a nearest polynomial $\tilde{f}$ as $f + \|f - \tilde{f}\|_\infty \sum_{i=1}^{n} \sigma_i e_i$, where $\sigma_i = \pm 1$. Therefore, it is sufficient to solve the following 2^{n-1} systems of equations

$$f(x) + t\left(e_1(x) + \sum_{i=2}^{n} \sigma_i e_i(x)\right) = f'(x) + t\left(e_1'(x) + \sum_{i=2}^{n} \sigma_i e_i'(x)\right) = 0, \quad (4)$$

under the conditions that $x \in I$ and $t \in \mathbb{R}$. Among the solutions we take one such that $|t|$ is minimal. If there is no solution satisfying these conditions, there is no nearest polynomial in this case.

For simplicity, we write $e_1(x) + \sum_{i=2}^{n} \sigma_i e_i(x)$ as $p(x)$. Since $(f(\alpha), f'(\alpha)) \neq (0, 0)$ for $\alpha \in I$, (4) has a solution at $x = \alpha \in I$ if and only if α is a zero of $\psi(x) = f(x)p'(x) - f'(x)p(x)$, and $(p(\alpha), p'(\alpha)) \neq (0, 0)$.

When $\psi(x)$ is not identically equal to 0, $\psi(x)$ has a finite number of zeros. Take $\alpha \in I$ such that $\psi(\alpha) = 0$ and $(p(\alpha), p'(\alpha)) \neq (0, 0)$. Then, (4) has a unique solution $t = -f(\alpha)/p(\alpha)$ (or $-f'(\alpha)/p'(\alpha)$ when $p(\alpha) = 0$).

Next, we consider the case where $\psi(x)$ is identically equal to 0. If $p'(x)$ is not identically equal to 0, then from similar arguments to those in the proof for Theorem 2, we have $f(x) = cp(x)$ for some $c \in \mathbb{R}$. Therefore, the zero polynomial is the only polynomial having a multiple zero in I. Since $1 \leq \deg(p)$ holds when $2 \leq n$, $p'(x)$ being identically equal to 0 implies that $n = 1$. In this case, e_1 is constant, say c. Note that $c \neq 0$. Then, (4) is $f(x) + ct = f'(x) = 0$. Take $\alpha \in I$ such that $f'(\alpha) = 0$. Then, $t = -f(\alpha)/c$.

In every case, there are only finitely many candidates $\alpha \in I$.

3.2 Second Case

We can write a nearest polynomial $\tilde{f}$ as $f(x) + \tilde{a}_\mu e_\mu(x) + \|f - \tilde{f}\|_\infty \sum_{i \neq \mu} \sigma_i e_i(x)$, where $\sigma_i = \pm 1$. Therefore, it is sufficient to solve the following $n \cdot 2^{n-2}$ systems of equations

$$\begin{cases} f(x) + t_\mu e_\mu(x) + t\left(e_\nu(x) + \displaystyle\sum_{i \neq \mu, \nu} \sigma_i e_i(x)\right) = 0, \\ f'(x) + t_\mu e_\mu'(x) + t\left(e_\nu'(x) + \displaystyle\sum_{i \neq \mu, \nu} \sigma_i e_i'(x)\right) = 0, \end{cases} \quad (5)$$

under the conditions that $x \in I$, t_μ, $t \in \mathbb{R}$, and $|t_\mu| < |t|$. Here, $\nu = 2$ if $\mu = 1$ and $\nu = 1$ otherwise. Among the solutions, we take one such that $|t|$ is minimal. If there is no solution satisfying these conditions, there is no nearest polynomial in this case.

For simplicity, we write $e_\nu(x) + \sum_{i \neq \mu, \nu} \sigma_i e_i(x)$ as $q_\mu(x)$. Take a multiple zero $\alpha \in I$ of $\tilde{f}(x)$. If the determinant $\begin{vmatrix} e_\mu(x) & q_\mu(x) \\ e_\mu'(x) & q_\mu'(x) \end{vmatrix}$ is 0 at $x = \alpha$, we can decrease $|t|$

such that the inequality $|t_\mu| < |t|$ and the equations in (5) hold. This contradicts the assumption that $\tilde{f}$ is a nearest polynomial. Therefore, the determinant is not 0 in a neighborhood $U \subset I$ of α. Hence, we can write t_μ and t as functions of x, that is, $t_\mu(x)$ and $t(x)$. If $|t(x)|$ does not have a minimum at $x = \alpha$, by moving β in I from α, we can decrease $|t(\beta)|$ such that the condition $|t_\mu(\beta)| < |t(\beta)|$ and we obtain a polynomial $g(x) = f(x) + \sum_{i=1}^{n} b_i e_i(x)$ $(b_i \in \mathbb{R})$ such that $g(\beta) = 0$ and $\|f - g\|_\infty < \|f - \tilde{f}\|_\infty$. This is a contradiction. Therefore, $|t(x)|$ has a minimum at $x = \alpha$ in U.

When $t(x)$ is a constant c, the candidate polynomials satisfy $|t_\mu(x)| < |c|$. We can take any $\alpha \in I$ as a candidate. (It is enough to take one $\alpha \in I$.) When $t(x)$ is not constant and $|t(x)|$ is minimal at $x = \alpha$, α is a zero of $\frac{dt(x)}{dx}$ or α is one of the endpoints of I. Therefore, there are only finitely many candidates $\alpha \in I$.

In every case, there are only finitely many candidates $\alpha \in I$.

3.3 Computational Complexity

The systems of equations to be solved total 2^{n-1} in the first case and $n \cdot 2^{n-2}$ in the second case. Furthermore, we compare the values of $t(\alpha)$, where t are rational functions with real algebraic coefficients and α are roots of the solved equations.

3.4 Examples

Consider the following examples, which are closely related to the famous example given by Wilkinson. Let $f(x) = \prod_{i=1}^{20}(x - i)$, $e_1(x) = x^{19}$, $e_2(x) = x^{18}$, and $I = \mathbb{R}$. For $F_1 = \{ f + c_1 e_1 \mid c_1 \in \mathbb{R} \}$ and $F_2 = \{ f + c_1 e_1 + c_2 e_2 \mid c_1, c_2 \in \mathbb{R} \}$, there are unique nearest polynomials to $f(x)$ having real multiple zeros. That is, $f(x) + 1.3508\ldots \times 10^{-10} x^{19} \in F_1$ with a multiple zero $15.4864\ldots$, and $f(x) + 1.2689\ldots \times 10^{-10}(x^{19} + x^{18}) \in F_2$ with a multiple zero $15.4869\ldots$.

4 Conclusion

For a given real univariate polynomial f and a prescribed closed real interval I, we proposed a method for finding a real univariate polynomial $\tilde{f}$ such that $\tilde{f}$ has a real multiple zero in I and $\|f - \tilde{f}\|_\infty$ is minimal.

The method is rigorous but its efficiency needs to be investigated, especially to perform for larger examples. Thus, avoiding redundant computations will be one of our studies. Considering similar problems in norms other than a weighted l^∞-norm is another direction of study.

References

1. Hitz, M.A., Kaltofen, E.: The Kharitonov theorem and its applications in symbolic mathematical computation. In: Proc. Workshop on Symbolic-Numeric Algebra for Polynomials (SNAP 1996), pp. 20–21 (1996)

2. Kharitonov, V.L.: Asymptotic stability of an equilibrium position of a family of systems of linear differential equations. Differentsial'nye Uravneniya 14(11), 2086–2088 (1978)
3. Bartlett, A.C., Hollot, C.V., Lin, H.: Root location of an entire polytope of polynomials: It suffices to check the edges. Mathematics of Controls, Signals and Systems 1, 61–71 (1988)
4. Qiu, L., Davison, E.J.: A simple procedure for the exact stability robustness computation of polynomials with affine coefficient perturbations. Systems and Control Letters 13, 413–420 (1989)
5. Rantzer, A.: Stability conditions for polytopes of polynomials. IEEE Trans. Auto. Control 37(1), 79–89 (1992)
6. Bhattacharyya, S.P., Chapellat, H., Keel, L.H.: Robust Control, The Parametric Approach. Prentice-Hall, Englewood Cliffs (1995)
7. Cheney, E.W.: Introduction to Approximation Theory, 2nd edn. Amer. Math. Soc. (1999)
8. Remez, E.Y.: General Computational Methods of Tchebycheff Approximation. Kiev (1957) (Atomic Energy Commission Translation 4491, 1–85)
9. Luenberger, D.G.: Optimization by Vector Space Methods. John Wiley & Sons Inc., Chichester (1969)
10. Graillat, S.: A note on a nearest polynomial with a given root. ACM SIGSAM Bulletin 39(2), 53–60 (2005)
11. Hitz, M.A., Kaltofen, E.: Efficient algorithms for computing the nearest polynomial with constrained roots. In: Proc. 1998 International Symposium on Symbolic and Algebraic Computation (ISSAC 1998), pp. 236–243 (1998)
12. Hitz, M.A., Kaltofen, E., Lakshman, Y.N.: Efficient algorithms for computing the nearest polynomial with a real root and related problems. In: Proc. 1999 International Symposium on Symbolic and Algebraic Computation (ISSAC 1999), pp. 205–212 (1999)
13. Kaltofen, E.: Efficient algorithms for computing the nearest polynomial with parametrically constrained roots and factors. In: Lecture at the Workshop on Symbolic and Numerical Scientific Computation (SNSC 1999) (1999)
14. Mosier, R.G.: Root neighborhoods of a polynomial. Math. Comp. 47(175), 265–273 (1986)
15. Rezvani, N., Corless, R.C.: The nearest polynomial with a given zero, revisited. ACM SIGSAM Bulletin 39(3), 73–79 (2005)
16. Sekigawa, H.: The nearest polynomial with a zero in a given domain. In: Proc. 2007 International Workshop on Symbolic-Numeric Computation (SNC 2007), pp. 190–196 (2007)
17. Sekigawa, H., Shirayanagi, K.: Locating real multiple zeros of a real interval polynomial. In: Proc. 2006 International Symposium on Symbolic and Algebraic Computation (ISSAC 2006), pp. 310–317 (2006)
18. Sekigawa, H., Shirayanagi, K.: On the location of zeros of an interval polynomial. In: Wang, D., Zhi, L. (eds.) Symbolic-Numeric Computation, pp. 167–184. Birkhäuser, Basel (2007)
19. Stetter, H.J.: The nearest polynomial with a given zero, and similar problems. ACM SIGSAM Bulletin 33(4), 2–4 (1999)
20. Zhi, L., Wu, W.: Nearest singular polynomials. J. Symbolic Computation 26(6), 667–675 (1998)
21. Karow, M.: Geometry of spectral value sets. PhD thesis, Universität Bremen (2003)

Practical and Theoretical Issues for the Computation of Generalized Critical Values of a Polynomial Mapping

Mohab Safey El Din

INRIA Paris-Rocquencourt Center, SALSA Project,
UPMC, Univ Paris 06, LIP6,
CNRS, UMR 7606, LIP6,
UFR Ingéniérie 919, LIP6 Passy-Kennedy,
Case 168, 4, Place Jussieu, F-75252 Paris Cedex, France
Mohab.Safey@lip6.fr

Abstract. Let $f \in \mathbb{Q}[X_1, \ldots, X_n]$ be a polynomial of degree D. Computing the set of *generalized critical values* of the mapping $\widetilde{f} : x \in \mathbb{C}^n \to f(x) \in \mathbb{C}$ (i.e. $\{c \in \mathbb{C} \mid \exists (x_k)_{k\in\mathbb{N}} \quad f(x_k) \to c$ and $||x_k||.||d_{x_k} f|| \to 0$ when $k \to \infty\}$) is an important step in algorithms computing sampling points in semi-algebraic sets defined by a single inequality.

A previous algorithm allows us to compute the set of generalized critical values of $\widetilde{f}$. This one is based on the computation of the critical locus of a projection on a plane P. This plane P must be chosen such that some global properness properties of some projections are satisfied. These properties, which are generically satisfied, are difficult to check in practice. Moreover, choosing randomly the plane P induces a growth of the coefficients appearing in the computations.

We provide here a new certified algorithm computing the set of generalized critical values of $\widetilde{f}$. This one is still based on the computation of the critical locus on a plane P. The certification process consists here in checking that this critical locus has dimension 1 (which is easy to check in practice), without any assumption of global properness. Moreover, this allows us to limit the growth of coefficients appearing in the computations by choosing a plane P defined by sparse equations. Additionally, we prove that the degree of this critical curve is bounded by $(D-1)^{n-1} - \mathfrak{d}$ where $\mathfrak{d}$ is the sum of the degrees of the positive dimensional components of the ideal $\langle \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_n} \rangle$.

We also provide complexity estimates on the number of arithmetic operations performed by a probabilistic version of our algorithm.

Practical experiments at the end of the paper show the relevance and the importance of these results which improve significantly in practice previous contributions.

1 Introduction

Consider $f \in \mathbb{Q}[X_1, \ldots, X_n]$ of degree D and the mapping $\widetilde{f} : x \in \mathbb{C}^n \to f(x)$. The set of generalized critical values of $\widetilde{f}$ is defined as the set of points $c \in \mathbb{C}$

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 42–56, 2008.
© Springer-Verlag Berlin Heidelberg 2008

such that there exists a sequence of points $(x_k)_{k\in\mathbb{N}}$ such that $f(x_k) \to c$ and $||x_k|||d_{x_k}f||| \to 0$ when k tends to ∞ (see [20]). From [14,20], this set of points contains:

- the classical set of critical values, i.e. the set of roots of the polynomial generating the principal ideal: $\langle f - T, \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_n}\rangle \cap \mathbb{Q}[T]$;
- the set of asymptotic critical values which is the set of complex numbers for which there exists a sequence of points $(x_k)_{k\in\mathbb{N}} \subset \mathbb{C}^n$ such that $||x_k||$ tends to ∞ and $\left|\left|\left(X_i\frac{\partial f}{\partial X_j}\right)(x_k)\right|\right|$ tends to 0 when k tends to ∞ for all $(i,j) \in \{1,\ldots,n\} \times \{1,\ldots,n\}$.

In this paper, we provide an efficient algorithm allowing us to compute the set of *generalized critical values* of the polynomial mapping $\widetilde{f}$.

Motivation and Description of the Problem. The interest of computing asymptotic critical values of a polynomial mapping comes from the following result which is proved in [28]: *Let $f \in \mathbb{Q}[X_1,\ldots,X_n]$, and $e \in]0, e_0[$ where e_0 is less than the smallest positive generalized critical value of the mapping $x \to f(x)$. If there exists $x \in \mathbb{R}^n$ such that $f(x) = 0$ then each connected component of the semi-algebraic set defined by $f > 0$ contains a connected component of the real algebraic set defined by $f - e = 0$.* Thus, computing generalized critical values is a preliminary step of efficient algorithms computing sampling points in a semi-algebraic set defined by a single inequality, testing the positivity of a given polynomial, etc. In [28], the computation of generalized critical values is also used to decide if a given hypersurface contains real regular points. Once generalized critical values are computed, it remains to compute at least one point in each connected component in a real hypersurface which can be tackled using algorithms relying on the critical point method introduced in [13] (see also [26] and [25] for recent developments leading to practical efficiency).

Given $\mathbf{A} \in GL_n(\mathbb{C})$, we denote by $f^{\mathbf{A}}$ the polynomial $f(\mathbf{AX})$. In [28], the following result is proved (see [28, Theorem 3.6]): *There exists a Zariski-closed subset $\mathcal{A} \subsetneq GL_n(\mathbb{C})$ such that for all $\mathbf{A} \in GL_n(\mathbb{Q}) \setminus \mathcal{A}$, the set of asymptotic critical values of $x \to f(x)$ is contained in the set of non-properness of the projection on T restricted to the Zariski-closure of the constructible set defined by $f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \cdots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}} = 0, \frac{\partial f^{\mathbf{A}}}{\partial X_n} \neq 0$.*

This result induces a probabilistic algorithm which consists in:

1. choosing randomly a matrix $\mathbf{A} \in GL_n(\mathbb{Q})$ and compute an algebraic representation of the Zariski-closure $\mathfrak{C}^{\mathbf{A}}$ of the constructible set defined by:

$$f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \cdots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}} = 0, \quad \frac{\partial f^{\mathbf{A}}}{\partial X_n} \neq 0$$

2. Compute the set of non-properness of the projection on T restricted to $\mathfrak{C}^{\mathbf{A}}$.

Certifying this algorithm is done by checking that for $i = 1,\ldots,n-1$ the projection $\pi_i : (x_1,\ldots,x_n,t) \in \mathbb{C}^{n+1} \to (x_{n-i+1},\ldots,x_n,t) \in \mathbb{C}^{i+1}$ restricted to the

Zariski-closure of the constructible set defined by

$$f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \cdots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-i}} = 0, \quad \frac{\partial f^{\mathbf{A}}}{\partial X_{n-i+1}} \neq 0$$

is proper.

The above algorithm allows us to deal with non-trivial examples and has been used to compute sampling points in a semi-algebraic set defined by a single inequality (see [7] for an application in computational geometry). Nevertheless, some improvements and theoretical issues could be expected:

1. how to limit the growth of coefficients appearing in the computations which are induced by the change of variables $\mathbf{A}$?
2. the certification of the above algorithm can be expensive on some examples; can we find a way to obtain a certified algorithm whose practical efficiency is better than the one of [28]?
3. can we improve the degree bounds on the geometric objects considered during the computations?

Main Contributions. The main result of this paper is the following (see Theorem 3 below): *Let f be a polynomial in $\mathbb{Q}[X_1, \ldots, X_n]$. Suppose that for all $i \in \{1, \ldots, n-1\}$, the Zariski-closure denoted by W_i of the constructible set defined by $f - T = \frac{\partial f}{\partial X_1} = \cdots = \frac{\partial f}{\partial X_{n-i}} = 0, \frac{\partial f}{\partial X_{n-i+1}} \neq 0$ has dimension i. Then, the set of asymptotic critical values of f is contained in the set of non-properness of the projection $(x_1, \ldots, x_n, t) \in \mathbb{C}^n \to t$ restricted to W_1.*

Note that this strongly simplifies the certification process of the algorithm designed in [28] since it is now reduced to compute the dimension of a Zariski-closed algebraic set. This also allows us to use simpler matrices $\mathbf{A}$ (for which the aforementioned projections π_i may be not proper) to avoid a growth of the coefficients. This result is obtained by using local properness of these projections π_i instead of global properness which is used in the proof of [28, Theorem 3.6].

Additionally, we prove that, *There exists a Zariski-closed subset $\mathcal{A} \subsetneq \mathbb{C}^n$ such that for all $(a_1, \ldots, a_n) \in \mathbb{C}^n \setminus \mathcal{A}$, the ideal*

$$\left(\langle L\frac{\partial f}{\partial X_1} - a_1, L\frac{\partial f}{\partial X_2} - a_2, \ldots, L\frac{\partial f}{\partial X_n} - a_n \rangle \cap \mathbb{Q}[X_1, \ldots, X_n] \right) + \langle f - T \rangle$$

has either dimension 1 in $\mathbb{Q}[X_1, \ldots, X_n, T]$ or it equals $\langle 1 \rangle$. Moreover, if the determinant of the Hessian matrix associated to f is not identically null, there exists a Zariski-closed subset $\mathcal{A} \subsetneq \mathbb{C}^n$ such that the above ideal has dimension 1 (see Proposition 1).

Thus, if the determinant of the Hessian matrix of f is not null, we are able to apply the aforementioned result by performing linear change of variables to compute asymptotic critical values by computing a set of non-properness of a projection restricted to a curve. The degree of this curve is crucial to estimate the complexity of our algorithm. We prove in Theorem 4 (see below) that it is bounded by $(D-1)^{n-1} - \mathfrak{d}$ where $\mathfrak{d}$ is the sum of the degrees of the positive

dimensional irreducible components of the variety associated to $\langle \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_n} \rangle$. Note that $\mathfrak{d}$ is an intrinsic quantity. This last result improves the degree bounds provided in [28].

We describe two versions of this algorithm. The first one is certified and uses Gröbner bases to perform algebraic elimination. The second one is probabilistic and uses the geometric resolution of algorithm of [19].

We have implemented the certified version of the algorithm we have obtained using Gröbner bases. We did experiments comparing

- the algorithm we obtained,
- the one which is designed in [28]
- the one designed in [20]
- an algorithm based on CAD computing the asymptotic critical values of a polynomial.

It appears that the algorithm we design in this paper is significantly faster than the previous ones. Compared to the one given [28] which is based on similar geometric techniques, the gain comes from the fact the growth of the coefficients appearing in our algorithm is indeed better controlled.

Organization of the Paper. Section 2 is devoted to recall basic definitions and properties about generalized critical values of a polynomial mapping. Section 3 is devoted to the proof of the results presented above. Section 4 is devoted to present practical experiments showing the relevance of our approach.

2 Preliminaries

In this section, we recall the definitions and basic properties of generalized critical values which can be found in [20].

Definition 1. *A complex number $c \in \mathbb{C}$ is a critical value of the mapping $\widetilde{f} : y \in \mathbb{C}^n \to f(y)$ if and only if there exists $z \in \mathbb{C}^n$ such that $f(z) = c$ and $\frac{\partial f}{\partial X_1}(z) = \cdots = \frac{\partial f}{\partial X_n}(z) = 0$.*

A complex number $c \in \mathbb{C}$ is an asymptotic critical value of the mapping $\widetilde{f} : y \in \mathbb{C}^n \to f(y)$ if and only if there exists a sequence of points $(z_\ell)_{\ell \in \mathbb{N}} \subset \mathbb{C}^n$ such that:

- *$f(z_\ell)$ tends to c when ℓ tends to ∞.*
- *$||z_\ell||$ tends to $+\infty$ when ℓ tends to ∞.*
- *for all $(i,j) \in \{1,\ldots,n\}^2$ $||X_i(z_\ell)||.||\frac{\partial f}{\partial X_j}(z_\ell)||$ 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 $\widetilde{f}$.

In the sequel, we denote by $K_0(f)$ the set of critical values of $\widetilde{f}$, by $K_\infty(f)$ the set of asymptotic critical values of $\widetilde{f}$, and by $K(f)$ the set of generalized critical values of $\widetilde{f}$ (i.e. $K(f) = K_0(f) \cup K_\infty(f)$).

In [20], the authors prove the following result which can be seen as a generalized Sard's theorem for generalized critical values.

Theorem 1. *Let f be a polynomial in $\mathbb{Q}[X_1, \ldots, X_n]$ of degree D. The set of generalized critical values $K(f)$ of the mapping $\widetilde{f} : x \in \mathbb{C}^n \to f(x) \in \mathbb{C}$ is Zariski-closed in $\mathbb{C}$. Moreover, $D\sharp K_\infty(f) + \sharp K_0(f) \leq D^n - 1$*

The main interest of the set of *generalized critical values* relies on its topological properties which are summarized below and proved in [20].

Theorem 2. *The mapping $f_{\mathbb{C}} : x \in \mathbb{C}^n \to f(x) \in \mathbb{C}$ realizes a locally trivial fibration in $\mathbb{C}^n \setminus f_{\mathbb{C}}^{-1}(K(f_{\mathbb{C}}))$.*
The mapping $f_{\mathbb{R}} : x \in \mathbb{R}^n \to f(x) \in \mathbb{R}$ realizes a locally trivial fibration in $\mathbb{R}^n \setminus f_{\mathbb{R}}^{-1}(K(f_{\mathbb{R}}))$.

Thus, $K(f)$ is Zariski-closed, degree bounds on $K(f)$ are Bézout-like degree bounds and its topological properties ensure that there is no topological change of the fibers of f taken above any interval of $\mathbb{R}$ which has an empty intersection with $K(f)$.

In the sequel, for the sake of simplicity, we identify a polynomial $f \in \mathbb{Q}[X_1, \ldots, X_n]$ with the mapping $f_{\mathbb{C}} : x \in \mathbb{C}^n \to f(x) \in \mathbb{C}$.

3 Main Results and Algorithms

3.1 Geometric Results

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 $\phi : V \to W$ of topological spaces is said to be proper at $w \in W$ if there exists a neighborhood B of w such that $f^{-1}(\overline{B})$ is compact (where $\overline{B}$ denotes the closure of B). The map ϕ is said to be proper if it is proper at all $w \in W$.

The following lemma is used in the proof of the main result of this section.

Lemma 1. *Let Δ_{n-j} be the Zariski-closure of the constructible set defined by*

$$\frac{\partial f}{\partial X_1} = \cdots = \frac{\partial f}{\partial X_{n-j}} = 0, \quad \frac{\partial f}{\partial X_{n-j+1}} \neq 0.$$

Suppose that for $j = 1, \ldots, n-1$, Δ_{n-j} has dimension j and that its intersection with the hypersurface defined by $\frac{\partial f}{\partial X_{n-j+1}} = 0$ is regular and non-empty.

Consider the projection $\pi_{n-j+2} : (x_1, \ldots, x_n) \in \mathbb{C}^n \to (x_{n-j+2}, \ldots, x_n) \in \mathbb{C}^{j-1}$ and suppose its restriction to Δ_{n-j} to be dominant. There exists a Zariski-closed subset $\mathcal{Z} \subsetneq \mathbb{C}^{j-1}$ such that if $\alpha \notin \mathcal{Z}$ and if there exists a sequence of points $(x_k)_{k\in\mathbb{N}} \in \pi_{n-j+2}^{-1}(\alpha) \cap \Delta_{n-j}$, such that $\frac{\partial f}{\partial X_{n-j+1}}(x_k) \to 0$ when $k \to \infty$, then there exists a point in $x \in \Delta_{n-j}$ such that $\pi_{n-j+2}(x) = \alpha$ and $\frac{\partial f}{\partial X_{n-j+1}}(x) = 0$.

Proof. Let x be a point of Δ_{n-j+1}, which has, by assumption, dimension $j-1$. Then x belongs to an irreducible component of dimension $j-1$ of the intersection of a component C' of the variety V defined by $\frac{\partial f}{\partial X_1} = \cdots = \frac{\partial f}{\partial X_{n-j}} = 0$ with the

hyperurface H defined by $\frac{\partial f}{\partial X_{n-j+1}} = 0$. The component C' has thus a dimension which is less than $j+1$. Remark now that each component of V has dimension greater than $j-1$ (since it is defined by the vanishing of $n-j$ polynomials). Thus, C' has dimension j and its intersection with the hypersurface H is regular. Then, C' is an irreducible component of Δ_{n-j}. Consider $\mathfrak{C}$ the union of such irreducible components containing points of Δ_{n-j+1}.

Finally, each point in Δ_{n-j+1} lies in $\mathfrak{C}$. Thus, it is sufficient to prove that for a generic choice of $\alpha \in \mathbb{C}^{j-1}$, $\pi_{n-j+2}^{-1}(\alpha) \cap \Delta_{n-j+1}$ is zero-dimensional and not isolated in the variety $\mathfrak{C}$.

Consider the ideal $I = \langle \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_{n-j}} \rangle : \langle \frac{\partial f}{\partial X_{n-j+1}} \rangle^{\infty} \subset \mathbb{Q}[X_1, \ldots, X_n]$ and the ideal $J = I + \langle \frac{\partial f}{\partial X_{n-j+1}} - U \rangle$. Remark that I is equi-dimensional since it has dimension j and contains $\langle \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_{n-j}} \rangle$ which has dimension at least j. Moreover, by assumption, $\dim(J + \langle U \rangle) = \dim(I) - 1$ and π_{n-j+2} is dominant. Then, for all $k \in \{1, \ldots, n-j+1\}$ $J \cap \mathbb{Q}[X_k, X_{n-j+2}, \ldots, X_n, U] + \langle U \rangle$ is generated by a non-constant polynomial P_k. If for all $k \in \{1, \ldots, n-j+1\}$, α does not belong to the leading coefficient of P_k seen as a univariate polynomial in X_k, $\pi_{n-j+2}^{-1}(\alpha)$ has a zero-dimensional intersection $\mathfrak{A}$ with the variety defined by Δ_{n-j+1}. This intersection lies in $\mathfrak{C}$.

Remark now that $\mathfrak{C}$ is equi-dimensional since I is equi-dimensional, so that the points in $\mathfrak{A}$ are not isolated in $\mathfrak{C}$. □

Theorem 3. *Let f be a polynomial in $\mathbb{Q}[X_1, \ldots, X_n]$. Suppose that for all $i \in \{1, \ldots, n-1\}$, the Zariski-closure denoted by W_i of the constructible set defined by $f - T = \frac{\partial f}{\partial X_1} = \cdots = \frac{\partial f}{\partial X_{n-i}} = 0, \frac{\partial f}{\partial X_{n-i+1}} \neq 0$ has dimension i. Then, the set of asymptotic critical values of f is contained in the set of non-properness of the projection $(x_1, \ldots, x_n, t) \in \mathbb{C}^n \to t$ restricted to W_1.*

The proof is based on similar arguments than the one of [28, Theorem 3.6]. We consider below the projections: $\Pi_i : (x_1, \ldots, x_n, t) \mapsto (x_{n-i+2}, \ldots, x_n, t)$ (for $i = n, \ldots, 2$).

Proof. Given an integer j in $\{n+1, \ldots, 2\}$, we say that property $\mathfrak{P}_j$ is satisfied if and only if the following assertion is true: *let $c \in K_\infty(f)$, there exists a sequence of points $(z_\ell)_{\ell \in \mathbb{N}}$ such that for all $\ell \in \mathbb{N}$, $z_\ell \in W_{j-1}$; $f(z_\ell) \to c$ when $\ell \to \infty$; $||z_\ell||$ tends to ∞ when ℓ tends to ∞; and $||z_\ell||.||d_{z_\ell} f|| \to 0$ when $\ell \to \infty$.*

Suppose now $\mathfrak{P}_{j+1}$ is true. We show below that this implies $\mathfrak{P}_j$. Since $\mathfrak{P}_{j+1}$ is supposed to be true, then there exists a sequence of points $(z_\ell)_{\ell \in \mathbb{N}}$ such that for all $\ell \in \mathbb{N}$, $z_\ell \in W_j$, $f(z_\ell) \to c$ when $\ell \to \infty$, $||z_\ell||$ tends to ∞ when ℓ tends to ∞ and $||z_\ell||.||d_{z_\ell} f|| \to 0$ when $\ell \to \infty$.

We prove below that one can choose such a sequence $(z_\ell)_{\ell \in \mathbb{N}}$ in W_{j-1}.

Consider the mapping $\phi : W_j \subset \mathbb{C}^{n+1} \to \mathbb{C}^{2j+1}$ which associates to a point $x = (x_1, \ldots, x_n, t) \in W_j$ the point:

$$\left(x_{n-j+2}, \ldots, x_n, t, \frac{\partial f}{\partial X_{n-j+1}}(x), \left(x_{n-j+r} \sum_{k=n-j+1}^{n} ||\frac{\partial f}{\partial X_k}(x)|| \right)_{r=1,\ldots,j} \right)$$

Remark that using the isomorphism between $\mathbb{C}^n$ and $\mathbb{R}^{2n}$, it is easy to prove that ϕ is a semi-algebraic map. Denote by

$$(a_{n-j+2}, \ldots, a_n, a_{n+1}, a_{0,n-j+1}, a_{n-j+1,n-j+1}, \ldots, a_{n,n-j+1})$$

the coordinates of the target space of ϕ.

By assumption, the restriction of Π_j to W_j has finite fibers. Then, there exists a semi-algebraic subset $\mathcal{Z} \subsetneq \mathbb{C}^{2j+1} \simeq \mathbb{R}^{4j+2}$ such that specializing the coordinates $(a_{n-j+2}, \ldots, a_n, a_{0,n-j+1}, a_{n-j+1,n-j+1})$ of the target space of ϕ to a point

$$\alpha_{n-j+2}, \ldots, \alpha_n, \alpha_{0,n-j+1}, \alpha_{n-j+1,n-j+1}$$

outside $\mathcal{Z}$ defines a finite set of points in the image of ϕ. Indeed, these points are the images of the points in W_j such that their X_i coordinate (for $i = n - j + 2, \ldots, n$) equals α_i and $X_{n-j+1} \sum_{k=n-j+1}^{n} ||\frac{\partial f}{\partial X_k}||$ equals $\alpha_{n-j+1,n-j+1}$.

Given a point $\underline{\alpha} = (\alpha_{n-j+2}, \ldots, \alpha_n) \in \mathbb{C}^{j-1}$ and a complex number $\theta = (\eta_1) \in \mathbb{C}$, such that $(\alpha_{n-j+2}, \ldots, \alpha_n, \eta_1) \notin \mathcal{Z}$, we denote by $y(\underline{\alpha}, \beta)$ a point in the image of ϕ obtained by specializing the first $(j-1)$ coordinates (corresponding to $x_{n-j+2}, \ldots, x_n$) to $\underline{\alpha}$ and the $j+2$-th coordinate (corresponding to $x_{n-j+1} \sum_{k=n-j+1}^{n} ||\frac{\partial f}{\partial X_k}||$). We also denote by $x(\underline{\alpha}, \theta)$ a point in the pre-image of $y(\underline{\alpha}, \theta)$ by ϕ.

Consider $c \in K_\infty(f)$. Then, since $\mathfrak{P}_{j+1}$ is supposed to be true, there exists a sequence of points $(z_\ell)_{\ell \in \mathbb{N}} \subset \mathbb{C}^n$ in the Zariski-closure of the constructible set defined by: $\frac{\partial f}{\partial X_1} = \cdots = \frac{\partial f}{\partial X_{n-j}} = 0, \quad \frac{\partial f}{\partial X_{n-j+1}} \neq 0$ such that $f(z_\ell)$ tends to c when ℓ tends to ∞, $||z_\ell||$ tends to ∞ when ℓ tends to ∞, and $||z_\ell||.||d_{z_\ell} f||$ tends to 0 when ℓ tends to ∞.

Consider the images by ϕ of the points $(z_\ell, f(z_\ell))$ and their first $j-1$ coordinates $\underline{\alpha_\ell}$ and θ_ℓ of their $j+2$-th coordinate. We consider now the double sequence $(\underline{\alpha_i}, \theta_\ell)_{(i,\ell) \in \mathbb{N} \times \mathbb{N}}$.

Note that, by construction, θ_ℓ tends to 0 when ℓ tends to ∞ and that the last $j+1$ coordinates of $y(\underline{\alpha}_{i_0}, \theta_\ell)$ tend to zero when i_0 is fixed and ℓ tends to ∞ if $X_{n-j+1}(x(\alpha_{i_0}, \theta_\ell))$ does not tend to 0 when ℓ tends to ∞.

If for all $\ell \in \mathbb{N}$, $\frac{\partial f}{\partial X_{n-j+1}}(z_\ell) = 0$ the result is obtained. Else, one can suppose that for all $\ell \in \mathbb{N}$, $\frac{\partial f}{\partial X_{n-j+1}}(z_\ell) \neq 0$.

Remark that without loss of generality, we can do the assumption: for all $(i,j) \in \mathbb{N} \times \mathbb{N}$, $x(\underline{\alpha_i}, \theta_\ell)$ is not a root of $\frac{\partial f}{\partial X_{n-j+2}}$ and $(\underline{\alpha_i}, \theta_\ell) \notin \mathcal{Z}$.

Moreover, if $j = n$ remark that the set of non-properness of Π_n restricted to the hypersurface defined by $f - T = 0$ is defined as the set of complex solutions of the leading coefficient of f seen as a univariate polynomial in X_1. Thus, without loss of generality, one can suppose that for all i and for all $t \in \mathbb{C}$, $(\underline{\alpha_i}, t)$ does not belong to this set of non-properness. Else, up to a linear change of variables on the variables $X_{n-j+2}, \ldots, X_n$, one can suppose that the assumptions of Lemma 1 are satisfied and then we choose $\underline{\alpha_i}$ outside the Zariski-closed subset $\mathcal{Z}$ exhibited in Lemma 1.

Remark that, since ϕ is semi-algebraic, $X_{n-j+1}(x(\underline{\alpha}, \theta))$ is a root of a univariate polynomial with coefficients depending on $\underline{\alpha}$ and θ. Then, for a fixed integer

i_0, since θ_ℓ tends to (0) when ℓ tends to ∞, $X_{n-j+1}(x(\underline{\alpha}_{i_0}, \theta_\ell))$ has either a finite limit or tends to ∞ when ℓ tends to ∞.

In the sequel, we prove that for $i_0 \in \mathbb{N}$, $y(\underline{\alpha}_{i_0}, \theta_\ell)$ has a finite limit in $\mathbb{C}^{2n+1}$ when ℓ tends to ∞. Suppose first that $X_{n-j+1}(x(\underline{\alpha}_{i_0}, \theta_\ell))$ has a finite limit when ℓ tends to ∞. Then, $f(x(\underline{\alpha}_{i_0}, \theta_\ell))$ remains bounded (since $X_{n-j+1}(x(\underline{\alpha}_{i_0}, \theta_\ell))$ has a finite limit and since $\frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_{n-j}}$ vanish at $x(\underline{\alpha}_{i_0}, \theta_\ell)$). Thus, it has consequently a finite limit. Moreover, without loss of generality, one can suppose that $X_{n-j+1}(x(\underline{\alpha}_{i_0}, \theta_\ell))$ does not tend to 0 which implies that $\frac{\partial f}{\partial X_{n-j+1}}(x(\underline{\alpha}_{i_0}, \theta_\ell))$ tends to 0 when ℓ tends to ∞.

Suppose now that $X_{n-j+1}(x(\underline{\alpha}_{i_0}, \theta_\ell))$ tends to ∞ when ℓ tends to ∞. This immediately implies that $\frac{\partial f}{\partial X_{n-j+1}}(x(\underline{\alpha}_{i_0}, \theta_\ell))$ tends to 0 when ℓ tends to ∞. Since $X_{n-j+1}(x(\underline{\alpha}_{i_0}, \theta_\ell))$ tends to ∞ when ℓ tends to ∞, and $\left(X_k \frac{\partial f}{\partial X_{n-j+1}}\right)(x(\underline{\alpha}_{i_0}, \theta_\ell))$ tends to 0 when ℓ (for $k \in \{n-j+1, \ldots, n\}$) tends to ∞, using [28, Remark 2.2] and the curve selection Lemma at infinity (see [20, Lemma 3.3, page 9], this implies there exists a semi-algebraic arc $\gamma_{i_0} : [0,1[\to \mathbb{R}^n$ such that:

- $\gamma_{i_0}([0,1[)$ is included in the intersection of W_j and of the linear subspace defined by $X_k = X_k(\underline{\alpha}_{i_0})$ for $k = n-j+2, \ldots, n$, which implies that

$$\sum_{p=1}^{n} \left(X_p \frac{\partial f}{\partial X_p}\right)(\gamma_{i_0}(\rho)) = \left(X_{n-j+1} \frac{\partial f}{\partial X_{n-j+1}}\right)(\gamma_{i_0}(\rho))$$

- $||\gamma_{i_0}(\rho)|| \to \infty$ and $||X_{n-j+1}(\gamma_{i_0}(\rho))||.||\frac{\partial f}{\partial X_{n-j+1}}(\gamma_{i_0}(\rho))|| \to 0$ when ρ tends to 1.

From Lojasiewicz's inequality at infinity [4, 2.3.11, p. 63], this implies that there exists an integer $N \geq 1$ such that: $\forall \rho \in [0,1[,\ ||\frac{\partial f}{\partial X_{n-j+1}}(\gamma_{i_0}(\rho)))|| \leq ||X_{n-j+1}(\gamma_{i_0}(\rho))||^{-1-\frac{1}{N}}$. Following the same reasoning as in [20, Lemma 3.4, page 9], one can re-parametrize γ_{i_0} such that γ_{i_0} becomes a semi-algebraic function from $[0,+\infty[$ to $\mathbb{R}^n$ and $\lim_{\rho \to 1} ||\dot{\gamma}_{i_0}(\rho)|| = 1$. Thus, the following yields: $\forall p \in [0,+\infty[,\ ||\frac{\partial f}{\partial X_{n-j+1}}(\gamma_{i_0}(\rho))||.||\dot{\gamma}_{i_0}(\rho)|| \leq ||X_{n-j+1}(\gamma_{i_0}(\rho))||^{-1-\frac{1}{N}}.||\dot{\gamma}_{i_0}(\rho)||$ and there exists $B \in \mathbb{R}$ such that $\int_0^\infty ||\gamma_{i_0}(\rho)||^{-1-\frac{1}{N}}.||\dot{\gamma}_{i_0}(\rho)||d\rho \leq B.$. Since

$$\int_0^\infty ||\gamma_{i_0}(\rho)||^{-1-\frac{1}{N}}.||\dot{\gamma}_{i_0}(\rho)||d\rho \geq \int_0^\infty ||X_{n-j+1}(\gamma_{i_0}(\rho))||^{-1-\frac{1}{N}}.||\dot{\gamma}_{i_0}(\rho)||d\rho$$

and $\int_0^\infty ||\frac{\partial f}{\partial X_{n-j+1}}(\gamma_{i_0}(\rho))||.||\dot{\gamma}_{i_0}(\rho)||d\rho \geq ||\int_0^\infty \frac{\partial f}{\partial X_{n-j+1}}(\gamma_{i_0}(\rho)).\dot{\gamma}_{i_0}(\rho)d\rho||$, one has finally $||\int_0^\infty \frac{\partial f}{\partial X_{n-j+1}}(\gamma_{i_0}(\rho)).\dot{\gamma}_{i_0}(\rho)d\rho|| \leq B$. Thus, the restriction of f is bounded along γ_{i_0}.

Finally, we have proved that $y(\underline{\alpha}_{i_0}, \theta_\ell)$ tends to a point whose $j+1$-th coordinates is null.

Let y_{i_0} be the limit of $y(\underline{\alpha}_{i_0}, \theta_\ell)$ and let $p_{i_0} \in \mathbb{C}^n$ be $(\underline{\alpha}_{i_0}, c_{i_0})$ and $p_\ell \in \mathbb{C}^n$ be the point whose coordinates are the j-first coordinates of $y(\underline{\alpha}_{i_0}, \theta_\ell)$.

We prove now that y_{i_0} belongs to the image of ϕ. If $j = n$ this is a consequence of the fact that (α_{i_0}, c_{i_0}) does not belong to the set of non-properness of Π_n restricted to the vanishing set of $f - T = 0$. If $j < n$ this is an immediate consequence of Lemma 1.

Thus, $\Pi_{j+1}^{-1}(p_{i_0}) \cap W_{j-1} \neq \emptyset$ and one can extract a converging subsequence from $(x(\underline{\alpha}_{i_0}, \theta_\ell))_{\ell \in \mathbb{N}}$ and let x_{i_0} be the limit of the chosen converging subsequence.

It remains to prove that:

- $(f(x_{i_0}))_{i_0 \in \mathbb{N}}$ tends to c when i_0 tends to ∞
- $\left(X_i \frac{\partial f}{\partial X_j}\right)(x_{i_0})$ for $(i,j) \in \{1, \ldots, n\}$ tends to 0 when i_0 tends to ∞.

which is a consequence of the continuity of the polynomials f and $X_i \frac{\partial f}{\partial X_j}$ for $i = 2, \ldots, n$, and the definition of the sequence of points $x(\alpha_i, \theta_\ell)$. □

Proposition 1. *Let $f \in \mathbb{Q}[X_1, \ldots, X_n]$ be a polynomial of degree $D \geq 2$. There exists a Zariski-closed subset $\mathcal{A} \subsetneq \mathbb{C}^n$ such that for all $(a_1, \ldots, a_n) \in \mathbb{C}^n \setminus \mathcal{A}$, the ideal*

$$\left(\langle L\frac{\partial f}{\partial X_1} - a_1, L\frac{\partial f}{\partial X_2} - a_2, \ldots, L\frac{\partial f}{\partial X_n} - a_n\rangle \cap \mathbb{Q}[X_1, \ldots, X_n]\right) + \langle f - T\rangle$$

has either dimension 1 in $\mathbb{Q}[X_1, \ldots, X_n, T]$ or it equals $\langle 1\rangle$. Moreover, if the determinant of the Hessian matrix associated to f is not identically null, there exists a Zariski-closed subset $\mathcal{A} \subsetneq \mathbb{C}^n$ such that the above ideal has dimension 1.

Proof. This is an immediate consequence of Sard's theorem (see [4, Theorem 2.5.11 and 2.5.12]) applied to the mapping $(x, \ell) \in \mathbb{C}^n \times \mathbb{C} \to \left(\ell \frac{\partial f}{\partial X_1}, \ldots, \ell \frac{\partial f}{\partial X_n}\right)$ □

Theorem 4. *Let $\mathfrak{d}$ be the sum of the degrees of the positive-dimensional primes associated to the ideal $\langle \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_n}\rangle$. The degree of the curve associated to the ideal $\left(\langle L\frac{\partial f}{\partial X_n} - 1, \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_{n-1}}\rangle \cap \mathbb{Q}[X_1, \ldots, X_n]\right) + \langle f - T\rangle$ is dominated by $(D-1)^{n-1} - \mathfrak{d}$.*

Proof. From [9], the sum of the degrees of the prime ideals associated to the radical of the ideal $I = \langle \frac{\partial f}{\partial X_2}, \ldots, \frac{\partial f}{\partial X_n}\rangle$ is dominated by $(D-1)^{n-1}$. Consider the intersection $\mathcal{P}$ of these primes which contain $\frac{\partial f}{\partial X_1}$. Remark now that $\mathcal{P} + \langle \frac{\partial f}{\partial X_n}\rangle = \mathcal{P}$ and then, that the variety associated to $\mathcal{P} + \langle \frac{\partial f}{\partial X_n}\rangle$ is the union of the irreducible components of positive dimension associated to the radical of the ideal $\langle \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_n}\rangle$. Note now that the degree of the curve defined by the ideal $J = \langle L\frac{\partial f}{\partial X_n} - 1, \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_{n-1}}\rangle \cap \mathbb{Q}[X_1, \ldots, X_n]$ is bounded by the one of $I : \mathcal{P}^\infty$ and then is bounded by $(D-1)^{n-1} - \mathfrak{d}$ since $\mathfrak{d}$ is the sum of the degrees of the primes associated to $\mathcal{P}$. At last, note that $J + \langle f - T\rangle$ has the same degree than the one of J.

3.2 Algorithms and Complexity

Our algorithm takes as input a polynomial $f \in \mathbb{Q}[X_1, \ldots, X_n]$ and outputs a non-zero univariate polynomial in $\mathbb{Q}[T]$ whose set of roots contains the set of generalized critical values of the mapping $x \in \mathbb{C}^n \to f(x)$. We focus on the computation of the *asymptotic critical values*, the case of the critical values being already investigated in [28]. Our procedure makes use of algebraic elimination algorithms to represent algebraic varieties defined as the Zariski-closure of the constructible sets defined by $f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \cdots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}} = 0, \quad \frac{\partial f}{\partial X_n} \neq 0$. Below, we show how to use Gröbner bases or the geometric resolution algorithm in our procedures computing the set of asymptotic critical values of f.

Using Gröbner Bases. From Proposition 1, if the determinant of the Hessian matrix of f is not zero, the set of matrices $\mathbf{A}$ such that the Zariski-closure $C_{\mathbf{A}}$ of the complex solution set of $f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \cdots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}} = 0, \quad \frac{\partial f^{\mathbf{A}}}{\partial X_n} \neq 0$ has dimension 1 is Zariski-opened in $GL_n(\mathbb{C})$. From Theorem 3, it suffices to find $\mathbf{A} \in GL_n(\mathbb{Q})$ such that $C_{\mathbf{A}}$ has dimension 1 and to compute the set of non-properness of the restriction to $C_{\mathbf{A}}$ of the projection $\pi : (x_1, \ldots, x_n, t) \to t$. The computation of the set of non-properness requires as input a Gröbner basis encoding the variety to which the considered projection is restricted. Such a routine is shortly described in [28] (see also [26] or [16] for a complete description); it is named `SetOfNonProperness` in the sequel.

Algorithm computing $K_\infty(f)$ using Gröbner bases

- **Input:** a polynomial f in $\mathbb{Q}[X_1, \ldots, X_n]$.
- **Output:** a univariate polynomial $P \in \mathbb{Q}[T]$ such that its zero-set contains $K_\infty(f)$.

- Let $D = \det(\texttt{Hessian}(f))$. If $D = 0$ then return 1
- Choose $\mathbf{A} \in GL_n(\mathbb{C})$.
- Compute a Gröbner basis G the ideal generated by

$$f^{\mathbf{A}} - T, \frac{\partial f^{\mathbf{A}}}{\partial X_1}, \ldots, \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}}, L\frac{\partial f^{\mathbf{A}}}{\partial X_n} - 1$$

  and its dimension d. If $d \neq 1$ then return to the previous step.
- Return $\texttt{SetOfNonProperness}(G, \{T\})$

In the above algorithm, one can first choose matrices $\mathbf{A}$ performing a sparse linear change of variables in order to reduce the height of the integers appearing in the computations. Nevertheless, the use of Gröbner bases as a routine of algebraic elimination does not allow us to obtain a complexity which is polynomial in the quantity bounding the degree of the curve defined as the Zariski-closure of the complex solution set of $f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \cdots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}} = 0 \frac{\partial f^{\mathbf{A}}}{\partial X_n} - 1 \neq 0$.

Using the Geometric Resolution Algorithm. To this end, we consider the geometric resolution algorithm (see [11], [12,19] and references therein). This algorithm is probabilistic and returns a rational parametrization of the complex solution set of the input (see [29] for situations where the input contains a parameter). Here is how it can be used to compute the set of asymptotic critical values of the mapping $x \in \mathbb{C}^n \to f(x) \in \mathbb{C}$.

Probabilistic Algorithm computing $K_\infty(f)$ using the Geometric Resolution Algorithm

- **Input:** a polynomial f in $\mathbb{Q}[X_1, \ldots, X_n]$.
- **Output:** a univariate polynomial $P \in \mathbb{Q}[T]$ such that its zero-set contains $K_\infty(f)$.

- Consider T as a parameter in the polynomial system $f^{\mathbf{A}} - T = \frac{\partial f^{\mathbf{A}}}{\partial X_1} = \cdots = \frac{\partial f^{\mathbf{A}}}{\partial X_{n-1}} = 0, \frac{\partial f^{\mathbf{A}}}{\partial X_n} \neq 0$ and compute a geometric resolution.
- Return the least common multiple of the denominators in the coefficients of the polynomial q.

Using Theorem 4 and Proposition 1, one obtains the following complexity result as a by-product of the complexity estimates given in [19].

Theorem 5. *The above probabilistic algorithm computing $K_\infty(f)$ performs at most $\mathcal{O}(n^7\delta^{4n})$ arithmetic operations in $\mathbb{Q}$ where δ is bounded by $(D-1)^{n-1} - \mathfrak{d}$, where $\mathfrak{d}$ denotes the sum of the degrees of the positive dimensional primes associated to the radical of the ideal generated by $\langle \frac{\partial f}{\partial X_1}, \ldots, \frac{\partial f}{\partial X_n} \rangle$.*

The above complexity estimate improves the one of [28, Theorem 4.3]. Remark that $\mathfrak{d}$ is intrinsic.

4 Practical Results

We have implemented the algorithm presented in the preceding section using Gröbner bases. The Gröbner engine which is used is FGb, release 1.26, [8] which is implemented in C by J.-C. Faugère. Computing rational parametrization of the complex roots of a zero-dimensional ideal from a Gröbner basis is done by RS, release 2.0.37, [21] which is implemented in C by F. Rouillier. Isolation of real roots of univariate polynomials with rational coefficients is done by RS using the algorithm provided in [23].

The resulting implementation is a part of the RAGLib Maple library (release 2.24) [24].

All the computations have been performed on a PC Intel Pentium Centrino Processor 1.86 GHz with 2048 Kbytes of Cache and 1024 MB of RAM.

4.1 Description of the Test-Suite

Our test-suite is based on polynomials coming from applications. Most of the time, the user-question is to decide if the considered polynomial has constant sign on $\mathbb{R}^n$ or to compute at least one point in each connected component outside its vanishing set. As explained in the introduction, the computation of generalized critical values is a preliminary step of efficient algorithms dealing with these problems.

The following polynomial appears in a problem of algorithmic geometry studying the Voronoi Diagram of three lines in $\mathbb{R}^3$. In [7], the authors focus on determining topology changes of the Voronoi diagram of three lines in $\mathbb{R}^3$. The question was first reduced to determining if the zero-set of the discriminant of the following polynomial with respect of the variable u contains real regular points.

This discriminant has degree 30. This discriminant is the product of a polynomial of degree 18 and several polynomials up to an odd power whom zero-set could not contain a real regular point since they are sums of squares. The polynomial of degree 18 is **Lazard II**. D. Lazard and S. Lazard have also asked to determine if the following polynomial which is denoted by **Lazard I** in the sequel is always positive.

$$
\begin{aligned}
&16\,a^2\left(\alpha^2+1+\beta^2\right)u^4+16\,a\left(-\alpha\,\beta\,a^2+a x\alpha+2\,a\alpha^2+2\,a+2\,a\beta^2+a y\beta-\alpha\,\beta\right)u^3+\\
&\left(\left(24\,a^2+4\,a^4\right)\alpha^2+\left(-24\,\beta\,a^3-24\,a\beta-8\,y a^3+24\,x a^2-8\,a y\right)\alpha+24\,a^2\beta^2+4\,\beta^2-\right.\\
&\left.8\,\beta\,x a^3+4\,y^2a^2+24\,y\beta\,a^2-8\,a x\beta+16\,a^2+4\,x^2a^2\right)u^2+\left(-4\,\alpha\,a^3+4\,y a^2-\right.\\
&\left.4\,a x-8\,a\alpha+8\,\beta\,a^2+4\,\beta\right)(\beta-a\alpha+y-a x)\,u+\left(a^2+1\right)(\beta-a\alpha+y-a x)^2
\end{aligned}
$$

The following polynomial appears in [15]. The problem consists in determining the conditions on a, b, c and d such that the ellipse defined by $\frac{(x-c)^2}{a^2}+\frac{(y-d)^2}{b^2}=1$ is inside the circle defined by $x^2+y^2-1=0$. The problem is reduced to compute at least one point in each connected component of the semi-algebraic set defined as the set of points at which the polynomial below (which is denoted by **Ellipse-Circle** in the sequel) does not vanish.

$$
\begin{aligned}
&4\,a^6c^2d^2+2\,a^2b^2d^6-6\,a^2b^2d^4+a^4c^4+2\,a^4c^2d^6-6\,a^2b^2c^4-6\,a^4b^2c^4+4\,a^6b^2d^2+\\
&a^8b^4+6\,b^4c^2d^2-2\,b^6c^4d^2+a^8d^4+6\,a^2b^6d^2-8\,a^4b^4d^2-4\,a^4b^2d^6-6\,b^4c^4d^2-8\,a^4b^4c^2+\\
&6\,a^6b^2c^2-8\,a^2b^4c^2+6\,a^4b^4d^4-2\,b^4c^2d^4-4\,a^2b^4c^6-4\,a^6b^4c^2-6\,a^2b^4d^4-2\,a^4c^4d^2+\\
&10\,a^4b^2d^4-2\,a^2b^8c^2-6\,a^2b^6c^4+a^4b^8+6\,a^2b^2d^2+6\,a^6b^4d^2-4\,a^4b^6d^2+b^4d^4+b^4c^8+\\
&10\,a^2b^4c^4+6\,a^2b^2c^2+4\,a^2b^6c^2+a^4d^8+4\,b^6c^2d^2+6\,a^4b^6c^2-8\,a^4b^2d^2+\\
&4\,a^4b^2c^2-2\,a^8b^2d^2+6\,a^4c^2d^2+4\,a^2b^4d^2-6\,a^6b^2d^4+6\,a^4b^4c^4-2\,a^6c^2d^4+\\
&2\,b^4c^6d^2+2\,a^2b^2c^6-6\,a^4c^2d^4+b^8c^4+2\,a^4b^2-4\,a^4d^2+a^4-2\,b^6-2\,a^6+a^8+\\
&b^8+b^4+2\,a^2b^4+2\,b^6c^6-2\,b^8c^2-6\,b^6c^4+2\,a^6b^4-2\,a^2b^2-2\,a^6b^6+2\,a^4b^6-\\
&2\,a^2b^8-6\,a^4b^2c^4d^2+2\,a^2b^4c^4d^2+2\,a^4b^2c^2d^4-6\,a^2b^4c^2d^4-6\,a^4b^2c^2d^2-6\,a^2b^4c^2d^2+\\
&4\,a^2b^2c^4d^4+2\,a^2b^2c^2d^6+2\,a^2b^2c^4d^2+2\,a^2b^2c^2d^4-10\,a^2b^2c^2d^2+6\,a^2b^6c^2d^2-\\
&6\,a^4b^4+2\,a^2b^6-2\,a^8b^2+2\,a^6b^2+6\,a^6b^2c^2d^2-10\,a^4b^4c^2d^2-4\,b^4c^6+6\,b^4c^4+6\,b^6c^2-\\
&2\,a^6c^2+2\,a^2b^2c^6d^2+a^4c^4d^4-2\,a^4c^2-2\,b^6d^2-4\,a^4d^6+2\,a^6d^6-\\
&2\,a^8d^2-6\,a^6d^4+6\,a^6d^2+b^4c^4d^4-4\,b^4c^2+6\,a^4d^4-2\,b^4d^2\left(a^2-b^2\right)
\end{aligned}
$$

The polynomials $\sum_{i=1}^{n} \prod_{j \neq i}(X_i - X_j)$ which are called in the sequel $\mathbf{LL}_n$ are studied in [18]. They are used as a benchmark for algorithms decomposing polynomials in sums of squares (see also [30]). In the sequel we consider **LL5** (which has degree 4 and contains 5 variables), **LL6** (which has degree 5 and contains 6 variables) and **LL7** (which has degree 6 and contains 7 variables).

We also consider polynomials coming from the Perspective-Three-Point Problem [10] which is an auto-calibration problem. Classifying the number of solutions on some instances of this problem leads to compute at least one point in each connected component outside a hypersurface. We consider two instances of this problem leading to study

- a polynomial denoted by **P3Piso** of degree 16 having 4 variables and 153 monomials.
- a polynomial denoted **P3P** of degree 16 having 5 variables and 617 monomials.

These polynomials are too big for being printed here.

4.2 Practical Results

We only report on timings for the computation of asymptotic critical values.

Below, in the column **JK** we give the timings for computing asymptotic critical values by using the algorithm of [20]. We obviously use the same Gröbner engine `FGb` for both algorithms.

Using similar arguments than the ones used in [1], one can prove that Cylindrical Algebraic Decomposition can compute a Zariski-closed set containing the generalized critical values of the mapping $f : x \to f(x)$ by computing a CAD adapted to $f - T$ (where T is a new variable) and considering T as the smallest variable. The column **CAD** contains the timings of an implementation of the open CAD algorithm in Maple which is due to G. Moroz and F. Rouillier.

The column **S07** contains the timings obtained using the *probabilistic* algorithm described in [28] to compute generalized critical values. In particular, we don't count the time required to certify the output of this algorithm.

The column **Algo** contains the timings obtained with the implementation of the algorithm described in this paper.

The symbol ∞ means that the computations have been stopped after 2 days of computations without getting a result.

It appears that on all the considered problems, the algorithm given in [20] does provide an answer in a reasonable amount of time.

On problems having at most 4 variables, the open CAD algorithm behaves well (except on polynomials having a big degree) and our implementation has comparable timings. On problems having more variables, our implementation ends with reasonable timings while open CAD either does not end after 2 days of computations or requires too much memory. This is mainly due to the highest

degrees appearing in the projection step of CAD while the degrees of the polynomials appearing during the execution of our algorithms is better controlled. The same conclusions hold when we take into account the computation of classical critical values.

In comparison with the algorithm provided in [28], our algorithm performs better on harder problems. On some problems, we obtain a speed-up of 30. This is mainly due to the fact that the growth of coefficients appearing in our algorithm is better controlled than the ones appearing in the algorithm designed in [28]: we take here advantage of Theorem 3 to choose sparse matrices $\mathbf{A}$. Note nevertheless that on smaller problems, our algorithm may be slower: this is mainly due to the search of an appropriate projection (preserving the sparsity of the initial problem) used for the computation of asymptotic critical values.

Table 1. Computation time obtained on a PC Intel Pentium Centrino Processor, 1.86 GHz with 2048 Kbytes of Cache and 1024 MB of RAM.

BM	♯vars	Degree	JK	Algo	S07	CAD
Lazard I	6	8	∞	14 sec.	2 sec.	∞
Lazard II	5	18	∞	192 sec.	3 hours	∞
Ellipse-Circle	4	12	∞	0.7 sec.	90 sec.	5 min.
LL5	5	4	∞	0.2 sec.	0.1 sec.	20 sec.
LL6	6	5	∞	9 sec.	2 sec.	∞
LL7	7	6	∞	28 sec.	139 sec.	∞
P3Piso	4	16	∞	1000 sec.	2 hours	20 min.
P3P	5	16	∞	1100 sec.	7 hours	∞.

References

1. Alcazar, J.G., Schicho, J., Sendra, J.R.: A delineability-based method for computing critical sets of algebraic surfaces. J. Symb. Comput. 42(6), 678–691 (2007)
2. Bank, B., Giusti, M., Heintz, J., Pardo, L.-M.: Generalized polar varieties and efficient real elimination procedure. Kybernetika 40(5), 519–550 (2004)
3. Basu, S., Pollack, R., Roy, M.-F.: Algorithms in real algebraic geometry. Springer, Heidelberg (2003)
4. Benedetti, R., Risler, J.-J.: Real algebraic and semi-algebraic sets. Actualités Mathématiques, Hermann (1990)
5. Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In: Brakhage, H. (ed.) GI Fachtagung 1975. LNCS, vol. 33, pp. 515–532. Springer, Heidelberg (1975)
6. Corvez, S., Rouillier, F.: Using computer algebra tools to classify serial manipulators. In: Winkler, F. (ed.) ADG 2002. LNCS (LNAI), vol. 2930, pp. 31–43. Springer, Heidelberg (2004)
7. Everett, H., Lazard, D., Lazard, S., Safey El Din, M.: The topology of the Voronoi diagram of three lines in $\mathbb{R}^3$. In: Proceedings of Symposium on Computational Geometry. ACM Press, South-Korea (2007)
8. Faugère, J.-C.: Gb/FGb, `http://fgbrs.lip6.fr`

9. Fulton, W.: Intersection Theory. Ergebnisse der Mathematik und ihrer Grenzgebiete, vol. 2. Springer, Heidelberg (1984)
10. Gao, X.-S., Hou, X.-R., Tang, J., Cheng, H.-F.: Complete Solution Classification for the Perspective-Three-Point Problem. IEEE Trans. Pattern Anal. Mach. Intell. 25(8), 930–943 (2003)
11. Giusti, M., Heintz, J., Morais, J.-E., Morgenstern, J., Pardo, L.-M.: Straight-line programs in geometric elimination theory. Journal of Pure and Applied Algebra 124, 101–146 (1998)
12. Giusti, M., Lecerf, G., Salvy, B.: A Gröbner free alternative for polynomial system solving. Journal of Complexity 17(1), 154–211 (2001)
13. Grigoriev, D., Vorobjov, N.: Solving systems of polynomials inequalities in subexponential time. Journal of Symbolic Computation 5, 37–64 (1988)
14. Jelonek, Z., Kurdyka, K.: On asymptotic critical values of a complex polynomial. Journal für die Reine und Angewandte Mathematik 565, 1–11 (2003)
15. Lazard, D.: Quantifier elimination: optimal solution for two classical examples. Journal of Symbolic Computation 5(1-2), 261–266 (1988)
16. Lazard, D., Rouillier, F.: Solving parametric polynomial systems. Journal of Symbolic Computation 42, 636–667 (2007)
17. Lecerf, G.: Kronecker magma package for solving polynomial systems, `http://www.math.uvsq.fr/~lecerf/software/`
18. Lax, A., Lax, P.: On sums of squares. Linear Algebra App. 20, 71–75 (1978)
19. Lecerf, G.: Computing the equidimensional decomposition of an algebraic closed set by means of lifting fibers. Journal of Complexity 19(4), 564–596 (2003)
20. Kurdyka, K., Orro, P., Simon, S.: Semi-algebraic Sard's theorem for generalized critical values. Journal of differential geometry 56, 67–92 (2000)
21. Rouillier, F.: RS, RealSolving, `http://fgbrs.lip6.fr`
22. Rouillier, F.: Solving zero-dimensional systems through the Rational Univariate Representation. AAECC Journal 9(5), 433–461 (1999)
23. Rouillier, F., Zimmermann, P.: Efficient isolation of polynomial real roots. Journal of Computational and Applied Mathematics 162(1), 33–50 (2003)
24. Safey El Din, M.: RAGLib (Real Algebraic Geometry Library) (September 2007), `http://www-spiral.lip6.fr/~safey/RAGLib`
25. Safey El Din, M., Schost, É.: Polar varieties and computation of one point in each connected component of a smooth real algebraic set. In: Proceedings of the 2003 international symposium on Symbolic and algebraic computation, pp. 224–231. ACM Press, New York (2003)
26. Safey El Din, M., Schost, É.: Properness defects of projections and computation of one point in each connected component of a real algebraic set. Discrete and Computational Geometry 32(3), 417–430 (2004)
27. Safey El Din, M.: Generalized critical values and solving polynomial inequalities. In: Proceedings of International Conference on Polynomial Systems (2004)
28. Safey El Din, M.: Testing sign conditions on a multivariate polynomial and applications. Mathematics in Computer Science, Special issue on Algorithms and Complexity 1(1), 177–207 (2007)
29. Schost, E.: Computing Parametric Geometric Resolutions. Journal of Applicable Algebra in Engineering, Communication and Computing 13(5), 349–393 (2003)
30. Schweighofer, M.: Global optimization of polynomials using gradient tentacles and sums of squares. SIAM Journal on Optimization 17(3), 920–942 (2006)

Which Symmetric Homogeneous Polynomials Can Be Proved Positive Semi-definite by Difference Substitution Method?

Liangyu Chen and Zhenbing Zeng

Software Engineering Institute, East China Normal University
Shanghai, China
{lychen,zbzeng}@sei.ecnu.edu.cn
http://www.sei.ecnu.edu.cn

Abstract. Recently a method based on substitution of difference of variables has been developed by Yang [12] for verifying the positive semi-definiteness of homogeneous polynomials. In this paper, we investigate the structure of the cone formed by all symmetric homogeneous polynomials whose positive semi-definiteness can proven by difference substitution method.

Keywords: Homogeneous symmetric polynomial, Positive Semi-Definiteness, Difference Substitution.

1 Introduction

Let R^n be the n-dimensional linear space and $R^n_{\geq 0}$ be the following subset of R^n:

$$R^n_{\geq 0} = \{(x_1, x_2, \cdots, x_n) | x_1 \geq 0, x_2 \geq 0, \cdots, x_n \geq 0\}.$$

We call $R^n_{\geq 0}$ the positive quadrant of R^n. Let $S^d_n \subset R[x_1, x_2, \cdots, x_n]$ be the set of all symmetric homogeneous polynomials (also called forms) of degree d. G.Polya [7] proved that if a form $f(x_1, x_2, \cdots, x_n)$ is strictly positive in $R^n_{\geq 0}$, then the coefficients of the following expression

$$\texttt{expand}((x_1, x_2, \cdots, x_n)^N f(x_1, x_2, \cdots, x_n))$$

are all positive if N is sufficiently large, where `expand` is the function that distribute products over sums, as defined in Maple. But this method leads to a rapid increase of computation complexity if it is used to check the positive semi-definiteness (PSD). Huang et al. in [8] suggested a method to construct a family of PSD ternary polynomials $f_1^{(d)}, f_2^{(d)}, \cdots, f_{k_d}^{(d)}$ (called Schur Basis) of degree d such that each ternary polynomial $f(x_1, x_2, x_3)$ can be denoted as a linear combination (called Schur Partition) of $f_i^{(d)} (i = 1, 2, \cdots, k_d)$ uniquely:

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 57–71, 2008.
© Springer-Verlag Berlin Heidelberg 2008

$$f = a_1 f_1^{(d)} + a_2 f_2^{(d)} + \cdots + a_{k_d} f_{k_d}^{(d)}.$$

Recently Schur Partition is generalized to forms with n variables in [2]. It is obvious that if the Schur Partition of a given polynomial has no negative coefficient then the polynomial must be PSD. There are examples to show that the reverse is not true, that is, a PSD form may have some negative coefficients in its Schur Partition. In this sense, Schur Partition provides a heuristic method for checking PSD of forms. Another easy heuristic method for checking PSD for symmetric polynomials is based on one easy observation and another not-so-easy observation as following. The easier one is that $f(x_1, x_2, \cdots, x_n)$ is PSD in the positive quadrant if and only if $f(x_1, x_2, \cdots, x_n) \geq 0$ for all $x_1, x_2, \cdots, x_n$ with $0 \leq x_1 \leq x_2 \leq \cdots \leq x_n$. Meanwhile, not many people observed that for sufficiently many PSD forms $f(x_1, x_2, \cdots, x_n)$, the coefficients of the following expression

$$\texttt{expand}(\texttt{subs}(x_1 = u_1, x_2 = u_1 + u_2, \cdots, x_n = u_1 + u_2 + \cdots + u_n), f))$$

are positive, where `subs` stands for substitution, as defined in MAPLE. In [12], This naive method was extended to asymmetric (homogeneous) polynomials by Yang in the following way: To each permutation $\sigma = (i_1, i_2, \cdots, i_n)$ of integers $1, 2, \cdots, n$, we associate the *difference substitution* S_σ defined by

$$S_\sigma = \{x_{i_1} = u_1, x_{i_1} = u_1 + u_2, \cdots, x_{i_n} = u_1 + u_2 + \cdots + u_n\}$$

and calculate $\texttt{expand}(\texttt{subs}(S_\sigma, f(x_1, x_2, \cdots, x_n))$. If all coefficients of such expressions are positive, for all $n!$ permutations, then the original polynomial f is PSD. If all coefficients of these expressions are negative, for at least one substitution, then f is not PSD. Otherwise, for some substitutions the coefficients are neither all-positive nor all-negative. In the third case, do the difference substitutions for these branches recursively. This method is called "successive difference substitution". Yang actually proved some quite complicated inequalities using this method.

In this paper we discuss the structure of the set D_n^d of positive semi-definite homogeneous symmetric polynomials, which can be proven by difference substitution method, the measure of the cone in the whole positive semi-definite polynomials, the relation between Schur Partition and difference substitution, and the way to enlarge the convex cone for proving more polynomial inequalities.

The remainder of this paper is organized as follows. In Section 2 we introduce the general notations. In Section 3, we prove that the D_n^d is a finitely generated cone, and give a procedure SolvExtr for calculating the extremal rays of the cones based on Chernikova's algorithm. In Section 4, we show the process of calculating extreme rays about D_4^4. In Section 5, we will show that all PSD forms which Schur Partition has no negative coefficients are contained in D_n^d. In Section 6, we give a method to extend the cone generated by difference substitution. Section 7 is the conclusion.

2 Notation

Assume in this paper that $n \in N, n \geq 2$ and the variables are $x_1, x_2, \cdots, x_n$.

Notation 1. Let $\sigma_1, \sigma_2, \cdots, \sigma_n \in R[x_1, x_2, \cdots, x_n]$ be elementary symmetric polynomials. They are viewed as:

$$\sigma_k = \sum_{1 \leq i_1 < i_2 < \cdots < i_k \leq n} x_{i_1} x_{i_2} \cdots x_{i_k}, k = 1, 2, \cdots, n.$$

Notation 2. It is well known that every homogeneous symmetric polynomial $f \in S_n^d$ can be denoted as a combination $F(\sigma_1, \sigma_2, \cdots, \sigma_n)$, where:

$$F(\sigma_1, \sigma_2, \cdots, \sigma_n) = \sum a_{d_1, d_2, \cdots, d_n} {\sigma_1}^{d_1} {\sigma_2}^{d_2} \cdots {\sigma_n}^{d_n},$$

$$d_1, d_2, \cdots, d_n \geq 0, d_1 + 2d_1 + \cdots + nd_n = d.$$

Then we define a list $\Sigma(n, d)$ whose element is a monomial ${\sigma_1}^{d_1} {\sigma_2}^{d_2} \cdots {\sigma_n}^{d_n}$, which satisfied the conditions

$$d_1, d_2, \cdots, d_n \geq 0,$$
$$d_1 + 2d_1 + \cdots + nd_n = d.$$

Also we mark p as the cardinality of $\Sigma(n, d)$.

Notation 3. A polyhedral cone $C \subset R^n$ is finitely generated, if it is generated by a finite set of vectors, i.e., if it has the form

$$\begin{aligned} C &= cone(\alpha_1, \cdots, \alpha_r) \\ &= \{x | x = \sum_{j=1}^{r} \mu_j \alpha_j, \mu_j \geq 0, \quad j = 1, \cdots, r\}, \end{aligned}$$

where $\alpha_1, \cdots, \alpha_r$ are some vectors in R^n, and r is a positive integer [1].
A cone C is convex if

$$\alpha x + (1 - \alpha) y \in C, \forall x, y \in C, \forall \alpha \in [0, 1].$$

A vector or point $x \in C$ is said to be an extreme point of C if there do not exist vectors $y \in C$ and $z \in C$, with $y \neq x$ and $z \neq x$, and a scalar $\alpha \in (0, 1)$ such that $x = \alpha y + (1 - \alpha) z$. Or equivalently, x can not be expressed as a combination of convex vectors in C, all of which are different from x.

Notation 4. Let C-PSD denote the set of ternary positive semi-definite homogeneous symmetric polynomials where the coefficients of final expanding expressions under *Schur Partition* are non-negative. Similarly, we use D-PSD to denote the set corresponding to *Difference Substitution*.

3 D-PSD Cone

In this section, we will describe the D-PSD topological structure.

Theorem 1. D_n^d is a finitely generated convex cone.

Proof: In Notation 2.2, we define $\Sigma(n,d) = [s_1, s_2, \cdots, s_p]$ whose elements are ordered by lexicographic term ordering. We can get the value of p, which is the cardinality of $\Sigma(n,d)$, by calculating the coefficients of z^d in expansion of the following series

$$(1 + z + z^2 + \cdots)(1 + z^2 + z^4 + \cdots) \cdots (1 + z^n + z^{2n} + \cdots).$$

We know that every symmetric polynomial $f \in S_n^d$ can be transformed into a combination of $s_1, s_2, \cdots, s_p$. Conversely, given an array $(\alpha_1, \cdots, \alpha_p) \in R^p$, we can find a polynomial $f(s_1, s_2, \cdots, s_p) = \alpha_1 s_1 + \alpha_2 s_2 + \cdots + \alpha_p s_p$. Substituting the elementary symmetric polynomials σ_k into $f(s_1, s_2, \cdots, s_p)$, we can get a new homogeneous symmetric polynomial $F(x_1, x_2, \cdots, x_n) \in S_d^n$, whose coefficients are combinations of $\alpha_1, \alpha_2, \cdots, \alpha_p$. So it can build up an one-to-one correspondence between linear space R^p and S_n^d.

We use difference substitution to transform $F(x_1, x_2, \cdots, x_n)$ into another polynomial

$$\phi(u_1, u_2, \cdots, u_n) = \sum_{k=1}^{q} L_k(\alpha_1, \alpha_2, \cdots, \alpha_p) U_k,$$

where every $U_k (k = 1, 2, \cdots, q)$ is a monomial of $u_1^{i_1} u_2^{i_2} \cdots u_n^{i_n}$ with $i_1, i_2, \cdots, i_n \geq 0$ and $i_1 + i_2 + \cdots + i_n = d$, and every $L_k (k = 1, 2, \cdots, n)$ is the linear combination of $\alpha_1, \alpha_2, \cdots, \alpha_p$.

Since p is the cardinality of

$$\{(d_1, d_2, \cdots, d_n) | d_1, d_2, \cdots, d_n \geq 0; d_1 + 2d_2 + \cdots + nd_n = d\}$$

and q is the cardinality of

$$\{(i_1, i_2, \cdots, i_n) | i_1, i_2, \cdots, i_n \geq 0; i_1 + i_2 + \cdots + i_n = d\},$$

it can easily be proven that $q > p$ when $n, d > 1$. We also list some values of p and q in Table 1.

Thus, $\phi(u_1, u_2, \cdots, u_n)$ corresponding to the point coordinates $(\alpha_1, \alpha_2, \cdots, \alpha_n) \in R^p$, belongs to D-PSD if and only if $(\alpha_1, \alpha_2, \cdots, \alpha_n)$ satisfies by the following group of inequalities:

$$L_k(\alpha_1, \alpha_2, \cdots, \alpha_p) \geq 0, \quad (k = 1, 2, \cdots, q).$$

From the Definition 3, we obtain that the semi-algebraic set of L_k is a polyhedral convex cone in R^p. Moreover, the cone is generated by finite half lines from

original point. The points on half lines are all the extreme points of polyhedral convex cone. We mark the cone of D-PSD as C_1.

To avoid confusion, we use a vector $\xi = (\xi_1, \xi_2, \cdots, \xi_p)$ to represent the whole points of a half line defined by the original point and the point $(\xi_1, \xi_2, \cdots, \xi_p)$. So

$$D_n^d = \{\alpha_1 s_1 + \alpha_2 s_2 + \cdots + \alpha_p s_p | (\alpha_1, \alpha_2, \cdots, \alpha_p) \in C_1\},$$

is a polyhedral convex cone in S_n^d. Furthermore, one can deduce that $\alpha_1 s_1 + \alpha_2 s_2 + \cdots + \alpha_p s_p$ is an extreme ray of D_n^d if and only if $(\alpha_1, \alpha_2, \cdots, \alpha_p)$ is an extreme point of C_1. □

From the above proof, provided every $\alpha_k \geq 0 \quad (k = 1, 2, \cdots, p)$, the polynomial $F = \alpha_1 s_1 + \alpha_2 s_2 + \cdots + \alpha_p s_p$ can be proven to be positive semi-definite easily by checking the sign of coefficients in the new polynomial under difference substitution. So we get $R_{\geq 0}^p \subset C_1$. This represents that the measure $\mu(C_1 \cap S)$ of intersection between the cone C_1 and the unit ball S in R^p is larger than zero.

It is well known that a polyhedral convex cone has two dual representations: halfspaces and extremal rays. To transform one form into the other, Chernikova's algorithm [3,4,5] is often used. In *Theorem 2*, we describe the outline of *General Chernikova's Algorithm* which does not limit in non-negative domain. More details can be found in [3,4,5,6,10,11,9].

Theorem 2. (General Chernikova's Algorithm)
Let $C = \{x | Ax \geq 0\}$ be a polyhedral cone where A is an $m \times n$ matrix and $H = \{x | cx \geq 0\}$ be a halfspace of R^n. Chernikova's algorithm calculates the extremal rays of cone C in an incremental manner. The target cone is pruned recursively as follows:

$$C_0 = R^n,$$
$$C_k = C_{k-1} \cap H_k \quad 1 \leq k \leq m.$$

Let $Q = \{r_1, r_2, \cdots, r_s\}$ be the irredundant set of unidirectional or extremal rays and $E = \{z_1, z_2, \cdots, z_t\}$ be the irredundant set of bidirectional rays of C. Let Q' and E' denote the unidirectional and bidirectional set associated of $C \cap H$ respectively.

Suppose $E_0 = \{e_1, e_2, \cdots, e_n\}$ is the set of canonical basis vectors and $Q_0 = \varnothing$. The result Q of C_m has the final extremal rays exactly. Let $\cdot$ denote the dot product between vectors. In each step, we may encounter two following cases:

Case A: If there exists z_k satisfying $c \cdot z_k \neq 0$, then it does the following operations

$$z'_k = \pm z_k \quad \& \quad c \cdot z'_k > 0,$$
$$i \neq k \quad \& \quad z'_i = \lambda z_i + \mu z_k \quad \& \quad \lambda > 0 \quad \& \quad c \cdot z'_i = 0,$$
$$r'_j = \lambda r_j + \mu z_k \quad \& \quad \lambda > 0 \quad \& \quad c \cdot r'_j = 0,$$
$$Q' = \{r'_1, r'_2, \cdots, r'_s, z'_k\},$$
$$E' = \{z'_1, \cdots, z'_{k-1}, z'_{k+1}, \cdots, z'_t\}.$$

Case B: If all z_k satisfy $c \cdot z_k = 0$, the bidirectional rays E remain unchangeably. The new extremal rays Q' are calculated by

$$Q' = Q^{=} \cup Q^{>} \cup \overline{Q},$$

where

$$\begin{aligned} Q^{=} &= \{r | r \in Q, r \cdot c = 0\}, \\ Q^{>} &= \{r | r \in Q, r \cdot c > 0\}, \\ Q^{<} &= \{r | r \in Q, r \cdot c < 0\}, \\ \overline{Q} &= \{r | r \cdot c = 0, r = \lambda r_1 + \mu r_2, \\ &\quad (r_1, r_2) \in Q^{>} \times Q^{<}, \lambda > 0\}. \end{aligned}$$

It is noted that the initial algorithm only got the non-negative solutions and generated many redundant unidirectional rays in computation, while Fernandez and Quinton [6] extended the original algorithm to get the general solutions by differentiating between unidirectional rays and bidirectional lines. Verge [11] proposed a new enhanced criterion to filter out redundant linear combinations. However, it is remarked that Chernikova's algorithm is inconvenient to be implemented with parallel computation as it is very complicated and non intuitive.

Theorem 3. Let $L_1, L_2, \cdots, L_q$ be a system of real homogeneous linear equations about variables $\alpha_1, \alpha_2, \cdots, \alpha_p$ and C be the coefficient matrix. Let C_1 be the a convex cone which is finitely generated by the system of inequalities $L_1 \geq 0, L_2 \geq 0, \cdots, L_q \geq 0$ in R^p and $extr(C_1)$ be the extreme points of cone C_1. Let the rank of coefficient matrix C be p, then for any $\xi = (\xi_1, \xi_2, \cdots, \xi_p) \in extr(C_1)$, it can construct a new matrix C' constituted by some row vectors from $L_1 \geq 0, L_2 \geq 0, \cdots, L_q \geq 0$ which satisfy $L_k(\xi) = 0$. The rank of coefficients matrix of C' must be $p - 1$.

Proof: It is obvious that for any $\xi \in extr(C_1)$, there exists at least one polynomial $L_k (k = 1, 2, \cdots, q)$ equal to zero at least. So we can assume that for $\xi \in extr(C_1)$,

$$\begin{aligned} L_k(\xi) &= 0, \quad (1 \leq k \leq m), \\ L_k(\xi) &> 0, \quad (m+1 \leq k \leq q), \end{aligned}$$

and the rank of coefficient matrix, which is constructed by $L_1, \cdots, L_m$, is smaller than $p-1$. Thus, we can get the basic solutions of equations $L_1 = 0, \cdots, L_m = 0$ as follows

$$\alpha_1, \cdots, \alpha_k, \alpha_j = c_{j,1}\alpha_1 + \cdots + c_{j,k}\alpha_k, 1 < k < p, k < j \leq p.$$

So assume the extreme point

$$\xi = (\xi_1, \cdots, \xi_k, c_{k+1,1}\xi_1 + \cdots + c_{k+1,k}\xi_k, \cdots, c_{p,1}\xi_1 + \cdots + c_{p,k}\xi_k),$$

where all $L_1(\xi), \cdots, L_m(\xi)$ are zero. Without loss of generality, we randomly select two points $\xi', \xi'' \in R^k$,

$$\xi' = (\xi'_1, \cdots, \xi'_k, c_{k+1,1}\xi'_1 + \cdots + c_{k+1,k}\xi'_k, \cdots, c_{p,1}\xi'_1 + \cdots + c_{p,k}\xi'_k),$$
$$\xi'' = (\xi''_1, \cdots, \xi''_k, c_{k+1,1}\xi''_1 + \cdots + c_{k+1,k}\xi''_k, \cdots, c_{p,1}\xi''_1 + \cdots + c_{p,k}\xi''_k),$$

which can satisfy the equations $L_1 = 0, \cdots, L_m = 0$. Especially, we can select

$$\xi'_1 = \xi_1 + \eta_1\epsilon, \cdots, \xi'_k = \xi_k + \eta_k\epsilon,$$
$$\xi''_1 = \xi_1 - \eta_1\epsilon, \cdots, \xi''_k = \xi_k - \eta_k\epsilon,$$

where ϵ is a random positive real number and $(\eta_1, \cdots, \eta_k)$ is an unit normal vector of $(\xi_1, \cdots, \xi_k)$. Therefore we can get

$$\xi', \xi'' \in R^p \setminus \{c\xi | c > 0\},$$
$$\xi' + \xi'' = 2\xi,$$
$$L_1(\xi') = \cdots = L_m(\xi') = 0,$$
$$L_1(\xi'') = \cdots = L_m(\xi'') = 0.$$

To complete the proving of this theorem, we need to prove the point $\xi', \xi'' \in C_1$. To ξ', ξ'', there remain $2(q-m)$ linear functions

$$L_{m+1}(\xi'), \cdots, L_q(\xi'), L_{m+1}(\xi''), \cdots, L_q(\xi'').$$

Based on the above hypotheses, we have

$$L_{m+1}(\xi) > 0, \cdots, L_q(\xi) > 0.$$

So according to the principle of sign preservation, we can select a sufficiently small non-zero number ϵ to make

$$L_{m+1}(\xi') > 0, \cdots, L_q(\xi') > 0,$$
$$L_{m+1}(\xi'') > 0, \cdots, L_q(\xi'') > 0.$$

Thus we have $\xi, \xi', \xi'' \in C_1$ synchronously, and $2\xi = \xi' + \xi''$. However, we know already that ξ is an extreme point of cone $C1$, then it is a contraction. So we complete this proof. □

Based on *Theorem 1* and *Theorem 3*, we give an algorithm *SolvExtr* to calculate extreme rays of cone C_1. *SolvExtr* is arranged to four steps.

Step 1. Let S be $\{L_1, L_2, \cdots, L_q\}$ with variables $\alpha_1, \cdots, \alpha_p$ and the set of extreme points ExtSet be null initially. Calculate the rank of coefficient matrix of S. If the rank r is less than $p-1$, then return the incorrectness about the inequalities definition of cone and end the algorithm.
Step 2. If the rank $r \geq p-1$, it can extract $p-1$ row vectors from the coefficient matrix and construct a new $(p-1) \times p$ matrix. we arrange the new matrixes into a list

$$M = (M_1, M_2, \cdots, M_k),$$

where each element is a $(p-1) \times p$ matrix and $\binom{k=q}{p-1}$.

Step 3. Calculate the extreme points until $j = k$ (Implement in parallel computation conveniently).

Step 3.1. Calculate the rank of all elements in M. For $M_j (1 \leq j \leq k)$, we can get the rank r_j. If $r_j = p - 1$ denotes that the dimension of base of sl_j is one, we can get the base as $(\alpha_1\alpha, \alpha_2\alpha, \cdots, \alpha_p\alpha)$ where α is a parameter.

Step 3.2. Substitute the base vector into $S \setminus M_j$, we can get $c_p\alpha, c_{p+1}\alpha, \cdots, c_q\alpha$. If each of $c_p, c_{p+1}, \cdots, c_q$ is non-negative, thus

$$ExtSet = ExtSet \cup \{(\alpha_1, \alpha_2, \cdots, \alpha_p)\}.$$

If every element of $c_p, c_{p+1}, \cdots, c_q$ is non-positive, then

$$ExtSet = ExtSet \cup \{(-\alpha_1, -\alpha_2, \cdots, -\alpha_p)\}.$$

Step 4. Return the set $ExtSet$ of extreme points.

Since $\binom{k=q}{p-1}$ is finite, the algorithm *SolvExtr* can terminate normally. We can sum up the above process to calculate all extreme points of a polyhedral convex cone D_n^d which is constituted by many positive semi-definite homogeneous symmetric polynomials under difference substitution for given $n, d \geq 0$.

DevSubExp Algorithm
Input: $n, d \geq 0$.
Output: the homogeneous coordinates of extreme points of D_n^d. **Step 1:** construct the list $\Sigma(n, d) = [s_1, s_2, \cdots, s_p]$.
Step 2: construct the semi-algebraic set of $L_k \ \ (k = 1, 2, \cdots, q)$.
Step 3: calculate the extreme points of the polyhedral convex cone constituted by $L_k \ \ (k = 1, 2, \cdots, q)$ based on *SolvExtr Algorithm* or *General Chernikova's Algorithm*.
Step 4: calculate the extreme points of D_n^d.

4 An Example for DevSubExp

In this section, we will show how to calculate the extreme points of D_4^4 based on *DevSubExp Algorithm*.

Firstly, we have

$$n = 4, d = 4.$$

According to the definition of $\Sigma(n, d)$, we get

$$\Sigma(4, 4) = [\sigma_1^4, \sigma_1^2\sigma_2, \sigma_1\sigma_3, \sigma_2^2, \sigma_4]$$

where $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ are the elementary symmetric polynomials.

The cardinality of $\Sigma(4, 4)$ is five, so we construct a polynomial

$$F = \alpha_1\sigma_1^4 + \alpha_2\sigma_1^2\sigma_2 + \alpha_3\sigma_1\sigma_3 + \alpha_4\sigma_2^2 + \alpha_5\sigma_4.$$

By substituting the elementary symmetric polynomials and the substitution operators

$$\begin{aligned}
&\sigma_1 = x_1 + x_2 + x_3 + x_4,\\
&\sigma_2 = x_1x_2 + x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 + x_3x_4,\\
&\sigma_3 = x_1x_2x_3 + x_1x_2x_4 + x_1x_3x_4 + x_2x_3x_4,\\
&\sigma_4 = x_1x_2x_3x_4,\\
&x_1 = u_1,\\
&x_2 = u_1 + u_2,\\
&x_3 = u_1 + u_2 + u_3,\\
&x_4 = u_1 + u_2 + u_3 + u_4,
\end{aligned}$$

the polynomial $F(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$ is transformed into $\phi(u_1, u_2, u_3, u_4)$. We can get the coefficients of $\phi(u_1, u_2, u_3, u_4)$ about $u_1^{d_1}u_2^{d_2}u_3^{d_3}u_4^{d_4}$ by lexicographic term ordering to construct the semi-algebraic set C_1 and calculate the hyperplanes L_k

$$\begin{aligned}
L_1 &= \alpha_1 + 36\alpha_2 + 16\alpha_3 + 96\alpha_4 + 256\alpha_5,\\
L_2 &= 3\alpha_1 + 108\alpha_2 + 48\alpha_3 + 288\alpha_4 + 768\alpha_5,\\
&\cdots\cdots,\\
L_{34} &= \alpha_4 + 8\alpha_5,\\
L_{35} &= \alpha_5.
\end{aligned}$$

Executing the *SolvExtr Algorithm*, we can get the extreme points of $C1$

$$\begin{aligned}
ExtSet = [&(1,0,0,0,0), (\frac{16}{15}, 0, -\frac{1}{15}, 0, 0), (\frac{5}{3}, -\frac{4}{9}, -\frac{2}{3}, \frac{5}{9}, -\frac{1}{9}),\\
&(\frac{64}{45}, -\frac{16}{45}, -\frac{16}{45}, \frac{16}{45}, -\frac{1}{15}), (\frac{8}{5}, -\frac{4}{15}, -\frac{7}{10}, \frac{7}{15}, -\frac{1}{10}),\\
&(\frac{6}{5}, \frac{1}{10}, -\frac{3}{10}, 0, 0), (0, 1, -4, 3, 0)].
\end{aligned}$$

With the polynomial $F(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$. The seven extremal rays of D_4^4 are

$$\begin{aligned}
E_1 &= x_1x_2x_3x_4,\\
E_2 &= \frac{1}{15}(x_1^2x_2x_3 + x_1^2x_2x_4 + x_1^2x_3x_4 + x_1x_2^2x_3 + x_1x_2^2x_4 + x_1x_2x_3^2 + x_1x_2x_4^2\\
&\quad + x_1x_3^2x_4 + x_1x_3x_4^2 + x_2^2x_3x_4 + x_2x_3^2x_4 + x_2x_3x_4^2 - 12x_1x_2x_3x_4),
\end{aligned}$$

$$
\begin{aligned}
E_3 = \frac{1}{9}(&x_1^4 - x_1^3x_2 - x_1^3x_3 - x_1^3x_4 + x_1^2x_2x_3 + x_1^2x_2x_4 + x_1^2x_3x_4 - x_1x_2^3 + x_1x_2^2x_3 \\
&+x_1x_2^2x_4 + x_1x_2x_3^2 - 3x_1x_2x_3x_4 + x_1x_2x_4^2 - x_1x_3^3 + x_1x_3^2x_4 + x_1x_3x_4^2 \\
&-x_1x_4^3 + x_2^4 - x_2^3x_3 - x_2^3x_4 + x_2^2x_3x_4 - x_2x_3^3 + x_2x_3^2x_4 + x_2x_3x_4^2 - x_2x_4^3 \\
&+x_3^4 - x_3^3x_4 - x_3x_4^3 + x_4^4), \\
E_4 = \frac{1}{45}(&3x_1^4 - 4x_1^3x_2 - 4x_1^3x_3 - 4x_1^3x_4 + 2x_1^2x_2^2 + 4x_1^2x_2x_3 + 4x_1^2x_2x_4 + 2x_1^2x_3^2 \\
&+4x_1^2x_3x_4 + 2x_1^2x_4^2 - 4x_1x_2^3 + 4x_1x_2^2x_3 + 4x_1x_2^2x_4 + 4x_1x_2x_3^2 - 24x_1x_2x_3x_4 \\
&+4x_1x_2x_4^2 - 4x_1x_3^3 + 4x_1x_3^2x_4 + 4x_1x_3x_4^2 - 4x_1x_4^3 + 3x_2^4 - 4x_2^3x_3 - 4x_2^3x_4 \\
&+2x_2^2x_3^2 + 4x_2^2x_3x_4 + 2x_2^2x_4^2 - 4x_2x_3^3 + 4x_2x_3^2x_4 + 4x_2x_3x_4^2 - 4x_2x_4^3 + 3x_3^4 \\
&-4x_3^3x_4 + 2x_3^2x_4^2 - 4x_3x_4^3 + 3x_4^4), \\
E_5 = &-2x_1^2x_4^2 - 2x_2^2x_3^2 - 2x_1^2x_2^2 - 2x_1^2x_3^2 + x_3^3x_4 + x_1x_4^3 + x_2x_4^3 + x_3x_4^3 + x_1^3x_2 \\
&+x_1^3x_3 + x_1^3x_4 + x_2^3x_3 + x_2^3x_4 + x_1x_3^3 + x_2x_3^3 + x_1x_2^3 - 2x_2^2x_4^2 - 2x_3^2x_4^2, \\
E_6 = \frac{1}{10}(&x_1^2x_2^2 - x_1^2x_2x_3 - x_1^2x_2x_4 + x_1^2x_3^2 - x_1^2x_3x_4 + x_1^2x_4^2 - x_1x_2^2x_3 - x_1x_2^2x_4 \\
&-x_1x_2x_3^2 - x_1x_2x_4^2 - x_1x_3^2x_4 - x_1x_3x_4^2 + x_2^2x_3^2 - x_2^2x_3x_4 + x_2^2x_4^2 - x_2x_3^2x_4 \\
&-x_2x_3x_4^2 + x_3^2x_4^2 + 6x_1x_2x_3x_4), \\
E_7 = \frac{1}{30}(&3x_1^4 - 2x_1^3x_2 - 2x_1^3x_3 - 2x_1^3x_4 - 2x_1^2x_2^2 + 3x_1^2x_2x_3 + 3x_1^2x_2x_4 - 2x_1^2x_3^2 \\
&+3x_1^2x_3x_4 - 2x_1^2x_4^2 - 2x_1x_2^3 + 3x_1x_2^2x_3 + 3x_1x_2^2x_4 + 3x_1x_2x_3^2 + 3x_1x_2x_4^2 \\
&-2x_1x_3^3 + 3x_1x_3^2x_4 + 3x_1x_3x_4^2 - 2x_1x_4^3 + 3x_2^4 - 2x_2^3x_3 - 2x_2^3x_4 - 2x_2^2x_3^2 \\
&+3x_2^2x_3x_4 - 2x_2^2x_4^2 - 2x_2x_3^3 + 3x_2x_3^2x_4 + 3x_2x_3x_4^2 - 2x_2x_4^3 + 3x_3^4 - 2x_3^3x_4 \\
&-2x_3^2x_4^2 - 2x_3x_4^3 + 3x_4^4 - 12x_1x_2x_3x_4).
\end{aligned}
$$

We also calculate the extremal rays for different p and q, listed in Table 1.

Table 1. Extremal Rays

(n,d)	(p,q)	Count of Extremal Rays
(3,3)	(3,10)	3
(3,4)	(4,15)	5
(3,5)	(5,21)	12
(3,6)	(7,28)	34
(3,7)	(8,36)	118
(4,3)	(3,20)	4
(4,4)	(5,35)	7
(4,5)	(6,56)	24

5 Relation of C-PSD and D-PSD

Let's consider ternary homogeneous symmetric polynomials. The variables are x_1, x_2, x_3, and the substitution operators are $x_1 = u_1, x_2 = u_1 + u_2, x_3 = u_1 +$

$u_2 + u_3$. Let D_3^d be the set of d-form ternary positive semi-definite homogeneous symmetric polynomials in S_3^d under difference substitution. It is easily deduced that

$$f, g \in D_3^d \Rightarrow f + g \in D_3^d,$$
$$f \in D_3^{d_1}, g \in D_3^{d_2} \Rightarrow f \cdot g \in D_3^{d_1+d_2}.$$

In [8], Huang gave an algorithm to partition a ternary positive semi-definite homogeneous symmetric polynomial into a linear combination of five bases. Though the algorithm is a sufficient condition for 3-ary d-forms to be positive semi-definite, it can be used to prove many non-trivial inequalities. Since both *Schur Partition* and *Difference Substitution* are useful tools for inequality proving, we want to discover the relation between the two methods. We have Theorem 4 as follows.

Theorem 4. For any given d, $C_3^d \subseteq D_3^d$.

Proof: The algorithm *Schur Partition* decomposes a ternary positive semi-definite homogeneous symmetric polynomial into a linear combination of five bases. The five bases are

$$\begin{aligned}
f_{0,1}^{(d)} &= x_1^{d-2}(x_1 - x_2)(x_1 - x_3) + x_2^{d-2}(x_2 - x_3) \\
&\quad (x_2 - x_1) + x_3^{d-2}(x_3 - x_1)(x_3 - x_2) \ \ (d \geq 2), \\
f_{0,2}^{(d)} &= x_1^{d-3}(x_2 + x_3)(x_1 - x_2)(x_1 - x_3) \\
&\quad + x_2^{d-3}(x_3 + x_1)(x_2 - x_3)(x_2 - x_1) \\
&\quad + x_3^{d-3}(x_1 + x_2)(x_3 - x_1)(x_3 - x_2) \ \ (d \geq 3), \\
f_{0,j}^{(d)} &= (x_1 + x_2 + x_3)^{n-2j}(x_1x_2 + x_2x_3 + x_3x_1)^{j-3} \\
&\quad (x_1 - x_2)^2(x_2 - x_3)^2(x_3 - x_1)^2 \ \ (3 \leq j \leq \lfloor d/2 \rfloor), \\
f_{0,\lfloor (d+2)/2 \rfloor}^{(d)} &= \begin{cases}
(x_2x_3)^{\frac{d-3}{2}}(x_2 + x_3)(x_1 - x_2)(x_1 - x_3) \\
+(x_3x_1)^{\frac{d-3}{2}}(x_3 + x_1)(x_2 - x_3)(x_2 - x_1) \\
+(x_1x_2)^{\frac{d-3}{2}}(x_1 + x_2)(x_3 - x_1)(x_3 - x_2) \\
\qquad (d \equiv 1 (mod 2), d \geq 5), \\
(x_2x_3)^{\frac{d-2}{2}}(x_1 - x_2)(x_1 - x_3) \\
+(x_3x_1)^{\frac{d-2}{2}}(x_2 - x_3)(x_2 - x_1) \\
+(x_1x_2)^{\frac{d-2}{2}}(x_3 - x_1)(x_3 - x_2) \\
\qquad (d \equiv 0 (mod 2), d \geq 4),
\end{cases} \\
f_{i,j}^{(d)} &= (x_1x_2x_3)^i f_{0,j}^{n-3i} \quad (1 \leq i \leq \lfloor \frac{d-2}{3} \rfloor).
\end{aligned}$$

For the first basis $f_{0,1}^{(d)}$, we substitute the substitution operators and obtain

$$\begin{aligned}
f_{0,1}^{(d)} &= u_1^{d-2}(u_2^2 + u_2u_3) + (u_1 + u_2)^{d-2}(-u_2u_3) + (u_1 + u_2 + u_3)^{d-2}(u_2u_3 + u_3^2) \\
&= u_1^{d-2}(u_2^2 + u_2u_3) + (u_1 + u_2 + u_3)^{d-2}u_3^2 + [(u_1 + u_2 + u_3)^{d-2} - (u_1 + u_2)^{d-2}]u_2u_3.
\end{aligned}$$

Because all $u_1, u_2, u_3 \geq 0$ and the coefficients of $[(u_1+u_2+u_3)^{d-2} - (u_1+u_2)^{d-2}]$ are non-negative, it is obvious that $f_{0,1}^{(d)} \in D_3^d$.

For the second basis $f_{0,2}^{(d)}$, we have

$$\begin{aligned} f_{0,2}^{(d)} &= u_1^{d-3}(2u_1u_2^2 + 2u_1u_2u_3 + 2u_2^3 + 3u_2^2u_3 + u_2u_3^2) - (u_1+u_2)^{d-3} \\ &\quad (2u_1u_2u_3 + u_2^2u_3 + u_2u_3^2) + (u_1+u_2+u_3)^{d-3}(2u_1u_2u_3 + 2u_1u_2^2 + u_2^2u_3 + u_2u_3^2) \\ &= u_1^{d-3}(2u_1u_2^2 + 2u_1u_2u_3 + 2u_2^3 + 3u_2^2u_3 + u_2u_3^2) + (u_1+u_2+u_3)^{d-3}(2u_1u_3^2) \\ &\quad +[-(u_1+u_2)^{d-3} + (u_1+u_2+u_3)^{d-3}](2u_1u_2u_3 + u_2^2u_3 + u_2u_3^2). \end{aligned}$$

Because the coefficients of $[(u_1+u_2+u_3)^{d-3} - (u_1+u_2)^{d-3}]$ are non-negative, it is obvious that $f_{0,2}^{(d)} \in D_3^d$.

For the third basis $f_{0,j}^{(d)} \quad (3 \leq \lfloor \frac{d}{2} \rfloor)$, these following propositions are true.

$$\begin{gathered} (x_1+x_2+x_3)^{d-2j} \ in D_3^{d-2j}, \\ (x_1x_2 + x_2x_3 + x_3x_1)^{j-3} \in D_3^{2j-6}, \\ (x_1-x_2)^2(x_2-x_3)^2(x_3-x_1)^2 \in D_3^6. \end{gathered}$$

Thus $f_{0,j}^{(d)} \in D_3^d$.

For the fourth basis $f_{0,\lfloor (d+2)/2 \rfloor}^{(d)}$, when $d \equiv 1(mod 2), d \geq 5$, let $m = \frac{d-3}{2}$, we have

$$\begin{aligned} f_{0,\lfloor (d+2)/2 \rfloor}^{(d)} &= (u_1+u_2)^m(u_1+u_2+u_3)^m(2u_1u_2^2 + 2u_1u_2u_3 + 2u_2^3 + 3u_2^2u_3 + u_2u_3^2) \\ &\quad +(u_1+u_2+u_3)^m u_1^m(-2u_1u_2u_3 - u_2^2u_3 - u_2u_3^2) \\ &\quad +u_1^m(u_1+u_2)^m(2u_1u_2u_3 + 2u_1u_2^2 + u_2^2u_3 + u_2u_3^2) \\ &= (u_1+u_2)^m(u_1+u_2+u_3)^m(2u_1u_2^2 + 2u_2^3 + 2u_2^2u_3) \\ &\quad +u_1^m(u_1+u_2)^m(2u_1u_2u_3 + 2u_1u_2^2 + u_2^2u_3 + u_2u_3^2) \\ &\quad +[(u_1+u_2)^m - (u_1)^m](u_1+u_2+u_3)^m(2u_1u_2u_3 + u_2^2u_3 + u_2u_3^2). \end{aligned}$$

when $d \equiv 0(mod 2), d \geq 4$, let $m = \frac{d-2}{2}$, we have

$$\begin{aligned} f_{0,\lfloor (d+2)/2 \rfloor}^{(d)} &= (u_1+u_2)^m(u_1+u_2+u_3)^m(u_2^2 + u_2u_3) \\ &\quad +(u_1+u_2+u_3)^m u_1^m(-u_2u_3) + u_1^m(u_1+u_2)^m(u_2u_3 + u_3^2) \\ &= (u_1+u_2)^m(u_1+u_2+u_3)^m u_2^2 + u_1^m(u_1+u_2)^m(u_2u_3 + u_3^2) \\ &\quad +[(u_1+u_2)^m - (u_1)^m](u_1+u_2+u_3)^m(2u_1u_2u_3 + u_2^2u_3 + u_2u_3^2). \end{aligned}$$

Because the coefficients of $[(u_1+u_2)^m - (u_1)^m]$ are non-negative, it is obvious that $f_{0,\lfloor (d+2)/2 \rfloor}^{(d)} \in D_3^d$.

Since the above four bases belong to D_3^d, we have $f_{i,j}^{(d)} \in D_3^d$ normally.

So for every ternary polynomial whose positive semi-definiteness can be proven by Schur Partition, it also can be proven by Difference Substitution. Thus we complete this proof. □

6 Extension of D-PSD Cone

It is remarked that the current D-PSD cone does not contain all positive semi-definite homogeneous symmetric polynomials. For example, we have the following 4-ary 4-form polynomial $T(x_1, x_2, x_3, x_4)$ which can not be proven under difference substitution in one round.

$$\begin{aligned} T = {} & 25x_1^4 - 42x_1^3x_2 - 42x_1^3x_3 - 42x_1^3x_4 + 43x_1^2x_2^2 + 24x_1^2x_2x_3 + 24x_1^2x_2x_4 \\ & +43x_1^2x_3^2 + 24x_1^2x_3x_4 + 43x_1^2x_4^2 - 42x_1x_2^3 + 24x_1x_2^2x_3 + 24x_1x_2^2x_4 + 24x_1x_2x_3^2 \\ & -42x_1x_2x_3x_4 + 24x_1x_2x_4^2 - 42x_1x_3^3 + 24x_1x_3^2x_4 + 24x_1x_3x_4^2 - 42x_1x_4^3 + 25x_2^4 \\ & +25x_4^4 - 42x_2^3x_3 - 42x_2^3x_4 + 43x_2^2x_3^2 + 24x_2^2x_3x_4 + 43x_2^2x_4^2 - 42x_2x_3^3 \\ & +24x_2x_3^2x_4 + 24x_2x_3x_4^2 - 42x_2x_4^3 + 25x_3^4 - 42x_3^3x_4 + 43x_3^2x_4^2 - 42x_3x_4^3. \end{aligned}$$

But we can prove the positive semi-definiteness of T by using two rounds of difference substitution. This means it can extend the D-PSD cone to approximate the set of all positive semi-definite homogeneous symmetric polynomials by using difference substitution successively.

Let's use C to denote the cone of a round of difference substitution and C' to be the cone of two rounds of difference substitution. In the following calculation, we will show the extension from C to C'.

In section 4, we get the extremal rays of S_4^4 and mark them as

$$Q = [E_1, E_2, E_3, E_4, E_5, E_6, E_7].$$

We construct a new homogeneous symmetric polynomial $F(x_1, x_2, x_3, x_4)$

$$F = (1 - m)E_5 + \frac{m}{6}(E_1 + E_2 + E_3 + E_4 + E_5 + E_6),$$

where m is a parameter. If $0 \leq m \leq 1$, it is easily proven that $F \geq 0$, thus F is contained in the current D-PSD cone. Since F is symmetric, so we can take the difference operators

$$\begin{gathered} x_1 = u_1, x_2 = u_1 + u_2, \\ x_3 = u_1 + u_2 + u_3, x_4 = u_1 + u_2 + u_3 + u_4 \end{gathered}$$

and get a new polynomial F' whose coefficients comprise m. If these coefficients are all non-negative, then the positive semi-definiteness of F is proven. So we can suppose all coefficients are non-negative and conclude that the maximal value of m is $\frac{60}{59}$. This means for every $m \in [0, \frac{60}{59}]$, F is positive since it is in D-PSD cone. For $m > \frac{60}{59}$, F is excluded of current D-PSD cone.

If we continue to do difference substitution, we need consider all possible orders between the variables since F' is asymmetric. There are $4! = 24$ variable orders. Suppose $u_1 \leq u_2 \leq u_3 \leq u_4$, we take the new difference operators

$$\begin{gathered} u_1 = v_1, u_2 = v_1 + v_2, \\ u_3 = v_1 + v_2 + v_3, u_4 = v_1 + v_2 + v_3 + v_4, \end{gathered}$$

into F' and get a polynomial $G(v_1, v_2, v_3, v_4)$. Let the coefficients of G be non-negative, then we can get the maximal value of m is $\frac{83700}{80083}$. By calculating all 24 variables orders, we get that the final maximal value of m is $\frac{30}{29}$. Through two rounds of difference substitution, the positive semi-definiteness of F with $m \in (\frac{60}{59}, \frac{30}{29}]$ is proven while the D-PSD cone is extended to a larger one. However, we now do not know what is the end point of expansion and topological structure of the final D-PSD cone.

7 Conclusions

In this paper, we have presented a plain method called Difference Substitution to verify the positive semi-definiteness of homogeneous symmetric polynomials. Several examples highlight the importance of this method. The topological structure of this method is proven as a finitely generated convex cone and the extremal rays can be calculated with *DevSubExp Algorithm* or *Chernikova's Algorithm*. An example of S_4^4 is also illustrated. Furthermore, our method can not only take the place of *Schur Partition* to prove the positive semi-definiteness of 3-ary n-forms but also prove more with higher dimension or degree. Additionally, the D-PSD cone is extended to a larger one, so as to prove more polynomials' positive semi-definiteness.

More work is needed on the following points. Both *DevSubExp Algorithm* and *Chernikova's Algorithm* encounter the problem of combinational explosion. Parallel computation may calculate faster for some special polynomial inequalities higher degrees or more variables. The extremal rays of extended D-PSD cone and the transformation of extremal rays may be considered carefully.

Finally, we should indicate that the successive difference substitution method we show in this paper is rather a heuristic one. We would like to express our appreciation to one of the referees who kindly provided the following example

$$f(x, y) = \left[a \cdot (x^2 + y^2) - (x + y)^2\right]^2,$$

whose positive semi-definiteness can not be verified by our method.

Acknowledgments. The work is supported in part by NKBRPC-2004CB318003 and NNSFC-10471044.

References

1. Bertsekas, D.P., Nedić, A., Ozdaglar, A.E.: Convex analysis and optimization, pp. 25–45. Athena Scientific and Tsinghua University Press (2006)
2. Chen, S.L., Yao, Y.: Schur Subspace of Real Symmetric Forms and Application. Acta Mathematica Sinica, Chinese Series 50, 1331–1348 (2007)
3. Chernikova, N.V.: Algorithm for finding a general formula for the non-negative solutions of a system of linear equations. U.S.S.R Computational Mathematics and Mathematical Physics 4, 151–158 (1964)

4. Chernikova, N.V.: Algorithm for finding a general formula for the non-negative solutions of a system of linear inequalities. U.S.S.R Computational Mathematics and Mathematical Physics 5, 228–233 (1965)
5. Chernikova, N.V.: Algorithm for discovering the set of all the solutions of a linear programming problem. U.S.S.R Computational Mathematics and Mathematical Physics 8, 282–293 (1968)
6. Fernández, F., Quinton, P.: Extension of Chernikova's algorithm for solving general mixed linear programming problems, Technical Report 437, IRISA-Rennes, France (1988)
7. Hardy, G.H., Littlewood, J.E., Pólya, G.: Inequalities, 2nd edn., pp. 57–59. Cambridge University Press, Cambridge (1952)
8. Huang, F.J., Chen, S.L.: Schur partition for symmetric ternary forms and readable proof to inequalities. In: ISSAC 2005, Beijing, China, pp. 185–192 (2005)
9. Rabl, T.: Volume calculation and estimation of parameterized integer polytopes, Diploma Thesis, Universität Passau, German (2006)
10. Wilde, D.K.: A library for doing polyhedral operations, Technical Report 785, IRISA-Rennes, France (1993)
11. Verge, H.L.: A note on Chernikova's algorithm, Technical Report 635, IRISA-Rennes, France (1994)
12. Yang, L.: Solving harder problems with lesser mathematics. In: Ju, C., Korea, S., Chu, S.-C., et al. (eds.) ATCM 2005, pp. 37–46. ATCM Inc., Blacksburg (2005)

Basis-Independent Polynomial Division Algorithm Applied to Division in Lagrange and Bernstein Basis

Manfred Minimair*

Department of Mathematics and Computer Science
Seton Hall University, 400 South Orange Avenue
South Orange, New Jersey 07079, USA
manfred@minimair.org
http://minimair.org

Abstract. Division algorithms for univariate polynomials represented with respect to Lagrange and Bernstein basis are developed. These algorithms are obtained by abstracting from the classical polynomial division algorithm for polynomials represented with respect to the usual power basis. It is shown that these algorithms are quadratic in the degrees of their inputs, as in the power basis case.

Keywords: polynomial division, Lagrange basis, Bernstein basis.

1 Introduction

Fundamental operations for polynomials represented with respect to bases other than the usual power basis are being intensely studied. Examples include computation of resultants and resultant matrices [8,16,22], gcds [10,12,17], generalized companion matrices [7,11,23,24], polynomial remainder sequences [6,14], and polynomial division [1,5,19]. Carrying out fundamental operations over alternative bases is motivated by the desire to avoid computational cost and numeric errors incurred by converting between different polynomial bases [9,15,20] and practical applications [3,4].

The current paper studies division of univariate polynomials represented with respect to the Lagrange basis as well as the Bernstein basis. Current algorithms for polynomial division over these bases ([1,2] and respectively [19]) proceed by setting up systems of linear equations for the coefficients of the quotient and remainder. Then solutions are computed by respectively using SVD and LU factorization. Since the sizes of these systems of equations are linear in the degrees of the input polynomials and the solution methods require a worst-case cubic number of arithmetic operations the worst-case complexity of these methods is expected to be cubic. However, it is well-known that the worst-case complexity of dividing polynomials represented with respect to the usual power basis is quadratic.

* Supported by the NSF grant CCF 0430741.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 72–86, 2008.
© Springer-Verlag Berlin Heidelberg 2008

For example, [21] shows that for polynomials represented in power basis division by the classical division algorithm is $O((n+1)(m-n+1))$, where m and n respectively is the degree of the dividend and divisor. Therefore there seems to be a gap in the current theory. It seems that an appropriate generalization of the classical division algorithm that exhibits quadratic worst-case complexity for Lagrange and Bernstein basis is lacking. The current paper addresses this gap by providing a generalized division algorithm (Section 2) and by showing quadratic worst-case complexities for the number of arithmetic operations required for division over Lagrange (Theorem 15) and Bernstein (Theorem 26) basis.

Before we conclude this introductory section, we discuss some other related works, besides [1,2,19]. The work [5] uses matrix techniques to divide polynomials over orthogonal bases which obviously do not include the Lagrange and Bernstein bases. Therefore these techniques seem not immediately applicable to dividing polynomials over Lagrange and Bernstein basis. These techniques are based on the Cayley-Hamilton Theorem and require generalized companion matrices, the comrade matrices. But recently, some generalized companion matrices over the Lagrange and Bernstein basis have been developed [11,23,24]. It would be interesting to attempt to generalize the techniques from [5] to these companion matrices. However it is not obvious if a resulting matrix-based division algorithm would be of quadratic worst-case complexity. In the case of the Bernstein basis it may even be doubtful because of the cost associated with computing the generalized companion matrix [24]. Another approach for addressing the problem of the current paper would be to efficiently convert the polynomials into power basis [9,15,20], to apply some fast polynomial division algorithm in power basis and to convert the result back into Lagrange or Bernstein basis. However, this is not the focus of the current paper. The goal of the current paper is to provide an algorithm that does not require basis conversion.

The paper is organized as follows. Section 2 provides a generalized (basis-independent) framework for polynomial division. The following two sections, 3 and 4, apply the generalized framework to polynomials represented respectively in Lagrange and Bernstein basis.

2 Basis-Independent Framework for Polynomial Division

The purpose of this section is to give a general framework, which is basis-independent, for the well-known classical division algorithm for polynomials given in power basis representation. In subsequent sections this framework will be specialized in order to obtain division algorithms for polynomials in Lagrange and in Bernstein basis.

2.1 Motivating Example

We divide the polynomial f_0 by the polynomial g presented in power basis by $f_0 = 6\,x^3 + 3\,x^2 + 12\,x - 3$ and $g = 2\,x + 1$. When carrying out the division we generate a sequence f_1, f_2, f_3 where f_{i+1} is obtained from f_i by removing the leading term, that is,

$$\begin{aligned} f_1 &= f_0 - 3\,x^2\,g = 0\,x^2 + 12\,x - 3 \\ f_2 &= f_1 \qquad\quad\;\; = 12\,x - 3 \\ f_3 &= f_2 - 6\,g \quad\;\; = -9. \end{aligned}$$

In this sequence each f_{i+1} has a smaller degree than its successor f_i. Notice that f_1 and f_3 are formed by subtracting multiples of g from f_0 and respectively f_2 such that the respective leading terms $6\,x^3$ and $12\,x$ vanish. In the case of f_2 no subtraction has to be carried out because $0\,x^2$, the leading term of f_1, contains a vanishing coefficient. Moreover, the sequence of f_i's stops with f_3 because the degree of f_3 is less than the degree of g. Obviously f_3 is the remainder of f_0 divided by g.

According to the sequence of f_i's we can also form a sequence of partial quotients $q_1 = 3\,x^2$, $q_2 = 0\,x$, $q_3 = 6$. Then the quotient of f_0 divided by g is the sum $q_1 + q_2 + q_3$ of the partial quotients.

So, what are the key operations needed for carrying out the division?

1. Extracting the coefficient of the leading term of a polynomial, e.g. when forming f_1 and f_3. Subsequently, we will denote this operation with the symbol Head.
2. Extracting the part of a polynomial minus the leading term, e.g. when forming f_2. Subsequently, we will denote this operation with the symbol Tail.
3. Multiple of g matching the degree of f_i, e.g. $x^2\,g$. Subsequently, we will denote this operation with the symbol Match.
4. Partial quotient, the multiplier for the operation Match, e.g. x^2 for $x^2\,g$. Subsequently, we will denote this operation with the symbol Quot.

2.2 Definition of Polynomial Division Algorithm

We define a polynomial division algorithm in a general setting independent from the bases in which the polynomials are represented. The algorithms uses some basic operators we define first.

Definition 1. *For some field F, the symbol $F[x]_m$ denotes the F-vector space of polynomials of degree up to m in the variable x. Moreover we let* $\deg(0) = -1$ *and for $m < 0$ the symbol $F[x]_m$ denotes the (trivial) F-vector space* $\{0\}$.

Definition 2. (Basic Operators)

Head: *By* $\mathrm{Head}_m(f)$ *we denote the coefficient of x^m of the polynomial f in $F[x]_m$.*

Tail: *By* Tail_m *we denote a function that identically maps any polynomial f in $F[x]_m$ of degree at most $m-1$ into $F[x]_{m-1}$.*

Match: *If f, g respectively is a polynomial of degree m and n, with $n \leq m$, then* $\mathrm{Match}_k(f,\, g) = q\,g$ *for some polynomial $q \in F[x]_k$ of degree $m - n \leq k$.*

Quot: $\mathrm{Quot}_k(f,\, g)$ *denotes the factor q in the definition of* $\mathrm{Match}_k(f,\, g)$.

The operator Tail in the above definition is important in computations. That is, if f is represented with respect to some basis of $F[x]_m$, then $\mathrm{Tail}_m(f)$ is f represented with respect to some basis of $F[x]_{m-1}$.

In the above definition of the operator Match, the polynomial q is only specified up to degree. Later, it will be specified precisely as suitable for computations in Lagrange and Bernstein basis. The objective of its definition will be to allow to efficiently compute the product $\mathrm{Match}_{m-n}(f,\, g) = \mathrm{Quot}_{m-n}(f,\, g) \cdot g$. Moreover note that the operator Match only depends on the degree of f, not on its coefficients. We included f in the definition rather than only its degree for the sake of a clearer presentation.

The subsequent examples illustrate the definitions for polynomials represented in the power basis $\{1,\, x,\, x^2,\, x^3, \ldots\}$.

Example 3. For polynomials in power basis representation,

1. $\mathrm{Head}_m(\sum_{k=0}^{m} a_k\, x^k) = a_m$,
2. $\mathrm{Tail}_m(0 \cdot x^m + \sum_{k=0}^{m-1} a_k\, x^k) = \sum_{k=0}^{m-1} a_k\, x^k$,
3. $\mathrm{Match}_{m-n}(f,\, g) = x^{m-n}\, g$, where f, g respectively is of degree m and n,
4. $\mathrm{Quot}_{m-n}(f,\, g) = x^{m-n}$.

Polynomial Division Algorithm

Input: polynomials $f \in F[x]_m$ and $g \in F[x]_n$, where g had degree $n \leq m$.
Output: the quotient Q and the remainder R of f divided by g

Let $f_0 = f$. Generate sequences of polynomials f_i and q_i, for $i = 1, \ldots, m-n+1$, such that

$$f_{i+1} = \mathrm{Tail}_{m-i}(f_i - \frac{\mathrm{Head}_{m-i}(f_i)}{\mathrm{Head}_{m-i}(\mathrm{Match}_{m-n}(f_i,\, g))}\, \mathrm{Match}_{m-n}(f_i,\, g)),$$

$$q_{i+1} = \frac{\mathrm{Head}_{m-i}(f_i)}{\mathrm{Head}_{m-i}(\mathrm{Match}_{m-n}(f_i,\, g))}\, \mathrm{Quot}_{m-n}(f_i,\, g).$$

Then $Q = q_1 + \cdots + q_{m-n+1}$ and $R = f_{m-n+1}$.

Remark 4. *It can happen that* $\mathrm{Head}(f_i) = 0$ *(see the motivating example above). In this case the definition of* f_{i+1} *simply is* $\mathrm{Tail}_{m-i}(f_i)$ *and respectively* q_{i+1} *is zero.*

2.3 Complexity of the Division Algorithm

We investigate the worst-case complexity of the number of arithmetic operations required by the polynomial division algorithm.

For the remainder of this section we assume that the polynomials are represented appropriately in order to guarantee the expected linear complexity of addition, subtraction and multiplication by a constant. This is obviously possible if a polynomial in $F[x]_m$ is represented by the list of $(m+1)$ coefficients with respect to a fixed basis. Therefore we make the following assumption.

Assumption 5. *The polynomials* $f, g \in F[x]_m$ *are represented such that the number of arithmetic operations needed for computing* $f \pm g$ *and* $c \cdot f$*, for any constant* c*, is* $\mathrm{O}(m)$.

Next we formulate some natural assumptions on the complexities for computing the basic operators from Definition 2, which are satisfied for computations in the standard power basis. Later we will investigate whether the basic operators for the Lagrange and Bernstein bases fulfill these assumptions.

Assumption 6. *The number of arithmetic operations needed for computing*

$$\begin{aligned} \mathrm{Head}_m(f) \ \textit{is} \ \mathrm{O}(m), \qquad & \mathrm{Match}_k(f,\, g) \ \textit{is} \ \mathrm{O}(n), \\ \mathrm{Tail}_m(f) \ \textit{is} \ \mathrm{O}(m), \qquad & \mathrm{Quot}_k(f,\, g) \ \textit{is} \ \mathrm{O}(k). \end{aligned}$$

Theorem 7. *Under Assumption 6 the number of arithmetic operations needed for computing the quotient and remainder of f by g is* $\mathrm{O}(m \cdot (m - n + 1))$.

Proof: Since the division algorithm generates $m - n + 1$ polynomials f_i and q_i. It remains to check that the number of arithmetic operations is $\mathrm{O}(m)$ for each i. Notice that by the definition of the operator Tail the degree of f_i is at most $m-i$. Therefore $\frac{\mathrm{Head}(f_i)}{\mathrm{Head}(\mathrm{Match}_{m-n}(f_i,\, g))}$ is $\mathrm{O}(m)$. Thus computing f_{i+1} and q_{i+1} is $\mathrm{O}(m)$. Furthermore, computing the sum in the quotient Q is $\mathrm{O}((m-n)(m-n+1))$ which is $\mathrm{O}(m \cdot (m - n + 1))$. □

Remark 8. By [21] division over the power basis is $\mathrm{O}(n \cdot (m - n + 1))$. Notice that the left-hand factor only depends on n and not on m. This is due to being able to retrieve the leading coefficient of a polynomial represented in power basis in constant time. For Lagrange and Bernstein basis this operation is non-constant, $\mathrm{O}(m)$. Hence we get the factor m in the complexity.

Next we prove the correctness of the division algorithm.

2.4 Correctness of the Division Algorithm

The sequence of f_i's satisfies the invariant $f_i = q_{i+1}\, g + f_{i+1}$. Thus $f_0 = (\sum_{i=1}^{m-n+1} q_i) g + f_{m-n+1}$. Since by the definition of the operator Tail the degrees of the polynomials of the sequence f_i are decreasing, the degree of f_{m-n+1} is less than the degree of g. Therefore $Q = \sum_{i=1}^{m-n+1} q_i$ is the quotient of the division of f_0 by g and $R = f_{m-n+1}$ is the remainder.

3 Division in Lagrange Basis

This section consists of three parts. The first part defines the basic operators required for polynomial division (Definition 2). The second part derives the complexity of the division algorithm. The third part proves the correctness of the definitions.

3.1 Definition of Basic Operators

We start with the definition of the Lagrange basis.

Definition 9 (Lagrange basis). *Let* $\lambda_{j,m} = \frac{\pi_{j,m}}{\overline{\pi}_{j,m}}$ *where* $\pi_{j,m} = \prod_{\substack{i=0 \\ i \neq j}}^{m} (x - x_i)$ *and* $\overline{\pi}_{j,m} = \prod_{\substack{i=0 \\ i \neq j}}^{m} (x_j - x_i)$. *Then* $\lambda_{m,m}, \ldots, \lambda_{0,m}$ *are called the Lagrange basis of degree* m.

Definition 10. *We define the head and tail operators as*

$$\mathrm{Head}_m(\sum_{j=0}^{m} a_j \ \lambda_{j,m}) = \sum_{j=0}^{m} \frac{a_j}{\overline{\pi}_{j,m}},$$
$$\mathrm{Tail}_m(\sum_{j=0}^{m} a_j \ \lambda_{j,m}) = \sum_{j=0}^{m-1} a_j \, \lambda_{j,m-1}\,.$$

Definition 11. *We define the match and quot operators as*

$$\mathrm{Match}_k(f, \sum_{j=0}^{n} b_j \, \lambda_{j,n})) = \sum_{j=0}^{n} b_j \, \overline{\pi}_{j,m} \, \overline{\pi}_{j,n}^{-1} \ \lambda_{j,m},$$
$$\mathrm{Quot}_k(f, \sum_{j=0}^{n} b_j \, \lambda_{j,n})) = \sum_{j=0}^{u} \overline{\pi}_{j,m} \, \overline{\pi}_{j,n}^{-1} \ \lambda_{j,k},$$

where $f \in F[x]_m$ *and* $u = \min(k, n)$, *if* $m > n$, *and* $u = k$, *if* $m = n$.

The essential properties of the match and quot operators are

$$\mathrm{Match}_k(f,\, g) = \mathrm{Quot}_k(f,\, g) \cdot g,$$
$$\mathrm{Quot}_k(f,\, g) = \prod_{i=n+1}^{m} (x - x_i)$$

which will be shown in the section on the correctness of the operators below. Notice that $\mathrm{Quot}_k(f,\, g)$ has been chosen such that the product $\mathrm{Quot}_k(f,\, g) \cdot g = \mathrm{Match}_k(f,\, g)$ can be easily computed in the basis for $F[x]_m$.

Next we give an example for polynomial division when using these operators. We use the same polynomials as in Section 2.1.

Example 12. With $x_0 = 0$, $x_1 = 1$, $x_2 = 2$, $x_3 = 3$, let the polynomials $f_0 = 222 \ \lambda_{3,3} + 81 \ \lambda_{2,3} + 18 \ \lambda_{1,3} - 3 \ \lambda_{0,3}$ and $g = 3 \ \lambda_{1,1} + 1 \ \lambda_{0,1}$. Then

$$\begin{aligned}
f_1 &= f_0 - 3\,(-2\,x\,(x-2) + 3\,(x-1)(x-2))\,g &&= f_0 - 3\,(2\ \lambda_{1,2} + 6\ \lambda_{0,2})\,g \\
&= 81\ \lambda_{2,2} - 21\ \lambda_{0,2}, \\
f_2 &= f_1 - 15\,(-(x-1)(x-2) + x(x-2))\,g &&= f_1 - 15\,(-1\ \lambda_{1,2} - 2\ \lambda_{0,2})\,g \\
&= 45\ \lambda_{1,1} + 9\ \lambda_{0,1} \\
f_3 &= f_2 - 18 \cdot 1 \cdot g &&= f_2 - 18\,(\lambda_{2,2} + \lambda_{1,2} + \lambda_{0,2})g \\
&= -9\ \lambda_{0,0}
\end{aligned}$$

3.2 Complexity

We investigate the number of arithmetic operations required to carry out the polynomial division algorithm. Before we state the main result, Theorem 15, we give some auxiliary lemmas.

Lemma 13. *The number of arithmetic operations required to compute*

$$(\overline{\pi}_{0,n}, \overline{\pi}_{0,n+1}, \ldots, \overline{\pi}_{0,m}), \ldots, (\overline{\pi}_{m,n}, \overline{\pi}_{m,n+1}, \ldots, \overline{\pi}_{m,m}) \quad \textit{is} \quad \mathrm{O}(m^2). \tag{1}$$

Proof: Since $\overline{\pi}_{j,i+1} = (x_j - x_{i+1}) \cdot \overline{\pi}_{j,i}$, for $j \leq i$ or $j > i+1$ and $\overline{\pi}_{j,i+1} = \overline{\pi}_{j,i}$, for $j = i+1$, computing $\overline{\pi}_{j,n}, \overline{\pi}_{j,n+1}, \ldots, \overline{\pi}_{j,m}$ is $\mathrm{O}(m)$ for each $j = 0, \ldots, m$. □

Lemma 14. *Given all required values of $\overline{\pi}_{j,i}$, the basic operators from Definitions 11 and 10 satisfy Assumption 6.*

Proof: The head operator requires computing the sum of $\frac{a_j}{\overline{\pi}_{j,m}}$ for $j = 0, \ldots, m$ which is $\mathrm{O}(m)$. Furthermore, the tail operator is $\mathrm{O}(1)$.

The match and quot operators require computing $\overline{\pi}_{j,m}^{-1} \overline{\pi}_{j,n}$ either for indices $j = 0, \ldots, \min(k, n)$ for $\mathrm{Quot}_k(f, g)$ and for $j = 0, \ldots, n$ for $\mathrm{Match}_k(f, g)$, where $g \in F[x]_n$, which is $\mathrm{O}(k)$ and respectively $\mathrm{O}(n)$. Moreover $\mathrm{Match}_k(f, g)$ requires computing $(\overline{\pi}_{j,m}^{-1} \overline{\pi}_{j,n}) \cdot b_j$ for $j = 0, \ldots n$ which is $\mathrm{O}(n)$. Thus $\mathrm{Match}_k(f, g)$ and $\mathrm{Quot}_k(f, g)$ respectively is $\mathrm{O}(n)$ and $\mathrm{O}(k)$. □

Now we are ready to show the complexity of division.

Theorem 15. *The number of arithmetic operations required for division in the Lagrange basis is $\mathrm{O}(m^2)$.*

Proof: We observe that $\mathrm{Match}_k(f, \sum_{j=0}^{n} a_j \lambda_{j,n})$, $\mathrm{Quot}_k(f, \sum_{j=0}^{n} b_j \lambda_{j,n})$ and $\mathrm{Head}_m(\sum_{j=0}^{m} a_j \lambda_{j,m})$ require the constants $\overline{\pi}_{j,n}$ and $\overline{\pi}_{j,m}$, for $j = 0, \ldots, m$. Thus, considering $n \leq m$, by Lemma 13, and Theorem 7, the division algorithm is $\mathrm{O}(m^2 + m\,(m - n + 1))$ which is $\mathrm{O}(m^2)$. □

3.3 Correctness

We prove the correctness of the definitions of the basic operators in the previous section. The correctness follows from two theorems, 16 and 19, which respectively verify the defining properties of the head/tail and match/quot operators.

Theorem 16. *If $\sum_{j=0}^{m} \frac{a_j}{\overline{\pi}_{j,m}} = 0$ then $\sum_{j=0}^{m} a_j \lambda_{j,m} = \sum_{j=0}^{m-1} a_j \lambda_{j,m-1}$. Furthermore, $\sum_{j=0}^{m} \frac{a_j}{\overline{\pi}_{j,m}}$ is the coefficient of x^m in the polynomial $\sum_{j=0}^{m} a_j \lambda_{j,m}$.*

The right-hand side of the above equality is used to define the tail operator and that the coefficient of x^m is returned by the head operator.

Proof: Notice that the leading coefficient of $\lambda_{j,m}$ is $\frac{1}{\overline{\pi}_{j,m}}$. Therefore the coefficient of x^m in $f = \sum_{j=0}^{m} a_j \lambda_{j,m}$ is $\sum_{j=0}^{m} \frac{a_j}{\overline{\pi}_{j,m}}$. If $\sum_{j=0}^{m} \frac{a_j}{\overline{\pi}_{j,m}} = 0$ then the degree of f is at most $m - 1$. In this case m interpolation points, say,

$(x_0, a_0), \ldots, (x_{m-1}, a_{m-1})$ are sufficient to determine f. Therefore, we get $f = \sum_{j=0}^{m-1} a_j \lambda_{j,m-1}$. □

Before proving the next theorem we give some auxiliary lemmas.

The following observation allows us to relate $\lambda_{j,m}$ to $\lambda_{j,n}$.

Lemma 17. $\lambda_{j,m} = \prod_{i=n+1}^{m} (x - x_i)\, \overline{\pi}_{j,m}^{-1}\, \overline{\pi}_{j,n}\, \lambda_{j,n}$ *if* $m \geq n$ *and* $j \leq n$.

Proof: For $j \leq n$,

$$\lambda_{j,m} = \overline{\pi}_{j,m}^{-1} \prod_{\substack{i=1 \\ i \neq j}}^{n} (x - x_i) \prod_{i=n+1}^{m} (x - x_i) = \overline{\pi}_{j,m}^{-1}\, \overline{\pi}_{j,n} \prod_{i=n+1}^{m} (x - x_i)\, \lambda_{j,n}\,.$$ □

The next lemma allows us to write the factor from Lemma 17 that relates $\lambda_{j,m}$ to $\lambda_{j,n}$ in terms of the Lagrange basis.

Lemma 18

$$\prod_{i=n+1}^{m} (x - x_i) = \sum_{j=0}^{u} \overline{\pi}_{j,m}\, \overline{\pi}_{j,n}^{-1}\, \lambda_{j,k},$$

where $u = \min k, n$, *if* $0 < m - n \leq k$, *and* $u = k$, *if* $m = n$.

Proof: Interpolating $\prod_{i=n+1}^{m} (x - x_i)$ for $x = x_0, \ldots, x_k$, yields

$$\prod_{i=n+1}^{m} (x - x_i) = \sum_{j=0}^{k} \Big(\prod_{i=n+1}^{m} (x_j - x_i) \Big)\, \lambda_{j,k}\,.$$

In the trivial case of $m = n$ the products for i ranging from $n+1$ to m are 1, and thus the lemma holds for this case. So, let us assume the other case $0 < m - n \leq k$. Observe that $\prod_{i=n+1}^{m} (x_j - x_i) = 0$ for $j \geq n+1$. Therefore,

$$\prod_{i=n+1}^{m} (x - x_i) = \sum_{j=0}^{\min(k,n)} \Big(\prod_{i=n+1}^{m} (x_j - x_i) \Big)\, \lambda_{j,k}\,.$$

Furthermore, for $j \leq n$,

$$\prod_{i=n+1}^{m} (x_j - x_i) = \prod_{i=n+1}^{m} (x_j - x_i) \cdot \frac{\prod_{\substack{i=1 \\ i \neq j}}^{n} (x_j - x_i)}{\prod_{\substack{i=1 \\ i \neq j}}^{n} (x_j - x_i)} = \frac{\prod_{\substack{i=1 \\ i \neq j}}^{m} (x_j - x_i)}{\prod_{\substack{i=1 \\ i \neq j}}^{n} (x_j - x_i)} = \frac{\overline{\pi}_{j,m}}{\overline{\pi}_{j,n}}.$$

□

The following theorem determines a multiple of a polynomial represented in Lagrange basis. This multiple is denoted by the operator Match and the factor by the operator Quot in the division algorithm.

Theorem 19

$$\sum_{j=0}^{n} b_j \, \overline{\pi}_{j,m} \, \overline{\pi}_{j,n}^{-1} \, \lambda_{j,m} \;=\; (\sum_{j=0}^{u} \overline{\pi}_{j,m} \, \overline{\pi}_{j,n}^{-1} \, \lambda_{j,k}) \, (\sum_{j=0}^{n} b_j \; \lambda_{j,n}),$$

where $u = \min(k, n)$*, if* $0 < m - n \leq k \leq m$*, and* $u = k$*, if* $m = n$*.*

Proof: The case $m = n$ is obvious because the left factor on the right-hand side of the equality is 1. So, let us consider the other case. By Lemmas 17 and 18,

$$\begin{aligned}(\sum_{j=0}^{\min(k,n)} \overline{\pi}_{j,m} \, \overline{\pi}_{j,n}^{-1} \, \lambda_{j,k}) \, (\sum_{j=0}^{n} b_j \; \lambda_{j,n}) \;&=\; \prod_{i=n+1}^{m} (x - x_i) \, (\sum_{j=0}^{n} b_j \; \lambda_{j,n}) \\ &=\; (\sum_{j=0}^{n} b_j \prod_{i=n+1}^{m} (x - x_i) \; \lambda_{j,n}) \;=\; (\sum_{j=0}^{n} b_j \, \overline{\pi}_{j,m} \, \overline{\pi}_{j,n}^{-1} \, \lambda_{j,m}).\end{aligned}$$

□

4 Division in Bernstein Basis

This section consists of three parts. The first part defines the basic operators required for polynomial division (Definition 2). The second part derives the complexity of the division algorithm. The third part proves the correctness of the definitions of the basic operators.

4.1 Definition of Basic Operators

We start with the definition of the Bernstein basis.

Definition 20 (Bernstein basis). *Let* $\beta_{j,m} = \binom{m}{j} x^j \, (1 - x)^{m-j}$*. Then the polynomials* $\beta_{m,m}, \ldots, \beta_{0,m}$ *are called the Bernstein basis of degree* m*.*

Definition 21. *We define the head and tail operators as*

$$\begin{aligned}\mathrm{Head}_m(\sum_{j=0}^{m} a_j \; \beta_{j,m}) &= \sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j, \\ \mathrm{Tail}_m(\sum_{j=0}^{m} a_j \; \beta_{j,m}) &= \sum_{j=0}^{m-1} (-1)^j \left(\sum_{i=0}^{j} (-1)^i \binom{m}{i} a_i \right) \binom{m-1}{j}^{-1} \beta_{j,m-1}\,.\end{aligned}$$

Definition 22. *We define the match and quot operators as*

$$\begin{aligned}\mathrm{Match}_k(f, \sum_{j=0}^{n} b_j \, \beta_{j,n}) &= \sum_{j=0}^{n} b_j \binom{n}{j} \binom{m}{j}^{-1} \beta_{j,m}, \\ \mathrm{Quot}_k(f, \sum_{j=0}^{n} b_j \, \beta_{j,n}) &= \sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} \binom{k}{j}^{-1} \beta_{j,k}\,.\end{aligned}$$

The essential properties of the match and quot operators are

$$\begin{aligned}\mathrm{Match}_k(f,\, g) &= \mathrm{Quot}_k(f,\, g)\cdot g,\\ \mathrm{Quot}_k(f,\, g) &= (1-x)^{m-n}\end{aligned}$$

which will be shown in the section on the correctness of the operators below. Notice that $\mathrm{Quot}_k(f,\, g)$ has been chosen such that the product $\mathrm{Quot}_k(f,\, g)\cdot g = \mathrm{Match}_k(f,\, g)$ can be easily computed in the basis for $F[x]_m$.

Next we give an example for polynomial division when using these operators. We use the same polynomials as in Section 2.1.

Example 23. Let

$$\begin{aligned}f &= 18\ \beta_{3,3} + 6\ \beta_{2,3} + 1\ \beta_{1,3} - 3\ \beta_{0,3}\\ g &= 3\ \beta_{1,1} + 1\ \beta_{0,1}\end{aligned}$$

Then

$$\begin{aligned}f_1 &= f_0 - 3\,(1-x)^2\, g &&= f_0 - 3\ \beta_{0,2}\ g &&= 18\ \beta_{2,2} - 6\ \beta_{0,2},\\ f_2 &= f_1 - (-6)\,(1-x)\, g &&= f_1 - (-6)\,(\tfrac{1}{2}\,\beta_{1,2} + \beta_{0,2})\, g &&= 18\ \beta_{1,1},\\ f_3 &= f_2 - 9\cdot 1\cdot g &&= f_2 - 9\,(\beta_{2,2} + \beta_{1,2} + \beta_{0,2})\, g &&= -9\ \beta_{0,0}\,.\end{aligned}$$

4.2 Complexity

We investigate the number of arithmetic operations required to carry out the polynomial division algorithm. Before we state the main result, Theorem 26, we give some auxiliary lemmas.

Lemma 24. *The number of arithmetic operations required to compute all*

$$\binom{i}{j}, \quad \textit{for } j = 0,\ldots,i \textit{ and for } i = \min(m-n, n-1),\ldots,m \tag{2}$$

is $\mathrm{O}(m^2 - \min(m-n, n-1)^2)$.

Proof: Since $\binom{i}{j+1} = \frac{i!}{(i-j-1)!\,(j+1)!} = \frac{i-j}{j+1}\binom{i}{j}$, computing $\binom{i}{0},\ldots,\binom{i}{i}$ is $\mathrm{O}(i)$. Thus computing $\binom{i}{j}$, for all $j = 0,\ldots,i$ and for all $i = \min(m-n, n-1),\ldots,m$ is of order $\sum_{i=\min(m-n,n-1)}^{m} i$ which is

$$\mathrm{O}((m + \min(m-n, n-1))(m - \min(m-n, n-1) + 1)),$$

that is, $\mathrm{O}(m^2 - \min(m-n, n-1)^2)$. □

Lemma 25. *Given all required values of* $\binom{i}{j}$*, the basic operators from Definitions 22 and 21 satisfy Assumption 6.*

Proof: The head operator requires computing the sum of $(-1)^{m-j}\binom{m}{j} a_j$ for $j = 0,\ldots,m$ which is $\mathrm{O}(m)$. Moreover the tail operator requires computing the

sum of $(-1)^j \gamma_j \binom{m-1}{j}^{-1}$ for $j = 0, \ldots, m-1$, where $\gamma_j = \sum_{i=0}^{j} (-1)^i \binom{m}{i} a_i$. Since $\gamma_{j+1} = \gamma_j + (-1)^{j+1} \binom{m}{j+1} a_{j+1}$, computing the head operator is $\mathrm{O}(m)$.

Given the constants $\binom{i}{j}$ the operator $\mathrm{Quot}_k(f, \sum_{j=0}^{n} b_j \beta_{j,n})$ is of complexity $\mathrm{O}(k-(m-n))$, which is $\mathrm{O}(k)$, and the operator $\mathrm{Match}_k(f, \sum_{j=0}^{n} b_j \beta_{j,n})$ requires computing $a_j \binom{n}{j} \binom{m}{j}^{-1}$ for $j = 0, \ldots, n$ which is $\mathrm{O}(n)$. □

Now we are ready to show the complexity of division.

Theorem 26. *The number of arithmetic operations required for division in the Bernstein basis is*

$$\mathrm{O}(m^2 - \min(m-n, n-1)^2).$$

Proof: We observe that $\mathrm{Quot}_k(f, \sum_{j=0}^{n} b_j \beta_{j,n})$ requires $\binom{k-(m-n)}{j}$, for all $j = 0, \ldots, k-(m-n)$ and that $\mathrm{Match}_k(f, \sum_{j=0}^{n} b_j \beta_{j,n})$ requires the constants $\binom{n}{j}$ and $\binom{m}{j}$, for $j = 0, \ldots, n$. Furthermore we observe that $\mathrm{Head}_m(\sum_{j=0}^{m} b_j \beta_{j,m})$ and $\mathrm{Tail}_m(\sum_{j=0}^{m} b_j \beta_{j,m})$ require the constants $\binom{m}{j}$, for $j = 0, \ldots, m$ and $\binom{m-1}{j}$, for $j = 0, \ldots, m-1$. Thus overall the division algorithm requires the constants $\binom{l}{j}$, for $j = 0, \ldots, l$ and for $l = \min(m-n, n-1), \ldots, m$. Therefore Lemma 24 provides for all required constants.

Considering $n \leq m$, by Lemmas 24, 25 and Theorem 7 the division algorithm is of order $m^2 - \min(m-n, n-1)^2 + (m+n)(m-n+1)$. This is equivalent to $\mathrm{O}(m^2 - \min(m-n, n-1)^2 + m^2 - n^2)$ which is $\mathrm{O}(m^2 - \min(m-n, n-1)^2)$. □

4.3 Correctness

We prove the correctness of the definitions of the basic operators in the previous section. The correctness follows from two theorems, 32 and 29, which respectively verify the defining properties of the head/tail and match/quot operators. Before proving these theorems we give some auxiliary lemmas.

Lemma 27

$$x^{j-1}(1-x)^{m-(j-1)} = x^{j-1}(1-x)^{(m-1)-(j-1)} - x^j (1-x)^{m-j}.$$

Proof: $x^{j-1}(1-x)^{m-(j-1)} = x^{j-1}(1-x)^{(m-1)-(j-1)} \cdot 1 + x^{j-1}(1-x)^{m-j}(-x)$. □

Lemma 28

$$x^j (1-x)^{m-j} = (-1)^{m-j} x^m + \sum_{i=j}^{m-1} (-1)^{i-j} x^i (1-x)^{(m-1)-i}. \tag{3}$$

Proof: Proof by induction on $(m-j)$. For $m-j = 0$, that is, $j = m$, we have

$$x^m (1-x)^0 = (-1)^0 x^m + 0.$$

Next we assume (3) and show (3) after replacing $m-j$ with $m-j+1$, that is, after replacing j with $j-1$. Thus we show

$$x^{j-1}\,(1-x)^{m-(j-1)} \;=\; (-1)^{m-(j-1)}\,x^m \;+\; \sum_{i=j-1}^{m-1} (-1)^{i-(j-1)}\,x^i\,(1-x)^{(m-1)-i}.$$

By Lemma 27 and by (3), we have

$$x^{j-1}\,(1-x)^{m-(j-1)} \;=\; x^{j-1}\,(1-x)^{(m-1)-(j-1)} \;-\; x^j\,(1-x)^{m-j} \;=$$

$$x^{j-1}\,(1-x)^{(m-1)-(j-1)} \;-\; ((-1)^{m-j}\,x^m \;+\; \sum_{i=j}^{m-1} (-1)^{i-j}\,x^i\,(1-x)^{(m-1)-i}).\ \square$$

Theorem 29

$$\sum_{j=0}^{m} a_j\ \beta_{j,m} \;=\; \left(\sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j\right) x^m$$
$$+\ \sum_{j=0}^{m-1} (-1)^j \left(\sum_{i=0}^{j} (-1)^i \binom{m}{i} a_i\right) \binom{m-1}{j}^{-1} \beta_{j,m-1}\,.$$

Furthermore, $\sum_{j=0}^{m} (-1)^j \binom{m}{j} a_j$ *is the coefficient of* x^m *in* $\sum_{j=0}^{m} a_j\ \beta_{j,m}$.

Proof: By Lemma 28,

$$\sum_{j=0}^{m} a_j\ \beta_{j,m} = \sum_{j=0}^{m} a_j \binom{m}{j} \left((-1)^{m-j}\,x^m \;+\; \sum_{i=j}^{m-1} (-1)^{i-j}\,x^i\,(1-x)^{(m-1)-i}\right)$$

$$= \sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j\,x^m \;+\; \sum_{j=0}^{m} a_j \binom{m}{j} \left(\sum_{i=j}^{m-1} (-1)^{i-j}\,x^i\,(1-x)^{(m-1)-i}\right)$$

$$= \sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j\,x^m \;+\; \sum_{j=0}^{m} \sum_{i=j}^{m-1} (-1)^{i-j} \binom{m}{j} a_j\,x^i\,(1-x)^{(m-1)-i}$$

$$= \sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j\,x^m \;+\; \sum_{j=0}^{m-1} \sum_{i=j}^{m-1} (-1)^{i-j} \binom{m}{j} a_j\,x^i\,(1-x)^{(m-1)-i}.$$

Next we change the order of summation in the last double sum. The summation indices of the double sum are the solutions of the set of inequalities

$$0 \le j \le m-1,\ \ 0 \le i \le m-1,\ \ j \le i \le m-1.$$

This set of inequalities is equivalent to

$$0 \le j \le m-1,\ \ 0 \le i \le m-1,\ \ 0 \le j \le i.$$

Therefore $\sum_{j=0}^{m} a_j\ \beta_{j,m}$ equals

$$\sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j\, x^m \ + \sum_{i=0}^{m-1} \sum_{j=0}^{i} (-1)^{i-j} \binom{m}{j} a_j\, x^i\, (1-x)^{(m-1)-i}$$

$$= \sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j\, x^m + \sum_{i=0}^{m-1} (-1)^i \left(\sum_{j=0}^{i} (-1)^j \binom{m}{j} a_j \right) x^i\, (1-x)^{(m-1)-i}$$

$$= \sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j\, x^m + \sum_{i=0}^{m-1} (-1)^i \left(\sum_{j=0}^{i} (-1)^j \binom{m}{j} a_j \right) \binom{m-1}{i}^{-1} \beta_{i,m-1}\,.$$

Furthermore, we observe that $\sum_{j=0}^{m} (-1)^{m-j} \binom{m}{j} a_j$ is the coefficient of x^m in the polynomial $\sum_{j=0}^{m} a_j\ \beta_{j,m}$ because $\beta_{i,m-1}$ is of degree $m-1$. □

Next we study the match and quot operators. We start with some auxiliary lemmas.

Lemma 30. $\beta_{j,m} \ = \ \binom{m}{j} \binom{n}{j}^{-1} (1-x)^{m-n}\ \beta_{j,n}$ *if* $m \geq n$.

Proof: $(1-x)^{m-j} \ = \ (1-x)^{m-n}\, (1-x)^{n-j}$. □

Lemma 31. *For* $k \geq m-n$

$$(1-x)^{m-n} \ = \ \sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} x^j\, (1-x)^{k-j}.$$

Proof:

$$\begin{aligned}(1-x)^{m-n} &= (x \ + \ (1-x))^{k-(m-n)}\, (1-x)^{m-n} \\ &= \left(\sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} x^j\, (1-x)^{k-(m-n)-j} \right) (1-x)^{m-n} \\ &= \sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} x^j\, (1-x)^{k-j}.\end{aligned}$$

□

The next theorem determines a multiple of a polynomial represented in Bernstein basis. This multiple is denoted by the operator Match and the factor by the operator Quot in the division algorithm.

Theorem 32. *For* $k \geq m-n$

$$\sum_{j=0}^{n} a_j \binom{n}{j} \binom{m}{j}^{-1} \beta_{j,m} \ = \ \Big(\sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} \binom{k}{j}^{-1} \beta_{j,k} \Big) \Big(\sum_{j=0}^{n} a_j\, \beta_{j,n} \Big).$$

Proof: By Lemmas 30 and 31,

$$\sum_{j=0}^{n} a_j \binom{n}{j} \binom{m}{j}^{-1} \beta_{j,m} = \sum_{j=0}^{n} a_j (1-x)^{m-n} \beta_{j,n}$$
$$= \Big(\sum_{j=0}^{k-(m-n)} \binom{k-(m-n)}{j} x^j (1-x)^{k-j} \Big) \Big(\sum_{j=0}^{n} a_j \beta_{j,n} \Big). \square$$

5 Conclusion and Future Directions

The current work provided polynomial division algorithms over the Lagrange and Bernstein basis of quadratic worst-case complexity. Future work may include studying if the computational complexities of the provided algorithms can be improved further and extending the algorithms to multi-variate polynomial division.

For practical applications it will be of interest to study numerical properties of these algorithms because in practice computations are often performed with floating point numbers. The case of Bernstein basis is particularly interesting. That is, polynomials presented in Bernstein basis have been shown to be well suited for stable numerical computations [13]. However, [18] has shown that the classical polynomial division algorithm *in power basis* is numerically quite unstable in certain cases.

References

1. Amiraslani, A.: Dividing polynomials when you only know their values. In: Gonzalez-Vega, L., Recio, T. (eds.) Proceedings of Encuentros de Álgebra Computacional y Aplicaciones (EACA) 2004, pp. 5–10 (2004), `http://www.orcca.on.ca/TechReports/2004/TR-04-01.html`
2. Amiraslani, A.: New Algorithms for Matrices, Polynomials and Matrix Polynomials. PhD thesis, University of Western Ontario, London, Ontario, Canada (2006)
3. Aruliah, D.A., Corless, R.M., Gonzalez-Vega, L., Shakoori, A.: Geometric applications of the bezout matrix in the lagrange basis. In: SNC 2007: Proceedings of the 2007 international workshop on Symbolic-numeric computation, pp. 55–64. ACM, New York (2007)
4. Aruliah, D.A., Corless, R.M., Shakoori, A., Gonzalez-Vega, L., Rua, I.F.: Computing the topology of a real algebraic plane curve whose equation is not directly available. In: SNC 2007: Proceedings of the 2007 international workshop on Symbolic-numeric computation, pp. 46–54. ACM Press, New York (2007)
5. Barnett, S.: Division of generalized polynomials using the comrade matrix. Linear Algebra Appl. 60, 159–175 (1984)
6. Barnett, S.: Euclidean remainders for generalized polynomials. Linear Algebra Appl. 99, 111–122 (1988)
7. Barnett, S.: Polynomials and linear control systems. Monographs and Textbooks in Pure and Applied Mathematics, vol. 77. Marcel Dekker Inc., New York (1983)

8. Bini, D.A., Gemignani, L., Winkler, J.R.: Structured matrix methods for CAGD: an application to computing the resultant of polynomials in the Bernstein basis. Numer. Linear Algebra Appl. 12(8), 685–698 (2005)
9. Bostan, A., Schost, É.: Polynomial evaluation and interpolation on special sets of points. J. Complexity 21(4), 420–446 (2005)
10. Cheng, H., Labahn, G.: On computing polynomial GCDs in alternate bases. In: ISSAC 2006, pp. 47–54. ACM, New York (2006)
11. Corless, R.: Generalized companion matrices in the lagrange basis. In: Gonzalez-Vega, L., Recio, T. (eds.) Proceedings of Encuentros de Álgebra Computacional y Aplicaciones (EACA 2004), pp. 317–322 (2004), http://www.apmaths.uwo.ca/~rcorless/frames/PAPERS/PABV/EACA2004Corless.pdf
12. Diaz-Toca, G.M., Gonzalez-Vega, L.: Barnett's theorems about the greatest common divisor of several univariate polynomials through Bezout-like matrices. J. Symbolic Comput. 34(1), 59–81 (2002)
13. Farouki, R.T., Goodman, T.N.T.: On the optimal stability of the Bernstein basis. Math. Comp. 65(216), 1553–1566 (1996)
14. Gemignani, L.: Manipulating polynomials in generalized form. Technical Report TR-96-14, Università di Pisa, Departmento di Informatica, Corso Italia 40, 56125 Pisa, Italy (December 1996)
15. Goldman, R.: Pyramid Algorithms: A Dynamic Programming Approach to Curves and Surfaces for Geometric Modeling, 1st edn. The Morgan Kaufmann Series in Computer Graphics. Morgan Kaufmann, San Francisco (2002)
16. Mani, V., Hartwig, R.E.: Generalized polynomial bases and the Bezoutian. Linear Algebra Appl. 251, 293–320 (1997)
17. Maroulas, J., Barnett, S.: Greatest common divisor of generalized polynomial and polynomial matrices. Linear Algebra Appl. 22, 195–210 (1978)
18. Stetter, H.J.: Numerical polynomial algebra. Society for Industrial and Applied Mathematics. SIAM, Philadelphia (2004)
19. Tsai, Y.-F., Farouki, R.T.: Algorithm 812: BPOLY: An object-oriented library of numerical algorithms for polynomials in Bernstein form. ACM Transactions on Mathematical Software 27(2), 267–296 (2001)
20. Vries-Baayens, A.: CAD product data exchange: conversions for curves and surfaces. PhD thesis, Delft University (1991)
21. Winkler, F.: Polynomial algorithms in computer algebra. In: Texts and monographs in symbolic computation. Springer, Heidelberg (1996)
22. Winkler, J.R.: A resultant matrix for scaled Bernstein polynomials. Linear Algebra Appl. 319(1-3), 179–191 (2000)
23. Winkler, J.R.: Properties of the companion matrix resultant for Bernstein polynomials. In: Uncertainty in geometric computations. Kluwer Internat. Ser. Engrg. Comput. Sci., vol. 704, pp. 185–198. Kluwer Acad. Publ., Boston (2002)
24. Winkler, J.R.: A companion matrix resultant for Bernstein polynomials. Linear Algebra Appl. 362, 153–175 (2003)

Computing the Greatest Common Divisor of Polynomials Using the Comrade Matrix

Nor'aini Aris and Shamsatun Nahar Ahmad

Department of Mathematics, Faculty of Science
Universiti Teknologi Malaysia, Malaysia
noraini@mel.fs.utm.my
http://www.matematik.utm.my/

Abstract. The comrade matrix of a polynomial is an analogue of the companion matrix when the matrix is expressed in terms of a general basis such that the basis is a set of orthogonal polynomials satisfying the three-term recurrence relation. We present the algorithms for computing the comrade matrix, and the coefficient matrix of the corresponding linear systems derived from the recurrence relation. The computing times of these algorithms are analyzed. The computing time bounds, which dominate these times, are obtained as functions of the degree and length of the integers that represent the rational number coefficients of the input polynomials. The ultimate aim is to apply these computing time bounds in the analysis of the performance of the generalized polynomial greatest common divisor algorithms.

Keywords: comrade matrix, orthogonal polynomials, three-term recurrence relation, greatest common divisor of generalized polynomials.

1 Introduction

Problems involving polynomials such as polynomial greatest common divisor (GCD) computation and polynomial factorization, provide the basis for many applications in symbolic computations and are important in many areas of contemporary applied mathematics. In the theory of linear multivariable control systems, polynomial GCD determination arises in the computation of Smith form of a polynomial matrix and problems associated with the minimal realization of a transfer function matrix.

We consider the problem of computing the GCD of two polynomials

$$a(x) = a_0 p_0(x) + a_1 p_1(x) + ... + a_n p_n(x) \ , \tag{1}$$

$$b(x) = b_0 p_0(x) + b_1 p_1(x) + ... + b_m p_m(x) \ . \tag{2}$$

with coefficients over any field, represented in a basis of orthogonal polynomials $\{p_i(x)\}_{i=0}^{n}$ defined by the three-term recurrence relation:

$$\begin{aligned} p_0(x) &= 1, \\ p_1(x) &= \alpha_0 x + \beta_0, \\ p_{i+1}(x) &= (\alpha_i x + \beta_i) p_i(x) - \gamma_i p_{i-1}(x) \ . \end{aligned} \tag{3}$$

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 87–96, 2008.
© Springer-Verlag Berlin Heidelberg 2008

for $i = 1, 2, .., n-1$ with $\alpha_i > 0$, $\gamma_i, \beta_i \geq 0$. We can assume without loss of generality that $m < n$. The associated comrade matrix (see [4]) is given by

$$\begin{pmatrix} \frac{-\beta_0}{\alpha_0} & \frac{1}{\alpha_0} & \dots & 0 & \dots & \dots & \dots & 0 \\ \frac{\gamma_1}{\alpha_1} & \frac{-\beta_1}{\alpha_1} & \frac{1}{\alpha_1} & 0 & \dots & \dots & \dots & 0 \\ 0 & \frac{\gamma_2}{\alpha_2} & \frac{-\beta_2}{\alpha_2} & \frac{1}{\alpha_2} & \dots & \dots & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \dots & \dots & \dots & \vdots \\ 0 & 0 & 0 & 0 & \dots & \dots & \dots & \frac{1}{\alpha_{n-2}} \\ \frac{-a_0}{a_n\alpha_{n-1}} & \dots & \dots & \dots & \dots & \frac{-a_{n-3}}{a_n\alpha_{n-1}} & \frac{-a_{n-2}+a_n\gamma_{n-1}}{a_n\alpha_{n-1}} & \frac{-a_{n-1}-a_n\beta_{n-1}}{a_n\alpha_{n-1}} \end{pmatrix}, \quad (4)$$

has $a(x)$ of the form (1) as its characteristic polynomial.

There has been considerable work on manipulation of polynomials represented in the generalized form. Barnett made used of the congenial matrix generalizations in order to give procedures for computing GCDs [3] and later division [6]. Our aim is to implement the theories and analytical results in [3], [5] by computing the GCDs in exact arithmetic environments. Rahman [2] investigates into the GCD of polynomials in the generalized form in floating point environment, applying the numerical results to solving polynomials exhibiting repeated roots. Although the method developed shows potential, as expected, the build up of rounding errors is so enormous as to render effective results for high degree polynomials. However, in the exact arithmetic environment, the modular approach can be applied to handle the problem of coefficient growth when manipulating with high degree monic polynomials having rational number cofficients involving multiprecision integers (as of polynomials in Legendre bases), which form the entries of the comrade matrix. The promising success of modular techniques in solving exact solutions for dense systems of linear equations indicated in[12] suggests an investigation on its application to Barnett's procedure.

The basic technique of converting to and from the standard form polynomials involves solving a system of linear equations. Barnett's procedure also involves solving systems of linear equations simultaneously, which at the same time leads to finding the actual solutions. Apart from that, by adhering to polynomials in the original form, the properties that have been derived for polynomials in these nonstandard bases can be applied and investigated. For example, Labahn and Cheng [10] generalized the well known subresultant GCD and the modular GCD algorithms of the standard form polynomials to polynomials in terms of Newton bases or orthogonal bases, which are also known as the alternate bases. In this work we apply the comrade matrix, which is a generalization of the companion matrix applied to computing the GCD of polynomials in terms of the orthogonal bases.

2 General Procedure and Related Work

Let $a(x)$ and $b(x)$ be two polynomials satisfying the relation [3], with $\deg(a(x)) \geq \deg(b(x))$. The method of finding $\gcd(a(x), b(x))$ using the comrade matrix can be outlined by the following procedure:

1. Construct the comrade matrix associated with $a(x)$.
2. Construct the coefficient matrix of the corresponding systems of equations obtained from a recursive equation derived from (3).
3. Reduce the coefficient matrix obtained from step 2 to its reduced echelon (RE) form and compute the coefficients of the GCD from the last row of the reduced matrix.

In [1], we describe the modular and non modular algorithms for computing the GCD of polynomials using the above procedure. With the bound of the true solution not known a priori, we have devised a way of detecting an unlucky reduction and construct a termination criterion, as given in [13], which is necessary for constructing an effective and efficient modular algorithm. Rahman and Aris [1] illustrates the efficiency of the modular algorithm in computing the GCD of two polynomials in the Legendre basis. The computing time takes only seconds when the degree bound for $a(x)$ is 100 and the coefficient bound 153 decimal digits.

In this paper the subalgorithms involved in the computations of the comrade matrix and the construction of its corresponding system of linear equations described in step 1 and 2 above are given in detail so that a detail analysis of their theoretical computing times can be conducted. The results of the analysis will be further applied to study the performance of the entire modular GCD algorithms.

3 The GCD of Generalized Polynomials

Let

$$\begin{aligned}\tilde{d}(x) &= \tilde{d}_0 + \tilde{d}_1 x + ... + x^k, \\ &= d_0 p_0(x) + d_1 p_1(x) + ... + p_k(x) \ , \end{aligned} \tag{5}$$

be the monic GCD of $a(x)$ and $b(x)$ of the form (1) and (2) respectively, such that $n = \deg a(x) \geq \deg b(x) = m$. If A is the comrade matrix given by (4), define

$$b(A) = b_0 I + b_1 p_1(A) + ... + b_m p_m(A), \tag{6}$$

then it is known that $k = n - \text{rank}(b(A))$. Using the defining relation (3), we obtain the rows in terms of the comrade matrix and the recurrence relation

$$\begin{aligned} r_0 &= (b_0, ..., b_{n-1}), \\ r_1 &= r_0(\alpha_0 A), \\ r_i &= r_{i-1}(\alpha_{i-1} A + \beta_{i-1} I) - \gamma_{i-1} r_{i-2} \ . \end{aligned} \tag{7}$$

for $i = 2, ..., n-1$.

Theorem 1. [4] *For $i = 1, 2, ..., n$ let c_i be the ith column of $b(A)$ in (6). The columns $c_{k+1}, ..., c_n$ are linearly independent and the coefficients $d_0, ..., d_{k-1}$ in (5) are given by:*

$$c_i = d_{i-1} c_{k+1} + \sum_{k+2}^{n} x_{ij} c_j \quad i = 1, 2, ..., k \text{ for some } x_{ij} \ . \tag{8}$$

4 Computing Time Analysis

We adopt the convention of dominance relation which was introduced in the analysis of the computing time of the polynomial resultant calculation and is further elaborated in [9]. An exposition of dominance and codominance and its application in determining the computing time analysis of the Euclidean algorithm is given in [8]. The notation have subsequently been adopted by several authors, as in [11] and [7]. Assuming classical matrix addition and multiplication, we apply the results on the computing time for the rational number product, sum, quotient and negative, denoted by the algorithms RNPROD, RNSUM, RNQ and RNNEG respectively, which are analyzed in [8], [9] and [14].

Let P be the set generalized polynomials relative to an orthogonal basis $(p_i(x))_{i=0}^n$ and decompose P into classes of polynomials with rational coefficients whose numerators and denominators are bounded by the integers u and v respectively. This defines the class,

$$\mathrm{P}(u,v,n,(p_i)_{i=0}^n) = \{a = \sum_{i=0}^{k} a_i p_i(x) \in P : |\mathrm{num}(a_i)| \le u, 0 < \mathrm{den}(a_i) \le v, \ni \gcd(u,v) = \gcd(\mathrm{num}(a_i), \mathrm{den}(a_i)) = 1 \text{ and } k \le n\} \ . \quad (9)$$

where num and den is the numerator and denominator of a_i.

5 Construction of the Comrade Matrix

The inputs for the algorithm CDEMOFP are the defining terms α, β, γ given by (3), and the generalized polynomials $a = (a_0, ..., a_{n-1}, 1) = a(x)$ as of Section 2. The output is the comrade matrix $A \in \mathrm{M}(n,n,\mathbb{Q})$. If $a_{ij} = r/s \in \mathbb{Q}$ then$A_{i-1,j-1,0} = A[i-1][j-1][0] = r$ and $A_{i-1,j-1,1} = A[i-1][j-1][1] = s$.

Algorithm CDEMOFP. /*The algorithm construct the comrade matrix (4)*/

begin
Step 1: set $a_{ij} = 0$ for each $i = 1, ..., n$ and $j = 1, ..., n$.
  for $i = 0$ to $n-1$ and $j = 0$ to $n-1$, **do** set $A_{i,j,0} = 0$ and $A_{i,j,1} = 1$
Step 2: assign the diagonal entries for rows 1 to $n-1$
  for $i = 0$ to $n-2$ **do**
    $t \leftarrow \mathrm{RNQ}(-bet(i+1), alph(i+1))$
    if $t = 0$ **then** continue; **else** $A_{i,i,0} \leftarrow t(1)$ and $A_{i,i,1} \leftarrow t(2)$
Step 3: assign the entries for $a_{i,i-1}$ for rows 2 to $n-1$.
  for $i = 1$ to $n-2$ **do**
    $t \leftarrow \mathrm{RNQ}(gamm(i+1), alph(i+1))$
    if $t = 0$ **then** continue; **else** $A_{i,i-1,0} \leftarrow t(1)$ and $A_{i,i-1,1} \leftarrow t(2)$
Step 4: assign the entries for $a_{i,i+1}$ for rows 1 to $n-1$.
  for $i = 0$ to $n-2$ **do**
    if $alph(i+1) = 0$ **then** continue
    else $A_{i,i+1,0} \leftarrow alph_2(i+1)$ and $A_{i,i+1,1} \leftarrow alph_1(i+1)$

Step 5: assign the entries for row n in columns $j = 1$ to $n-2$
  for $j = 0$ to $n-3$ **do**
    $t \leftarrow \mathrm{RNQ}(-a(j+1), alph(n))$
    if $t = 0$ **then** continue; **else** $A_{n-1,j,0} \leftarrow t(1)$ and $A_{n-1,j,1} \leftarrow t(2)$
Step 6: assign the entries $a_{n,n-1}$ and $a_{n,n}$.
  for $j = n-2$ to $n-1$ **do**
    if $j = n-2$ **then** $t \leftarrow \mathrm{RNQ}(\mathrm{RNSUM}(-a(j+1), gamm(n)), alph(n))$
    if $j = n-1$ **then** $t \leftarrow \mathrm{RNQ}(\mathrm{RNSUM}(-a(j+1), -bet(n)), alph(n))$
    if $t = 0$ **then** continue; **else** $A_{n-1,j,0} \leftarrow t(1)$ and $A_{n-1,j,1} \leftarrow t(2)$
Step 7: convert array form to list form: A convert to $comL$
Step 8: output($comL$)
end

Theorem 2. *For the polynomials a and b in $\mathrm{P}(u, v, n, (p_i)_{i=0}^{n})$ such that (p_i) is the set of Legendre basis, $t^{+}_{\mathrm{CDEMOFP}}(u, v, n) \preceq n^2 + n\mathrm{L}(u) + \mathrm{L}(u)\mathrm{L}(\mathrm{L}(u)) + \mathrm{L}(v)$.*

Proof. Step 1 involves copying n^2 entries. Each entry comprises of two components which are 0 and 1. Therefore $t_1 \sim n^2$. Since $t = 0$ in step 2, $t_2 \sim n$.

$t_{\mathrm{RNQ}}(\gamma_i, \alpha_i) \sim 1$. $\sum_{i=1}^{n-2} t_{\mathrm{RNQ}}(\gamma_i, \alpha_i) \preceq n$. For each $1 \leq i \leq n-2$, $\frac{\gamma_i}{\alpha_i} = \frac{i}{2i+1}$. The total cost of copying the numerators and denominators of the entries of the comrade matrix $A(i, i-1)$ for $1 \leq i \leq n-2$ is dominated by $\mathrm{L}(2n+1)$ $\mathrm{L}(n)$. $t_3 \preceq n + \mathrm{L}(n) \sim n$.

For $0 \leq i \leq n-2$, $\frac{1}{\alpha_i} = \frac{i+1}{2i+1}$. The cost of copying the numerators and denominators equals $\sum_{i=0}^{n-2} \mathrm{L}(i+1) + \sum_{i=0}^{n-2} \mathrm{L}(2i+1) \sim \mathrm{L}(2n-1) \sim 1$. Therefore, $t_4 \sim 1$.

For $0 \leq j \leq n-3$, step 5 computes the values $A(n-1, j) = \frac{-a_j}{\alpha_{n-1}}$. $\mathrm{L}(\mathrm{num}(-a_j)) \leq \mathrm{L}(u)$ and $\mathrm{L}(\mathrm{den}(-a_j)) \leq \mathrm{L}(v)$. $\alpha_{n-1} = \frac{2n-1}{n}$. Let $d = \mathrm{L}(\gcd(u, 2n-1))$. $d \leq \mathrm{L}(2n-1) \sim 1$. Let $e = \mathrm{L}(\gcd(v, n))$. $e \leq \mathrm{L}(n) \sim 1$. Therefore $k = \min(d, e) = 1$. $m = \max(\mathrm{L}(-a_j), \mathrm{L}(\alpha_{n-1})) = \mathrm{L}(u)$ and $n = \min(\mathrm{L}(-a_j), \mathrm{L}(\alpha_{n-1})) = 1$. From Theorem 2.4.10, pg 42 in [14], $t_{RNQ}(-a_j, \alpha_{n-1}) \preceq \mathrm{L}(u)$. For each $0 \leq j \leq n-3$, the cost of copying the numerators and denominators of $A(n-1, j)$ is dominated by $\mathrm{L}(u)$. The total cost of all these operations repeated $n-3$ times gives $t_5 \preceq n\mathrm{L}(u)$.

Let $x = -a_{n-2} + \gamma_{n-1}, y = -a_{n-1} - \beta_{n-1}$. $t_{\mathrm{RNNEG}}(a_{n-2}) \preceq \mathrm{L}(u)$ and $t_{\mathrm{RNSUM}}(-a_{n-2}, \gamma_{n-1}) \preceq \mathrm{L}(u)\mathrm{L}(\mathrm{L}(u))$. Since both the numerator and denominator of γ_{n-1} is of $\mathrm{L}(1)$ and $\mathrm{L}(\mathrm{num}(-a_{n-2})) \leq \mathrm{L}(u)$, $\mathrm{L}(\mathrm{den}(-a_{n-2})) \leq \mathrm{L}(v)$. From $\mathrm{L}(x) \preceq \mathrm{L}(u) + \mathrm{L}(v)$ and $\mathrm{L}(\alpha_{n-1}) = 1$, gives $t_{\mathrm{RNQ}}(x, \alpha_{n-1}) \preceq \mathrm{L}(u) + \mathrm{L}(v)$. The total cost in computing $\frac{x}{\alpha_{n-1}}$ is thus dominated by $\mathrm{L}(u)\mathrm{L}(\mathrm{L}(u)) + \mathrm{L}(v)$. Similarly we obtain $t_{\mathrm{RNNEG}}(a_{n-1}) \preceq \mathrm{L}(u)$. Since $\beta_i = 0$ for all i, $t_{\mathrm{RNSUM}}(-a_{n-1}, -\beta_{n-1}) \sim \mathrm{L}(u)$. From $\mathrm{L}(-a_{n-1} - \beta_{n-1}) \leq \mathrm{L}(u) + \mathrm{L}(v)$, we obtain $t_{\mathrm{RNQ}}(y, \alpha_{n-1}) \preceq \mathrm{L}(u) + \mathrm{L}(v)$. So, the total cost of computing the values of $\frac{y}{\alpha_{n-1}}$ is dominated by $\mathrm{L}(u)$. Now let $a = \frac{u}{v} + \frac{n-1}{n}$. $\mathrm{L}(x) \leq \mathrm{L}(a)$. $\frac{a}{\alpha_{n-1}} = \frac{nu+v(n-1)}{v(2n-1)} \leq \frac{u+v}{v}$. Therefore, the cost of copying their numerator and denominator of $\frac{x}{\alpha_{n-1}}$ is dominated by $\mathrm{L}(u) + \mathrm{L}(v)$. The cost of copying the numerator and denominator of $\frac{y}{\alpha_{n-1}}$ is dominated by $\mathrm{L}(u)$. This gives $t_6 \preceq \mathrm{L}(u)\mathrm{L}(\mathrm{L}(u)) + \mathrm{L}(v)$.

For $0 \leq i \leq n-2$ and $0 \leq j \leq n-2$, $\mathrm{L}(A(i,j)) \sim 1$. For $0 \leq j \leq n-3$, $\mathrm{L}(A(n-1,j)) \leq \mathrm{L}(\frac{u}{v}) = \mathrm{L}(u)$.

From step 6, $\mathrm{L}(A(n-1,n-2)) \leq \mathrm{L}(\frac{u+v}{v}) \preceq \mathrm{L}(u) + \mathrm{L}(v)$. Also $\mathrm{L}(A(n-2,n-1)) \leq \mathrm{L}(\frac{u}{v}) = \mathrm{L}(u)$. Thus, $t_7 \preceq (n-1)^2 + n\mathrm{L}(u) + \mathrm{L}(v) \sim n^2 + n\mathrm{L}(u) + \mathrm{L}(v)$.

From steps 1 to 7, $t_{\mathrm{CDEMOFP}}(u,v,n) \preceq n^2 + n\mathrm{L}(u) + \mathrm{L}(u)\mathrm{L}(\mathrm{L}(u)) + \mathrm{L}(v)$, from which Theorem 2 is immediate.

6 The Coefficient Matrix

The algorithm CSFCDEM applies the recurrence relation (7) to construct the coefficient matrix described in Theorem 1. The parameter lists for the inputs α, β and γ are $alph, bet, gamm$ respectively, $b = (b_0, ..., b_m, ..., b_{n-1})$ for the polynomial $b(x)$ as of Section 2, and $CMDE$ for the comrade matrix obtained from Algorithm CDEMOFP.

Algorithm CSFCDEM. /*To construct the coefficient matrix by (7)*/

begin
Step 1: obtain the first row of coefficient matrix from b.
$CS \leftarrow \mathrm{COMP}(b, NIL)$
Step 2: initializing and obtaining the second row of CS.
$R0_{1\times n} \leftarrow CS$
$R1_{1\times n} \leftarrow \mathrm{MRNPROD}(R0_{1\times n}, \mathrm{MRNSPR}(alph(1), CMDE_{n\times n}))$
$CS \leftarrow \mathrm{COMP}(\mathrm{FIRST}(R1_{1\times n}), CS)$
Step 3: apply recurrence equation for remaining rows of CS.
Step3a: composing ith row
for $i = 3$ to n **do**
$S_{n\times n} \leftarrow \mathrm{MRNSPR}(alph(i-1), CMDE_{n\times n})$
$T_{n\times n} \leftarrow \mathrm{MRNSPR}(bet(i-1), I_{n\times n})$
$U_{1\times n} \leftarrow \mathrm{MRNSPR}(-gamm(i-1), R0_{1\times n})$
$R_{1\times n} \leftarrow \mathrm{MRNSUM}(\mathrm{MRNPROD}(R1_{1\times n}, \mathrm{MRNSUM}(S_{n\times n}, T_{n\times n})), U)$
$CS \leftarrow \mathrm{COMP}(\mathrm{FIRST}(R_{1\times n}), CS)$
Step3b: initialization: to set $R0 = R1$
$R0 \leftarrow NIL$ and $R1 \leftarrow \mathrm{FIRST}(R1_{1\times n})$ /*$R1$ in vector representation*/
for $j = 1$ to n **do**
$R0 \leftarrow \mathrm{COMP}(R1(j), R0)$ /*$R0$ is in vector representation*/
$R0 \leftarrow \mathrm{INV}(R0)$ and $R0_{1\times n} \leftarrow \mathrm{COMP}(R0, NIL)$
Step3c: to set $R1 = R$
(similar steps to Step 3b)
$CS \leftarrow \mathrm{INV}(CS)$
Step4: integerize CS
Step5: sorting columns of CS to obtain $CSNEW$ satisfying (8)
Step6: output($CSNEW$)
end

The construction of algorithm CSFCDEM is based upon the recursive equations:

$$R_0 = (b_0, ..., b_m, b_{m+1} + .. + b_{n-1}), \tag{10}$$

$$R_1 = R_0(\alpha_0 A), \tag{11}$$

$$R_i = R_{i-1}(\alpha_{i-1}A) - \gamma_{i-1}R_{i-2}\ , \tag{12}$$

for $2 \leq i \leq n-1$. In (10), $b_j = 0$ for $m < j \leq n-1$ if $m < n-1$. In (12), we assume that $\beta_i = 0$ for all i and is omitted from the equation. We begin by first computing the length of R_i for each $0 \leq i \leq n-1$.

Lemma 1. *Let the polynomials a and b be in $\mathrm{P}(u, v, n, (p_i)_{i=0}^n)$ such that (p_i) is the set of Legendre basis. Let A be the comrade matrix associated with a. Referring to* (10) - (12)*, $\mathrm{L}(R_i(j)) \leq \mathrm{L}(\frac{f(u,v)}{g(v)}) \preceq \mathrm{L}(u) + \mathrm{L}(v)$, for $0 \leq i \leq n-1$ and $1 \leq j \leq n$ where f and g are polynomials in the variables u or v with single precision coefficients and with degree bound $i+1$.*

Proof. For $1 \leq i, j \leq n$, let c_{ij} be the entries of the comrade matrix obtained from algorithm CDEMOFP. But referring to steps 6 and 7 in the proof of Theorem 2, $\mathrm{L}(c_{ij}) \leq \mathrm{L}(z)$ such that $z = \frac{u+v}{v}$. Considering the values of the numerators and denominators that will contribute to the greatest possible length and replacing an element of single precision with 1, we have for $1 \leq j \leq n$, $\mathrm{L}(R_0(j)) \leq \mathrm{L}(\frac{u}{v})$ and $\mathrm{L}(R_1(j)) \leq \mathrm{L}(\frac{u}{v} * z)$. For a fixed j, the function whose length dominates $\mathrm{L}(R_0(j))$ and $\mathrm{L}(R_1(j))$ is a quotient of polynomials in u and v whose coefficients of length 1 and has degree bounds 1 and 2 respectively. This gives $\mathrm{L}(R_2) \leq \mathrm{L}(y)$ such that $y = \frac{u}{v} * z^2 + \frac{u}{v}$, a quotient of polynomials in u and v with coefficient of length 1 and degree bound equals to 3. Proceeding by induction, suppose that for $1 \leq j \leq n$, $\mathrm{L}(R_{l-1}(j)) \leq \mathrm{L}(x1)$ and $\mathrm{L}(R_{l-2}(j)) \leq \mathrm{L}(x2)$ such that

$$x1 = \frac{f_{l-1}(u,v)}{g_{l-1}(v)},\ x2 = \frac{f_{l-2}(u,v)}{g_{l-2}(v)}\ , \tag{13}$$

such that f_{l-k}, g_{l-k}, $k = 1, 2$, has single precision coefficients and degree bounds equal to l and $l-1$ respectively. Then $\mathrm{L}(R_l)(j) \leq \mathrm{L}(y)$ such that $y = x1 * z + x2$. Hence, y is of the form $\frac{f(u,v)}{g(v)}$, where the coefficients of f and g are single precision integers with degree bound equals to $l+1$. Applying the properties of the length of the nonzero integers, that is $\mathrm{L}(a \pm b) \preceq \mathrm{L}(a) + \mathrm{L}(b)$, $\mathrm{L}(ab) \sim \mathrm{L}(a) + \mathrm{L}(b)$ and $\mathrm{L}(a^b) \sim b\mathrm{L}(a)$, $a \geq 2$ in the later, we then have for $0 \leq i \leq n-1$, and $1 \leq j \leq n$, $\mathrm{L}(R_i(j)) \preceq \mathrm{L}(u) + \mathrm{L}(v)$.

Theorem 3. *Let $a, b \in \mathrm{P}(u, v, n, (p_i)_{i=0}^n)$ with (p_i) the Legendre basis. Then $t^+_{\mathrm{CSFCDEM}}(u, v, n) \preceq n^3\mathrm{L}^2(u)(\mathrm{L}(q) + \mathrm{L}(r))$ such that $q = \max(r, n\mathrm{L}(v))$ and $r = \mathrm{L}(u) + \mathrm{L}(v)$.*

Proof. Let $t(V)$ be the computing time to compute the components of the vector V and let the jth component of the vector V be denoted $V(j)$. Let $A = (a_{ij})$ be comrade matrix obtained from Algorithm CDEMOFP.

1. From $\mathrm{L}(b_i) \leq \mathrm{L}(u)$, $0 \leq i \leq m$, $t_1 \preceq m\mathrm{L}(u)$.
2. $t(R0) \preceq m\mathrm{L}(u)$. To compute

$$t(R1) = t_{\mathrm{MRNSPR}}(\alpha_0, A) + t_{\mathrm{MRNPROD}}(R0, \alpha_0 A),$$

(a) $t_{\mathrm{MRNSPR}}(\alpha_0, A) = \sum_{i=1}^{n-1}\sum_{j=1}^{n} t_{\mathrm{RNPROD}}(\alpha_0, a_{ij}) + \sum_{j=1}^{n} t_{\mathrm{RNPROD}}(\alpha_0, a_{nj})$ where $\mathrm{L}(a_{ij}) \sim 1$ for $1 \le i \le n-1$ and $1 \le j \le n$ and $\mathrm{L}(a_{nj}) \preceq \mathrm{L}(u) + \mathrm{L}(v)$ for $1 \le j \le n$. Thus, $t_{\mathrm{MRNSPR}}(\alpha_0 A) \preceq n^2 + n(\mathrm{L}(u) + \mathrm{L}(v))\mathrm{L}(\mathrm{L}(u) + \mathrm{L}(v))$.

(b) Let $1 \le j \le n$. $R1(j) = \sum_{i=1}^{n} R0(i)\alpha_0 a_{ij}$, where $t(R1(j))$ is given by

$$\sum_{i=1}^{n} t_{\mathrm{RNPROD}}(R0(i), \alpha_0 a_{ij}) + \sum_{i=1}^{n-1} t_{\mathrm{RNSUM}}(R0(i)\alpha_0 a_{ij}, R0(i+1)\alpha_0 a_{i+1,j}).$$

$\forall 1 \le i \le n$, $t_{\mathrm{RNPROD}}(R0(i), \alpha_0 a_{ij}) \preceq \mathrm{L}^2(u)\mathrm{L}(\mathrm{L}(u) + \mathrm{L}(v))$. $\forall j$, the total cost of computing these products is dominated by $n\mathrm{L}^2(u)\mathrm{L}(\mathrm{L}(u)+\mathrm{L}(v))$. Taking $\mathrm{L}(R0(i)\alpha_0 a_{ij}) \preceq \mathrm{L}(u) + \mathrm{L}(v), \forall i, j$, $t_{\mathrm{RNSUM}}(R0(i)\alpha_0 a_{ij}, R0(i+1)\alpha_0 a_{i+1,j}) \preceq \{\mathrm{L}^2(u)+\mathrm{L}^2(v)\}\mathrm{L}(\mathrm{L}(u)+\mathrm{L}(v))$. The cost of the $n-1$ sums is dominated by $n\{\mathrm{L}^2(u) + \mathrm{L}^2(v)\}\mathrm{L}(\mathrm{L}(u) + \mathrm{L}(v))$. Summing the cost for all $1 \le j \le n$, $t(R1) \preceq n^2\{\mathrm{L}^2(u) + \mathrm{L}^2(v)\}\mathrm{L}(\mathrm{L}(u) + \mathrm{L}(v))$.

3. For $1 \le i, j \le n$, the respective length of $R_{i-1}(j)$ and $\alpha_{i-1}c_{ij}$ is dominated by $\mathrm{L}(u) + \mathrm{L}(v)$. For $2 \le i \le n-1$, we compute the cost of computing row $i+1$ given by the equation $R_i = R_{i-1}\alpha_{i-1}A - \gamma_{i-1}R_{i-2}$. For a fixed value of i, the jth component of R_i is given by

$$R_i(j) = \sum_{k=1}^{n} R_{i-1}(k)\alpha_{i-1}a_{kj} - \gamma_{i-1}R_{i-2}(j)$$

The cost of computing $\sum_{k=1}^{n} R_{i-1}(k)\alpha_{i-1}a_{kj}$ is dominated by $n(\mathrm{L}(u) + \mathrm{L}(v))^2\mathrm{L}(\mathrm{L}(u) + \mathrm{L}(v))$. $\mathrm{L}(\sum_{k=1}^{n} R_{i-1}(k)\alpha_{i-1}a_{kj}) \sim \sum_{k=1}^{n} \mathrm{L}(R_{i-1}(k)\alpha_{i-1}a_{kj}) \preceq n(\mathrm{L}(u) + \mathrm{L}(v))$. $\mathrm{L}(\gamma_{i-1}R_{i-2}(j)) \preceq \mathrm{L}(u) + \mathrm{L}(v)$. Therefore,

$$t_{\mathrm{RNSUM}}(\sum_{k=1}^{n} R_{i-1}(k)\alpha_{i-1}a_{kj}, -\gamma_{i-1}R_{i-2}(j)) \preceq n(\mathrm{L}(u)+\mathrm{L}(v))^2\mathrm{L}(\mathrm{L}(u)+\mathrm{L}(v)).$$

For each $2 \le j \le n-1$, the total cost of computing $R_i(j)$ is dominated by $n^2(\mathrm{L}(u) + \mathrm{L}(v))^2\mathrm{L}(\mathrm{L}(u) + \mathrm{L}(v))$. It follows that the cost of computing the vector R_i for $2 \le i \le n-1 \preceq n^3(\mathrm{L}(u) + \mathrm{L}(v))^2\mathrm{L}(\mathrm{L}(u) + \mathrm{L}(v)) = t_3'$

4. The integerization of CS in step 4 of algorithm CSFCDEM transforms the rational number matrix CS to its row equivalent integer matrix $CSNEW$. For each row $i = 1$ to n,

(a) find the integer least common multiple (Saclib ILCM) of the n denominators of the n column elements. From Lemma 1, the denominators of $R_i(j)$ for $0 \le i \le n-1$ and $1 \le j \le n$, that will constitute the maximum length of the elements of the comrade matrix is a polynomial $g(v)$ with single precision coefficients and degree, that is, dominated by $\mathrm{L}(v)$. Suppose that at the first $k-2$ iterations, we have computed the LCM of the integers $x_1x_2...x_{k-2}$ and x_{k-1} where x_i constitute the denominators of a certain row of A and suppose that the LCM is of the maximum possible length, which is equal to $x_1x_2...x_{k-1}$. At the end of the $k-1$ iteration we compute the least common multiple,

that is $\mathrm{lcm}(x_1x_2...x_{k-1}, x_k) = \frac{x_1x_2...x_{k-1}x_k}{\gcd(x_1x_2...x_{k-1},x_k)}$, (refer to Theorem 1.4.8 in [14]). Applying the Euclidean algorithm to compute the GCD and applying the results of Theorem 3 in [8], we have

$$t_{\mathrm{M}}(x_1x_2...x_{k-1}, x_k) \sim k\mathrm{L}^2(v) \tag{14}$$

and

$$t_{\mathrm{E}}(x_1x_2...x_{k-1}, x_k) \preceq k\mathrm{L}^2(v) \tag{15}$$

where $y = \gcd(x_1x_2...x_{k-1}, x_k)$ and $\mathrm{L}(x_i), \mathrm{L}(y) \preceq \mathrm{L}(v)$. Likewise, it can be calculated from the results for the computing time of the classical division algorithm [8], which gives $t_{\mathrm{D}}(x_1x_2...x_k, y) \preceq k\mathrm{L}^2(v)$. Thus, the computing time for the $n-1$ iterations in computing the LCM of a row is dominated by $n\mathrm{L}^2(v)$. The computing time for n rows is then dominated by $n^2\mathrm{L}^2(v)$.

(b) multiply the LCM of the denominators of row i with each of the elements in row i to integerize the row. Suppose that the LCM of a row is $x_1x_2...x_n$ whose length is therefore dominated by $n\mathrm{L}(v)$. The length of each numerator of A is dominated by $\mathrm{L}(u)+\mathrm{L}(v)$. For each $1 \le i, j \le n$,

$$t_{\mathrm{RNPROD}}(y, a_{ij}) \preceq n(\mathrm{L}(u)+\mathrm{L}(v))\mathrm{L}(v) \le ngth(\max(\mathrm{L}(u)+\mathrm{L}(v), n\mathrm{L}(v)))$$

which gives a total cost of $n^3(\mathrm{L}(u)+\mathrm{L}(v))\mathrm{L}(v)\mathrm{L}(\max(\mathrm{L}(u)+\mathrm{L}(v), n\mathrm{L}(v)))$, which in general is dominated by n^4.

7 Concluding Remarks and Further Work

Two versions of the modular algorithms have been constructed and implemented by integrating the modular approach from step 2 or step 3 of the procedure described in Section 2. The results for the empirical implementations of these algorithms (refer to [1]), reveal that the execution time for the application of rational number arithmetic in CDEMOFP and CSFCDEM algorithms exceed the execution time for the application of modular arithmetic in step 3, only when the size of the GCD is sufficiently small. When the size of the GCD is sufficiently big, the modular approach of reducing the systems coefficient matrix to its RE form and computing the rational number solution is about the same as the time taken for the rational number arithmetics in CDEMOFP and CSFCDEM algorithms. This suggests that a hybrid of rational number and modular computations can be applied to its utmost advantage in constructing the most efficient algorithm.

The empirical computing time results of the generalized polynomial GCD algorithms presented in [1] can be further studied using the deterministic approach to computing time analysis, as adopted in [8], [11] and [7]. A unified theoretical and empirical computing time analysis can then be concluded for generalized polynomial GCD algorithms.

Acknowledgements

The research is funded by the Fundamental Research Grant, RMC vot 71565, Universiti Teknologi Malaysia. The authors wish to express his gratitude to George Collins for some useful assistance on the implementations of SACLIB and providing some useful references on the theoretical computing time analysis of algorithms. An appreciation is also due to A.A. Rahman and the the referees of ASCM 2007 for some very useful comments and suggestions.

References

1. Rahman, A.A., Aris, N.: The State of the art in Exact polynomial GCD computations. In: Proceedings Malaysian Science and Technology Conference (2002)
2. Rahman, A.A.: The use of GCD computation to remove repeated zeroes from a floating point polynomial. In: Proceedings SCAN 1992, Oldedenberg, Germany (1992)
3. Barnett, S.: A companion matrix analogue for orthogonal polynomials. Linear Algebra and its Applications 12, 197–208 (1975)
4. Barnett, S., Maroulas, J.: Greatest common divisor of generalized polynomial and polynomial matrices. Linear Algebra and its Applications 22, 195–210 (1978)
5. Barnett, S.: Polynomial and Linear Control Systems. Marcel Dekker, New York (1983)
6. Barnett, S.: Division of generalized polynomials using the comrade matrix. Linear Algebra and its Applications 60, 159–175 (1984)
7. Brown, W.S.: On Euclid's algorithm and polynomial greatest common divisors 18(1), 478–504 (1971)
8. Collins, G.E.: The computing time of the Euclidean algorithm. SIAM Journal on Computing 3(1), 1–10 (1974)
9. Collins, G.E., Mignotte, M., Winkler, F.: Arithmetic in basic algebraic domains: Computing Suppl. 4, 189–220 (1982)
10. Labahn, G., Cheng, H.: On computing polynomial GCDs in alternate bases. In: Proceedings ISSAC 2006, pp. 47–54 (2006)
11. McClellan, M.T.: The exact solution of systems of linear equations with polynomial coefficients. J. ACM 20(4), 563–588 (1973)
12. McClellan, M.T.: A comparison of algorithms for the exact solutions of linear equations. ACM Trans. Math. Software 3(2), 147–158 (1977)
13. Aris, N., Rahman, A.A.: On the division of generalized polynomials. Lecture Notes Series On Computing, vol. 10, pp. 40–51. World Scientific Computing, Singapore (2003)
14. Rubald, C.M.: Algorithms for Polynomials over a Real Algebraic Number Field. University of Wisconsin: Ph.D. Thesis (1973)

Efficient Algorithms for Computing Nœther Normalization

Amir Hashemi

[1] Department of Mathematical Sciences, Isfahan University of Technology, Isfahan 84156-83111, Iran
Amir.Hashemi@cc.iut.ac.ir
[2] Inria-Salsa project/ Lip6-Spiral team, 104 Avenue du President Kennedy, 75016 Paris, France
Amir.Hashemi@lip6.fr

Abstract. In this paper, we provide first a new algorithm for testing whether a monomial ideal is in Nœther position or not, *without using its dimension*, within a complexity which is quadratic in input size. Using this algorithm, we provide also a new algorithm to put an ideal in this position within an *incremental* (one variable after the other) random linear change of the last variables without using its dimension. We describe a modular (probabilistic) version of these algorithms for any ideal using the modular method used in [2] with some modifications. These algorithms have been implemented in the distributed library noether.lib [17] of Singular, and we evaluate their performance via some examples.

1 Introduction

The aim of this paper is first to present a new simple algorithm to decide whether a monomial ideal is in Nœther position or not. This algorithm has a complexity which is quadratic in input size, and it does not need to use the dimension of the ideal. Moreover, if the response is positive, it may return also the dimension of the ideal without any more computation. We provide also a modular version of this algorithm for any ideal in which we use a modular method for computing the Gröbner bases with a high probability.

We provide also a new algorithm to put an ideal in Nœther position using the above algorithms without using the dimension of the ideal. For this, it makes an incremental (one variable after the other) random linear change of variables. The computation of Gröbner bases by this strategy is less expensive than the case it changes all the last variables in one step. Also, we show that for some examples we do not need to put all the last variables (whose cardinal is the dimension of the ideal) in generic position. These show that doing this computation incrementally is more efficient than doing it in one step. All the results have been implemented by author in the distributed library noether.lib [17] of Singular [16]. We compare the performance of our algorithms with the functions `NoetherPosition` (from library mregular.lib [5]) of Singular and `noetherNormalization` (from library NoetherNormalization.m2) of Macaulay2 [14] via some examples.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 97–107, 2008.
© Springer-Verlag Berlin Heidelberg 2008

To explain our motivation on working on the notion of Nœther position, let R be the polynomial ring $K[x_1, \ldots, x_n]$ over an arbitrary field K, and I be an ideal of R which has dimension d. The ideal I is in *Nœther position* if $K[x_{n-d+1}, \ldots, x_n] \hookrightarrow R/I$ is an integral ring extension, i.e. the image in R/I of x_i for any $i = 1, \ldots, n-d$ is a root of a polynomial

$$X^s + g_1 X^{s-1} + \cdots + g_s = 0$$

where s is an integer and $g_1, \ldots, g_s \in K[x_{n-d+1}, \ldots, x_n]$ (see [11] for example). This notion is used in affine ring theory, dimension theory, ring normalization and primary decomposition (see [15] Chapter 3). For example, it was used by Giusti et al. to compute the dimension of a variety (see [13]), by Krick and Logar to compute the radical of an ideal (see [19]), by Lecerf to solve a system of polynomial equations and inequations (see [18]) and by Bardet to bound the complexity of Gröbner bases computation by Faugère's F_5 algorithm (see [3]).

Now, we give the structure of the paper. In Section 2, we present our algorithm to decide whether a monomial ideal is in Nœther position or not. In Section 3, we describe a modular algorithm to test whether a general ideal is in Nœther position or not. In Section 4, we provide our algorithms to put an ideal in Nœther position. In Section 5, we show the performance of our algorithms with the existing algorithms of Singular and Macaulay2 via some examples. Section 6 presents our conclusions.

2 Nœther Position Test

In this section, we provide a new algorithm to decide whether a monomial ideal is in Nœther position or not. This algorithm has a complexity which is quadratic in input size, and it does not need the dimension of the ideal. Moreover, if the answer is positive, it returns also the dimension of the ideal without any more computation. For this, let us recall a Nœther position test from [6] and a lemma to compute the dimension of a monomial ideal from [9] page 431.

Let $R = K[x_1, \ldots, x_n]$ be a polynomial ring over an arbitrary field K, and $J \subset R$ be a monomial ideal of dimension d (recall that the dimension $\dim(X)$ of an ideal X is the dimension of the corresponding quotient ring).

Lemma 1 (Nœther position test). *The following conditions are equivalent:*

(1) *J is in Nœther position.*
(2) *There exists $r_i \in \mathbb{N} - \{0\}$ for any $i = 1, \ldots, n-d$ such that $x_i^{r_i} \in J$.*
(3) $\dim(J + \langle x_{n-d+1}, \ldots, x_n \rangle) = 0$.

Lemma 2. *Let*

$$\mathcal{M} = \left\{ \{x_{i_1}, \ldots, x_{i_t}\} \subset \{x_1, \ldots, x_n\} \mid x_{i_1} \ldots x_{i_t} \notin \sqrt{J} \ \text{ and } \ |\{i_1, \ldots, i_t\}| = t \right\}$$

where $|A|$ denotes the number of the elements of a set A. Then, $\dim(J)$ *is equal to* $\max\{|L| \mid L \in \mathcal{M}\}$.

Algorithm 1. NP-test

Input: $\{m_1, \ldots, m_k\}$ a system of generators for a monomial ideal $J \subset R$
Output: The answer of "Is J in Nœther position?" and $\dim(J)$ if the answer is positive
$Tab := [0, \ldots, 0]$ (zero vector of length n)
for $i = 1, \ldots, k$ **do**
  $j :=$ the smallest integer ℓ s.t. $x_\ell \mid m_i$
  if there exists α s.t. $m_i = x_j^\alpha$ **then**
    $Tab[j] := 1$
  end if
  if $Tab[j] = 0$ **then**
    $Tab[j] := -1$
  end if
end for
if there is j s.t. $Tab[j] = -1$ **then**
  return "No"
end if
if there is j s.t. $Tab[j] = 1$ and $Tab[j-1] = 0$ **then**
  return "No"
end if
if $Tab = [1, \ldots, 1, 0, \ldots, 0]$ (with d components equal to 0) **then**
  return "Yes, and dimension= d"
end if

Using these lemmas, we describe an algorithm to test whether a monomial ideal is in Nœther position or not.

Proof of algorithm. The termination of the algorithm is trivial. To prove its correctness, we justify its answer in each case. Let d' be the dimension of J. Suppose that there exists an integer j such that $Tab[j] = -1$. Two cases are possible: If $j < n - d' + 1$ there is not any power of x_j which belongs to J. Thus J is not in Nœther position by Lemma 1(2). If $j \geq n - d' + 1$ then $x_j \cdots x_n \in \sqrt{J}$ from definition of j, and thus if J is in Nœther position $\dim(J) < n - j + 1 \leq d'$ by Lemma 2 which is a contradiction. Therefore, in this case the response is "No". Now, suppose that there exists an integer j such that $Tab[j] = 1$ and $Tab[j-1] = 0$. This implies that J is not in Nœther position from Lemma 1(2) and definition of notations. Finally, suppose that $Tab = [1, \ldots, 1, 0, \ldots, 0]$ with d components equal to 0. It is enough to prove (ad absurdum) that $d = d'$ by Lemma 1. If $x_{n-d+1} \cdots x_n \in \sqrt{J}$ there exists α_i such that $x_{n-d+1}^{\alpha_{n-d+1}} \cdots x_n^{\alpha_n} \in J$. Therefore, $Tab[n - d + 1] = -1$ from definition of notations which is a contradiction. This follows that $x_{n-d+1} \cdots x_n \notin \sqrt{J}$ and thus the dimension of J is equal to d using Lemma 2 and the fact that $x_1, \ldots, x_{n-d} \in \sqrt{J}$. □

Remark 1. The integer d is the dimension of J if the answer of the algorithm is "Yes" (see the proof of algorithm).

Theorem 1. *The complexity of this algorithm is quadratic in kn.*

Proof. One can easily see that the number of operations in the loop "For" is kn^2. Thus the complexity of the algorithm is quadratic in kn. □

Example 1. There is a monomial ideal for which `NP-test` *algorithm runs faster than* `is-NP` *algorithm from library mregular.lib of* SINGULAR. For the monomial ideals that the computation of the dimension is difficult, our algorithm is more efficient than `is-NP`. For example, let $J = \langle x_0x_1, x_1x_2, \ldots, x_{48}x_{49}, x_{49}x_{50}\rangle$ be a monomial ideal of the ring $\mathbb{Q}[x_0, \ldots, x_{50}]$ (see [8]). `NP-test` returns "No" in 0.1 sec. while `is-NP` returns "No" in 27 sec..

3 Modular Nœther Position Test

In this section, we describe a modular algorithm to test whether a general ideal is in Nœther position or not. For this, we recall first Arnold's method (with some modifications) to find a "lucky" prime for computing the initial ideal of the input ideal.

Let $I = \langle f_1, \ldots, f_k\rangle$ be an ideal of the ring $R = \mathbb{Q}[x_1, \ldots, x_n]$ where $\mathbb{Q}$ is the field of rational numbers. We scale appropriately such that each f_i is in $\mathbb{Z}[x_1, \ldots, x_n]$ with $\mathbb{Z}$ the ring of integer numbers. We consider the ideal $I_p = \langle \bar{f}_1, \ldots, \bar{f}_k\rangle$ in the ring $\mathbb{Z}_p[x_1, \ldots, x_n]$ where p is a prime number.

Let us recall the definition of the *degree reverse lexicographic ordering*, denoted by $\prec$, on the monomials of R. For this, we denote respectively by $\deg(m)$ and $\deg_i(m)$ the total degree and the degree in x_i of a monomial m. If m and m' are monomials, then $m \prec m'$ if and only if the last non zero entry in the sequence $(\deg_1(m') - \deg_1(m), \ldots, \deg_n(m') - \deg_n(m), \deg(m) - \deg(m'))$ is negative. Thus we have $x_n \prec x_{n-1} \prec \cdots \prec x_1$. Let also $\mathrm{in}(f) \in R$ be the initial (greatest) monomial of a polynomial $f \in R$ with respect to $\prec$. Then, the initial ideal of I is defined $\mathrm{in}(I) = \langle \mathrm{in}(f) \mid f \in I\rangle$.

To test whether a general ideal I is in Nœther position or not, it is wise to test it on $\mathrm{in}(I)$ (see [6] Lemma 3.1). To compute $\mathrm{in}(I)$, we have to compute the Gröbner basis of I. But during the computation of Gröbner basis (by the Buchberger algorithm), the coefficients of the intermediate polynomials may become much too big and make the computation fail. Thus, an efficient way is to use a modular method to compute $\mathrm{in}(I)$. For this purpose, Pauer [21] has defined a *Pauer lucky* prime. A prime p is Pauer lucky for I if p does not divides any leading coefficient of any polynomial in the reduced Gröbner basis of I. He has proved that for such a prime p, we have $\mathrm{in}(I) = \mathrm{in}(I_p)$. But, he did not provide any criterion for detecting a Pauer lucky prime. Thus, Arnold [2] (see also [1]) has proposed an efficient method to find (with a high probability) such a prime. For this, she has defined a *Hilbert lucky* prime, and she has proved that it is equivalent to Pauer prime ([2] Theorem 5.13). Here, we recall a Hilbert lucky

prime with some modifications which simplifies its computation. Recall that the *Hilbert function* of $I \subset R$ is defined by:

$$\mathrm{HF}_I(t) = \dim_{\mathbb{Q}} \left(\frac{R}{I}\right)_{\leq t}$$

where $\dim_{\mathbb{Q}}(R/I)_{\leq t}$ is the dimension of the set of the elements of degree at most t of R/I as a $\mathbb{Q}$-vector space. The *Hilbert series* of an ideal I is the following power series:

$$\mathrm{HS}_I(t) = \sum_{s=0}^{\infty} \mathrm{HF}_I(s) t^s.$$

Definition 1. *A prime number p is called* Hilbert lucky *for I if* $\mathrm{HS}_I = \mathrm{HS}_{I_p}$.

Arnold has proposed the same definition replacing the Hilbert series by the Hilbert function. Our modification does not change the definition, but it allows us to choose a Hilbert lucky prime more efficiently. To prove this claim, let us recall the following proposition from [12] page 130.

Proposition 1. *The Hilbert series of an ideal I has the form:*

$$\mathrm{HS}_I(t) = \frac{P(t)}{(1-t)^d}$$

where P is a polynomial such that $P(1) \neq 0$ and d is the dimension of I.

To determine the relative Hilbert luckiness of two primes p and q, we define an ordering on the Hilbert series of two ideals: Let d (resp. d') be the dimension of I_p (resp. I_q) and N (resp. N') be the numerator of HS_{I_p} (resp. HS_{I_q}). Then, $\mathrm{HS}_{I_p} > \mathrm{HS}_{I_q}$ if either $d > d'$ or if $d' = d$ the leading coefficient of $N - N'$ is positive. In this case, we say that p is unlucky with respect to q. While by Arnold's criterion p is unlucky with respect to q if $\mathrm{HF}_{I_p}(t) > \mathrm{HF}_{I_q}(t)$ for some t. This criterion is equivalent to our criterion, but in general, *Arnold's criterion needs more computation than our criterion because sometimes one has to do this comparison up-to Hilbert regularity* (the degree from which Hilbert function is polynomial). For example, let m be an arbitrary positive integer. Let

$$I = \langle x^{2m+1}, xy, y^{m+1}, yz^m \rangle$$

be an ideal of the ring $R = K[x, y, z]$ where K is an infinite field. One can see from Hilbert series of I that its Hilbert regularity is m. On the other hand, Hilbert series of I may be computed by $2^4 = 16$ operations (see [4] for example). Thus, Hilbert luckiness of two primes may be computed with 32 operations using our criterion while it needs at least $2m$ operations by Arnold's criterion (see [1]). So, for m enough big our method is more efficient.

When $\mathrm{HS}_{I_p} = \mathrm{HS}_{I_q}$, to be able to choose a more lucky prime between two primes, we define an ordering on the initial ideals modulo the primes: $\mathrm{in}(I_q) > \mathrm{in}(I_p)$ if there exists a polynomial in the Gröbner basis of I_q which has a leading

term greater than that of its corresponding polynomial in the the Gröbner basis of I_p. By [2] Theorem 5.6 if $\mathrm{HS}_{I_p} = \mathrm{HS}_{I_q}$, we choose (from p and q) the prime which has the greater initial ideal.

We describe now an algorithm to find a Hilbert lucky prime. In this algorithm, using the above criterion, we compare a candidate prime with the best Hilbert lucky prime which has already been computed. In the beginning, we start by a random prime number.

Algorithm 2. HLP

Input: $I \subset R$ an ideal, p a (Hilbert lucky) prime number, G_p the Gröbner basis of I_p with respect to $\prec$ and HS_{I_p}
Output: A Hilbert lucky prime q, Gröbner basis of I_q and HS_{I_q}
Flag:= 0
while Flag= 0 **do**
  $q :=$ a random prime number
  $G_q :=$ the Gröbner basis of I_q with respect to $\prec$
  if $\mathrm{HS}_{I_p} > \mathrm{HS}_{I_q}$ or ($\mathrm{HS}_{I_p} = \mathrm{HS}_{I_q}$ and $\mathrm{in}(I_q) >= \mathrm{in}(I_p)$) **then**
    Flag:= 1
  end if
end while
return $(q, G_q, \mathrm{HS}_{I_q})$

Then, using this algorithm, we describe a p-adic algorithm to test (with a high probability) whether a general ideal is in Nœther position or not.

Algorithm 3. MNP-test

Input: $I \subset R$ an ideal
Output: The answer of "Is I in Nœther position?"
$p :=$ a random prime number
$G_p :=$ the Gröbner basis of I_p with respect to $\prec$
$(q_1, G_{q_1}, \mathrm{HS}_{I_{q_1}}) :=\mathrm{HLP}(I, p, G_p, \mathrm{HS}_{I_p})$
$(q_2, G_{q_2}, \mathrm{HS}_{I_{q_2}}) :=\mathrm{HLP}(I, q_1, G_{q_1}, \mathrm{HS}_{I_{q_1}})$
while NP-test($\mathrm{in}(G_{q_1})$) $\neq$ NP-test($\mathrm{in}(G_{q_2})$) **do**
  $G_{q_1} := G_{q_2}$
  $(q_2, G_{q_2}, \mathrm{HS}_{I_{q_2}}) :=\mathrm{HLP}(I, q_2, G_{q_2}, \mathrm{HS}_{I_{q_2}})$
end while
return NP-test($\mathrm{in}(G_{q_1})$)

Example 2. There is an ideal for which `MNP-test` *algorithm runs faster than* `NP-test` *and* `is-NP` *algorithm from library mregular.lib of* SINGULAR. Let I be the ideal of cyclic 6-roots over $\mathbb{Q}$. Then, `MNP-test` returns "Yes" in 2.1 sec. while `NP-test` and `is-NP` could not return the answer in less than 12 hours.

4 Putting an Ideal in Nœther Position

In this section, using `NP-test` and `MNP-test` algorithms, we provide two incremental and efficient algorithms to put an ideal in Nœther position. For this, we recall first Nœther normalization lemma from [15] page 213.

Lemma 3 (Nœther normalization). *Let $I \subset K[x_1, \ldots, x_n]$ be an ideal and K be a field. Then there exist an integer $s \leq n$ and an isomorphism*

$$\phi : K[x_1, \ldots, x_n] \longrightarrow A := K[y_1, \ldots, y_n]$$

such that canonical map $K[y_{s+1}, \ldots, y_n] \longrightarrow A/\phi(I)$, $y_i \longmapsto y_i \bmod \phi(I)$ is injective and finite. Moreover, if K is infinite then ϕ can be chosen to be linear, $\phi(x_i) = a_{i,1}y_1 + \cdots + a_{i,n}y_n$ with $a_{i,j} \in K$.

Definition 2. *A finite and injective map $K[y_{s+1}, \ldots, y_n] \longrightarrow A/\phi(I)$ is called a* Nœther normalization *of $A/\phi(I)$, and ϕ puts I in Nœther position.*

This lemma states that an ideal can be put in Nœther position by a suitable change of the variables. If the characteristic of K is zero, we show that for almost all linear change of variables an ideal is in Nœther position. Let $I \subset R$ be an ideal of dimension d. For all $\Lambda = (a_1, \ldots, a_N) \in K^N$ with $N = nd - d(d-1)/2$, denote by $\Lambda(I)$ the result of the following substitution in I:

$$\begin{aligned}
x_n &\mapsto x_n + a_1 x_{n-1} + a_2 x_{n-2} &&+ \cdots + a_{n-1}x_1 \\
x_{n-1} &\mapsto x_{n-1} + a_n x_{n-2} &&+ \cdots + a_{2n-3}x_1 \\
&\vdots \\
x_{n-d+1} &\mapsto x_{n-d+1} + a_{N-n+d+1}x_{n-d} &&+ \cdots + a_N x_1
\end{aligned}$$

By [7] Theorem A.1, there exists a dense open Zariski subset U of $\mathbb{A}_K^N$ such that $\Lambda(I)$ is in Nœther position for all $\Lambda \in U$.

Algorithm 4. NP

Input: $I \subset R$ an ideal
Output: A set of linear maps ϕ such that $\phi(I) \subset R$ is in Nœther position
$I' :=$ the Gröbner basis of I with respect to $\prec$
if NP-test(in(I')) ="Yes" **then**
  Return The ideal is already in Nœther position
end if
$\phi := \{\}$
for $i = n, \ldots, 2$ **do**
  $\psi :=$ The map $x_i \longmapsto x_i + a_{i,i-1}x_{i-1} + \cdots + a_{i,1}x_1$ with $a_{i,j} \in K$ a random element
  $\phi := \phi \cup \{\psi\}$
  $I' :=$ the Gröbner basis of $\psi(I')$ with respect to $\prec$
  if NP-test(in(I')) ="Yes" **then**
    Return ϕ
  end if
end for

It is certain that if we do this change of variables in one step, it destroys the possible sparsity and may therefore transform an easy problem in an intractable one. Thus, as an interesting algorithmic issue, it is better to do it incrementally

(one variable after the other) than to do it in one step. This follows from the fact that the computation of Gröbner bases by this strategy is less expensive (see Section 5). Also, it is certain that the number of variables that one has to change is at most the dimension of the input ideal (see Lemma 1). But, by this strategy, this number may be strictly less than than the dimension of the ideal (see Section 5 for some examples). Another technique is to verify if a permutation of the variable could put the ideal in this position in the case that the input ideal is not in Nœther position. For example, let $I = \langle y^2, z^2 \rangle$ be an ideal of the ring $K[x, y, z]$. Thus, for this ideal we do not need to change any variable, and only the permutation $x \longmapsto z$, $y \longmapsto y$ and $z \longmapsto x$ could put I in Nœther position (see Section 5 for more examples). Also, as the Gröbner basis of the input ideal has to be computed for testing Nœther position (see Section 3), one could use it to minimize the time of computation and the number of non-zero coefficients in the change of variables.

We describe an algorithm to put an ideal in Nœther position by the above techniques. Then, replacing `NP-test` by `MNP-test`, we can describe a modular (probabilistic) algorithm to do this computation when $K = \mathbb{Q}$.

5 Experiments and Remarks

We have implemented `NP-test`, `HPL`, `MNP-test`, `NP` and `MNP` algorithms with the computer algebra system SINGULAR (version 3-0-1), and they were distributed in the library noether.lib [17] of this system. We choose SINGULAR since it has many functionalities needed related to a multivariate polynomial ring. Also, to be able to compare our algorithms with the existing `NoetherPosition` function (from library mregular.lib [5]) of SINGULAR which is base on Bermejo et al.'s paper [6].

Table 1. Comparison of our algorithms with SINGULAR and MACAULAY2

			MNP		NP		SINGULAR		MACAULAY2	
Example	var	dim	ch var	time	ch var	time	ch var	time	ch var	time
1	4	1	1	11.6	1	∗322	1	∞	?	?
2	5	3	2	0.2	2	0.2	3	0.2	3	0.1
3	5	3	0	0.1	0	0.1	3	1	3	1.9
4	5	1	0	0.1	0	0.2	1	0.7	?	?
5	5	1	1	∗6228	1	∞	1	∞	?	?
6	6	3	3	0.7	3	1.8	3	3.4	?	?
7	7	2	2	1.5	2	6.8	2	133.2	2	12.9
8	8	2	1	0.6	1	2.6	2	288.4	?	?
9	9	3	3	0.4	3	0.6	3	1.4	?	?
10	9	4	4	2.3	4	∗1.8	4	13203.1	?	?
11	10	1	1	34.8	1	∗597	1	∞	1	∗0.2
12	10	9	0	0.6	0	∗0.1	9	130.2	0	0.1
13	11	5	5	0.8	5	2.4	5	29.7	?	?
14	17	3	3	44.6	3	∗190.6	3	12495.5	3	∗9.6

We compare also our implementation with the function `noetherNormalization` (from library NoetherNormalization.m2) of MACAULAY2 [14] which is based on Logar's paper [20].

For this experiments, we use some examples from [10] and Posso list[1]. The results are shown in Table 1. All the computations are done over $\mathbb{Q}$. The monomial ordering is always degree reverse lexicographical ordering. In this table, Example 1 (Haas Example[2]) is the ideal $\langle x^8 + zy^4 - y, y^8 + tx^4 - x, 64x^7y^7 - 16x^3y^3zt + 4y^3z + 4x^3t - 1\rangle$ of the ring $\mathbb{Q}[x, y, z, t]$. Examples $3, 4, 6, 7, 8, 9, 10, 11, 12, 14$ are respectively Macaulay, Cassou, Horrocks, Gerdt, Möller, Sturmfels/Eisenbud, Butcher, Wang1, Shimoyama/Yokoyama and Gonnet Examples from [10]. Examples 2 and 5 are respectively Bronstein2 and Czapor Examples from Posso list. Example 13 is the ideal $\langle x_0x_1, x_1x_2, \ldots, x_9x_{10}, x_{10}x_0\rangle \subset \mathbb{Q}[x_0, \ldots, x_{10}]$ from [8].

In Table 1, column var (resp. dim and ch var) shows the number of variables (resp. the dimension of the ideal and the number of variables that the algorithm changes to put the ideal in Nœther position). The column time shows the timing of computing. Timing is measured in seconds (with a precision of tenths of a second) The symbol ∞ means more than 24 hours and the symbol ? indicates that memory was exhausted (memory error message for MACAULAY2 was: Too many heap sections: Increase MAXHINCR or MAX-HEAP-SECTS) and it made the computation fail. The symbol $*$ shows the timing which is important in comparison with the other timings. Computations were conducted on a personal computer with 3.2GHz, Intel Pentium 4 and 1024 MB memory under the Linux operating system.

For Example 11, the reason for which the function `noetherNormalization` is faster than `NP` is that the former makes the linear change in a space of dimension 1 while the later makes it in a space of dimension 9.

The experiments we made seem to show that these first implementations are already very efficient. As one can see in Table 1, the comparison of the columns time of `NP` and the others shows that doing an incremental (one variable after the other) random linear change of variables makes the computation of Gröbner bases less expensive than the case that we change all the last variables in one step. Also, for some examples, we do not need to change any variable, and a permutation of variables could put the ideal in Nœther position (in column of ch var, we show this case by 0).

6 Conclusion

In this paper, we have presented first two algorithms `NP-test` and `MNP-test` to test whether a given ideal is in Nœther position or not. `NP-test` is a deterministic algorithm which has a quadratic complexity in input (a monomial ideal) size. This algorithm does not use the dimension of the ideal. `MNP-test` algorithm whose input ideal is a general ideal, uses a modular (probabilistic) algorithm to compute the initial ideal and `NP-test` algorithm for Nœther position test. Then,

[1] `http://www-sop.inria.fr/saga/POL/BASE/3.posso/`

[2] `http://www.math.tamu.edu/~rojas/extreme.html`

we have presented two algorithms `NP` (using `NP-test`) and `MNP` (using `MNP-test`) to put an ideal in such a position via an incremental random linear change of variables. The advantages of these algorithms are: *The computation of Gröbner bases by this strategy is less expensive than the case that we change all the last variables in one step. Also, for some systems of positive dimension we do not need to change all the last variables. Finally, it does not use the dimension of the ideal.*

Acknowledgment

The results of this paper are from my Ph.D. thesis at university of Paris VI. I would like to thank my advisor prof. Daniel Lazard for his very helpful comments. This work was supported in part by the CEAMA, Isfahan University of Technology, Isfahan 84156, Iran.

References

1. Arnold, E.A.: Computing Gröbner Bases with Hilbert Lucky Primes. PhD thesis, University of Maryland (2000)
2. Arnold, E.A.: Modular algorithms for computing Gröbner bases. J. Symbolic Comput. 35(4), 403–419 (2003)
3. Bardet, M.: Étude des systèmes algébriques surdéterminés: Applications aux codes correcteurs et à la cryptographie. PhD thesis, Université Paris6 (2004)
4. Bayer, D., Stillman, M.: Computation of Hilbert functions. J. Symbolic Comput. 14(1), 31–50 (1992)
5. Bermejo, I., Gimenez, Ph., Greuel, G.-M.: mregular.lib. A Singular 3.0.1 distributed library for computing the Castelnuovo-Mumford regularity (2005)
6. Bermejo, I., Gimenez, P.: Computing the Castelnuovo-Mumford regularity of some subschemes of $\mathbb{P}^n_K$ using quotients of monomial ideals. J. Pure Appl. Algebra 164(1-2), 23–33 (2001); Effective methods in algebraic geometry (Bath, 2000)
7. Bermejo, I., Gimenez, P.: Saturation and Catelnuovo-Mumford regularity. J. Algebra 303, 592–617 (2006)
8. Bigatti, A.M., Conti, P., Robbiano, L., Traverso, C.: A Divide and Conquer algorithm for Hilbert-Poincaré series, multiplicity and dimension of monomial ideals. In: Moreno, O., Cohen, G., Mora, T. (eds.) AAECC 1993. LNCS, vol. 673, pp. 76–88. Springer, Heidelberg (1993)
9. Cox, D., Little, J., O'Shea, D.: Ideals, varieties, and algorithms. In: Undergraduate Texts in Mathematics, 2nd edn. Springer, New York (1997); (An introduction to computational algebraic geometry and commutative algebra)
10. Decker, W., Greuel, G.-M., Pfister, G.: Primary decomposition: algorithms and comparisons. In: Algorithmic algebra and number theory (Heidelberg, 1997), pp. 187–220. Springer, Berlin (1999)
11. Eisenbud, D.: Commutative algebra with a view toward algebraic geometry. Graduate Texts in Mathematics, vol. 150. Springer, New York (1995)
12. Fröberg, R.: An introduction to Gröbner bases, New York. Pure and Applied Mathematics. John Wiley & Sons Ltd., Chichester (1997)

13. Giusti, M., Hägele, K., Lecerf, G., Marchand, J., Salvy, B.: The projective Noether Maple package: computing the dimension of a projective variety. J. Symbolic Comput. 30(3), 291–307 (2000)
14. Grayson, D.R., Stillman, M.E.: Macaulay 2 version 1.1. software system for research in algebraic geometry, http://www.math.uiuc.edu/Macaulay2/
15. Greuel, G.-M., Pfister, G.: A Singular introduction to commutative algebra. Springer, Berlin (2002); With contributions by Olaf Bachmann, Christoph Lossen and Hans Schönemann, With 1 CD-ROM (Windows, Macintosh, and UNIX)
16. Greuel, G.-M., Pfister, G., Schönemann, H.: Singular 3.0.3. A Computer Algebra System for Polynomial Computations, Centre for Computer Algebra, University of Kaiserslautern (2005), http://www.singular.uni-kl.de
17. Hashemi, A.: noether.lib. A Singular 3.0.3 distributed library for computing the nœther normalization (2007)
18. Lecerf, G.: Computing the Equidimensional Decomposition of an Algebraic Closed Set by means of Lifting Fibers. Journal of Complexity 19(4), 564–596 (2003)
19. Krick, T., Logar, A.: An algorithm for the computation of the radical of an ideal in the ring of polynomials. In: Mattson, H.F., Rao, T.R.N., Mora, T. (eds.) AAECC 1991. LNCS, vol. 539, pp. 195–205. Springer, Heidelberg (1991)
20. Logar, A.: A Computational Proof of the Noether Normalization Lemma. In: Mora, T. (ed.) AAECC 1989. LNCS, vol. 357, pp. 259–273. Springer, Heidelberg (1989)
21. Pauer, F.: On lucky ideals for Gröbner basis computations. J. Symbolic Comput. 14, 471–482 (1992)

Stability of GPBiCG_AR Method Based on Minimization of Associate Residual

Moe Thuthu[1] and Seiji Fujino[2]

[1] Graduate School of Information Science and Electrical Engineering
moethu@zeal.cc.kyushu-u.ac.jp
[2] Research Institute for Information Technology,
Kyushu University, Japan
fujino@cc.kyushu-u.ac.jp

Abstract. GPBi-CG method is an attractive iterative method for the solution of a linear system of equations with nonsymmetric coefficient matrix. However, the popularity of GPBi-CG method has diminished over time except for the minority. In this paper, we consider a new algorithm based on minimization of the associate residual of 2-norm in place of reconstruction of the algorithm. We refer to a method with new algorithm as GPBiCG with Associate Residual (abbreviated as GPBiCG_AR) method. Moreover we will introduce preconditioned GPBiCG_AR (abbreviated as P_GPBiCG_AR). Then, we will support that GPBiCG_AR and P_GPBiCG_AR methods yield safety convergence through numerical experiments.

Keywords: GPBi-CG, GPBiCG_AR, nonsymmetric coefficient matrix, Associate Residual, precondition.

1 Introduction

Generalized Product Bi-Conjugate Gradient (abbreviated as GPBi-CG) method [6] is an attractive iterative method for the solution of a linear system of equations with nonsymmetric coefficient matrix. However, the popularity of GPBi-CG method has diminished over time except for the context of limited field of analysis becuase of unstability of convergence rate. Therefore some versions of GPBi-CG method which have stability of convergence compared with the original GPBi-CG method has been proposed.

We proposed a safety variant (abbreviated as BiCGSafe) of Generalized Product type Bi-CG method from the viewpoint of reconstruction of the residual polynomial and determination of two acceleration parameters ζ_n and η_n [3]. It embodied that a particular strategy for remedying the instability of convergence, acceleration parameters are decided from minimization of the associate residual of 2-norm [4]. However, we could not reveal the origin of instability of GPBi-CG method because of reconstruction of the algorithm. Though both convergence rate and stability of BiCGSafe method were improved, instability itself of GPBi-CG method could not correspond directly to its algorithm.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 108–120, 2008.
© Springer-Verlag Berlin Heidelberg 2008

In this paper, we consider a new algorithm based on minimization of the associate residual of 2-norm in place of reconstruction of the algorithm. We refer to as GPBiCG with Associate Residual (abbreviated as GPBiCG_AR) method. Moreover we will introduce preconditioned GPBiCG_AR (abbreviated as P_GPBiCG_AR). After that, we will make verification of stability of the iterative method, and make it clear that the origin of instability of GPBi-CG method corresponds to the two auxiliary vectors of the algorithm. We will support that GPBiCG_AR and P_GPBiCG_AR methods yield safety convergence through numerical experiments.

2 GPBi-CG and GPBiCG_AR Methods

We consider iterative methods for solving a linear system of equations

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{1}$$

where $A \in R^{N\times N}$ is a given nonsymmetric matrix, and $\boldsymbol{x}$, $\boldsymbol{b}$ is a solution vector and right-hand side vector, respectively. When A is a large, sparse matrix which arises from realistic problems, the efficient solution of (1) is substantially very difficult. This difficulty has led to the development of a rich variety of generalized CG type methods having varying degrees of success (see, e.g., [5]).

The bi-conjugate gradient (BiCG) method based of the Lanczos algorithm is a crucial example of a generalized CG method. In many cases, the Lanczos algorithm gives some of the fastest solution times and stability of convergence among all generalized CG methods. The Lanczos algorithm, however, is known to break down in some cases. In practice, the occurrence of breakdown can cause failure to irregularly converge to the solution of (1). The fact that the Lanczos algorithm performs well in some cases but fail in others heightens the need for further insight and development of the Lanczos type iterative methods.

We note that the basic recurrence relations between Lanczos polynomials $R_n(\lambda)$ and $P_n(\lambda)$ hold as follows:

$$R_0(\lambda) = 1, \quad P_0(\lambda) = 1, \tag{2}$$

$$R_{n+1}(\lambda) = R_n(\lambda) - \alpha_n \lambda P_n(\lambda), \tag{3}$$

$$P_{n+1}(\lambda) = R_{n+1}(\lambda) + \beta_n P_n(\lambda), \quad n = 1, 2, \ldots. \tag{4}$$

Then we can introduce the three-term recurrence relations for Lanczos polynomial $R_n(\lambda)$ only by eliminating $P_n(\lambda)$ from (2)-(4) as follows:

$$R_0(\lambda) = 1, \ R_1(\lambda) = (1 - \alpha_0\lambda)R_0(\lambda), \tag{5}$$

$$R_{n+1}(\lambda) = (1 + \frac{\beta_{n-1}}{\alpha_{n-1}}\alpha_n - \alpha_n\lambda)R_n(\lambda) - \frac{\beta_{n-1}}{\alpha_{n-1}}\alpha_n R_{n-1}(\lambda), \ n = 1, 2, \ldots \tag{6}$$

Zhang [6] discovered that an often excellent convergence property can be gained by choosing for acceleration polynomial $H_n(\lambda)$ that is built up in the

three-term recurrence form as polynomial $R_n(\lambda)$ in (5) and (6) by adding suitable undetermined parameters ζ_n and η_n as follows:

$$H_0(\lambda) = 1,\ H_1(\lambda) = (1 - \zeta_0\lambda)H_0(\lambda), \tag{7}$$

$$H_{n+1}(\lambda) = (1 + \eta_n - \zeta_n\lambda)H_n(\lambda) - \eta_n H_{n-1}(\lambda),\ n = 1, 2, \ldots. \tag{8}$$

The polynomial $H_n(\lambda)$ satisfies $H_n(0) = 1$ and the relation as $H_{n+1}(0) - H_n(0) = 0$ for all n. Here we introduce an auxiliary polynomial $G_n(\lambda)$ as

$$G_n(\lambda) := \frac{H_n(\lambda) - H_{n+1}(\lambda)}{\zeta_n\lambda}. \tag{9}$$

By reconstruction of (6) using the acceleration polynomials $H_n(\lambda)$ and $G_n(\lambda)$, we have the following coupled two-term recursion of the form as

$$H_0(\lambda) = 1,\ G_0(\lambda) = \zeta_0, \tag{10}$$

$$H_n(\lambda) = H_{n-1}(\lambda) - \lambda G_{n-1}(\lambda), \tag{11}$$

$$G_n(\lambda) = \zeta_n H_n(\lambda) + \eta_n G_{n-1}(\lambda),\ n = 1, 2, \ldots. \tag{12}$$

Using these acceleration polynomials $H_n(\lambda)$ and $G_n(\lambda)$, Zhang's discover led to the generalized product-type methods based on Bi-CG method for solving the linear system with nonsymmetric coefficient matrix. He refered as GPBi-CG method [6]. However, the original Lanczos algorithm is also known to break down or nearly break down in some cases. In practice, the occurrence of a break down causes failure to converge to the solution of linear equations, and the increase of the iterations introduce numerical error into the approximate solution. Therefore, the convergence of the generalized product-type methods is affected. Comparatively little is known about the theoretical properties of the generalized product-type methods. The fact that the generalized product-type methods perform very well in some cases but fail in other cases, motivates the need for further insight into the construction of polynomials for the product-type residual $H_{n+1}(\lambda)R_{n+1}(\lambda)$. In a usual approach, acceleration parameters are decided from local minimization of the residual vector of 2-norm $||\boldsymbol{r}_{n+1}(:= H_{n+1}(\lambda)R_{n+1}(\lambda))||_2$, where $R_{n+1}(\lambda)$ denotes the residual polynomial of the Lanczos algorithm and $H_{n+1}(\lambda)$ denotes the acceleration polynomial for convergence. Instead, it embodies that a particular strategy for remedying the instability of convergence. That is, the algorithm of GPBiCG_AR method based on local minimization of an associate residual $\boldsymbol{a\_r}_n(:= H_{n+1}(\lambda)R_n(\lambda))$ is written as follows:

$$\boldsymbol{a\_r}_n = \boldsymbol{r}_n - \eta_n A\boldsymbol{z}_{n-1} - \zeta_n A\boldsymbol{r}_n. \tag{13}$$

Here $\boldsymbol{r}_n$ is the residual vector of the algorithm. Matrix-vector multiplications of $A\boldsymbol{u}_n$ and $A\boldsymbol{r}_{n+1}$ are directly computed according to definition of multiplication of matrix A and vector. On the other hand, $A\boldsymbol{p}_n$ and $A\boldsymbol{z}_n$ are computed using their recurrence. In the algorithm of GPBiCG_AR method, modification parts which differ from the original GPBi-CG method are indicated with underlines. The description of compute $A\boldsymbol{u}_n$ means that multiplication of $A\boldsymbol{u}_n$ is done according to its definition.

Algorithm 1. GPBiCG_AR method

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
choose $\boldsymbol{r}_0^*$ such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$,
set $\beta_{-1} = 0$, compute $A\boldsymbol{r}_0$,
for $n = 0, 1, \cdots$ until $||\boldsymbol{r}_{n+1}|| \leq \varepsilon \, ||\boldsymbol{r}_0||$ do :
begin

$$\boldsymbol{p}_n = \boldsymbol{r}_n + \beta_{n-1}(\boldsymbol{p}_{n-1} - \boldsymbol{u}_{n-1}),$$
$$A\boldsymbol{p}_n = A\boldsymbol{r}_n + \beta_{n-1}(A\boldsymbol{p}_{n-1} - A\boldsymbol{u}_{n-1}),$$
$$\alpha_n = \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)}{(\boldsymbol{r}_0^*, A\boldsymbol{p}_n)},$$
$$\boldsymbol{a}_n = \boldsymbol{r}_n, \ \boldsymbol{b}_n = A\boldsymbol{z}_{n-1}, \ \boldsymbol{c}_n = A\boldsymbol{r}_n,$$

$$\zeta_n = \frac{(\boldsymbol{b}_n, \boldsymbol{b}_n)(\boldsymbol{c}_n, \boldsymbol{a}_n) - (\boldsymbol{b}_n, \boldsymbol{a}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)}{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{b}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)},$$
$$\eta_n = \frac{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{a}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{a}_n)}{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{b}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)},$$
$$(\text{if } n = 0, \text{ then } \zeta_n = \frac{(\boldsymbol{c}_n, \boldsymbol{a}_n)}{(\boldsymbol{c}_n, \boldsymbol{c}_n)}, \ \eta_n = 0)$$
$$\boldsymbol{u}_n = \zeta_n A\boldsymbol{p}_n + \eta_n(\boldsymbol{t}_{n-1} - \boldsymbol{r}_n + \beta_{n-1}\boldsymbol{u}_{n-1}),$$
compute $A\boldsymbol{u}_n$,
$$\boldsymbol{t}_n = \boldsymbol{r}_n - \alpha_n A\boldsymbol{p}_n,$$
$$\boldsymbol{z}_n = \zeta_n \boldsymbol{r}_n + \eta_n \boldsymbol{z}_{n-1} - \alpha_n \boldsymbol{u}_n,$$
$$A\boldsymbol{z}_n = \zeta_n A\boldsymbol{r}_n + \eta_n A\boldsymbol{z}_{n-1} - \alpha_n A\boldsymbol{u}_n,$$
$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \alpha_n \boldsymbol{p}_n + \boldsymbol{z}_n,$$
$$\boldsymbol{r}_{n+1} = \boldsymbol{t}_n - A\boldsymbol{z}_n,$$
compute $A\boldsymbol{r}_{n+1}$,
$$\beta_n = \frac{\alpha_n}{\zeta_n} \cdot \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_{n+1})}{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)},$$

end

2.1 Preconditioned GPBi-CG and GPBiCG_AR Methods

This section discusses the preconditioned GPBiCG_AR (abbreviated as P_GPBiCG_AR) and GPBi-CG (abbreviated as P_GPBi-CG) methods. We choose a matrix K ($K \approx A = K_1 K_2$) as a preconditioner, the following preconditioned equation can be solved:

$$K^{-1}Ax = K^{-1}b \tag{14}$$

or

$$\tilde{A}\tilde{x} = \tilde{b}, \tag{15}$$

where $\tilde{A} = (K_1^{-1}AK_2^{-1})$, $\tilde{x} = (K_2x)$, $(\tilde{b} = (K_1^{-1}b)$ are defined. When GP-BiCG_AR method apply to the equation (2.15), the solution vector and residual vector can be defined as follows:

$$\tilde{\boldsymbol{x}}_n := K_2\boldsymbol{x}_n,\ \tilde{\boldsymbol{r}}_n := K_1^{-1}\boldsymbol{r}_n. \tag{16}$$

The auxiliary vectors of GPBiCG_AR method can be also defined as follows:

$$\begin{aligned}&\tilde{\boldsymbol{p}}_n := K_2\boldsymbol{p}_n,\ \tilde{\boldsymbol{t}}_n := K_1^{-1}\boldsymbol{t}_n,\\ &\tilde{\boldsymbol{u}}_n := K_2\boldsymbol{u}_n,\ \tilde{\boldsymbol{z}}_n := K_2\boldsymbol{z}_n,\ \tilde{\boldsymbol{r}}_0^* := K_1^H\boldsymbol{r}_0^*,\end{aligned} \tag{17}$$

where K^H is Hermitian matrix. Then, P_GPBiCG_AR method is as follows:

Algorithm 2. Preconditioned GPBiCG_AR method

$\boldsymbol{x}_0$ is an initial guess, $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$,
Choose $\boldsymbol{r}_0^*$ such that $(\boldsymbol{r}_0^*, \boldsymbol{r}_0) \neq 0$,
set $\beta_{-1} = 0$,
for $n = 0, 1, \cdots$ until $||\boldsymbol{r}_n|| \leq \varepsilon\ ||\boldsymbol{r}_0||$ do :
begin

$$\boldsymbol{p}_n = K^{-1}\boldsymbol{r}_n + \beta_{n-1}(\boldsymbol{p}_{n-1} - \boldsymbol{u}_{n-1}),$$
$$\underline{A\boldsymbol{p}_n = AK^{-1}\boldsymbol{r}_n + \beta_{n-1}(A\boldsymbol{p}_{n-1} - A\boldsymbol{u}_{n-1}),}$$
$$\alpha_n = \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)}{(\boldsymbol{r}_0^*, A\boldsymbol{p}_n)},$$
$$K^{-1}\boldsymbol{t}_n = K^{-1}\boldsymbol{r}_n - \alpha_n K^{-1}A\boldsymbol{p}_n,$$
$$\underline{\boldsymbol{a}_n = \boldsymbol{r}_n,\quad \boldsymbol{b}_n = A\boldsymbol{z}_{n-1},\quad \boldsymbol{c}_n = AK^{-1}\boldsymbol{r}_n,}$$

$$\zeta_n = \frac{(\boldsymbol{b}_n, \boldsymbol{b}_n)(\boldsymbol{c}_n, \boldsymbol{a}_n) - (\boldsymbol{b}_n, \boldsymbol{a}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)}{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{b}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)},$$
$$\eta_n = \frac{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{a}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{a}_n)}{(\boldsymbol{c}_n, \boldsymbol{c}_n)(\boldsymbol{b}_n, \boldsymbol{b}_n) - (\boldsymbol{b}_n, \boldsymbol{c}_n)(\boldsymbol{c}_n, \boldsymbol{b}_n)},$$
$$(\text{if } n = 0, \text{ then } \zeta_n = \frac{(\boldsymbol{c}_n, \boldsymbol{a}_n)}{(\boldsymbol{c}_n, \boldsymbol{c}_n)},\ \eta_n = 0)$$
$$\boldsymbol{u}_n = \zeta_n K^{-1}A\boldsymbol{p}_n + \eta_n(K^{-1}\boldsymbol{t}_{n-1} - K^{-1}\boldsymbol{r}_n + \beta_{n-1}\boldsymbol{u}_{n-1}),$$
$$\underline{\text{compute } A\boldsymbol{u}_n},$$
$$\boldsymbol{t}_n = \boldsymbol{r}_n - \alpha_n A\boldsymbol{p}_n,$$
$$\boldsymbol{z}_n = \zeta_n K^{-1}\boldsymbol{r}_n + \eta_n\boldsymbol{z}_{n-1} - \alpha_n\boldsymbol{u}_n,$$
$$\underline{A\boldsymbol{z}_n = \zeta_n AK^{-1}\boldsymbol{r}_n + \eta_n A\boldsymbol{z}_{n-1} - \alpha_n A\boldsymbol{u}_n,}$$
$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + \alpha_n\boldsymbol{p}_n + \boldsymbol{z}_n,$$
$$\underline{\boldsymbol{r}_{n+1} = \boldsymbol{t}_n - A\boldsymbol{z}_n,}$$

$$\underline{\text{compute } A\boldsymbol{r}_{n+1}},$$

$$\beta_n = \frac{\alpha_n}{\zeta_n} \cdot \frac{(\boldsymbol{r}_0^*, \boldsymbol{r}_{n+1})}{(\boldsymbol{r}_0^*, \boldsymbol{r}_n)},$$

end

3 Numerical Experiments

3.1 Non Preconditioned GPBiCG_AR and GPBi-CG Methods

In this section numerical experiments of non preconditioned GPBiCG_AR and GPBi-CG methods will be presented. We are primarily concerned with the GPBi-CG by Zhang[6] and GPBiCG_AR (with associate residual vector) methods. All computations were done in double precision floating point arithmetics, and performed on Hitachi SR11000 with CPU of Power5, clock of 1.9GHz, main memory of 128GB, OS of AIX 5L 5.3. All codes were compiled with the "-64 -Oss -nolimit -noscope -noparallel" optimization option. The right-hand side $\boldsymbol{b}$ was imposed from the physical load conditions. The stopping criterion for successful convergence of the iterative methods was less than 10^{-7} of the relative residual 2-norm $||\boldsymbol{r}_{n+1}||_2/||\boldsymbol{r}_0||_2$. In all cases the iteration was started with the initial guess solutions $\boldsymbol{x}_0$ are equal to all zeroes. The maximum number of iterations was fixed as 10^4. The initial shadow residual $\boldsymbol{r}_0^*$ of GPBiCG_AR and GPBi-CG methods was equal to the initial residual $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$. We examined stability of convergence of GPBiCG_AR and GPBi-CG methods. As test matrices, 29 matrices in total were taken from Florida sparse matrix collection[2].

Table 1 shows the numerical results of GPBiCG_AR and GPBi-CG methods. "Itr." means number of iterations, "time" means computation time in seconds and "ratio" means the ratio of computation time of GPBiCG_AR method to that of GPBi-CG method. The symbol "∞" denotes non-convergence until iterations reach at the maximum iteration counts.

From Table 1, the following observations can be made.

1. As for matrices af23560 and lung2, GPBiCG_AR method only converge. On the contrary, GPBi-CG method does not converge until maximum iterations because of stagnation of the residual.
2. For 11 matrices shown in Table 1, GPBiCG_AR method converge faster than GPBi-CG method. The time-ratios of GPBiCG_AR method to GPBi-CG method vary from 0.67 to 0.96. As a result, we can see that GPBiCG_AR method is fairly more efficient than GPBi-CG method.
3. However, for saga005 matrix shown in Table 1, both GPBiCG_AR and GPBi-CG methods can not converge.
4. For 15 matrices except the matrices shown in Table 1, computation times of GPBiCG_AR method are nearly as same as those of GPBi-CG method.

In Fig.1 we demonstrate history of relative residual 2-norm of GPBiCG_AR (solid line) and GPBi-CG (dotted line) methods for two matrices (a)af23560 and (b)lung2.

Table 1. Iterations and computation time in seconds of GPBiCG_AR and GPBi-CG methods

No.	matrix	GPBiCG_AR			GPBi-CG	
		Itr.	time	ratio	Itr.	time
1	cage12	9	0.22	0.96	9	0.23
2	xenon1	482	4.59	0.94	507	4.87
3	xenon2	572	22.42	0.94	599	23.97
4	wang4	366	0.84	0.93	380	0.90
5	memplus	565	1.01	0.92	605	1.10
6	torso2	24	0.33	0.92	25	0.36
7	stomach	197	6.87	0.90	213	7.67
8	sme3db	4205	78.58	0.79	5336	99.31
9	fidap035	889	2.10	0.77	1120	2.71
10	sme3da	4000	27.98	0.73	5467	38.35
11	ibm_matrix_2	6051	53.06	0.67	8850	78.88
12	af23560	1915	7.85	-	∞	
13	lung2	4573	43.81	-	∞	
14	saga005	-			-	

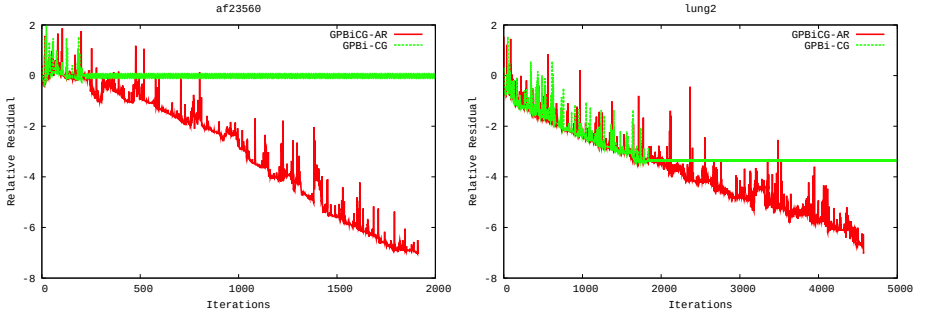
From Fig.1(a), it is clear that the relative residual of GPBi-CG method for matrix af23560 oscillates violently around the residual level of $10^{-0.3} \sim 10^{+0.3}$. Moreover, from Fig.1(b), we can observe that the relative residual of GPBi-CG method for matrix lung2 stagnates under the residual level of approximately $10^{-3.3}$. On the other hand, the relative residual of GPBiCG_AR method converges nicely for these matrices. As a result, we can verify the stability of convergence of GPBiCG_AR method compared with that of GPBi-CG method.

In Fig.2 we show history of relative residual 2-norm of GPBiCG_AR and GPBi-CG methods for two matrices (a)sme3da and (b)ibm_matrix_2. Moreover in Fig.3 we depict history of relative residual 2-norm of GPBiCG_AR and GPBi-CG methods for two matrices (a)sme3db and (b)fidap035. From these Figs., experimental results indicate that GPBiCG_AR method realizes efficient and safety convergence.

3.2 Preconditioned GPBiCG_AR and GPBi-CG Methods

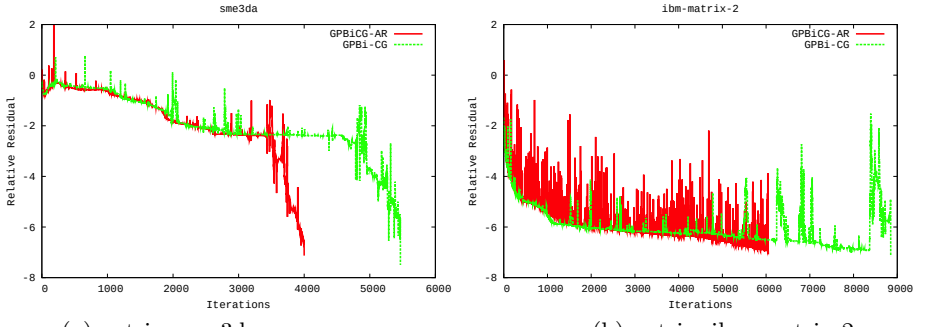
Basic computational conditions are the same with non preconditioned case in the preceding subsection. In this subsection, we use acceleration coefficient γ to raise approximation accuracy of P_GPBiCG_AR and P_GPBi-CG methods.

Table 2 shows the numerical results of P_GPBiCG_AR and P_GPBi-CG methods. The value of acceleration coefficient γ varies from 1.0 until 1.3 at the interval of 0.01. That is, we examined 31 cases in total for each matrix. For fairness, we show the average iterations and computation times in Table 2. Minimum iterations and computation times are also shown together in Table 2. "No. of conv." means number of times for successful convergence of P_GPBiCG_AR and P_GPBi-CG methods. "Minimum time" and "Minimum itr." mean computation times in seconds and number of iterations when P_GPBiCG_AR and



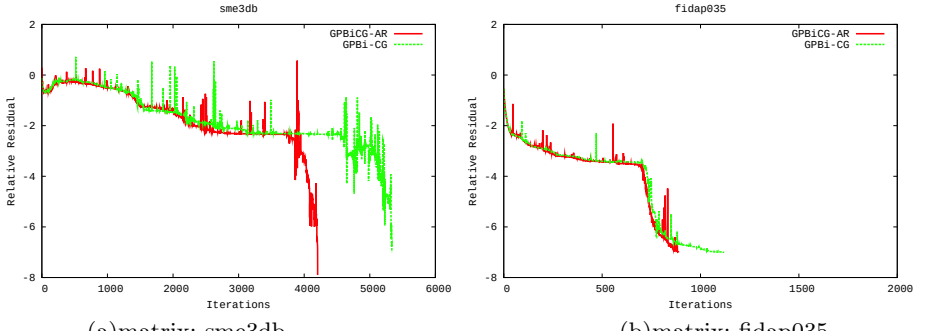
(a)matrix: af23560 (b)matrix: lung2

Fig. 1. History of relative residual 2-norm of GPBiCG_AR (solid line) and GPBi-CG (dotted line) methods for two matrices af23560 and lung2



(a)matrix: sme3da (b)matrix: ibm_matrix_2

Fig. 2. History of relative residual 2-norm of GPBiCG_AR (solid line) and GPBi-CG (dotted line) methods for two matrices sme3da and ibm_matrix_2



(a)matrix: sme3db (b)matrix: fidap035

Fig. 3. History of relative residual 2-norm of GPBiCG_AR (solid line) and GPBi-CG (dotted line) methods for two matrices sme3db and fidap035

Table 2. Iterations and computation time of P_GPBiCG_AR and P_GPBi-CG methods

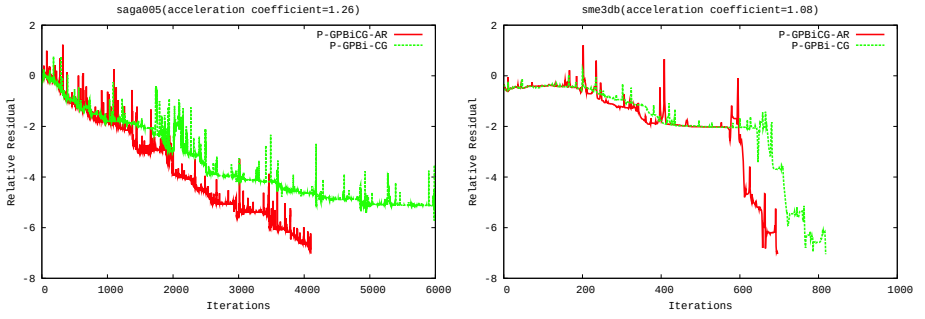
No.	matrix	P_GPBiCG_AR					P_GPBi-CG				
		No. of	Average		Minimum		No. of	Average		Minimum	
		conv.	itr.	time	itr.	time	conv.	itr.	time	itr.	time
1	cage12	31	3	0.92	3	0.92	31	3	0.92	3	0.91
2	lung2	31	4	0.23	3	0.22	31	4	0.23	3	0.21
3	stomach	31	6	1.72	5	1.61	31	5	1.67	5	1.61
4	ibm_max_2	31	25	1.37	5	1.61	31	24	1.35	14	1.12
5	wang4	31	44	0.33	33	0.26	31	43	0.33	32	0.25
6	memplus	31	112	0.57	91	0.47	31	111	0.57	90	0.47
7	sme3da	31	575	9.87	505	8.80	31	670	11.50	520	9.13
8	sme3db	31	800	43.71	696	38.28	31	981	53.12	801	43.87
9	xenon1	30	313	9.19	125	3.96	30	321	9.72	126	4.08
10	xenon2	30	442	54.00	141	17.71	29	329	40.60	150	19.31
11	ex19	20	97	0.64	53	0.40	20	111	0.71	52	0.39
12	saga005	20	4491	150.57	4101	138.29	11	5875	197.96	4428	149.25
13	olafu	7	7533	119.63	5583	90.28	1	7020	114.44	7020	114.44

P_GPBi-CG methods converge in the least time. 29 test matrices are taken from Florida sparse matrix collection [2] which are the same with the preceding subsection.

From Table 2, the following observations can be made.

1. For the eight matrices from the upper block, both P_GPBiCG_AR and P_GPBi-CG methods can converge well with 31 values of acceleration coefficient γ. These two methods converge nearly same time and iterations. Then, for the five matrices from the lower block, these methods converge sometimes. However, the numbers of convergence of P_GPBiCG_AR method is more than that of P_GPBi-CG method.
2. Moreover, P_GPBiCG_AR method converges 7 times for matrix olafu. The average iterations and computation times of P_GPBiCG_AR method are larger than that of P_GPBi-CG method. Because P_GPBiCG_AR method take too long time and much number of iterations to converge at acceleration coefficient γ value of 1.08. However, the minimum iterations and computation times of P_GPBiCG_AR method are less than that of P_GPBi-CG method. On the other hand, P_GPBi-CG has only one time of successful convergence.
3. For 16 matices except the matrices shown in Table 2, the computation times of preconditioned P_GPBiCG_AR method are nearly as same as those of P_GPBi-CG method.

In Fig.4 we demonstrate history of relative residual 2-norm of P_GPBiCG_AR and P_GPBi-CG methods for (a) saga005 and (b) sme3db matrices. At Fig.4 (a), although P_GPBiCG_AR can converge successfully when acceleration coefficient γ is 1.26, P_GPBi-CG cannnot converge at the same acceleration coefficient γ. Then, we can see both methods can converge when acceleration coefficient γ is 1.08 in Fig.4 (b). At first, these two methods compete with each other until



(a)matrix: saga005 (b)matrix: sme3db

Fig. 4. History of relative residual 2-norm of P_GPBiCG_AR (solid line) and P_GPBi-CG (dotted line) methods for two matrices saga005 and sme3db

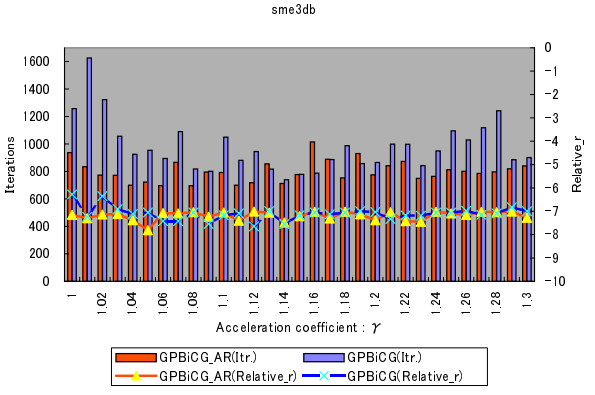
residual level 10^{-2}. Finally, P_GPBiCG_AR can converge with less number of iterations than the other.

Fig.5 illustrates the number of iterations and the relative residuals with the various acceleration coefficient γ for two matrices (a)sme3db and (b)ex19. In these figures, the vertical axis on right-hand side means number of iterations and the vertical axis on left-hand side also means relative residual. The bars in red represent number of iterations for P_GPBiCG_AR method. Similarly the bars in blue represent number of iterations for P_GPBi-CG method. The solid line in red and dotted line in blue are relative residual of P_GPBiCG_AR and P_GPBi-CG methods, respectively.

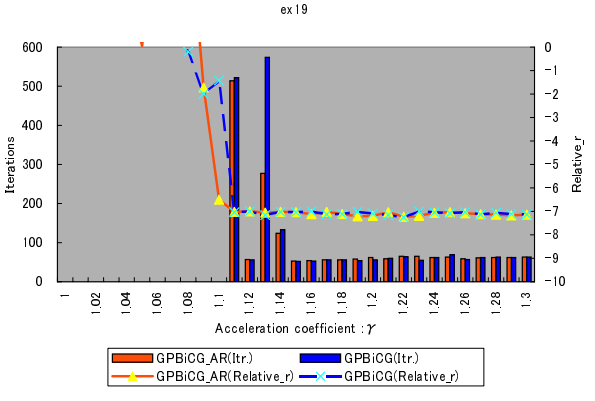
In Fig.5(a), both P_GPBiCG_AR and P_GPBi-CG methods can converge well with a variety of acceleration coefficient γ. Furthermore, P_GPBiCG_AR method can converge with less number of iterations than that of P_GPBi-CG method.

In Fig.5(b), both P_GPBiCG_AR and P_GPBi-CG methods diverge when acceleration coefficient γ is smaller than 1.12. Although P_GPBiCG_AR method cannot converge at acceleration coefficient γ of 1.11, its relative residual reached closely at 10^{-7} of the stopping criterion. When acceleration coefficients γ's become both 1.12 and 1.14, the bars are very high. On the contrary, when acceleration coefficient γ becomes greater than 1.14, the bars are very low and flat. These facts mean that P_GPBiCG_AR and P_GPBi-CG methods converge very fast and stably. Hence, acceleration coefficient γ's which become larger than 1.14 are preferable for solving efficiently the matrix ex19.

Fig.6 illustrates the number of iterations and the relative residuals with the various acceleration coefficient γ for two matrices (a)saga005 and (b)olafu. In these figures, the vertical axis on right-hand side means number of iterations and the vertical axis on left-hand side means relative residual. The bars in red represent number of iterations for P_GPBiCG_AR method. The bars in blue represent number of iterations for P_GPBi-CG method. The solid line in red and dotted line in blue are relative residual of P_GPBiCG_AR and P_GPBi-CG methods, respectively.



(a)matrix: sme3db

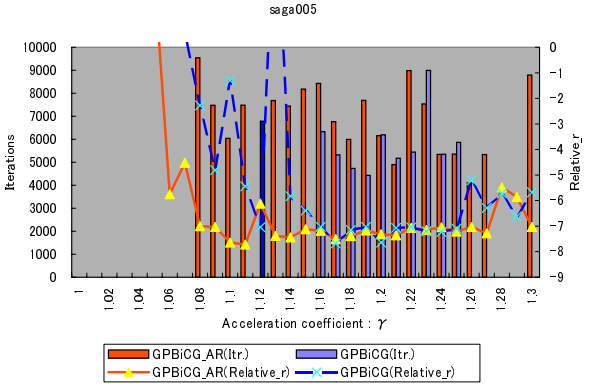


(b)matrix: ex19

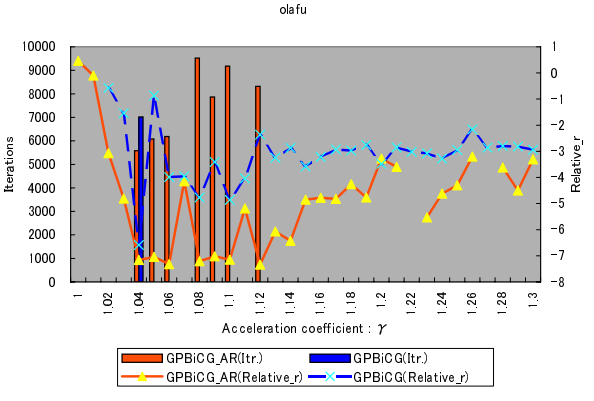
Fig. 5. Number of iterations and relative residuals with various acceleration coefficient γ for two matrices sme3db and ex19

In Fig.6(a), some bars are very high and some bars are low. This means that the convergences of P_GPBiCG_AR and P_GPBi-CG methods are not stable. However, the number of bars of P_GPBiCG_AR method is more than that of P_GPBi-CG method. In addition, P_GPBiCG_AR method can converge in wide range of acceleration coefficient γ. Hence, we can conclude that new P_GPBiCG_AR method has safety convergence compared with P_GPBi-CG method.

We remark Fig.6(b) as follows. In Fig.6(b), the solide line in red which plots for relative residual of P_GPBiCG_AR method disappear at acceleration coefficient γ's of 1.22 and 1.27. The dotted line in blue which plots for relative residual of P_GPBi-CG method also disappear at acceleration coefficient γ of 1.01. These cases happen because of the occurence of breakdown. Although the breakdown



(a)matrix: saga005



(b)matrix: olafu

Fig. 6. Number of iterations and relative residuals with various acceleration coefficient γ for two matrices saga005 and olafu

occurs during iteration process of P_GPBiCG_AR, this method converges seven times and P_GPBi-CG method converges only one time. On the other hand, the dotted line in blue which plots for the relative residual of P_GPBi-CG method oscillates between residual level of 10^{-2} and 10^{-4} when acceleration coefficient γ is larger than 1.12. Actually P_GPBi-CG fails to converge when the maximum number of iterations becomes larger. On the other hand, the solid line which plots for the relative residual of P_GPBiCG_AR method reaches at relative level of 10^{-6} at acceleration coefficient γ of 1.24. Hence, P_GPBiCG_AR method can be expected that it may converge if we increase the maximum number of iterations. From these remarks, Fig.6(b) proves that P_GPBiCG_AR method is preferable than P_GPBi-CG method from the viewpoint of safety convergence.

Finally, from Figs. 4-6, we can conclude that P_GPBiCG_AR method is superior to P_GPBi-CG method.

4 Conclusions and Future Work

In this paper, GPBiCG_AR and P_GPBiCG_AR are proposed for the purpose of removal of unstability of GPBi-CG. From numerical experiments, we can conclude that GPBiCG_AR method works very well compared with the original GPBi-CG method from the view of stability of convergence and computation times. P_GPBiCG_AR method is also superior to P_GPBi-CG method. Moreover, experimental evidences indicate that GPBiCG_AR and P_GPBiCG_AR yield safety convergence.

As you understand well from the above the conclusions, P_GPBiCG_AR method can converge in a richness of wide range of acceleration coefficient γ compared with P_GPBi-CG method. Therefore, we should find out to decide optimum acceleration coefficient γ as future work.

References

[1] Saad, Y.: Iterative Methods for Sparse Linear Systems. SIAM, Philadelphia (2003)
[2] Tim Davis' sparse matrix collection of Florida University, `http://www.cise.ufl.edu/research/sparse/matrices/`
[3] Fujino, S., Fujiwara, M., Yoshida, M.: BiCGSafe method based on minimization of associate residual, JSCES 2005 (2005), `http://www.jstage.jst.go.jp/article/jsces/2005/0/20050028_pdf/-char/ja/` (in Japanese)
[4] van der Vorst, H.A.: Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. SIAM J. Sci. Stat. Comput. 13, 631–644 (1992)
[5] van der Vorst, H.A.: Iterative Krylov preconditionings for large linear systems. Cambridge University Press, Cambridge (2003)
[6] Zhang, S.-L.: GPBi-CG: Generalized product-type methods based on Bi-CG for solving nonsymmetric linear systems. SIAM J. Sci. Comput. 18, 537–551 (1997)

Evaluation of a Java Computer Algebra System

Heinz Kredel

IT-Center, University of Mannheim, 68131 Mannheim, Germany
kredel@rz.uni-mannheim.de

Abstract. This paper evaluates the suitability of Java as an implementation language for the foundations of a computer algebra library. The design of basic arithmetic and multivariate polynomial interfaces and classes have been presented in [1]. The library is type-safe due to its design with Java's generic type parameters and thread-safe using Java's concurrent programming facilities. We evaluate some key points of our library and differences to other computer algebra systems.

1 Introduction

We have presented an object oriented design of a Java Computer Algebra System (called JAS in the following) as type safe and thread safe approach to computer algebra in [1,2,3]. JAS provides a well designed library for algebraic computations implemented with the aid of Java's generic types. The library can be used as any other Java software package or it can be used interactively or interpreted through an jython (Java Python) front end. The focus of JAS is at the moment on commutative and solvable polynomials, Groebner bases and applications. By the use of Java as implementation language, JAS is 64-bit and multi-core cpu ready. JAS has been developed since 2000 (see the weblog in [3]).

This work is interesting for computer science and mathematics, since it explores the Java [4] type system for expressiveness and eventual short comings. Moreover it employs many Java packages, and stresses their design and performance in the context of computer algebra, in competition with systems implemented in other programming languages.

JAS contains interfaces and classes for basic arithmetic of arbitrary precision integers and rational numbers and multivariate polynomials with such coefficients. Additional packages in JAS are:

- The package `edu.jas.ufd` contains classes for unique factorization domains. Like the interface `GreatestCommonDivisor`, an abstract class providing commonly useful methods and classes with implementations for polynomial remainder sequences and modular algorithms.
- The package `edu.jas.ring` contains classes for polynomial and solvable polynomial reduction, Groebner bases and ideal arithmetic as well as thread parallel and distributed versions of Buchbergers algorithm like `ReductionSeq`, `GroebnerBaseParallel` and `GroebnerBaseDistributed`.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 121–138, 2008.
© Springer-Verlag Berlin Heidelberg 2008

- The package `edu.jas.module` contains classes for module Groebner bases, syzygies for polynomials and solvable polynomials like `ModGroebnerBase` or `SolvableSyzygy`.
- Finally, the package `edu.jas.application` contains applications of Groebner bases, such as ideal intersections and ideal quotients in the classes `Ideal` or `SolvableIdeal`.

The emphasis of this paper is the evaluation of the JAS library design with respect to the points: interfaces as types, generics and inheritance, dependent types, method semantics, recursive types, design patterns, code reuse, performance, applications, parallelization, libraries, and the Java environment.

1.1 Related Work

In this section we briefly give some pointers to related work, for details see [1,2]. For an overview on other computer algebra systems see [5]. Typed computer algebra systems with own programming languages are described for example in [6,7,8]. Computer algebra systems implemented in other programming languages and libraries are: in C/C++ [9,10,11], in Modula-2 [12], in Oberon [13] and in FORTRAN [14]. A Python wrapper for computer algebra systems is presented in [15]. Java computer algebra implementations have been discussed in [16,17,18,19,20,21]. Newer approaches are discussed in [22,23,24].

The expression of mathematical requirements for generic algorithms in programming language constructs have been discussed in [25,26].

More related work, together with an evaluation of the design, is discussed in section 3. Due to limited space we have not discussed the related mathematical work on solvable polynomials, Groebner base and greatest common divisor algorithms, see [27,28] for some introduction. This paper contains an expanded, revised and corrected part of [2] and is a revised version of [29].

```
T[0] = 1
T[1] = x
T[2] = 2 x^2 - 1
T[3] = 4 x^3 - 3 x
T[4] = 8 x^4 - 8 x^2 + 1
T[5] = 16 x^5 - 20 x^3 + 5 x
T[6] = 32 x^6 - 48 x^4 + 18 x^2 - 1
T[7] = 64 x^7 - 112 x^5 + 56 x^3 - 7 x
T[8] = 128 x^8 - 256 x^6 + 160 x^4 - 32 x^2 + 1
T[9] = 256 x^9 - 576 x^7 + 432 x^5 - 120 x^3 + 9 x
```

Fig. 1. Chebychev polynomials

1.2 Outline

In the next section 2, we give some examples on using the JAS library and give an overview of the JAS type system for polynomials. Due to limited space we must assume that you are familiar with the Java programming language [30] and object oriented programming. Section 3 evaluates the presented design and compares JAS to other computer algebra systems. In particular it discusses interfaces as types, generics and inheritance, dependent types, method semantics, recursive types, design patterns, code reuse, performance, applications, parallelization, libraries, and the Java development environment. Finally section 4 draws some conclusions.

2 Introduction to JAS

In this section we show some examples for the usage of the JAS library, and then discuss the general layout of the polynomial types. Some parts of this section are similar to the JAS introduction in [3].

2.1 Using the JAS Library

To give a first idea about the usage of the library, we show the computation of Chebychev polynomials. They are defined by the recursion: $T[0] = 1$, $T[1] = x$,

```
 1.  int m = 10;
 2.  BigInteger fac = new BigInteger();
 3.  String[] var   = new String[]{ "x" };
 4.  GenPolynomialRing<BigInteger> ring
 5.                 = new GenPolynomialRing<BigInteger>(fac,1,var);
 6.  List<GenPolynomial<BigInteger>> T
 7.                 = new ArrayList<GenPolynomial<BigInteger>>(m);
 8.  GenPolynomial<BigInteger> t, one, x, x2;
 9.  one = ring.getONE();
10.  x   = ring.univariate(0); // polynomial in variable number 0
11.  x2  = ring.parse("2 x");
12.  T.add( one );    // T[0]
13.  T.add( x );      // T[1]
14.  for ( int n = 2; n < m; n++ ) {
15.      t = x2.multiply( T.get(n-1) ).subtract( T.get(n-2) );
16.      T.add( t );   // T[n]
17.  }
18.  for ( int n = 0; n < m; n++ ) {
19.      System.out.println("T["+n+"] = " + T.get(n) );
20.  }
```

Fig. 2. Computing Chebychev polynomials

$T[n] = 2x \times T[n-1] - T[n-2] \in \mathbb{Z}[x]$. The first ten Chebychev polynomials are shown in figure 1.

The polynomials have been computed with the Java program in figure 2. In lines 4 and 5 we construct a polynomial factory `ring` over the integers, in one variable `"x"`. This factory object itself needs at least a factory for the creation of coefficients and the number of variables. Additionally the term order and names for the variables can be specified. With this information the polynomial ring factory can be created by `new GenPolynomialRing <BigInteger> (fac,1,var)`, where `fac` is the coefficient factory, `1` is the number of variables, and `var` is an `String` array of names. In lines 8 to 11 the polynomials for the recursion base, `one` and `x` are created. Both are generated from the polynomial factory with method `ring.getONE()` and `ring.univariate(0)`, respectively. The polynomial $2x$ is, for example produced by the method `ring.parse("2 x")`. The string argument of method `parse()` can be the TEX-representation of the polynomial, except that no subscripts may appear. Note, `x2` could also be created from the coefficient factory by using `x.multiply( fac.fromInteger(2) )` or, directly by `x.multiply( new BigInteger(2) )`.

In lines 6 and 7 a list `T` is defined and created to store the computed polynomials. Then, in the for-loop, the polynomials $T[n]$ are computed using the definition, and stored in the list `T` for further use. In the last for-loop, the polynomials are printed, producing the output shown in figure 1. The string representation of the polynomial object can be created, as expected, by `toString()`.

To use other coefficient rings, one simply changes the generic type parameter, say, from `BigInteger` to `BigComplex` and adjusts the coefficient factory. The factory would then be created as `c = new BigComplex()`, followed by `new GenPolynomialRing<BigComplex>(c,1,var)`. This small example shows that this library can easily be used, just as any other Java package or library.

2.2 JAS Class Overview

The central interface is `RingElem` (see figure 3, left part) which extends `AbelianGroupElem` with the additive methods and `MonoidElem` with the multiplicative methods. Both extend `Element` with methods needed by all types. `RingElem` is itself extended by `GcdRingElem`, which includes greatest common divisor methods and `StarRingElem`, containing methods related to (complex) conjugation.

The interface `RingElem` defines a recursive type which specifies the functionality of the polynomial coefficients and is also implemented by the polynomials. So polynomials can be taken as coefficients for other polynomials, thus defining a recursive polynomial ring structure. We separate the creational aspects of ring elements into ring factories with sufficient context information. The minimal factory functionality is defined by the interface `RingFactory` (see figure 3, right part). Constructors for polynomial rings will then require factories for the coefficients so that the construction of polynomials poses no problem.

The `RingElem` interface (with type parameter `C`), defines the commonly used methods required for ring arithmetic, such as `C sum(C S)`, `C subtract(C S)`, `C abs()`, `C negate()`, `C multiply(C s)`, `C divide(C s)`, `C remainder(C s)`,

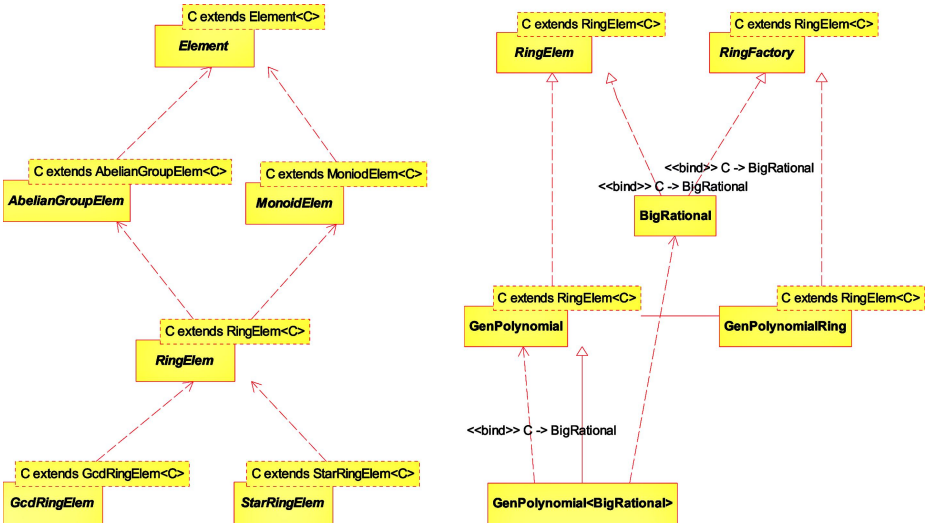


Fig. 3. Overview of some algebraic types and of generic polynomials

and `C inverse()`. In addition to the arithmetic methods there are testing methods such as `boolean isZERO()`, `isONE()`, `isUnit()` and `int signum()`. The first three test if the element is 0, 1 or a unit in the respective ring. The `signum()` method computes the sign of the element (in case of an ordered ring). The methods `equals(Object b)`, `int hashCode()` and `int compareTo(C b)` are required by Java's object machinery. The last method `C clone()` can be used to obtain a copy of the actual element.

The `RingFactory` interface defines the methods `C getZERO()`, `C getONE()`, which create 0 and 1 of the ring, respectively. The creation of the 1 is most difficult, since for a polynomial it implies the creation of the 1 from the coefficient ring, i.e. we need a factory for coefficients at this point. Further there are methods to embed a natural number into the ring and create the corresponding ring element, for example `C fromInteger(long a)`. Other important methods are `C random (int n)`, `C copy(C c)`, `C parse (String s)`, and `C parse (Reader r)`. The `random(int n)` method creates a random element of the respective ring. The two methods `parse(String s)` and `parse(Reader r)` create ring elements from some external representations. The methods `boolean isField()`, `isCommutative()` or `isAssociative()` query specific properties of the ring.

Generic polynomials are implemented in class `GenPolynomial`, which has a type parameter `C` that extends `RingElem<C>` for the coefficient type (see figure 3, right part). All operations on coefficients, that are required in polynomial arithmetic and manipulation are guaranteed to exist by the `RingElem` interface. The constructors of the polynomials always require a matching polynomial factory. The generic polynomial factory is implemented in the class `GenPolynomialRing`, again with type parameter `C extends RingElem<C>` (not

`RingFactory`). The polynomial factory implements the interface `RingFactory<C extends RingElem<C>>` so that it can also be used as coefficient factory. The constructors for `GenPolynomialRing` require at least the parameters for a coefficient factory and the number of variables of the polynomial ring.

The design of the other types and classes together with aspects of implementation are discussed in detail in [1].

3 Evaluation

In this section we discuss, without striving for completeness, some key points of our library and differences to other systems. Due to space restrictions, we assume some knowledge of [1] in the following, see also [3] and the related work in the introduction.

3.1 Interfaces as Types

In [31,32] the authors argue, and give counter examples, that a type system based only on (multiple) inheritance, is not sufficient to implement mathematical libraries, in particular, it is not sufficient to implement computer algebra libraries. As a solution they advocate interfaces, called signatures in their paper, as we find them now in Java. With the aid of interfaces it is possible to define a abstract type system separate of any implementation types defined by class hierarchies, as was already pointed out by [33]. This approach was partly anticipated in the Axiom system [6] with *categories* and *domains*. A *category* in Axiom is a kind of interface, but with the possibility to give implementations for certain methods, like an Java abstract class. A *domain* is similar to a Java class. In [34] the necessary flexibility for the type system was achieved by a decoupling of classes from *views* (interfaces in Java). In defining *views*, one could however, give arbitrary mappings for the *view* method names to the implementing class method names. Java allows only exact matching names, or one has to resort to some facade or adaptor pattern [35] to map names during runtime. The definition of the type hierarchy from `Element` to `RingElem` is perfectly suited to abstract from the common characteristics of coefficients and polynomials to make them exchangeable and inter-operable.

In section 2.4 (*solvable polynomials*) in [1] a problem appeared with the type erasure the compiler does for generic types to generate the raw implementation classes. Further investigation revealed, that this is not a problem of type erasure, but a feature of any generic object oriented programming language. A sub-class can not be allowed to implement a generic interface with a type parameter of the sub-class. Since this would require the compiler to check that no method of the super class, which is not overwritten in the sub-class, uses a super class constructor. This is not feasible to check for the compiler and impossible for separately compiled class libraries. This implies that our proposal to solve the type erasure problem, in [2], is wrong. In our case of the `GenPolynomial` super class we assured by using factory methods of the sub-class `GenSolvablePolynomial`

that any super class method returns objects of the sub-class. I.e. we changed the semantics of the super class methods to return sub-class objects but a compiler can not suss this.

3.2 Generics and Inheritance

The first version of the JAS library was implemented without generic types [36]. One obvious reason was, that generics were only introduced to the Java language in JDK 1.5. But it was well known from papers, such as [37], that generics are not necessarily required, when the programming language has, or allows the construction of a well-designed type hierarchy. In our previous implementation (up to 2005) we employed an interface `Coefficient`, which was implemented by coefficient classes and used in the `Polynomial` interface. `Polynomial` also extends `Coefficient` and so, polynomials could be used as coefficients of other polynomials. The `Coefficient` interface has now become the `RingElem` interface. However, with the old non-generic approach one looses some type safety, i.e. one could inadvertently multiply a polynomial with `BigInteger` coefficients by a polynomial with, say `BigRational` coefficients, and the compiler could not determine a problem. To prevent this, we had incorporated the name of the coefficient type in the name of the polynomial type, for example `RatPolynomial` or `IntPolynomial` in that release. A second reason for this first design was non existent coefficient factories, which could construct coefficients, say for the constant polynomial 1. Although the coefficient specific polynomials, for example `RatPolynomial`, have been extended from an abstract polynomial base class, for example `MapPolynomial`, it lead to much duplication of code. With the current generic type parameter approach we have removed all duplicate code for polynomial implementations. Moreover the type of the polynomial is clearly visible from the denotation, for example `GenPolynomial<BigInteger>`, of polynomial variables.

3.3 Dependent Types

The problem of dependent types is that we cannot distinguish polynomials in, say 3 variables from polynomials in, say 5 variables from their type. This carves a hole in the type safety of our library. I.e. the polynomial factories `GenPolynomialRing <BigInteger> (c, 3)` and `GenPolynomialRing <BigInteger> (c, 5)` create polynomials with the same type `GenPolynomial <BigInteger>`, but will most likely produce a run-time error, when, say, they are added together. Of course, the method `equals()` of the factory will correctly determine, that the rings are not equal, but the compiler is not able to infer this information and we are not able to denote hints.

This problem also occurs in the class `ModInteger` and `ModIntegerRing`. The type depends on the value of the modulus chosen. I.e. `ModIntegerRing(5)` and `ModIntegerRing(7)` are incompatible, but are denoted by the same type. Although the implementation of arithmetic methods of `ModInteger` will always choose the modulus of the first operand and therefore there will not be a run-time error, but this could lead to wrong results in applications.

The SmallTalk system [34] could use a elegant solution for this problem. Since types are first class objects, they can be manipulated as any other object in the language. For example one could define the following (in Java like syntax)

```
class Mod7 = ModIntegerRing(7);
Mod7 x = new Mod7(1);
```

Now `Mod7` is a type, which can be used to define and create new objects.

A minor problem of the same kind occurs with the term order defined in the polynomial factory, see [1]. It too, could be incompatible at run-time and this fact it is not expressed in the type. The actual implementation ignores this problem and arithmetic methods will produce a correct result polynomial, with a term order chosen from one of its input polynomials. However applications could eventually be confused by this behavior, for example Groebner base calculations.

Other computer algebra systems, for example [6], treat the polynomial dependent type case with some coercion facility. I.e. in most cases it is clear how to coerce a polynomial in 3 variables to a polynomial in 5 variables by adding variables with exponent zero or to coerce both to polynomials in 8 variables if variable names are disjoint.

A type correct solution to the dependent type problem in Java would be, to introduce an new type for every expected variable number, for example `Var1`, `Var2`, `Var3`, and to use this as additional type parameter for polynomials

```
GenPolynomialRing<BigRational,Var5>.
```

The types `Var*` could be implemented by interfaces or more suitably by extension of an abstract base class defining an abstract method `numberOfVariables` which could be used to query the number of variables at runtime. However, such a solution is impractical, since the number of variables of polynomials in applications is often determined at run-time and not during compile time.

How dependent types can correctly be handled in a programming language by the compiler, is discussed in [38].

3.4 Method Semantics

The interface `RingElem` defines several methods which cannot be implemented semantically correct in all classes. For example

- the method `signum()` makes no sense in unordered rings,
- the methods `divide()` and `remainder()` are not defined, if the divisor is zero or only of limited value for multivariate polynomials,
- the method `inverse()` may fail, if the element is not invertible, e.g. for `r = new ModIntegerRing(6)`, `a = new ModInteger(r,3)`, `a.inverse()` fails, since 3 is not invertible in $\mathbb{Z}_6$.

More examples for other systems can be found in [39]. We have adopted the policy to allow any meaningful reaction in these cases. For example the method `signum()` in `BigComplex` returns 0 if the number is equal to 0 and a non zero

value otherwise. The case of division by zero is in Java usually handled by throwing a run-time exception, and so do we. This is meaningful, since such a case is mostly an input error, which should have been handled by the calling programs.

For `inverse()`, the situation is slightly different. If the element is zero it is an error and a run-time exception can be thrown. But in the context of dependent types there are elements, which are not zero, but can nevertheless not be inverted. As in the above example 3 is not zero, but is not invertible in $\mathbb{Z}_6$. Also matrices can be non-zero but are eventually not invertible. In Axiom [6] such cases are handled by returning a special constant `"failed"`, with obvious problems arising for the type system. In Java we have the mechanism of checked exceptions. So for `inverse()`, it should be considered to add a `throws` clause in the definition, to make the user aware of some potential problem. At the moment we throw a run-time exception, but we will explore this variant in future refactorings.

In JAS there are testing methods to determine such cases. For example `isZERO()` or `isUnit()` to check if an element is invertible. For `isUnit()` the computing time can be as high as the computing time for `inverse()`, which doubles the computing time at this point and may not always be practical. In the factories there are methods to check further conditions. For example `isField()`, to test if the ring is a field and therefore if all non-zero elements are invertible.

There are proposals in [25,26] to formalize the semantic requirements for methods and types, so that the compiler can check them during compilation. Also Axiom [6] has some capabilities to specify and check method constraints. Some of this functionality can be provided in Java by polymorphic factories, see section 3.6.

3.5 Recursive Types

We have exercised some care in the definition of our interfaces to assure, that we can define recursive polynomials. First, the interface `RingElem` is defined as

```
RingElem<C extends RingElem<C>>.
```

So the type parameter `C` can (and must) itself be a subtype of `RingElem`. For polynomials we can define a polynomial with polynomials as coefficients

```
GenPolynomial< GenPolynomial<BigRational> >
```

In some applications presented in [1], for example the Groebner base algorithms, we make no use of this feature. However, there are many algebraic algorithms which are only meaningful in a recursive setting. For example greatest common divisors or factorization of multivariate polynomials. Although a study of this will be covered by a future publication, one observation is, that our type system will unfortunately lead to code duplication. The code for `baseGcd()` and `recursiveGcd()` is practically the same, but the methods have different parameter types. Further, by the type erasure for generic type parameters, they must also have different names.

We have successfully implemented a greatest common divisor package for multivariate polynomials, using these recursive type features. There is a clean interface `GreatestCommonDivisor` with only the most useful methods. These are `gcd()`, `lcm()`, `squarefreePart()`, `squarefreeFactors()`, `content()`, `primitivePart()`, and `resultant()`. The generic algorithms then work for all implemented (commutative) field coefficients.

The abstract class `GreatestCommonDivisorAbstract` implements the full set of methods, specified in the interface. Only two methods must be implemented for the different gcd algorithms. Details on the problems and solutions of this package will be covered by a future publication.

3.6 Factory Pattern

The usage of the factory pattern [35] to create objects with complex parametrization requirements is a standard technique in object oriented programming. Surprisingly, it has already been used in the SmallTalk implementation presented in [34]. Recently this approach was advocated again in [17,18]. But, otherwise we see this pattern seldom in computer algebra systems. The mainly used way to create polynomials or matrices is via constructors or by coercions from general algebraic expressions [6].

There is a nice application of the factory pattern in the `ufd` package. The factory `GCDFactory` can be used to select one from the many greatest common divisor implementations. This allows non-experts of computer algebra to choose the right algorithm for their problem.

```
GreatestCommonDivisor<BigInteger> engine
        = GCDFactory.<BigInteger>getImplementation( fac );
c = engine.gcd(a,b);
```

The static overloaded methods `getImplementation()` construct a suitable gcd implementation for the given type. The selection of the `getImplementation()` method takes place at compile time. I.e. depending on the desired actual type, different methods are selected. The coefficient factory parameter `fac` is used at run-time to check for example if the coefficients are from a field, to further optimize the selection. For the special cases of `BigInteger` and `ModInteger` coefficients, the modular algorithms can be employed.

This factory approach contrasts the approach taken in [25,26] to provide programming language constructs to specify the requirements for the selection of appropriate algorithm implementations. We can define requirements in Java with interface names and specifications together with the `extends` clause in the generic type parameter definition.

3.7 Code Reuse

With the help of generic programming we could drastically reduce the code of the earlier MAS [12] (and of the SAC-2 [14]) libraries. MAS has three major polynomial implementations, called distributive and recursive representation,

and univariate dense representation. For each representation there are three or more implementations. One 'class' for integer coefficients, one for rational number coefficients and one for modular integer coefficients. In JAS there is only one polynomial class, which works for all implemented coefficients.

In MAS, a *arbitrary domain polynomial* implementation exists. Here, the coefficients consist of a *domain descriptor* and a *domain value*. With the domain descriptor it was possible to select at run-time the corresponding implementation for the domain values and provide further context information for the called 'methods'. 13 coefficient domains have been implemented. Besides the lack of type safety, the introduction of a new coefficient implementation required the update of dispatching tables for all methods. The run-time selection of the implementation added a performance penalty (of about 20%), see [12] and the references there. With Java we have no performance loss for the generic coefficients and no need for recoding if new coefficients are introduced in the future.

A particular nice example for code reuse is the computation of e-Groebner bases, see for example [27]. They are d-Groebner bases but with e-reduction instead of d-reduction. This can be expressed by sub-classing with a different constructor.

```
public class EGroebnerBaseSeq<C extends RingElem<C>>
             extends DGroebnerBaseSeq<C> {
  public EGroebnerBaseSeq(EReductionSeq<C> red) {...} /*empty*/ }
```

3.8 Performance

The performance of Java is discussed in section 3.12. For our polynomial implementation (see [1]), performance is mainly determined by

1. the performance of coefficient arithmetic,
2. the sorted map implementation and
3. the exponent vector arithmetic.

Coefficient arithmetic of polynomials is based on the Java `BigInteger` class. `BigInteger` was initially implemented by a native C library, but since JDK 1.3 it is implemented in pure Java [40]. The new implementation has better performance than the C library. The sorted map implementation is from the Java collections package, which uses known efficient algorithms for this purpose, and it is comparable to other libraries, such as the C++ STL. However, we are not aware of general performance comparisons of the collection frameworks.

The exponent vector implementation is based on Java arrays of `long`s. It could be faster by using arrays of `int`s or `short`s as most computations seldom require polynomials of degrees larger than 2^{16}. This would reduce the memory footprint of a polynomial and so improve cache performance. If Java would allow elementary types as type parameters, it would be possible to make the `ExpVector` class generic, for example `ExpVector<long>` or `ExpVector<int>`.

Table 1. JAS polynomial multiplication benchmark

Computing times in seconds on AMD 1.6 GHz CPU.
Options are: coefficient type, term order: G = graded, L = lexicographic, big c = using the big coefficients, big e = using the big exponents, s = server JVM.

options, system	JDK 1.5	JDK 1.6
BigInteger, G	16.2	13.5
BigInteger, L	12.9	10.8
BigRational, L, s	9.9	9.0
BigInteger, L, s	9.2	**8.4**
BigInteger, L, big e, s	9.2	**8.4**
BigInteger, L, big c	66.0	59.8
BigInteger, L, big c, s	45.0	**45.8**

However, using objects like `Long` or `Integer` as exponents, would imply autoboxing and introduce too much performance penalties. To make the library useful for a wide range of applications we decided to stay with the implementation using `long`s.

There is a simple benchmark for comparing the multiplication of sparse polynomials in [41]. It times the computation of the product of two polynomials $q = p \times (p + 1)$, with integer coefficients, where $p = (1 + x + y + z)^{20}$, with bigger coefficients $p = (10000000001(1 + x + y + z))^{20}$, or with bigger exponents $p = (1 + x^{2147483647} + y^{2147483647} + z^{2147483647})^{20}$. The results for JAS are shown in table 1 and for other systems in table 2. The timings are partly from [2] and show that JAS is more than 3 times faster than the old MAS system but also 3.5 times slower than the Singular system. However Singular is not able to compute the example with bigger exponents. For this example JAS is 45% faster than Maple, and 65% faster than Mathematica. This shows that JAS (and the Java VM) matches the performance of general purpose systems. For further details see the discussion in [2].

3.9 Applications

As an application of the generic polynomials we have implemented some more advanced algorithms, such as polynomial reduction or Buchbergers algorithm to compute Groebner bases. The algorithms are also implemented for polynomial rings over principal ideal domains and Euclidean domains and for solvable polynomial rings (with left and two-sided variants) and modules over these rings. The performance of these implementations will be covered in a future publication.

3.10 Parallelization

JAS has been implemented with the goal of being *thread safe* from the beginning. This is mainly achieved by implementing all algebraic elements by immutable objects. This avoids any synchronization constructs for the methods at the cost of some more object creations. We have, however, not studied the impact of this on the performance.

Table 2. Polynomial multiplication, other systems

Computing times in seconds on AMD 1.6 GHz CPU and Intel 2.7 GHz CPU. Options are: coefficient type is rational number for MAS, integer for Singular and JAS, and it is unknown for Maple an Mathematica, big c = using the big coefficients, big e = using the big exponents, term order G = graded, L = lexicographic.

options, system	time	@2.7GHz
MAS 1.00a, L, GC = 3.9	**33.2**	
Singular 2-0-6, G	2.5	
Singular, L	**2.2**	
Singular, G, big c	12.95	
Singular, L, big exp	out of memory	
Maple 9.5	**15.2**	9.1
Maple 9.5, big e	19.8	11.8
Maple 9.5, big c	64.0	38.0
Mathematica 5.2	**22.8**	13.6
Mathematica 5.2, big e	30.9	18.4
Mathematica 5.2, big c	30.6	18.2
JAS, s	8.4	5.0
JAS, big e, s	8.6	5.1
JAS, big c, s	47.8	28.5

In the beginning, we had developed some utility classes for easier parallelization of the algorithms. In the mean time some classes are no more required, since equivalent substitutions exist in `java.util.concurrent` since JDK 1.5. We have replaced some of them in the latest refactorings of the library.

In the `ufd` package there is a nice parallel proxy class, which provides effective employment of the fastest algorithms at *run-time*. In the time of multi-core CPU computers, we compute the gcd with two (or more) different implementations in parallel. Then we return the result from the fastest algorithm, and cancel the other still running one. The gcd proxy can be generated from `GCDFactory.<C>getProxy()`. For example in a Groebner base computation with rational function coefficients, requiring many gcd computations, the fastest was 3610 times the sub-resultant algorithm and 2189 times a modular algorithm.

3.11 Libraries

The advantage of (scientific) libraries is apparent. Javas [4] success is greatly influenced by the availability of its comprehensive libraries of well tested and efficient algorithms. Also languages like Perl or PHP profit greatly from their comprehensive sets of available libraries. JAS is an attempt to provide a library for polynomial arithmetic and applications. There are other activities in this direction, however they are not all open source projects using the GPL license.

The goal of the jscl-meditor [22] project "is to provide a java symbolic computing library and a mathematical editor acting as a front-end to the former." jscl has a similar mathematical scope as JAS and we are looking for possibilities to cooperate in future work. The project JScience [24] aims to provide "the

most comprehensive Java library for the scientific community." The library has a broader perspective than JAS, in that it wants to support not only mathematics, but also physics, sociology, biology, astronomy, economics and others. There is the Orbital library [23], which provides algorithms from (mathematical) logic, polynomial arithmetic with Groebner bases and optimizations with genetic (*sic*) algorithms. The Apache software foundation distributes a numerical mathematical library as part of the `org.apache.commons` package [42]. It is a "library of lightweight, self-contained mathematics and statistics components addressing the most common problems not available in the Java programming language".

3.12 Java Environment

In [36] we have advocated the usage of standard libraries in favor of special implementations. In earlier computer algebra systems the creators had to implement many standard data structures by themselves. But now, we have the situation, that many of these data structures are available in form of well designed and tuned libraries, like the Java collection framework or the standard template library (STL) from C++. With this approach one can save effort to implement well known data structures again and again. Moreover, one profits from any improvements in the used libraries and improvements of the Java virtual machine (JVM). This has been exemplified by the performance improvements between JDK 1.5 and JDK 1.6 in section 3.8, table 1. Since Java is 64-bit ready we have been able to run Groebner base computations in 50 GB main memory.

In the discussion following my talk at ASCM2007 the performance of Java was a point of great controversy along the "Java is slow" myth. We admit that the first Java versions have been slow (compare [19]), but now we see evidence, that this is no more the case. Since the introduction of the just-in-time compilation (JIT) to the Java virtual machine the Java byte-code is compiled into native machine code on the fly [4]. So there is no principle difference in execution speed compared to C or C++. Already in 2001 (with JDK 1.3) the authors of [43] conclude "On Intel Pentium hardware, especially with Linux, the performance gap is small enough to be of little or no concern to programmers." Newer benchmarks (up to JDK 1.4) showing Java to be faster than C/C++ are discussed in [44,45,46]. The performance of generic programming implementations in C++, Java and C# compared to special hand-coded programs is discussed in [47]. There is a natural "abstraction penalty" in execution speed for all high-level languages, but on the other hand we see many software engineering benefits for more abstraction. As a starting point for further information also on performance issues see Google's directory [48].

4 Conclusions

JAS provides a consistent object oriented design and implementation of a library for algebraic computations in Java. For the first time we have produced a type safe library using generic type parameters in a contemporary programming

language. With Javas interfaces we are able to implement all algebraic types required for a computer algebra system. The generic programming of Java allows a type safe implementation of polynomial classes. It also helped to drastically reduce code size and facilitates code reuse. Type safety in JAS is only limited by the dependent type problem, which cannot be avoided in popular contemporary programming languages. With checked and unchecked exceptions we can model all necessary algebraic semantics of methods. The recursive `RingElem` and polynomial design allows the implementation of all important multivariate polynomial greatest common divisor algorithms. The usage of design patterns, for example the factory pattern for object creation, allows a clean concept for object oriented algebraic programming. Although object orientation looks strange to mathematicians, it is state of the art in modern programming languages. The performance of the library is comparable to general purpose computer algebra systems, but can not match the performance of special hand tuned systems. We have demonstrated that a large portion of algebraic algorithms can be implemented in a convenient Java library, for example non-commutative solvable polynomials or greatest common divisors. The parallel and distributed implementations of Groebner base algorithms draw heavily on the Java packages for concurrent programming and inter-networking. For the main working structures we built on the Java packages for multi-precision integers and the collection framework. The steady improvements of Java and its package implementations, leverage the performance and capabilities of JAS. A problem with Java's type erasure was identified as a general feature of generic object oriented programming languages and is not specific to Java.

In the future we will implement more of 'multiplicative ideal theory', i.e. multivariate polynomial factorization.

Acknowledgments

I thank Thomas Becker for discussions on the implementation of a polynomial template library and Raphael Jolly for the discussions on the generic type system suitable for a computer algebra system. With Samuel Kredel I had many discussions on the C++ type system and implementation choices for algebraic algorithms in C++. Thanks also to Markus Aleksy and Hans-Guenther Kruse for encouraging and supporting this work. Finally I thank the anonymous referees for suggestions to improve the paper.

References

1. Kredel, H.: On the Design of a Java Computer Algebra System. In: Proc. PPPJ 2006, University of Mannheim, pp. 143–152 (2006)
2. Kredel, H.: On a Java Computer Algebra System, its performance and applications. In: PPPJ 2006, Science of Computer Programming. Elsevier, Amsterdam (in print, 2008) (special issue)
3. Kredel, H.: The Java algebra system. Technical report (since 2000), `http://krum.rz.uni-mannheim.de/jas/`

4. Sun Microsystems, Inc.: The Java development kit. Technical report (accessed 2007) May (1994-2007), http://java.sun.com/
5. Grabmaier, J., Kaltofen, E., Weispfenning, V. (eds.): Computer Algebra Handbook. Springer, Heidelberg (2003)
6. Jenks, R., Sutor, R. (eds.): axiom The Scientific Computation System. Springer, Heidelberg (1992)
7. Bronstein, M.: Sigma^{it} - a strongly-typed embeddable computer algebra library. In: Proc. DISCO 1996, University of Karlsruhe, pp. 22–33 (1996)
8. Watt, S.: In: Aldor. in Computer Algebra Handbook, pp. 265–270. Springer, Heidelberg (2003)
9. Greuel, G.M., Pfister, G., Schönemann, H.: Singular - A Computer Algebra System for Polynomial Computations. In: Computer Algebra Handbook, pp. 445–450. Springer, Heidelberg (2003)
10. Buchmann, J., Pfahler, T.: LiDIA. In: Computer Algebra Handbook, pp. 403–408. Springer, Heidelberg (2003)
11. Noro, M., Takeshima, T.: Risa/Asir–a computer algebra system. In: Proc. ISSAC 1992, pp. 387–396. ACM Press, New York (1992)
12. Kredel, H., Pesch, M.: MAS: The Modula-2 Algebra System. In: Computer Algebra Handbook, pp. 421–428. Springer, Heidelberg (2003)
13. Gruntz, D., Weck, W.: A Generic Computer Algebra Library in Oberon. Manuscript available via Citeseer (1994)
14. Collins, G.E., Loos, R.G.: ALDES and SAC-2. ACM SIGSAM Bull. 12(2), 19 (1982)
15. Stein, W.: SAGE Mathematics Software (Version 2.7). The SAGE Group (2007) (accessed 2007, November), http://www.sagemath.org
16. Whelan, C., Duffy, A., Burnett, A., Dowling, T.: A Java API for polynomial arithmetic. In: Proc. PPPJ 2003, pp. 139–144. Computer Science Press, New York (2003)
17. Niculescu, V.: A design proposal for an object oriented algebraic library. Technical report, Studia Universitatis Babes-Bolyai (2003)
18. Niculescu, V.: OOLACA: an object oriented library for abstract and computational algebra. In: OOPSLA Companion, pp. 160–161. ACM, New York (2004)
19. Bernardin, L., Char, B., Kaltofen, E.: Symbolic computation in Java: an appraisement. In: Dooley, S. (ed.) Proc. ISSAC 1999, pp. 237–244. ACM Press, New York (1999)
20. Conrad, M.: The Java class package com.perisic.ring. Technical report (accessed 2006) (September 2002-2004), http://ring.perisic.com/
21. Becker, M.Y.: Symbolic Integration in Java. PhD thesis, Trinity College, University of Cambridge (2001)
22. Jolly, R.: jscl-meditor - java symbolic computing library and mathematical editor. Technical report (accessed 2007, September) (since 2003), http://jscl-meditor.sourceforge.net/
23. Platzer, A.: The Orbital library. Technical report, University of Karlsruhe(2005), http://www.functologic.com/
24. Dautelle, J.M.: JScience: Java tools and libraries for the advancement of science. Technical report (accessed 2007, May 2005-2007), http://www.jscience.org/
25. Musser, D., Schupp, S., Loos, R.: Requirement oriented programming - concepts, implications and algorithms. In: Jazayeri, M., Musser, D.R., Loos, R.G.K. (eds.) Dagstuhl Seminar 1998. LNCS, vol. 1766, pp. 12–24. Springer, Heidelberg (2000)

26. Schupp, S., Loos, R.: SuchThat - generic programming works. In: Jazayeri, M., Musser, D.R., Loos, R.G.K. (eds.) Dagstuhl Seminar 1998. LNCS, vol. 1766, pp. 133–145. Springer, Heidelberg (2000)
27. Becker, T., Weispfenning, V.: Gröbner Bases - A Computational Approach to Commutative Algebra. Graduate Texts in Mathematics. Springer, Heidelberg (1993)
28. Geddes, K.O., Czapor, S.R., Labahn, G.: Algorithms for Computer Algebra. Kluwer, Dordrecht (1993)
29. Kredel, H.: Evaluation of a Java Computer Algebra System. In: Proceedings ASCM 2007, National University of Singapore (2007)
30. Arnold, K., Gosling, J., Holmes, D.: The Java Programming Language, 4th edn. Addison-Wesley, Reading (2005)
31. Stansifer, R., Baumgartner, G.: A Proposal to Study Type Systems for Computer Algebra. Technical Report 90-07, Johannes Kepler University, Linz, Austria (1990)
32. Baumgartner, G., Russo, V.F.: Signatures: A language extension for improving type abstraction and subtype polymorphism in C++. Software - Practice and Experience 25(8), 863–889 (1995)
33. Liskov, B.: Data abstraction and hierarchy. In: OOPSLA 1987: Addendum to the proceedings on Object-oriented programming systems, languages and applications (Addendum), pp. 17–34. ACM, New York (1987)
34. Abdali, S.K., Cherry, G.W., Soiffer, N.: An object-oriented approach to algebra system design. In: Char, B.W. (ed.) Proc. SYMSAC 1986, pp. 24–30. ACM Press, New York (1986)
35. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns. Addison-Weseley (1995)
36. Kredel, H.: A systems perspective on A3L. In: Proc. A3L: Algorithmic Algebra and Logic 2005, University of Passau, pp. 141–146 (April 2005)
37. Meyer, B.: Genericity versus inheritance. In: OOPSLA, pp. 391–405 (1986)
38. Poll, E., Thomson, S.: The type system of Aldor. Technical report, Computing Science Institute Nijmegen (1999)
39. Fateman, R.J.: Advances and trends in the design and construction of algebraic manipulation systems. In: Proc. ISSAC 1990, pp. 60–67. ACM Press, New York (1990)
40. Sun Microsystems, Inc.: Improvements to program execution speed (accessed 2007, May 10 2004),
`http://java.sun.com/j2se/1.5.0/docs/guide/performance/speed.html`
41. Fateman, R.J.: Draft: Comparing the speed of programs for sparse polynomial multiplication (accessed 2007, May 5 2002),
`http://www.cs.berkeley.edu/~fateman/papers/fastmult.pdf`
42. Apache Software Foundation: Commons-Math: The Jakarta mathematics library. Technical report (accessed 2007, May 18 2003-2007),
`http://jakarta.apache.org/commons/`
43. Bull, J.M., Smith, L.A., Pottage, L., Freeman, R.: Benchmarking Java against C and Fortran for scientific applications. In: Proc. Joint ACM Java Grande and ISCOPE 2001 Conf. ACM Press, New York (2001)
44. Click, C.: Performance myths revisited. In: JavaOne (2005) (accessed January 2008),
`http://gceclub.sun.com.cn/java_one_online/2005/TS-3268/ts-3268.pdf`
45. Lewis, J., Neumann, U.: Performance of Java versus C++. Technical report (accessed 2008) (January 2004),
`http://www.idiom.com/~zilla/Computer/javaCbenchmark.html`

46. Lea, K.: The Java is faster than C++. Technical report (accessed 2008) (January 2005), http://www.kano.net/javabench/
47. Dragan, L., Watt, S.: Performance Analysis of Generics in Scientific Computing. In: Proceedings of Seventh International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, pp. 90–100. IEEE Computer Society, Los Alamitos (2005)
48. GoogleDirectoryContributors: Computers - programming - languages - comparison and review. Technical report (accessed 2008) (January 2008), http://directory.google.com/Top/Computers/Programming/Languages/Comparison_and_Review/

A New Property of Hamming Graphs and Mesh of *d*-ary Trees

Alain Bretto[1], Cerasela Jaulin[1], Luc Gillibert[1], and Bernard Laget[2]

[1] Université de Caen, GREYC CNRS UMR-6072,Campus II, Bd Marechal Juin BP 5186, 14032 Caen cedex, France
alain.bretto@info.unicaen.fr, cerasela.jaulin@info.unicaen.fr, luc.gillibert@info.unicaen.fr

[2] ENISE: 58, rue Jean Parot - 42023 Saint Etienne Cedex 02, France
laget@enise.fr

Abstract. In this article we characterize two well-known graphs used in many applications, particularly in network applications: Hamming graphs and meshes of d-ary trees $MT(d, 1)$. More precisely, we show that they are so-called $\mathbb{G}$-graphs. $\mathbb{G}$-graphs are a new type of graphs constructed from a group. They have nice algebraic proprieties and can be regular or semi-regular.

1 Introduction

Group theory is simultaneously a branch of computational algebra [6,7] and the study of symmetry. Graph theory [14] is one of the most important parts of computer science and combinatorics. Many areas of science use graph theory as a basic concept, for example : spatial geometric constraint solving based on k-connected graph decomposition [11]. There has been significant interaction between abstract group theory and the theory of graph automorphisms.[13] For example it has already been proved that every finite group is isomorphic to the automorphism group of a finite graph. Another link between groups and graphs is provided by Cayley Graphs, which have acquired additional applications, notably in the design of parallel computer architectures [8]. Much work has been done on these graphs [1]. Cayley graphs have nice properties. Their regularity and underlying algebraic structure make them good candidates for interconnecting nodes of a network [9].

Another family of graphs constructed from groups are $\mathbb{G}$-*graphs*. These graphs, introduced in [2], [3], [5], have highly-regular properties but they can be regular or semi-regular. Like Cayley graphs, they can be used in many areas of science and have applications in error codes theory [4]. A lot of classical graphs are $\mathbb{G}$-*graphs*: bipartite complete graphs, $2n \times n$ 4-grids connected thorus, the cuboctahedral graph, the square, some of the generalized Petersen's graphs, some of the grids on a 3D torus, Heawood's. A very important problem is the generation of both symmetric and semi-symmetric graph. This problem started in 1934 with Foster Census list. Because the algorithm to construct $\mathbb{G}$-graphs is

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 139–150, 2008.
© Springer-Verlag Berlin Heidelberg 2008

simple, it becomes easy to construct new symmetric and semi-symmetric cubic, quadratic, quantic, .. graphs. Moreover, thanks to $\mathbb{G}$-*graphs* we improved some upper bounds in the cage graphs problem.

The purpose of this paper is to show that two well-known infinite families of graphs, Hamming graphs and meshes of d-ary trees $MT(d, 1)$*, are* $\mathbb{G}$*-graphs.*

Hamming graphs have many applications: interconnection networks, SIMD architecture, and parallel computing, to name a few. The most famous Hamming graphs are the hypercubes. They offer good communication performance (routing, broadcasting, connectivity)[10]. Hypercubes used in interconnection networks are often studied, as in *the carvingwidth of hypercubes* [16].
Meshes of d-ary trees $MT(d, 1)$ are semi-regular graphs. Combining tree-like and grid-like graphs qualities, they are interesting in architectural networks for parallel computing. They are a satisfactory example of the use of $\mathbb{G}$-graphs to characterize graphs families which are not Cayley graphs. Consequently $\mathbb{G}$-graphs are interesting and can be used also in areas of science where Cayley graphs occur, and also in other areas.

Our work is structured as follows:

- Section 2 - We give the definitions we need;
- Section 3 - We introduce $\mathbb{G}$-graphs and their most important proprieties ;
- Section 4 - We define Hamming graphs as a generalization of hypercubes and as a Cayley graph, then we construct an isomorphism between the Hamming graphs and $\mathbb{G}$-graphs;
- Section 5 - Finally, we construct a group for mesh of d-ary trees $MT(d, 1)$ and we conclude that these graphs are $\mathbb{G}$-graphs.

2 Preliminaries

2.1 Graph Definitions

We define a graph, $\Gamma = (V; E; \epsilon)$ in the following way:

- $V(\Gamma)$ is the set of vertices.
- $E(\Gamma)$ is the set of edges.
- ϵ is a map from E to $P_2(V)$ (where $P_2(V)$ is the set of subsets of V having 1 or 2 elements).

In this article graphs are finite, (sets V and E have finite cardinality). For convenience, if $a \in E$, we denote $\epsilon(a) = [x; y]$ with the meaning that the extremities x, y of a may be equal (*loop*) or not. For $x, y \in V$, the set $M = \{a \in E, \epsilon(a) = [x; y]\}$ is called *multiedge* or *p-edge* if the cardinality of M is p. If $0 \leq p \leq 1$ without loop we call the graph simple and we note $\Gamma = (V; E)$. We denote an edge $([x; y]; a_i)$ with $([x; y]; a_i) \in E$. The *degree* of $x \in V$ is the number of edges incident with x: a loop at x will contribute 2 to the degree of x. We will denote the degree by $d(x) = \{a \in E, x \in \epsilon(a)\}$. Given a graph $\Gamma = (V; E; \epsilon)$, we denote the *neighborhood* of a vertex x by $\Gamma(x)$, i.e.

The set formed by all the vertices adjacent to x, and $W \subset V$, the *induced subgraph* on W is $\Gamma' = (W; E'; \epsilon')$ where $E' = \{a \in E, \epsilon(a) \subset W\}$ and $\epsilon' = \epsilon|_{E'}$; if $\forall x, y \in W \quad \{x, y\} \in E', \Gamma'$ is a *clique*. The *line graph* of the *simple* (i.e. without loops and multiedges) graph $\Gamma = (V; E; \epsilon)$ is $L(\Gamma) = (E; A; \eta)$ where $A = \{\alpha = \{a, b\}, a, b \in E, \epsilon(a) \cap \epsilon(b) \neq \phi\}$ and $\eta(\alpha) = \{a, b\}$.

Let $\Gamma_1 = (V_1; E_1; \epsilon_1)$ and $\Gamma_2 = (V_2; E_2; \epsilon_2)$ be two graphs, a *morphism* from $\Gamma_1 = (V_1; E_1; \epsilon_1)$ to $\Gamma_2 = (V_2; E_2; \epsilon_2)$ is a couple $(f, f^{\#})$ where $f : V_1 \longrightarrow V_2$ is a map and $f^{\#} : E_1 \longrightarrow E_2$ is a map such that: if $\epsilon_1(a) = [x; y]$ then $\epsilon_2(f^{\#}(a)) = [f(x); f(y)]$. A morphism $(f, f^{\#})$ is an *isomorphism* if and only if f is a bijection and $f^{\#}$ is a bijection ; for example $(id_V, id_V^{\#})$; in particular $Aut(\Gamma)$ is a group. A graph $\Gamma = (V; E; \epsilon)$ is a *k-graph* if we have a partition of V in k parts such that any part does not contain any edge other than loops ; we will write $\Gamma = (\sqcup_{i \in I} V_i; E; \epsilon)$; when $|I| = k$ is finite, Γ is *k-partite* ; when k is minimal such that Γ is a *k-graph*. sub-group of $Aut\Gamma$ consisting of automorphisms $(f, f^{\#})$ of Γ satisfying $\forall i \in I \quad f(V_i) = V_i$. A graph is k-semiregular if inside each partition of it's k-representation, all vertices have same degree. We can easily see that a regular graph is a particular case of a semiregular one.

2.2 Group Definitions

Recall that an *action* of a group G (with unity element e) on a set X is a map $G \times X \to X \quad (g, x) \mapsto g.x$ satisfying $e.x = x$ and $g.(g'.x) = (gg').x$, for every $x \in X, g, g' \in G$.The action is *transitive* if $\forall x, y \in X \quad \exists g \in G$ such that $g.x = y$. For $x \in X$ the *stabilizer* of x is $Stab_G x = \{g, g.x = x\}$. We consider, for any a $s \in S$ the (left) action of $< s >$ (subgroup generated by s) on G ; this gives a partition $G = \sqcup_{x \in T_s} < s > x$, where T_s is a right transversal of $< s >$. If $o(s) = | < s > |$ is the *order* of s, we have the cycles $(s)x = (x, sx, s^2x, \ldots s^{o(s)-1}x)$ of the permutation $g_s : x \mapsto s\,x \quad (x \in T_s)$.

Semi-direct-product of groups. Let H and Q be groups and let $\Theta : Q \to Aut(H)$ be a group homomorphism. The semi-direct product of H and Q by Θ (notation: $H \rtimes_\Theta Q$) is defined to be the group with underlying set $\{(h, q)/h \in H, q \in Q\}$ and the group operation:

$$(h, q) \cdot (h', q') = (h \cdot \Theta(q)h', q \cdot q')$$

Cayley Graphs. Let G be a group and $S \subseteq G$ a set of generators. We associate a digraph called *Cayley graph* whose the set of vertices is the set of elements of G and two vertices x, y are adjacent if and only if there exists $s \in S$ such that $y = sx$. If $S = S^{-1}$ the graph is undirected and if we choose for S a multi-set (repeating some generators) we get a *Cayley multi-graph*.

3 Introduction to $\mathbb{G}$-Graphs

Let (G, S) be a group with a set of generators S. For any $s \in S$, we consider the left action of the subgroup $H = \langle s \rangle$ on G. Thus, we have a partition $G =$

$\bigsqcup_{x \in T_s} \langle s \rangle x$, where T_s is a right transversal of $\langle s \rangle$. The cardinality of $\langle s \rangle$ is $o(s)$ where $o(s)$ is the order of the element s. Let us consider the cycles $(s)x = (x, sx, s^2x, \ldots, s^{o(s)-1}x)$ of the permutation g_s: $x \longmapsto sx$. Notice that $\langle s \rangle x$ is the support of the cycle $(s)x$. We now define a new graph denoted by $\Phi(G; S) = (V; E; \epsilon)$ as follows:

- The vertices of $\Phi(G; S)$ are the cycles of g_s, $s \in S$, i.e., $V = \sqcup_{s \in S} V_s$ with $V_s = \{(s)x, x \in T_s\}$.
- For all $(s)x, (t)y \in V$, $\{(s)x, (t)y\}$ is a p-edge if $card(\langle s \rangle x \cap \langle t \rangle y) = p,\ p \geq 1$.

Thus, $\Phi(G; S)$ is a k-partite graph and any vertex has a $o(s)$-loop. We denote $\widetilde{\Phi}(G; S)$ the graph $\Phi(G; S)$ without loops. By construction, one edge stands for one element of G. We can remark that one element of G labels several edges. Both graphs $\Phi(G; S)$ and $\widetilde{\Phi}(G; S)$ are called *graph from a group* or $\mathbb{G}$*-graphs* [5] and we say that the graph is *generated* by the group $(G; S)$.

3.1 Algorithmic Procedure

The following procedure constructs a graph from a group G and a subset S of G. A list of vertices and a list of edges represent the graph:

Procedure Group_To_Graph(G, S)
  Data:
    G a group
    $S = \{s_1, s_2, s_3, \ldots, s_k\}$, a subset of G
Cycles computing
  $L = \emptyset$
  for all a in S
    $l_2 = \emptyset$
    $g_s = \emptyset$
    for all x in G
      if x not in l_2 **then**
        $l_1 = \emptyset$
        for $k = 0$ to $k = Order(a) - 1$
          Add $(a^k) \times x$ to l_1
          Add $(a^k) \times x$ to l_2
        end for
        Add l_1 to g_s
      end if
    end for
    Add g_s to L
  end for
Graph computing
  for all s in L
    Add s to V
    for all s' in L
      for all x in s

```
            for all y in s'
                if x = y then
                    Add (s, s') to E
                end if
            end for
        end for
    end for
end for
Return (V, E)
```

3.2 Complexity and Example

It is easy to see that the complexity of our implementation is $O(n^2 \times k^2)$ where n is the order of the group G and k is the cardinal of the family S.

Let $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ be a a group generated by $S = \{(1,0,0);(0,1,0);(0,0,1)\}$.

Let us compute the graph $\widetilde{\Phi}(G;S)$.

The cycles of the permutation $g_{(1,0,0)}$ are:

$((1,0,0))(0,0,0) = ((0,0,0),(1,0,0)+(0,0,0)) = ((0,0,0),(1+0,0+0,0+0)) = ((0,0,0),(1,0,0))$

$((1,0,0))(0,1,0) = ((0,1,0),(1,0,0)+(0,1,0)) = ((0,1,0),(1+0,0+1,0+0)) = ((0,1,0),(1,1,0))$

$((1,0,0))(0,0,1) = ((0,0,1),(1,0,0)+(0,0,1)) = ((0,0,1),(1+0,0+0,0+1)) = ((0,0,1),(1,0,1))$

$((1,0,0))(0,1,1) = ((0,1,1),(1,0,0)+(0,1,1)) = ((0,1,1),(1+0,0+1,0+1)) = ((0,1,1),(1,1,1))$

The cycles of the permutation $g_{(0,1,0)}$ are:

$((0,1,0))(0,0,0) = ((0,0,0),(0,1,0)+(0,0,0)) = ((0,0,0),(0+0,1+0,0+0)) = ((0,0,0),(0,1,0))$

$((0,1,0))(1,0,0) = ((1,0,0),(0,1,0)+(1,0,0)) = ((1,0,0),(0+1,1+0,0+0)) = ((1,0,0),(1,1,0))$

$((0,1,0))(0,0,1) = ((0,0,1),(0,1,0)+(0,0,1)) = ((0,0,1),(0+0,1+0,0+1)) = ((0,0,1),(0,1,1))$

$((0,1,0))(1,0,1) = ((1,0,1),(0,1,0)+(1,0,1)) = ((1,0,1),(1+0,1+0,0+1)) = ((1,0,1),(1,1,1))$

The cycles of the permutation $g_{(0,0,1)}$ are:

$((0,0,1))(0,0,0) = ((0,0,0),(0,0,1)+(0,0,0)) = ((0,0,0),(0+0,0+0,1+0)) = ((0,0,0),(0,0,1))$

$((0,0,1))(1,0,0) = ((1,0,0),(0,0,1)+(1,0,0)) = ((1,0,0),(0+1,0+0,1+0)) = ((1,0,0),(1,0,1))$

$((0,0,1))(0,1,0) = ((0,1,0),(0,0,1)+(0,1,0)) = ((0,1,0),(0+0,0+1,1+0)) = ((0,1,0),(0,1,1))$

$((0,0,1))(1,1,0) = ((1,1,0),(0,0,1)+(1,1,0)) = ((1,1,0),(0+1,0+1,0+1)) = ((1,1,0),(1,1,1))$

The $\widetilde{\Phi}(\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}; S = \{(1,0,0);(0,1,0);(0,0,1)\})$ is isomorphic to the graph shown figure 1.

In the construction of the $\mathbb{G}$-graph we consider the left action of the subgroup $H = \langle s \rangle x$ on G ($s \in S$) and the $\mathbb{G}$-graph would have to be called *left* $\mathbb{G}$*-graph*. We can consider the right action, hence we have a *right* $\mathbb{G}$*-graph*. The following lemma justifies the designation of $\mathbb{G}$-graph.

Lemma 1. *Let $\Phi_r((G;S)) = (V_1;E_1;\epsilon_1)$ and $\Phi_l((G;S)) = (V_2;E_2;\epsilon_2)$ be the right and left $\mathbb{G}$-graphs of G. These two graphs are isomorphic.*

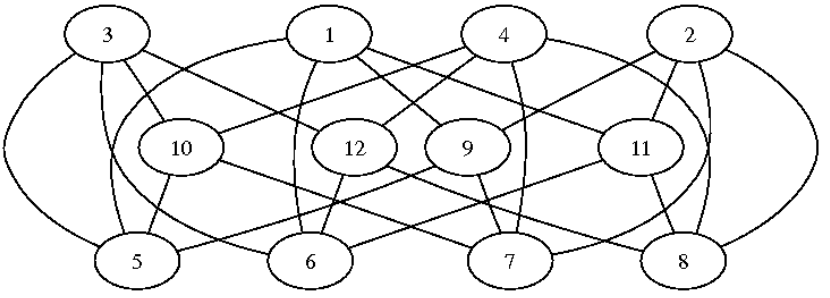


Fig. 1. G-graph of the group $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ generated with $S = \{(1,0,0);(0,1,0);(0,0,1)\}$

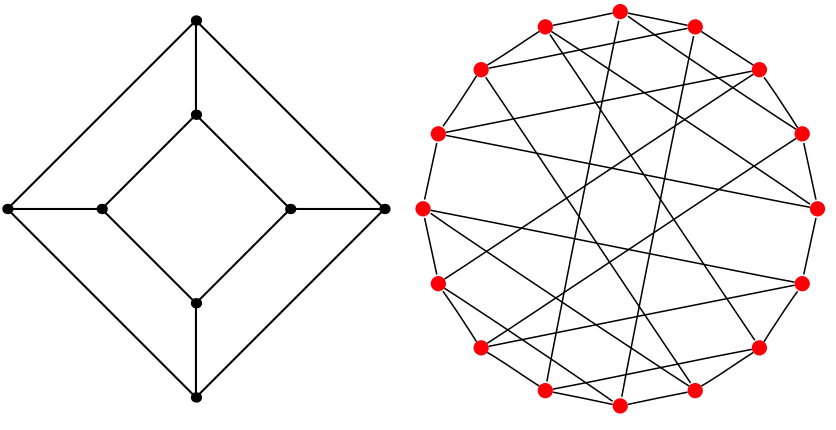
Proof. Main idea of the proof:
Let x be an element of G. There exists just one $y \in G$ such that $x = ys^i$. Settle $f : V_1 \longrightarrow V_2$ such that $f((s)x) = y(s)$, hence $f^{\#}$ is a bijection and $(f, f^{\#})$ is an isomorphism.

4 Hamming Graphs Are $\mathbb{G}$-Graphs

Definition 1. *The vertices set of the Hamming graph, denoted by $H(n,d)$ consists of all d-tuples $(x_1, x_2, \cdots, x_d)$, $0 \leq x_i \leq n$. Two vertices are an edge, if as d-tuples, they agree in all except one coordinate. In other way:*

- $V(H(n,d)) = (\mathbb{Z}/n\mathbb{Z})^d$.
- $[(x_1, x_2, \cdots, x_d), (y_1, y_2, \cdots, y_d)] \in E(H(n,d)) \Leftrightarrow \exists! 1 \leqslant i \leqslant d\ x_i \neq y_i$.

Example: $H(2,3)$ and $H(4,2)$ are Hamming graphs:



Some properties of Hamming's graphs:

- $H(n,d)$ is a regular graph with n^d vertices and $\frac{d(n-1)n^d}{2}$ edges;
- Each vertex has a degree equal to $d(n-1)$;

The following set $D(n,d) = \{(\underbrace{i, i \cdots, i, i}_{d}), 0 \leqslant i \leqslant n-1\}$ is a normal subgroup of $\mathbb{Z}/n\mathbb{Z}$. Consequently $B(n,d) = \frac{(\mathbb{Z}/n\mathbb{Z})^d}{D(n,d)}$ is a group.

Lemma 2. *$(B(n,d),+)$ is isomorphic to $(\mathbb{Z}/n\mathbb{Z})^{d-1}$.*

Proof. Main idea of the proof:
Let $\overline{x} = (\overline{x_1, x_2, \cdots x_{d-1}, x_d})$ be an element in $B(n,d)$. Settle $\Theta_1 : B(n,d) \to (\mathbb{Z}/n\mathbb{Z})^{d-1}$ such as $\Theta_1(\overline{x}) = (x_1 - x_d, x_2 - x_d, \cdots x_{d-1} - x_d, 0)$.
Θ_1 is an isomorphism of groups.

From now on we write each element in $B(n,d)$ as $x = (x_1, x_2, \cdots, x_{d-1}, x_d)$.

Let $\varphi : \mathbb{Z}/d\mathbb{Z} \to Aut(B(n,d))$ be the morphism $q \mapsto \varphi(q)$ defined by:
$\varphi(1)(x) = (x_2, x_3, \cdots, x_d, x_1)$
$\varphi(2)(x) = (x_3, x_4, \cdots, x_d, x_1, x_2)$
$\vdots$
$\varphi(i)(x) = (x_{i+1}, x_{i+2}, \cdots, x_d, x_1, x_{i-1}, x_i)$
$\vdots$
$\varphi(d)(x) = (x_1, x_2, \cdots, x_d) = \varphi(0) = x$

This morphism defines an action on $\mathbb{Z}/d\mathbb{Z} \times B(n,d) \mapsto B(n,d)$ where $\varphi(i)(x)$ is the inverse circular permutation of i-coordinates, of an element of $B(n,d)$. From now on we write $i \cdot x$ for $\varphi(i)(x)$.

We settle $G(n,d) = B(n,d) \rtimes_k (\mathbb{Z}/d\mathbb{Z})$ with $((x,k),(y,p)) \in (G(n,d))^2$ and we can see that the following operation " $\diamondsuit$ ".

$$(x,k)\diamondsuit(y,p) = (x + \varphi(k) \cdot y, k+p) = (x + k \cdot y, k+p) \tag{1}$$

give rise to a group: $(G(n,d), \diamondsuit)$ defined as:

$$G(n,d) = \frac{(\mathbb{Z}/n\mathbb{Z})^d}{D(n,d)} \rtimes_k (\mathbb{Z}/d\mathbb{Z})$$

Lemma 3. *Let $S = \{t_k | t_k \in G(n,d),\ t_k = ((k,0,0\cdots,0),1),\ 0 \leqslant k \leqslant n-1\}$.*

(a) For all $1 \leqslant i \leqslant d$ we have $t_k^i = ((\underbrace{k, \cdots k}_{i}, 0, \cdots, 0), i)$ and the order of the elements of S is d.

(b) The set S is a generating set for the group $(G(n,d), \diamondsuit)$.

Proof. Main idea of the proof:
To prove assertion (a) use induction.
To proving (b), we have :

$$(x,0)\diamondsuit t_0^b = ((x + 0 \cdot (0,0,\cdots,0)), 0+b) = (x,b) \tag{2}$$

and:

$$(x,0) = ((x_1,0,\cdots,0),0)\diamondsuit((0,x_2,0,\cdots,0),0)\diamondsuit\cdots\diamondsuit((0,0,\cdots,x_d),0) \tag{3}$$

and we also have:

$$((\underbrace{0\cdots 0}_{i-1}x_i 0\cdots 0),0) = (t_{n-x_i}^{i-1}\diamondsuit t_0^{d-(i-1)})\diamondsuit(t_{x_i}^{i}\diamondsuit t_0^{d-i}) \tag{4}$$

From (2)+(3)+(4) we can see that each element of the group $G(n,d)$ can be written as a product of elements of S. So S is a generating set.

Lemma 4. *Let $\tilde{\Phi}(G(n,d),S)$ be the right $\mathbb{G}$-graph. We have the following properties:*

(a) For any vertex $a \in V_{t_k}(\tilde{\Phi}(G(n,d),S))$, $k \in \{0,1,2,\cdots,n-1\}$:

$$a = ((x,0),((x,0)\diamondsuit t_k),((x,0)\diamondsuit t_k^2),\cdots,((x,0)\diamondsuit t_k^{d-1}))$$

(b) For two vertices of $\tilde{\Phi}(G(n,d),S)$, $a=(x,0)(t_k)$ and $a'=(y,0)(t_j)$ the two following assertions are equivalent:
(i) $[a;a'] \in E(\tilde{\Phi}(G(n,d),S))$

(ii) $y \in \{x, x+(t,0,\cdots,0), x+(t,t,\cdots,0),\cdots,x+(t,t,\cdots,t,0)\}$ with $t \in \{0,1,2,\cdots,n-1\}$.

Proof. Main idea of the proof:
We obtain:

$$((x+(\underbrace{k,\cdots,k}_{b},0,0,\cdots,0),b)\diamondsuit t_k^{d-b+i} = (x,0)\diamondsuit t_k^i$$

From above it is easy to see that (a) is true.
For (b) we show that $y = x+(\underbrace{t,\cdots,t}_{p},0,\cdots,0)$ with $t=k-j$.

Theorem 1. *The $\mathbb{G}$-graph $\tilde{\Phi}(G(n,d),S)$ is isomorphic to $Cay((\mathbb{Z}/n\mathbb{Z})^d,S^*)$ with:*

$$S^* = \{c_{ki}|c_{ki} = (\underbrace{k,\cdots,k}_{i},0,\cdots,0,k),\ 1\le k\le n-1,\ 0\le i\le d-1\}$$

Proof. Main idea of the proof:
Settle:

$$\Theta : \tilde{\Phi}(G(n,d),S) \to Cay((\mathbb{Z}/n\mathbb{Z})^d,S^*)$$

such that $\Theta(a) = (\Theta_1(x),k)$, Θ_1 being defined in the proof of Lemma 2.
The application Θ is an isomorphism.
It is easy to see that S^* is a generating set.
Let $a \in V_{t_k}(\tilde{\Phi}(G(n,d),S))$ with $a=(x,0)(t_k)$ and settle:

$$\Theta : \tilde{\Phi}(G(n,d),S) \to Cay((\mathbb{Z}/n\mathbb{Z})^d,S^*)$$

such that $\Theta(a) = (\Theta_1(x),k)$, Θ_1 being defined in the proof of Lemma 2.

The application Θ is a bijection because:

Let $a = (x,b)(t_k)$ and $a' = (y,b)(t_j)$ be two vertices of $\tilde{\Phi}(G(n,d),S)$. Let us consider that $\Theta(a) = \Theta(a')$. Hence $(\Theta_1(x),k) = (\Theta_1(y),j)$ so $k = j$ and $\Theta_1(x) = \Theta_1(y)$. Because Θ_1 is an isomorphism we have $x = y$. So Θ is injectif. Both graphs have the same number of vertices.

Consequently Θ is a bijection.

The application Θ is a morphism because:

Let $a = (x,b)(t_k)$ and $a' = (y,b)(t_j)$ be two vertices of $\tilde{\Phi}(G(n,d),S)$ and suppose that $[a;a'] \in E(\tilde{\Phi}(G(n,d),S))$.

We have $\Theta(a) = (\Theta_1(x),k)$ and $\Theta(a) = (\Theta_1(y),j)$ From the proof of Lemma 4 the two following assertions are equivalent:

(i) $[a;a'] \in E(\tilde{\Phi}(G(n,d),S))$

(ii) $\overline{y} \in \{x, x+(t,0,\cdots,0), x+(t,t,0,\cdots,0),\cdots, x+(t,t,\cdots,t,0)\}$ with $t = j-k$

From Lemma 2 we have:
$\Theta_1(y) = \Theta_1(x) + \Theta_1(\underbrace{j-k, j-k,\cdots,j-k}_{i},0,0,\cdots,0) =$
$\Theta_1(x) + (\underbrace{j-k,\cdots,j-k}_{i},0,\cdots,0)$, $i \in \{1,2,\cdots,d-1\}$. That leads to $\Theta(a') =$
$(\Theta_1(y),j) = (\Theta_1(x) + (\underbrace{j-k,\cdots,j-k}_{i},0,\cdots,0), k+j-k) = (\Theta_1(x),k) +$
$(\underbrace{j-k,\cdots,j-k}_{i},0,\cdots,0,j-k) = \Theta(a') + c_{j-k,i}$

Settle:
$x = (x_1,\cdots,x_d)$ and $y = (y_1,\cdots,y_d)$ two vertices on $Cay((\mathbb{Z}/n\mathbb{Z})^d, S^*)$. So $y = x + c_{ki}$ and this is equivalent to :
$(y_1,y_2,\cdots,y_d) = (x_1,x_2,\cdots,x_d) + (\underbrace{k,\cdots,k}_{i},0,\cdots,0,k)$. Hence $(\Theta_1(y),y_d) =$
$(\Theta_1(x) + (\Theta_1(\underbrace{k,\cdots,k}_{i},0,\cdots,0),k+x_d))$ same with
$(\Theta_1(y)) = (\Theta_1(x + (\underbrace{k,\cdots,k}_{i},0,\cdots,0)), x_d + k)$ and $y_d = x_d + k$.
Consequently we have $y = x + (\underbrace{k,\cdots,k}_{i-1},k,0,\cdots,0)$.
We conclude that: $\mathbb{G}$-graph $\tilde{\Phi}(G(n,d),S) \simeq Cay((\mathbb{Z}/n\mathbb{Z})^d, S^*)$.

Lemma 5. *We have the following isomorphism:*

$$Cay((\mathbb{Z}/n\mathbb{Z})^d, S^*) \simeq Cay((\mathbb{Z}/n\mathbb{Z})^d, S')$$

with the set $S' = \{e_k^i = (\underbrace{0,0,\cdots,0}_{i-1},k,0,\cdots,0)$ *and* $1 \leqslant i \leqslant d$, $0 \leqslant k \leqslant n-1\}$.

Proof. It is obvious that the following application $\Theta_2 : (\mathbb{Z}/n\mathbb{Z})^d \to (\mathbb{Z}/n\mathbb{Z})^d$ such as:

$$\Theta_2(c_{ki}) = e_{ki} \qquad k \in \{0, 1, \cdots, n-1\} \qquad i \in \{1, 2, \cdots, d\}$$

is an isomorphism from $((\mathbb{Z}/n\mathbb{Z})^d, S^*)$ to $((\mathbb{Z}/n\mathbb{Z})^d, S')$.

It follows that $Cay((\mathbb{Z}/n\mathbb{Z})^d, S^*) \simeq Cay((\mathbb{Z}/n\mathbb{Z})^d, S')$.

Theorem 2. *[15] The Cayley graph $Cay((\mathbb{Z}/n\mathbb{Z})^d, S')$ with the generating set S' defined above, is isomorphic to $H(n, d)$.*

Now we have the main result of this section:

Theorem 3. *The $\mathbb{G}$-graph $\tilde{\Phi}(G(n, d), S)$ is isomorphic to $H(n, d)$ - the Hamming graph.*

Proof. Immediately from lemma 5 and theorem 1.

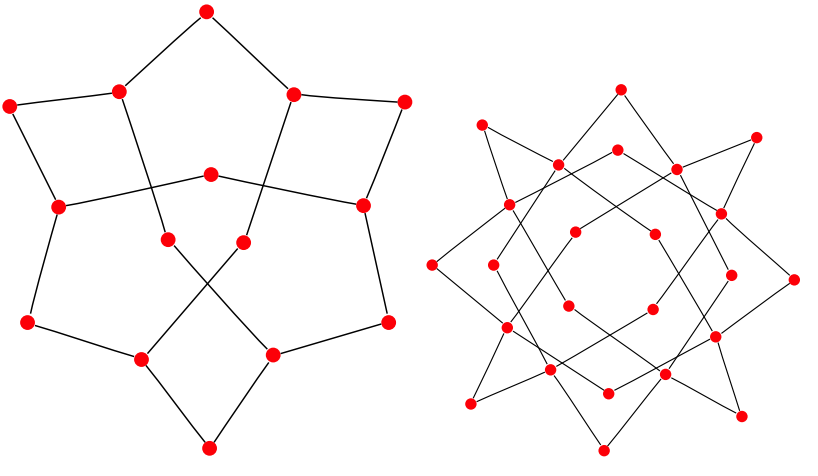
5 Mesh of *d*-ary Trees

The proofs in this section are straitforward from the proofs of *Hamming graphs are $\mathbb{G}$-graphs* section.

Let B an alphabet of d-letters and u a word on B. We denote $|u|$ the length of the word u. The *mesh of d-ary trees* $MT(d, h)$ is the graph with the vertex set $V = \{(u, v) || u| = h \text{ or } |v| = h\}$, and $[(u, v), (u', v')] \in E(MT(d, h))$ if and only if $|u| = h$,$u = u'$ and $v = v'\lambda$ or $|v| = h$,$v = v'$ and $u = u'\lambda$ with $\lambda \in B$. Key properties of grid d-Tree graphs are as follows:

- Number of vertices $N_v = d^h(d^h + 2\frac{d^h-1}{d-1})$;
- Number of edges $N_e = 2d(\frac{d^{h+1}-1}{d-1} - 1)$;
- Diameter $D = 4h$;
- The mesh of d-ary trees is not a Cayley graph;
- The mesh of d-ary trees is not vertex-transitive.

In this paper we are only interested in the mesh of d-ary trees on a 1-dimensional mesh $MT(d, 1)$. The following diagrams show $MT(3, 1)$ and $MT(4, 1)$:



We consider the following group:

$$G(n,2) = ((\mathbb{Z}/n\mathbb{Z}) \times (\mathbb{Z}/n\mathbb{Z})) \rtimes_{\varphi} (\mathbb{Z}/2\mathbb{Z})$$

with $\varphi : \mathbb{Z}/2\mathbb{Z} \to Aut((\mathbb{Z}/n\mathbb{Z}) \times (\mathbb{Z}/n\mathbb{Z}))$ the morphism defined by $\varphi(1)(x_1, x_2) = (x_2, x_1)$ and $\varphi(0)(x_1, x_2) = (x_1, x_2)$. We denote by " $\diamondsuit$ " the operation on this group.

Lemma 6. *Let $S = \{t_0 = ((0,0),1), t_1 = ((1,0),0)\}$.*

(a) The order of t_0 is 2, the order of t_1 is n.
(b) The set S is a generating set for the group $G(n,2)$.

Lemma 7. *Let $\tilde{\Phi}(G(n,2),S)$ be the right $\mathbb{G}$-graph. We have the following properties:*

(a) For any vertex $a \in V_{t_k}(\tilde{\Phi}(G(n,2),S))$, $k \in \{0,1\}$:

(i) For $a \in V_{t_0}$, $a = (((x_1,x_2),0);((x_1,x_2),1))$
(ii) For $a \in V_{t_1}$:

$$a = ((x_1,x_2),0)(t_1) = ((0,x_2),1);((1,x_2),1);\cdots;((n-1,x_2),1))$$

$$\text{or } a = ((x_1,x_2),0)(t_1) = ((x_1,0),0);((x_1,1),0);\cdots;((x_1,n-1),0))$$

(b) For two vertices of $\tilde{\Phi}(G(n,2),S)$:
$a = ((x_1,x_2),k)(t_0)$ and $a' = ((y_1,y_2),k)(t_1)$, the two following assertions are equivalent:
(i) $[a;a'] \in E(\tilde{\Phi}(G(n,2),S))$

(ii) $y_1 = x_1$ and $k = 0$ or $y_2 = x_2$ and $k = 1$.

Proof. We can easily see that $((x_1,x_2),0)\diamondsuit t_0 = ((x_1,x_2) + 0(0,0)), 0+1) = ((x_1,x_2),1)$ and this leads to (a.a). In the same way we show $((x_1,x_2),0)\diamondsuit t_0^i = ((x_1,x_2)+0(i,0)),0+1) = ((x_1+i,x_2),1)$, and also $((x_1,x_2),1)\diamondsuit t_0^i = ((x_1,x_2)+1(i,0)),1+1) = ((x_1,x_2+i),0)$. We can conclude that (a.ii) is true.

From the definition of a $\mathbb{G}$-graph we know that an edge exists only between one vertex a of $V(t_0)$ and one vertex a' of $V(t_1)$. From above, we can easily see that, for a', the first element or the second from the group (x_1,x_2) has all the values from 0 to $n-1$. For the vertex a the third element takes all values in $\mathbb{Z}/2\mathbb{Z}$. For an edge it is enough that (x_1,x_2) and (y_1,y_2) are equal. So from (a) we can easily see that this leads to (b.ii).

Theorem 4. *The $\mathbb{G}$-graph $\tilde{\Phi}(G(n,2),S)$ is isomorphic to $MT(n,1)$.*

Proof. Let us consider a and a' from lemma 7. Set $\Theta : \tilde{\Phi}(G(n,2),S) \to MT(n,1)$ as: $\Theta(a) = (x_1,x_2)$ and $\Theta(a') = (y_1,e)$ for $k = 0$ or $\Theta(a') = (e,y_2)$ for $k = 1$ where e is the empty word. It is obvious that Θ is an isomorphism.

References

1. Babai, L.: Handbook of combinatorics. In: Automorphism groups, isomorphism, reconstruction, ch. 27 (1994)
2. Bretto, A., Faisant, A.: Another way for associating a graph to a group. Math.Slovaca 55(1), 1–8 (2005)
3. Bretto, A., Gilibert, L., Laget, B.: Symmetric and Semisymmetric Graphs Construction Using G-graphs. In: Kauers, M. (ed.) International Symposium on Symbolic and Algebraic Computation (ISSAC 2005), Beijing, China, July 24-27, 2005, pp. 61–67. ACM press, New York (2005) ISBN:1-59593-095-7
4. Bretto, A., Gilibert, L.: G-graphs for the cage problem: a new upper bound. In: International Symposium on Symbolic and Algebraic Computation (ISSAC 2007), Waterloo, Ontario, Canada, July-29 August-1st 2007. ACM press, New York (2007) ISBN:978-1-59593-743-8
5. Bretto, A., Faisant, A., Gillibert, L.: G-graphs: A new representation of groups. Journal of Symbolic Computation 42(5), 549–560 (2007)
6. Cannon, J.J., Holt, D.F.: Automorphism group computation and isomorphism testing in finite groups. Journal of Symbolic Computation 35(3), 241–267 (2003)
7. Cannon, J.J., Holt, D.F.: Computing conjugacy class representatives in permutation groups. Journal of Algebra 300(1), 213–222 (2006) MR2228644
8. Cooperman, G., Finkelstein, L., Sarawagi, N.: Applications of Cayley Graphs. In: Sakata, S. (ed.) AAECC 1990. LNCS, vol. 508, pp. 367–378. Springer, Heidelberg (1991)
9. Cooperman, G., Finkelstein, L.: New Methods for Using Cayley Graphs in Interconnection Networks. Discrete Applied Mathematics 37/38, 95–118 (1992)
10. Fraigniaud, P., Konig, J.-C., Lazard, E.: Oriented hypercubes. Networks 39(2), 98–106 (2002)
11. Zhang, G.-F., Gao, X.-S.: Spatial geometric constraint solving based on k-connected graph decomposition. In: Proceedings of the ACM symposium on Applied computing Dijon, France 2006 SESSION: Geometric computing and reasoning (GCR), pp. 979–983. ACM Press, New York (2006)
12. The GAP Team, (06 May 2002), GAP - Reference Manual, Release 4.3, `http://www.gap-system.org`
13. Lauri, J., Scapellato, R.: Topics in Graphs Automorphisms and Reconstruction, London Mathematical Society Student Texts (2003)
14. Lauri, J.: Constructing graphs with several pseudosimilar vertices or edges. Discrete Mathematics 267(1-3), 197–211 (2003)
15. Rockmore, D., Hordijk, W., Kostelec, P., Stadler, P.F.: Fast Fourier Transform for Fitness Landscapes. Applied and Computational Harmonic Analysis 12(1), 57–76 (2002)
16. Sunil Chandran, L., Kavitha, T.: The carvingwidth of hypercubes. Discrete Mathematics 306(18), 2270–2274 (2006)

An Interpolation Method That Minimizes an Energy Integral of Fractional Order

H. Gunawan[1], F. Pranolo[1], and E. Rusyaman[2]

[1] Analysis and Geometry Group, Faculty of Mathematics and Natural Sciences, Bandung Institute of Technology, Bandung, Indonesia
hgunawan@math.itb.ac.id, fei101@students.itb.ac.id
[2] Department of Mathematics, Faculty of Mathematics and Natural Sciences, Padjadjaran University, Bandung, Indonesia
rusyaman@gmail.com

Abstract. An interpolation method that minimizes an energy integral will be discussed. To be precise, given $N+1$ points (x_0, c_0), $(x_1, c_1), \ldots,$ (x_N, c_N) with $0 = x_0 < x_1 < \cdots < x_N = 1$ and $c_0 = c_N = 0$, we shall be interested in finding a sufficiently smooth function u on $[0,1]$ that passes through these $N+1$ points and minimizes the energy integral $E_\alpha(u) := \int_0^1 |u^{(\alpha)}(x)|^2 dx$, where $u^{(\alpha)}$ denotes the fractional derivative of u of order α. As suggested in [1], a Fourier series approach as well as functional analysis arguments can be used to show that such a function exists and is unique. An iterative procedure to obtain the function will be presented and some examples will be given here.

1 Introduction

Many interpolation methods have been developed for many decades. For recent results in interpolation, see for instance [3, 7, 8, 9, 15] and the references therein. In [1], Alghofari discussed the following interpolation problem: Given $N+1$ points (x_0, c_0), $(x_1, c_1), \ldots,$ (x_N, c_N) with $0 = x_0 < x_1 < \cdots < x_N = 1$ and $c_0 = c_N = 0$, find a continuously differentiable function u on $[0,1]$ that passes through these $N+1$ points and minimizes the energy integral

$$E_2(u) := \int_0^1 |u''(x)|^2 dx.$$

To solve the problem, Alghofari used a Fourier series approach as well as functional analysis arguments. In particular, he showed that the problem has a unique solution and gave a hint to approximate the solution.

In this note, we shall generalize Alghofari's results by replacing the energy integral $E_2(u)$ with

$$E_\alpha(u) := \int_0^1 |u^{(\alpha)}(x)|^2 dx,$$

where $u^{(\alpha)}$ denotes the fractional derivative of u of order $\alpha \geq 0$. We show that for $\alpha > \frac{1}{2}$, the problem has a unique solution u which is continuous on $[0,1]$.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 151–162, 2008.
© Springer-Verlag Berlin Heidelberg 2008

In addition, an iterative procedure to obtain the solution will be presented and some examples will be given.

Note that for $\alpha = 2$, $E_2(u)$ represents the curvature (or the strain energy of bending) of u and the solution to the problem is a cubic spline (see [4, 5, 12]). For $\alpha = 1$, $E_1(u)$ represents the tension (or the potential energy of axial load) of u and the solution is a piecewise linear function. From this point of view, the interpolation that we discuss here can be considered as a generalization of the polynomial spline interpolation. Our results may be related to those in [14].

2 The Problem and Its Solution

We begin with the classical Fourier series discussion. Let $u : [0,1] \to \mathbb{R}$ be a continuous function with $u(0) = u(1) = 0$. If, for instance, u is piecewise smooth, then u may be expressed as a Fourier sine series

$$u(x) = \sum_{n=1}^{\infty} a_n \sin n\pi x, \quad x \in [0,1],$$

where

$$a_n = 2\int_0^1 u(x) \sin n\pi x \, dx, \quad n = 1, 2, 3, \ldots.$$

Parseval's identity states that $2\int_0^1 |u(x)|^2 dx = \sum_{n=1}^{\infty} a_n^2$. Further, if u is of class $C^{(k-1)}$ and $u^{(k-1)}$ is piecewise smooth (so that $u^{(k)}$ exists except at finitely many points and is piecewise continuous), then the Fourier sine coefficients a_n's satisfy the condition

$$\sum_{n=1}^{\infty} n^{2k} a_n^2 < \infty. \tag{1}$$

Conversely, if the coefficients a_n's satisfy the condition (1), then the functions $u, \ldots, u^{(k-1)}$ are absolutely continuous and $u^{(k)}$ is square integrable with

$$\|u^{(k)}\|_2^2 := 2\int_0^1 |u^{(k)}(x)|^2 dx = \pi^{2k} \sum_{n=1}^{\infty} n^{2k} a_n^2$$

(see, for instance, [6, 13]). All these tell us that we may identify $u^{(k)}$ with the square summable sequence $(n^k a_n)$. Here $n^k a_n$'s are the Fourier coefficients of $u^{(k)}$, from which we can recover $u^{(k)}$ almost everywhere through the formula

$$u^{(k)}(x) = \pi^k \sum_{n=1}^{\infty} n^k a_n \sin(n\pi x + k\tfrac{\pi}{2}).$$

Note that $\pi^k n^k \sin(n\pi x + k\frac{\pi}{2})$ is nothing but the k-th derivative of $\sin(n\pi x)$.

Inspired by the above facts, we may define the fractional derivative of u of order $\alpha \geq 0$, denoted by $u^{(\alpha)}$, almost everywhere by the following formula

$$u^{(\alpha)}(x) = \pi^\alpha \sum_{n=1}^{\infty} n^\alpha a_n \sin(n\pi x + \alpha\tfrac{\pi}{2}),$$

provided that $\sum_{n=1}^{\infty} n^{2\alpha} a_n^2 < \infty$. Notice that $\pi^\alpha n^\alpha \sin(n\pi x + \alpha\frac{\pi}{2})$ is the fractional derivative of $\sin n\pi x$ of order α (see [11]). Here we may check that the family $\{\sin(n\pi x + \alpha\frac{\pi}{2})\} : n \in \mathbb{N}\}$ forms an orthogonal system and that

$$2\int_0^1 |u^{(\alpha)}(x)|^2 dx = \pi^{2\alpha} \sum_{n=1}^{\infty} n^{2\alpha} a_n^2.$$

Accordingly, $u^{(\alpha)}$ is a square integrable function on $[0,1]$, which may be identified with the square summable sequence $(n^\alpha a_n)$.

Our problem is the following. Given $N+1$ points $(x_0, c_0), (x_1, c_1), \ldots, (x_N, c_N)$ with $0 = x_0 < x_1 < \cdots < x_N = 1$ and $c_0 = c_N = 0$, we wish to find an interpolant u which is continuous on $[0,1]$ and minimizes the energy integral

$$E_\alpha(u) := \int_0^1 |u^{(\alpha)}(x)|^2 dx. \tag{2}$$

To solve this problem, we consider the space $W = W_\alpha$ consisting of all functions u on $[0,1]$ of the form $u(x) = \sum_{n=1}^{\infty} a_n \sin n\pi x$ with $\sum_{n=1}^{\infty} n^{2\alpha} a_n^2 < \infty$. On W, we define the inner product

$$\langle u, v\rangle := \sum_{n=1}^{\infty} n^{2\alpha} a_n b_n,$$

where a_n's and b_n's are the coefficients of u and v, respectively. Here minimizing the integral $\int_0^1 |u^{(\alpha)}(x)|^2 dx$ in W is equivalent to minimizing the sum $\sum_{n=1}^{\infty} n^{2\alpha} a_n^2 =: \|u\|^2$. With respect to the above inner product, W is complete, that is, $(W, \langle\cdot,\cdot\rangle)$ is a Hilbert space. Indeed, given a Cauchy sequence in W, one may show that it is convergent to an element in W.

Hereafter, we shall assume that $\alpha > \frac{1}{2}$, unless otherwise stated. As we shall see, this is not only a sufficient condition but also necessary to have a continuous solution. Let us first prove the following lemma.

Lemma 2.1. *Suppose that $\|u_m - u\| \to 0$ as $m \to \infty$. Then, (u_m) converges uniformly to u on $[0,1]$. More generally, $\big(u_m^{(\beta)}\big)$ converges uniformly to $u^{(\beta)}$ on $[0,1]$ for $0 \le \beta < \alpha - \frac{1}{2}$.*

Proof. For $m \in \mathbb{N}$, let $a_{m,n}$'s and a_n's be the coefficients of u_m and u. Let $0 \le \beta < \alpha - \frac{1}{2}$. Then, for each $x \in [0,1]$, we have

$$\begin{aligned} |u_m^{(\beta)}(x) - u^{(\beta)}(x)| &= \Big|\pi^\beta \sum_{n=1}^{\infty} n^\beta (a_{m,n} - a_n) \sin n\pi x\Big| \\ &\le \pi^\beta \Big[\sum_{n=1}^{\infty} n^{2\alpha}(a_{m,n} - a_n)^2\Big]^{1/2} \Big[\sum_{n=1}^{\infty} \frac{\sin^2 n\pi x}{n^{2(\alpha-\beta)}}\Big]^{1/2} \\ &\le C\|u_m - u\|, \end{aligned}$$

where C is independent of x. Hence $\big(u_m^{(\beta)}\big)$ converges uniformly to $u^{(\beta)}$. □

Corollary 2.2. *If $u \in W$, then $u^{(\beta)}$ is continuous for $0 \leq \beta < \alpha - \frac{1}{2}$. In particular, every function in W is continuous.*

Proof. For each β with $0 \leq \beta < \alpha - \frac{1}{2}$, $u^{(\beta)}$ is a limit, and hence a uniform limit, of its partial sums. Now since the partial sums are continuous, $u^{(\beta)}$ too must be continuous. □

Now consider the subspace V of W consisting of all functions u that vanish at $x_i,\ i = 1, ..., N-1$; that is,

$$V := \{u \in W\ :\ u(x_i) = 0,\ i = 1, \ldots, N-1\}.$$

Meanwhile, let U be the subset of W given by

$$U := \{u \in W\ :\ u(x_i) = c_i,\ i = 1, \ldots, N-1\}.$$

Then, as for the case $\alpha = 2$ discussed in [1], we have:

Lemma 2.3. *V is closed, while U is nonempty, closed and convex.*

Proof. Let u be the limit of a convergent sequence (u_m) in V. Then, for each $i = 1, \ldots, N-1$, it follows from Lemma 2.1 that $u(x_i) = 0$ because $u_m(x_i) = 0$ for every $m \in \mathbb{N}$. Therefore V is closed. Similarly, U is closed. Next, it is nonempty because one can easily find a function $u_0(x) = \sum_{j=1}^{N-1} b_j \sin j\pi x$ satisfying the following system of equations

$$\sum_{j=1}^{N-1} b_j \sin j\pi x_i = c_i, \quad i = 1, \ldots, N-1.$$

Finally, if u_1 and u_2 in U, then $\alpha u_1 + \beta u_2 \in U$ provided that $\alpha + \beta = 1$. This tells us particularly that U is convex. □

The following theorem is a generalization of Alghofari's result [1].

Theorem 2.4. *The minimization problem (2) has a unique solution in W, and the solution is given by*

$$u = u_0 - \mathrm{proj}_V(u_0),$$

where u_0 is an arbitrary element of U and $\mathrm{proj}_V(u_0)$ denotes the orthogonal projection of u_0 on V.

Proof. Let u_0 be an element in U. Then, for any $v \in V$, $u_0 - v$ is also in U. Since U is a convex subset of W, there must exist a unique element $v_0 \in V$ such that $\|u_0 - v_0\|$ is of smallest norm [2]. Thus $u := u_0 - v_0$ is the unique solution in W for our minimization problem (2). By the theory of best approximation in Hilbert spaces, the element $v_0 \in V$ for which $\|u_0 - v_0\|$ is minimized is the orthogonal projection of u_0 on V, that is, $v_0 = \mathrm{proj}_V(u_0)$. □

As we have indicated before, to find an element in U is easy. What is rather difficult is to find an orthonormal basis for V. In the next section, we develop a procedure to find an initial element in U and an orthonormal basis for V, and to obtain the minimum solution iteratively through finite computations.

3 The Procedure to Obtain the Solution

Given $N+1$ points (x_0, c_0), $(x_1, c_1), \ldots,$ (x_N, c_N) with $0 = x_0 < x_1 < \cdots < x_N = 1$, we can obtain (or approximate) the solution to (2) in W through the following steps.

Step 1. To obtain an initial element in U, we solve the system of equations

$$\sum_{j=1}^{N-1} b_j \sin j\pi x_i = c_i, \quad i = 1, \ldots, N-1,$$

for the coefficients b_j's. The $(N-1) \times (N-1)$ matrix $[\sin j\pi x_i]_{i,j}$ is always nonsingular (see [2]), and so the above system has a solution. Having found b_j's, we put $u_0(x) = \sum_{j=1}^{N-1} b_j \sin j\pi x$.

Step 2. To obtain a basis for V, we consider the system of equations

$$\sum_{n=1}^{\infty} a_n \sin n\pi x_i = 0, \quad i = 1, \ldots, N-1,$$

each of which contains infinitely many unknowns a_n's. However, we can tackle this system by writing it as

$$\sum_{j=1}^{N-1} a_j \sin j\pi x_i = -\sum_{n=N}^{\infty} a_n \sin n\pi x_i, \quad i = 1, \ldots, N-1.$$

From this we can express $a_1, \ldots, a_{N-1}$ in terms of $a_n,\ n \geq N$.

Now if $(a_1, \ldots, a_{N-1}, a_N, a_{N+1}, a_{N+2} \ldots)$ stands for $\sum_{n=1}^{\infty} a_n \sin n\pi x$, then by expressing $a_1, \ldots, a_{N-1}$ in terms of a_n with $n \geq N$, every element in V can be expressed as

$$a_N(*, \ldots, *, 1, 0, 0, 0, \ldots) + a_{N+1}(*, \ldots, *, 0, 1, 0, 0, \ldots) + \\ + a_{N+2}(*, \ldots, *, 0, 0, 1, 0, \ldots) + a_{N+2}(*, \ldots, *, 0, 0, 0, 1, \ldots) + \cdots,$$

where the first $N-1$ terms marked by asterisks come from $a_1, \ldots, a_{N-1}$. The following sequence form a basis for V:

$$v_1 := (*, \ldots, *, 1, 0, 0, 0, \ldots),\ v_2 := (*, \ldots, *, 0, 1, 0, 0, \ldots),$$
$$v_3 := (*, \ldots, *, 0, 0, 1, 0, \ldots),\ v_4 := (*, \ldots, *, 0, 0, 0, 1, \ldots), \ldots.$$

Step 3. The minimum solution u is given by $u = u_0 - \mathrm{proj}_V(u_0)$. To find (or approximate) it, we compute the orthogonal projection of u_0 on the subspace $V_m := \mathrm{span}\{v_1, \ldots, v_m\}$ for $m = 1, 2, 3, \ldots$ iteratively. (But since v_n's may not be orthogonal, we might need to orthogonalize them first.) Now if $u_m := u_0 - \mathrm{proj}_{V_m}(u_0)$, then the sequence (u_m) approximates the minimum solution u. Indeed, $\|u_m\|$ gets smaller and $\|u_m - u\| \to 0$ as $m \to \infty$.

In practice, we may stop the iteration process at u_M basically when we have $\|u_M - u_{M-1}\| < \epsilon$ for a given value of ϵ. Note that the larger the value of α the faster the convergence of (u_m).

To illustrate how our procedure works, we present a few examples. The first one is simple; the reader can follow the computations in details.

Example 3.1. (a) Suppose that we wish to find a continuous, piecewise smooth function u on $[0,1]$ that minimizes the integral

$$E_1(u) := \int_0^1 |u'(x)|^2 \, dx, \tag{3}$$

subject to the condition that $u(0) = u(1) = 0$ and $u\big(\frac{1}{2}\big) = 1$.

For this, consider the subspace V of W consisting of all functions u that vanish at $\frac{1}{2}$; that is,

$$V := \{u \in W \,:\, u\big(\tfrac{1}{2}\big) = 0\},$$

and the subset U of W given by

$$U := \{u \in W \,:\, u\big(\tfrac{1}{2}\big) = 1\}.$$

Our initial approximation is $u_0(x) = \sin n\pi x$. Next, if $v(x) := \sum_{n=1}^{\infty} a_n \sin n\pi x$ is in V, then $v\big(\frac{1}{2}\big) = 0$ is equivalent to

$$a_1 - a_3 + a_5 - a_7 + - \cdots = 0,$$

for which we get

$$a_1 = a_3 - a_5 + a_7 - a_9 + - \cdots.$$

Hence, every element $(a_1, a_2, a_3, a_4, a_5, \ldots)$ in V can be expressed as

$$\begin{aligned} &a_2(0,1,0,0,0,\ldots) + a_3(1,0,1,0,0,\ldots) + \\ &+ a_4(0,0,0,1,0,\ldots) + a_5(-1,0,0,0,1,\ldots) + \cdots. \end{aligned}$$

From this we get the following basis for V:

$$\begin{aligned} &v_1 := (0,1,0,0,0,\ldots),\ v_2 := (1,0,1,0,0,\ldots), \\ &v_3 := (0,0,0,1,0,\ldots),\ v_4 := (-1,0,0,0,1,\ldots),\ \ldots. \end{aligned}$$

If one carries out Step 3 as prescribed, one will get $u_1 = (1,0,0,0,0,\ldots),\ u_2 = u_3 = \frac{9}{10}(1,0,-\frac{1}{3^2},0,0,\ldots)$, and so on. The limiting solution is

$$u = \frac{8}{\pi^2}(1,0,-\frac{1}{3^2},0,\frac{1}{5^2},\ldots).$$

Alternatively, one can compute the orthogonal complement of u_0 with respect to V directly as follows. If $u = (b_1, b_2, b_3, \ldots)$ is orthogonal to V, then $u \perp v_m$ for each $m \in \mathbb{N}$, and so $b_2, b_4, b_6, \ldots$ must be equal to 0 and

$$b_1 = -3^2 b_3 = 5^2 b_5 = -7^2 b_7 = \cdots.$$

Hence $u = b_1(1, 0, -\frac{1}{3^2}, 0, \frac{1}{5^2}, \ldots)$, that is,

$$u(x) = b_1\Big(\sin \pi x - \frac{1}{3^2}\sin 3\pi x + \frac{1}{5^2}\sin 5\pi x - \frac{1}{7^2}\sin 7\pi x + - \cdots\Big).$$

But $u\big(\frac{1}{2}\big) = 1$ gives

$$b_1 = \Big(\sum_{n=1}^{\infty} \frac{1}{(2n-1)^2}\Big)^{-1} = \frac{8}{\pi^2},$$

and therefore

$$u(x) = \frac{8}{\pi^2}\Big(\sin \pi x - \frac{1}{3^2}\sin 3\pi x + \frac{1}{5^2}\sin 5\pi x - \frac{1}{7^2}\sin 7\pi x + - \cdots\Big).$$

Notice that this is nothing but the Fourier sine series of the piecewise linear function f given by

$$f(x) = \begin{cases} 2x, & 0 \le x \le \frac{1}{2} \\ 2(1-x), & \frac{1}{2} < x \le 1. \end{cases}$$

The difference between the sequence (u_m) and the Fourier partial sums is that each u_m passes through the point $\big(\frac{1}{2}, 1\big)$ while the Fourier partial sums do not.

(b) In general, given $N+1$ points (x_0, c_0), $(x_1, c_1), \ldots, (x_N, c_N)$ with $0 = x_0 < x_1 < \cdots < x_N = 1$, the solution to the minimization problem (2) for $\alpha = 1$ is the Fourier sine series of the piecewise linear function f for which $f(x_i) = c_i$ and f is linear on each subinterval $[x_{i-1}, x_i]$.

For example, let $x_i = \frac{i}{4}$, $i = 0, \ldots, 4$, and $c_0 = 0$, $c_1 = \frac{7}{10}$, $c_2 = 1$, $c_3 = \frac{3}{10}$, $c_4 = 0$. With a computer program, we apply our procedure and get a sequence (u_m) that approximates the solution in W. We stop the iterations at u_M basically when $\|u_M - u_{M-1}\| < \epsilon$. For $\epsilon = 0.01$, the iterations stop at u_{182}. Figure 1 shows the graphs of u_0, u_5, u_{30}, and $u_M = u_{182}$, which clearly indicate that the limiting series must be that of the piecewise linear function passing through the points (x_i, c_i), $i = 0, \ldots, 4$.

For $\alpha = 1$, one may observe that the piecewise linear function f that passes through the given points always solves the minimization problem (2). This follows from the following fact.

Fact 3.2. *On every interval $[a, b]$ where $u(a)$ and $u(b)$ are fixed, the integral $\int_a^b |u'(x)|^2 dx$ is minimized (among continuously differentiable functions u) if and only if u is linear.*

Proof. Let $m := \frac{u(b)-u(a)}{b-a}$. Then, by the Fundamental Theorem of Calculus, we have

$$\begin{aligned} \int_a^b |u'(x) - m|^2 dx &= \int_a^b |u'(x)|^2 dx - 2m \int_a^b u'(x)\, dx + m^2(b-a) \\ &= \int_a^b |u'(x)|^2 dx - 2m[u(b) - u(a)] + m^2(b-a) \\ &= \int_a^b |u'(x)|^2 dx - m^2(b-a). \end{aligned}$$

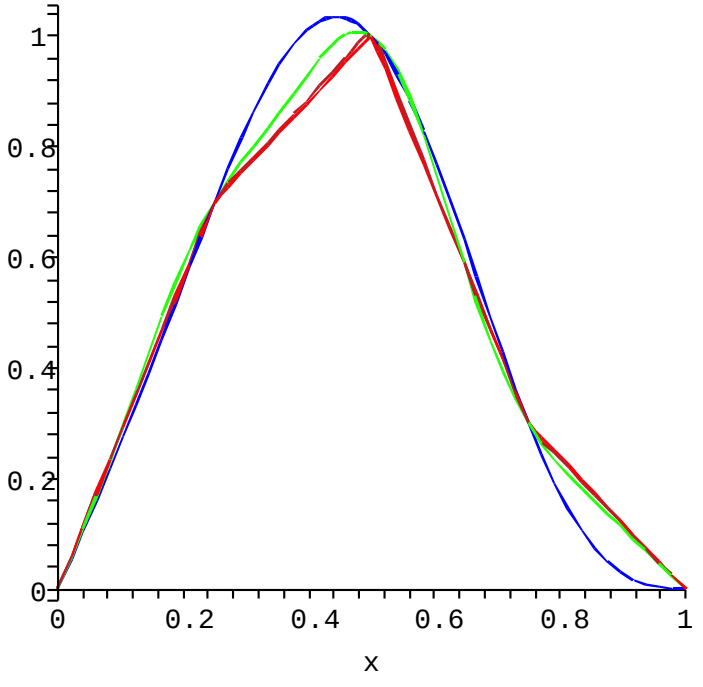


Fig. 1. $\alpha = 1;\ u(0) = 0,\ u\left(\frac{1}{4}\right) = \frac{7}{10},\ u\left(\frac{1}{2}\right) = 1,\ u\left(\frac{3}{4}\right) = \frac{3}{10},\ u(1) = 0;\ \epsilon = 0.01$

Hence $\int_a^b |u'(x)|^2 dx \geq \int_a^b m^2\, dx$, and $\int_a^b |u'(x)|^2 dx$ is minimized if and only if $u'(x) = m$ for every x in $[a, b]$; that is, if and only if u is linear. □

Example 3.3. Suppose that we wish to find a continuous function u that is twice differentiable almost everywhere on $[0, 1]$ and minimizes the integral

$$E_2(u) := \int_0^1 |u''(x)|^2\, dx, \tag{4}$$

subject to the condition that $u(0) = u(1) = 0$ and $u\left(\frac{1}{2}\right) = 1$.

Then, as in Example 3.1 (a), we will get

$$u(x) = \frac{96}{\pi^4}\left(\sin \pi x - \frac{1}{3^4}\sin 3\pi x + \frac{1}{5^4}\sin 5\pi x - \frac{1}{7^4}\sin 7\pi x + - \cdots\right),$$

which is the Fourier sine series of the cubic spline

$$f(x) = \begin{cases} 3x - 4x^3, & 0 \leq x \leq \frac{1}{2} \\ 3(1-x) - 4(1-x)^3, & \frac{1}{2} < x \leq 1. \end{cases}$$

In the next examples, we apply our procedure for fractional $\alpha's$ and we see what happens particularly when $\alpha > 1$ and $\frac{1}{2} < \alpha < 1$.

Example 3.4. (a) Suppose that $\alpha = 1.5$ and we wish to find a sufficiently smooth function u on $[0,1]$ that minimizes the integral $E_\alpha(u)$, subject to the condition that $u(0) = u(1) = 0$ and $u\left(\frac{1}{2}\right) = 1$.

Compared to Example 3.1(a), the function u here must be smoother at $\frac{1}{2}$. (From Lemma 2.2, we know that u has the fractional derivative u^β of order $\beta < 1$ which is continuous on $[0,1]$.) With a computer program, we apply our procedure with $\epsilon = 0.01$ and the iterations stop at u_{52}. Note that the convergence of (u_m) here is faster than that in Example 3.1(a). Figure 2 shows the graph of the approximate solution.

(b) Suppose now that $\alpha = 0.6$ and we wish to find a continuous function u on $[0,1]$ that minimizes the integral $E_\alpha(u)$, subject to the condition that $u(0) = u(1) = 0$ and $u\left(\frac{1}{2}\right) = 1$.

As one would expect, the function u now will be less smooth at $\frac{1}{2}$. Again, with a computer program, we apply our procedure with $\epsilon = 0.05$ and the iterations stop at u_{76} (we use a relatively large value of ϵ because the rate of convergence of (u_m) is expected to be low for small α). The graph of the approximate solution is shown in Figure 3.

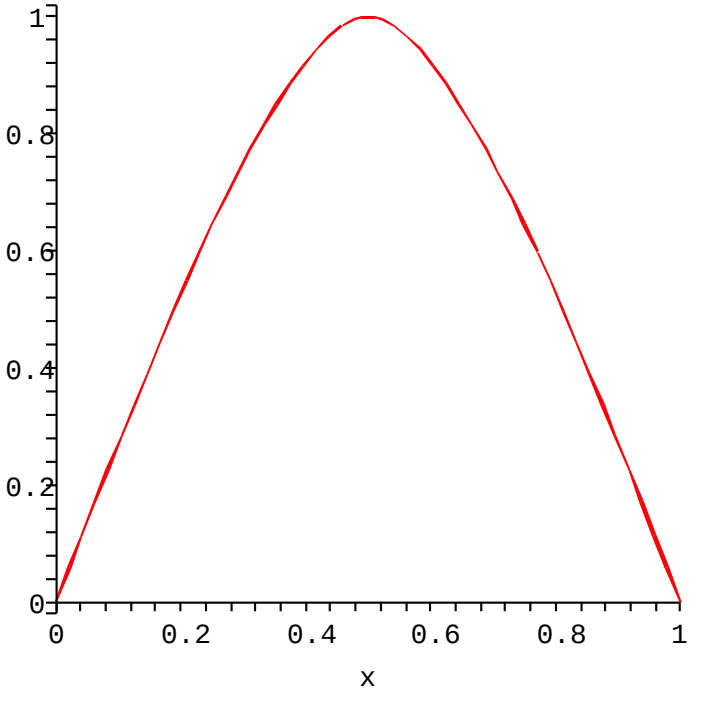


Fig. 2. $\alpha = 1.5;\ u(0) = 0,\ u\left(\frac{1}{2}\right) = 1,\ u(1) = 0;\ \epsilon = 0.01$

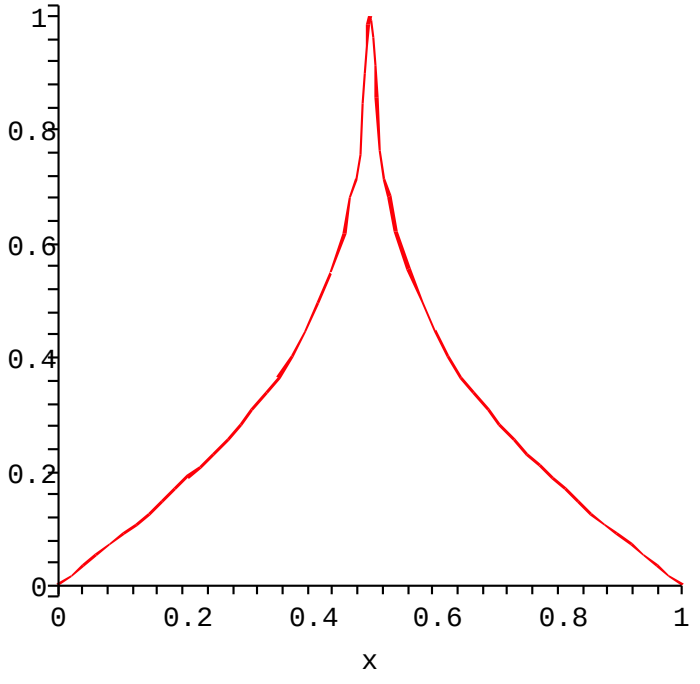


Fig. 3. $\alpha = 0.6;\ u(0) = 0,\ u\left(\frac{1}{2}\right) = 1,\ u(1) = 0;\ \epsilon = 0.05$

Remark. Our procedure also works for an energy functional which is a linear combination of several E_α's with at least one of the $\alpha's$ is greater than $\frac{1}{2}$. Moreover, we have been successful in extending our method to solve an analogous problem in 2-dimensional setting.

4 What Happens When $0 \leq \alpha \leq \frac{1}{2}$

Suppose that $0 \leq \alpha \leq \frac{1}{2}$ and we are trying to find a continuous function u on $[0, 1]$ that minimizes the integral $E_\alpha(u)$, subject to the condition that $u(0) = u(1) = 0$ and $u\left(\frac{1}{2}\right) = 1$.

To solve this problem, we consider the space W consisting of all functions u on $[0, 1]$ of the form $u(x) = \sum_{n=1}^{\infty} a_n \sin n\pi x$ with $\sum_{n=1}^{\infty} n^{2\alpha} a_n^2 < \infty$, equipped with the inner product

$$\langle u, v \rangle := \sum_{n=1}^{\infty} n^{2\alpha} a_n b_n,$$

where a_n's and b_n's are the coefficients of u and v, respectively.

As in Example 3.1, we consider the subspace V of W consisting of all functions u that vanish at $\frac{1}{2}$; that is,

$$V := \{u \in W\ :\ u\left(\tfrac{1}{2}\right) = 0\},$$

and the subset U of W given by

$$U := \{u \in W \,:\, u\big(\tfrac{1}{2}\big) = 1\}.$$

If $(a_1, a_2, a_3, \ldots) \in V$, then $a_1 - a_3 + a_5 - a_7 + - \cdots = 0$. Accordingly, the following vectors

$$\begin{aligned} v_1 &:= (0,1,0,0,0,\ldots),\ v_2 := (1,0,1,0,0,\ldots),\\ v_3 &:= (0,0,0,1,0,\ldots),\ v_4 := (-1,0,0,0,1,\ldots),\ \ldots \end{aligned}$$

form a basis for V.

As we can see, the above vectors also span W. Indeed, if $u = (b_1, b_2, b_3, \ldots)$ is orthogonal to V, then $u \perp v_m$ for each $m \in \mathbb{N}$, and so $b_2, b_4, b_6, \ldots$ are all 0 and

$$b_1 = -3^{2\alpha} b_3 = 5^{2\alpha} b_5 = -7^{2\alpha} b_7 = \cdots.$$

Hence $u = b_1(1, 0, -\frac{1}{3^{2\alpha}}, 0, \frac{1}{5^{2\alpha}}, \ldots)$. But then

$$\|u\|^2 = b_1^2\Big(1 + \frac{1}{3^{2\alpha}} + \frac{1}{5^{2\alpha}} + \cdots\Big) < \infty \text{ if and only if } b_1 = 0.$$

This means that $V^\perp = \{0\}$ or $V = W$. (In the infinite dimensional case, an equation like $a_1 - a_3 + a_5 - a_7 + - \cdots = 0$ does not have to define a hyperplane.)

Consequently, starting with our initial approximation $u_0 = (1, 0, 0, 0, \ldots)$, we will end up with $u = (0, 0, 0, 0, \ldots)$ or $u(x) = 0$ almost everywhere. Our procedure guarantees that the function u will satisfy $u\big(\frac{1}{2}\big) = 1$; but obviously u cannot be continuous at $\frac{1}{2}$.

Acknowledgement. H. Gunawan and F. Pranolo are supported by ITB Research Program No. 174/ K01.07/PL/2007. We are grateful to Dr. A.R. Alghofari for useful discussion about the subject. We also thank the reviewers for their comments on the earlier version of this paper.

References

[1] Alghofari, A.R.: Problems in Analysis Related to Satellites, Ph.D. Thesis, The University of New South Wales, Sydney (2005)
[2] Atkinson, K., Han, W.: Theoretical Numerical Analysis. Springer, New York (2001)
[3] Coleman, J.P.: Mixed interpolation methods with arbitrary nodes. J. Comput. Appl. Math. 92, 69–83 (1998)
[4] de Boor, C.: A Practical Guide to Splines. Springer, New York (2001)
[5] Farouki, R.: Pythagorean-Hodograph Curves: Algebra and Geometry Inseparable. Springer, New York (2008)
[6] Folland, G.B.: Fourier Analysis and Its Applications. Wadsworth & Brooks/Cole, Pacific Grove (1992)
[7] Jiang, T., Evans, D.J.: A discrete trigonometric interpolation method. Int. J. Comput. Math. 78, 13–22 (2001)

[8] Kim, K.J.: Polynomial-fitting interpolation rules generated by a linear functional. Commun. Korean Math. Soc. 21, 397–407 (2006)
[9] Kozak, J., Žagar, E.: On geometric interpolation by polynomial curves. SIAM J. Numer. Anal. 42, 953–967 (2004)
[10] Langhaar, H.L.: Energy Methods in Applied Mechanics. John Wiley & Sons, New York (1962)
[11] Oldham, K.B., Spanier, J.: The Fractional Calculus. Academic Press, New York (1974)
[12] von Petersdorff, T.: Interpolation with polynomials and splines, an applet (November 2007), http://www.wam.umd.edu/~petersd/interp.html
[13] Pinsky, M.A.: Introduction to Fourier Analysis and Wavelets. Brooks/Cole, Pacific Grove (2002)
[14] Unser, M., Blu, T.: Fractional splines and wavelets. SIAM Review 42, 43–67 (2000)
[15] Wallner, J.: Existence of set-interpolating and energy-minimizing curves. Comput. Aided Geom. Design 21, 883–892 (2004)

Solving Biomechanical Model Using Third-Order Runge-Kutta Methods

R.R. Ahmad[1,⋆], A.S. Rambely[1], and L.H. Lim[2]

[1] School of Mathematical Sciences, Faculty of Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia
Tel.: +60 3 89213716; Fax: +60 3 89254519
{rozy,asr}@ukm.my
[2] Maths Dept, Universiti Tunku Abdul Rahman, Petaling Jaya, Selangor, Malaysia
kellylimlh@yahoo.com
http://www.ukm.my

Abstract. A certain biomechanical model involves ordinary differential equations. This research focuses on solving a biomechanical model of a cyclist coasting downhill. The objective of this study is to establish the velocity of the model, using two numerical methods, i.e., the third-order Runge-Kutta methods. The two methods are the existing classical Runge-Kutta and a modified Runge-Kutta method formed by Wazwaz. The numerical results obtained from these two methods are compared with the exact solution and the relative errors are produced.

Keywords: Biomechanics problem, Runge-Kutta, Modified Runge-Kutta.

1 Introduction

Ordinary differential equations arise frequently in almost every discipline of science and engineering, such as biochemistry, biomedical system, weather prediction, mathematical biology and electronics, as a result of modeling and simulation activities. Numerical methods are techniques for solving these ordinary differential equations to give approximate solutions. These methods can be used not only to solve complicated problems such as the non-linear differential equations which usually do not have analytical solution but it also can solve functions that require a substantial computation. One of the widely used numerical methods is the Runge-Kutta methods, which comprise the second-order, third-order and fourth-order Runge-Kutta methods. This study focuses on the classical third-order Runge-Kutta method which is applied to a biomechanical problem in order to get the approximate numerical solution. The third-order Runge-Kutta method [1] is represented by

$$y_{n+1} = y_n + \frac{1}{6}K_1 + \frac{4}{6}K_2 + \frac{1}{6}K_3$$

⋆ Corresponding author.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 163–168, 2008.
© Springer-Verlag Berlin Heidelberg 2008

with

$$\begin{aligned} K_1 &= hf\left(t_n, y_n\right) \\ K_2 &= hf\left(t_n + \tfrac{1}{2}h, y_n + \tfrac{1}{2}K_1\right) \\ K_3 &= hf\left(t_n + h, y_n - K_1 + 2K_2\right). \end{aligned}$$

The comparison to this third-order method is a modified third-order Runge-Kutta method [3], which utilized the geometric and arithmetic mean, i.e.

$$\frac{(geometric\ \ mean)^2}{arithmetic\ \ mean}$$

in the construction of the formula. This modified third-order Runge-Kutta method created by Wazwaz [3] can be written in the form of

$$y_{n+1} = y_n + h\left(\frac{k_1 k_2}{k_1 + k_2} + \frac{k_2 k_3}{k_2 + k_3}\right)$$

with

$$\begin{aligned} k_1 &= hf\left(t_n, y_n\right) \\ k_2 &= hf\left(t_n + \tfrac{2}{3}h, y_n + \tfrac{2}{3}k_1\right) \\ k_3 &= hf\left(t_n + \tfrac{2}{3}h, y_n - \tfrac{2}{3}k_1 + \tfrac{4}{3}k_2\right) \end{aligned}$$

These two methods can be use to obtained approximate numerical solution in biomechanics models such as cycling, jumping, bending, smashing and golfing. The objective of this research is to determine the velocity of a cyclist coasting downhill using the classical third-order Runge-Kutta method and compare the numerical outcomes to the results obtained using the modified third-order Runge-Kutta method by Wazwaz [3]. Relative errors are calculated by matching up the numerical solutions with the exact solutions in order to deduce whether classical third-order Runge-Kutta method or the modified third-order Runge-Kutta method [3] will give a better numerical solution.

2 The Biomechanics Model of a Cyclist

Cycling is a sport activity which provides a lot of benefit to a cyclist such as to maintain a healthy life style, stabilize heart beating and decrease the risk of getting cardiovascular illness. The model used in this study is a cyclist coasting downhill with the assumption that there is no friction acting in it. All of the components on the surface are force and movement components. Therefore all the forces acting on the cyclist are gravitational force, normal force and air force. Figure 1 shows the body diagram of the model. The acceleration's equation of a cyclist coasting downhill is

$$a = g\sin\alpha - \frac{k}{m}v^2 \tag{1}$$

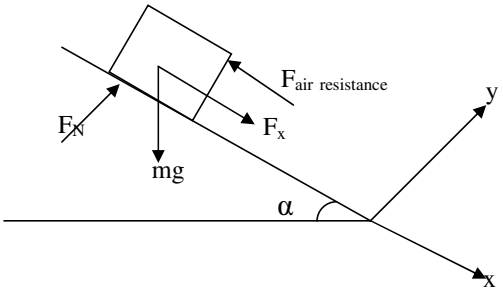


Fig. 1. Body diagram of a cyclist coasting downhill

with a representing acceleration of the cyclist, g is the gravitation acceleration, α is the angle of the hill from horizontal line, m is total mass of the bicycle and cyclist, k is the constant value of air resistance and ν is the velocity of cyclist coasting downhill. Since acceleration, a is the differentiation of velocity with respect to time, t that is

$$a = v'(t)$$

Therefore equation (1) can be written as

$$v' = g \sin \alpha - \frac{k}{m} v^2$$

with the exact solution for this model is given by

$$v = \sqrt{\frac{mg \sin \alpha}{k}} \tanh \left(\sqrt{g \sin \alpha} \frac{\sqrt{k}}{\sqrt{m}} t \right)$$

3 Numerical Solution

The values of g, m, α and k that had been chosen to solve this model are $g = 9.81 \text{ms}^{-1}$, $m = 75\text{kg}$, $\alpha = 10^0 = 0.17453$ rad and $k = 0.135\text{Nm}^{-1}$ where the cyclist is at 1000 m altitude. Therefore equation (1) becomes

$$v' = 9.81 \sin 10^0 - \frac{0.135}{75} v^2$$

with exact solution

$$v = \sqrt{\frac{735.75 \sin 10^0}{0.135}} \tanh \left(\sqrt{\frac{1.32435 \sin 10^0}{75}} t \right)$$

Table 1. Exact solution and relative errors for each method

t	Exact value	RK3	Relative Error 1	RK3WW	Relative Error 2
0	0.0000000000	0.0000000000	0.0000000E+00	0.0000000000	0.0000000E+00
10	15.3320065707	15.3321322126	1.2564185E-04	15.3320610377	5.4466989E-05
20	24.0316037944	24.0318581084	2.5431399E-04	24.0316948026	9.1008171E-05
30	27.4773130788	27.4775138952	2.0081640E-04	27.4773749250	6.1846147E-05
40	28.6424478483	28.6425552281	1.0737976E-04	28.6424779292	3.0080928E-05
50	29.0148517740	29.0148996642	4.7890192E-05	29.0148644389	1.2664924E-05
60	29.1317165822	29.1317360048	1.9422591E-05	29.1317215348	4.9526618E-06
70	29.1681782756	29.1681857238	7.4481887E-06	29.1681801290	1.8534229E-06
80	29.1795336964	29.1795364505	2.7540841E-06	29.1795343700	6.7358290E-07
90	29.1830681679	29.1830691606	9.9269620E-07	29.1830684076	2.3969770E-07
100	29.1841681089	29.1841684600	3.5111190E-07	29.1841681929	8.3948301E-08
110	29.1845103960	29.1845105184	1.2238950E-07	29.1845104251	2.9034901E-08
120	29.1846169095	29.1846169516	4.2170598E-08	29.1846169194	9.9410968E-09

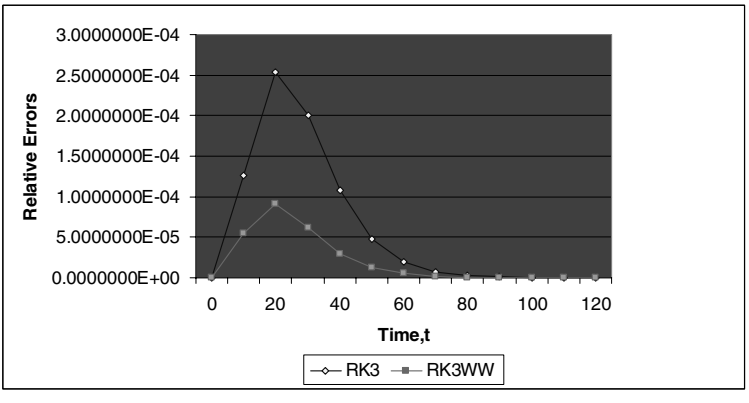


Fig. 2. Relative errors for each method

is considered over the range of time $0 \leq t \leq 120$. Using Mathematica software [4] with step size $h = 1$, the numerical results of every 10 steps using the classical third-order Runge-Kutta method (RK3) and the Wazwaz's [3] modified third-order Runge-Kutta method (RK3WW) is exhibited. For comparison, the relative errors are attained using both third-order Runge-Kutta methods as shown in Figure 2 with the exact solution (see Table 1).

Several different step sizes had also been chosen to obtain numerical solutions and relative errors for both third-order Runge-Kutta methods at time $t = 120$s as shown in Figure 3 (see also Table 2).

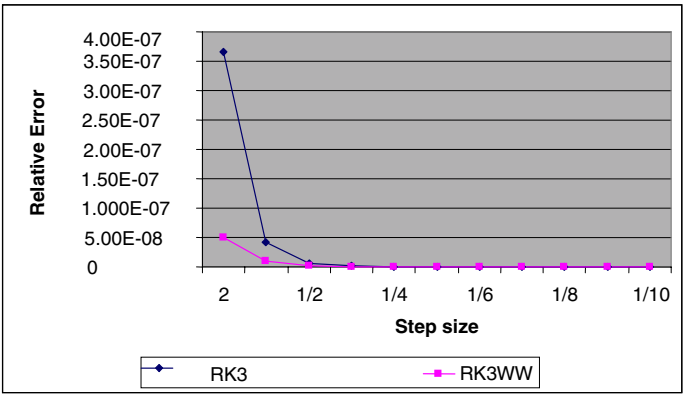


Fig. 3. Relative errors for each method at time $t = 120$s

Table 2. Exact solution and relative errors for each method at time $t = 120$s

Step size	Exact value	RK3	Relative Error 3	RK3WW	Relative Error 4
2	29.184611747224	29.184613217487	3.6697680E-07	29.184611500211	5.0780301E-08
1	29.184611747224	29.184616951641	4.2170598E-08	29.184616919412	9.9410968E-09
1/2	29.184611747224	29.184616914519	5.0482001E-09	29.184616911493	2.0220980E-09
1/3	29.184611747224	29.184616910945	1.4742980E-09	29.184616910138	6.6679817E-10
1/4	29.184611747224	29.184616910088	6.1749716E-10	29.184616909766	2.9490010E-10
1/5	29.184611747224	29.184616909786	3.1479885E-10	29.184616909626	1.5509727E-10
1/6	29.184611747224	29.184616909652	1.8169999E-10	29.184616909562	9.1297636E-11
1/7	29.184611747224	29.184616909585	1.1419843E-10	29.184616909529	5.8197003E-11
1/8	29.184611747224	29.184616909547	7.6397555E-11	29.184616909510	3.9300119E-11
1/9	29.184611747224	29.184616909524	5.3599791E-11	29.184616909499	2.7799985E-11
1/10	29.184611747224	29.184616909510	3.9097614E-11	29.184616909491	2.0399682E-11

4 Conclusion

This research generally discusses about a biomechanical model of a cyclist coasting downhill, which is solved using two different third-order Runge-Kutta methods namely, the classical third-order Runge-Kutta and the modified third-order Runge-Kutta methods. The numerical results obtained are compared between both methods. The numerical solutions obtained using the modified third-order Runge-Kutta method [3] gives better numerical results when compared to the classical third-order Runge-Kutta method. The results of this study also conclude that both third-order Runge-Kutta methods can be utilized to obtain numerical solutions for the biomechanical model. The results show excellent accuracy of both methods as compared with the exact solutions.

Acknowledgement. The financial support received from Universiti Kebangsaan Malaysia under the Research University grant (Project code: UKM-GUP-TMK-07-02-107) is gratefully acknowledged.

References

1. Rao, S.S.: Applied Numerical Methods for Engineers and Scientists. Prentice Hall, New Jersey (2002)
2. Nigg, B.M., Herzog, W. (eds.): Biomechanics of the Musculo-Skeletal system. John Wiley and Sons, England (2001)
3. Wazwaz, A.-M.: A Modified Third-order Runge-Kutta method. Appl. Math Letter. 3(3), 123–125 (1990)
4. Wolfram, S.: Mathematica: A System For Doing Mathematics By Computer,2nd edn. Addison-Wesley, Reading (1991)

An Efficient Fourth Order Implicit Runge-Kutta Algorithm for Second Order Systems

Basem S. Attili

University of Sharjah
Mathematics Department
P. O. Box 27272
Sharjah - United Arab Emirates
b.attili@uaeu.ac.ae

Abstract. We will present an algorithmic approach to the implementation of a fourth order two stage implicit Runge-Kutta method to solve periodic second order initial value problems. The systems involved will be solved using some type of factorization that usually involves both complex and real arithmetic. We will consider the real type case which will be efficient and leads to a system that is one fourth the size of similar systems using normal implicit Runge-Kutta method. We will present some numerical examples to show the efficiency of the method.

1 Introduction

The problem under consideration is the implicit second order initial value problem of the form

$$\begin{aligned} y'' &= f(t,y);\ t \geq a \\ y(a) &= y_a;\ y'(a) = y'_a \end{aligned} \tag{1}$$

where $f : R \times R^n \longrightarrow R^n$. In applied sciences, problems of this form arise in nonlinear oscillation problems where they have the form

$$My'' = f(t,y);\ t > 0;\ y(0) \text{ and } y'(0) \text{ given} \tag{2}$$

with M a positive definite $n \times n$ matrix called the mass matrix and f is a differentiable function. They may also arise in other applied sciences and engineering such as structural mechanics, see Carpenter et. al.[5]. The solution to (2) is usually oscillatory. When numerical methods are applied to test problems of the form $y'' = -wy$; $w > 0$ stability problems arise since the general solution is of the form $y = A\cos(wt + \alpha)$, see Burder[3] and Sharp, Fine and Burrage[19].

This problem was treated by several authors using different numerical methods and analysis. We mention for example well posed problems that have some singularities at some boundaries were treated by Attili et. al.[1] who considered explicit Runge-Kutta methods for such singular problems. Sharp et. al.[19] developed one class of numerical methods based on Runge-Kutta Nystrom methods.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 169–178, 2008.
© Springer-Verlag Berlin Heidelberg 2008

They have appropriate stability and oscillation properties. For error analysis see de-Swart and Soderlind[10] and Olsson and Soderlind[15]. Cash [6,7] developed a p-stable method for periodic initial value problems. Others like Chawla[8] developed an unconditionally stable Noumerov-type method, Cooper and Butcher [9] considered an iterative method, Butcher and Chatier[4] presented a one stage method, Xiao[21] considered a two stage method and Gladwell and Wang[12] presented analysis of two- and three step methods for second order systems and Shampine[18] dealt with implicit methods for solving ODE's. Other examples of the implementation of implicit Runge-kutta methods are Attili et. al.[2] who considered second order systems, Ramos et.al.[16] who developed a fourth-order method of BDF-type for solving stiff initial-value problems and Imoni et.al.[13] who considered second-order ordinary differential equations possessing oscillatory solutions. Parallel implementation of the implicit Runge-Kutta and use of predictor corrector can be found in Li and Gan[14] and Voss and Muir [20].

We will consider the efficient implementation of a fourth order two stage implicit Runge-Kutta method to solve second order systems of the form given in (2). The method is known to be stable. The technique used to solve the resulting systems will be to factorize the operator involved after the discretization. One factorization will involve complex arithmetic while two other suggested factorizations will avoid such complex arithmetic. We will consider the ones that involve real arithmetic. Such factorization will make the systems involved efficient and smaller in size. This type of treatment will be considered in Section 2. The numerical details will be done in Section 3 and finally in Section 4, we will present some numerical examples to show the efficiency of the suggested algorithm.

2 The Implicit Runge-Kutta Method

We will consider the initial value problem of the form (1.2); that is,

$$M\ddot{y} = f(y); y(0) = y_0; \dot{y}(0) = \dot{y}_0; t > 0. \tag{3}$$

We may rewrite (3) as a first order system of the form

$$Y' = F(Y);\ Y(0) = Y_0;\ t > 0 \tag{4}$$

where $Y = \begin{bmatrix} y \\ \dot{y} \end{bmatrix}$; $F(Y) = \begin{bmatrix} \dot{y} \\ M^{-1}f(y) \end{bmatrix}$ and $Y_0 = \begin{bmatrix} y_0 \\ \dot{y}_0 \end{bmatrix}$.

The implicit fourth order Runge-Kutta method for (4) will be

$$Y_{n+1} = Y_n + \frac{h}{2}\left[F(Y_1) + F(Y_2)\right] \tag{5}$$

where

$$\begin{aligned} Y_1 &= Y_n + \frac{h}{4}\left[F(Y_1) + \alpha m F(Y_2)\right] \\ Y_2 &= Y_n + \frac{h}{4}\left[\alpha p F(Y_1) + F(Y_2)\right] \end{aligned} \tag{6}$$

with $\alpha m = 1 - \frac{2\sqrt{3}}{3}$ and $\alpha p = 1 + \frac{2\sqrt{3}}{3}$, see Ehle and Picel[11], Cooper and Butcher[9], Serbin[17] and Gladwell and Wang[12]. To implement the method, one solves (6) for Y_1 and Y_2 using Newton's method. Then substitute back into (5) to obtain a new Y_{n+1}. To do so, the Newton's iterates will be

$$J(Y^{p-1})\Delta Y^p = -F(Y^{p-1})$$

or in details

$$\begin{bmatrix} I - \frac{h}{4}\frac{\partial F}{\partial y} & -\alpha m \frac{h}{4}\frac{\partial F}{\partial y} \\ -\alpha p \frac{h}{4}\frac{\partial F}{\partial y} & I - \frac{h}{4}\frac{\partial F}{\partial y} \end{bmatrix} \begin{bmatrix} \Delta Y_1^p \\ \Delta Y_2^p \end{bmatrix} \tag{7}$$
$$= - \begin{bmatrix} Y_1^{p-1} - Y_n - \frac{h}{4}F(Y_1^{p-1}) - \frac{h}{4}\alpha m F(Y_2^{p-1}) \\ Y_2^{p-1} - Y_n - \frac{h}{4}\alpha p F(Y_1^{p-1}) - \frac{h}{4}F(Y_2^{p-1}) \end{bmatrix}$$

If the system in (3) is of order s, then the system in (7) will be of order $4s$. This means this approach is inefficient for large s. Instead let us derive another approach. Consider (6); that is,

$$Y_1 = Y_n + \frac{h}{4}\left[F(Y_1) + \alpha m F(Y_2)\right] \tag{8}$$

$$Y_2 = Y_n + \frac{h}{4}\left[\alpha p F(Y_1) + F(Y_2)\right]. \tag{9}$$

From (8), solve for $\frac{h}{4}F(Y_2)$ to obtain

$$\frac{h}{4}F(Y_2) = \left[Y_1 - Y_n - \frac{h}{4}F(Y_1)\right]\frac{1}{\alpha m}$$

or

$$\frac{h}{4}F(Y_2) = \left[-3\alpha p Y_1 + 3\alpha p Y_n + \frac{3h}{4}\alpha p F(Y_1)\right]. \tag{10}$$

Substitute (10) into (9) to obtain

$$Y_2 = Y_n + \frac{h}{4}\alpha p F(Y_1) - 3\alpha p Y_1 + 3\alpha p Y_n + \frac{3h}{4}\alpha p F(Y_1)$$

which when simplified leads to

$$Y_2 = \left[Y_n(1 + 3\alpha p) - 3\alpha p Y_1 + h\alpha p F(Y_1)\right]. \tag{11}$$

Substituting (11) in (8), we obtain

$$Y_1 - Y_n - \frac{h}{4}F(Y_1) - \alpha m \frac{h}{4}F\left[Y_n(1 + 3\alpha p) - 3\alpha p Y_1 + h.\alpha p F(Y_1)\right] = 0 \tag{12}$$

a system of order $2s$. This means one can solve (12) using Newton's method for Y_1 then recover Y_2 from (11) and update Y from (5). Again (12) can be solved

using Newton's method. To carry out the work in the first part, the Jacobian of (12) is

$$\begin{aligned} J &= I - \frac{h}{4}\frac{\partial F}{\partial y} - \alpha m \frac{h}{4}\frac{\partial F}{\partial y}\left(-3\alpha p + \alpha p.h\frac{\partial F}{\partial y}\right) \\ &= I - \frac{h}{4}\frac{\partial F}{\partial y} + \alpha p.\alpha m\frac{3h}{4}\frac{\partial F}{\partial y} - \alpha p.\alpha m\frac{h^2}{4}\left(\frac{\partial F}{\partial y}\right)^2 \end{aligned}$$

with $\alpha p.\alpha m = \frac{-1}{3}$ and using some approximations. Hence the Jacobian simplifies to (similar systems were considered by Cooper and Butcher[9])

$$J = I - \frac{h}{2}\frac{\partial F}{\partial y} + \frac{h^2}{12}\left(\frac{\partial F}{\partial y}\right)^2. \tag{13}$$

As a result, Newton's method will be

$$J(Y_1^{p-1})\Delta Y_1^p = -H\left(Y_1^{p-1}\right)$$

where $\Delta Y_1^p = Y_1^p - Y_1^{p-1}$ and

$$H\left(Y_1^{p-1}\right) = Y_1^{p-1} - Y_n - \frac{h}{4}F(Y_1^{p-1}) - \alpha m\frac{h}{4}F\left[Y_n(1+3\alpha p) - 3\alpha p Y_1^{p-1} + h\alpha p F(Y_1^{p-1})\right]. \tag{14}$$

One can easily see that the operator (13) can be factorized as

$$J = \left(I - rh\frac{\partial F}{\partial y}\right)\left(I - \bar{r}h\frac{\partial F}{\partial y}\right);\ r = \frac{1}{4} + i\frac{\sqrt{3}}{12}$$

where r is unfortunately complex. This has the drawback of having to use complex arithmetic. To avoid so, one can use a perfect square factorization of the form

$$J = \left(I - rh\frac{\partial F}{\partial y}\right)^2$$

with $r = \frac{\sqrt{3}}{6}$ leading to discrepancy in the linear term or $r = \frac{1}{4}$ leading to discrepancy in the quadratic term. With either of the latter factorizations, (13) becomes

$$\left(I - rh\frac{\partial F}{\partial y}\right)^2 \Delta Y_1^p = -H\left(Y_1^{p-1}\right). \tag{15}$$

This implies that the solution can be obtained in two stages but using the same matrix; that is, solve

$$\begin{aligned} \left(I - rh\frac{\partial F}{\partial y}\right) Z &= -H\left(Y_1^{p-1}\right) \\ \left(I - rh\frac{\partial F}{\partial y}\right)\Delta Y_1^p &= Z. \end{aligned} \tag{16}$$

One factorization with two back substitutions.

3 The Algorithm

To reflect back on our original system given by (4), we will carry out the details of the systems given by (16). We can write

$$H\left(Y_1^{p-1}\right) = \begin{bmatrix} H_1^{p-1} + M^{-1}H_2^{p-1} \\ H_3^{p-1} + M^{-1}H_4^{p-1} \end{bmatrix} \text{ and } Y_1^{p-1} = \begin{bmatrix} u_1^{p-1} \\ u_2^{p-1} \end{bmatrix} \tag{17}$$

where

$$\begin{aligned} H_1^{p-1} &= y_n - u_1^{p-1} + \frac{h}{4}(\alpha m - 1)\dot{y}_n + \frac{h}{2}u_2^{p-1} \\ H_2^{p-1} &= \frac{-h^2}{12}f(u_1^{p-1}); \quad H_3^{p-1} = \dot{y}_n - u_2^{p-1} \\ H_4^{p-1} &= \frac{h}{4}f(u_1^{p-1}) + \frac{h}{4}\alpha m f((1+3\alpha p)y_n - 3\alpha p u_1^{p-1} + h\alpha p u_2^{p-1}). \end{aligned} \tag{18}$$

Notice also that the matrix involved in the right hand side of (16) is of the form

$$I - rh\frac{\partial F}{\partial y} = \begin{bmatrix} I & -rhI \\ -rhM^{-1}\frac{\partial f}{\partial y} & I \end{bmatrix}. \tag{19}$$

Here $\frac{\partial f}{\partial y} = J$ is assumed a constant Jacobian at each step and let $z = (z_1 \quad z_2)^T$.

Now for elimination purposes, we multiply the first part of (16) from left by

$$C = M\begin{bmatrix} I & rhI \\ 0 & I \end{bmatrix}. \tag{20}$$

to obtain

$$\begin{bmatrix} M - r^2h^2J & 0 \\ -rhJ & M \end{bmatrix}\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} M\left(H_1^{p-1} + rhH_3^{p-1}\right) + H_2^{p-1} + rhH_4^{p-1} \\ MH_3^{p-1} + H_4^{p-1} \end{bmatrix}.$$

This leads to

$$\left(M - r^2h^2J\right)z_1 = M\left(H_1^{p-1} + rhH_3^{p-1}\right) + H_2^{p-1} + rhH_4^{p-1}. \tag{21}$$

From the first row of the first part of (16), we will have

$$z_1 - rhz_2 = H_1^{p-1} + M^{-1}H_2^{p-1}$$

and hence

$$z_2 = \left(z_1 - H_1^{p-1} - M^{-1}H_2^{p-1}\right)\Big/ rh \tag{22}$$

which means z_1 and z_2 can be computed. Now for the second part of (16) again premultiplying by the matrix C given by (20) will lead to

$$\begin{bmatrix} M - r^2h^2J & 0 \\ -rhJ & M \end{bmatrix} \begin{bmatrix} u_1^p - u_1^{p-1} \\ u_2^p - u_2^{p-1} \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}.$$

In a similar way we will have

$$\left(M - r^2h^2J\right)\left(u_1^p - u_1^{p-1}\right) = M\left(z_1 + rhz_2\right)$$

and as in (22)

$$\left(M - r^2h^2J\right)\left(u_1^p - u_1^{p-1}\right) = M\left(2z_1 - H_1^{p-1}\right) - H_2^{p-1}. \tag{23}$$

We will repeat a similar argument here to compute $u_2^p - u_2^{p-1}$ and hence u_2^p; that is, from the first row of the second part of (16), we will have

$$\left(u_1^p - u_1^{p-1}\right) - rh\left(u_2^p - u_2^{p-1}\right) = z_1$$

leading to

$$u_2^p = u_2^{p-1} + \left[\left(u_1^p - u_1^{p-1}\right) - z_1\right] \Big/ rh. \tag{24}$$

Having u_1^p and u_2^p we can compute Y_2 from (11) and Y_1 from (12). Instead if we substitute Y_1 and Y_2 in (5) directly and simplify the result, we obtain

$$\begin{aligned} My_{n+1} &= \frac{h^2}{2}\alpha p f(u_1) + M\left[y_n + \frac{h}{2}\dot{y}_n(1 + 3\alpha p) + \frac{h}{2}u_2(1 - 3\alpha p)\right] \\ M\dot{y}_{n+1} &= M\dot{y}_n + \frac{h}{2}f(u_1) + \frac{h}{2}f\left(y_n(1 + 3\alpha p) - 3\alpha p u_1 + \alpha p h u_2\right). \end{aligned} \tag{25}$$

This will lead to the following algorithm:

Algorithm: Assume y_n, $\dot{y}_n$ and $J = \frac{\partial f}{\partial y}$ have been computed. Then to compute y_{n+1} and $\dot{y}_{n+1}$

1. Set $p = 1$ and predict u_1^0 and u_2^0 then carry out Newton's iteration,
2. Evaluate $f(u_1^{p-1})$ and $f\left(y_n(1 + 3\alpha p) - 3\alpha p u_1^{p-1} + \alpha p h u_2^{p-1}\right)$.
3. Form H_1^{p-1}, H_2^{p-1}, H_3^{p-1} and H_4^{p-1} using (18).
4. Solve the systems (21), (22), (23) and (24) for z_1, z_2, u_1^p and u_2^p.
5. Set $p = p + 1$ and repeat until convergence

Then calculate y_{n+1} and $\dot{y}_{n+1}$ from (25).

Note that it is advisable to deal with hz_2 and hw_2 in (22) and in (24) respectively in order to avoid dividing by what possibly might be very small step size h.

4 Numerical Experimentation

For numerical examples we consider

Example 1: Consider

$$y'' + 64y = 0$$
$$y(0) = \frac{1}{4}, \quad y'(0) = -\frac{1}{2},$$

Solving using the Algorithm with $h = 0.01$, the results obtained are plotted against the exact solution in Figure 1. The exact solution for this system is $y(t) = \frac{\sqrt{17}}{16}\sin(8t + \theta)$; $\theta = \pi - \arctan(4)$. The errors at $x = 1$ for step sizes $h = 0.01$ and $h = 0.005$ were respectively 0.1084×10^{-6} and 0.6776×10^{-8}. This means the Algorithm is producing an order of 3.99979; that is, $O(h^4)$ approximations to the solution of the system as expected by the algorithm.

Example 2: Consider the nonlinear initial value problem

$$y'' + y^3 = 0$$
$$y(0) = 1, \quad y'(0) = 1.$$

We solved the problem using NDSolve Mathematica routine and the Algorithm. The solutions obtained and the errors at selected points are given in Table 4.1 for $h = 0.01$.

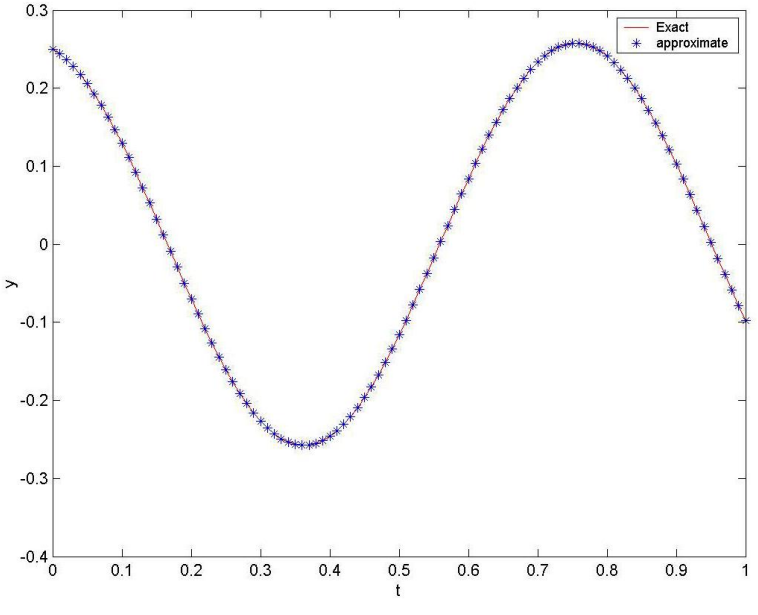


Fig. 1. The approximate solution against the exact

x	Mathematica Sol.	Algorithm	Error
0.0	1.0	1.0	0.0
0.1	1.0944893860584493	1.09448938558520	4.7324-10
0.2	1.1758644626560342	1.17586446434978	1.6937-9
0.3	1.2410056491546735	1.24100565087605	1.7213-9
0.4	1.2870881345737024	1.28708814223123	7.6575-9
0.5	1.3119284869961894	1.31192851904109	3.2044-8
0.6	1.314283942920852	1.31428399609513	5.3174-8
0.7	1.2940342408482368	1.29403429001151	4.9163-8
0.8	1.2521995980449039	1.25219965290282	5.4857-8
0.9	1.1907903129619792	1.19079031982329	6.8613-9
1.0	1.1125262352847067	1.11252624109804	5.813-9

Table 4.1: Results for Example 2 at selected points.

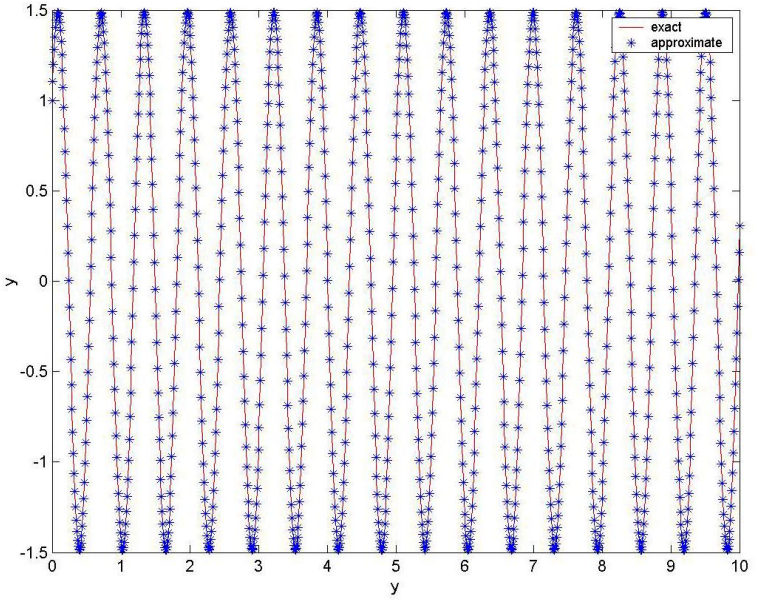


Fig. 2. The solution against exact with $\varpi = 100$

Example 3

$$y'' = -\varpi^2 y$$
$$y(0) = 1, \quad y'(0) = 1 + \varpi,$$

which has $y = \cos \varpi x + \sin \varpi x$ as exact solution. With $\varpi = 100$, the approximate solutions against the exact ones are given in Figure 2. As seen from Figure 2

and since $\varpi = 100$, the solution is oscillating and yet the results obtained using the algorithm are accurate.

In conclusion and from the examples above, it is clear that the proposed algorithm produced a fourth order accuracy even when solutions are highly oscillating. The amount of work done is significantly less since the systems solved are one fourth the size of those using normal implicit Runge-Kutta method.

5 Conclusions

We have presented a fourth order two stage implicit Runge-Kutta method for solving second order systems. The method used is known to be stable. To implement the method we considered the factorization of the discretized operator that involves real arithmetic. This resulted in systems that are one fourth the size of the original systems. Leading to significant saving in the amount of work done without sacrificing the fourth order accuracy as the numerical examples show. Some future work will be to explore the possibility of using such an approach in solving differential algebraic systems.

References

1. Attili, B., Elgindi, M., Elgebeily, M.: Initial Value Methods for the Eigenelements of Singular Two-Point Boundary Value Problems, AJSE, 22(2C), pp. 67–77 (1997)
2. Attili, B., Furati, K., Syam, M.: An Efficient Implicit Runge-Kutta Method for Second Order Systems. Applied Math. Comput. 178, 229–238 (2006)
3. Burder, J.: Linearly Implicit Runge-Kutta Methods Based on Implicit Runge-Kutta Methods. Appl. Numer. Math. 13, 33–40 (1993)
4. Butcher, J., Chartier, P.: The Effective Order of Singly-Implicit Runge-Kutta Methods. Numer. Algorithms 20, 269–284 (1999)
5. Carpenter, M.H., Kennedy, C.A., Bij, H., Viken, S.A., Vatsa, V.N.: Fourth-order Runge-Kutta schemes for fluid mechanics applications. J. Sci. Comput. 25, 157–194 (2005)
6. Cash, J.: High Order P-stable Formulae for Periodic Initial Value Problems. Numer. Math. 37, 355–370 (1981)
7. Cash, J.: Efficient P-stable Methods for Periodic Initial Value Problems. BIT 24, 252–284 (1984)
8. Chawla, M.: Unconditionally Stable Noumerov-Type Methods for Second Order Differential Equations. BIT 23, 541–552 (1983)
9. Cooper, J., Butcher, J.: An Iterative Scheme for Implicit Runge-Kutta Methods. IMA J. Numer. Anal. 3, 127–140 (1983)
10. de-Swart, J., Soderlind, G.: On the Construction of Error Estimators for Implicit Runge-Kutta Methods. J. Comput. Appl. Math. 86, 347–358 (1997)
11. Ehle, B., Picel, Z.: Two Parameter Arbitrary Order Exponential Approximations for Stiff Equations. Math. Comp. 29, 501–511 (1975)
12. Galdwell, I., Wang, J.: Iterations and Predictors for Second Order Systems. In: Ames, W.F. (ed.) Proceedings, 14th IMACS World Congress on Comput. and Appl. Math., Georgia, vol. 3, pp. 1267–1270 (1994)

13. Imoni, S.O., Otunta, F.O., Ramamohan, T.R.: Embedded Implicit Runge-Kutta Nystrom Method for Solving Second-Order Differential Equations. Int. J. Comput. Math. 83, 777–784 (2006)
14. Li, S.F., Gan, S.: A class of Parallel Multistep Runge-Kutta Predictor-Corrector Algorithms. J. Numer. Methods Comput. Appl. 17, 1–11 (1995)
15. Olsson, H., Soderlind, G.: Stage Value Predictors and Efficient Newton Iterations in Implicit Runge-Kutta Methods. SIAM J. Sci. Comput. 20, 185–202 (1999)
16. Ramos, H., Vigo-Aguiar, J.: A fourth-Order Runge-Kutta Method Based on BDF-Type Chebyshev Approximations. J. Comput. Appl. Math. 204, 124–136 (2007)
17. Serbin, M.: On Factoring A class of Complex Symmetric Matrices Without Pivoting. Math. Comput. 35, 1231–1234 (1980b)
18. Shampine, L.: Implementation of Implicit Formulas for the Solution of ODE's. SIAM J. Sci. Stat. Comput. 1, 103–118 (1980)
19. Sharp, P., Fine, J., Burrage, K.: Two-Stage and Three Stage Diagonally Implicit Runge-Kutta Nystrom Methods of Order Three and Four. IMA J. Numer. Anal. 10, 489–504 (1990)
20. Voss, D., Muir, P.: Mono-Implicit Runge-Kutta Schemes for the Parallel Solutions of Initial Value ODE's. J. Comput. Appl. Math. 102, 235–252 (1999)
21. Xiao, A.: Order Results for Algebraically Stable Mono-Implicit Runge-Kutta Methods. J. Comput. Appl. Math. 17, 639–644 (1999)

Laplace Equation Inside a Cylinder: Computational Analysis and Asymptotic Behavior of the Solution

Suvra Sarkar and Sougata Patra

Department of Electronics and Communication Engineering
Haldia Institute of Technology
Indian Centre for Advancement in Research and Education
Haldia, West Bengal
ssarkar.ece@gmail.com, spatra.ece@gmail.com

Abstract. The Laplacian in the cylindrical coordinate space has been considered to approximate the solution of a conservative field within a restricted domain.

$$\frac{\partial^2\psi}{\partial\rho^2}+\frac{1}{\rho}\frac{\delta\psi}{\delta\rho}+\frac{1}{\rho^2}\frac{\partial^2\psi}{\partial\phi^2}+\frac{\partial^2\psi}{\partial z^2}=0$$

Solutions of the Laplacian are represented by expansion in series of the appropriate orthonormal functions. By using asymptotic relations of Bessel Series and Fourier Bessel series, we establish some criteria for the solution to properly reflect the nature of the conservative field.

Keywords: Bessel functions, Fourier-Bessel Series, Kronecker Delta, Laplace Equation.

1 Introduction

Many problems in electrostatics involve boundary surfaces on which either the potential or the surface charge density is specified. A somewhat idealized solution to the practical situations has been presented here. Consequently a number of approaches to electrostatic boundary value problems have been developed. The differential equation involving the conservative field has been approached through expansions in orthogonal functions. The solution of the Laplace equation so generated was decomposed into a product of factors for the three variables ρ, z and φ. The use of Bessel functions in theoretical physics literature is overwhelming; these form an orthogonal, complete set of functions, which are solutions to the radial component of the Laplacian. In the present problem we have tried to predict the behavior of the conservative field and predict its nature using graphical analysis through computational methods. As a first step, a cylindrical coordinate space was used for the given situation and the required Laplace's equation was formed. The partial differential equations so obtained were simplified using the separation of variables, and three independent equations for ρ, φ and z were obtained.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 179–187, 2008.
© Springer-Verlag Berlin Heidelberg 2008

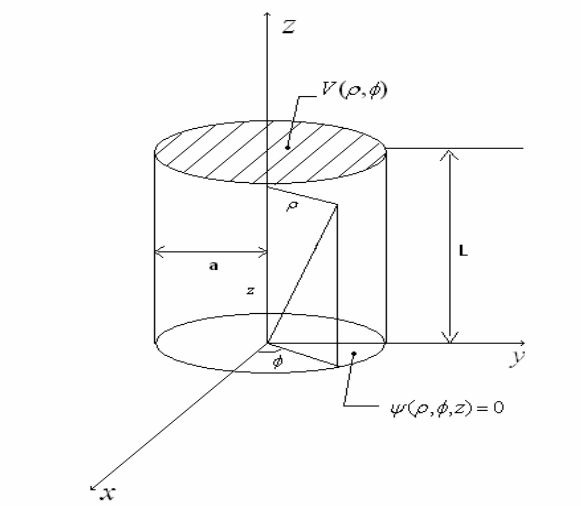


Fig. 1. A Cylindrical shaped domain with the upper edge at a specified potential

The generated ordinary differential equations were then solved using mathematical techniques. Bessel equation of the first and second kind was exhaustively used for generating the solution of the ordinary differential equations so obtained. The solutions were analyzed by framing the required Dirichlet and Neumann boundary conditions. The simplified equation so generated gave us the nature of the conservative field function within the specified domain. The nature of the equipotential lines of the electric field was probed further through numerous simulations.

2 Solution to the Boundary Value Problem

For the solution inside the cylinder with prescribed value of potential on its surface, we consider a conducting cylinder of radius a and height z. The partial differential equations of mathematical physics are often conveniently solved by the method of separation of variables.We now consider the solution by separation of variables of the three-dimensional Laplace equation in cylindrical polar co-ordinates.Referring to Fig. 1, considering the general cylindrical coordinate system, ρ, Φ and z, as shown in the figure, we can write the scalar Laplacian for the potential $\psi(\rho,\phi,z)$ as

$$\frac{\partial^2\psi}{\partial\rho^2}+\frac{1}{\rho}\frac{\delta\psi}{\delta\rho}+\frac{1}{\rho^2}\frac{\partial^2\psi}{\partial\phi^2}+\frac{\partial^2\psi}{\partial z^2}=0 \tag{1}$$

Now using separation of variables, we can write:

$$\psi(\rho,\phi,z)=R(\rho)Q(\phi)Z(z) \tag{2}$$

Substituting equation (2) in the Laplacian for cylindrical co-ordinate space

$$\frac{1}{R}\frac{d^2R}{d\rho^2}+\frac{1}{R\rho}\frac{dR}{d\rho}+\frac{1}{Q\rho^2}\frac{d^2Q}{d\phi^2}+\frac{1}{Z}\frac{d^2Z}{dz^2}=0 \quad (3)$$

Using standard computation approaches, we may write the solution for each independent variable as

$$Z(z)=e^{\pm qz},\ Q(\phi)=e^{\pm iv\phi},\ R(\rho)=CJ_v(q\rho)+DJ_{-v}(q\rho) \quad (4)$$

Here we introduce $-\frac{1}{Q}\frac{d^2Q}{d\phi^2}=v^2$

In the Bessel solution shown above, J_v and J_{-v} are given by:

$$J_v(x)=\left(\frac{x}{2}\right)^v\sum_{j=0}^{\infty}\frac{(-1)^j}{j!\Gamma(j+v+1)}\left(\frac{x}{2}\right)^{2j}\quad J_{-v}(x)=\left(\frac{x}{2}\right)^{-v}\sum_{j=0}^{\infty}\frac{(-1)^j}{j!\Gamma(j-v+1)}\left(\frac{x}{2}\right)^{2j}$$

These solutions are called Bessel Functions of the First kind and the series converges for all finite values of x where v may or may not be an integer. If v is an integer, it is required to introduce another linearly independent solution, the Neuman function, given by

$$\Upsilon_v(x)=\frac{1}{\sin v\pi}\left[\cos(v\pi)J_v(x)-J_{-v}(x)\right]$$

For the given situation, x has been replaced by $q\rho$ in the statements that follow.

In order that the potential be single-valued inside the cylinder, we define the Dirichlet boundary condition for this present problem as

(i) For ρ =0, the potential must be finite

(ii) For ρ =a, the potential must tend to zero

(iii) At z=L, $\psi(\rho,\phi,z)=V(\rho,\Phi)$

Since the potential is finite and real valued at ρ =0, we set D=0, so that $R(\rho)$ is well-defined as, $J_{-v}(q\rho)\big|_{\rho=0}=\infty$. As long as this condition is valid, we may redefine the solution for $R(\rho)$ as $R(\rho)=CJ_v(q\rho)$. For x>>1, the asymptotic form of Bessel functions can be written as,

$$J_m(x)=\sqrt{\frac{2}{\pi x}}\cos\left(x-\frac{m\pi}{2}-\frac{\pi}{4}\right)$$

Without any loss of any generality we may write

$$J_m(qa)=\sqrt{\frac{2}{\pi qa}}\cos\left(qa-\frac{m\pi}{2}-\frac{\pi}{4}\right)$$

The requirement that the potential vanish at ρ =a, means that $q_{m,n}$ (see below) can take on only the specified values as $J_m(qa)\to 0$. For higher roots, the asymptotic formula is

$q = \frac{1}{a}\left[\left(\frac{m}{2}+n\right)\pi - \frac{\pi}{4}\right]$ Here n=1,2,3.....∞, since, for n=0, $J_m(qa) \neq 0$ and this does not satisfy the boundary condition. Combining all the above conditions and replacing the solutions of $R(\rho), Q(\phi)$ and $Z(z)$ in (2), and (1) we have

$$\psi(\rho,\phi,z) = \sum_{m=0}^{\infty}\sum_{n=1}^{\infty} J_m(q_{m,n}\rho)\sinh(q_{m,n}z)(A_{m,n}\sin m\phi + B_{m,n}\cos m\phi) \quad (5)$$

Now, there is one final boundary condition that must be realized. We substitute at z=L in eq (5).Substituting z=L in, we obtain:

$$\psi(\rho,\phi,L) = V(\rho,\phi) = \sum_{m=0}^{\infty}\sum_{n=1}^{\infty} J_m(q_{m,n}\rho)\sinh(q_{m,n}L)\left(A_{m,n}\sin m\phi + B_{m,n}\cos m\phi\right) \quad (6)$$

Taking the Fourier Sine Transform and simplifying in the above equation, we obtain

$$\int_0^{2\pi} V(\rho,\phi)\sin(l\phi)d\phi = \sum_m\sum_n \sinh(q_{m,n}L)J_m(q_{m,n}\rho)\left[A_{m,n}\int_0^{2\pi}\sin(m\phi)\sin(l\phi)d\phi + B_{m,n}\int_0^{2\pi}\cos(m\phi)\sin(l\phi)d\phi\right] \quad (7)$$

Simplifying the above equation and performing the integrations,

$$\int_0^{2\pi} V(\rho,\phi)\sin(l\phi)d\phi = \sum_m\sum_n \sinh(q_{m,n}L)J_m(q_{m,n}\rho)A_{m,n}\pi\delta_{m,l}$$

where $\delta_{m,l}$ is the Kronecker Delta.Since $\pi\delta_{m,l}$ blows up for $m \neq l$, we may write

$$\int_0^{2\pi} V(\rho,\phi)\sin(l\phi)d\phi = \pi\sum_n \sinh(q_{l,n}L)J_l(q_{l,n}\rho)A_{l,n} \quad (8)$$

because $\delta_{m,l} = 1$ for $m = l$. Equation (8) is a Fourier series in φ and a Fourier-Bessel series in ρ. Integrating within the limits $0 < \rho < a$, we obtain

$$A_{m,n} = \frac{2}{\pi a^2}\frac{\cos ech(q_{m,n}L)}{J_{m+1}^2(q_{m,n}a)}\int_0^{2\pi}\int_0^{a} V(\rho,\phi)\sin(l\phi)J_m(q_{m,n}\rho)\rho\, d\rho\, d\phi$$

And similarly,

$$B_{m,n} = \frac{2}{\pi a^2}\frac{\cos ech(q_{m,n}L)}{J_{m+1}^2(q_{m,n}a)}\int_0^{2\pi}\int_0^{a} V(\rho,\phi)\cos(l\phi)J_m(q_{m,n}\rho)\rho\, d\rho\, d\phi \quad (9)$$

Now that we have computed the constants, we can write the final solution as,

$$\psi(\rho,\phi,z) = \sum_{m=0}^{\infty}\sum_{n=1}^{\infty} J_m(q_{m,n}\rho)\sinh(q_{m,n}z)\left(\left[\frac{2}{\pi a^2}\frac{\cos ech(q_{m,n}L)}{J_{m+1}^2(q_{m,n}a)}\int_0^{2\pi}\int_0^{a} V(\rho,\phi)\sin(l\phi)J_m(q_{m,n}\rho)\rho d\rho d\phi\right]\sin m\phi + \left[\frac{2}{\pi a^2}\frac{\cos ech(q_{m,n}L)}{J_{m+1}^2(q_{m,n}a)}\int_0^{2\pi}\int_0^{a} V(\rho,\phi)\cos(l\phi)J_m(q_{m,n}\rho)\rho d\rho d\phi\right]\cos m\phi\right)$$

The particular form of expansion is governed by the requirement that the potential vanish at z=0 for arbitrary ρ and ρ =a for arbitrary z. For different boundary conditions the expansion would take a different form.

3 Simulation and Analysis Results

For the situation depicted in the problem statement the behavior of the conservative field function was simulated for finite values of applied potential on the top surface of the cylinder. The simulations were henceforth done in Matlab through the Partial Differential Equation toolbox. The Laplacian was extensively simulated in the cylindrical domain and the nature of the equipotential lines was analyzed. A constant dc 10 volt was applied to a 100 mm long cylinder, and the situation was simulated. The solution to the elliptic partial differential equation generated shows a peculiar tendency of the potential. For finite values of applied potential on one of the edges, the potential starts increasing exponentially for increasing height for constant radial distances. As the distance from the central axis increases, the potential decreases as we approach the boundary, which is because of the boundary condition imposed by us. A similar behavior can be seen along the negative axes on the cylindrical domain. That is, along the left half of the cylinder a similar thing is seen to happen. In simple terms we may say that the behavior of the potential on the two halves is somewhat like mirror images. The contours are prominent for the upper half of the cylinder and we see a remarkable distinction between the high and low potential regions as we approach the upper boundary of the cylindrical domain.

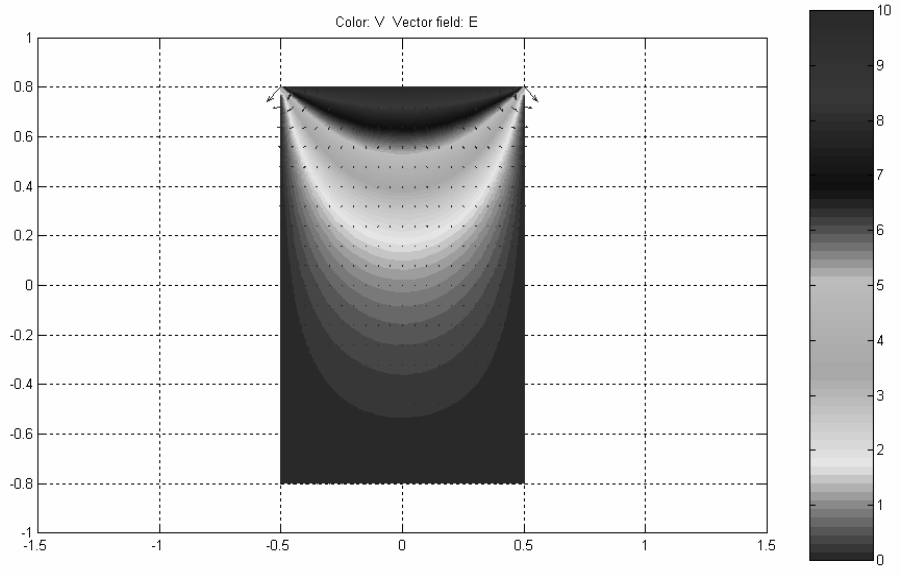


Fig. 2. Simulation of the nature of equipotential lines of electric field inside the Cylinder

We see that there isn't much happening in the lower part of the cylinder. The region below the lowest contour shown is all below 100 volts. The potential is poorly behaved in the corners where the equipotentials meet. We have to take the blame for that, because we imposed a boundary condition which is discontinuous at the corner. In order to test the behavior of the potential in practical situations we numerically solved the final equation obtained using an algorithm developed by us. The algorithm was implemented in a standard programming platform and the values of the potential were generated for various combinations of the radial distance and the height. We examined the behavior by obtaining plots of the potential as a function of the height for certain values of the radial distance and also for the behavior as a function of the increasing radial distance for constant values of the separation from the upper edge. The behavior is perfectly in agreement with the simulation results, as shown above. The results have been analyzed in brief in the sections that follow. Stress has been laid on the behavior of the solutions due to the inadequacy of the Fourier-Bessel series to give accurate, converging results. The plots shown below depict the nature of the potential as a function of the radial distance from the central axis, for finite values of the distance from the top surface. The nature of the plot is similar in nature to the one obtained by us in the Matlab Simulation results. The lower regimes show almost negligible penetration of the electric potential. Also noteworthy is the nature of the curves, where for each value of the longitudinal separation, the potential shows a gradual decrease for increasing values of the radial distance. Here we may draw a similarity between the plot obtained by us and the one obtained in the Simulation results. If one observes closely, it will be noticed that at the centre region of the cylindrical volume, the strength of the electric potential is very high and well behaved at the top surfaces, but as one goes toward the boundary, the strength of the electric potential decreases steadily until finally tending to zero at the side walls.

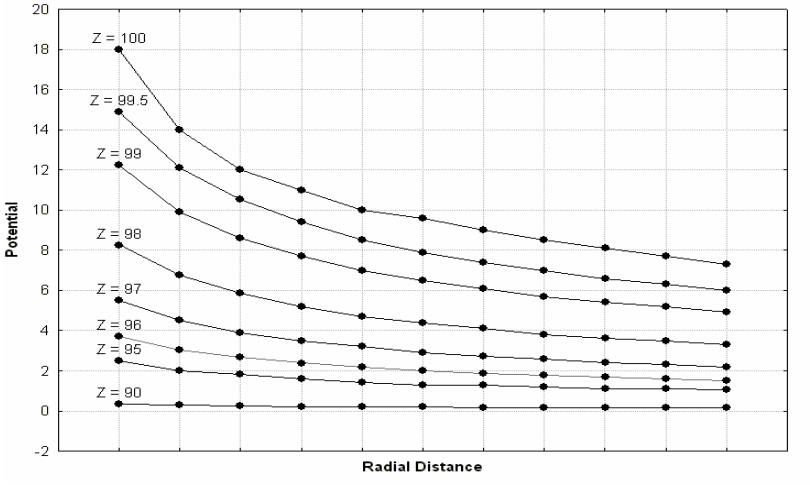


Fig. 3. Plot of the Potential vs the Radial Distance ρ for finite values of the distance from the upper edge of the Cylinder

The nature can be well understood from the regime where the field penetration is of average levels. One can easily notice that the potential starts falling as one moves away from the central axis, until finally becoming zero, as per our boundary condition. We also examined the nature of the potential for increasing height for constant values of separation from the central axis.

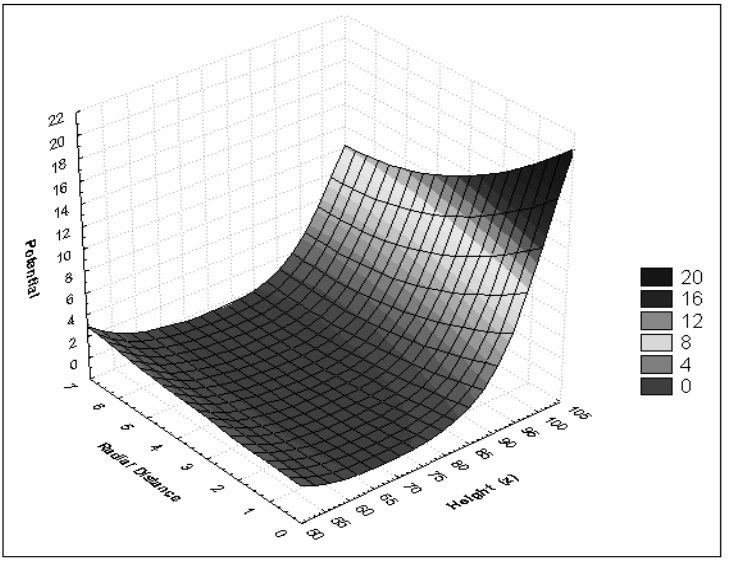


Fig. 4. Exponentially increasing values of the potential are plotted here as a function of the radial distance and the height**.** For the lower contours the potential is poorly behaved and starts increasing for increasing values of the height.

From the nature of the plots shown below, for the behaviour of the potential for increasing distance from the grounded base, for constant values of ρ , we may conclude that the potential is exponentially divergent with increasing values of z**.** From the 3 dimension overview shown above we see that our results are indeed similar to the simulation results, as can be seen above. The potential steadily increases for increasing height and attains its maximum value of that of the applied potential. We also confirm our findings in the simulation result by the nature of the plots obtained by us.

The results lead us to conclude that the situation is quite unlike the behaviour in actual practical observations. From the analysis point of view we may also note that in formulating the behavior, the solution takes only certain values for which the problem is consistent. For every other value the problem gives unrealistic results. Also, we must note that at certain instances the solution gives values of the potential which exceeds the maximum value of the applied voltage, which though are within acceptable limits and do not result in significant divergence from the true nature. This is due to the fact that while generating the coordinates for graphical analysis, certain assumptions have been taken, and the terms in the series solutions have been assumed only unto a certain order. Moreover, the integrations over the cylindrical boundary

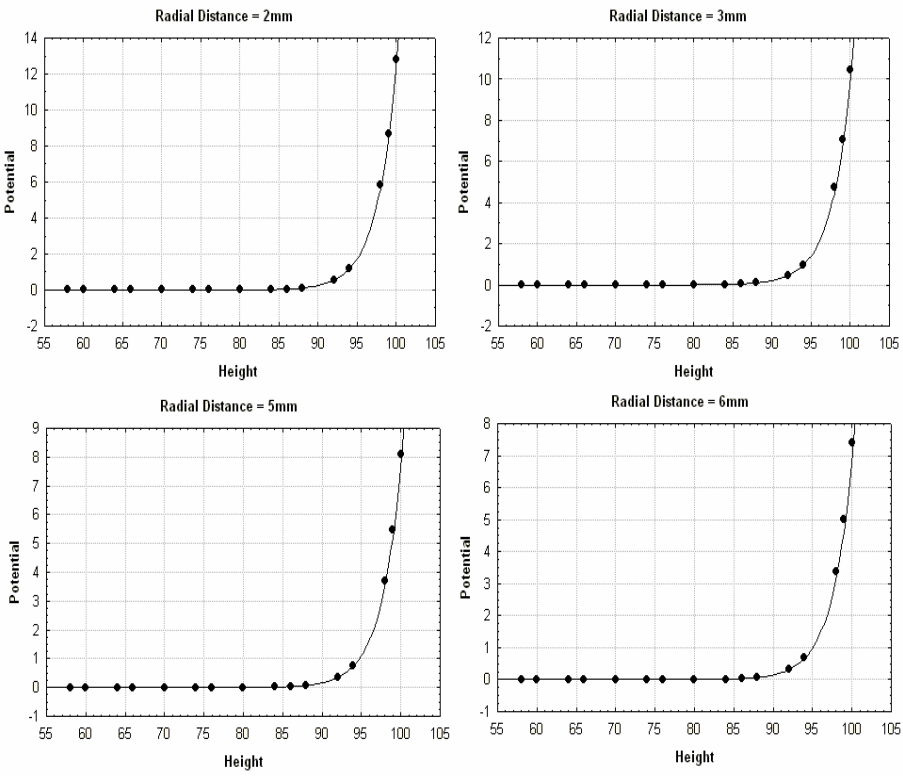


Fig. 5. Plot of the Potential as a function of the Height for Radial Distance, ρ =2, 3, 5 and 6mm. We notice he decrease in potential for increasing radial distance, from the nature of the plots.

were done numerically and certain assumptions have also been made there. That is for the series solutions involving m and n; we only took these values unto a certain limit for certain restrictions on the programming platform. But the nature of the solution may well be extended to realizable situations and other complex and mixed boundary value problems as well. But the divergent nature of the potential inside the cylinder needs to be probed further, and possibly ample scope remains for others to develop reasons for the diverging nature.

4 Conclusion

For elliptic partial differential equations involved in the Laplacian in strict boundary value problems, the series solutions were obtained and analyzed numerically. The consistency of the solutions was examined for a cylindrical domain and the behavior of the potential function was simulated. Certain inconsistencies that may require a detailed study is the diverging nature of the potential as we approach the boundary. Also associated with the numerical solution are certain assumptions that one must

probe into while solving the problem numerically, including the series expansions of the Fourier and Fourier-Bessel functions. The non convergent nature of the potential is very much intriguing and poses several questions that must be looked into in more detail. Here, we attribute this non convergent nature to the inherent drawbacks of the methods used to generate the solution to the problem. That is, the approximations that came into consideration while taking the series expansions and those in the numerical solutions to the integrals. A more detailed study on these aspects may lead to more intriguing behavior with possible reasons and inferences.

Acknowledgements

The authors would like to thank Dr.A.K.Ganguly (Department of Engineering Science, Haldia Institute of Technology), for the help provided by him in computation methods and theoretical discussions. The authors are also grateful to Dr.K.Ghosh (Department of Engineering Science, Haldia Institute of Technology) and Prof.M.S Saroa (Department of Engineering Science, Haldia Institute of Technology) for fruitful consultations. The authors would like to thank all those who have constantly reviewed this paper and made the necessary changes.

References

[1] Jackson, J.D.: Classical Electrodynamics, 3rd edn. Wiley, New York (1975)
[2] Watson, G.N.: A Treatise on the Theory of Bessel Functions. Cambridge University Press, Cambridge (1944)
[3] Lighthill, M.J.: Fourier analysis and Generalized Functions. Cambridge University Press, Cambridge (1975)
[4] Partial Differential Equation Toolbox Users Guide, The MathWorks, `http://www.mathworks.com/access/helpdesk/help/toolbox/pde/pde.shtml`

A Method and Its Implementation for Constructing Bäcklund Transformations to Nonlinear Evolution Equations

Zhibin Li[1,2], Yinping Liu[2], and Haifeng Qian[2]

[1] Institute of Theoretical Computing,
[2] Department of Computer Science
East China Normal University, Shanghai 200062, China

Abstract. An algorithmic method to construct a kind of auto Bäcklund transformations (BTs) is proposed. A Maple package `AutoBT`, which can entirely automatically generate auto BT is presented. `AutoBT` has been effectively applied to many nonlinear evolution equations with physical significance. Not only are previously known BT recovered but also in some cases new and more general form of BT are obtained.

1 Introduction

Nonlinear evolution equations are important mathematical models to describe physical phenomena. They are also an important field in the contemporary study of nonlinear physics, especially in soliton theory. The research on the explicit solution and integrability is helpful in clarifying the movement of matter under nonlinear interactivity and plays an important role in scientifically explaining the corresponding physical phenomena.

The Bäcklund transformation (BT), originated in the study of surfaces of constant negative curvature, is such a system of equations, relating the solution of a given equation either to another solution of the same equation or to a solution of other equation. The former is called auto BT. Generally, the Bäcklund transformations of nonlinear partial differential equations(PDEs) plays an important role in soliton theory. They are used to construct an infinite number of conserved quantities and to provide exact solutions for nonlinear PDEs. In particular, the nonlinear iterative principle from BT, converts the problem of solving nonlinear PDEs to that of purely algebraic calculations.

It is difficult to find the Bäcklund transformations for a given nonlinear PDE. Various methods have been developed for different classes of equations, such as, the Painlevé analysis method[1-3], the homogeneous balance method[4,5], the Hirota method[6-8] and variational method[9-11]. Correspondingly, Bäcklund transformations can be shown in different forms. To our best knowledge, many nonlinear evolution equations have a form of

$$E(u,\, u_t,\, u_x,\, u_{xx},\, \cdots) = 0 \tag{1}$$

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 188–198, 2008.
© Springer-Verlag Berlin Heidelberg 2008

that admit auto BTs with a form

$$(u+av)_t = P(u,\, v,\, u_x,\, v_x, \cdots, u_{nx}, v_{nx}), \quad (cu+v)_x = Q(u+av), \tag{2}$$

where a and c are constants, P and Q are functions in their variables, respectively. If u is a solution of equation (1) and v satisfies the transformation (2), then v is also a solution of equation (1).

For example, by transformation $w = u_x$, the KdV equation

$$w_t + 6\, w\, w_x + w_{xxx} = 0 \tag{3}$$

can be rewritten as

$$u_{xt} + \left(3(u_x)^2 + u_{xxx}\right)_x = 0. \tag{4}$$

An auto BT of (4) has been found as

$$(u-v)_t = (v-u)_{xxx} + 3((v_x)^2 - (u_x)^2), \quad (u+v)_x = -\frac{1}{2}(u-v)^2 + \beta, \tag{5}$$

where β is a constant. If u is a solution of the KdV equation (4), the BT (5) shows that a second solution v of KdV equation (4) may be constructed by integration of the pair of first-order equations.

Another example, known one hundred years ago, is the sine-Gordon equation

$$u_{xt} = \sin u, \tag{6}$$

which admits an auto BT

$$(u+v)_t = \frac{2}{d}\sin\frac{v-u}{2}, \quad (v-u)_x = 2\, d \sin\frac{u+v}{2}, \tag{7}$$

where d is an arbitrary parameter. From Bäcklund transformation (7), a relation may be derived:

$$U = u + 4\tan^{-1}\left[\frac{d_2-d_1}{d_2-d_1}\tan\left(\frac{u_2-u_1}{4}\right)\right]. \tag{8}$$

The relation (8) represents a nonlinear iterative principle which acts on the solution set $\{u, u_1, u_2\}$ to produce a new solution U.

Other nonlinear PDEs, such as Liouville equation, mKdV equation, Gardner equation, generalized KdV and the fifth order equations of the KdV hierarchies, and so on, admit auto BT in the form (2). In this paper, a method to construct BT in the form (2) is proposed. A Maple package for delivering the auto BT entirely automatically is presented.

The paper is organized as follows. In Section 2, the algorithm is introduced. The implementation of the algorithm is described in Section 3. In the last section, several examples are given to demonstrate the effectiveness of the package.

2 An Algorithm for Bäcklund Transformation

It is well known that most nonlinear evolution equations can be converted into the form

$$u_{xt} = H(u, u_x, u_{xx}, \cdots, u_{nx}), \tag{9}$$

where u_{nx} means the nth-derivative of u with respect to x, H is a function of u and its derivatives. For example, both the KdV equation (4) and the sine-Gordon equation (6) are in the form (9). Similarly, by a transformation the Burgers equation

$$w_t + ww_x - pw_{xx} = 0 \tag{10}$$

can be converted into the form (9)

$$u_{xt} = (pu_{xx} - \frac{1}{2}(u_x)^2)_x. \tag{11}$$

For equation (9), suppose that it admits auto BT in the form (2). We consider two special cases.

2.1 The First Case: Differential Form

We assume that H in (9) can be written as

$$H = F(u_x, u_{xx}, \cdots, u_{(n-1)x})_x, \tag{12}$$

where F is a polynomial in its variables. In order to construct auto BT in the form (2), we first duplicate (9) with another variable v

$$u_{xt} = F(u_x, u_{xx}, \cdots, u_{(n-1)x})_x, \tag{13}$$

$$v_{xt} = F(v_x, v_{xx}, \cdots, v_{(n-1)x})_x. \tag{14}$$

Then we construct the auto BT by following steps:

Step 1: Multiply both sides of (14) by a, add the obtained equation to (13), we have

$$(u + a\,v)_{xt} = (F(u_x, u_{xx}, \cdots, u_{(n-1)x}) + a\,F(v_x, v_{xx}, \cdots, v_{(n-1)x}))_x, \tag{15}$$

integrate (15) with respect to x, which gives the t-part of the autoBT:

$$(u + a\,v)_t = F(u_x, u_{xx}, \cdots, u_{(n-1)x}) + a\,F(v_x, v_{xx}, \cdots, v_{(n-1)x}). \tag{16}$$

Step 2: To get the x-part of the autoBT, we first multiply both sides of (13) by c and add it to (14), we have

$$(c\,u + v)_{xt} = (c\,F(u_x, u_{xx}, \cdots, u_{(n-1)x}) + F(v_x, v_{xx}, \cdots, v_{(n-1)x}))_x. \tag{17}$$

Then differentiate both sides of the second equation in (2) with respect to t, we have

$$(c\,u + v)_{xt} = (u + a\,v)_t\,Q'(u + a\,v). \tag{18}$$

Substituting (16) into (18), we obtain

$$(c\,u+v)_{xt}=[F(u_x,u_{xx},\cdots,u_{(n-1)x})+a\,F(v_x,v_{xx},\cdots,v_{(n-1)x})]\,Q'(u+a\,v). \tag{19}$$

Step 3: The main task in this step is to compute $Q(u+a\,v)$ by combining (17) with (19). For convenience, a transformation is introduced as follows

$$U=u+a\,v,\quad V=c\,u+v,\qquad (ac\neq 1), \tag{20}$$

in this way, we have

$$u=\frac{a\,V-U}{a\,c-1},\qquad\qquad v=\frac{c\,U-V}{a\,c-1}. \tag{21}$$

By using (21), we can reduce (19) as

$$V_{xt}=G_1(U_x,V_x,\cdots,U_{(n-1)x},V_{(n-1)x})Q'(U), \tag{22}$$

and (17) can be expressed as

$$V_{xt}=G_2(U_x,V_x,\cdots,U_{nx},V_{nx}). \tag{23}$$

Eliminating V_{xt} in (22) and (23) leads to

$$G_1(U_x,V_x,\cdots,U_{(n-1)x},V_{(n-1)x})Q'(U)-G_2(U_x,V_x,\cdots,U_{nx},V_{nx})=0. \tag{24}$$

From the assumption (2), we know that

$$V_x=Q,\quad V_{xx}=Q'U_x,\quad V_{3x}=Q''(U_x)^2+Q'U_{xx},\quad \cdots. \tag{25}$$

Substituting (25) into (24), and combining the same power with respect to U and its derivatives, we get

$$\sum_{i_0,i_1,\cdots,i_n} T_{i_0,i_1,\cdots,i_n}(Q,Q',\cdots)U^{i_0}(U_x)^{i_1}\cdots(U_{nx})^{i_n}=0. \tag{26}$$

Let the coefficients of the same powers of U and its derivatives be zero. We obtain a differential system for Q as well as a,c and parameters appeared in the original equation as follows

$$T_{i_0,i_1,\cdots,i_n}(Q,Q',\cdots,Q^{(n)})=0, \tag{27}$$

If system (27) possesses a solution $Q(U)$ then the Bäcklund relation with the form (2) can be established. If it is inconsistent then we conclude only, that the possible Bäcklund transformations of (9) are not in the form of (2).

We illustrate the algorithm by using it to establish the auto BT of KdV equation (4). From **Step 1**, the t-part of an auto BT relation is established as

$$(u+a\,v)_t=-(a\,v+u)_{xxx}-3(a\,(v_x)^2+(u_x)^2), \tag{28}$$

in which a will be determined later.

From **Step 2** and **Step 3**, we obtain a system for $Q(U)$

$$\begin{cases} Q(U)\,(6\,c+6\,a\,c)=0, \\ 6\,c+6\,c^2-3Q''(U)\,(2\,a\,c-c^2\,a^2-1)=0, \\ Q'''(U)\,(2\,a\,c+c^2\,a^2+1)=0, \\ Q'(U)\,Q(U)^2\,(3\,a^2+3\,a)=0, \\ Q'(U)\,Q(U)\,(6+6\,c\,a^2+6\,a\,c+6\,a)=0, \\ Q'(U)\,(3+3\,a\,c^2+6\,c+6\,a\,c)=0. \end{cases} \tag{29}$$

Solving the system (29), it follows

$$Q(U)=-\frac{1}{2}U^2+c_1U+c_2, \quad a=-1, c=1, \tag{30}$$

where c_1, c_2 are integral constants. Therefore, the x-part of auto BT is established.

Thus, an auto BT which is more general than (5) is found

$$(u-v)_t=(v-u)_{xxx}+3((v_x)^2-(u_x)^2),\ (u+v)_x=-\frac{1}{2}(u-v)^2+c_1(u-v)+c_2. \tag{31}$$

The algorithm in the first case stops here.

2.2 The Second Case: Differential-Free Form

Suppose that H in (9) is just a function of u, *i.e.* $H=F(u)$. In this case, we suggest that the t-part of Bäcklund transformation (2) is in simple form

$$(u+av)_t=P(cu+v), \tag{32}$$

and we proceed as follows:

Step 1: Similar to with the case 1, we have

$$U_t=P(V), \quad V_x=Q(U) \tag{33}$$

from (21) and transformation (2). Differentiating both sides of the first equation in (33) with respect to x and we have

$$U_{xt}=P'(V)\,V_x, \tag{34}$$

similarly, from the second equation in (33) we obtain

$$V_{xt}=Q'(U)U_t. \tag{35}$$

Substituting (33) into (34) and (35) leads to

$$U_{xt}=P'(V)\,Q(U), \quad V_{xt}=Q'(U)P(V). \tag{36}$$

Step 2: By combining the original equation with (36), we get a coupled first-order differential system

$$P'(V)Q(U)=F(u)+aF(v), \quad Q'(U)P(V)=cF(u)+F(v). \tag{37}$$

To keep the consistency of the coupled system, we have

$$\int [F(u) + aF(v)]dV = \int [cF(u) + F(v)]dU, \tag{38}$$

from which, the relation for a, c is determined, for example, $a = -\frac{1}{c}$. If $c = 1$, then $a = -1$.

Step 3: In this way the coupled system (37) is simplified as

$$P(V)Q(U) = \int [F(u) + aF(v)]dV, \tag{39}$$

If one can find functions $P(V)$ and $Q(U)$ satisfying the final function equation (39), then an auto BT in the form (33) can be established. Otherwise, the considered equation may have no auto BTs in form (33). Since solving the coupled equation (39) is very difficult, some times we are powerless in facing it.

We illustrate this algorithm by considering the sine-Gordon equation (6). From **Step 1** and **Step 2**, we obtain the coupled differential system

$$P'(V)\,Q(U) = \sin(u) + a\,\sin(v), \quad Q'(U)\,P(V) = c\,\sin(u) + \sin(v). \tag{40}$$

The compatibility condition of system (40) implies

$$c\,[a^2\cos(v) - \cos(u)] = a\,[c^2\cos(u) - \cos(v)]. \tag{41}$$

From the equation (41) for variable a, c, we have $a = -\frac{1}{c}$. Taking $c = 1$, then $a = -1$, and we obtain the final function equation

$$P(V)Q(U) = -2\,[\cos(\frac{U+V}{2}) - \cos(\frac{U-V}{2})]. \tag{42}$$

Fortunately, we can solve equation (42) and obtain P and Q. Return to the original variables, the auto BT relation (7) is established.

3 The Maple Package AutoBT

The method described in Section 2, while relatively simple in principle, can be very tedious in practice by hand. We have developed a package `AutoBT` written in *Maple 10* that fully automates the method and delivers possible BT relations as output. Further, possible parameters constraints can be discovered using the package.

As an example of the use of `AutoBT`, we consider the Gardner equation[12]

$$w_t - 6ww_x + w_{xxx} - 12qw^2w_x = 0, \tag{43}$$

where q is a parameter. To find the auto BT of (43) one proceeds as follows:

```
>eq:=diff(w(x,t),t)-6*w(x,t)*diff(w(x,t),x)+diff(w(x,t),x$3)-12*q*
w(x,t)^2*diff(w(x,t),x);
>AutoBT(eq);
```

`AutoBT` returns the result immediately as follows:
The input EQ is:

$$\frac{\partial}{\partial t}\, w - 6\, w\, \frac{\partial}{\partial x}\, w + \frac{\partial^3}{\partial x^3}\, w - 12 q\, w^2\, \frac{\partial}{\partial x}\, w$$

The input EQ in u reads:

$$\frac{\partial}{\partial t}\,(\frac{\partial}{\partial x}\, u) + \frac{\partial}{\partial x}[-4\, q\,(\frac{\partial}{\partial x} u)^3 - 3\,(\frac{\partial}{\partial x} u)^2 + \frac{\partial^3}{\partial x^3} u]$$

where

$$w = \frac{\partial}{\partial x}\, u$$

Two groups of relations between the functions u and v can be found

$$\frac{\partial}{\partial t}(u - v) = 3\,[(\frac{\partial}{\partial x} u)^2 - (\frac{\partial}{\partial x} v)^2] + \frac{\partial^3}{\partial x^3}(u - v),$$

$$\frac{\partial}{\partial x}\,(u + v) = \frac{1}{2}(u - v)^2 + c_1 (u - v) + c_2$$

with the parameters constraints: $q = 0$,

$$\frac{\partial}{\partial t}(u - v) = 4\, q\,[(\frac{\partial}{\partial x} u)^3 - (\frac{\partial}{\partial x} v)^3] + 3\,[(\frac{\partial}{\partial x} u)^2 - (\frac{\partial}{\partial x} v)^2] + \frac{\partial^3}{\partial x^3}(u - v),$$

$$\frac{\partial}{\partial x}\,(u + v) = c_1\, e^{\sqrt{2\,q}(u - v)} + c_2\, e^{-\sqrt{2\,q}(u - v)} - \frac{1}{2\,q}$$

with the parameters constraints: $q \neq 0$.

From this example, we can see that once the given equation is input, `AutoBT` will generate automatically the possible auto BT relations as well as possible parameters constraints. It is to be stressed that when $q = 0$, the Gardner equation (43) is reduced to KdV equation. Therefore, the first one above is no other than the auto BT of KdV equation.

The Maple package `AutoBT` is comprised of the main function **AutoBT()** and six other functions **new-int()**, **transform-eq()**, **built-tBT()**, **get-eqsQ()**, **solve-eqsQ()**, **final-BT()**. We outline each one as follows.

new-int(): As integral function **int** in Maple is very weak for figuring out results. In most cases, it is merely expressed by integral sign. So we recoded a function **new-int**, based on the basic rules of computing integral. This function greatly improve the computing capability of integral.

transform-eq(): Suppose the input equation is in function w, we first convert it in u by a transformation $w = u_x$ or $w = u$ according to whether there exists u_t or u_{xt} in input equation. Then we further introduce transformation $u + a\, v = U$, $c\, u + v = V$. Thus the obtained equations in u, v can be converted into U, V, and denote the equations in U, V as eqsUV.

built-tBT(): Assume the required auto BT as (2). If $H = F_x$ in (9), the function **built-tBT** builds the t-part of BT relation in U, V, and denote it as BTtUV.

get-eqsQ(): In this procedure, we first convert (2) into U, V, the obtained results are denoted by $\mathcal{BTUV}$. Then two branches are considered. In the first case, we differentiate the second equation in $\mathcal{BTUV}$ with respect to t, and denote the results as BT2t. By comparing equations eqsUV with BT2t, and also making use of $\mathcal{BTUV}$, an equation in Q, U, V and their derivatives is built, which is denoted as eqQUV. Then starting from $\mathcal{BTUV}$ we define $V_{xx}, V_{xxx}, \cdots, V_{nx}$, and substitute them into eqQUV. The obtained equation contains Q, U and their derivatives. Collecting the same powers for terms $U^{i_0}(U_x)^{i_1}\cdots(U_{nx})^{i_n}$ and setting their coefficients to zero. Thus a system for Q, a, c as well as parameters contained in input equations is built, which is denoted as $\mathcal{PS}$. In the second case, differentiating the first and second equation in BTUV with respect to x, t, respectively, the obtained equations are denoted as $eqsxt$. By comparing $eqsxt$ with eqsUV, a system for P, Q is built and denoted as $\mathcal{PS}$.

solve-eqsQ(): Simplify the differential system $\mathcal{PS}$ by using the subpackage **simp**, which is embedded in Maple 10. Then solve the simplified equations by using the command **dsolve** or **solve**. The obtained solutions set is denoted as $\mathcal{SOL}'$.

final-BT(): We first eliminate all trivial solutions from $\mathcal{SOL}'$, and denote the remaining nontrivial solutions set as SOL. For each element of SOL, substitute it into BTUV, and with the relation (20), final auto BT relations are established. At the same time, some corresponding parameters constraints are also generated. Output each auto BT as well as its parameters constraints in given format.

We have computed near 30 nonlinear evolution equations utilizing the package `AutoBT` on our PC. The package is efficient; for most of the equations it can entirely automatically deliver required results within 20 seconds. If no auto BT of the above form exists for an equation, our program will output "Can not find non-trivial auto BT in such form!". Furthermore, it can identify illegal form of input equation and output "The package can only work for EQs, which can be converted in the form $u_{xt} = H(u, u_x, u_{xx}, ...)$".

4 The Application of the Package AutoBT

To illustrate the effectiveness of the package `AutoBT`, some examples are considered in this section

Example 1. Consider the Burgers equation

$$w_t + w\, w_x - p\, w_{xx} = 0, \tag{44}$$

in which p is a parameter. `AutoBT` gives one BT relation, which reads,

$$\begin{cases} (u-v)_t = p\,(u_{2x}-v_{2x}) + \frac{1}{2}({v_x}^2 - {u_x}^2), \\ v_x = c_1 + c_2\, e^{\frac{v-u}{2\,p}}, \end{cases} \tag{45}$$

here $w = u_x$. We notice that parameters $a = -1$, $c = 0$ in this example. To our knowledge, the BT relation (45) are first reported.

Example 2. Consider the mKdV equation

$$w_t + p\,w^2\,w_x + w_{3x} = 0, \tag{46}$$

in which p is a positive parameter. `AutoBT` gives one BT relation as follows,

$$\begin{cases} (u + a\,v)_t = -(a\,v_{3x} + u_{3x}) - \frac{p}{3}(a\,{v_x}^3 + {u_x}^3), \\ (cu+v)_x = c_1\, e^{\frac{\sqrt{p}}{\sqrt{3ac-3}}(u+av)} + c_2\, e^{-\frac{\sqrt{p}}{\sqrt{3ac-3}}(u+av)}, \end{cases} \tag{47}$$

in which c_1, c_2 are integral constants and $w = u_x$. Note that in this example a, c are arbitrary. Taking $p = 6,\ a = -1$, $c = 1$, the auto BT (47) is reduced to the BT relation given in excise 10.10 in [14]. It can be seen that our obtained auto BT (47) is more general than all the known ones.

Example 3. Consider a generalized 5-order nonlinear evolution equation[13]

$$w_t + p\,w\,w_{3x} + q\,w_x\,w_{xx} + r\,w^2\,w_x + w_{5x} = 0, \tag{48}$$

in which p, q, r are parameters. In the literature, with $(p,\ q,\ r) = (30, 60, 270)$ or $(20, 40, 120)$, and or $(10, 20, 30)$, it reduces to standard 5th-order KdV equation; with $(p,\ q,\ r) = (30, 30, 180)$ or $(5, 5, 5)$, and or $(-15, -15, 45)$, it reduces to the Sawada-Kotera equation; and with $(p,\ q,\ r) = (30, 75, 180)$ or $(10, 25, 20)$, it reduces to the Kaup-Kupershmit equation.

For this example, `AutoBT` gives one BT relation, it reads:

$$\begin{cases} (u-v)_t = \frac{q}{2}(v_x v_{3x} - u_x u_{3x}) + \frac{q}{4}({v_{xx}}^2 - {u_{xx}}^2) + (v-u)_{5x} + \frac{r}{3}({v_x}^3 - {u_x}^3), \\ (u+v)_x = -\frac{q}{40}\,(u-v)^2 + c_1\,(u-v) + c_2, \end{cases} \tag{49}$$

with the parameters constraint:

$$p = \frac{q}{2},$$

in which $w = u_x$. It can be seen from the above that the parameters constraints are satisfied only by standard 5th-order KdV equation. So we can conclude that the Sawada-Kotera equation and the Kaup-Kupershmit equation do not possess BT relation in the form (2). Taking $c_1 = 0$, the auto BT (49) is reduced to the BT relation (4.23) and (4.24) in [11]. This shows that our BT (49) is a more general BT relation.

Example 4. Consider Liouville equation[14]

$$w_{xt} = e^w. \tag{50}$$

For this equation, `AutoBT` gives one BT relation, it reads:

$$\begin{cases} (u+v)_t = c_1\,(e^{\frac{u-v}{2}} - e^{-\frac{u-v}{2}}), \\ (u-v)_x = \frac{2}{c_1}\,e^{\frac{u+v}{2}}, \end{cases} \tag{51}$$

in which c_1 is an integral constant and $w = u$. The auto BT (51) is no other than the BT relation (10.1.14) in [14].

Example 5. Consider the Vakhnenko equation[15]

$$(1 + w_t)w_x + w_{xxt} = 0. \tag{52}$$

For this equation, `AutoBT` can not generate its BT relations, and it just outputs prompt information "The package can only work for EQs, which can be converted in the form $u_{xt} = H(u, u_x, u_{xx}, ...)$".

5 Summary

Bäcklund transformation is an effective method used in the search of exact solutions of nonlinear differential equations, and various methods have been developed to construct BT relations. An algorithmic method to construct a kind of BT relations is presented and implemented in Maple, in which the package `AutoBT` can entirely automatically deliver this kind of BT relations as well as possible parameters constraints. The package `AutoBT` has been effectively applied to many nonlinear PDEs. Not only are previously known BT relations recovered, but also some new or more general BTs are obtained. However, due to the difficulty of solving differential system, sometimes our package may not be strong enough to deal with all equations with the form (9). We will further improve it to deal with more equations with the development of both computer algebra and Maple.

Acknowledgment

This work was supported by National Key Basic Research Project of China (2004CB318000) and by Shanghai Leading Academic Discipline Projec, Project Number: B412.

References

1. Steeb, W.H., Grauel, A., Kloke, M., Spieker, B.M.: Nonlinear diffusion equations, integrability and the Painleve property. Phys. Scripta. 31, 5 (1985)
2. Xu, G.Q., Li, Z.B.: A maple package for the Painleve test of nonlinear partial differential equations. Chin. Phys. Lett. 20, 975 (2003)
3. Lou, S.Y.: Painleve test for the integrable dispersive long waves. Phys. Lett. A 176, 96 (1993)

4. Wang, M.L., Zhou, Y.B., Li, Z.B.: Application of a homogeneous balance method to exact solutions of nonlinear equations in mathematical physics. Phys. Lett. A 216, 67 (1996)
5. Fan, E.G.: Two new applications of the homogeneous balance method. Phys. Lett. A 265, 353 (2000)
6. Hirota, R.: The Direct Method in Soliton Theory. Cambridge University Press, Cambridge (2004)
7. Ma, W.X., Geng, X.G.: Bäcklund transformations of soliton systems from symmetry constraints. In: Bäcklund and Darboux Transformations. The Geometry of Solitons. CRM Proceedings & Lecture Notes, vol. 29. American Mathematics Society (2001)
8. Lin, J., Lou, S.Y.: Multisoliton solutions of the (3+1)-dimensional Nizhnik-Novikov-Veselov equation. Comm. Theor. Phys. 37, 265–268 (2002)
9. Van de Leur, J.: Bäcklund transformations for new integrable hierarchies related to the polynomial Lie algebra. J. Geo. Phys. 57, 435–447 (2007)
10. Sokalski, K., Wietecha, T., Lisowski, Z.: Variational approach to the Bäcklund transformations. Acta Physica Polonica B 32, 17–28 (2001)
11. Sokalski, K., Wietecha, T., Sokalska, D.: Existence of dual equations by means of strong necessary conditions - Analysis of integrability of partial differential nonlinear equations. J. Non. Math. Phys. 12, 31–52 (2005)
12. Fu, Z.T., Liu, S.K., Liu, S.D.: New kinds of solutions to Gardner equation. Chaos, Solitons and Fractals 20, 301–309 (2004)
13. Kichenassamy, S., Oliver, P.J.: Existence and nonexistence of solitary wave solutions to higher-order model evolution equations. SIAM J. Math. Anal. 23, 1141–1166 (1992)
14. Liu, S.K., Liu, S.D.: Nonlinear Equations in Physics. Peking University Press, Beijing (2000)
15. Morrison, A.J., Parks, E.J.: The N-soliton solution of the modified generalised Vakhnenko equation. Chaos, Solitons and Fractals 16, 13–26 (2003)

On the Invariant Properties of Hyperbolic Bivariate Third-Order Linear Partial Differential Operators

Ekaterina Shemyakova and Franz Winkler

Research Institute for Symbolic Computation (RISC),
J. Kepler University,
Altenbergerstr. 69, A-4040 Linz, Austria
{kath,Franz.Winkler}@risc.uni-linz.ac.at
http://www.risc.uni-linz.ac.at

Abstract. Bivariate, hyperbolic third-order linear partial differential operators under the gauge transformations $L \to g(x,y)^{-1} \circ L \circ g(x,y)$ are considered. The existence of a factorization, the existence of a factorization that extends a given factorization of the symbol of the operator are expressed in terms of the invariants of some known generating set of invariants. The operation of taking the formal adjoint can be also defined for equivalent classes of LPDOs, and explicit formulae defining this operation in the space invariants were obtained.

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 [1,2,3,4,5,6]). Both the property of having a factorization, and the property of having a factorization that extends a certain factorization of the (principal) symbol are invariant under Gauge transformations of LPDOs, *viz.* $L \to g(x,y)^{-1} \circ L \circ g(x,y)$, and therefore can be described invariantly in terms of the invariants of a generating set of invariants, if such a set is known.

The Laplace Transformations Method [7] is an example of the use of an invariant description of factorization properties for a second-order hyperbolic operator. The normalized form of such operators is

$$L = D_x \circ D_y + aD_x + bD_y + c \ , \tag{1}$$

where all the coefficients are functions of x and y, and the *Laplace invariants*

$$h = c - a_x - ab, \quad k = c - b_y - ab \tag{2}$$

form a generating set of invariants with respect to the Gauge transformations. It is easy to see that L is factorable if and only if h or k is zero. Moreover, the

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 199–212, 2008.
© Springer-Verlag Berlin Heidelberg 2008

factorization of the principal symbol $\mathrm{Sym}(L) = X \cdot Y$ can be extended if and only if $h = 0$, while $\mathrm{Sym}(L) = Y \cdot X$ can be extended if and only if $k = 0$.

The method of Laplace starts with an initial operator L and applies two transformations $L \to L_1$ and $L \to L_{-1}$ called *Laplace transformations* until one of the transformed operators is factorable (the Laplace transformations are admitted by operators of the form (1)). The Laplace invariants of the transformed operators L_1 and L_{-1} can be expressed in terms of the invariants of the initial operator:

$$h_1 = 2h - k - \partial_{xy}(\ln|h|),\ k_1 = h,\ h_{-1} = k,\ k_{-1} = 2k - h - \partial_{xy}(\ln|k|)\ .$$

So assuming that L is not factorable, and so $h \neq 0, k \neq 0$, only one invariant for each of the transformed operators can vanish. In such the way, instead of a sequence of operators, one considers the chain of their Laplace invariants

$$\ldots \leftrightarrow k_{-2} \leftrightarrow k_{-1} \leftrightarrow k \leftrightarrow h \leftrightarrow h_1 \leftrightarrow h_2 \leftrightarrow \ldots\ . \tag{3}$$

One iterates the Laplace transformations until one of the Laplace invariants in the sequence (3) vanishes. In this case, one can solve the corresponding transformed equation in quadratures and then use the inverse substitution to obtain the complete solution of the original equation. What is more, one may prove (see for example [8]) that if the chain (3) is finite in both directions, then one may obtain a quadrature-free expression for the general solution of the original equation.

In the case considered by Laplace, the invariants h and k can be simply obtained from the incomplete factorizations, $L = (D_x + b) \circ (D_y + a) + h = (D_y + a) \circ (D_x + b) + k$. That is why the invariant necessary and sufficient conditions of factorizations becomes so simple ($h = 0$ or $k = 0$). For hyperbolic operators of the next order — order three — the situation become much more difficult: the "remainder" of an incomplete factorization is not invariant in the generic case, and the invariant conditions are not trivial.

In the present paper we find invariant necessary and sufficient conditions of factorizations extending given (we consider all the possibilities) factorizations of the principal symbol of third-order bivariate hyperbolic linear partial differential operators. These invariant conditions are given in terms of invariants of the generating set of invariants found in [9]. Also in the scope of the paper we investigate the classical operation of taking the formal adjoint of an operator, define it on the equivalent classes of the considered LPDOs, and obtain explicit formulae in the space of invariants. Some instances of the latter result allow us to reduce the number of case considerations when finding an invariant definition of the property of the existence of a factorization.

The paper is organized as follows. In Section 2 preliminaries facts and definitions are given. In Section 3 we discuss connections between factorization of LPDOs and invariants of a family of LPDOs under the gauge transformations, also we show how we reduce the number of factorization types to consider to

just four ones. In Sections 4, 5, and 6, the existence of factorizations of these four factorization types has been expressed in terms of invariants of the generating system of invariants found in [9]. In Section 7 the operation of taking the formal adjoint is defined in the space of invariants.

2 Definitions and Notations

Consider a field K 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 $D_i a = aD_i + \partial_i(a)$. 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 $\mathrm{Sym}_L = \sum_{i+j=d} a_{ij} X^i Y^j$ 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.

Let K^* denote the set of invertible elements in K. For $L \in K[D]$ and every $g \in K^*$ consider the gauge transformation $L \to g^{-1} \circ L \circ g$. 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 basis in which all possible differential invariants can be expressed.

We use the usual abbreviations: LPDO for Linear Partial Differential Operator, LPDE for Linear Partial Differential Equation.

3 Factorization Via Invariants

Any hyperbolic third-order LPDO in some system of coordinates has the form

$$L = (pD_x + qD_y)D_xD_y + \sum_{i+j=0}^{2} a_{ij} D_x^i D_y^j \ , \tag{4}$$

where all the coefficients belong to K (they are some functions of x and y) and where $p, q \neq 0$.

Remark 1. Note that the normalized form of such operators is slightly simpler than above, namely, one can put without loss of generality $p = 1$. The introduction of the parameter p makes all the reasoning symmetric with respect to x and y, and therefore reduces the number of cases requiring consideration on the way to our main goal.

Operators of the form (4) admit gauge transformations, and p, q are the trivial invariants.

Theorem 1. *[9] The following form a generating set of invariants for operators of the form (4):*

$$\left.\begin{aligned}
&I_p = p\ ,\\
&I_q = q\ ,\\
&I_1 = 2q^2a_{20} - qa_{11}p + 2a_{02}p^2\ ,\\
&I_2 = -qp^2a_{02y} + a_{02}p^2q_y + q^2a_{20x}p - q^2a_{20}p_x\ ,\\
&I_3 = a_{10}p^2 + (2q_yp - 3qp_y)a_{20} + a_{20}^2q - a_{11y}p^2 + a_{11}p_yp + qpa_{20y}\\
&\quad -a_{11}a_{20}p\ ,\\
&I_4 = a_{01}q^2 + (2qp_x - 3pq_x)a_{02} + a_{02}^2p - a_{11x}q^2 + a_{11}qq_x + qpa_{02x}\\
&\quad -a_{02}a_{11}q\ ,\\
&I_5 = a_{00}p^3q + 2a_{02}p^3a_{20x} - 2q^2a_{20}^2p_x - a_{02}a_{10}p^3 - a_{01}a_{20}p^2q\\
&\quad +\tfrac{1}{2}a_{11x}p_yp^2q + \tfrac{1}{2}a_{11y}p_xp^2q + (\tfrac{1}{2}p_{xy}p^2q - p_xp_ypq)a_{11}\\
&\quad +a_{11}pqa_{20}p_x - \tfrac{1}{2}a_{11xy}p^3q + (qq_xp^2 - q^2p_xp)a_{20y} - 2a_{02}p^2a_{20}p_x\\
&\quad -a_{11}p^2qa_{20x} + (qp^2q_y - pq^2p_y)a_{20x} + 2q^2a_{20}a_{20x}p +\\
&\quad (qq_{xy}p^2 - q^2p_{xy}p + 4q^2p_xp_y - 2qp_xq_yp - 2qq_xpp_y)a_{20}\\
&\quad +a_{20}a_{11}a_{02}p^2\ .
\end{aligned}\right\} \qquad (5)$$

Any set of values of these invariants uniquely defines an equivalent class of operators of the form (4). All the invariant properties of such operators can be described in terms of the invariants of the above generating set.

Lemma 1. *The property of having a factorization (or a factorization extending a certain factorization of the symbol) is invariant.*

Proof. Let $L = F_1 \circ F_2 \circ \ldots \circ F_k$, for some operators $F_i \in K[D]$. For every $g \in K^*$

$$g^{-1} \circ L \circ g = \left(g^{-1} \circ F_1 \circ g\right) \circ \left(g^{-1} \circ F_2 \circ g\right) \circ \ldots \circ \left(g^{-1} \circ F_k \circ g\right)\ ,$$

and since the gauge transformations do not alter the symbol of an LPDO, we prove the statement of the theorem.

Remark 2. Recall that as for two LPDOs $L_1, L_2 \in K[D]$ we have

$$\mathrm{Sym}_{L_1 \circ L_2} = \mathrm{Sym}_{L_1} \cdot \mathrm{Sym}_{L_2}\ ,$$

any factorization of an LPDO extends some factorization of its symbol. In general, if $L \in K[D]$ and $\mathrm{Sym}_L = S_1 \cdot \ldots \cdot S_k$, then we say that the factorization

$$L = F_1 \circ \ldots \circ F_k, \qquad \mathrm{Sym}_{F_i} = S_i,\ \forall i \in \{1, \ldots, k\},$$

is of the *factorization type* $(S_1) \ldots (S_k)$.

Consider all possible factorizations of the symbol of an LPDO (4), namely $\mathrm{Sym}_L = (pX + qY)XY$. Owing to the non-commutativity of LPDOs one has to consider factorizations of the polynomial Sym_L assuming that factors do not commute. Thus $\mathrm{Sym}_L = (pX + qY)XY$ has 12 different factorizations:

$$
\begin{array}{l}
(S)(XY)\ , \\
(XY)(S)\ , \\
(X)(YS)\ ,\ (Y)(XS)\ , \\
(YS)(X)\ ,\ (XS)(Y)\ , \\
(S)(X)(Y)\ ,\ (S)(Y)(X)\ , \\
(X)(S)(Y)\ ,\ (Y)(S)(X)\ , \\
(X)(Y)(S)\ ,\ (Y)(X)(S)\ ,
\end{array}
$$

where $S = (pX + qY)$. By Remark (1) it is enough to consider one of the factorizations for each of the lines of the list above. Thus, there are seven cases to consider. Proceeding further, we can almost half this number of cases (i.e. 7 cases) once we know how to express generating invariants of the formal adjoint $L^\dagger$ of an LPDO L in terms of generating invariants of L. In Section 7 we find such formulae, and so only the the following cases need to be considered:

$$
\begin{array}{l}
(S)(XY)\ , \\
(X)(YS)\ , \\
(S)(X)(Y)\ , \\
(X)(S)(Y)\ .
\end{array}
$$

4 Factorization Type $(pX + qY)(XY)$

Theorem 2. *Consider an equivalent class of (4) given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). The operators of the class have a factorization of the factorization type $(pX+qY)(XY)$ if and only if the following two conditions hold.*

$$
\begin{array}{l}
I_3q^3 - I_4p^3 + pq(pI_{1x} - qI_{1y}) + pq(q_y - p_x)I_1 + 2(p_yq^2 - q_xp^2)I_1 - 3pqI_2 = 0\ , \\
I_sI_2 + I_r + 2pq^2I_{2x} + q^3I_{2y} = 0\ .
\end{array}
$$

Proof. First, using the formulae of the invariants (5), we express the coefficients a_{11}, a_{10}, a_{01}, a_{00} of (4) in terms of these invariants and a_{20}, a_{02}. We have, for example, $a_{11} = (-I_1 + 2q^2a_{20} + 2a_{02}p^2)/(pq)$, and other expressions are too large to give them here explicitly. Then an operator (4) of the class has factorization $F_{(pX+qY)(XY)} = (pD_x+qD_y+r)\circ(D_{xy}+aD_x+bD_y+c)$, where all the coefficients are functions of x and y, takes place if and only if $L - F_{(pX+qY)(XY)} = 0$. Equating the coefficients at D_{xx}, D_{xy}, D_{yy}, D_y on the both sides of this equality, one computes

$$
\begin{array}{l}
a = a_{20}/p\ , \quad b = a_{02}/q\ , \quad r = -\dfrac{1}{pq}\ I_1 + \dfrac{q^2a_{20} + a_{02}p^2}{pq}\ , \\
c = (I_4p^2 - qpI_{1x} + 2q^3pa_{20x} + (2q_xp + qp_x)I_1 - 2q^3p_xa_{20} \\
+a_{02}a_{20}q^2p - q^2p^2a_{02y} + qp^2a_{02}q_y)/(q^3p^2)
\end{array}
$$

as p and q are known to be different from zero. While equating the coefficients of D_x and the "free" coefficients of both sides of that, we get two conditions for the existence of a factorization, which still involve the coefficients a_{20} and a_{02} and, therefore, are not invariant. On the other hand, by Lemma 1, there should be a way to describe existence of a factorization (a factorization extending certain factorization of the symbol) invariantly.

Consider the first condition, which after multiplication by p^2q^3, can be noticed to be equivalent to the following constrain for invariants of L:

$$C_{10} = I_3q^3 - I_4p^3 + pq(pI_{1x} - qI_{1y}) + pq(q_y - p_x)I_1 + 2(p_yq^2 - q_xp^2)I_1 - 3pqI_2 = 0\ . \tag{6}$$

Consider the second condition multiplied for convenience on both sides by p^2q^4 (denote the result as $C_{00} = 0$). It is a large expression. Consider all the terms of C_{00} with second-order derivatives of a_{20}, a_{02}:

$$-2p^2q^4a_{20xx}\ , -pq^5a_{20xy}\ , 2q^3p^3a_{02xy}\ , 2p^2q^4a_{02yy}\ .$$

Thus, subtracting $2pq^2I_{2x} + q^3I_{2y}$ from C_{00}, we cancel the terms with second-order derivatives of a_{20}, a_{02}. Denote the result of the subtraction by C_{001}. Consider terms of C_{001} containing first-order derivatives of a_{20}, a_{02}:

$$q^3(I_1 + 2q^2p_y + 2qpq_y + 4p^2q_x + 4pqp_x - 3a_{02}p^2)a_{20x}\ , \tag{7}$$

$$-q^2p(I_1 + 2q^2p_y + 2qpq_y + 4p^2q_x + 4pqp_x - 3a_{02}p^2)a_{02y}\ , \tag{8}$$

and compare them with those in I_2. One can see that the ratio of the coefficient at a_{20x} in (7) to that in I_2 equals to the ratio of the coefficient at a_{02y} in (8) to that in I_2, and this ratio is

$$s = I_s - 3pqa_{02}\ ,$$

where $I_s = \frac{q}{p}(4p(qp_x + pq_x) + 2q(pq_y + qp_y) + I_1)$, that is an invariant. Subtracting sI_2 from C_{001} (denote the result of the subtraction by C_{002}), we cancel all the terms contaning first-order derivatives of a_{20}, a_{02}, and get

$$C_{002} = (I_3q^3 - I_4p^3 + qp^2I_{1x} - pq^2I_{1y} + pq(q_y - p_x)I_1 + 2(p_yq^2 - q_xp^2)I_1)a_{02} + I_r\ , \tag{9}$$

where $I_r = \frac{q^3p}{2}I_{1xy} - qp^2(qI_{4y} - pI_{4x}) + \frac{q^3}{p}I_5 + q^2p^2I_{1xx} - \frac{3q^2pq_x}{2}I_{1y} + pI_1I_4 + \Big(- 2qp^2q_{xx} + 6q_x^2p^2 + q^2q_xp_y + 4qpq_xp_x - q^2pp_{xx} + q^2p_xq_y - \frac{3q^2pq_{xy}}{2} + 5qpq_xq_y + 2p_x^2q^2 - \frac{q^3p_xp_y}{p}\Big)I_1 + 3p^2(qq_y + pq_x)I_4 + \Big(2q_x + \frac{qp_x}{p}\Big)I_1^2 - pq\Big(\frac{3qq_y}{2} + 2qp_x + 4pq_x\Big)I_{1x} - qI_1I_{1x}$ is an invariant. Comparing (9) with (6), one can notice that the coefficient at a_{02} in C_{002} equals $(C_{10} + 3pqI_2)$. As $C_{10} = 0$ is a necessary condition for L to be factorable with the considered factorization type, the coefficient at a_{02} in C_{002} becomes just $3pqI_2$. Which is fortunately is canceled in expression for C_{00}, when we combine the results:

$$\begin{aligned} C_{00} &= (C_{10} + 3pqI_2)a_{02} + (I_s - 3pqa_{02})I_2 + I_r + 2pq^2I_{2x} + q^3I_{2y} \\ &= C_{10}a_{02} + I_sI_2 + I_r + 2pq^2I_{2x} + q^3I_{2y}\ . \end{aligned}$$

Corollary 1 (case $p = 1$). *Consider equivalent classes of (4) possessing the property $p = 1$, and given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). The operators of the class have a factorization of the factorization type $(X+qY)(XY)$ if and only if*

$$\begin{cases} I_3q^3 - I_4 + q(I_{1x} - qI_{1y}) + qq_yI_1 - 2q_xI_1 - 3qI_2 = 0 \ , \\ I_sI_2 + I_r + 2q^2I_{2x} + q^3I_{2y} = 0 \ . \end{cases}$$

where $I_s = q(4q_x + 2qq_y + I_1)$ *and* $I_r = \frac{q^3}{2}I_{1xy} - q(qI_{4y} - I_{4x}) + q^3I_5 + q^2I_{1xx} - \frac{3q^2q_x}{2}I_{1y} + I_1I_4 + \Big(-2qq_{xx} + 6q_x^2 - \frac{3q^2q_{xy}}{2} + 5qq_xq_y\Big)I_1 + 3(qq_y + q_x)I_4 + 2q_xI_1^2 - q\Big(\frac{3qq_y}{2} + 4q_x\Big)I_{1x} - qI_1I_{1x}$.

5 Factorization Type $(X)(YS)$

Theorem 3. *Consider an equivalent class of (4) given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). The operators of the class have a factorization of the factorization type $(X)(pXY + qY^2)$ if and only if*

$$\begin{cases} I_4 - 2q_xp_xq + 2q_x^2p - qpq_{xx} + q^2p_{xx} = 0 \ , \\ -4p^2q_xI_2 + p^2qI_{2x} + I_r = 0 \ , \end{cases}$$

where $I_r = -3/2q_xqp^2I_{1y} + I_5q^2 + \frac{1}{2}I_{1xy}q^2p^2 - q^3pI_{3x} + (q^2pq_x + 2p_xq^3)I_3 + (-p_{xy}q^2p + 3q_xp_yqp + 2q_xq_yp^2 - \frac{1}{2}q_{xy}qp^2 + p_xp_yq^2)I_1 + (-p_yq^2p - \frac{1}{2}q_yqp^2)I_{1x}$.

Proof. The case we consider here is much easier than that of section 4. As we do there first we express $a_{00}, a_{10}, a_{01}, a_{11}$ in terms of a_{20}, a_{02} and the invariants (5). Then for an operator L (4) of the class consider a factorization of the form

$$L = (D_x + r) \circ (pD_{xy} + qD_{yy}aD_x + bD_y + c) \ , \tag{10}$$

where all the coefficients belong to K (some functions of x and y). Substituting just found expressions for a_{00}, a_{10}, a_{01}, a_{11}, and equating the coefficients at D_{yy}, D_{xx}, D_{xy}, D_x on the both sides of (10), one computes $r = (a_{02} - q_x)/q$, $a = a_{20}$, $b = -(I_1 - 2q^2a_{20} - a_{02}p^2 - p^2q_x + p_xqp)/q/p$, $c = -(-I_3q^2 + a_{20}qI_1 - a_{20}^2q^3 - a_{20}qa_{02}p^2 + q^3p_ya_{20} + qpI_{1y} - q^3pa_{20y} - 2qp^3a_{02y} - q_ypI_1 + 2q_yp^3a_{02} - 2qp_yI_1 - qa_{20}p^2q_x + a_{20x}q^2p^2)/q^2/p^2$, as p and q are known to be different from zero. Equating the coefficients at D_y we get first constrain on invariants,

$$I_4 - 2q_xp_xq + 2q_x^2p - qpq_{xx} + q^2p_{xx} = 0 \ . \tag{11}$$

Equating the "free" coefficients of the both sides of (10), we get a condition of existence of a factorization in particular in terms of a_{20} and a_{02}. To cancel denominators, multiply this condition on the both sides by p^3q^3 (denote the result as $C_{00} = 0$). Consider all the terms of C_{00} with second-order derivatives of a_{20}, a_{02}:

$$p^3q^3a_{20xx} \ , -q^2p^4a_{02xy} \ .$$

Thus, subtracting p^2qI_{2x} from C_{00}, we kill all the terms with second-order derivatives of a_{20}, a_{02}. Denote the result of the subtraction by C_{001}. Consider terms of C_{001} containing first-order derivatives of a_{20}, a_{02}:

$$-4q_xp^3q^2a_{20x}\ , 4q_xqp^4a_{02y}\ ,$$

and compare them with those in I_2. One can see that subtracting $-4p^2q_xI_2$ from C_{001} we cancel all the terms containing first-order derivatives of a_{20}, a_{02}. Denote the result of this subtraction by C_{002}, then

$$C_{002} = (I_4qp^2 - 2q^2p^2q_xp_x + q^3p^2p_{xx} + 2qp^3q_x^2 - q^2p^3q_{xx})a_{20} + I_r\ , \qquad (12)$$

where $I_r = -3/2q_xqp^2I_{1y} + I_5q^2 + \frac{1}{2}I_{1xy}q^2p^2 - q^3pI_{3x} + (q^2pq_x + 2p_xq^3)I_3 + (-p_{xy}q^2p + 3q_xp_yqp + 2q_xq_yp^2 - \frac{1}{2}q_{xy}qp^2 + p_xp_yq^2)I_1 + (-p_yq^2p - \frac{1}{2}q_yqp^2)I_{1x}$ is an invariant. The constrain (11) implies that the coefficients at a_{02} in C_{002} is zero provided the factorization (10) takes place. Thus, combining the results, we have

$$C_{00} = -4p^2q_xI_2 + p^2qI_{2x} + I_r\ .$$

Corollary 2 (case $p = 1$). *Consider equivalent classes of (4) possessing the property $p = 1$, and given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). The operators of the class have a factorization of the factorization type $(X)(XY + qY^2)$ if and only if*

$$\begin{cases} I_4 + 2q_x^2 - qq_{xx} & = 0\ , \\ I_5q^2 - 4p^2q_xI_2 + p^2qI_{2x} + \frac{1}{2}I_{1xy}q^2 - I_{3x}q^3 - & \\ \frac{3}{2}q_xI_{1y}q - \frac{1}{2}q_yI_{1x}q + q_xI_3q^2 + (-\frac{1}{2}q_{xy}q + 2q_xq_y)I_1 & = 0\ . \end{cases}$$

6 Factorization Types $(pX + qY)(X)(Y)$ and $(X)(pX + qY)(Y)$

Here we omit all the proofs as they employ similar to the section 4 ideas and are much simpler.

Theorem 4. *Consider an equivalent class of (4) given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). The operators of the class have a factorization of the factorization type $(pX + qY)(X)(Y)$ if and only if*

$$\begin{cases} I_3q^2 - qpI_{1y} + (q_yp + 2qp_y)I_1 & = 0\ , \\ I_4p^2 - I_1xqp + (2qxp + pxq)I_1 & = 0\ , \\ I_5q^2 + (p_xpq^2 + \frac{1}{2}q_xp^2q)I_{1y} - \frac{1}{2}I_{1xy}p^2q^2 + (p_ypq^2 + \frac{1}{2}q_yp^2q)I_{1x} + & \\ (-3p_xp_yq^2 - p_xq_ypq + p_{xy}pq^2 + \frac{1}{2}q_{xy}p^2q - q_xp_ypq - q_xq_yp^2)I_1 & = 0\ . \end{cases}$$

Theorem 5. *Consider an equivalent class of (4) given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). The operators of the class have a factorization of the factorization type $(X)(pX+qY)(Y)$ if and only if*

$$\begin{cases} I_3q^2 - qpI_{1y} + q_ypI_1 + 2qp_yI_1 & = 0\ , \\ 2pq_x^2 - qpq_{xx} + q^2p_{xx} + I_4 - 2q_xp_xq & = 0\ , \\ I_5q^2 - \frac{1}{2}p^2q^2I_{1xy} + \frac{1}{2}q_xqp^2I_{1y} + p_xq^2pI_{1y} + \\ (p_yq^2p + \frac{1}{2}q_yqp^2)I_{1x} + (-q_xq_yp^2 + \frac{1}{2}q_{xy}qp^2 - \\ 3p_xp_yq^2 + p_{xy}q^2p - q_xp_yqp - p_xq_yqp)I_1 & = 0\ . \end{cases}$$

7 Formal Adjoint

In this section we consider the operation of taking the formal adjoint of an LPDO, and define such operation on the equivalent classes of third-order bivariate non-hyperbolic LPDO. At the end of the section we apply this knowledge to complete the cases' consideration in the finding of invariant condition of the property of the existence of a factorization of certain factorization type.

For an operator $L = \sum_{|J|\le d} a_J D^J$, where $a_J \in K$, $J \in \mathbf{N}^n$ and $|J|$ is the sum of the components of J, the *formal adjoint* is defined as

$$L^\dagger(f) = \sum_{|J|\le d} (-1)^{|J|} D^J(a_J f)\ ,\ \forall f \in K\ .$$

The formal adjoint possesses the following useful for the factorization theory properties:

$$(L^\dagger)^\dagger = L\ ,\quad (L_1 \circ L_2)^\dagger = L_2^\dagger \circ L_1^\dagger\ ,\quad \mathrm{Sym}_L = (-1)^{\mathrm{ord}(L)}\mathrm{Sym}_{L^\dagger}\ .$$

The property of having a factorization is invariant under the operation of taking the formal adjoint, while the property of having a factorization of certain factorization type is not invariant, and an operator L has a factorization of some factorization type $(S_1)(S_2)$ (where $\mathrm{Sym}_L = S_1S_2$) if and only if $L^\dagger$ has that of factorization type $(S_2)(S_1)$.

Lemma 2. *The operation of taking the formal adjoint can be defined on the equivalent classes of LPDOs.*

Proof. Show that operation of taking the formal adjoint and the gauge transformations of LPDOs commute. For every $g \in K^*$, and $f = g^{-1}$ we have

$$(g^{-1} \circ L \circ g)^\dagger = g^\dagger \circ L^\dagger \circ (g^{-1})^\dagger = g \circ L^\dagger \circ g^{-1} = f^{-1} \circ L^\dagger \circ f.$$

Example 1 (LPDOs of order 2). For operators of the form

$$L = D_{xy} + aD_x + bD_y + c$$

there is a complete generating set of invariants that consists of first-order invariants: $h = c - a_x - ab$ and $k = c - b_y - ab$. For the formal adjoint

$$L^\dagger = D_{xy} - aD_x - bD_x + c - a_x - b_y$$

they are $h^\dagger = c - b_y - ab$ and $k^\dagger = c - a_x - ab$, and so $h_t = k$, $k_t = h$.

Theorem 6 (formal adjoint for equivalent classes). *Consider the equivalent classes of (4) given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). Then the operation of taking of the formal adjoint is defined by the following formulae*

$$\left.\begin{aligned}
I_1^\dagger =& I_1 - 2q^2p_y - 2p^2q_x + 2p_xqp + 2q_yqp \ , \\
I_2^\dagger =& -I_2 - qp^2q_{xy} + q_yp^2q_x + q^2pp_{xy} - q^2p_xp_y \ , \\
I_3^\dagger =& -I_3 + \tfrac{1}{q^2}\Big(2pI_2 - (2p_yq + q_yp)I_1 + qpI_{1y} - 2p_yq_yq^2p + \\
& \qquad 2q^3p_y^2 + q_{yy}q^2p^2 - q^3pp_{yy}\Big) \ , \\
I_4^\dagger =& -I_4 + \tfrac{1}{p^2}\Big(- 2qI_2 - (p_xq + 2q_xp)I_1 + qpI_{1x} + 2p^3q_x^2 - 2p^2q_xqp_x \\
& \qquad +p_{xx}q^2p^2 - qp^3q_{xx}\Big) \ , \\
I_5^\dagger =& I_5 + p_1I_1 + p_3I_3 + p_4I_4 + p_{12}I_{1y} + p_{11}I_{1x} + p^2I_{1xy} - qpI_{3x} - \tfrac{p^3}{q}I_{4y} + p_0 \\
& -pI_{2y} + \tfrac{p^2}{q}I_{2x} + (-2q^2p^3q_x + 4p_yq^4p - q^2pI_1 - 2q^3p^2p_x)/(q^4p)I_2 \ ,
\end{aligned}\right\} \tag{13}$$

where $p_1 = (4q_xp_yp + p_xq_yp - 2q_{xy}p^2)/q + (4q_xq_yp^2)/q^2 + 3p_xp_y - p_{xy}p$, $p_3 = 2qp_x + pq_x$, $p_4 = (2q_yp^3 + p^2p_yq)/q^2$, $p_0 = p^3q_xq_{yy} - 2q^2p_xp_y^2 - qq_xp^2p_{yy} + q^2p_xpp_{yy} - qp^2q_{yy}p_x - 2p^2p_yq_yq_x + 2qq_xpp_y^2 + 2qp_ypq_yp_x$, $p_{11} = -(2p_ypq + q_yp^2)/q$, $p_{12} = -(p_xpq + 2q_xp^2)/q$.

Proof. Consider an operator L in the form (4) of some equivalent class and express the coefficients a_{11}, a_{10}, a_{01}, a_{00} of it in terms of the invariants (5) and a_{20}, a_{02}. Then compute the formal adjoint $L^\dagger$, and compute the invariants (5). The first invariant of $L^\dagger$ is already given in terms of the invariants of L and in the same form as in the statement of the theorem. The second invariant of $L^\dagger$ is

$$I_2^\dagger = qp^2a_{02y} - qp^2q_{xy} - a_{02}p^2q_y + q_yp^2q_x + q^2pp_{xy} - q^2a_{20x}p - q^2p_xp_y + q^2a_{20}p_x \ .$$

Employing the expression for the invariant I_2 we eliminate a_{20} and a_{02} from this expression and get $I_2^\dagger$ as it is in the statement of the theorem. Analogously, we obtain the forms for $I_3^\dagger$, $I_4^\dagger$ that are given in the statement of the theorem.

The fifth invariant $I_5^\dagger$ of $L^\dagger$ is a large expression containing a_{20} and a_{02}, and their second and first derivatives. The terms containing a_{02yy} are canceled if we add pI_{2y} to $I_5^\dagger$. Then the only term containing a_{20xx} is p^3qa_{20xx}, and we cancel it by subtraction of p^2I_{2x}/q. Then no second-order derivatives are left, and we notice that the ratio

$$C = (-2q^2p^3q_x + 4p_yq^4p - q^2pI_1 - 2q^3p^2p_x)/(q^4p)$$

of the coefficient for a_{20x} in the obtained expression to that in I_2 is equal to the ratio of the coefficient for a_{02y} in the obtained expression to that in I_2. Thus, subtracting CI_2, we cancel first-order derivatives, and have as the result the invariant expression

$$I_{55} = I_5 + p_1I_1 + p_3I_3 + p_4I_4 + p_{12}I_{1y} + p_{11}I_{1x} + p^2I_{1xy} - qpI_{3x} - \frac{p^3}{q}I_{4y} + p_0 \ ,$$

where $p_1 = (4q_xp_yp + p_xq_yp - 2q_{xy}p^2)/q + (4q_xq_yp^2)/q^2 + 3p_xp_y - p_{xy}p$, $p_3 = 2qp_x + pq_x$, $p_4 = (2q_yp^3 + p^2p_yq)/q^2$, $p_0 = p^3q_xq_{yy} - 2q^2p_xp_y^2 - qq_xp^2p_{yy} + q^2p_xpp_{yy} - qp^2q_{yy}p_x - 2p^2p_yq_yq_x + 2qq_xpp_y^2 + 2qp_ypq_yp_x$, $p_{11} = -(2p_ypq + q_yp^2)/q$, $p_{12} = -(p_xpq + 2q_xp^2)/q$ are differential-algebraic expressions of p and q. Thus,

$$I_5^\dagger = I_{55} - pI_{2y} + \frac{p^2}{q}I_{2x} + CI_2 \ .$$

Theorem 6 is the one that allows us to half the cases necessary to consider to describe existence of factorizations of different factorizations types. Below is an example on how to obtain invariant conditions of existence of a factorization of the certain type of factorizations $(XY)(pX + qY)$, if those are given (found in the section 4) for the "symmetric" factorization type $(pX + qY)(XY)$.

Corollary 3. *Consider an equivalent class of (4) given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). Operators of the class have a factorization of factorization type $(XY)(pX + qY)$ if and only if*

$$\begin{cases} 0 = q_0 - qpI_2 + q^3I_3 - p^3I_4 \ , \\ 0 = p_0 + p_1I_1 - 4pqq_xI_2 + p_3I_3 + p_4I_4 + \frac{q^3}{p}I_5 - q^4I_{3x} \\ \quad -(pq^2q_y/2 + q^3p_y)I_{1x} + p^3qI_{4x} + pI_1I_4 + (pq^3)/2I_{1xy} \\ \quad +pq^2I_{2x} - 3pq^2q_x/2I_{1y} \ , \end{cases}$$

where q_0, p_0, p_1, p_3, p_4 are expressions of p, q and their derivations, more precisely, $q_0 = -2p_yq_yq^3p + q_{yy}q^3p^2 - q^4pp_{yy} + q^3p^2p_{xy} - q^2p^3q_{xy} - p_{xx}q^2p^3 + qp^4q_{xx} + 2q^4p_y^2 - 2p^4q_x^2 - q^3pp_xp_y + q_yqp^3q_x + 2p^3q_xqp_x$, $p_0 = 2pq^3p_xq_xp_y - 2p^2q^2q_xq_yp_x + 2p^2q^3p_{xx}p_x + 8p^4qq_xq_{xx} - 10p^4q_x^3 - 5p^3p_{xx}q^2q_x - p^4q^2q_{xxx} + p^3q^3p_{xxx} - 5p^3q^2q_{xx}p_x - 4p^2q^2p_x^2q_x - pq^4p_{xx}p_y + 14p^3q_x^2qp_x + 2p^3qq_x^2q_y + p^2q^3p_{xx}q_y + p^2q^3q_{xx}p_y - p^3q^2q_yq_{xx} - 2p^2q^2q_x^2p_y$, $p_1 = 3q^2q_xp_y - 2pqq_xp_x - p^2qq_{xx} - q^3p_{xy} + 1/pq^3p_xp_y + 2p^2q_x^2 + pq^2p_{xx} + 2pqq_xq_y - \frac{1}{2}pq^2q_{xy}$, $p_3 = 2q^4p_x/p + q^3q_x$, $p_4 = 2p^2p_xq + p^2qq_y - 5p^3q_x - pp_yq^2$.

Proof. Operators of the class have a factorization of factorization type $(XY)(pX + qY)$ if and only if their formal adjoints $L^\dagger$ have a factorization of factorization type $(-pX - qY)(XY)$, which by theorem 2 is true if and only if $-I_3^\dagger q^3 + I_4^\dagger p^3 + pq(-pI_{1tx} + qI_{1ty}) + pq(-q_y + p_x)I_1^\dagger + 2(-p_yq^2 + q_xp^2)I_1^\dagger - 3pqI_2^\dagger = 0$ and $I_{st}I_2^\dagger + I_{rt} - 2pq^2I_{2tx} - q^3I_{2ty} = 0$. Using the results of section 7, these conditions can be rewritten in terms of the five invariants (5) of L, and after simplifications the expressions given in the statement of the theorem can be obtained.

Consider the special case where p and q are constants. Then without loss of generality one can assume $p = q = 1$.

Corollary 4 (case of the symbol with constant coefficients). *An LPDO (4) with $p = q = 1$ has a factorization of factorization type $(XY)(X + Y)$ if and only if*

$$\begin{cases} I_3 = I_2 + I_4 \ , \\ 0 \ = I_5 + \frac{1}{2}I_{1xy} + I_4I_1 \ . \end{cases}$$

8 Symbol of Constant Coefficients

In the criteria for the existence of factorizations of different factorization types, the coefficients p and q of the symbol, and their derivatives occur fairly often. Therefore, it is interesting to look at the structure of the formulae in the important particular case in which p and q are constants, and, therefore, there exists a normal form of the operator with the (principal) symbol $(X+Y)XY$. Thus, without loss of generality one can assume $p = q = 1$, and then combining the results of the previous sections we obtain the necessary and sufficient conditions of the existence of factorizations for each of the 12 different types.

Theorem 7. *Consider equivalent classes of (4) possesing the property $p = q = 1$, and given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). Operators of the class have a factorization of factorization type*

$(S)(XY)$ *if and only if*

$$\left.\begin{array}{l} I_3 - I_4 + I_{1x} - I_{1y} - 3I_2 = 0\ , \\ I_1 I_2 + I_r + 2I_{2x} + I_{2y} \quad = 0\ , \end{array}\right\} \tag{14}$$

where $I_r = \frac{1}{2}I_{1xy} - I_{4y} + I_{4x} + I_5 + I_{1xx} + I_1 I_4 - I_1 I_{1x}$;
$(S)(X)(Y)$ *if and only if*

$$(14) \quad \& \quad I_2 - I_4 + I_{1x} = 0\ ;$$

$(S)(Y)(X)$ *if and only if*

$$(14) \quad \& \quad -2I_2 - I_4 + I_{1x} = 0\ ;$$

$(X)(SY)$ *if and only if*

$$I_4 = 0 \quad \& \quad I_{2x} + I_5 - I_{3x} + I_{1xy}/2 = 0\ ; \tag{15}$$

$(X)(S)(Y)$ *if and only if*

$$(15). \quad \& \quad I_3 - I_{1y} - 2I_2 = 0\ ;$$

$(X)(Y)(S)$ *if and only if*

$$(15). \quad \& \quad I_3 = I_2\ ;$$

$(XY)(S)$ *if and only if*

$$I_4 = I_3 - I_2 \quad \& \quad I_{1xy}/2 + I_1 I_4 + I_5 = 0\ .$$

$(YS)(X)$ *if and only if*

$$I_4 = I_{1x} - 2I_2 \quad \& \quad I_5 = I_1 I_2 \ .$$

$(XS)(Y)$ *if and only if*

$$I_3 - I_{1y} - 2I_2 = 0 \quad \& \quad I_5 = I_{2x} + I_{1xy}/2 \ ;$$

$(Y)(SX)$ *if and only if*

$$I_3 = 0 \quad \& \quad I_5 = (I_4 + I_2)_y + I_1 I_2 - I_{1xy}/2 \ ; \tag{16}$$

$(Y)(X)(S)$ *if and only if*

$$(16) \quad \& \quad I_4 = -I_2 \ ;$$

$(Y)(S)(X)$ *if and only if*

$$(16) \quad \& \quad I_4 - I_{1x} = -2I_2 \ ;$$

Theorem 8 (formal adjoint for equivalent classes). *Consider the equivalent classes of (4) possessing the properties $p = 1$ and $q = 1$ and which are given by the values of the invariants I_1, I_2, I_3, I_4, I_5 (5). Then the operation of taking of the formal adjoint is defined by the following formulae*

$$\left.\begin{aligned} I_1^\dagger &= I_1 \ , \\ I_2^\dagger &= -I_2 \ , \\ I_3^\dagger &= -I_3 + 2I_2 + I_{1y} \ , \\ I_4^\dagger &= -I_4 - 2I_2 + I_{1x} \ , \\ I_5^\dagger &= I_5 + I_{1xy} - I_{3x} - I_{4y} - I_{2y} + I_{2x} - I_1 I_2 \ . \end{aligned}\right\}$$

9 Conclusion

We obtained invariant necessary and sufficient conditions for the existence of factorizations extending given factorizations of the principal symbol of operators (any such factorization of the symbol corresponds to a factorization type). We defined the classical operation of taking the formal adjoint of an operator for the equivalent classes of the considered LPDOs. In particular, this result allows us to reduce the number of case considerations when finding an invariant definition of the property of the existence of a factorization. The existence criterium are found explicitly for the factorization types $(S)(XY)$, $(X)(YS)$, $(S)(X)(Y)$, $(X)(S)(Y)$, where $S = (pX + qY)$. Invariant conditions for the other eight possibilities of factorization types can be derived from these ones, and consideration of the most difficult case $(XY)(S)$ is provided as an example of such derivation.

For the future, it would be interesting to find such conditions in an algorithmic way for operators of general order. Another line of investigations might be the derivation of invariant conditions for generalized factorization in the sense of Tsarev [6].

Acknowledgments. This work was supported by Austrian Science Foundation (FWF) under the project DIFFOP.

References

1. Anderson, I., Juras, M.: Generalized Laplace invariants and the method of Darboux. Duke J. Math. 89, 351–375 (1997)
2. Anderson, I., Kamran, N.: The variational bicomplex for hyperbolic second-order scalar partial differential equations in the plane. Duke J. Math. 87, 265–319 (1997)
3. Athorne, C.: A $z \times r$ toda system. Phys. Lett. A. 206, 162–166 (1995)
4. Grigoriev, D., Schwarz, F.: Generalized loewy-decomposition of d-modules. In: ISSAC 2005: Proceedings of the 2005 international symposium on Symbolic and algebraic computation, pp. 163–170. ACM, New York (2005)
5. Tsarev, S.: Generalized laplace transformations and integration of hyperbolic systems of linear partial differential equations. In: ISSAC 2005: Proceedings of the 2005 international symposium on Symbolic and algebraic computation, pp. 325–331. ACM Press, New York (2005)
6. Tsarev, S.: Factorization of linear partial differential operators and darboux' method for integrating nonlinear partial differential equations. Theo. Math. Phys. 122, 121–133 (2000)
7. Darboux, G.: Leçons sur la théorie générale des surfaces et les applications géométriques du calcul infinitésimal, vol. 2. Gauthier-Villars (1889)
8. Goursat, E.: Leçons sur l'intégration des équations aux dérivées partielles du seconde ordre a deux variables indépendants, Paris, vol. 2 (1898)
9. Shemyakova, E., Winkler, F.: A full system of invariants for third-order linear partial differential operators in general form. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2007. LNCS, vol. 4770, pp. 360–369. Springer, Heidelberg (2007)

Symbolic Solution to Magnetohydrodynamic Hiemenz Flow in Porous Media

Seripah Awang Kechil[1] and Ishak Hashim[2]

[1] Department of Mathematics, Universiti Teknologi MARA, 40450 Shah Alam Selangor, Malaysia
seripah@tmsk.uitm.edu.my
[2] School of Mathematical Sciences, National University of Malaysia, 43600 Bangi Selangor, Malaysia
ishak_h@ukm.my

Abstract. A system of nonlinear ordinary differential equations governing the boundary layers of magnetohydrodynamic (MHD) Hiemenz flow in porous media is solved using a simple and efficient analytical technique of Adomian decomposition method (ADM) and Padé approximant through the computer algebra package system Maple. Several symbolic features of the Maple system are utilized to develop specific routines that compute the approximate analytical solutions of the stream, velocity and temperature functions for some exemplary prescribed parameters. Comparative study shows the well agreement of the present symbolic results with the existing numerical results.

1 Introduction

Multitudinous physical phenomena that exhibit nonlinear behaviour are typically modeled by systems of nonlinear ordinary or partial differential equations. Besides numerical approaches, computer algebra software packages and mathematical methods that provide exact and approximate analytical solutions are powerful tools for solving nonlinear differential equations. Computer algebra system expedites tedious and massive computations by manipulating mathematical expressions symbolically involving rote skills in algebra and calculus. Powerful systematic routines can be developed to obtain symbolic solutions to wide range of differential equations and enable the qualitative and quantitative properties of the problem to be easily investigated with less time and efforts.

Designing effective symbolic algorithms to handle large group of differential equations is a difficult task and various solution approaches can be considered such as the power series expansion method and the perturbation technique. Yahaya et al. [1] demonstrated the performance of the multistage modified Adomian decomposition method for solving the N-th order initial value problems of linear and nonlinear ordinary differential equations using Maple system. Li [2] proposed the use of Taylor's expansion and conversion from linear ordinary differential equation with variable coefficients to systems of linear equations.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 213–223, 2008.
© Springer-Verlag Berlin Heidelberg 2008

Edneral [3] investigated the periodic solutions of ODE systems using the normal form method by transforming the systems to simpler sets treated by multivariable power series and Mathematica. Pratibha and Jeffrey [4] used Maple system to solve a second-order nonlinear ordinary differential equation of Stokes-flow problem by developing routines to obtain large numbers of terms in the series solutions at a regular singular point and ordinary points.

This paper addresses a computational procedure involving a simple analytical approach of the Adomian decomposition method [5] based on series expansion and Padé approximant [6] for solving boundary-layer equations of MHD Hiemenz flow in porous media [7] through the computer algebra system Maple as an essential analytical tool. The ADM has been successfully applied to solve a wide class of linear and nonlinear differential equations in multiple disciplines [5,8,9,10,11]. The ADM can be extended to solve systems of differential equations and yields approximate analytical solutions in terms of a rapidly convergent infinite series with easily computable terms. It permits a reliably accurate quantitative solution and avoids the need for discretisation, perturbation, linearisation or unrealistic assumptions [5].

In fluid dynamics, by means of similarity transformations, boundary-layer equations of steady flows are normally reduced to a single or a system of nonlinear ordinary differential equation(s) which model the momentum, heat and mass transfer in the flow field. Studies on boundary-layer equations using ADM include the recent works of Hashim [8] on the classical Blasius' equation, Zheng et al. [12] on the laminar boundary layer equation of Marangoni convection in In-Ga-Sb system and Wazwaz [13] on boundary layer equation of viscous flow due to a moving sheet. Very recently, the ADM has been shown efficient in obtaining approximate analytical solution to an unsteady boundary layer problem over an impulsively stretching sheet by Awang Kechil and Hashim [14] and a system of coupled nonlinear ordinary differential equations of free-convective boundary layer [15]. Awang Kechil et al. [16] demonstrated that a ADM solution to a general boundary value problem can be used to obtain specific solutions to boundary layer equations with identical mathematical context but under different physical conditions.

2 Boundary Layer Equations of MHD Hiemenz Flow

Consider a steady two-dimensional laminar forced convection in MHD Hiemenz flow of an incompressible, electrically conducting viscous fluid against a flat plate through porous media with variable wall temperature and uniform surface mass flux. A transverse magnetic field is applied and the fluid is assumed to have constant properties. The magnetic Reynolds number is assumed small and the induced magnetic field, the Hall effect and the viscous dissipation terms are neglected. The governing boundary layer equations consist of a system of nonlinear ordinary differential equations, cf. Yih [7],

$$\mathrm{Pr} f''' + f f'' + (1 - f'^2) + \Omega(1 - f') + M^2(1 - f') = 0 \ , \tag{1}$$
$$\theta'' + f\theta' - \lambda f'\theta = 0 \ , \tag{2}$$

subject to boundary conditions

$$f(0) = f_w, \quad f'(0) = 0, \quad f'(\infty) = 1 \ , \tag{3}$$
$$\theta(0) = 1, \quad \theta(\infty) = 0 \ , \tag{4}$$

where f is the stream function, θ temperature, Pr Prandtl number, Ω permeability parameter, M Hartmann number, λ wall temperature exponent and f_w suction or injection parameter, $f_w < 0$ is for injection and $f_w > 0$ is for suction while $f_w = 0$ corresponds to an impermeable surface. Prime denotes differentiation with respect to η.

The system of the differential equations (1) and (2) will be solved in the domain $[0, \infty)$ subject to the initial boundary conditions and the asymptotic behaviour at the unbounded domain (3) and (4). Padé approximant is used to handle the difficulty that arises when trying to match the boundary conditions at infinity in order to determine the unknowns $f''(0)$ and $\theta'(0)$. Yih [7] employed the implicit finite difference of Keller-box method to obtain numerical solutions to the problem. We will employ the ADM and Padé approximant by developing several small routines in computer algebra package Maple to carry out specific symbolic tasks. Algorithms for recursive relations will be developed to calculate the Adomian polynomials and the ADM series solutions. The powerful command `Padé` in Maple system enables the long expression of the ADM series to be conveniently simplified into rational functions which approximate the exact analytical solution that valid over a large interval of η.

3 Solution Procedure

Now we shall demonstrate the simple applications of the ADM and Padé approximants to solve the system (1)–(4) for some selected values of the parameters involved. The general expressions of the Adomian polynomials and the recursive decomposition of the ADM series will be derived and coded in the computer algebra system Maple. First, let us introduce the two linear differential operators $L_1 \equiv \mathrm{d}^3/\mathrm{d}\eta^3$ and $L_2 \equiv \mathrm{d}^2/\mathrm{d}\eta^2$ and their inverse operators, $L_1^{-1}(\cdot) \equiv \int_0^\eta \int_0^\eta \int_0^\eta (\cdot)\mathrm{d}t\mathrm{d}t\mathrm{d}t$ and $L_2^{-1}(\cdot) \equiv \int_0^\eta \int_0^\eta (\cdot)\mathrm{d}t\mathrm{d}t$, respectively. Eqs. (1) and (2) written in operator forms are

$$L_1 f = -\frac{1}{\mathrm{Pr}} \left[f f'' + \left(1 - f'^2\right) + \Omega \left(1 - f'\right) + M^2 \left(1 - f'\right) \right] \ , \tag{5}$$
$$L_2 \theta = \lambda f'\theta - f\theta' \ , \tag{6}$$

Applying the inverse operators L_1^{-1} and L_2^{-1} on both sides of (5) and (6) respectively together with the boundary conditions at $\eta = 0$ in (3) and (4), we obtain

$$f = f_w + \frac{1}{2}\beta_1\eta^2 - \frac{1}{6\mathrm{Pr}}\left(1+\Omega+M^2\right)\eta^3 + \frac{1}{\mathrm{Pr}}L_1^{-1}\Big[\left(\Omega+M^2\right)f' + f'^2 - ff''\Big] \ , \tag{7}$$

$$\theta = 1 + \beta_2\eta + L_2^{-1}\left(\lambda f'\theta - f\theta'\right) \ , \tag{8}$$

where $\beta_1 = f''(0)$ and $\beta_2 = \theta'(0)$.

The nonlinear terms in (7) and (8) are expressed as functions of f, θ and their derivatives, $N_1(f) = f'^2 - ff''$ and $N_2(f,\theta) = \lambda f'\theta - f\theta'$. Thus (7) and (8) are

$$f = f_w + \frac{1}{2}\beta_1\eta^2 - \frac{1}{6\mathrm{Pr}}\left(1+\Omega+M^2\right)\eta^3 + \frac{1}{\mathrm{Pr}}L_1^{-1}\Big[\left(\Omega+M^2\right)f' + N_1(f)\Big] \ , \tag{9}$$

$$\theta = 1 + \beta_2\eta + L_2^{-1}\left[N_2(f,\theta)\right] \ . \tag{10}$$

The dimensionless functions $f(\eta)$ and $\theta(\eta)$ and the nonlinear functions $N_1(f)$ and $N_2(f,\theta)$ are decomposed into series as follows,

$$f(\eta) = \sum_{k=0}^{\infty} f_k(\eta), \quad \theta(\eta) = \sum_{k=0}^{\infty}\theta_k(\eta) \ , \tag{11}$$

$$N_1(f) = \sum_{k=0}^{\infty} A_k, \quad N_2(f,\theta) = \sum_{k=0}^{\infty} E_k \ , \tag{12}$$

where A_k and E_k are the so-called Adomian polynomials [5]. The convergence aspects of the Adomian's series (11) were studied by Cherruault [17], Cherruault and Adomian [18] and Hosseini and Nasabzadeh [19].

The Adomian polynomials A_k and E_k can be derived from the formula [5],

$$A_k = \frac{1}{k!}\left[\frac{\mathrm{d}^k}{\mathrm{d}\lambda^k}N\left(\sum_{i=0}^{\infty}\lambda^i f_i\right)\right]_{\lambda=0}, \quad k \geq 0 \ . \tag{13}$$

Hence, for the nonlinear terms $N_1(f)$ and $N_2(f,\theta)$, the Adomian polynomials A_0 and E_0 are,

$$A_0 = (f_0')^2 - f_0 f_0'' \ , \tag{14}$$

$$E_0 = \lambda f_0'\theta_0 - f_0\theta_0' \ , \tag{15}$$

and the general expressions of A_k and E_k for $k \geq 1$,

$$A_k = \begin{cases} 2\sum_{i=0}^{(k-1)/2} f_i' f_{k-i}' - \sum_{i=0}^{k} f_i f_{k-i}'' & \text{if } k \text{ odd} \ , \\ 2\sum_{i=0}^{(k-2)/2} f_i' f_{k-i}' + \left(f_{k/2}'\right)^2 - \sum_{i=0}^{k} f_i f_{k-i}'' & \text{if } k \text{ even} \ , \end{cases} \tag{16}$$

$$E_k = \sum_{i=0}^{k}\Big(\lambda f_i'\theta_{k-i} - f_i\theta_{k-i}'\Big) \ . \tag{17}$$

Substituting the relations (11) and (12) into (9) and (10) yield

$$\sum_{k=0}^{\infty} f_k(\eta) = f_w + \frac{1}{2}\beta_1\eta^2 - \frac{1}{6\mathrm{Pr}}(1+\Omega+M^2)\eta^3 + \frac{1}{\mathrm{Pr}}L_1^{-1}\left[(\Omega+M^2)\sum_{k=0}^{\infty} f_k' + \sum_{k=0}^{\infty} A_k\right] , \tag{18}$$

$$\sum_{k=0}^{\infty} \theta_k(\eta) = 1 + \beta_2\eta + L_2^{-1}\left(\sum_{k=0}^{\infty} E_k\right) . \tag{19}$$

Following Adomian [5] and Wazwaz [13], we arrange the respective individual decompositions of (18) and (19) in recursive relations for $k \geq 0$ as,

$$f_0 = f_w + \frac{1}{2}\beta_1\eta^2 , \tag{20}$$

$$f_1 = -\frac{1}{6Pr}(1+\Omega+M^2)\eta^3 + \frac{1}{\mathrm{Pr}}L_1^{-1}\left[(\Omega+M^2)f_0' + A_0\right] , \tag{21}$$

$$f_{k+2} = \frac{1}{\mathrm{Pr}}L_1^{-1}\left[(\Omega+M^2)f_{k+1}' + A_{k+1}\right] , \tag{22}$$

$$\theta_0 = 1 + \beta_2\eta , \tag{23}$$

$$\theta_{k+1} = L_2^{-1}(E_k) . \tag{24}$$

In the environment of the computer algebra package Maple, the variable `Digits` controlling the number of significant digits in all the calculations is set to 16. The Adomian polynomials (14)–(17) and recursive relations (20)–(24) with the prescribed values of the parameters involved are coded to generate the first k terms of f_k and θ_k of the series solutions ϕ_k and ω_k. We observe that as many terms as required can be generated within the computer's memory limitation and the accuracy improves as more terms in the series are included. So in our case, with the selected values of the parameters involved, we calculate up to the 39-term approximations of $f(\eta)$ and $\theta(\eta)$ and denote them by $\phi_{39} = \sum_{k=0}^{38} f_k(\eta)$ and $\omega_{39} = \sum_{k=0}^{38} \theta_k(\eta)$ respectively.

The truncated series solutions ϕ_{39}' and ω_{39} are transformed into Padé approximants of order $[N/N]$ denoted by $\phi_{39[N/N]}'$ and $\omega_{39[N/N]}$ respectively with the range of N is from 2 to 20. The approximants are matched to the boundary conditions at the unbounded domain (3) and (4) to determine the skin friction coefficient $f''(0)$ and the heat transfer coefficient $\theta'(0)$. We then evaluate the $\lim_{\eta\to\infty} \phi_{39[N/N]}' = 1$ and as N increases, the numerical values of $\beta_1 = f''(0)$ stabilize quickly to a solution within the range of the accuracy needed. The convergence rate differs for every set of parameters values considered due to the strength of nonlinearity of the resulted differential equations.

The approximation of $\beta_2 = \theta'(0)$ is done by substituting $\beta_1 = f''(0)$ in ω_{39} and following Boyd [20] we solve $\omega_{39[N/N]} = 0$ for some intermediate η. In the neighbourhood of η selected, as N increases, common converged values are observed. This approach is adopted in order to avoid massive computation in higher order

approximation or failure to match the asymptotic behaviour. We then substitute the calculated values of $f''(0)$ and $\theta'(0)$ in the Padé approximants of order [20/20] to obtain analytical approximations of the exact closed form solutions of the dimensionless stream function $f(\eta)$, velocity function $f'(\eta)$ and temperature $\theta(\eta)$.

We summarise the algorithms as follows:

1. Initialize all the parameters.
2. Initialize A_0, E_0, f_0 and θ_0.
3. Calculate f_1 and θ_1.
4. Do loop $k = 1$ to 37
   (a) Calculate the Adomian polynomials A_k and E_k.
   (b) Calculate the terms f_{k+1} and θ_{k+1}.
   End loop.
5. Summation of $f_k(\eta)$ and $\theta_k(\eta)$ to obtain the ADM series ϕ_{39} and ω_{39}.
6. Differentiate ϕ_{39} to obtain ϕ'_{39}.
7. Do loop $N = 2$ to 20
   (a) Calculate Padé approximant of order $[N/N]$ of ϕ'_{39}.
   (b) Solve $\lim_{\eta\to\infty} \phi'_{39[N/N]} = 1$.
   End Loop.
8. Determine the converged value as a solution to β_1.
9. Do loop $N = 2$ to 20
   (a) Calculate Padé approximant of order $[N/N]$ of θ_{39}.
   (b) Solve $\theta_{39[N/N]} = 0$ for some intermediate η.
   End Loop.
10. Determine the converged value of β_2 and calculate the Lanalytical approximations for f, f' and θ by substituting β_1 and β_2 in $\phi_{39[20/20]}$, $\phi'_{39[20/20]}$ and $\theta_{39[20/20]}$.
11. Plot graphs.

The algorithms above are straightforward and can be easily coded in Maple commands. Besides several applicable numerical techniques such as Runge-Kutta and the implicit finite-difference schemes of Crank-Nicholson and Keller-box, another recent analytical method which is based on series expansion and much more involved is the homotopy analysis method (HAM). HAM is first developed by Liao [21] and uses base functions to generate series solution together with choices of homotopy value for a converging series. In ADM [5], the initial functions derived from the initial conditions are used to obtain the series expansion and the series is treated by Padé approximants to accelerate convergence for the approximation of the closed form solution.

4 Results and Discussion

From the Maple output lists of 39 terms, for brevity, we list only the first three general terms of f_k and θ_k as follows,

$$f_0 = f_w + \frac{\beta_1}{2}\eta^2 \ , \tag{25}$$

$$f_1 = -\frac{1}{6\mathrm{Pr}}\left(1+\Omega+M^2+f_w\beta_1\right)\eta^3 + \frac{\beta_1}{24\mathrm{Pr}}\left(\Omega+M^2\right)\eta^4 + \frac{\beta_1^2}{120\mathrm{Pr}}\eta^5 \ , \quad (26)$$

$$\begin{aligned} f_2 = {} & \frac{f_w}{24\mathrm{Pr}^2}\left(f_w\beta_1+\Omega+M^2+1\right)\eta^4 - \frac{1}{120\mathrm{Pr}^2}\Big[\Omega\left(\Omega+1\right) \\ & +M^2\left(2\Omega+1+M^2\right)+2f_w\beta_1\left(M^2+\Omega\right)\Big]\eta^5 \\ & +\frac{\beta_1}{720\mathrm{Pr}^2}\Big[M^2\left(M^2+2\Omega-2\right)-2-3\beta_1 f_w-2\Omega+\Omega^2\Big]\eta^6 \\ & +\frac{\beta_1^2}{2520\mathrm{Pr}^2}\left(\Omega+M^2\right)\eta^7 - \frac{\beta_1^3}{40320\mathrm{Pr}^2}\eta^8 \ , \end{aligned} \quad (27)$$

$$\theta_0 = 1+\beta_2\eta \ , \quad (28)$$

$$\theta_1 = -\frac{f_w\beta_2}{2}\eta^2 + \frac{\lambda\beta_1}{6}\eta^3 + \frac{\beta_1\beta_2}{24}\left(2\lambda-1\right)\eta^4 \ , \quad (29)$$

$$\begin{aligned} \theta_2 = {} & \frac{f_w^2\beta_2}{6}\eta^3 - \frac{\lambda}{24\mathrm{Pr}}\left(\Omega+M^2+f_w\beta_1+1+\mathrm{Pr}f_w\beta_1\right)\eta^4 \\ & +\frac{1}{120\mathrm{Pr}}\Big[\beta_2\left(\Omega+M^2-3\lambda+1\right)+\lambda\beta_1\left(\Omega+M^2\right) \\ & -3\lambda\beta_2\left(\Omega+M^2\right)+f_w\beta_1\beta_2\left(4\mathrm{Pr}-\lambda+1-3\lambda\right)\Big]\eta^5 \\ & +\frac{\beta_1}{720\mathrm{Pr}}\Big[\lambda\beta_1\left(1-6\mathrm{Pr}+4\mathrm{Pr}\lambda\right)+\lambda\beta_2\left(4M^2+4\Omega\right)-\beta_2\left(M^2+\Omega\right)\Big]\eta^6 \\ & +\frac{1}{5040\mathrm{Pr}}\beta_1^2\beta_2\left(10\mathrm{Pr}+10\lambda^2\mathrm{Pr}-25\lambda\mathrm{Pr}-1+5\lambda\right)\eta^7 \ . \end{aligned} \quad (30)$$

In order to verify the accuracy of the ADM and Padé approximation, the numerical results of $f''(0)$ and $-\theta'(0)$ or $-\theta'(0)\sqrt{\mathrm{Pr}}$ are tabulated in Tables 1–4 for some prescribed values as representative examples. We compare our results with those of Lin and Lin [22] and Yih [7]. Table 1 shows the numerical results of $\beta_1 = f''(0)$ for $Pr = 1$, $\Omega = 0$, M =0, 1, 2, 5, 10 and $f_w = 1$. The values of $\beta_2 = \theta'(0)$ for various M and f_w when Pr=1, $\Omega = 0$ and $\lambda = 0$ are tabulated in Table 2. We consider the Prandtl number as unity, $\mathrm{Pr} = 1$ where momentum and energy transfer by diffusion are comparable. As M increases, convergence can be seen at the lower order of N and our results and Yih [7] are in well agreement. We also present the values of $f''(0)$ and $-\theta'(0)\sqrt{\mathrm{Pr}}$ from a low Prandtl number to a high Prandtl number as shown in Tables 3 and 4 respectively and our values of $-\theta'(0)\sqrt{\mathrm{Pr}}$ agree with those of Lin and Lin [22] and Yih [7].

Table 1. Values of $\beta_1 = f''(0)$ at Padé $[N/N]$ for various M when $\Omega = 0$, $\mathrm{Pr} = 1$ and $f_w = 1$

M	[13/13]	[17/17]	[19/19]	[20/20]	Yih [7]
0	1.8793745305	1.8972260572	1.8847352359	1.8846843733	1.889314
1	2.2025186785	2.2034463464	2.2027370176	2.2014335458	2.202940
2	2.9201372952	2.9201097536	2.9201142023	2.9201142028	2.920111
5	5.6768303297	5.6768303423	5.6768303422	5.6768303421	5.676830
10	10.5883674769	10.5883674767	10.5883674767	10.5883674767	10.588367

Table 2. Values of $-\theta'(0)$ for $M = 0, 1, 2$ and $f_w = -1, 0, 1$ when $\lambda = \Omega = 0$ and $\Pr = 1$

M	$f_w = -1$		$f_w = 0$		$f_w = 1$	
	Yih [7]	ADM-Padé	Yih [7]	ADM-Padé	Yih [7]	ADM-Padé
0	0.116752	0.11677	0.570465	0.57035	1.323691	1.32368
1	0.140002	0.14000	0.595346	0.59539	1.338060	1.33804
2	0.173124	0.17312	0.634132	0.63418	1.364466	1.36446

Table 3. Values of $\beta_1 = f''(0)$ at Padé $[N/N]$ for various Pr when $\Omega = f_w = M = 0$

Pr	[11/11]	[12/12]
0.0001	124.8293627537	124.8297834328
0.001	39.4745105172	39.4746435463
0.01	12.4829362754	12.4829783429
0.1	3.94745105174	3.94746435479
10	0.3947451051	0.3947464354
100	0.1248293627	0.1248297834
1000	0.0394745105	0.0394746435
10000	0.0124829362	0.0124829783

Table 4. Values of $-\theta'(0)\sqrt{\Pr}$ for various Pr when $\Omega = f_w = M = 0$

Pr	$\lambda = 0$			$\lambda = 1$	
	Lin and Lin [22]	Yih [7]	ADM-Padé	Yih [7]	ADM-Pad'e
0.01	0.075973	0.075973	0.07573	0.116372	0.11610
0.1	0.219505	0.219503	0.21949	0.324927	0.32400
1	0.570466	0.570465	0.57035	0.811301	0.81164
10	1.33880	1.338796	1.33843	1.861577	1.86149
100	2.98634	2.986329	2.98658	4.115021	4.11576
1000	6.52914	6.529137	6.52976	8.963783	8.96368

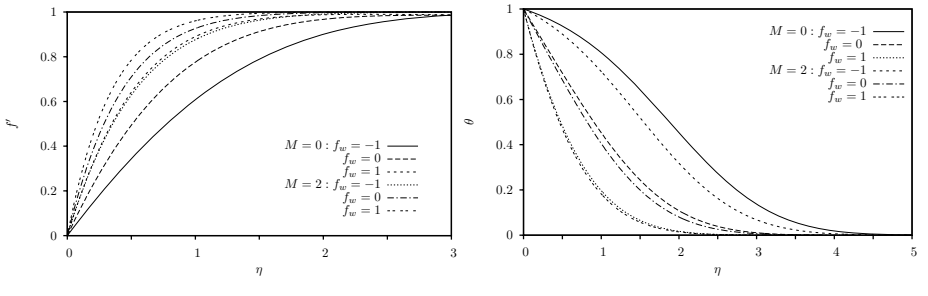


Fig. 1. Velocity profiles f' and temperature profiles θ for $M = 0, 2$ and $f_w = -1, 0, 1$ when $\Omega = \lambda = 0$ and Pr=1

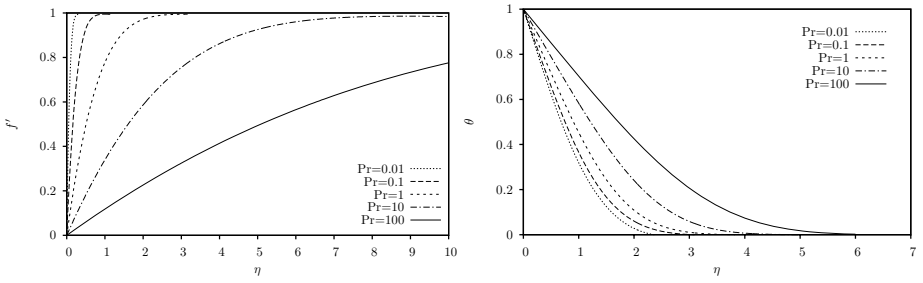


Fig. 2. Velocity profiles f' and temperature profiles θ for various Pr when $M = \Omega = \lambda = f_w = 0$

The non-dimensional velocity $f'(\eta)$ and temperature $\theta(\eta)$ profiles for various parameters $M, \Pr$ and f_w are illustrated in Figs. 1 and 2. Fig. 1 shows the effects of suction/injection parameter and magnetic parameter M. As M or f_w increases, the velocity profile increases and the thermal boundary layer thickness decreases. The magnetic field has a pronounced effect on the temperature distribution for injection while its influence can be neglected in the case of suction. On the effects of suction/injection parameter on the thermal boundary layers, Fig. 1 shows that the temperature profile decreases as f_w increases from injection to suction and leads to the thinning of the thermal boundary layers. The temperature decreases due to suction since the fluid ambient temperature is brought closer to the wall. The heat transfer from the wall into the convecting fluid increases and thus reduces the temperature and the thermal boundary layer thickness. Injection increases the temperature of the fluid and thickens the thermal boundary layer. The action of injection pushes the heated fluid farther from the wall and forms an insulating layer of nearly the same temperature of the wall which decreases the rate of heat transfer from the wall and leads to slower cooling.

Fig. 2 illustrates the influence of the Prandtl number Pr. As Pr diminishes, the velocity profile increases and the temperature profile decreases while lesser variation can be seen in both velocity and temperature profiles. For low Prandtl number $\Pr < 1$, the velocity profile moves closer to the wall and the free-stream velocity exists throughout the boundary layers. These results are therefore less important for highly ionized gases with low Prandtl numbers but quite significant for fluids of high Prandtl numbers such as oils.

5 Conclusions

The Adomian decomposition method and Padé approximant are employed to solve the MHD Hiemenz flow against a flat plate in a porous medium through symbolic manipulations in the Maple system environment. Developed routines yield approximate analytical solutions for momentum and thermal boundary layers and the effects of various influential parameters on the flow are presented. The

numerical results are in well agreement with the existing results and therefore elucidate the reliability and efficiency of the technique. The integrated application of the computer algebra system Maple, the Adomian decomposition method and the Padé approximant is shown to be a useful and powerful analytical tool in solving systems of nonlinear differential equations of boundary-layer flows.

Acknowledgements

The authors gratefully acknowledge the financial supports received from the Universiti Teknologi MARA and the UKM Engineering Mathematics Group.

References

1. Yahaya, F., Hashim, I., Ismail, E.S., Zulkifle, A.K.: Direct Solutions of N-th Order Initial Value Problems in Decomposition Series. Int. J. Nonlinear Sci. Numer. Simul. 8(3), 385–392 (2007)
2. Li, X.F.: Approximate Solution of Linear Ordinary Differential Equations with Variable Coefficients. Math. Comput. Simul. 75, 113–125 (2007)
3. Edneral, V.F.: Looking for Periodic Solutions of ODE Systems by the Normal Form Method. In: Wang, D., Zheng, Z. (eds.) Differential Equations with Symbolic Computations. Trends in Mathematics, pp. 173–200. Birkhäuser Verlag, Basel (2006)
4. Pratibha, J.D.J.: Stokes-Flow Problem Solved Using Maple. In: Sunderam, V.S., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2005. LNCS, vol. 3516, pp. 667–670. Springer, Heidelberg (2005)
5. Adomian, G.: Solving Frontier Problems of Physics: the Decomposition Method. Kluwer Academic, Dordrecht (1994)
6. Baker, G.A.: Essentials of Padé Approximants. Academic Press, New York (1975)
7. Yih, K.A.: The Effect of Uniform Suction/Blowing on Heat Transfer of Magnetohydrodynamic Hiemenz Flow Through Porous Media. Acta Mechanica 130, 147–158 (1998)
8. Hashim, I.: Comments on A New Algorithm for Solving Classical Blasius Equation by L. Wang. Appl. Math. Comput. 176, 700–703 (2006)
9. Hashim, I.: Adomian Decomposition Method for Solving BVPs for Fourth-Order Integro-Differential Equations. J. Comp. Appl. Math. 193, 658–664 (2006)
10. Hashim, I., Noorani, M.S.M., Ahmad, R., Bakar, S.A., Ismail, E.S., Zakaria, A.M.: Accuracy of the Adomian Decomposition Method Applied to the Lorenz System. Chaos Solitons Fractals 28, 1149–1158 (2006)
11. Noorani, M.S.M., Hashim, I., Ahmad, R., Bakar, S.A., Ismail, E.S., Zakaria, A.M.: Comparing Numerical Methods for the Solutions of the Chen System. Chaos, Solitons Fractals 32, 1296–1304 (2007)
12. Zheng, L.C., Chen, X.H., Zhang, X.X., He, J.H.: An Approximately Analytical Solution for the Marangoni Convection in an In-Ga-Sb System. Chin. Phys. Lett. 21, 1983–1985 (2004)
13. Wazwaz, A.M.: The Modified Decomposition Method and Padé Approximants for a Boundary Layer Equation in Unbounded Domain. Appl. Math. Comput. 177, 737–744 (2006)
14. Awang Kechil, S., Hashim, I.: Series Solution for Unsteady Boundary-Layer Flows Due to Impulsively Stretching Plate. Chin. Phys. Lett. 24, 139–142 (2007)

15. Awang Kechil, S., Hashim, I.: Non-perturbative Solution of Free-Convection Boundary-Layer Equation by Adomian Decomposition Method. Phys. Lett. A 363, 110–114 (2007)
16. Awang Kechil, S., Hashim, I., Jiet, S.S.: Approximate Analytical Solutions for a Class of Laminar Boundary-Layer Equations. Chin. Phys. Lett. 24, 1981–1984 (2007)
17. Cherruault, Y.: Convergence of Adomian's Method. Math. Comput. Model. 14, 83–86 (1990)
18. Cherruault, Y., Adomian, G.: Decomposition Method: A New Proof of Convergence. Math. Comput. Model 18, 103–106 (1993)
19. Hosseini, M.M., Nasabzadeh, H.: On the convergence of Adomian decomposition method. Appl. Math. Comput. 182, 536–543 (2006)
20. Boyd, J.P.: Padé Approximant Algorithm for Solving Nonlinear Ordinary Differential Equation Boundary Value Problems on an Unbounded Domain. Comput. Phys. 11, 299–303 (1997)
21. Liao, S.J.: The Proposed Homotopy Analysis Technique for the Solution of Nonlinear Problems. Ph.D. Thesis, Shanghai Jiao Tong University (1992)
22. Lin, H.T., Lin, L.K.: Similarity Solutions for Laminar Forced Convection Heat Transfer from Wedges to Fluids of Any Prandtl Number. Int. J. Heat Mass Transfer 30, 1111–1118 (1987)

Local Similarity Solutions for Laminar Boundary Layer Flow along a Moving Cylinder in a Parallel Stream

Anuar Ishak[1], Roslinda Nazar[1,⋆], and Ioan Pop[2]

[1] School of Mathematical Sciences, Faculty of Science and Technology, Universiti Kebangsaan Malaysia, 43600 UKM Bangi, Selangor, Malaysia
Tel.: +60 3 89213371; Fax: +60 3 89254519
{anuarishak,rmn72my}@yahoo.com
http://www.ukm.my
[2] Faculty of Mathematics, University of Cluj , R-3400 Cluj, CP 253, Romania
pop.ioan@yahoo.co.uk

Abstract. The present paper deals with a numerical method to analyze the axisymmetric boundary layer flow of a viscous and incompressible fluid along a static or moving cylinder, using local similarity approximation. Both parallel and reverse moving boundary to the free stream are considered. Local similarity solutions are obtained to show the effects of the velocity ratio parameter and the curvature parameter on the surface shear stress. The numerical results are comparable very well with the existing results available in the literature for some particular cases of the present problem. Moreover, the results indicate that dual solutions exist when the cylinder and the free stream move in opposite directions.

Keywords: Boundary layer, Dual solutions, Local similarity solutions, Moving cylinder, Numerical solutions.

1 Introduction

The axisymmetric boundary layer flow along static cylinders with constant surface temperature was first considered by Seban and Bond [1], using a perturbation method, while for moving cylinders, it was first studied by Sakiadis [2], using Karman-Pohlhausen approximate method. The laminar boundary layer flow and heat transfer along a static and moving cylinders with constant velocity was considered by Lin and Shih [3], who found that this problem does not admit similarity solution. The similar problem was then extended by the same authors [4], to vertically moving cylinder in which the buoyancy effects have to be taken into account. These problems [3,4] are different from those of linearly stretching cylinder considered by Ishak et al. [5], and a static cylinder in a moving stream with a linear velocity along the axial direction of the cylinder considered by Mahmood and Merkin [6] and Ishak et al. [7], who found that the similarity solutions do exist for those cases.

⋆ Corresponding author.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 224–235, 2008.
© Springer-Verlag Berlin Heidelberg 2008

The local similarity assumption introduced by Lloyd and Sparrow [8] was used by Lin and Shih [3,4] to solve the above mentioned problems. This assumption was also used by Narain and Uberoi [9] to solve the problem of mixed convection from vertical thin needles in a uniform stream, and by the same authors [10] to a non-uniform free stream, and found that similarity solutions exist when the free stream varies like $x^{1/2}$ for isothermal thin needle, and $x^{2/3}$ for uniform wall heat flux needle, where x is the axial coordinate.

Motivated by the above investigations, we present in this paper the local similarity solutions for laminar boundary layer flow along a fixed or moving cylinder in a viscous and incompressible fluid, which moves in the same or opposite direction to the cylinder. The governing partial differential equation is converted into ordinary differential equation by the local similarity approximation, before it is solved numerically by a finite-difference method. The numerical results obtained are compared with the results available in the literature for two particular cases of the present problem, namely static cylinder in a moving fluid, and moving cylinder in a quiescent fluid considered by Lin and Shih [3].

2 Problem Formulation

Consider a steady, axisymmetric boundary layer flow of a viscous and incompressible fluid along a continuously moving cylinder as shown in Fig. 1. It is assumed that the cylinder moves with a constant velocity U_w in a parallel stream of constant velocity U_∞. Both cases when the cylinder and the free stream move in the same and opposite directions are considered. Under these assumptions along with the boundary layer approximations, the equations which model the problem under consideration are

$$\frac{\partial}{\partial x}(ru) + \frac{\partial}{\partial r}(rv) = 0, \tag{1}$$

$$u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial r} = \frac{\nu}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right), \tag{2}$$

subject to the boundary conditions

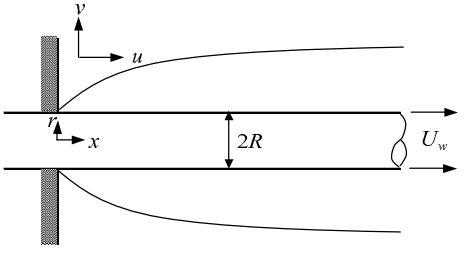


Fig. 1. Physical model and coordinate system

$$u = U_w,\ v = 0 \quad \text{at} \quad r = R,$$

$$u \to U_\infty \quad \text{as} \quad r \to \infty, \tag{3}$$

where u and v are the velocities in the x and y directions, respectively, ν is the kinematic viscosity, and R is the radius of the cylinder. This problem does not admit similarity solutions due to the curvature effect of the cylinder [3]. To solve the present problem, we define the following dimensionless groups and variables (cf. [3,4]):

$$\xi(x) = \frac{4}{R}\left(\frac{\nu x}{U}\right)^{1/2},\ \eta(x,r) = \frac{r^2 - R^2}{\xi R^2},\ \psi(x,r) = R(\nu x U)^{1/2} f(\xi,\eta), \tag{4}$$

where $U = U_w + U_\infty$, and ψ is a stream function such that $u = r^{-1}\partial\psi/\partial r$ and $v = -r^{-1}\partial\psi/\partial x$ which identically satisfy Eq. (1). Using Eq. (4), Eq. (2) becomes

$$(1+\xi\eta)\, f''' + (\xi + f)\, f'' + \xi\left(f''\frac{\partial f}{\partial \xi} - f'\frac{\partial f'}{\partial \xi}\right) = 0, \tag{5}$$

where primes denote differentiation with respect to η.

3 Solution Procedure

3.1 Local Similarity Assumption

When the value of ξ and (or) the values of $\partial f/\partial\xi$ and $\partial f'/\partial\xi$ are small, the terms containing partial derivatives with respect to ξ in Eq. (5) can be neglected. This approach is called local similarity assumption [3,4,8]. Thus, the local similarity solution of Eq. (5) subjected to the appropriate boundary conditions is obtained by deleting the terms containing partial derivatives with respect to ξ, and considers ξ as a parameter. By employing this assumption, Eq. (5) reduces to

$$(1+\xi\eta)\, f''' + (\xi + f)\, f'' = 0, \tag{6}$$

and the transformed boundary conditions are

$$f'(0) = 2\lambda,\ f(0) = 0,\ f'(\infty) = 2(1-\lambda), \tag{7}$$

where $\lambda = U_w/U$. We notice that $\lambda = 0$ corresponds to a static cylinder in a moving fluid, and $\lambda = 1$ corresponds to a moving cylinder in a quiescent fluid. Both cases of $\lambda = 0$ and $\lambda = 1$ were considered by Lin and Shih [3], for which the present results can be compared with.

3.2 Finite-Difference Method

To solve the transformed differential equation (6) subject to the boundary conditions (7), Eq. (6) is first converted into a system of three first-order equations, and the difference equations are then expressed using central differences.

For this purpose, we introduce new dependent variables $p(\eta)$ and $q(\eta)$ so that Eq. (6) can be written as

$$f' = p, \tag{8}$$

$$p' = q, \tag{9}$$

$$(1 + \xi\eta)\, q' + (\xi + f)\, q = 0. \tag{10}$$

In terms of the new dependent variables, the boundary conditions (7) are given by

$$p(0) = 2\lambda,\ f(0) = 0,\ p(\infty) = 2(1 - \lambda). \tag{11}$$

We now consider the segment $\eta_{j-1}\eta_j$, with $\eta_{j-1/2}$ as the midpoint, and is defined as below:

$$\eta_0 = 0, \quad \eta_j = \eta_{j-1} + h_j\ (j = 1, 2, \cdots, J), \quad \eta_J = \eta_\infty, \tag{12}$$

where h_j is the $\Delta\eta$-spacing and j is a sequence number that indicates the coordinate location. The finite-difference approximations to the ordinary differential equations (8)–(10) are written for the midpoint $\eta_{j-1/2}$ of the segment $\eta_{j-1}\eta_j$. This procedure gives

$$\frac{f_j - f_{j-1}}{h_j} = \frac{p_j + p_{j-1}}{2} = p_{j-1/2}, \tag{13}$$

$$\frac{p_j - p_{j-1}}{h_j} = \frac{q_j + q_{j-1}}{2} = q_{j-1/2}, \tag{14}$$

$$\left(1 + \xi\eta_{j-1/2}\right) \frac{q_j - q_{j-1}}{h_j} + \left(\xi + f_{j-1/2}\right) q_{j-1/2} = 0. \tag{15}$$

Rearranging expressions (13)–(15) give

$$f_j - f_{j-1} - \frac{1}{2} h_j\, (p_j + p_{j-1}) = 0, \tag{16}$$

$$p_j - p_{j-1} - \frac{1}{2} h_j\, (q_j + q_{j-1}) = 0, \tag{17}$$

$$(1 + \xi\gamma)\, (q_j - q_{j-1}) + \frac{1}{2} h_j \xi (q_j + q_{j-1}) + \frac{1}{4} h_j (f_j + f_{j-1})(q_j + q_{j-1}) = 0, \tag{18}$$

where $\gamma = (\eta_j + \eta_{j-1})/2$ and $(\ )_{j-1/2} = [(\ )_j + (\ )_{j-1}]/2$.

Equations (16)–(18) are imposed for $j = 1, 2, 3, \cdots, J$, and the transformed boundary layer thickness η_J is to be sufficiently large so that it is beyond the edge of the boundary layer. The boundary conditions yield are

$$f_0 = 0,\ p_0 = 2\lambda,\ p_J = 2(1 - \lambda). \tag{19}$$

3.3 Newton's Method

To solve the nonlinear system (16)–(18), we use Newton's method, by introducing the following iterates:

$$f_j^{(k+1)} = f_j^{(k)} + \delta f_j^{(k)}, \quad p_j^{(k+1)} = p_j^{(k)} + \delta p_j^{(k)}, \quad q_j^{(k+1)} = q_j^{(k)} + \delta q_j^{(k)}, \tag{20}$$

where $k = 0, 1, 2, \cdots$. We then insert the left-hand side expressions in place of f_j, p_j and q_j in Eqs. (16)–(18) and drop the terms that are quadratic in $\delta f^{(k)}$, $\delta p^{(k)}$ and $\delta q^{(k)}$. This procedure yields the following linear system (the superscript k is dropped for simplicity):

$$\delta f_j - \delta f_{j-1} - \frac{h_j}{2}\left(\delta p_j + \delta p_{j-1}\right) = (r_1)_{j-1/2}\,, \tag{21}$$

$$\delta p_j - \delta p_{j-1} - \frac{h_j}{2}\left(\delta q_j + \delta q_{j-1}\right) = (r_2)_{j-1/2}\,, \tag{22}$$

$$(a_1)\delta q_j + (a_2)\delta q_{j-1} + (a_3)\delta f_j + (a_4)\delta f_{j-1} = (r_3)_{j-1/2}, \tag{23}$$

where

$$(a_1)_j = 1 + \xi\gamma + \frac{1}{2}\xi h_j + \frac{1}{2}h_j f_{j-1/2}, \quad (a_2)_j = (a_1)_j - 2(1+\xi\gamma)$$

$$(a_3)_j = \frac{1}{2}h_j q_{j-1/2}, \quad (a_4)_j = (a_3)_j, \tag{24}$$

and

$$(r_1)_{j-1/2} = -f_j + f_{j-1} + h_j p_{j-1/2}, \quad (r_2)_{j-1/2} = -p_j + p_{j-1} + h_j q_{j-1/2},$$

$$(r_3)_{j-1/2} = -(1+\xi\gamma)(q_j - q_{j-1}) - \xi h_j q_{j-1/2} - h_j (fq)_{j-1/2}. \tag{25}$$

The boundary conditions (19) become

$$\delta f_0 = 0,\ \delta p_0 = 0,\ \delta p_J = 0, \tag{26}$$

which express the requirement for the boundary conditions to remain constant during the iteration process.

3.4 Block-Elimination Method

The linearized difference equations (21)–(23) can be solved by the block-elimination method as outlined by Na [11] and Cebeci and Bradshaw [12], since the system has block-tridiagonal structure. Commonly, the block-tridiagonal structure consists of variables or constants, but here an interesting feature can be observed that it consists of block matrices. In a matrix-vector form, Eqs. (21)–(23) can be written as

$$\mathbf{A}\boldsymbol{\delta} = \mathbf{r} \tag{27}$$

where

$$\mathbf{A} = \begin{bmatrix} [A_1] & [C_1] & & & & \\ [B_2] & [A_2] & [C_2] & & & \\ & \ddots & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & [B_{J-1}] & [A_{J-1}] & [C_{J-1}] \\ & & & & [B_J] & [A_J] \end{bmatrix}, \quad \boldsymbol{\delta} = \begin{bmatrix} [\delta_1] \\ [\delta_2] \\ \vdots \\ [\delta_{J-1}] \\ [\delta_J] \end{bmatrix},$$

and

$$\mathbf{r} = \begin{bmatrix} [r_1] \\ [r_2] \\ \vdots \\ [r_{J-1}] \\ [r_J] \end{bmatrix}.$$

The elements of the matrices are as follows:

$$[A_1] = \begin{bmatrix} 0 & 1 & 0 \\ -\frac{1}{2}h_1 & 0 & -\frac{1}{2}h_1 \\ (a_2)_1 & (a_3)_1 & (a_1)_1 \end{bmatrix}, \tag{28}$$

$$[A_j] = \begin{bmatrix} -\frac{1}{2}h_j & 1 & 0 \\ -1 & 0 & -\frac{1}{2}h_j \\ 0 & (a_3)_j & (a_1)_j \end{bmatrix}, \ 2 \le j \le J, \tag{29}$$

$$[B_j] = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -\frac{1}{2}h_j \\ 0 & (a_4)_j & (a_2)_j \end{bmatrix}, \ 2 \le j \le J, \tag{30}$$

$$[C_j] = \begin{bmatrix} -\frac{1}{2}h_j & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ 1 \le j \le J-1, \tag{31}$$

$$[\delta_1] = \begin{bmatrix} \delta q_0 \\ \delta f_1 \\ \delta q_1 \end{bmatrix}, \ [\delta_j] = \begin{bmatrix} \delta p_{j-1} \\ \delta f_j \\ \delta q_j \end{bmatrix}, \ 2 \le j \le J, \tag{32}$$

and

$$[r_j] = \begin{bmatrix} (r_1)_{j-1/2} \\ (r_2)_{j-1/2} \\ (r_3)_{j-1/2} \end{bmatrix}, \ 1 \le j \le J. \tag{33}$$

To solve Eq. (27), we assume that $\mathbf{A}$ is nonsingular and it can be factorized as

$$\mathbf{A} = \mathbf{LU}, \tag{34}$$

where

$$\mathbf{L} = \begin{bmatrix} [\alpha_1] & & & & \\ [B_2] & [\alpha_2] & & & \\ & \ddots & & & \\ & & \ddots & [\alpha_{J-1}] & \\ & & & [B_J] & [\alpha_J] \end{bmatrix} \text{ and } \mathbf{U} = \begin{bmatrix} [I] & [\Gamma_1] & & & \\ & [I] & [\Gamma_2] & & \\ & & \ddots & & \\ & & \ddots & & \\ & & & [I] & [\Gamma_{J-1}] \\ & & & & [I] \end{bmatrix},$$

where $[I]$ is a 3×3 identity matrix, while $[\alpha_i]$ and $[\Gamma_i]$ are 3×3 matrices in which the elements are determined by the following equations:

$$[\alpha_1] = [A_1], \tag{35}$$

$$[A_1]\,[\Gamma_1] = [C_1], \tag{36}$$

$$[\alpha_j] = [A_j] - [B_j]\,[\Gamma_{j-1}], \ j = 2, 3, \cdots, J, \tag{37}$$

$$[\alpha_j]\,[\Gamma_j] = [C_j], \ j = 2, 3, \cdots, J-1. \tag{38}$$

Substituting Eq. (34) into Eq. (27), we obtain

$$\mathbf{LU}\boldsymbol{\delta} = \mathbf{r}. \tag{39}$$

If we define

$$\mathbf{U}\boldsymbol{\delta} = \mathbf{W}, \tag{40}$$

Eq. (39) becomes

$$\mathbf{LW} = \mathbf{r}, \tag{41}$$

where

$$\mathbf{W} = \begin{bmatrix} [W_1] \\ [W_2] \\ \vdots \\ [W_{J-1}] \\ [W_J] \end{bmatrix},$$

and $[W_j]$ are 3×1 column matrices. The elements of $\mathbf{W}$ can be determined from Eq. (40) by the following relations:

$$[\alpha_1]\,[W_1] = [r_1], \tag{42}$$

$$[\alpha_j]\,[W_j] = [r_j] - [B_j]\,[W_{j-1}], \ 2 \le j \le J. \tag{43}$$

When the elements of $\mathbf{W}$ have been found, Eq. (40) gives the solution for $\boldsymbol{\delta}$ in which the elements are found from the following relations:

$$[\delta_J] = [W_J], \tag{44}$$

$$[\delta_j] = [W_j] - [\Gamma_j]\,[\delta_{j+1}]\,,\ 1 \le j \le J-1. \tag{45}$$

Once the elements of $\boldsymbol{\delta}$ are found, Eqs. (21)–(23) can be used to find the $(k+1)$th iteration in Eq. (20). These calculations are repeated until the convergence criterion is satisfied. In laminar boundary layer calculation, the wall shear stress parameter $q(0)$ is commonly used as the convergence criterion [13]. This is probably because in boundary layer calculations, it is found that the greatest error usually appears in the wall shear stress parameter. Thus, this convergence criterion is used in the present study. Calculations are stopped when

$$\left|\delta q_0^{(k)}\right| < \epsilon_1, \tag{46}$$

where ϵ_1 is a small prescribed value. In this study, $\epsilon_1 = 0.0000001$ is used, which gives about six decimal places accuracy for most predicted quantities.

Table 1. Values of $f''(0)$ for the flow along a static cylinder ($\lambda = 0$), and a moving cylinder ($\lambda = 1$)

	Lin and Shih [3]		Present results	
ξ	$\lambda = 0$	$\lambda = 1$	$\lambda = 0$	$\lambda = 1$
0	1.32823	−1.77497	1.328229	−1.774973
0.0001	1.32829	−1.77501	1.328286	−1.775009
0.0005	1.32852	−1.77514	1.328515	−1.775139
0.001	1.32880	−1.77529	1.328801	−1.775292
0.005	1.33108	−1.77654	1.331084	−1.776544
0.01	1.33393	−1.77810	1.333934	−1.778103
0.05	1.35648	−1.79077	1.356475	−1.790768
0.1	1.38449	−1.80622	1.384492	−1.806219
0.5	1.59496	−1.93135	1.594957	−1.931348
1.0	1.83485	−2.10325	1.834846	−2.103250
1.5	2.05683	−2.24323	2.056828	−2.243225
2.0	2.26609	−2.41258	2.266093	−2.412581

It is worth mentioning that the step size $\Delta\eta$ in η, and the position of the edge of the boundary layer η_∞ have to be adjusted for different values of parameters to maintain accuracy. In this study, the values of $\Delta\eta$ between 0.001 and 0.1 were used, depending on the values of the parameters used, in order that the numerical values obtained are independent of $\Delta\eta$ chosen, at least to six decimal places. However, a uniform grid of $\Delta\eta = 0.001$ was found to be satisfactory for a convergence criterion of 10^{-7} which gives accuracy to six decimal places, in nearly all cases. On the other hand, the boundary layer thickness η_∞ between 3 and 50 was chosen where the infinity boundary condition is achieved. For some values of the parameters, there is a possibility that two values of η_∞ are obtained which gives two different velocity profiles, and in consequence produces two different values of the surface shear stress, as shown in Fig. 2. The existence of the dual solutions is supported by the nature of the velocity profiles shown in Fig. 3, where two different profiles are obtained for one value of λ and ξ. To

assess the accuracy of the present method, comparisons with previously reported data by Lin and Shih [3] are made for several particular values of parameters, as given in Table 1, which shows an excellent agreement.

The present method has second-order accuracy, unconditionally stable and is easy to be programmed, thus making it highly attractive for production use. The only disadvantage is the large amount of once-and-for-all algebra needed to write the difference equations and to set up their solutions [14].

4 Results and Discussion

The exact solution of Eq. (6) subjected to (7) is not available, and thus a numerical method has to be used. We study the effects of the transverse curvature parameter ξ and the velocity ratio parameter λ on the values of $f''(0)$, which represent the surface shear stress.

Fig. 2 shows that the solution is unique when $\lambda \geq 0$, while dual solutions are found to exist for certain ranges of $\lambda < 0$, i.e. when the cylinder and the free stream move in the opposite directions. The values of $f''(0)$ for the lower branch solutions for selected values of ξ and λ are given in Table 2. From Fig. 2, it is seen that the values of $f''(0)$ are positives when $\lambda < 0.5$, and they become negatives when the value of λ exceeds 0.5, for both $\xi = 0$ and $\xi = 1$. Physically, positive value of $f''(0)$ means that the fluid exerts a drag force on the surface of the cylinder, and negative value means the opposite. The zero value of $f''(0)$ when $\lambda = 0.5$ does not mean separation, but it corresponds to the equal velocity of the cylinder and the free stream. The results reported by Lin and Shih [3] (as given in Table 1) are also included in this figure, and they are found to be in good agreement with the present results.

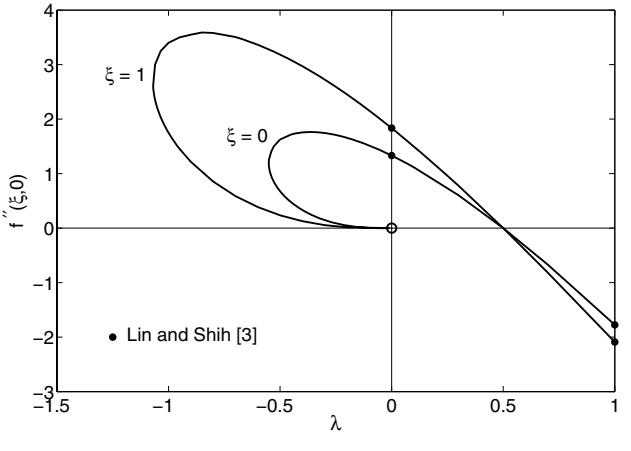


Fig. 2. Skin friction coefficient $f''(\xi, 0)$ as a function of λ when $\xi = 0$ and $\xi = 1$

Table 2. Values of $f''(0)$ for the lower branch solutions, for selected values of ξ and λ

ξ	$\lambda = -0.5$	$\lambda = -0.8$	$\lambda = -1$
0	0.684124	–	–
0.5	0.314326	1.562556	–
1	0.226594	0.857920	1.762304

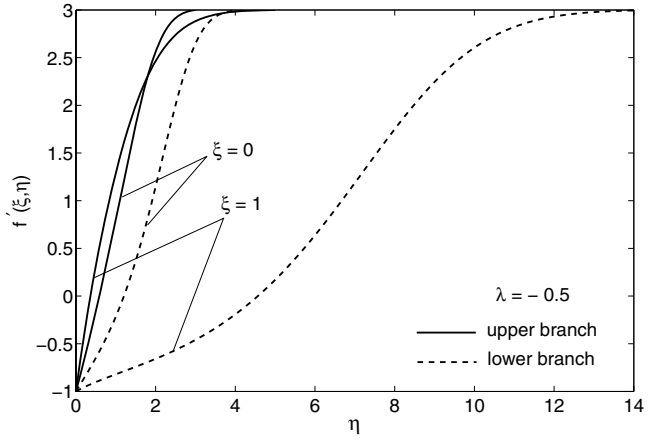


Fig. 3. Velocity profiles $f'(\xi, \eta)$ for $\xi = 0$ and $\xi = 1$ when $\lambda = -0.5$

For a particular value of ξ, the solution exists up to a certain critical value of λ, say λ_c, beyond which the boundary layer approximations breakdown, and thus no solution is obtained. To get further solution, the full Navier-Stokes equations have to be solved. The boundary layer separated from the surface at $\lambda = \lambda_c$. Based on our computations, $\lambda_c = -0.5505$ and -1.0691 for $\xi = 0$ and 1, respectively. In contrast with the classical boundary layer theory, the separation occurs when the skin friction coefficient $f''(0) > 0$, and not at the point of vanishing wall-shear, $f''(0) = 0$. This finding is in agreement with those reported by Schneider [15], Schneider and Wasel [16] and Ishak et al. [17], for the problems of mixed convection above a horizontal plate with the buoyancy force is taken into account. Moreover, Sears and Telionis [18] suggested that the name "separation" should not be given to vanishing wall-shear.

We identify the upper and lower branch solutions in the following discussion by how they appear in Fig. 2, i.e. the upper branch solution has higher values of $f''(0)$ for given values of ξ and λ than the lower branch solution. It is not possible to determine which solution would occur in practice since a stability analysis has not been carried out. However, we expect the upper branch solution to be stable and physically relevant, whereas the lower branch solution is unstable and not physically relevant, since it is the only solution for the case $\lambda \geq 0$. The saddle-node bifurcation at $\lambda = \lambda_c$ corresponds to a change in the (temporal) stability of the solution and, unless there is a change in stability on the upper branch for

$\lambda \neq \lambda_c$, the saddle-node bifurcation gives a change in stability from stable (upper branch) to unstable (lower branch). Although the lower branch solutions seem to deprive of physical significance, they are nevertheless of interest so far as the differential equations are concerned. Similar results may arise in other situations where the corresponding solutions have more realistic meaning [19,20].

Finally, the velocity profiles presented in Fig. 3 show that the boundary conditions (7) are satisfied, and thus support the numerical results obtained, besides supporting the dual nature of the solutions when $\lambda = -0.5$.

Acknowledgement. The financial supports received from the Universiti Kebangsaan Malaysia (Engineering Mathematics Group) under the Research University grants (UKM-OUP-BTT-25/2007 and UKM-GUP-BTT-07-25-174) are gratefully acknowledged.

References

1. Seban, R.A., Bond, R.: Skin Friction and Heat Transfer Characteristics of a Laminar Boundary Layer on a Cylinder in Axial Incompressible Flow. J. Aeronaut. Sci. 18, 671–675 (1951)
2. Sakiadis, B.C.: Boundary Layer Behavior on Continuous Solid Surfaces: III. The Boundary Layer on a Continuous Cylindrical Surface. A.I.Ch.E. J. 7, 467–472 (1961)
3. Lin, H.-T., Shih, Y.-P.: Laminar Boundary Layer Heat Transfer along Static and Moving Cylinders. J. Chin. Ins. Eng. 3, 73–79 (1980)
4. Lin, H.-T., Shih, Y.-P.: Buoyancy Effects on the Laminar Boundary Layer Heat Transfer along Vertically Moving Cylinders. J. Chin. Ins. Eng. 4, 47–51 (1981)
5. Ishak, A., Nazar, R., Pop, I.: Uniform Suction/Blowing Effect on Flow and Heat Transfer due to a Stretching Cylinder. Appl. Math. Modell. 32, 2059–2066 (2008)
6. Mahmood, T., Merkin, J.H.: Similarity Solutions in Axisymmetric Mixed-Convection Boundary-Layer Flow. J. Eng. Math. 22, 73–92 (1988)
7. Ishak, A., Nazar, R., Pop, I.: The Effects of Transpiration on the Boundary Layer Flow and Heat Transfer over a Vertical Slender Cylinder. Int. J. Non-Linear Mech. 42, 1010–1017 (2007)
8. Lloyd, J.R., Sparrow, E.M.: Combined Forced and Free Convection Flow on Vertical Surfaces. Int. J. Heat Mass Transfer 13, 434–438 (1970)
9. Narain, J.P., Uberoi, M.: Combined Forced and Free-Convection Heat Transfer from Vertical Thin Needles in a Uniform Stream. Phys. Fluid 15, 1879–1882 (1972)
10. Narain, J.P., Uberoi, M.: Combined Forced and Free-Convection over Thin Needles. Int. J. Heat Mass Transfer 16, 1505–1512 (1973)
11. Na, T.Y.: Computational Methods in Engineering Boundary Value Problems. Academic Press, New York (1979)
12. Cebeci, T., Bradshaw, P.: Physical and Computational Aspects of Convective Heat Transfer. Springer, New York (1988)
13. Cebeci, T., Bradshaw, P.: Momentum Transfer in Boundary Layers. Hemisphere, Washington (1977)
14. Cebeci, T., Chang, K.C., Bradshaw, P.: Solution of a Hyperbolic System of Turbulence-Model Equations by the "BOX" Scheme. Comp. Method Appl. Mech. Eng. 22, 213–227 (1980)

15. Schneider, W.: A Similarity Solution for Combined Forced and Free Convection Flow over a Horizontal Plate. Int. J. Heat Mass Transfer 22, 1401–1406 (1979)
16. Schneider, W., Wasel, M.G.: Breakdown of the Boundary-Layer Approximation for Mixed Convection above a Horizontal Plate. Int. J. Heat Mass Transfer 28, 2307–2313 (1985)
17. Ishak, A., Nazar, R., Pop, I.: The Schneider Problem for a Micropolar Fluid. Fluid Dyn. Res. 38, 489–502 (2006)
18. Sears, W.R., Teleonis, D.P.: Boundary-Layer Separation in Unsteady Flow. SIAM J. Appl. Math. 28, 215–235 (1975)
19. Ridha, A.: Aiding Flows Non-Unique Similarity Solutions of Mixed-Convection Boundary-Layer Equations. J. Appl. Math. Phys (ZAMP) 47, 341–352 (1996)
20. Ridha, A.: Three-Dimensional Mixed Convection Laminar Boundary-Layer near a Plane of Symmetry. Int. J. Eng. Sci. 34, 659–675 (1996)

An Algorithm for Transforming Regular Chain into Normal Chain*

Banghe Li and Dingkang Wang

Key Laboratory of Mathematics Mechanization
Academy of Mathematics and Systems Science
Chinese Academy of Sciences
Beijing 100080, China
libh@amss.ac.cn, dwang@mmrc.iss.ac.cn

Abstract. We present an improved algorithm to compute the normal chain from a given regular chain such that their saturation ideals are the same. Our algorithm is based on solving a system of linear equations and the original method computes the resultants of multivariate polynomials. From the experiments, for the random polynomials, our algorithm is much more efficient than the original one.

1 Introduction

Characteristic set method has been successfully applied to automatic theorem proving by Wu [11]. In fact, this method also can be used for solving systems of polynomial equations. In order to solve a system of polynomial equations, the polynomial system should be decomposed into chains. Wu himself proposed an algorithm to compute such decompositions. The regular zeros of the chain in the decomposition maybe empty and some redundant components may be introduced by using Wu's method. Yang introduced regular chain and the regular zeros of a regular chain should not be empty [12]. Both Yang and Kalbrener presented algorithms to decompose a system of polynomials into a series of regular chains such that the zeros of the system of polynomials are the union of the regular zeros of the regular chains. Efficient algorithms to decompose a system of polynomials into regular chains have been proposed by Moreno [6] and Wang [10].

In [3], Gao introduced the concept of p-chain in order to solve systems of equations of parametric polynomials. To avoid the confusion, we will rename the p-chain as normal chain in this paper. To compute the normal chain from a given regular chain, the resultants of multivariate polynomials will be computed and the computation of resultants will be too costly. In [9], normal chain is introduced and an algorithm is proposed to compute a normal chain from a chain if it does not fail. An algorithm to decompose polynomial system into normal chains is given in [8].

* Partially supported by NKBRPC (2004CB318000) and NSFC (10771206).

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 236–245, 2008.
© Springer-Verlag Berlin Heidelberg 2008

In all the existed algorithms to compute the normal chain from a regular chain, the computation of polynomial resultant is needed and resultant computation of polynomials is quite cost.

In this paper, we will present a new algorithm to compute the normal chain from a regular chain. Our algorithm is based on solving system of linear equations. It is not needed to compute the resultants of multivariate polynomials in our algorithm and the experiment results show that our algorithm is much more efficient than the original one.

After giving some preliminaries in section 2, we will give an algorithm to compute the inverse of a uni-variable polynomial modulo another uni-variable polynomial. A table to record the timings for computing the inverses of random polynomials is given in section 3. An algorithm will be given to compute a normal chain from a regular chain such that they have the same saturation ideal in section 4. An example to decompose the Lorentz polynomial system into normal chains will be reported in section 5. The conclusions will be given in the last section.

2 Preliminaries

Let $\mathrm{K}[u_1, \cdots, u_p, y_1, \cdots, y_s]$ be the polynomial ring with $u_1, \cdots, u_p, y_1, \cdots, y_s$ as indeterminates and coefficients in a field K. Let $\mathrm{U} = \{u_1, \cdots, u_p\}$. $\mathrm{Y} = \{y_1, \cdots, y_s\}$. $\mathrm{K}[u_1, \cdots, u_p, y_1, \cdots, y_s]$ is denoted by $\mathrm{K}[\mathrm{U}, \mathrm{Y}]$. In this paper, we always assume that the variable ordering is $u_1 < \cdots < u_p < y_1 < \cdots < y_s$.

Let E be an algebraic closed extension field containing K. For a polynomial set $\mathbb{F}$, $(\mathbb{F})$ denotes the ideal generated by $\mathbb{F}$ over the ring $\mathrm{K}[\mathrm{U}, \mathrm{Y}]$. $\mathrm{Zero}(\mathbb{F})$ denotes the common zeros in $\mathrm{E}^{(p+s)}$ of the polynomials in $\mathbb{F}$. Let D be a polynomial, $\mathrm{Zero}(\mathbb{F}/D)$ denotes the common zeros in $\mathrm{E}^{(p+s)}$ of the polynomials in $\mathbb{F}$ which are not zeros of D.

For any nonzero polynomial P, the leading variable of P is denoted as v_P, the leading coefficient of P with respect to v_P is called the initial of P, denoted by $\mathrm{I}(P)$. We denote $\deg(P, y_i)$ the degree of P w.r.t. y_i.

Let $\mathcal{A} : A_1, \cdots, A_s$ be a chain and the leading variables of A_i is y_i. We will use $I_{\mathcal{A}}$ to denote the product of the initials of the polynomials in $\mathcal{A}$, i.e. $I_{\mathcal{A}} = \prod_{i=1}^{s} \mathrm{I}(A_i)$.

For two univariable polynomials P and Q, the remainder of P divided by Q w.r.t. y will be denoted by $\mathrm{rem}(P, Q, y)$. If P and Q are multivariable polynomials, the psudoremainder of P divided by Q w.r.t. y_i will be denoted by $\mathrm{prem}(P, Q, y_i)$. For two polynomials P, Q , the Sylvester resultant of P and Q with respect to y_i is denoted by $\mathrm{res}(P, Q, y_i)$.

Definition 1. *Let P be a polynomial, $\mathcal{A} = A_1, \cdots, A_s$ be a chain with y_i as the leading variable of A_i. Let $R_s = P$, $R_{i-1} = \mathrm{res}(R_i, A_i, y_i)$ for $i = s, \cdots, 1$. R_0 is called the resultant of P with respect to $\mathcal{A}$, denoted by* $\mathrm{Res}(P; \mathcal{A})$.

It is easy to see that R_0 is in $K[u_1, \cdots, u_p]$. There are polynomials F, G_i for $i = 1, \cdots, s$ such that

$$FP + \sum_{i=1}^{s} G_i A_i = \mathrm{Res}(P; \mathcal{A}) \tag{1}$$

Definition 2. *Suppose $\mathcal{A}$ is a chain, if $\xi \in E^n$ and $\xi \in Zero(\mathcal{A}/I_{\mathcal{A}})$, then ξ is called a regular zero of $\mathcal{A}$.*

The regular zeros of a chain maybe empty.

Let $\mathcal{A} = A_1, \cdots, A_s$ be a chain, the ideal generated by $\mathcal{A}$ over $\mathrm{K}[\mathrm{U}, y_1, \cdots, y_s]$ will be denoted by $(\mathcal{A})$. Let $\mathcal{A}_i = A_1, \cdots, A_i$ for $1 \leq i \leq s$, each $\mathcal{A}_i$ is also a chain and the ideal generated by $\mathcal{A}_i$ over $\mathrm{K}[\mathrm{U}, y_1, \cdots, y_i]$ will be denoted by $(\mathcal{A}_i)$.

Definition 3. *Let $\mathcal{A} = A_1, \cdots, A_s$ be a chain in* $\mathrm{K}[\mathrm{U}, \mathrm{Y}]$ *and P be a polynomial in* $\mathrm{K}[\mathrm{U}, \mathrm{Y}]$. *$P$ is said to be invertible w.r.t. $\mathcal{A}$ if $(\mathcal{A}, P) \cap \mathrm{K}[\mathrm{U}] \neq \{0\}$*

If P is invertible w.r.t. $\mathcal{A}$, then there exist Q in $\mathrm{K}[\mathrm{U}, \mathrm{Y}]$ and $M \neq 0$ in $\mathrm{K}[\mathrm{U}]$ such that $PQ \equiv M \mod (\mathcal{A})$

An algorithm to test if a polynomial is invertible with respect to a chain is given in [2]. Procedures to compute the inverse of a polynomial with respect to a chain are given in [2,5,6].

Definition 4. *Let $\mathcal{A} = A_1, \cdots, A_s$ be a chain. Let $\mathcal{A}_i = A_1, \cdots, A_i$ for $i = 1, \cdots s$. $\mathcal{A}$ is a regular chain if s=1 or* $\mathrm{Res}(\mathrm{I}(A_i); \mathcal{A}_{i-1}) \neq 0$ *for $i = 2, \cdots, s$.*

The regular chain is introduced by Yang et. al. in [12]. The above definition implies that the regular zeros of a regular chain are not empty.

Definition 5. *Let $\mathcal{A} = A_1, \cdots, A_s$ be a chain and $\xi \in E^{(p+s)}$ be a zero of $\mathcal{A}$. $\xi = (\xi_1, \cdots, \xi_p, \xi_{p+1}, \cdots, \xi_{p+s})$, ξ is called to be a generic regular zero of $\mathcal{A}$ if $(\xi_1, \cdots, \xi_p)$ are algebraically independent over* K.

The following theorem establishes the relationship between regularity of a chain and invertibility of its initials.

Theorem 1. *Let $\mathcal{A} = A_1, \cdots, A_s$ be a chain, the following statements are equivalent:*

1. *$\mathcal{A}$ is a regular chain*
2. *For $i = 1, \cdots, s$,* $\mathrm{I}(A_i)$ *is invertible w.r.t. $\mathcal{A}$.*
3. *For any generic regular point ξ,* $\mathrm{I}(A_i)(\xi) \neq 0$ *for $i = 1, \cdots, s$.*

Please see [2] or [10] for the proof of the theorem.

Definition 6. *A chain $\mathcal{A} = A_1, \cdots, A_s$ is called a normal chain if* $\mathrm{I}(A_i)$ *is in* $\mathrm{K}[\mathrm{U}]$ *for $i = 1, \cdots, s$.*

This definition means that a normal chain must be a regular chain. To compute the regular zeros of a normal chain is much easier than to compute the regular zeros of a regular chain. A normal chain is also called a p-chain in Gao and Chou [3]. Some properties about normal chains have been discussed in Wang [10].

Definition 7. *Let $\mathcal{A} = A_1, \cdots, A_s$ be a chain, the saturation ideal of $\mathcal{A}$, denoted by $(\mathcal{A}) : I_{\mathcal{A}}^{\infty}$, is defined as follows*

$$(\mathcal{A}) : I_{\mathcal{A}}^{\infty} = \{P | I_{\mathcal{A}}^{k} P \in (\mathcal{A}) \ \textit{for some integer } k \geq 0 \ \} \tag{2}$$

Lemma 1. *Let* $\mathcal{A} = A_1, \cdots, A_s$ *be a regular chain, let* P *be a polynomial in* K[U, Y], $P \in (\mathcal{A}) : I_{\mathcal{A}}^{\infty}$ *if and only if there exist a polynomial* L *in* K[U]\{0} *such that* $LP \in (\mathcal{A})$.

Please refer [2,5] for the proof.

3 An Algorithm to Compute the Inverse of a Polynomial Modulo an Ideal

Let P, Q be polynomials in K[x]. If P and Q have no common divisors, there exist polynomial P' and Q' such that $\deg(P', x) < \deg(Q, x)$, $\deg(Q', x) < \deg(P, x)$ and $PP' + QQ' = 1$. The extended Euclidean algorithm can compute out P' and Q'. Let $d = \deg(Q, x)$, suppose $P' = a_{d-1}x^{d-1} + \cdots + a_0$, from $\mathrm{rem}(PP' - 1, Q, x) = 0$, we can get a system of linear equations on the variables $a_0, \cdots, a_{d-1}$. It is easy to solve all the a_i for $i = 0, \cdots, d-1$.

Let's see a simple example:

$P = 4x^3 + 8x^2 + 7x + 3$, $Q = 5x^4 + 4x^3 + 3x^2 + 6$.
Let $P' = a_3x^3 + a_2x^2 + a_1x + a_0$,
$\mathrm{rem}(PP' - 1, Q, x) = (-\frac{61}{125}a_3 + \frac{19}{25}a_2 + \frac{24}{5}a_1 + 4a_0)x^3 + (-\frac{657}{125}a_3 + \frac{3}{25}a_2 + \frac{23}{5}a_1 + 8a_0)x^2 + (-\frac{144}{25}a_3 - \frac{24}{5}a_2 + 3a_1 + 7a_0)x + (-\frac{114}{125}a_3 - \frac{144}{25}a_2 - \frac{24}{5}a_1 + 3a_0 - 1)$.
If $\mathrm{rem}(PP' - 1, Q, x) = 0$, we have

$$\begin{array}{l} -\frac{61}{125}a_3 + \frac{19}{25}a_2 + \frac{24}{5}a_1 + 4a_0 = 0 \\ -\frac{657}{125}a_3 + \frac{3}{25}a_2 + \frac{23}{5}a_1 + 8a_0 = 0 \\ -\frac{144}{25}a_3 - \frac{24}{5}a_2 + 3a_1 + 7a_0 = 0 \\ -\frac{114}{125}a_3 - \frac{144}{25}a_2 - \frac{24}{5}a_1 + 3a_0 - 1 = 0 \end{array} \tag{3}$$

The solution is $a_0 = \frac{2194}{13255}, a_1 = -\frac{1614}{13255}, a_2 = -\frac{709}{79530}, a_3 = \frac{2309}{15906}$. then $P' = \frac{2309}{15906}x^3 - \frac{709}{79530}x^2 - \frac{1614}{13255}x + \frac{2194}{13255}$. P' is the inverse of P modulo the polynomial Q.

For polynomials P and Q, the following algorithm will compute the inverse of P modulo Q.

Algorithm 1. InverseModUniVarPol

Input : P,Q are two polynomials in K[x] which have no common divisors.
Output: P' such that $PP' = 1 \bmod (\mathrm{Q})$.
$d \leftarrow \deg(Q, x)$
$P' \leftarrow a_{d-1}x^{d-1} + \cdots + a_0$
$r \leftarrow \mathrm{rem}(PP', Q, x)$
$H \leftarrow coeffs(r - 1, x)$; The set of the coefficients r w.r.t x.
$S \leftarrow solution$ of $H = 0$ for a_i for $i = 0, \cdots, d-1$
$P' \leftarrow subs(S, P')$; substitute the solutions for the a_i's in P'
return P'

Suppose $Q = x^d + a_{d-1}x^{d-1} + \cdots + a_0$ is a monic polynomial , the companion matrix of Q is the $n \times n$ square matrix

$$G = \begin{pmatrix} 0\,0\cdots & 0 & -a_0 \\ 1\,0\cdots & 0 & -a_1 \\ 0\,1\cdots & 0 & -a_2 \\ \vdots\,\vdots\,\ddots & \ddots & \vdots \\ 0\,0\cdots & 1 & -a_{d-1} \end{pmatrix}$$

To compute the inverse of a polynomial modulo a monic polynomial, we have the following theorem.

Theorem 2. *Suppose $P, Q \in K[x]$, Q is monic, and P, Q have no common divisors. Let G be the companion matrix of Q, $P' = b_{d-1}x^{d-1} + \cdots + b_0$, then P' is the inverse of P modulo Q, i.e. $PP' = 1$ mod Q if and only if $(b_0, \cdots, b_{d-1})^T$ is the solution of $P(G)y = (1, 0, \cdots, 0)^T$.*

Proof. Let I be the ideal generated by Q in $K[x]$. $K[x]/I$ can be thought as a finite dimensional linear vector space over K. $\{1, x, \cdots, x^{d-1}\}$ is a monomial basis of $K[x]/I$. M_x is a linear map from $K[x]/I$ to itself, M_x is defined by $M_x(F) = xF \mod \mathrm{I}$ for any F in $K[x]/I$. It is easy to check that G is the matrix representation of M_x on the monomial basis $\{1, x, \cdots, x_{d-1}\}$. Let $M_P(F) = PF \mod \mathrm{I}$, M_P is a linear map defined by P on $K[x]/I$ and $P(G)$ is the matrix representation of M_P, hence $M_P(P') = PP' = 1 \mod \mathrm{I}$ if and only if $(b_0, \cdots, b_{d-1})^T$ is the solution of $P(G)y = (1, 0, \cdots, 0)^T$.

Another proof of this theorem also can be found in [7].

If P, Q are polynomial in $\mathrm{K}[\mathrm{U}, y_1, \cdots, y_i]$, we can extend the above algorithm to find polynomial P' in $\mathrm{K}[\mathrm{U}, y_1, \cdots, y_i]$ and M in $\mathrm{K}[\mathrm{U}, y_1, \cdots, y_{i-1}]$ such that $\mathrm{prem}(PP' - M, Q, y_i) = 0$.

Algorithm 2. InverseModPol

Input : P,Q are two polynomials in $\mathrm{K}[\mathrm{U}, y_1, \cdots, y_i]$ which have no common divisors.
Output: $P' \in \mathrm{K}[\mathrm{U}, y_1, \cdots, y_i]$ and $M \in \mathrm{K}[\mathrm{U}, y_1, \cdots, y_{i-1}]$ such that $\mathrm{prem}(PP' - M, Q, y_i) = 0$.
$d \leftarrow \deg(Q, y_i)$
$P' \leftarrow a_{d-1}y_i^{d-1} + \cdots + a_0$
$r \leftarrow \mathrm{prem}(PP' - M, Q, y_i)$
$H \leftarrow coeffs(r, y_i)$; The set of the coefficients r w.r.t x.
$S \leftarrow solution$ of $H = 0$ for a_i for $i = 0, \cdots, d-1$ and M is considered as a parameter
$P' \leftarrow subs(S, P')$; substitute the solutions for the a_i's in P'
$M \leftarrow denom(P')$; M is the denominator of P'.
return (P', M)

For P, Q are polynomial in $\mathrm{K[U}, y_1, \cdots, y_i]$, we know that there exists P' and Q' in $\mathrm{K[U}, y_1, \cdots, y_i]$ and M in $\mathrm{K[U}, y_1, \cdots, y_{i-1}]$ such that $P'P + Q'Q = M$.

We have implemented the above algorithm. The following is a table which records the timings to compute the inverse of a polynomial modulo another polynomial.

Timings of Computing the Inverse

Number of Variables	Total Degree	Timings	
		InverseModPol	InverseModPol-SubRes
3	3	0.046	0.040
3	4	0.198	0.311
3	5	0.880	1.608
3	6	4.492	9.917
3	7	23.343	64.644
3	8	119.563	407.158
3	9	587.672	2138.584
4	3	0.145	0.067
4	4	1.728	1.564
4	5	43.377	55.396
4	6	1109.934	1426.564
4	7	19673.498	39052.547

If we apply the above algorithm successively, then we can compute the the inverse of a polynomial modulo the saturation ideal of a regular chain. Let P be a polynomial, $\mathcal{A} = A_1, \cdots, A_s \subset \mathrm{K[U, Y]}$ be a regular chain, P is invertible w.r.t. $\mathcal{A}$, then there exist polynomial $P' \in \mathrm{K[U, Y]}$ and $M \in \mathrm{K[U]}$ such that $PP' - M = 0 \bmod ((\mathcal{A}) : \mathrm{I}_{\mathcal{A}}^{\infty})$.

The following algorithm will compute the inverse of a polynomial modulo the saturation ideal of a regular chain.

Algorithm 3. InverseModSat

Input : P is a polynomial in $\mathrm{K[U, Y]}$, $\mathcal{A} = A_1, \cdots, A_s$ is a regular chain in $\mathrm{K[U, Y]}$, y_i is the leading variable of A_i for $i = 1, \cdots, s$, P is invertible w.r.t. $\mathcal{A}$.

Output: $P' \in \mathrm{K[U, Y]}$, $M \in \mathrm{K[U]}$ such that $PP' = M \bmod ((\mathcal{A}) : \mathrm{I}_{\mathcal{A}}^{\infty})$.

$Q \leftarrow 1$
for *i from s to 1 step -1* **do**
  $(P', M) \leftarrow \mathrm{InverseModPol}(P, Q, y_i)$
  $P \leftarrow M$
  $Q \leftarrow QP'$
end
$P' \leftarrow Q$
$M \leftarrow P$
return (P', M)

4 Transforming Regular Chain into Normal Chain

Let $\mathcal{A} = A_1, \cdots, A_s$ be a regular chain and $\mathcal{A}_i = A_1, \cdots, A_i$ for $i = 1, \cdots, s$, let I_i be the initial of A_i, i.e. $I_i = \mathrm{I}(A_i)$, $I_1 \in \mathrm{K[U]}$, for $i = 2, \cdots, s$, I_i is invertible w.r.t. $\mathcal{A}$, then there exist I_i' in $\mathrm{K}[\mathrm{U}, y_1, \cdots, y_{i-1}]$, M_i in K[U] such that $\mathrm{prem}(I_i I_i' - M_i; \mathcal{A}_{i-1}) = 0$. i.e. $I_i I_i' = M_i + N_i$ and $N_i \in (\mathcal{A}_{i-1}) : I_{\mathcal{A}_{i-1}}^{\infty}$. We let $B_1 = A_1$, $H_1 = 1$, and for $i = 2, \cdots, s$, $B_i = M_i y_i^{n_i} + I_i' R_i$ and $H_i = M_2 \cdots M_i$, with $A_i = I_i y_i^{n_i} + R_i$. Let $\mathcal{B}_i = B_1, \cdots, B_i$.

Theorem 3. *For $\mathcal{A}$ and $\mathcal{B}$ as above, $\mathcal{A}$ is a regular chain and $\mathcal{B}$ is a normal chain , we have $(\mathcal{A}) : I_{\mathcal{A}}^{\infty} = (\mathcal{B}) : I_{\mathcal{B}}^{\infty}$.*

Proof. We will prove $(\mathcal{A}_i) : I_{\mathcal{A}_i}^{\infty} = (\mathcal{B}_i) : I_{\mathcal{B}_i}^{\infty}$ for $i = 1, \cdots, s$.

($\subset$) We will prove $(\mathcal{A}_i) : I_{\mathcal{A}_i}^{\infty} \subset (\mathcal{B}_i) : I_{\mathcal{B}_i}^{\infty}$ by induction on i. It is true for $i = 1$. Suppose it is also true for $i - 1$, we will prove it for i.

For any $P \in (\mathcal{A}_i) : I_{\mathcal{A}_i}^{\infty}$ then there exist a nonzero polynomial $L_i \in \mathrm{K[U]}$ such that $L_i P \in (\mathcal{A}_i)$ by lemma 1, i.e. there exist polynomial Q_i such that $L_i P = Q_i A_i \bmod (\mathcal{A}_{\mathrm{i}-1})$.

$$\begin{aligned} H_i L_i P &= H_{i-1} M_i Q_i A_i \bmod (\mathcal{A}_{\mathrm{i}-1}) \\ &= H_{i-1} Q_i (I_i I_i' - N_i) A_i \bmod (\mathcal{A}_{\mathrm{i}-1}) \\ &= H_{i-1} Q_i (I_i B_i + N_i y_i^{n_i} - N_i A_i) \bmod (\mathcal{A}_{\mathrm{i}-1}) \end{aligned}$$

By induction, we know that $H_i L_i P$ is in $(\mathcal{B}_i) : I_{\mathcal{B}_i}^{\infty}$. Since $H_i, L_i \in \mathrm{K[U]}$, we know that P is in $(\mathcal{B}_i) : I_{\mathcal{B}_i}^{\infty}$ by lemma 1.

($\supset$) For any $P \in (\mathcal{B}_i) : I_{\mathcal{B}_i}^{\infty}$, then there exist a nonzero polynomial $L_i \in \mathrm{K[U]}$ such that $L_i P$ is in $(\mathcal{B}_i)$ by lemma 1. From $I_i' A_i = B_i + N_i y_i^{n_i}$, we know that $L_i P \in (\mathcal{A}_i) : I_{\mathcal{A}_i}^{\infty}$. Since L_i is in K[U] and $\mathcal{A}_i$ is a regular chain, then P is in $(\mathcal{A}_i) : I_{\mathcal{A}_i}^{\infty}$.

Let $B_1' = B_1$, $\mathcal{B}_1' = B_1$, for $i = 2, \cdots, s$, let B_i' be the remainder of B_i w.r.t $\mathcal{B}_{i-1}'$ and $\mathcal{B}_i' = B_1', \cdots, B_i'$. Let $\mathcal{B}' = \mathcal{B}_s'$. It is easy to see that $\mathcal{B}'$ is a normal chain and $(\mathcal{B}) : I_{\mathcal{B}}^{\infty} = (\mathcal{B}') : I_{\mathcal{B}'}^{\infty}$. $\mathcal{B}'$ is called the normalization of $\mathcal{A}$. It is easy to check that

$$\mathrm{Zero}(\mathcal{B}'/I_{\mathcal{B}'}) \subset \mathrm{Zero}(\mathcal{A}/I_{\mathcal{A}}) \subset \mathrm{Zero}(\mathcal{A}) \subset \mathrm{Zero}(\mathcal{B}')$$

According to the above theorem, we have the following algorithm to transform a regular chain into a normal chain.

From the above theorem, we have

Corollary 1. *For a polynomial set $\mathbb{F}$, there is an algorithm to compute a series of normal chains $\mathcal{B}_i$ such that*

$$\mathrm{Zero}(\mathbb{F}) = \bigcup_i \mathrm{Zero}((\mathcal{B}_i) : I_{\mathcal{B}_i}^{\infty}) \tag{4}$$

Proof. For a polynomial set $\mathbb{F}$, there are algorithms to compute a series of regular chains $\mathcal{A}_i$ such that

$$\mathrm{Zero}(\mathbb{F}) = \bigcup_i \mathrm{Zero}((\mathcal{A}_i) : I_{\mathcal{A}_i}^{\infty})$$

Algorithm 4. Reg2Norm

Input : $\mathcal{A} = A_1, \cdots, A_s$ is a regular chain in K[U, Y], y_i is the leading variable of A_i for $i = 1, \cdots, s$
Output: $\mathcal{B} = B_1, \cdots, B_s$ is a normal chain in K[U, Y] such that $(\mathcal{A}) : I_{\mathcal{A}}^{\infty} = (\mathcal{B}) : I_{\mathcal{B}}^{\infty}$

$Q \leftarrow 1$
if *s=1* **then return** $\mathcal{A}$
$B_1 \leftarrow A_1$
for $i \leftarrow 2$ **to** s **do**
  $I_i \leftarrow \mathrm{I}(A_i)$
  $(I_i', M_i) \leftarrow \mathrm{InverseModSat}(I_i, \mathcal{A}_{i-1})$
  $n_i \leftarrow \deg(A_i, y_i)$
  $R_i \leftarrow A_i - I_i y_i^{n_i}$
  $B_i \leftarrow M_i y_i^{n_i} + I_i' R_i$
end
$\mathcal{B} \leftarrow B_1$
for $i \leftarrow 2$ **to** s **do**
  $\mathcal{B} \leftarrow \mathcal{B}, Reduce(B_i, \mathcal{B})$
end
return $\mathcal{B}$

where $\mathcal{A}_i$'s are regular chains. For each $\mathcal{A}_i$, the above algorithm will compute a normal chain $\mathcal{B}_i$ such that $(\mathcal{A}_i) : I_{\mathcal{A}_i}^{\infty} = (\mathcal{B}_i) : I_{\mathcal{B}_i}^{\infty}$, and so

$$\mathrm{Zero}(\mathbb{F}) = \bigcup_i \mathrm{Zero}((\mathcal{A}_i) : I_{\mathcal{A}_i}^{\infty}) = \bigcup_i \mathrm{Zero}((\mathcal{B}_i) : I_{\mathcal{B}_i}^{\infty})$$

The corollary is proved.

If the polynomial system $\mathbb{F}$ is zero dimensional, then we have

$$\mathrm{Zero}(\mathbb{F}) = \bigcup_i \mathrm{Zero}(\mathcal{B}_i)$$

Corollary 2. *For a polynomial set $\mathbb{F}$, there is an algorithm to compute a series of normal chains $\mathcal{A}_i$ such that*

$$\mathrm{Zero}(\mathbb{F}) = \bigcup_i \mathrm{Zero}(\mathcal{A}_i / I_{\mathcal{A}_i})) \tag{5}$$

5 Examples

Example 1. Solving the following Lorentz problem:

$$\begin{aligned} f_1 &= x_2(x_3 - x_4) - x_1 + c = 0 \\ f_2 &= x_3(x_4 - x_1) - x_2 + c = 0 \\ f_3 &= x_4(x_1 - x_2) - x_3 + c = 0 \\ f_4 &= x_1(x_2 - x_3) - x_4 + c = 0 \end{aligned}$$

where x_1, x_2, x_3 and x_4 are variables and c is a parameter.

This problem has been discussed in [3]. In order to solve this system of equations of parametric polynomials. We will decompose this polynomial system into normal chains.

Let $F = \{f_1, f_2, f_3, f_4\}$, for a variable order $x_4 > x_3 > x_2 > x_1 > c$, we have the zero decomposition

$$\text{Zero}(F) = \bigcup_{i=1}^{9} \text{Zero}(\mathcal{A}_i / I_{\mathcal{A}_i})$$

where $\mathcal{A}_i$'s are regular chains.

We can transform the regular chains $\mathcal{A}_i$ into normal chains $\mathcal{B}_i$ by using algorithm *Reg2Norm* such that $\bigcup_{i=1}^{9} \text{Zero}(\mathcal{B}_i / I_{\mathcal{B}_i}) \subset \text{Zero}(F)$. For this example, we have

$$\text{Zero}(F) = \bigcup_{i=1}^{9} \text{Zero}(\mathcal{B}_i / I_{\mathcal{B}_i})$$

where $\mathcal{B}_i$ are normal chains. By our new algorithm, it takes 63 seconds to get the normal chains while the old algorithm will cost 106 seconds.

The following is a table which records the length of the chain and the number of the terms of the polynomials in the normal chains which are in the decomposition of the Lorentz polynomial system.

The normal chains in the decomposition

normal chains	length of the chain	number of terms				
1	4	2	2	2	2	
2	4	1291	1289	410	13	
3	5	1	2	1	2	2
4	5	2	2	1	2	3
5	5	2	2	2	2	2
6	5	3	6	7	5	3
7	5	1	2	2	2	3
8	5	9	9	9	5	5
9	5	15	15	15	8	8

6 Conclusions

We give a new algorithm to compute the normal chain from a regular chain such that their saturation ideals are the same. Our algorithm is based on solving system of linear equations and it is much more efficient than the original algorithm to compute the normalization of a regular chain.

References

1. Aubry, P., Lazard, D., Maza, M.M.: On the Theories of Triangular Sets. J. Symbolic Computation 28, 105–124 (1999)
2. Bouziane, D., Kandri Rody, A.K., Maarouf, H.: Unmixed-dimensional Decomposition of a Finitely Generated Perfect Differential Ideal. J. Symbolic Computation. 31, 631–649 (2001)

3. Gao, X.S., Chou, S.C.: Solving parametric algebraic systems. In: Proceedings ISSAC 1992, Berkeley, July 27-29, pp. 335–341. Association for Computing Machinery, New York (1992)
4. Kalbrener, M.: A Generalized Euclidean Algorithm for Computing Triangular Representations of Algebraic Varieties. J. Symbolic Computation 15, 143–167 (1993)
5. Lazard, D.: A new method for solving algebraic systems of positive demension. Discrete Appl. Math. 33, 147–160 (1991)
6. Moreno, M.M.: On triangular decompositions of algebraic varieties. In: MEGA 2000, Bath, England (presented, 2000)
7. Pan, V.Y.: Sturctured Matrices and Polynomials. Birkhäuser, Boston (2001)
8. Wang, D.K., Zhang, Y.: An algorithm for decomposing a polynomial system into normal ascending sets. Science in China, Series A: Mathematics 50(10), 1441–1450 (2007)
9. Wang, D.M.: Some Notes on Algebraic Method for Geometric Theorem Proving
10. Wang, D.M.: Elimination Method. Springer, New York (2001)
11. Wu, W.T.: Basic principles of mechanical theorem proving in elementray geometries. J. Syst. Sci. Math. Sci. 4, 20–235 (1984)
12. Yang, L., Zhang, J.Z.: Search dependency between algebraic equations: An algorithm applied to automated reasoning. Technical Report ICTP/91/6, International Center For Theoretical Physics, Trieste (1991)

A Modified Van der Waerden Algorithm to Decompose Algebraic Varieties and Zero-Dimensional Radical Ideals*

Jia Li and Xiao-Shan Gao

Key Laboratory of Mathematics Mechanization
Institute of Systems Science, AMSS,
Chinese Academy of Sciences,
Beijing 100080, China
{lijia,xgao}@mmrc.iss.ac.cn

Abstract. In this paper, we introduce a modified Van der Waerden algorithm to decompose a variety into the union of irreducible varieties. We give an effective representation for irreducible varieties obtained by the algorithm, which allows us to obtain an irredundant decomposition easily. We show that in the zero dimensional case, the polynomial systems for the irreducible varieties obtained in the Van der Waerden algorithm generate prime ideals. As a consequence, we have an algorithm to decompose the radical ideal generated by a finite set of polynomials as the intersection of prime ideals and the degree of the polynomials in the computation is bounded by $O(d^n)$ where d is the degree of the input polynomials and n is the number of variables.

Keywords: Algebraic variety, zero dimensional variety, resultant, irredundant decomposition.

1 Introduction

A fundamental construction in commutative algebra is to decompose a radical ideal into the intersection of prime ideals. From a geometric viewpoint, this is equivalent to decomposing an algebraic variety into irreducible varieties. The geometric version is slightly weaker than the algebraic version, since an irreducible variety is not necessarily represented by a prime ideal.

In recent years, there appeared a lot of work on this kind of decomposition algorithms. Algorithms for computing the prime decomposition of radical ideals have been proposed in [9,10,12,13]. All of these algorithms are based on the Gröbner basis and have no complexity analysis. Corresponding to the prime decomposition of radical ideals, algorithms for irreducible decomposition of varieties are developed in [2,3,5,6,17,18,19]. The method in [6] uses Bezontian matrices. It only gives the generic point of each irreducible variety, but

* Partially supported by a National Key Basic Research Project of China under Grant No. 2004CB318000.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 246–262, 2008.
© Springer-Verlag Berlin Heidelberg 2008

can not give the generating polynomial systems of these varieties. This method also needs combinatorial selection of parameters. The algorithms developed in [2,3] use multivariate resultants. They introduce ϵ as a perturbed parameter to obtain a regular system F_ϵ and recover the information about the isolated roots from the trailing coefficient in ϵ of the determinant of the Macaulay matrix associated to the system F_ϵ. Another approach is the characteristic set method [4,5,7,13,14,17,19], which can decompose a variety into the union of irreducible varieties represented by irreducible ascending chains. Generating polynomial systems for the irreducible varieties can also be found via Chow form or Gröbner bases[4,7,17,18]. This approach has a very high worst case complexity. A characteristic set method with single exponential complexity is given in [14]. This method only decomposes the variety as unmixed ones represented by their characteristic sets.

In this paper, we give an algorithm to compute the irredundant irreducible variety decomposition of a given variety defined by the following polynomial equations

$$f_1(x_1, \ldots, x_n) = 0, \ldots, f_m(x_1, \ldots, x_n) = 0. \tag{1}$$

We give a complexity analysis of this algorithm. We prove that our algorithm also gives an irredundant prime ideal decomposition of the radical ideal generated by $f_1, \ldots, f_n$ when this ideal is zero-dimensional.

We use an algorithm developed by B. L. Van der Waerden in [15] (abbr, VDW algorithm) to decompose the variety generated by (1) into irreducible varieties. This algorithm computes $2n$ resultants at most, and the degree of the polynomials which occur in the computation in any variable is bounded by d^{4^n} where $d = \max_{i\in\{1,\ldots,n\}}\{\deg_{x_i}(f_1), \ldots, \deg_{x_i}(f_m)\}$. As a consequence, we obtain an irreducible decomposition for a variety by computing at most $2n$ resultants of polynomials with degrees bounded by d^{4^n}.

We present two major improvements of this algorithm. First, we give an effective representation for the irreducible varieties obtained by the algorithm. Using this representation, we can easily decide the inclusion relationship of two varieties. As a consequence, we may give an irredundant decomposition. Note that the generating polynomial systems given in the VDW algorithm are not prime ideals in the general case and deciding the inclusion relationship of two varieties need compute Gröbner bases without our method. Second, we show that in the zero-dimensional case, the VDW algorithm gives prime ideals. Combining the ideas from [2] and the result in this paper, we give an algorithm to decompose the zero-dimensional radical ideal generated by (1) as intersection of prime ideals and the degree of the polynomials in the computation is bounded by $O(d^n)$.

Comparing to the Gröbner basis method, the computation step of the VDW algorithm is very "large" in the sense that each step eliminates one variable in all the polynomials. In the Gröbner basis computation, each step is very "small", which only eliminates one monomial. The characteristic set method is in between: each step decreases the degree of a polynomial. Also note that to obtain a prime decomposition requires to compute the Gröbner bases for many times. The VDW algorithm is quite similar to the one in [2]. The difference

is that the VDW algorithm uses a top-to-down approach in the sense that it computes the components with higher dimensions first, while the algorithm in [2] computes components with lower dimension first.

2 Preliminaries

Notations and results needed in this paper are summarized in this section.

In what follows the reader is assumed to be familiar with the basic notions about characteristic sets for which we refer to [8,19].

Let K be a computable field of characteristic zero, e.g., $\mathbf{Q}$. We use $K[x_1, \ldots, x_n]$ or $K[x]$ to denote the ring of polynomials in the indeterminates $x_1, \ldots, x_n$. Unless explicitly mentioned otherwise, all polynomials in this paper are in $K[x]$. Let E be a *universal extension field* of K, i.e., an algebraically closed field extension of K which contains sufficiently many independent indeterminates over K. We will consider zeros of polynomials in the field E.

Let P be a polynomial. The *class* of P, denoted by $\text{class}(P)$, is the largest p such that some x_p actually occurs in P. If $P \in K, \text{class}(P) = 0$. A sequence of polynomials $\mathcal{A} = A_1, \ldots, A_p$ is said to be an *ascending chain*(asc chain), or simply, a *chain*, if either $r = 1$ and $A_1 \neq 0$ or $0 < \text{class}(A_i) < \text{class}(A_j)$ for $1 \leq i < j$ and A_k is of higher degree than A_m for $m > k$ in x_{n_k} where $n_k = \text{class}(A_k)$.

Definition 1. *The dimension of an irreducible chain* $\mathcal{A} = A_1, \ldots, A_p$ *is defined to be* $\dim(\mathcal{A}) = n - p$ *which is the number of parameters of* $\mathcal{A}$.

Definition 2. *A characteristic set* (abbr. char set) *of an ideal* $\mathbb{I}$ *is a chain* $\mathcal{A}$ *in* $\mathbb{I}$ *such that for all* $P \in \mathbb{I}$, $\text{prem}(P, \mathcal{A}) = 0$.

Theorem 1. *[13,19] If* $\mathcal{A}$ *is an irreducible chain then* $\text{sat}(\mathcal{A})$ *is a prime ideal with dimension* $\dim(\mathcal{A})$*, and* $\mathcal{A}$ *is a char set of* $\text{sat}(\mathcal{A})$*. Conversely, each char set of a prime ideal is an irreducible chain.*

Lemma 1. *[19] Let* $\mathcal{A}$ *be an irreducible ascending chain with parameters* $u_1, \ldots, u_q$*. If* Q *is a polynomial not in* $\text{sat}(\mathcal{A})$*, then we can find a* $P \in K[u]$ *such that* $P \in ideal(\mathcal{A} \bigcup \{Q\})$.

An ideal distinct from (1) and (0) is called nontrivial.

Lemma 2. *Let* $\mathcal{A}$ *be an irreducible chain with parameters* $u_1, \ldots, u_q$*, we can find an irreducible chain* $\mathcal{A}'$ *such that* $\text{sat}(\mathcal{A}) = \text{sat}(\mathcal{A}')$ *and the initials of the polynomials in* $\mathcal{A}'$ *are in* $K[u]$.

Definition 3. *Let* $\mathbb{I}$ *be a nontrivial prime ideal in* $K[x]$*. We can divide the* x *into two groups,* $u_1, \ldots, u_q$ *and* $y_1, \ldots, y_p$*,* $p + q = n$*, such that* $\mathbb{I} \cap K[u_1, \ldots, u_q] = \emptyset$*, while, for* $i = 1, \ldots, p$*,* $\mathbb{I}$ *contains a nonzero polynomial in* y_i *and the* u *alone. We call the* u *a parameter set of* $\mathbb{I}$.

Lemma 3. *[8] Use the notations as above. Let* $\mathbb{I}$ *be a nontrivial prime ideal in* $K[x]$*. A char set of* $\mathbb{I}$ *under the variable order* $u_1 < \ldots < u_q < y_1 < \ldots < y_p$ *is of the form*

$$\mathcal{A} = A_1(u, y_1), A_2(u, y_1, y_2), \ldots, A_p(u, y_1, \ldots, y_p) \tag{2}$$

where A_i *is a polynomial involving* y_i *effectively. Conversely, for an irreducible chain like (2), the* u *consists of a parameter set of the prime ideal* $\mathrm{sat}(\mathcal{A})$.

Let $f_1, f_2, \ldots, f_r$ be r polynomials in a single variable x of given degrees with indeterminate coefficients. The *resultant system* of the polynomials $f_1, f_2, \ldots, f_r$ can be computed in the following way[16].

First, we transform the polynomials $f_1, f_2, \ldots, f_r$ into polynomials of the same degree by multiplying every polynomial f_i by x^{n-n_i} and $(x-1)^{n-n_i}$ respectively where n_i is the degree of f_i in x, and n is the greatest of n_i. We designate these new polynomials by $g_1, g_2, \ldots, g_s$.

Next, from $g_1, g_2, \ldots, g_s$, we form the linear combinations

$$g_u = u_1 g_1 + u_2 g_2 + \cdots + u_s g_s; g_v = v_1 g_1 + v_2 g_2 + \cdots + v_s g_s,$$

where u, v are indeterminates which are adjoined to the field K.

Finally, let $R = \mathrm{resl}(g_u, g_v, x)$ be the resultant of g_u and g_v wrt x. If we arrange R according to the power products of the u and v, and denote the coefficients by $D_1, D_2, \ldots, D_h$. Then $D_1, D_2, \ldots, D_h$ are the resultant system of $f_1, \ldots, f_r$.

Remark 1. If it is known beforehand that the leading coefficient of one of the polynomials f_v, say f_1, does not vanish, we may omit the entire preliminary operation whereby the polynomials f_v are transformed into polynomials of the same degree. Moreover the calculations may then be simplified by forming the resultant of f_1 and $v_2 f_2 + v_3 f_3 + \ldots + v_r f_r$ rather than that of g_u and g_v.

3 Outline of the VDW Decomposition Algorithm

In this section, we outline the algorithm developed by van der Waerden. The details of this algorithm can be found in [15]. Suppose that we want to decompose the zero set of the following polynomial equations

$$f_1(x_1, x_2, \ldots, x_n) = 0, f_2(x_1, x_2, \ldots, x_n) = 0, \ldots, f_m(x_1, x_2, \ldots, x_n) = 0 \tag{3}$$

into irreducible varieties. Denote this variety by $M = \mathrm{Zero}(f_1, f_2, \ldots, f_m)$. The polynomial set $\{f_1, f_2, \ldots, f_m\}$ is called the *definition system* of M.

To construct this algorithm, we introduce a new variable z as follows

$$z - u_1 x_1 - \ldots - u_n x_n = 0 \tag{4}$$

where $u_1, \ldots, u_n$ are indeterminates.

Now, let us explain the algorithm described in [15] that computes the zero decomposition of (3) by eliminating the variables successively from x_n to x_1 in n steps. In the first step, we rename x_j and u_j in (3),(4) as $x_j^{(0)}$ and $u_j^{(0)}$ respectively, where i=1,...,n. We give a sketch to describe this algorithm:

VDW Algorithm

$$
\begin{array}{ccccccc}
\mathbb{P}^{(1)} & \longrightarrow & \mathbb{P}^{(2)} & \longrightarrow \ldots \longrightarrow & \mathbb{P}^{(i)} & \longrightarrow \ldots \longrightarrow & \mathbb{P}^{(n)} \\
\downarrow & & \downarrow & & \downarrow & & \downarrow \\
\bar{\mathbb{P}}^{(1)} & & \bar{\mathbb{P}}^{(2)} & & \bar{\mathbb{P}}^{(i)} & & \bar{\mathbb{P}}^{(n)} \\
\downarrow & & \downarrow & & \downarrow & & \downarrow \\
g_j^{(1)} & & g_j^{(2)} & & g_j^{(i)} & & g_j^{(n)} \\
\downarrow & & \downarrow & & \downarrow & & \downarrow \\
h^{(1)},\, l_j^{(1)} & & h^{(2)},\, l_j^{(2)} & & h^{(i)},\, l_j^{(i)} & & h^{(n)},\, l_j^{(n)} \\
\downarrow \;\downarrow & & \downarrow \;\downarrow & & \downarrow \;\downarrow & & \downarrow \\
h_{\mu k}^{(1)}\; e_j^{(1)} & & h_{\mu k}^{(2)}\; e_j^{(2)} & & h_{\mu k}^{(i)}\; e_j^{(i)} & & h_{\mu k}^{(n)}
\end{array}
$$

In the above sketch, $\mathbb{P}^{(1)} = \{(3), (4)\}$, and $\mathbb{P}^{(i)} = \{l_j^{(i-1)}, e_j^{(i-1)}\}$, $i = 2, \ldots, n$.

Now, let us explain the ***i*-th step** of above algorithm. To discuss conveniently, we denote $\mathbb{P}^{(i)} = \{f_1^{(i)}, \ldots, f_s^{(i)}\}$, where $f_k^{(i)} \in K[u^{(i-1)}, x_1^{(i-1)}, \ldots, x_{n-i+1}^{(i-1)}, z]$ and at least one $f_k^{(i)}$ said $f_1^{(i)}$ is in $K[x_1^{(i-1)}, \ldots, x_{n-i+1}^{(i-1)}]$.

STEP i.1: To avoid the resultant system vanishing identically, we do a linear transformation:

$$\mathbf{x}^{(i-1)} = M_{i-1}\mathbf{x}^{(i)}, \quad \mathbf{u}^{(i-1)} = N_{i-1}\mathbf{u}^{(i)}$$

where $\mathbf{x}^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})'$, $\mathbf{u}^{(i)} = (u_1^{(i)}, \ldots, u_n^{(i)})'$, and

$$
M_{i-1} = \begin{pmatrix}
1 & \cdots & 0 & v_1^{(i-1)} & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 1 & v_{n-i}^{(i-1)} & 0 & \cdots & 0 \\
0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\
0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & 0 & 0 & \cdots & 1
\end{pmatrix}, N_{i-1} = \begin{pmatrix}
1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\
-v_1^{(i-1)} & \cdots & -v_{n-i}^{(i-1)} & 1 & 0 & \cdots & 0 \\
0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & 0 & 0 & \cdots & 1
\end{pmatrix}, \tag{5}
$$

and $v_k^{(i-1)}, k = 1, \ldots, n-i$ are constants in K such that the leading coefficient of $f_1^{(i)}(x_1^{(i)} + v_1^{(i-1)} x_{n-i+1}^{(i)}, \ldots, x_{n-i}^{(i)} + v_{n-i}^{(i-1)}, x_{n-i+1}^{(i)})$ in $x_{n-i+1}^{(i)}$ is a nonzero constant in K.

After this transformation, z is still a linear form in $x^{(i)}$ and $u^{(i)}$:

$$z = u_1^{(i)} x_1^{(i)} + \ldots + u_n^{(i)} x_n^{(i)}.$$

Let $\bar{\mathbb{P}}^{(i)}$ to be the set of new polynomials obtained from $\mathbb{P}^{(i)}$ by doing the above transformation.

STEP i.2: Compute $\{g_j^{(i)}\}$, the resultant system of the polynomials in $\bar{\mathbb{P}}^{(i)}$ wrt $x_{n-i+1}^{(i)}$.

STEP i.3: Compute $h^{(i)}$, the greatest common factor of all $g_j^{(i)}$, which is called the *i-th partial resultant* of (3) and $l_j^{(i)} = g_j^{(i)}/h^{(i)}$. So $l_j^{(i)}$ are co-prime.

STEP i.4: To obtain equations only involving $x^{(i)}$, compute the resultant system of $\{l_j^{(i)}\}$ wrt z, and arrange them according to the power products of the $u^{(i)}$, and denote $\{e_j^{(i)}\}$ to be these coefficients. Since $l_j^{(i)}$ are co-prime, $e_j^{(i)}$ do not vanish identically.

STEP i.5: Factorizing $h^{(i)}$ into irreducible factors in $K[u^{(i)}, x_1^{(i)}, \ldots, x_{n-i}^{(i)}, z]$:

$$h^{(i)}(u^{(i)}, x_1^{(i)}, \ldots, x_{n-i}^{(i)}, z) = \Theta(u^{(i)}, x_1^{(i)}, \ldots, x_{n-i}^{(i)}) \prod_{\mu} h_\mu^{(i)}(u^{(i)}, x_1^{(i)}, \ldots, x_{n-i}^{(i)}, z)^{\sigma_\mu}. \quad (6)$$

Replacing z by $u_1^{(i)}x_1^{(i)} + \ldots + u_n^{(i)}x_n^{(i)}$ in $h_\mu^{(i)}$, and arranging it according to the power products of the $u^{(i)}$. Denote $\{h_{\mu k}^{(i)}\}$ to be these coefficients.

Definition 4. *$u_1, \ldots, u_s$ are different indeterminates. η is said to be independent with $u_1, \ldots, u_s$ if $\eta \in K$ or η and $u_1, \ldots, u_s$ are algebraic independent over K. $(\eta_1, \ldots, \eta_t)$ is said to be independent with $u_1, \ldots, u_s$ if η_i is independent with $u_1, \ldots, u_s$ for all i.*

After all the x are eliminated. We obtain a sequence of partial resultants

$$h^{(1)}(u^{(1)}, x_1^{(1)}, \ldots, x_{n-1}^{(1)}, z) = 0, \ldots, h^{(n-1)}(u^{(n-1)}, x_1^{(n-1)}, z) = 0, h^{(n)}(u^{(n)}, z) = 0. \quad (7)$$

Theorem 2. *[15] Assume (η, ζ) to be a solution of (3)$\cup$(4). When the η are independent with the u, (η, ζ) is a solution of one of the $h^{(i)} = 0$ in (7).*

Theorem 3. *[15] Assume $\xi_1, \ldots, \xi_{n-r}$ to be given(constants or variables). Then every solution of the r-th equation of (7) is of the form*

$$\zeta = u_1^{(r)}\xi_1 + \ldots + u_n^{(r)}\xi_n \quad (8)$$

where ξ_k are independent with $u_i^{(r)}$ and $\{\xi_k\}$ is a set of solutions of (3).

Now we consider an irreducible factor $h_\mu^{(i)}$ of $h^{(i)}$. According to Theorem 3, if we set $x_1^{(i)} = \xi_1, \ldots, x_{n-i}^{(i)} = \xi_{n-i}$ where $\xi_1, \ldots, \xi_{n-i}$ are indeterminates independent with the u, then the zeros of $h^{(i)}(u^{(i)}, x_1^{(i)}, \ldots, x_{n-i}^{(i)}, z)$ are of the form referred in (8). So factorizing $h_\mu^{(i)}$ into linear factors in certain extensional field of $K(u^{(i)}, \xi)$, we have

$$h_\mu^{(i)}(u^{(i)}, x_1^{(i)}, \ldots, x_{n-i}^{(i)}, z) = \gamma_\mu \prod_{\nu} (z - u_1^{(i)}\xi_1 - \cdots - u_n^{(i)}\xi_n^{(\nu)}) \quad (9)$$

where different $\xi^{(\nu)} = \{\xi_1, \ldots, \xi_{n-i}, \xi_{n-i+1}^{(\nu)}, \ldots, \xi_{n-1}^{(\nu)}, \xi_n^{(\nu)}\}$ are conjugate to each other. Let $\xi_\mu^{(i)} = \xi^{(1)}$. Then ξ is a $(n-i)$-dimensional point in variety M because $\xi_1, \ldots, \xi_{n-i}$ are indeterminates and $\xi_{n-1+1}, \ldots, \xi_n$ are their algebraic functions.

The set of equations

$$h_{\mu 1}^{(i)}(x^{(i)}) = 0, \ldots, h_{\mu m_\mu}^{(i)}(x^{(i)}) = 0. \quad (10)$$

defines a variety $M_\mu^{(i)}$ with the following property.

Theorem 4. *[15] Let $M_\mu^{(i)}$ be the variety defined by the irreducible factors of the (i)-th partial resultant of M according to (10). Then $M_\mu^{(i)}$ is an irreducible variety of dimension $n - i$ with $\xi_\mu^{(i)}$ as a generic point. Furthermore, we have*

$$M = \bigcup_{i=1,\ldots,n;\mu} M_\mu^i.$$

We give following example to illustrate this algorithm.

Example 1. Let $M = \text{Zero}(f_1, f_2, f_3)$ where $f_1 = x_1x_2x_3 - x_1x_2^2 - x_3 - x_1 - x_2, f_2 = x_1x_3 - x_1^2 - x_2 - x_3 + x_1$, and $f_3 = x_3^2 - x_1^2 - x_2^2$.

Let $\mathbb{P}^{(1)} = \{f_1, f_2, f_3, f_4 = z - u_1x_1 - u_2x_2 - u_3x_3\}$. Since the leading coefficient of f_3 in x_2 is nonzero, we will eliminate x_2 first. The resultant system of $\mathbb{P}^{(1)}$ wrt x_2 is

$$\begin{aligned}
g_1^{(1)} &= -5x_1^3x_3 + 2x_1^4x_3 - 2x_1 + x_1^2 + 3x_1^2x_3 - x_1^5 - x_1^3 + 3x_1^2x_3^2 - x_1^3x_3^2 - x_1x_3 - 2x_1x_3^2 + 2x_1^4,\\
g_2^{(1)} &= x_1x_3u_2 - z + x_1u_2 + u_1x_1 - x_1^2u_2 - x_3u_2 + u_3x_3,\\
g_3^{(1)} &= -4x_1^2x_3 + 2x_1^3x_3 - 2x_1^2 + 2x_1x_3 - x_1^4 + 2x_1x_3^2 + 2x_1^3 - x_1^2x_3^2;
\end{aligned}$$

The GCD of $g_1^{(1)}, g_2^{(1)}, g_3^{(1)}$ is $h^{(1)} = 1$. So $l_1^{(1)} = g_1, l_2^{(1)} = g_2, l_3^{(1)} = g_3$. The resultant system of $l^{(1)}$ w.r.t z is

$$e_1^{(1)} = -4x_1^2x_3 + 2x_1^3x_3 - 2x_1^2 + 2x_1x_3 - x_1^4 + 2x_1x_3^2 + 2x_1^3 - x_1^2x_3^2.$$

We obtain $\mathbb{P}^{(2)} = \{l_1^{(1)}, l_2^{(1)}, l_3^{(1)}, e_1^{(1)}\}$. We compute the resultant system $\mathbb{P}^{(2)}$ wrt x_1 and obtain the second partial resultant:

$$h^{(2)} = x_3u_2 - u_3x_3 + z.$$

Substituting $z = u_1x_1 + u_2x_2 + u_3x_3$ into $h^{(2)}$, we obtain the coefficients of the power products of the u:

$$h_{21}^{(2)} = x_1, h_{12}^{(2)} = x_3 + x_2.$$

Now we obtain an 1-dimensional irreducible variety $M_1^{(2)} = \text{Zero}(h_{11}^{(2)}, h_{12}^{(2)})$.

Continuing our computation, we obtain the third partial resultant

$$h^{(3)} = 68719476736z(-z + u_2 - u_3)^2(u_1 - z - u_3)^2(3u_1 + 4u_2 - z + 5u_3)$$

which has four irreducible factors:

$$h_1^{(3)} = z, h_2^{(3)} = -z + u_2 - u_3, h_3^{(3)} = u_1 - z - u_3, h_4^{(3)} = 3u_1 + 4u_2 - z + 5u_3.$$

Substituting $z = u_1x_1 + u_2x_2 + u_3x_3$ into the above polynomials, we obtain the coefficients of the power products of the u:

$$\begin{array}{lll}
h_{11}^{(3)} = x_1, & h_{12}^{(3)} = x_2, & h_{13}^{(3)} = x_3;\\
h_{21}^{(3)} = -x_1, & h_{22}^{(3)} = -x_2 + 1, & h_{23}^{(3)} = -x_3 - 1;\\
h_{31}^{(3)} = 1 - x_1, & h_{32}^{(3)} = -x_2, & h_{33}^{(3)} = -x_3 - 1;\\
h_{41}^{(3)} = 3 - x_1, & h_{42}^{(3)} = 4 - x_2, & h_{43}^{(3)} = -x_3 + 5.
\end{array}$$

We obtain four zero-dimensional irreducible varieties: $M_1^{(3)}$=Zero$(h_{11}^{(3)}, h_{12}^{(3)}, h_{13}^{(3)})$, $M_2^{(3)}$=Zero$(h_{21}^{(3)}, h_{22}^{(3)}, h_{23}^{(3)})$, $M_3^{(3)}$=Zero$(h_{31}^{(3)}, h_{32}^{(3)}, h_{33}^{(3)})$, and $M_4^{(3)} = \text{Zero}(h_{41}^{(3)}, h_{42}^{(3)}, h_{43}^{(3)})$. The final decomposition is:

$$M = M_1^{(2)} \bigcup M_1^{(3)} \bigcup M_2^{(3)} \bigcup M_3^{(3)} \bigcup M_4^{(3)}.$$

In the following result, we give an estimation for the degrees of the polynomials in the computation procedure.

Theorem 5. *Let (3) define a variety M, and $d = \max_{i\in\{1,\ldots,n\}}\{\deg_{x_i}(f_1),\ldots,\deg_{x_i}(f_m)\}$. Then, in the VDW Algorithm, the maximal degree in any x_i of any polynomial occurring in the computation is bounded by d^{4^n}.*

Proof. The only computation in the algorithm increasing the degree is the computation of the resultant. After computing a resultant wrt x_n, the maximal degree of $g_j^{(1)}$ in any variable is bounded by d^2. So the degree bound of h and l_j is also d^2. Similarly, the maximal degree of $e_i^{(1)}$ is d^4. Then the degree bound for $\mathbb{P}^{(1)}$ is d^4. Repeat the operation, the degree bound for $\mathbb{P}^{(2)}$ is $(d^4)^2 = d^8$. Finally, the degree bound of $\mathbb{P}^{(n-1)}$ is d^{4^n}. ■

As a consequence, we may obtain an irreducible decomposition for a variety by computing $2n$ resultants of polynomials with degrees bounded by d^{4^n}.

In the following sections, we will discuss the applications and improvements of this algorithm.

4 Irredundant Decomposition

In Sect. 3, we know that the VDW algorithm can be used to decompose a variety into the union of irreducible varieties. In general, this decomposition is redundant. In this section, we will show how to obtain an irredundant decomposition. In order to do that, we need to remove those varieties $M_\mu^{(i)}$ which are contained in varieties $M_\eta^{(k)}$ with higher dimensions. As suggested in [15], since we know the basis of $M_\mu^{(i)}$, the irredundant decomposition can be reached in principle with the methods such as the characteristic set method [19] or the Groebner basis method [1]. But to use these general methods needs extra work. We will give a direct method to find an irredundant decomposition.

Let $h_\mu^{(i)}(u^{(i)}, x_1^{(i)}, \ldots, x_{n-i}^{(i)}, z) \in K[u^{(i)}, x_1^{(i)}, \ldots, x_{n-i}^{(i)}, z]$ be an irreducible factor of the i-th partial resultant from (6). Introduce the following notations.

$$\begin{aligned}
\mathbb{H}_\mu^{(i)} &= \{h_{\mu 1}^{(i)}, \ldots, h_{\mu m_\mu}^{(i)}\} \\
\mathbb{D}_\mu^{(i)} &= (h_{\mu 1}^{(i)}, \ldots, h_{\mu m_\mu}^{(i)}) \\
M_\mu^{(i)} &= V(h_{\mu 1}^{(i)}, \ldots, h_{\mu m_\mu}^{(i)}) \\
\mathbb{K}_\mu^{(i)} &= (h_{\mu 1}^{(i)}, \ldots, h_{\mu m_\mu}^{(i)}, z - (u_1^{(i)} x_1^{(i)} + \ldots + u_n^{(i)} x_n^{(i)})) \\
\mathbb{J}_\mu^{(i)} &= \sqrt{\mathbb{D}_\mu^{(i)}} \\
\mathbb{I}_\mu^{(i)} &= (\mathbb{J}_\mu^{(i)}, z - (u_1^{(i)} x_1^{(i)} + \ldots + u_n^{(i)} x_n^{(i)}))
\end{aligned} \tag{11}$$

where the $h_{\mu j}^{(i)}$ are from (10). By Theorem 4, we know that $\mathbb{J}_\mu^{(i)}$ and hence $\mathbb{I}_\mu^{(i)}$ are prime ideals.

In the following discussion, to simplify the expressions, we still use x, u to denote $x^{(i)}, u^{(i)}$ when we do not need to distinguish $x^{(i)}, u^{(i)}$ and $x^{(j)}, u^{(j)}$.

Lemma 4. *Use the notations in (11). Under the variable order* $u < x_1 < \ldots < x_{n-i} < z < x_{n-i+1} < \ldots < x_n$*, the ideal* $\mathbb{I}_\mu^{(i)}$ *has a char set of the form*

$$\begin{aligned} R_{\mu 0}^{(i)} &= h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, z) \\ R_{\mu 1}^{(i)} &= I_{\mu 1}^{(i)}(u, x_1, \ldots, x_{n-i}, z)x_{n-i+1} + U_{\mu 1}^{(i)}(u, x_1, \ldots, x_{n-i}, z) \\ &\cdots \\ R_{\mu i}^{(i)} &= I_{\mu i}^{(i)}(u, x_1, \ldots, x_{n-i}, z)x_n + U_{\mu i}^{(i)}(u, x_1, \ldots, x_{n-i}, z) \end{aligned} \tag{12}$$

Proof. We know that $\mathbb{J}_\mu^{(i)}$ is a prime ideal whose parameter set is $u, x_1, \ldots, x_{n-i}$. $z - (u_1x_1 + \ldots + u_nx_n)$ is a polynomial linear in z, so $\mathbb{I}_\mu^{(i)}$ is also a prime ideal whose parameter set is $u, x_1, \ldots, x_{n-i}$. According to Lemma 3, $\mathbb{I}_\mu^{(i)}$ has a char set $\mathcal{A}$ of the form

$$\begin{aligned} &R_{\mu 0}^{(i)}(u, x_1, \ldots, x_{n-i}, z), \\ &R_{\mu 1}^{(i)}(u, x_1, \ldots, x_{n-i}, z, x_{n-i+1}), \\ &\ldots, \\ &R_{\mu i}^{(i)}(u, x_1, \ldots, x_{n-i}, z, x_{n-i+1}, \ldots, x_n). \end{aligned}$$

where $R_{\mu 0}^{(i)}$ contains z and $R_{\mu j}^{(i)}$ contains x_{n-i+j} effectively. Furthermore, $\mathbb{I}_\mu^{(i)} = \mathrm{sat}(\mathcal{A})$. Note that $R_{\mu 0}^{(i)}$ is an irreducible polynomial in $u, x_1, \ldots, x_{n-i}, z$ and $h_\mu^{(i)} \in \mathbb{I}_\mu^{(i)}$, so we can assume $R_{\mu 0}^{(i)} = h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, z)$.

Let $\alpha_1, \ldots, \alpha_n, \tau_1, \ldots, \tau_{n-i}, \beta, \tau_{n-i+1}, \ldots, \tau_n$ be a generic zero of $\mathbb{I}_\mu^{(i)}$. So $h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, z)$ vanishes at this generic zero, and hence $\beta = \alpha_1\tau_1 + \ldots + \alpha_n\tau_n$. Furthermore,

$$\begin{aligned} &\tfrac{dh_\mu^{(i)}}{d\alpha_j}(\alpha, \tau_1, \ldots, \tau_{n-j}, \beta) = \tfrac{\partial h_\mu^{(i)}}{\partial u_j}(u, x_1, \ldots, x_{n-i}, z) \\ &+ x_j \tfrac{\partial h_\mu^{(i)}}{\partial z}(u, x_1, \ldots, x_{n-i}, z)|_{(u,x,z)(\alpha,\tau,\beta)} = 0, j = n-i+1, \ldots, n. \end{aligned}$$

That is $B_{j-n+i} = \frac{\partial h_\mu^{(i)}}{\partial u_j}(u, x_1, \ldots, x_{n-i}, z) + x_j \frac{\partial h_\mu^{(i)}}{\partial z}(u, x_1, \ldots, x_{n-i}, z)$ vanishes at the generic zero of $\mathbb{I}_\mu^{(i)}$, $j = n-i+1, \ldots, n$. So $B_k, k = 1, \ldots, i$ are in $\mathbb{I}_\mu^{(i)}$ and they are linear in x_{n-i+k}. Also note that $\frac{\partial h_\mu^{(i)}}{\partial z}(u, x_1, \ldots, x_{n-i}, z)|_{(u,x,z)=(\alpha,\tau,\beta)} \neq 0$. From the definition of irreducible chain, we can choose $R_{\mu k}^{(i)} = \mathrm{prem}(B_k, h_\mu^{(i)})$ as $R_{\mu k}^{(i)}$. ∎

From the above lemma, we know that $\alpha_1, \ldots, \alpha_n, \tau_1, \ldots, \tau_{n-i}, \beta, \tau_{n-i+1}, \ldots, \tau_n$ given above is a generic zero of ideal $\mathbb{I}_\mu^{(i)}$. So $\tau_1, \ldots, \tau_n$ is a generic zero of prime ideal $\mathbb{J}_\mu^{(i)}$ and a generic point of the corresponding irreducible variety $M_\mu^{(i)}$ of $\mathbb{J}_\mu^{(i)}$.

From Lemma 4, we obtain the following *effective representation* for the prime ideal $\mathbb{J}_\mu^{(i)}$:

$$\mathbb{H}_\mu^{(i)} = \{h_{\mu 1}^{(i)}, \ldots, h_{\mu m_\mu}^{(i)}\} \text{ and } \mathcal{A}_\mu^{(i)} = R_{\mu 0}^{(i)}, R_{\mu 1}^{(i)}, \ldots, R_{\mu i}^{(i)} \tag{13}$$

where $R_{\mu j}^{(i)}$ is from (12). This representation is called effective due to the following reasons.

Lemma 5. *A polynomial $g \in K[x]$ is in $\mathbb{J}_\mu^{(i)}$ if and only if* $\mathrm{prem}(g, \mathcal{A}_\mu^{(i)}) = 0$.

Proof. We need only to prove $\mathbb{J}_\mu^{(i)} = \mathbb{I}_\mu^{(i)} \bigcap K[x]$. Obviously, $\mathbb{J}_\mu^{(i)} \subseteq \mathbb{I}_\mu^{(i)} \bigcap K[x]$. On the other hand, assume $P \in \mathbb{I}_\mu^{(i)} \bigcap K[x]$. So $P(x^{(i)}) \in \sqrt{(\mathbb{D}_\mu^{(i)}, z - \sum_{i=1}^{n} u_i^{(i)} x_i^{(i)})}$. Then there exists an integer $r > 0$ and polynomials $A_j \in K[u^{(i)}, x^{(i)}, z], j = 0, \ldots, n$ such that $P(x^{(i)})^r = \sum_{j=1}^{n} A_j h_{\mu j}^{(i)} + A_0(z - \sum_{i=1}^{n} u_i^{(i)} x_i^{(i)})$. Substituting $z = \sum_{i=1}^{n} u_i^{(i)} x_i^{(i)}$ into this equation, we obtain $P(x)^r = \sum_{j=1}^{n} \hat{A}_j h_{\mu j}^{(i)}$. That is, $P \in \sqrt{\mathbb{D}_\mu^{(i)}} = \mathbb{J}_\mu^{(i)}$. So $\mathbb{I}_\mu^{(i)} \bigcap K[x] \subseteq \mathbb{J}_\mu^{(i)}$. We prove the lemma. ■

As a consequence, we have the following result.

Theorem 6. *For $i > k$, $M_\mu^{(i)} \subset M_\eta^{(k)}$ if and only if* $\mathrm{prem}(\tilde{h}_{\eta j}^{(k)}(x_1^{(i)}, \ldots, x_n^{(i)}), \mathcal{A}_\mu^{(i)}) = 0$ *for $j = 1, \ldots, m_\eta$, where $\mathcal{A}_\mu^{(i)}$ is defined in (13) and $\tilde{h}_{\eta j}^{(k)}(x_1^{(i)}, \ldots, x_n^{(i)})$ is obtained by substituting $\mathbf{x}^{(k)} = M_k \cdots M_{i-1}\mathbf{x}^{(i)}$ into $h_{\eta j}^{(k)}$. As a consequence, we give an irredundant decomposition of variety M with the VDW algorithm.*

Proof. When we consider the including relationship between two varieties, we need to present them in the same coordinate. Note that

$$\mathbf{x}^{(k)} = M_k \cdots M_{i-1}\mathbf{x}^{(i)}$$

is the linear transformation between $x^{(i)}$ and $x^{(k)}$.

If $\mathrm{prem}(\tilde{h}_{\eta j}^{(k)}(x_1^{(i)}, \ldots, x_n^{(i)}), \mathcal{A}_\mu^{(i)}) = 0$ for $j = 1, \ldots, m_\eta$. According to Lemma 5, $\tilde{h}_{\eta j}^{(k)} \in \mathbb{J}_\mu^{(i)}$. So $\mathbb{D}_\eta^{(k)} \subset \mathbb{J}_\mu^{(i)}$. According to the Hilbert's Nullstellensatz, $M_\mu^{(i)} \subset M_\eta^{(k)}$.

On the other hand, if $M_\mu^{(i)} \subset M_\eta^{(k)}$, then $\mathbb{J}_\eta^{(k)} \subset \mathbb{J}_\mu^{(i)}$. So $\mathbb{D}_\eta^{(k)} \subset \mathbb{J}_\eta^{(k)} \subset \mathbb{J}_\mu^{(i)}$. According to Lemma 5, $\mathrm{prem}(\tilde{h}_{\eta j}^{(k)}(x_1^{(i)}, \ldots, x_n^{(i)}), \mathcal{A}_\mu^{(i)}) = 0$. ■

From the above theorem, we have a new and more efficient method to decide the including relationship of irreducible components.

Example 2. Continue Example 1. Compute the irredundant irreducible decomposition of variety $M = \mathrm{Zero}(f_1, f_2, f_3)$. We have computed the irreducible decomposition of the variety in Example 1. We need only remove those irreducible varieties which are included in some higher dimensional varieties.

For $M_1^{(3)}$, the ascending chain corresponding to it is $\mathcal{A}_1^{(3)} = \{z, x_3, x_1, x_2\}$. $\mathrm{prem}(h_{11}^{(2)}, \mathcal{A}_1^{(3)}) = 0$ and $\mathrm{prem}(h_{12}^{(2)}, \mathcal{A}_1^{(3)}) = 0$. So according to Lemma 5 and Theorem 6, we know $M_1^{(3)} \subset M_1^{(2)}$.

Similarly, we can decide $M_2^{(3)} \subset M_1^{(2)}$, $M_3^{(3)} \not\subseteq M_1^{(2)}$ and $M_4^{(3)} \not\subseteq M_1^{(2)}$.

We obtain the following irredundant irreducible decomposition of M:

$$M = M_1^{(2)} \bigcup M_3^{(3)} \bigcup M_4^{(3)}.$$

5 Decomposing Zero-Dimensional Radical Ideals

In this section, we will show that if the given polynomial system is zero-dimensional, then we can decompose the radical ideal generated by them as the intersection of prime ideals with the VDW Algorithm. We also give a modified algorithm using Macaulay resultant to decompose the zero-dimensional radical ideals.

5.1 Decomposing Zero-Dimensional Radical Ideals Using VDW Algorithm

Lemma 6. *Use the notations defined in (11) and (13). Then* $\mathcal{A}_\mu^{(i)}$ *is a char set of* $\mathbb{K}_\mu^{(i)}$.

Proof. Let $p = z - (u_1x_1 + \ldots + u_nx_n)$. Then $h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, z) = h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, p + u_1x_1 + \ldots + u_nx_n) = \bar{h}_\mu^{(i)}(u, x, p)p + h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, u_1x_1 + \ldots + u_nx_n)$, where $\bar{h}_\mu^{(i)}(u, x, p)$ is a polynomial in u, x, p. Since $p, h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, u_1x_1 + \ldots + u_nx_n) \in \mathbb{K}_\mu^{(i)}$, $h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, z)$ is in $\mathbb{K}_\mu^{(i)}$. Use the same method, we have

$$\begin{aligned} & B_k(u, x_1, \ldots, x_{n-i}, z, x_{k+n-i}) \\ = & \frac{dh_\mu^{(i)}}{du_{n-i+k}}(u, x_1, \ldots, x_{n-i}, z) \\ = & \frac{dh_\mu^{(i)}}{du_{n-i+k}}(u, x_1, \ldots, x_{n-i}, p + u_1x_1 + \ldots + u_nx_n) \\ = & B'_k p + \frac{dh_\mu^{(i)}}{du_{n-i+k}}(u, x_1, \ldots, x_{n-i}, u_1x_1 + \ldots + u_nx_n). \end{aligned}$$

Let $h_\mu^{(i)}(u, x_1, \ldots, x_{n-i}, u_1x_1 + \ldots + u_nx_n) = \sum_{j=1}^{s} h_{\mu j}^{(i)} u_1^{r_{j,1}} \ldots u_n^{r_{j,n}}$. Then

$$\begin{aligned} & \frac{dh_\mu^{(i)}}{du_{n-i+k}}(u, x_1, \ldots, x_{n-i}, u_1x_1 + \ldots + u_nx_n) \\ = & \sum_{j=1}^{s} r_{j,n-i+k} h_{\mu j}^{(i)} u_1^{r_{j,1}} \ldots u_{n-i+k}^{r_{j,n-i+k}-1} \ldots u_n^{r_{j,n}} \in \mathbb{K}_\mu^{(i)}. \end{aligned}$$

So $R_{\mu k}^{(i)} = \mathrm{prem}(B_k, h_\mu^{(i)}) \in \mathbb{K}_\mu^{(i)}$. Hence $\mathcal{A}_\mu^{(i)} \subset \mathbb{K}_\mu^{(i)}$. $\mathbb{K}_\mu^{(i)} \subset \mathbb{I}_\mu^{(i)}$, so $\forall g \in \mathbb{K}_\mu^{(i)}$, we have $\mathrm{prem}(g, \mathcal{A}_\mu^{(i)}) = 0$. According to the definition of char sets, $\mathcal{A}_\mu^{(i)}$ is a char set of $\mathbb{K}_\mu^{(i)}$. ■

Theorem 7. *Use the notations introduced above. If the ideal* $\mathbb{D}_\mu^{(i)}$ *is zero-dimensional, that is,* $i = n$*, then* $\mathbb{D}_\mu^{(n)}$ *is prime.*

Proof. We need only to prove $\mathbb{D}_\mu^{(n)} = \mathbb{J}_\mu^{(n)} = \sqrt{\mathbb{D}_\mu^{(n)}}$. It is clear that $\mathbb{D}_\mu^{(n)} \subset \mathbb{J}_\mu^{(n)}$. On the other hand, we need to prove $\mathbb{J}_\mu^{(n)} \subset \mathbb{D}_\mu^{(n)}$. $\mathcal{A}_\mu^{(n)}$ is a char set of $\mathbb{J}_\mu^{(n)}$ by Lemma 6. According to the proof of Lemma 2, we can get a new char set $\tilde{A}_\mu^{(n)} = \{\tilde{R}_{\mu 0}^{(n)}, \tilde{R}_{\mu 1}^{(n)}, \ldots, \tilde{R}_{\mu n}^{(n)}\}$ of $\mathbb{J}_\mu^{(n)}$ where

$$\tilde{R}_{\mu 0}^{(n)} = R_{\mu 0}^{(n)}, \tilde{R}_{\mu i}^{(n)} = Q_i R_{\mu i}^{(n)} + (\sum_{i=0}^{i-1} Q_k R_{\mu k}^{(n)} x_i)(k = 1, \ldots, n) \tag{14}$$

and the initials of $\tilde{R}_{\mu i}^{(n)}$ are in $K[u]$.

If $g(x) \in \mathbb{J}_\mu^{(n)} \subset \mathbb{I}_\mu^{(n)}$, by Lemma 6, there exist Q, a product of the powers of the initials of polynomials in $\tilde{A}_\mu^{(n)}$, and a chain $C_0, C_1, \ldots, C_n \in K[u, x, z]$ such that

$$Q(u)g(x) = \sum_{j=0}^{n} C_j \tilde{R}_{\mu j}^{(n)}.$$

Substituting (14) into the above equation and arranging the right hand polynomial according to $R_{\mu i}^{(n)}$, we get

$$Q(u)g(x) = C_0' R_{\mu 0}^{(n)} + \sum_{j=1}^{n} C_j' R_{\mu j}^{(n)},$$

where $C_i' \in K[u, x, z]$. We have $R_{\mu j}^{(n)} = \mathrm{prem}(B_j, R_{\mu 0}^{(n)})$, where B_j are defined in the proof of Lemma 4. So there exist $Q_j(u) \in K[u]$, a power of the initial of $R_{\mu 0}^{(n)}$ and $P_j \in K[u, x, z]$ such that

$$R_{\mu j}^{(n)} = Q_j B_j - P_j R_{\mu 0}^{(n)} (j = 1, \ldots, n).$$

So the above equation becomes

$$Q(u)g(x) = C_0'' R_{\mu 0}^{(n)} + \sum_{j=1}^{n} C_j'' B_j$$

where $C_i'' \in K[u, x, z]$. Replacing z with $u_1 x_1 + \ldots + u_n x_n$ in the above equation, we have

$$\begin{aligned} Q(u)g(x) &= C_0''(u, x) \textstyle\sum_{j=1}^{s} h_{\mu j}^{(n)} u_1^{r_{j,1}} \ldots u_n^{r_{j,n}} \\ &+ \textstyle\sum_{i=1}^{n} (C_i''(u, x) \sum_{j=1}^{s} r_{j,n-i+k} h_{\mu j}^{(n)} u_1^{r_{j,1}} \ldots u_{n-i+k}^{r_{j,n-i+k}-1} \ldots u_n^{r_{j,n}}). \end{aligned}$$

Substitute $u = u_0 \in K$, we have

$$g(x) = \sum_{j=1}^{s} h_{\mu j}^{(n)}(x) T_j(x) \in \mathbb{D}_\mu^{(n)}. \qquad \blacksquare$$

As a consequence of Theorem 7, we have that if the ideal $\mathbb{I}$ generated by (3) is zero-dimensional, the VDW Algorithm gives a prime decomposition of the radical ideal $\sqrt{\mathbb{I}}$. We write this result as a corollary.

Corollary 1. *Let (3) define a zero-dimensional ideal* $\mathbb{I}$ *and* $\mathbb{D}_\mu^{(n)} = (h_{\mu 1}^{(n)}(x), \ldots, h_{\mu m_\mu}^{(n)}(x))$ *where* $h_{\mu 1}^{(n)}$ *are from (10). Then,* $\sqrt{\mathbb{I}} = \cap_\mu \mathbb{D}_\mu^{(n)}$ *decomposes the radical ideal* $\sqrt{\mathbb{I}}$ *as an intersection of prime ideals.*

Corollary 2. *Let* $\mathbb{H}_\mu^{(i)}$ *be defined in (11). Then the ideal generated by* $\mathbb{H}_\mu^{(i)}$ *in the ring* $K(x_1, \ldots, x_{n-i})[x_{n-i+1}, \ldots, x_n]$ *is prime.*

Proof. Let $\tilde{\mathbb{D}}_\mu^{(i)}$ be the ideal generated by $\mathbb{H}_\mu^{(i)}$ in the ring $K(x_1, \ldots, x_{n-i})[x_{n-i+1}, \ldots, x_n]$. Obviously $\tilde{\mathbb{D}}_\mu^{(i)}$ is a zero-dimensional ideal in $K(x_1, \ldots, x_{n-i})[x_{n-i+1}, \ldots, x_n]$. Then the result is a consequence of Theorem 7 $\blacksquare$

Unfortunately, Theorem 7 is false when ideal $\mathbb{D}_\mu^{(i)}$ is not zero-dimensional as shown by the following example.

Example 3. $f_1 = x_1 + x_2 + x_3, f_2 = x_1^2 + x_2^2 + x_1x_2$.

Using the algorithm given in Sect. 3, we obtain the following polynomials:

$\mathbb{P}^{(1)} = \{f_1, f_2, f_3 = z - u_1x_1 - u_2x_2\}$,
$g_1^{(1)} = -x_2x_1 - x_1^2 - x_2^2, g_2^{(1)} = -z + u_1x_1 + u_2x_2 - u_3x_1 - u_3x_2$,
$h^{(1)} = 1$,
$l_1^{(1)} = -x_2x_1 - x_1^2 - x_2^2, l_2^{(1)} = -z + u_1x_1 + u_2x_2 - u_3x_1 - u_3x_2$,
$e_1^{(1)} = -x_2x_1 - x_1^2 - x_2^2$;
$\mathbb{P}^{(2)} = \{l_1, l_2, e_1\}$,
$g_1^{(2)} = -z^2 + 2zu_1x_1 - zu_3x_1 - u_1^2x_1^2 + u_1x_1^2u_3 - x_1u_2z + x_1^2u_2u_1 + x_1^2u_2u_3 - x_1^2u_2^2 - x_1^2u_3^2$,
$h^{(2)} = -z^2 + 2zu_1x_1 - zu_3x_1 - u_1^2x_1^2 + u_1x_1^2u_3 - x_1u_2z + x_1^2u_2u_1 + x_1^2u_2u_3 - x_1^2u_2^2 - x_1^2u_3^2$,
$h_{11}^{(2)} = -x_2x_1 - x_1^2 - x_2^2, h_{12}^{(2)} = -x_1x_3 - x_2x_1 - 2x_2x_3 + x_1^2, h_{13}^{(2)} = -x_3^2 - x_1x_3 - x_1^2$.

Let $\mathbb{D}_1^{(2)} = (h_{11}^{(2)}, h_{12}^{(2)}, h_{13}^{(2)})$. Although $\sqrt{\mathbb{D}_1^{(2)}} = \sqrt{(h_{11}^{(2)}, h_{12}^{(2)}, h_{13}^{(2)})}$ is prime, $\mathbb{D}_1^{(2)}$ itself is not a prime ideal. Compute the primary decomposition of $\mathbb{D}_1^{(2)}$, we have

$\mathbb{D}_1^{(2)} = (x_1+x_2+x_3, x_1^2+x_2^2+x_1x_2) \cap (2x_2x_3+x_1x_3+x_2x_1, x_1^2, x_2^2+x_2x_1, x_3^2+x_1x_3)$.

5.2 Decomposing Zero-Dimensional Radical Ideals Using Macaulay Resultant

Denote

$$F = \{f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n)\}. \tag{15}$$

In this subsection we always assume that the ideal generated by the polynomials in F is zero-dimensional. We have $m \geq n$. We first consider the case $m = n$.

Denote

$$F^h = \{f_1^h(x_0, x_1, \ldots, x_n), \ldots, f_n^h(x_0, x_1, \ldots, x_n), f_{n+1}^h(x_0, x_1, \ldots, x_n, z)\},$$

where f_i^h are the polynomials obtained from f_i(denote $f_{n+1} = z - \sum_{i=1}^{n} u_ix_i$) by homogenizing them with x_0.

Lemma 7. *Using the notations given above. Assume $V(F^h)$ is still a zero-dimensional variety in the projective space, that is the polynomial system F^h has a finite number of solutions in which $x_0 = 0$. Then we can compute a nonzero polynomial $R(z) \in K[u, z]$ such that all the solutions of F^h vanish this polynomial and any z_0 such that $R(z_0) = 0$ can be extended to a solution of F^h.*

Proof. Using the method in [11], we can compute the Macaulay resultant of the polynomial system F w.r.t $x_1, \ldots, x_n$. Denote it to be $R(z)$. According to the assumption that both $V(F)$ and $V(F^h)$ are zero-dimensional varieties, we know

that $R(z)$ does not vanish identically. From the properties of resultant, we get the conclusions of this lemma. ■

If $< F >$ is a zero-dimensional radical ideal then $R(z)$ has the same properties as the $n-$th partial resultant of (3). So we can compute the prime decomposition of $< F >$ from $R(z)$ according to method in Sect. 3.

When $m > n$, denote

$$f'_i = \sum_{j=1}^{m} c_{ij} f_j, \quad i = 1, \ldots, n,$$

where c_{ij} are random chosen integers. Denote $F' = \{f'_1(x_1, \ldots, x_n), \ldots, f'_n(x_1, \ldots, x_n)\}$.

Lemma 8. *Using the notations given above, we have*

$$< f_1, \ldots, f_m > = < f'_1, \ldots, f'_n >$$

with probability almost 1. As a consequence, $V(F) = V(F')$.

Proof. $< f'_1, \ldots, f'_n > \subseteq < f_1, \ldots, f_m >$ is obviously valid. Conversely, we prove that $< f_1, \ldots, f_m > \subseteq < f'_1, \ldots, f'_n >$ is correct unless c_{ij} are selected to be zeros of a nonzero linear system. $< f_1, \ldots, f_m >$ is a zero-dimensional ideal, so it has a Gröbner basis containing n elements. We denote this Gröbner basis to be $G = \{g_1, \ldots, g_n\}$. There exists a unique set of integers $\{b_{jk}\}$ such that

$$f_j = \sum_{k=1}^{n} b_{jk} g_k, \quad j = 1, \ldots, n.$$

Then

$$f'_i = \sum_{j=1}^{m} \sum_{k=1}^{n} c_{ij} b_{jk} g_k.$$

Denote B to be an $m \times n$ matrix whose element in $i-$th row and $j-$th column is b_{ij} and C to be an $n \times m$ matrix whose element in $j-$th row and $k-$th column are c_{jk}. According to the theory of linear equation systems, if $\det(CB) \neq 0$ then all g_k can be uniquely represented by the linear combinations of $f'_1, \ldots, f'_n$. That is unless $\{c_{ij}\}$ is zeros of $\det(CB) = 0$, $< g_1, \ldots, g_n > \subseteq < f'_1, \ldots, f'_n >$. Now we prove this lemma. ■

If $m > n$, we can use F' to compute the Macaulay resultant $R(z)$ and to decompose ideal into the intersection of prime ideas.

Lemma 9. *Using notations given above, let* $f_1(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \ldots, f_m(x_1, x_2, \ldots, x_n)$ *define a zero-dimensional variety in the affine space and* $f'_1(x_0, x_1, x_2, \ldots, x_n), f'_2(x_0, x_1, x_2, \ldots, x_n), \ldots, f'_m(x_0, x_1, x_2, \ldots, x_n)$ *define a zero-dimensional variety in the projective space, and* $d = \max\{\deg(f_1), \ldots, \deg(f_m)\}$ *where* $\deg(f_i)$ *is the total degree of* f_i *in variables* $x_1, \ldots, x_n$. *Then, using Macaulay resultant, the maximal total degree in* $x_1, \ldots, x_n$ *of polynomials occurring in the computation is bounded by* d, *and the degree of resultant* $R(z)$ *is bounded by* d^n.

Proof. These conclusions are consequences of properties of the Macaulay resultant [11]. ■

Obviously, when the ideal is a zero-dimensional radical ideal, it is more efficient to use Macaulay resultant than using the VDW algorithm.

We summarize the results in this section as the following result.

Theorem 8. *For a zero dimensional polynomial equation system like (15), we can decompose $< F >$ as the irredundant intersection of prime ideals with probability one. Also the polynomials occurring in the computation is bounded in degree by $O(d^n)$.*

Example 4. $f_1 = x_1^2 + x_2^2 + x_3^2 - 4, f_2 = x_1^2 + x_2^2 - x_3^2 - 2, f_3 = (x_1 + x_2 + x_3 - 1)(x_1 + x_2 - x_3 + 2)$.

The ideal I defined by f_1, f_2, f_3 is a zero-dimensional radical ideal. So we can compute the prime decomposition of this ideal using Macaulay resulsant.

First introducing the new polynomial:

$$f_0 = z - u_1x_1 - u_2x_2 - u_3x_3.$$

Then homogenizing f_0, f_1, f_2, f_3 with x_0 and computing the Macaulay resultant of the new polynomial system with respect to the variables x_0, x_1, x_2, x_3:

$$\begin{aligned} h = 16384&(3u_1^2 + 3u_1z + z^2 + 3u_1u_2 + 3u_2^2 + 3u_2z + 3u_3u_1 + 2u_3z + 3u_2u_3 + u_3^2)\\ &(-3u_1^2 + 2z^2 + 6u_1u_2 - 3u_2^2 - 4u_3z + 2u_3^2)\\ &(-u_1^2 + u_1z + z^2 + 3u_1u_2 + u_2z - u_2^2 - u_3u_1 - 2u_3z - u_2u_3 + u_3^2)\\ &(u_1^2 - 4u_1z + 2z^2 + 6u_1u_2 - 4u_2z + u_2^2 - 4u_3u_1 + 4u_3z - 4u_2u_3 + 2u_3^2). \end{aligned}$$

The resultant h has four irreducible factors containing z, we know that I has four prime components and they can be computed from

$$\begin{aligned} h_1 &= 3u_1^2 + 3u_1z + z^2 + 3u_1u_2 + 3u_2^2 + 3u_2z + 3u_3u_1 + 2u_3z + 3u_2u_3 + u_3^2,\\ h_2 &= -3u_1^2 + 2z^2 + 6u_1u_2 - 3u_2^2 - 4u_3z + 2u_3^2,\\ h_3 &= -u_1^2 + u_1z + z^2 + 3u_1u_2 + u_2z - u_2^2 - u_3u_1 - 2u_3z - u_2u_3 + u_3^2,\\ h_4 &= u_1^2 - 4u_1z + 2z^2 + 6u_1u_2 - 4u_2z + u_2^2 - 4u_3u_1 + 4u_3z - 4u_2u_3 + 2u_3^2. \end{aligned}$$

Substituting $z = u_1x_1 + u_2x_2 + u_3x_3$ into above polynomials and arranging them according to the power products of the u, we get four zero-dimensional prime ideals:

$$\begin{aligned} &I_1 =< h_{11}, h_{12}, h_{13}, h_{14}, h_{15}, h_{16} >, I_2 =< h_{21}, h_{22}, h_{23}, h_{24}, h_{25}, h_{26} >,\\ &I_3 =< h_{31}, h_{32}, h_{33}, h_{34}, h_{35}, h_{36} >, I_4 =< h_{41}, h_{42}, h_{43}, h_{44}, h_{45}, h_{46} > . \end{aligned}$$

where

$$\begin{aligned} &h_{11} = 2x_1 + 3 + 3x_3 + 2x_1x_3, && h_{12} = 2x_1x_2 + 3x_1 + 3x_2 + 3, && h_{13} = 3x_1 + 3 + x_1^2,\\ &h_{14} = 2x_2 + 3 + 3x_3 + 2x_2x_3, && h_{15} = 3x_2 + x_2^2 + 3, && h_{16} = x_3^2 + 2x_3 + 1;\\ &h_{21} = -4x_1 + 4x_1x_3, && h_{22} = 4x_1x_2 + 6, && h_{23} = -3 + 2x_1^2,\\ &h_{24} = -4x_2 + 4x_2x_3, && h_{25} = 2x_2^2 - 3, && h_{26} = 2x_3^2 - 4x_3 + 2;\\ &h_{31} = -2x_1 - 1 + x_3 + 2x_1x_3, && h_{32} = 2x_1x_2 + x_1 + 3 + x_2, && h_{33} = -1 + x_1 + x_1^2,\\ &h_{34} = x_3 + 2x_2x_3 - 2x_2 - 1, && h_{35} = -1 + x_2 + x_2^2, && h_{36} = x_3^2 - 2x_3 + 1;\\ &h_{41} = 4x_1 - 4 - 4x_3 + 4x_1x_3, && h_{42} = 4x_1x_2 - 4x_1 + 6 - 4x_2, && h_{43} = 1 - 4x_1 + 2x_1^2,\\ &h_{44} = -4x_3 + 4x_2x_3 + 4x_2 - 4, && h_{45} = 1 - 4x_2 + 2x_2^2, && h_{46} = 2x_3^2 + 4x_3 + 2. \end{aligned}$$

Then we get the prime decomposition of given ideal:

$$I = I_1 \bigcup I_2 \bigcup I_3 \bigcup I_4.$$

6 Conclusions

In this article, we introduce the WDW algorithm, and use this algorithm to solve two problems:

1. The irredundant irreducible decomposition of varieties.
2. The prime decomposition of zero-dimensional radical ideals.

We also give the complexity of this algorithm.

References

1. Buchberger, B.: Gröbner bases: an algorithmic method in polynomial ideal theory. In: Bose, N.K. (ed.) Recent Trends in Multidimensional Systems theory, D.Reidel Publ. Comp. (1985)
2. Canny, J.: Generalised characteristics polynomials. Journal of Symbolic Computation 9, 241–250 (1990)
3. Chistov, A.: Algorithm of polynomial complexity for factoring polynomials and finding the components of varieties in subexponential time. J. Sov. Math. 4, 1838–1882 (1986)
4. Chou, S.C.: Mechanical geometry theorem proving. D.Reidel Publishing Company, Dordrecht (1988)
5. Chou, S.C., Gao, X.S.: Ritt-Wu's decomposition algorithm and geometry theorem proving. In: Stickel, M.E. (ed.) CADE 1990. LNCS, vol. 449, pp. 207–220. Springer, Heidelberg (1990)
6. Elkadi, M., Mourrain, B.: A new algorithm for the geometric decomposition of a variety. In: Proc. of ISSAC 1999, pp. 9–16. ACM Press, New York (1999)
7. Gao, X.S., Chou, S.C.: On the dimension for arbitrary ascending chains. Chinese Bull. of Scis. 38, 396–399 (1993)
8. Gao, X.S., Chou, S.C.: On the theory of resolvents and its applications. Systems Science and Mathematical Sciences 12, 17–30 (1999)
9. Gianni, P., Trager, B., Zacharias, G.: Gröbner bases and primary decomposition of polynomial ideals. Journal of Symbolic Computation 6, 149–167 (1988)
10. Laplagne, S.: An algorithm for the computation of the radical of an ideal. In: Proc. of ISSAC 2006, pp. 191–195. ACM Press, New York (2006)
11. Macaulay, F.S.: The algebraic theory of modular systems. Cambridge University Press, Cambridge (1916)
12. Sausse, A.: A new approach to primary decompositon. Journal of Symbolic Computation 31, 243–257 (2001)
13. Ritt, J.F.: Differential algebra, American Mathematical Society (1950)
14. Szántó, Á.: Computation with polynomial systems, PhD thesis, Cornell University (1999)
15. Van der Waerden, B.L.: Einfürung in die algebraischen geometrie. Springer, Berlin (1973)

16. Van der Waerden, B.L.: Modern algebra II. Frederick Ungar Pub., New York (1953)
17. Wang, D.M.: Irreducible decomposition of algebraic varieties via characteristic sets and Gröbner bases. Computer Aided Geometric Design 9, 471–484 (1992)
18. Wang, D.M.: Decomposing algebraic varieties. In: Wang, D., Yang, L., Gao, X.-S. (eds.) ADG 1998. LNCS (LNAI), vol. 1669, pp. 180–206. Springer, Heidelberg (1999)
19. Wu, W.T.: Basic principles of mechanical theorem-proving in elementary geometries. Journal of System Science and Mathematical Sciences 4, 207–235 (1984); Re-published in Journal of Automated Reasoning 2, 221–252 (1986)

Regular Decompositions

Guillaume Moroz

INRIA, Paris-Rocquencourt Center, SALSA Project
UPMC, Univ Paris 06, LIP6, CNRS, UMR 7606, LIP6
UFR Ingéniérie 919, LIP6 Passy-Kennedy, Case 169, 4, Place Jussieu, F-75252 Paris
guillaume.moroz@lip6.fr

Abstract. We introduce the notion of *regular decomposition* of an ideal and present a first algorithm to compute it. Designed to avoid generic perturbations and eliminations of variables, our algorithm seems to have a good behaviour with respect to the sparsity of the input system. Beside, the properties of the regular decompositions allow us to deduce new algorithms for the computation of the radical and the weak equidimensional decomposition of an ideal. A first implementation shows promising results.

1 Introduction

Let R be a Cohen-Macaulay ring and I an ideal of R. A fundamental problem is to describe geometrically I. Such a description is given by a decomposing the ideal.

Main Result

We present a new kind of decomposition, so called *strict regular decomposition* or SRD. To that end, we introduce in section 2 the notion of *regular set*. The main idea is to decompose I in sequences regular in specific extensions of R.

We present properties verified by regular decompositions. In particular it is easy to compute an equidimensional decomposition from an SRD of I. And in the case where R is a polynomial ring whose coefficient field has a characteristic 0, we show how to obtain the radical of I from its SRD.

The main motivations for SRDs are their computation facilities. Indeed it is possible to obtain an SRD of I without using random perturbations of its generators or variables. Moreover, we present an algorithm avoiding eliminations of variables which can sometime be time consuming. The method is essentially based on saturations of an ideal by a polynomial. A first implementation in the computer algebra system Singular confirms the good behaviour of the algorithm on various systems in comparison of other decomposition methods.

State of the Art

The methods using regular sequences for describing an ideal go back to Bertini [1]. He proved that in characteristic 0, if c is the codimension of I and S a sequence of c generic combinations of the generators, then S is a regular sequence

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 263–277, 2008.
© Springer-Verlag Berlin Heidelberg 2008

on R. In this case, Eisenbud, Huneke and Vasconcelos show in [2] how to compute the radical of the equidimensional hull of I, using a quotient with the Jacobian ideal of S. However, they point out that the random combinations spoils the sparsity of the input. The negative impact on the computation leads them to exhibit another algorithm avoiding random combinations. With this in mind, our SRDs are computed without any generic combinations of the generators of I.

Another tool using regular sequences are the triangular sets. Following the work of Ritt, Wu introduced first triangular sets in [3]. Then Lazard in [4] and Kalkbrener in [5] independently enriched this notion. On overview of these methods and others may be found in [6] and more recent developments are in [7] and [8] among others. A triangular set is a sequence of polynomial $p_1, \ldots, p_k$ regular in an extension R, with the strong condition that p_i is a polynomial in the i first independent variables of the polynomial ring. They are usually computed incrementally, without generic perturbations. Our notion of regular sets may be seen as an extension of the triangular sets, where we only keep the regularity condition. In particular the main difference is that we don't directly eliminate variables while computing regular sets, nor rely on a particular global variable ordering.

In [9], G.Lecerf introduces a different approach and computes equidimensional decompositions by mean of Kronecker parametrizations, which are more structured than triangular sets. His algorithm is also incremental. For each new generator of the ideal, the already computed parametrization is split if necessary and updated. For the computation of the SRD, we use the same incremental strategy using less structured mathematical object. Our main problem is the design of an efficient splitting function.

Let's also mention the work of [10] and later [11] who developed methods based on block order Gröbner basis to compute prime and equidimensional decompositions of an ideal. Following their work, S.Laplagne presents in [12] a method to avoid redundancies appearing during the computations. We see in section 5 that this method can also remove the redundancies in SRDs.

More generally, Kalkbrener presents in [13] a theoretical framework to describe a decomposition algorithm, a key tool being the Split function. One of the main contribution of this article is a new way to handle the Split step.

2 Regular Set

In this article, K will always denote a field and R a Cohen-Macaulay ring.

First, we introduce the notion of regular set, which is mainly a sequence regular on an extension of R. General definitions of Cohen-Macaulay rings and regular sequences may be found in [14].

Notation 1

- *Let E be a multiplicatively closed set in the ring R. We denote by $E^{-1}R$ the ring of fractions of the form r/e, with $r \in R$ and $e \in E$, equipped with the usual operations on the fractions.*

- *Let F be a subset of the ring R. We denote by $\widetilde{F}$ the multiplicative closure of F in R.*

$$\widetilde{F} = \{f_1^{n_1} \cdots f_k^{n_k} | k \geq 1, n_1 \geq 0, \ldots, n_k \geq 0, f_1 \in F, \ldots, f_k \in F\}$$

- *If I is an ideal of $K[X_1, \ldots, X_n]$, it is customary to write the zero set of I in K_c^n as $\mathcal{V}(I)$, where K_c is the algebraic closure of K.*
- *If V is a set of K_c^n, then $\overline{V}$ denotes its Zariski closure.* □

Definition 1. *(Regular sets)*
Let R be a Cohen-Macaulay ring. Let S be the sequence $s_1, \ldots, s_k$ of k polynomials in R and F a finite subset of R.

The pair (S, F) is called a regular set *if S is a regular sequence in $\widetilde{F}^{-1}R$.*

Definition 2. *(Regular sets notions)*

We call saturated ideal *of (S, F) and denote by $\mathcal{I}(S, F)$ the ideal*

$$\langle S \rangle : \prod_{f \in F} f^{\infty}$$

If R is a polynomial ring over K, then:

We call algebraic zeros *of (S, F) and denote by $\mathcal{Z}(S, F)$ the variety*

$$\overline{\mathcal{V}(S) \setminus \bigcup_{f \in F} \mathcal{V}(F)}$$

We call constructible zeros *of (S, F) and denote by $\mathcal{C}(S, F)$ the variety*

$$\mathcal{V}(S) \setminus \bigcup_{f \in F} \mathcal{V}(F)$$

The number of polynomials in the sequence S will be called the height *of (S, F), and the total degree of the variety $\mathcal{Z}(S, F)$ will be referred to as the* degree *of (S, F).*

We can now define the main notion of this article.

Definition 3. *(Regular decomposition) Let I be an ideal of R. If there exist k regular sets $(S_i, F_i), 1 \leq i \leq k$ such that:*

$$\sqrt{I} = \bigcap_{i=1}^{k} \sqrt{\mathcal{I}(S_i, F_i)}$$

or equivalently when R is a polynomial ring such that:

$$\mathcal{V}(I) = \bigcup_{i=1}^{k} \mathcal{Z}(S_i, F_i)$$

then the set of pairs (S_i, F_i) is called regular decomposition *of I.*

Moreover, if the constructible zeros *$\mathcal{C}(S_i, F_i)$ are pairwise disjoint, the regular sets are said to form a* strict regular decomposition *also called* SRD *of I.*

In section 4, we prove that such a decomposition always exists by exhibiting an algorithm computing an SRD of any ideal.

3 Applications

In this section, we present some useful properties of the *regular decomposition*.

Property 1. (Equi-dimensional)
Let (S, F) be a regular set. Then $\mathcal{I}(S, F)$, the saturated ideal of (S, F), is equidimensional and its codimension is the number of polynomials of the sequence S.

Thus the *height of* (S, F) equals the codimension of $\mathcal{I}(S, F)$.

This result is deduced from the properties of the regular sequences. As a corollary, given D an SRD of an ideal I, the set of the $\mathcal{I}(S, F)$ for $(S, F) \in D$ is a weak equidimensional decomposition of I.

Property 2. (Radical computation)
Let R be a polynomial ring $K[X_1, \ldots, X_n]$ and K a field of characteristic 0. Let (S, f) be a regular set of height k of R and J be the Jacobian ideal generated by the $k \times k$ minors of the Jacobian matrix of S. Then, the ideal

$$\mathcal{I}(S, F) : J$$

is the radical of $\mathcal{I}(S, F)$.

Remark 1. We remind that in the general case, the lemma is wrong. For example, let I be the ideal generated by X^2, Y^2 and XY in $\mathbb{Q}[X, Y]$.I is 0-dimensional and its Jacobian matrix M is:

$$\begin{bmatrix} 2X & 0 \\ 0 & 2Y \\ Y & X \end{bmatrix}$$

The Jacobian ideal J is generated by the 2×2 minors of M: $4XY$, $2X^2$ and $-2Y^2$. Thus $I : J$ is the trivial ideal generated by 1 while the radical of I is generated by X and Y.

Proof. Since R is regular in $\widetilde{F}^{-1}R$, $\mathcal{I}(S, F)$ is generically in complete intersection. Hence the theorem 2.1 of [2] allows us to conclude.

This lemma allows to transform an SRD in the radical of an ideal I.

Now let's see how to construct such a strict regular decomposition.

4 Algorithm

As we recalled in the introduction, different methods exist to decompose an ideal I. Here we propose a new method based on saturations, involving no generic transformation, and no elimination of the input variables.

Our base algorithm handles incrementally the polynomials of the input. We do not need a global variable ordering. We assume that we know how to saturate, compute the dimension and test the radical membership of any ideal in R.

When R is a polynomial ring, these operations can be done by linear algebra methods in polynomial space ([15]) or by gröbner bases computations. The table 1 in the last section shows that in practice, using gröbner bases is not necessarily expensive, and usually cheaper than the worst theoretical exponential space case.

4.1 Main Idea of the Algorithm

Let $g_1, \ldots, g_m$ be a list of generators of I. At each step we compute a strict regular decomposition of $\langle g_1, \ldots, g_c \rangle$ based on a strict regular decomposition of $\langle g_1, \ldots, g_{c-1} \rangle$.

We begin with a simple regular set defined as the pair of the sequence (g_1) and the empty set. This regular set forms trivially a regular decomposition of the ideal $\langle g_1 \rangle$.

At step c, let D_{c-1} be a strict regular decomposition of $\langle g_1, \ldots, g_{c-1} \rangle$. For $(S, F) \in D_{c-1}$, let J_{SF} be the ideal generated by S and g_c, and saturated by the polynomials in F:

$$J_{SF} = (\langle S \rangle + \langle g_c \rangle) : \prod_{f \in F} f^{\infty}$$

The incremental step consists in computing SRDs of the ideals J_{SF} for all $(S, F) \in D_{c-1}$, such that their union forms a new strict regular decomposition D_c of $\langle g_1, \ldots, g_c \rangle$.

4.2 Recursion Step

Main case distinction

Let (S, F) be a regular set, p a polynomial, and J_{SF} the ideal defined as:

$$J_{SF} = (\langle S \rangle + \langle p \rangle) : \prod_{f \in F} f^{\infty}$$

Moreover, let R' be the ring of fraction $\widetilde{F}^{-1}R$, and M be the quotient $R'-$ module R'/SR'.

Now we want to compute a strict regular decomposition of J_{SF}, so we need to consider two cases:

- p is a nonzerodivisor on M.
- p is a zerodivisor on M

The first case is the easy one. The sequence $S' = S, p$ is regular on R', such that (S', F) is a regular set. And the algebraic zeros of (S', F) trivially equal the zeros of J_{SF}, such that (S', F) forms an SRD of J_{SF}.

In the second case, we split J_{SF} in two regular sets. The idea is to compute a polynomial $h \in \mathcal{I}(S, F) : p^{\infty}$ such that $p + h$ is a nonzerodivisor on M. We will see in the next section that such a h exists and how to construct it. Then :

$$(S, F \cup \{h\}) \text{ and } ((S, h), F)$$

are two regular sets. And we will prove that their constructible zeros are disjoint and the union of their algebraic zeros is exactly $\mathcal{V}(J_{SF})$.

Split Algorithm

We need some preparatory lemmas.

Lemma 1. *Let R be a Cohen-Macaulay ring, and I an ideal generated by a regular sequence in R. Let p be a zerodivisor on the quotient module R/IR.*
Then $I : p^\infty$ is not included in $\sqrt{I}$.

Remark 2. Note that if R is a Cohen-Macaulay ring and F is a multiplicatively closed set, then $F^{-1}R$ is also a Macaulay ring [16].

Proof. I being generated by a regular sequence, its associated primes $ass(I)$ are isolated and $ass(I) = ass(\sqrt{I})$. Moreover, p is a zerodivisor on R/IR and is thus contained in an associated prime $\mathcal{P}$ of I. In particular, $I : p^\infty$ is not included in $\mathcal{P}$, neither in $\sqrt{I}$.

Lemma 2. *Let I be an ideal and p, q two polynomials of a polynomial ring such that $pq \in \sqrt{I}$.*
Then, the following equality holds: $\sqrt{I + \langle p, q\rangle} = \sqrt{I + \langle p + q\rangle}$.

Proof. The inclusion from left to right is obvious. For he other side, we remark that pq, $p(p+q)$ and $q(p+q)$ belongs to $\sqrt{I + \langle p+q\rangle}$, and thus p^2 and q^2 too, which achieves the proof.

Geometrically speaking, this means that if $\mathcal{V}(I) \subset \mathcal{V}(p) \cup \mathcal{V}(q)$, then $\mathcal{V}(I) \cap \mathcal{V}(p) \cap \mathcal{V}(q) = \mathcal{V}(I) \cap \mathcal{V}(p+q)$. This is the key tool that will allow us to get a regular sequence without using a full random combination of the generators.

Our split algorithm is the main part of the recursive step. Given a regular set (S, F) and a polynomial p, let I be the ideal generated by S in $R' := \widetilde{F}^{-1}R$. The following algorithm computes a polynomial $h \in I : p^\infty$ such that $p + h$ is a nonzerodivisor on R'/IR'.

Algorithm 1 *(*Split*)*
Input*: a regular set (S, F) and a polynomial p in a ring R*
($R' := \widetilde{F}^{-1}R$ and I is the ideal generated by S in R')
Output*: a polynomial h in $\mathcal{I}(S, F) : p^\infty$*
such that $p + h$ is a nonzerodivisor on R'/SR'

- *h:=0*
- while *$p + h$ is a zerodivisor on R'/IR'* do
  - *J:=$I : (p + h)^\infty$*
  - choose $g \in \left(J \setminus \sqrt{I}\right) \cap R$
  - *h:=$h + g$*
- done
- return h □

Remark 3

- $p+h$ is a zerodivisor on R'/IR' if and only if the height of $I + \langle p + h\rangle$ equals the height of (S, F), which allows us to test this condition by computing the dimension of an ideal.

- The operation CHOOSE consists in testing if the generators of J belongs to $\sqrt{I}$ until we find one outside $\sqrt{I}$.
- Note that we do not need to compute fully J. The saturation of I by $p+h$ may be stopped as soon as we get a polynomial g which does not belong to $\sqrt{I}$.
- The number of iteration of the loop is bounded by the number of primes associated to I, as proven in the following proof of correctness,

Proof. of correctness For the correctness of this algorithm, we need to prove that h is actually in $\mathcal{I}(S,F) : p^\infty$, and that the algorithm will eventually stop.

The first assertion comes from the fact that J is always included in $I : p^\infty$. Indeed, let $h \in I : p^\infty$, then if $x \in I : (p+h)^\infty$, it is easy to check that $x \in I : p^\infty$, such that $J \subset I : p^\infty$. Thus if $h \in I : p^\infty$ then $g + h$ also belongs to $I : p^\infty$ which allows us to conclude by recurrence that the returned polynomial is in $I : p^\infty \cap R = \mathcal{I}(S,F) : p^\infty$.

To prove the termination of the algorithm we use the associated primes of I, denoted by $ass(I)$. More precisely, if $p + h$ is a zerodivisor on I, let k be the number of primes containing $p + h$. We write

$$ass(I) = \{P_1, \ldots, P_k, Q_1, \ldots, Q_s\}$$

such that $p + h \in \bigcap_{i=1}^{k} P_i$ and $I : (p + h)^\infty \subset \bigcap_{i=1}^{s} Q_i$.

As in the algorithm, let $g \in I : (p + h)^\infty$ be a polynomial such that $g \notin \sqrt{I}$. We show that in this case $p + h + g$ is contained at most in $k - 1$ associated primes of I. For all $1 \leq i \leq s$,

$$g \in Q_i, p + h \notin Q_i \Longrightarrow p + h + g \notin Q_i$$

Beside, since $g \notin \sqrt{I}$ there exists $1 \leq i_0 \leq k$ such that $g \notin P_{i_0}$.This implies that $p + h + g \notin P_{i_0}$ and may only be included in at most $k - 1$ primes of I.

By recurrence, we see that the number of associated primes of I containing $p + h$ decreases strictly at each loop until it reaches zero, in which case $p + h$ is a nonzerodivisor on R'/IR' and the computation stops.

Now let's come back to our main algorithm.

Lemma 3. *Let (S, F) be a regular set. Using the notations of the beginning of section 4.2, let p be a polynomial of R zerodivisor on R'/IR', and h be the polynomial returned by the* SPLIT *algorithm.*

Then $\{(S, F \cup \{h\}), ((S, p + h), F)\}$ is a strict regular decomposition of J_{SF}.

Proof. To prove that the two regular sets form a regular decomposition of J_{SF}, it is sufficient to prove that the following equality holds in R',

$$\sqrt{I + \langle p \rangle} = \sqrt{I} : h^\infty \cap \sqrt{I + \langle p + h \rangle}$$

Since $h \in I : p^\infty$, $ph \in \sqrt{I}$ and the lemma 2 allows us to conclude that $\sqrt{I + \langle p + h \rangle} = \sqrt{I + \langle p, h \rangle}$. Beside, $p \in \sqrt{I} : h^\infty$. Thus, we have $\sqrt{I + \langle p \rangle} \subset$

$\sqrt{I} : h^\infty \cap \sqrt{I + \langle p + h \rangle}$. For the other inclusion, let $x \in \sqrt{I} : h^\infty \cap \sqrt{I + \langle p + h \rangle}$. Then there exists $q \in I$,$r \in R'$ and integers k, l such that

$$\begin{cases} x^k = q + r(p + h) \\ h^l x^k \in I \end{cases}$$

Thus we have $h^l r(p + h) \in I$, and in particular, $hr \in \sqrt{I + \langle p \rangle}$. This allows us to conclude that $x \in \sqrt{I + \langle p \rangle}$ and proves the inclusion.

Finally, to prove that the regular decomposition is strict, we remark that since $h \in \sqrt{I + \langle p + h \rangle}$, the constructible zeros of $((S, p + h), F)$ are distinct from those of $(S, F \cup \{h\})$.

4.3 Complete Algorithm

The complete algorithm may be written recursively as follow.

Algorithm 2 *(*SRD*)*
INPUT*:- a regular set* (S, F)
- *the list of the remaining generators* $g_c, \ldots, g_m$
OUTPUT*: a* SRD *of* $(\langle S \rangle + \langle g_c, \ldots, g_m \rangle) : \prod_{f \in F} f^\infty$

- IF *the list of remaining generators is empty* THEN
  - RETURN $\{(S, F)\}$
- J:=$\mathcal{I}(S, F)$
- IF g_c *is a nonzerodivisor on* R/JR THEN
  - S':=S, g_c
  - RETURN $\mathrm{SRD}((S', F), (g_{c+1}, \ldots, g_m))$
- ELSE
  - h:=SPLIT$((S, F), g_c)$
  - S':=$S, p + h$
  - F':=$F \cup \{h\}$
  - RETURN $\mathrm{SRD}((S, F'), (g_{c+1}, \ldots, g_m)) \cup \mathrm{SRD}((S', F), (g_{c+1}, \ldots, g_m))$ □

Proof. of correctness
We show by recurrence on the number of remaining polynomials that the function SRD computes correctly a strict regular decomposition of

$$(\langle S \rangle + \langle g_c, \ldots, g_m \rangle) : \prod_{f \in F} f^\infty$$

First, if the remaining list is empty, then SRD returns $\{(S, F)\}$, which is trivially a strict regular decomposition of $(\langle S \rangle) : \prod_{f \in F} f^\infty$.

Now we suppose that the output of SRD is correct when the number of remaining polynomials is $m - c$, and show it is still correct for $m - c + 1$ polynomials. The input of SDR is (S, F) and $(g_c, \ldots, g_m)$. If g_c is a nonzerodivisor on $R/\mathcal{I}(S, F)R$, then the recurrence assumption allows us directly to conclude.

If not, the choice of the polynomial h and the recurrence assumption assures us as in lemma 3 that the union of the SRDs of

$$J_1 := (\langle S\rangle + \langle g_{c+1}, \ldots, g_m\rangle) : \left(h \prod_{f\in F} f\right)^\infty$$

and

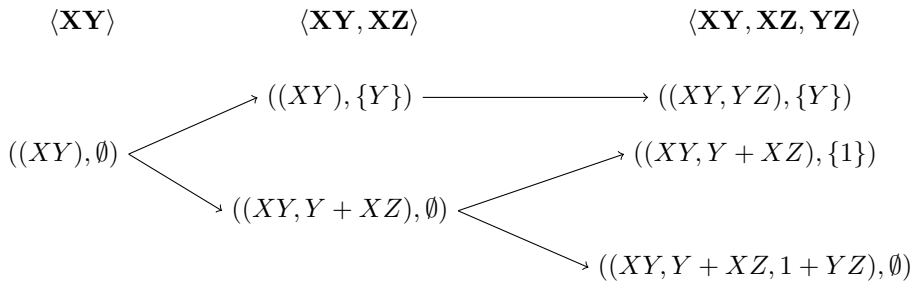
$$J_2 := (\langle S\rangle + \langle g_c + h, g_{c+1}, \ldots, g_m\rangle) : \prod_{f\in F} f^\infty$$

is a regular decomposition of $(\langle S\rangle + \langle g_c, \ldots, g_m\rangle) : \prod_{f\in F} f^\infty$. Moreover, this decomposition is strict since $\mathcal{V}(J_2) \subset \mathcal{V}(h)$ and for all regular set (S, F) in the decomposition of J_1, we have $h \in F$.

4.4 Examples

Here, we take a small examples to see how the basic algorithm work.

Example 1.
Our ring is $\mathbb{Q}[X, Y, Z]$. We consider the ideal $I = \langle XY, XZ, YZ\rangle$



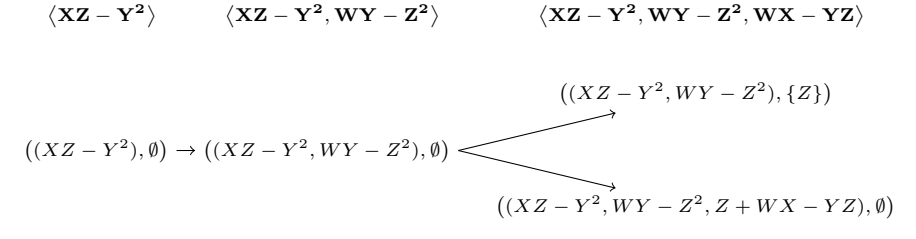
In the diagram, the first line represents the ideal $\langle g_1, \ldots, g_c\rangle$ for c from 1 to the number m of generator of the ideal. On each level of the tree (column), we can read the successive regular decompositions of $\langle g_1, \ldots, g_c\rangle$ for $1 \le c \le m$.

Finally in this example, we end with 3 components whose algebraic zeros are respectively:

- a line: the y-axis
- two lines: the x and z axis
- the empty set

Here is another example: the twisted curve.

Example 2. (Twisted curve)
$I = \left\langle XZ - Y^2, WY - Z^2, WX - YZ\right\rangle \subset \mathbb{Q}[W, X, Y, Z]$.



In this decomposition, we may notice that if we obtain an SRD of the ideal defining the twisted curve, the algebraic zeros associated are redundant. We see in section 5 how to avoid such redundancies.

5 Optimizations

In this section, we will see how to use known methods to improve our algorithm. The modified algorithm won't be purely incremental anymore. The three first subsections are dedicated to improve the practical behaviour of the algorithm. The two last subsection show how to compute an SRD whose algebraic zeros are not redundant.

5.1 Pruning the Tree (I)

In the algorithm stated previously, we compute a tree of regular sets. However, each time we split a regular set, we may create branches which will lead to regular sets with empty algebraic zeros. We show how to prune these branches.

Let I be an ideal generated by $g_1, \ldots, g_m$, and (S, F) be a regular set of the SRD of $\langle g_1, \ldots, g_c \rangle$ with $c \leq m$. By construction, its child in the computation tree form an SRD its saturated ideal. Thus, if D_m is an SRD of I, the set D_{SF} of regular sets $(S', F') \in D_m$ who are descendants of (S, F) in the computation tree verifies the following equality:

$$\bigcup_{(S',F') \in D_{SF}} \mathcal{Z}(S', F') = \mathcal{V}\left((\langle S \rangle + I) : \prod_{f \in F} f^{\infty} \right)$$

This gives us a useful criterion to prune the tree.

Algorithm 3 *(Pruning test)*
INPUT*: a regular set* (S, F) *and an ideal* I
OUTPUT*:* PRUNE REJECT*,*PRUNE KEEP*(*G*) or* CONTINUE

- G*:=*$(I + \langle S \rangle) : \prod_{f \in F} f^{\infty}$
- IF $G = \langle 1 \rangle$ THEN
  - RETURN PRUNE REJECT
- ELSE IF *the number of generators of* G *equals the height of* (S, F) THEN
  - RETURN PRUNE KEEP*(*G*)*
- ELSE
  - RETURN CONTINUE □

5.2 Degree ordering

As a first step, we can compute a Gröbner basis of the ideal I we want to decompose and sort the generators from the smallest to the highest degree. This may allow us to work with intermediate ideals of smaller degrees.

Beside, in order to reduce the number of input polynomials, we may also remove all polynomials g_c included in $\sqrt{g_1, \ldots, g_{c-1}}$. Else this step is done during

the computation of SRD for all the regular sets of the regular decomposition of $\sqrt{g_1, \ldots, g_{c-1}}$.

5.3 Fraction Field

When the working field is a fraction field $K(T_1, \ldots, T_s)$, most of the tests in our algorithm may be stepped-up by specializing the parameters by random values. Indeed, in the SRD and SPLIT algorithm, the test of zerodivision may be done with specialized parameters, as well as the membership test to the radical of an ideal. The pruning test may also be done with specialized parameters.

Finally, in the SPLIT algorithm, the saturation of I by $p + h$ may first be done with the parameters specialized in order to test the minimal degree bound d necessary to obtain a polynomial outside of $\sqrt{I}$. Then we can use d to bound the degrees in the saturation process with symbolic parameters.

Thus this algorithm seems well-suited for computing in fraction coefficients field at the expense of using probabilistic tests.

Remark 4. If $K = \mathbb{Q}$, we can also choose a random prime number μ and compute the tests in characteristic μ.

5.4 Zero-Dimensional Case

As seen in section 3, we can derive a radical equidimensional decomposition of an ideal from its SRD. In the zero-dimensional case the SRD of an ideal I has the good property of not being redundant. Indeed:

Lemma 4. *Let (S, F) be a regular set of $K[X_1, \ldots, X_n]$, of height n. Then the following equality hold:*

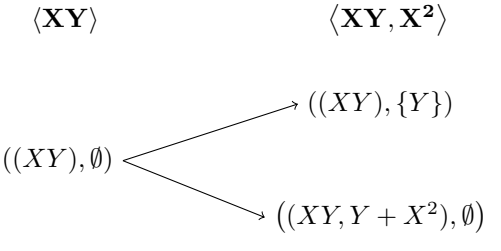
$$\mathcal{Z}(S, F) = \mathcal{C}(S, F)$$

Proof. Under the assumptions, $\mathcal{Z}(S, F)$ is a 0-dimensional variety and $\mathcal{C}(S, F) \subset \mathcal{Z}(S, F)$. If the inclusion is strict then there exists an isolated point p in $\mathcal{Z}(S, F) \setminus \mathcal{C}(S, F)$ and $p \notin \overline{\mathcal{C}(S, F)} = \mathcal{Z}(S, F)$, which is a contradiction.

Since for an SRD, the constructible zeros of its regular sets are pairwise disjoint, this implies no redundancy of the algebraic zeros.

Remark 5. In the general case, we should remind that this decomposition may be redundant as we may see in the following example.

Example 3. Let $I = \langle XY, X^2 \rangle \subset \mathbb{Q}[X, Y]$. Here is the computation tree produced by our algorithm computing an SRD of I.



In this example, the SRD of I is a set of two regular sets, whose algebraic zeros are redundant:

- $\mathcal{Z}((XY), \{Y\}) = \mathcal{V}(\langle X \rangle)$
- $\mathcal{Z}((XY, Y + X^2), \emptyset) = \mathcal{V}(\langle X, Y \rangle)$

Thus, with the previous optimizations, while computing an SRD of a zero-dimensional ideal, the algorithm computes no useless regular set. Moreover if the coefficient field is a fraction field, then we can use the optimizations of section 5.3.

5.5 Pruning the Tree (II)

Finally, using the ideas of [12] we can compute an SRD whose algebraic zeros are not redundant. Roughly, it consists in computing regular sets of higher dimension first in the tree. The detailed algorithm is as follow:

Algorithm 4
INPUT: *An ideal I of a polynomial ring $K[X_1, \ldots, X_n]$*
OUTPUT: *A SRD of I with no redundant algebraic zeros*

- $D := \emptyset$
- $P := \langle 1 \rangle$
- WHILE NOT $P \subset \sqrt{I}$
  - CHOOSE $g \in P \setminus \sqrt{I}$
  - $J := I : g^\infty$
  - FIND A MAXIMAL INDEPENDENT SET $X_{i_1}, \ldots, X_{i_s}$ *with respect to* J.
  - $D_0 :=$SRD(J) *over the fraction field* $K(X_{i_1}, \ldots, X_{i_s})$
  - $D_c :=$CONTRACT *the regular sets of* D_0 *to* $K[X_1, \ldots, X_n]$
  - $P := P \cap \bigcap_{(S,F) \in D_c} \mathcal{I}(S, F)$
  - $D := D \cup D_c$
- DONE
- RETURN D □

Given a regular set (S, F), the contraction from $K(X_1, \ldots, X_s)[X_{s+1}, \ldots, X_n]$ to $K[X_1, \ldots, X_n]$ of (S, F) is a crucial step of the algorithm. Let S_c be a sequence of polynomials in $K[X_1, \ldots, X_n]$, equal to S up to multiples in $K[X_1, \ldots, X_s]$. Let f be a polynomial in $K[X_1, \ldots, X_s]$ and in the prime components $\mathcal{I}(S_c, F) \setminus \mathcal{I}(S_c, F)^{ec}$, computed as in Chapter 8, section 7 of [17] for example. Then, the output of CONTRACT(S, F) is $(S_c, F \cup \{f\})$.

6 Practical Behaviour

To validate our approach, we implemented our optimized algorithm in SINGULAR. Using the examples of [18], we present the time of different available decompositions methods. Here is a summary of the different functions implemented in Singular (more details may be found in [18]):

- `minAssGTZ` computes the minimal associated primes of an ideal using the algorithm of [10].
- `minAssChar` computes the minimal associated primes using Ritt-Wu characteristic sets.
- `equidim` computes a weak equidimensional decomposition using Gröbner basis properties presented in [10] or [11]. The embedded components may be replaced by others with the same radical.
- `equidim-EHV` computes a weak equidimensional decomposition using the algorithm of [2].

Table 1. Time of equidimensional decompositions (in hundredths of second)

	minAssGTZ	minAssChar	equidim	equidim-EHV	SRD
DGP1	13	7	1	1	25
DGP2	20	16	41	16	42
DGP3	3	1	4	5	3
DGP4	9	3	4	2	7
DGP5	37	*	238	*	88
DGP6	7	19	473	*	213
DGP7	12	32	16	16	24
DGP8	3	397	1	1	6
DGP9	32	1916	1	1	6
DGP10	8	*	0	1	7
DGP11	*	*	8	6	64
DGP12	43	*	0	0	3
DGP13	19	*	0	1	6
DGP14	4	5	2	1	0
DGP15	22	281	1	0	47
DGP16	330	3721	153	143	410
DGP17	99	*	0	0	5
DGP18	6	213	1	1	11
DGP19	7	*	4	3	14
DGP20	8	304	5	195	26
DGP21	1	1	10	13	5
DGP22	13	13	59	*	42
DGP23	47	40	30	*	44
DGP24	4	8	9	20	3
DGP25	55	142	921	*	242
DGP26	28	*	0	1	16
DGP27	6	21	0	0	2
DGP28	13	11	1	0	1
DGP29	2	0	2499	*	4
DGP30	91	46	8	*	30
DGP31	5	2	0	0	1
DGP32	5	5	68	*	19
DGP33	4	3	2	1	11
DGP34	*	*	3	3	62

* Means that the computation took more than 60 seconds.
(the cpu is a 32 bits, 2.8GHz Intel pentium)

Finally, SRD denotes the algorithm presented in this paper.

One can see on the table 1 that an SRD computation may be faster on some examples, and slower on others. Beside, in most of the cases the order of magnitude is less than one second and the running time never goes beyond 5 seconds.

Acknowledgements

I would like to thank the referees for their accurate remarks and careful review.

References

1. Bertini, E.: Sui sistemi lineari. Istit. Lombardo Accad. Sci. Lett. Rend. A Istituto 15(II), 24–28 (1982)
2. Eisenbud, D., Huneke, C., Vasconcelos, W.: Direct methods for primary decomposition. Invent. Math. 110(2), 207–235 (1992)
3. Wu, W.J.: On zeros of algebraic equations—an application of Ritt principle. Kexue Tongbao (English Ed.) 31(1), 1–5 (1986)
4. Lazard, D.: A new method for solving algebraic systems of positive dimension. Discrete Appl. Math. 33(1-3), 147–160 (1991); Applied algebra, algebraic algorithms, and error-correcting codes, Toulouse (1989)
5. Kalkbrener, M.: Three Contributions to Elimination Theory. Technical report, Johannes Kepler University, Linz, Austria (1991)
6. Aubry, P., Moreno Maza, M.: Triangular sets for solving polynomial systems: a comparative implementation of four methods. J. Symbolic Comput. 28(1-2), 125–154 (1999); Polynomial elimination—algorithms and applications
7. Wang, D.: Computing triangular systems and regular systems. J. Symbolic Computation 30(2), 221–236 (2000)
8. On triangular decompositions of algebraic varieties, Technical Report TR 4/99, NAG Ltd., Oxford, UK. Presented at the MEGA 2000 (1999)
9. Lecerf, G.: Computing an equidimensional decomposition of an algebraic variety by means of geometric resolutions. In: Proceedings of ISSAC 2000. ACM, New York (2000)
10. Gianni, P., Trager, B., Zacharias, G.: Gröbner bases and primary decomposition of polynomial ideals. J. Symbolic Comput. 6(2-3), 149–167 (1988); Computational aspects of commutative algebra
11. Caboara, M., Conti, P., Traverso, C.: Yet another ideal decomposition algorithm. In: Mattson, H.F., Mora, T. (eds.) AAECC 1997. LNCS, vol. 1255, pp. 39–54. Springer, Heidelberg (1997)
12. Laplagne, S.: An algorithm for the computation of the radical of an ideal. In: ISSAC 2006, pp. 191–195. ACM, New York (2006)
13. Kalkbrener, M.: Algorithmic properties of polynomial rings. J. Symbolic Comput. 26(5), 525–581 (1998)
14. Eisenbud, D.: Commutative algebra with a view toward algebraic geometry. Graduate Texts in Mathematics, vol. 150. Springer, Heidelberg (1994)
15. Matera, G., Turull Torres, J.M.: The space complexity of elimination theory: upper bounds. In: Foundations of computational mathematics, pp. 267–276. Springer, Heidelberg (1997)

16. Kaplansky, I.: Commutative rings. Revised edn. The University of Chicago Press, Chicago (1974)
17. Becker, T., Weispfenning, V.: Gröbner bases. Graduate Texts in Mathematics, vol. 141. Springer, New York (1993)
18. Decker, W., Greuel, G.M., Pfister, G.: Primary decomposition: algorithms and comparisons. In: Algorithmic algebra and number theory, pp. 187–220. Springer, Heidelberg (1997/1999)

Floating-Point Gröbner Basis Computation with Ill-conditionedness Estimation*

Tateaki Sasaki[1] and Fujio Kako[2]

[1] Institute of Mathematics, University of Tsukuba
Tsukuba-shi, Ibaraki 305-8571, Japan
sasaki@math.tsukuba.ac.jp
[2] Department of Comp. Sci., Nara Women's University
Nara-shi, Nara 630-8506, Japan
kako@ics.nara-wu.ac.jp

Abstract. Computation of Gröbner bases of polynomial systems with coefficients of floating-point numbers has been a serious problem in computer algebra for many years; the computation often becomes very unstable and people did not know how to remove the instability. Recently, the present authors clarified the origin of instability and presented a method to remove the instability. Unfortunately, the method is very time-consuming and not practical. In this paper, we first investigate the instability much more deeply than in the previous paper, then we give a theoretical analysis of the term cancellation which causes loss of accuracy in various cases. On the basis of this analysis, we propose a practical method for computing Gröbner bases with coefficients of floating-point numbers. The method utilizes multiple precision floating-point numbers, and it removes the drawbacks of the previous method almost completely. Furthermore, we present a practical method of estimating the ill-conditionedness of the input system.

1 Introduction

Algebraic computation of polynomials with floating-point numbers is a recent hot theme in computer algebra, and many works have been done on the approximate GCD (greatest common divisor), on the approximate polynomial factorization, and so on [15]. However, computation of Gröbner bases with floating-point numbers (*floating-point Gröbner bases*, in short) is just at the beginning of research, although it is a very important theme in approximate algebraic computation (*approximate algebra*). There are two kinds of floating-point Gröbner bases: the first kind is where the coefficients of input polynomials are exact (algebraic numbers or real/complex numbers) but we approximate them by floating-point numbers for some reasons, and the second kind is where the coefficients are inexact hence we express them by floating-point numbers. This paper deals with the second kind.

* Work supported in part by Japan Society for the Promotion of Science under Grants 19300001.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 278–292, 2008.
© Springer-Verlag Berlin Heidelberg 2008

The first kind of floating-point Gröbner bases were studied by Shirayanagi and Sweedler [11], [12], [13]. The second kind of floating-point Gröbner bases were studied by Stetter [14], Fortuna, Gianni and Trager [5], Traverso and Zanoni [18], [17], Weispfenning [19], Kondratyev, Stetter and Winkler [8], Gonzalez-Vega, Traverso and Zanoni [6], Stetter [16], Bodrato and Zanoni [2], Mourrain and his coworkers [9], and so on. How to compute floating-point Gröbner bases stably was, however, an open problem for many years. A breakthrough was attained recently by [10], in which the authors clarified the origin of instability of computation and proposed a stable method.

According to [10], there are two origins of instability: one is main-term cancellation (for main terms, see the beginning of Subsect. 2.1), and the other is the appearance of fully erroneous terms (the leading digit is an error). In the computation of Gröbner bases, the main terms of two polynomials sometimes cancel one another in the subtraction, causing large numerical errors. The main-term cancellation is often exact, and exact cancellation with floating-point numbers usually yields a fully erroneous term. If a fully erroneous term appears as a leading term, subsequent computation will be fully wrong.

In [10], the authors classified the main-term cancellation into two types, cancellation due to *self-reduction* and *intrinsic cancellation*. Self-reduction is caused by a polynomial with small or large leading term, just as the elimination by a small pivot row causes large cancellations in Gaussian elimination. The numerical errors due to self-reduction are avoidable, as we will explain later. The intrinsic cancellation is similar to cancellation which occurs in ill-conditioned numerical matrix; see Example 1 in Sect. 2. We want to know the amounts of intrinsic cancellations. One reason is that the accuracy of floating-point Gröbner basis is reduced by the amounts. Another reason is that knowing the amounts seems to be crucial for computing *approximate Gröbner bases*; see [10].

In [10], in order to remove the instability of computation due to self-reduction, the authors proposed to replace each small leading coefficient by an independent symbol and, in the case of large leading term, multiply a symbol to the terms other than the leading term. We call this method *symbolic coefficient method*. They remove fully erroneous terms by representing numeric coefficients by "effective floating-point numbers (*efloats*)"; we explain the efloat in Subsect. 4.2. The efloats work quite well. However, the symbolic coefficient method has two serious drawbacks: 1) it is very time-consuming because we must handle polynomials with symbolic coefficients, and 2) it cannot completely remove the errors due to self-reduction, because even a leading term of relative magnitude 0.3, say, may cause considerable errors.

In this paper, we propose a new method for avoiding the errors due to self-reduction. The new method does not introduce any symbol but it employs multiple precision effective floating-point numbers (*big-efloats*), hence the method is much more efficient than the symbolic coefficient method. In the new method, self-reduction is not avoided but we will show that it does not reduce the accuracy of the Gröbner basis computed. Furthermore, we propose a method to estimate the amount of intrinsic cancellation.

2 Instability Due to Self-reduction

First of all, we emphasize that **we compute Gröbner bases by successive eliminations of leading terms**. This is crucial in the following arguments.

By F, G, etc., we denote multivariate polynomials with coefficients of floating-point numbers. The norm of polynomial F is denoted by $\|F\|$; we employ the infinity norm, i.e., the maximum of the absolute values of numerical coefficients of F. For notions on Gröbner bases, we follow [4]. A power product is a term with no coefficient. By $\mathrm{lt}(F)$, $\mathrm{lc}(F)$ and $\mathrm{rt}(F)$ we denote the leading term, the leading coefficient and the reductum, respectively, of F, w.r.t. a term order $\succ$: $F = \mathrm{lt}(F) + \mathrm{rt}(F)$ with $\mathrm{lt}(F) \succ \mathrm{rt}(F)$. By $\mathrm{Spol}(F, G)$ and $\mathrm{Lred}(F, G)$ we denote the S-polynomial of F and G and the reduction of leading term of F by G, respectively. By reduction of F by G, we mean $\mathrm{Lred}(F, G)$. $\mathrm{Lred}(F, G)$ is often expressed as $F \xrightarrow{G} \tilde{F}$. By $F \xrightarrow{G}\!\!\!\!\rightarrow \tilde{F}$ we denote successive reductions of F by G so that $\mathrm{lt}(\tilde{F})$ is no more reducible by G.

We explain intrinsic cancellation by an example. In order to ease the reader to check our computation of examples, we construct examples by converting rational number coefficients into double precision floating-point numbers.

Example 1. Simple example which exhibits intrinsic cancellation.

$$\left\{\begin{array}{l} P_1 = 57/56\, x^2y + 68/67\, xz^2 - 79/78\, xy + 89/88\, x \\ P_2 = \qquad\qquad xyz^3 - \quad xy^2z + \quad xyz \\ P_3 = 56/57\, xy^2 - 67/68\, yz^2 + 78/79\, y^2 - 88/89\, y \end{array}\right\} \tag{2.1}$$

We convert P_1, P_2, P_3 into erroneous polynomials by converting their coefficients into double precision floating-point numbers. Then, we compute a Gröbner basis w.r.t. the total-degree order with $x \succ y \succ z$, using 30-digit floating-point numbers. We obtain the following unreduced Gröbner basis (correct figures are underlined).

$$\left\{\begin{array}{l} \qquad P_1,\ P_2,\ P_3 \ \text{ are unchanged,} \\ P_6 = y^2z^2 - \underline{2.99543694773255264453831970037}0\, xy^2 \\ \qquad\qquad - \underline{1.00207821651237482}57674951096740\, y^3 \\ \qquad\qquad + \underline{1.99832546917}37245140192885621560\, xy + \ \cdots\ , \\ P_7 = xz^2 - \underline{1.76431634237}04266614293919973\,20\mathrm{e}{-3}\, yz^2 \\ \qquad\qquad - \underline{9.94723245018}680541945733244338\,0\mathrm{e}{-1}\, xy \\ \qquad\qquad + \underline{1.76798297372}619363856479275314\,80\mathrm{e}{-3}\, y^2 + \ \cdots\ . \end{array}\right.$$

We see that some relative errors have been increased by about 10^4. □

2.1 Clones and Self-reduction Caused by Small Leading Terms

We use notation $F \approx G$ if $\|F - G\| \ll \|G\|$ and $\|F\| = O(\|G\|)$ if $\eta < \|F\|/\|G\| < 1/\eta$, where η is a positive number less than 1 but not much less than 1. (In our computer program, we set $\eta = 0.2$ and specify $\|G\| \ll \|F\|$ to be $\|G\| < 0.2\,\|F\|$.) We call a polynomial F *normal* if $|\mathrm{lc}(F)| = O(\|\mathrm{rt}(F)\|)$. We call a term T of F a *main term* if $\|T\| = O(\|F\|)$.

Definition 1 (clone). *Let R be either* $\mathrm{Spol}(F,G)$, $\mathrm{Lred}(F,G)$ *or* $F \xrightarrow{G}\!\!\!\!\rightarrow R$. *If* $R \approx M\,\mathrm{rt}(G)$, *with* M *a monomial, then* R *is called a* clone *of* G *and denoted by* $\mathrm{clone}(G)$. *Let* $\|F\| = \|G\| = 1$. *We call* $\|R\|/\|\mathrm{rt}(G)\|$ likeness *of the clone.*

Let F_1 and F_2 be normal polynomials and G be a polynomial with small leading term, $|\mathrm{lc}(G)| \ll \|G\|$. Suppose that F_1 and F_2 are reduced by G as

$$F_1 \xrightarrow{G}\!\!\!\!\rightarrow \tilde{F}_1, \quad F_2 \xrightarrow{G}\!\!\!\!\rightarrow \tilde{F}_2 \qquad (F_1 \neq \tilde{F}_1,\ F_2 \neq \tilde{F}_2)\,. \tag{2.2}$$

Then, so long as $|\mathrm{lc}(F_i)|/\|F_i\| \gg |\mathrm{lc}(G)|/\|G\|$ $(i=1,2)$, we usually have

$$\tilde{F}_1 \approx M_1\,\mathrm{rt}(G) \quad \text{and} \quad \tilde{F}_2 \approx M_2\,\mathrm{rt}(G)\,, \tag{2.3}$$

where M_1 and M_2 are the monomial multipliers in the last reductions, hence $\tilde{F}_1$ and $\tilde{F}_2$ are clones of G. We consider $\mathrm{Spol}(\tilde{F}_1,\tilde{F}_2)$; we do not consider $\mathrm{Lred}(\tilde{F}_1,\tilde{F}_2)$ or $\mathrm{Lred}(\tilde{F}_2,\tilde{F}_1)$, because $\mathrm{Spol}(\tilde{F}_1,\tilde{F}_2) = \mathrm{Lred}(\tilde{F}_1,\tilde{F}_2)$ if $\mathrm{lt}(\tilde{F}_2)\,|\,\mathrm{lt}(\tilde{F}_1)$ and $\mathrm{Spol}(\tilde{F}_1,\tilde{F}_2) = -\mathrm{Lred}(\tilde{F}_2,\tilde{F}_1)$ if $\mathrm{lt}(\tilde{F}_1)\,|\,\mathrm{lt}(\tilde{F}_2)$. Let $\mathrm{Spol}(\tilde{F}_1,\tilde{F}_2) = \tilde{M}_1\tilde{F}_1 - \tilde{M}_2\tilde{F}_2$, where $\tilde{M}_1$ and $\tilde{M}_2$ are monomials. Note that we may have $\mathrm{lt}(\tilde{F}_i) \succ M_i\,\mathrm{rt}(G)$ $(i \in \{1,2\})$. In order to avoid this case, we assume that

$$\mathrm{lt}(\tilde{F}_1) \approx \mathrm{lt}(M_1\,\mathrm{rt}(G)) \quad \text{and} \quad \mathrm{lt}(\tilde{F}_2) \approx \mathrm{lt}(M_2\,\mathrm{rt}(G))\,. \tag{2.4}$$

Under condition (2.4), we have $\mathrm{Spol}(\tilde{F}_1,\tilde{F}_2) \approx \tilde{M}_1M_1\,\mathrm{rt}(G) - \tilde{M}_2M_2\,\mathrm{rt}(G)$, hence we have $\|\mathrm{Spol}(\tilde{F}_1,\tilde{F}_2)\| \approx \|\tilde{M}_1M_1\,\mathrm{rt}(G) - \tilde{M}_2M_2\,\mathrm{rt}(G)\| \ll \|\tilde{M}_1M_1\,\mathrm{rt}(G)\|$. This means that all the main terms of $\tilde{M}_1M_1\,\mathrm{rt}(G)$ and $\tilde{M}_2M_2\,\mathrm{rt}(G)$ nearly cancel each other; the cancellation is exact if

$$\mathrm{lt}(\tilde{F}_1) = \mathrm{lt}(M_1\,\mathrm{rt}(G)) \quad \text{and} \quad \mathrm{lt}(\tilde{F}_2) = \mathrm{lt}(M_2\,\mathrm{rt}(G))\,. \tag{2.5}$$

Obviously, the above argument is valid for the case of $\tilde{F}_1 = \mathrm{Spol}(F_1,G)$ and/or $\tilde{F}_2 = \mathrm{Spol}(F_2,G)$. The above near cancellation of all the main terms in clones was called "self-reduction" in [10].

We must be careful in treating binomials with small leading terms. Let F_1 and F_2 be normal polynomials as given above, and let the reducer G be a binomial with small leading term: $G = g_1T_1 + g_2T_2$ with $|g_1| \ll |g_2|$, where T_1 and T_2 are power products. Then, $\mathrm{Lred}(F_1,G)$ becomes a polynomial with one large term, and so is $\mathrm{Lred}(F_2,G)$. Let $\tilde{F}_i = \mathrm{Lred}(F_i,G) \approx M_iT_2$ $(i=1,2)$, where M_i is a monomial. If $\mathrm{lt}(\tilde{F}_i) \approx M_iT_2$ $(i=1,2)$ then $\mathrm{Spol}(\tilde{F}_1,\tilde{F}_2)$ does not cause self-reduction. Self-reduction occurs only when $\mathrm{lt}(\tilde{F}_i) \succ M_iT_2$ $(i=1,2)$, $\mathrm{lt}(\tilde{F}_1)M_2 \approx \mathrm{lt}(\tilde{F}_2)M_1$ and $|\mathrm{lc}(\tilde{F}_1)|/\|\tilde{F}_1\| \approx |\mathrm{lc}(\tilde{F}_2)|/\|\tilde{F}_2\|$, which is unlikely to occur. We must notice, however, that G generates a polynomial with one large term. If the large term is the leading term then self-reduction may occur later, as we will explain below. Even if the large term is not the leading term, subsequent reductions may generate a polynomial with large leading term.

2.2 Self-reduction in Three Other Cases

Particularly large leading terms can also cause self-reductions, but the situation is pretty different. Let F_1 and F_2 be polynomials with large leading terms, and G be a normal polynomial:

$$|\mathrm{lc}(F_i)| \ll \|\mathrm{rt}(F_i)\| \ \ (i=1,2)\ , \qquad |\mathrm{lc}(G)| = O(\|\mathrm{rt}(G)\|)\ . \tag{2.6}$$

Then, we can express $\mathrm{Lred}(F_i, G)$ $(i=1,2)$ as follows:

$$\mathrm{Lred}(F_i, G) = F_i - \mathrm{lc}(F_i)/\mathrm{lc}(G)\cdot T_i G \approx -\mathrm{lc}(F_i)/\mathrm{lc}(G)\cdot T_i\,\mathrm{rt}(G)\ , \tag{2.7}$$

where T_1 and T_2 are power products. Therefore, $\mathrm{Lred}(F_i, G)$ is a clone of G, and self-reduction may occur in $\mathrm{Spol}(\mathrm{Lred}(F_1, G), \mathrm{Lred}(F_2, G))$. Note that self-reduction requires two polynomials with large leading terms. Therefore, self-reduction by polynomials with large leading terms is not frequent. Note further that the reduction of a polynomial F with a large leading term by a polynomial G with a small leading term generates a clone of very large likeness: the likeness is $(|\mathrm{lc}(F)|/\|\mathrm{rt}(F)\|)\cdot(\|G\|/|\mathrm{lc}(G)|)$.

Polynomial F may be reduced by $G_1, \ldots, G_m$ successively: $F \xrightarrow{G_1}\!\!\!\!\rightarrow \cdots \xrightarrow{G_m}\!\!\!\!\rightarrow \tilde{F}$. Here, $G_1, \ldots, G_m$ are polynomials with small leading terms and the reduction by each G_j $(1 \le j \le m)$ generates a clone(G_j). In this case, we call $\tilde{F}$ an m *multiple clone*, and represent it as clone$(G_1, \ldots, G_m)$.

We have a more complicated self-reduction which we call *paired self-reduction*. Let normal polynomials F_1 and F_2 be reduced, respectively, by G_1 and G_2 which are polynomials with small leading terms: $F_i \xrightarrow{G_i} \tilde{F}_i$ $(i=1,2)$. There may occur self-reduction in $\mathrm{Spol}(\tilde{F}_1, \tilde{F}_2)$, if F_1, F_2, G_1 and G_2 satisfy several conditions which are seldom satisfied. Because of the page limit, we omit the explanation of paired self-reduction.

Example 2. Simple system causing large errors (an example given in [10]).

$$\left\{\begin{array}{l} P_1 = x^3/10.0 + 3.0x^2y + 1.0y^2 \\ P_2 = 1.0x^2y^2 - 3.0xy^2 - 1.0xy \\ P_3 = y^3/10.0 + 2.0x^2 \end{array}\right\}$$

We compute a Gröbner basis w.r.t. the total-degree order with $x \succ y \succ z$, using double precision floating-point numbers, just as we compute a Gröbner basis over $\mathbb{Q}$. We show about two-thirds of the steps.

$$\begin{array}{lll}
1: & \mathrm{Spol}(P_3, P_2) \xrightarrow{P_1}\xrightarrow{P_1}\xrightarrow{P_2}\xrightarrow{P_3}\boxed{\xrightarrow{P_1}}\ \boxed{P_4} & /*\ P_4 = \mathrm{clone}(P_1) \\
2: & P_4 = x^2y + 29.8\cdots xy^2 + 3.33\cdots y^3 + 10.0xy + 0.333\cdots y^2 & \\
3: & P_2 \xrightarrow{P_4}\xrightarrow{P_3}\boxed{\xrightarrow{P_1}}\boxed{\xrightarrow{P_4}}\ \boxed{P_2'} & /*\ P_2' = \mathrm{clone}(P_1, P_4) \\
4: & P_2' = xy^2 + 0.111\cdots y^3 + 0.334\cdots xy - 0.000041\cdots y^2 & \\
5: & \mathrm{Spol}(P_3, P_2') \xrightarrow{P_3}\boxed{\xrightarrow{P_1}}\boxed{\xrightarrow{P_4}}\boxed{\xrightarrow{P_2'}}\xrightarrow{P_3} P_5 & /*\ \textbf{self-reduction} \\
6: & P_5 = x^2 + 7.14\cdots xy + 0.573\cdots y^2 & \\
7: & P_4 \xrightarrow{P_5}\xrightarrow{P_2'}\xrightarrow{P_3}\boxed{\xrightarrow{P_5}}\ \boxed{P_4'} & /*\ P_4' = \mathrm{clone}(P_5) \\
8: & P_4' = xy + 0.0844\cdots y^2 & \\
9: & P_2' \xrightarrow{P_4'}\xrightarrow{P_3}\boxed{\xrightarrow{P_5}}\boxed{\xrightarrow{P_4'}} P_2'' & /*\ \textbf{self-reduction}
\end{array}$$

Here, the polynomials boxed show clones and reducers which generate clones; the clones and self-reductions are commented in the right column. The above computation causes a very large cancellation: self-reductions in Steps 5 and 9 cause cancellations of $O(10^8)$ and $O(10^2)$, respectively. Other steps of computation cause almost no cancellation.

In Step 1, $\mathrm{Spol}(P_3, P_2)$ is a polynomial with large leading term and two reductions by P_1 give a clone of very large likeness, but it is erased by the subsequent reduction by P_2; P_3 is a binomial but the reduction by P_3 does not generate a polynomial with a large term, so we do not mind the reduction; the final reduction by P_1 gives a clone, i.e., $P_4 = \mathrm{clone}(P_1)$. In Step 3, the first reduction by P_4 gives a clone but the clone is erased by the subsequent reduction by P_3; the reduction by P_1 gives a clone, and the clone is reduced by P_4 having a small leading term, hence P_2' is a double clone. In Step 5, reductions by P_1 and P_4 give a double clone, and the double clone is reduced by another double clone P_2', hence there occurs self-reduction between double clones. □

We explain why such large cancellations occur in Example 2. P_2' in Step 3 is a double clone generated by successive reductions by P_1 and P_4, and so is the clone(P_1, P_4) obtained in Step 5. Following Theorem 1 in the next section, one may think that the amount of cancellation caused by self-reduction is $O((\|P_1\|/|\mathrm{lc}(P_1)|)(\|P_4\|/|\mathrm{lc}(P_4)|))$. Actually, we encounter a much larger cancellation. The reason for this superficial discrepancy is that, before the reduction by P_1, the polynomial concerned has been reduced by a binomial P_3 with a small leading term. Hence, $\mathrm{Lred}(\mathrm{Lred}(\mathrm{Lred}(\star, P_3), P_1), P_4)$ becomes a polynomial of very large likeness. The analysis in the next section shows that the actual amount of cancellations occurred is $O((\|P_1\|/|\mathrm{lc}(P_1)|)^2 (\|P_3\|/|\mathrm{lc}(P_3)|)^2)$. In fact, the symbolic coefficient computation in [10] shows this symbolically.

3 Analysis of Self-reductions Given in Sect. 2

In [10], we analyzed only the typical self-reduction by single clones. In this section, we analyze the self-reductions given in Sect. 2, in particular, self-reduction by multiple clones.

Following Collins [3], we introduce the concept of *associated polynomial*. Let polynomials P_i $(i = 1, \ldots, n)$ be expressed as $P_i = c_{i1}T_1 + \cdots + c_{im}T_m$, where $T_1, \ldots, T_m$ are power products, and let $M = (c_{ij})$ be an $n \times m$ matrix, where $n \le m$. The polynomial associated with M, which we denote by $\mathrm{assP}(M)$, is defined as follows.

$$\mathrm{assP}\begin{pmatrix} c_{11} & \cdots & c_{1n} & \cdots & c_{1m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} & \cdots & c_{nm} \end{pmatrix} \stackrel{\mathrm{def}}{=} \sum_{i=0}^{m-n} \begin{vmatrix} c_{11} & \cdots & c_{1,n-1} & c_{1,n+i} \\ \vdots & \ddots & \vdots & \vdots \\ c_{n1} & \cdots & c_{n,n-1} & c_{n,n+i} \end{vmatrix} T_{n+i}\,. \qquad (3.1)$$

3.1 Analysis of Self-reduction by Double Clones

Let polynomials F and F' be expressed as $F = f_1S_1 + f_2S_2 + \cdots + f_mS_m$ and $F' = f'_1S'_1 + f'_2S'_2 + \cdots + f'_mS'_m$, where S_i and S'_i $(i \geq 1)$ are power products satisfying $S_i \succ S_{i+1}$, $S_i = SS'_i$ for some power product S, and $f_1f'_1 \neq 0$ (some of f_j or f'_j $(j > 1)$ may be 0). Let polynomials G and G' be $G = g_1T_1 + g_2T_2 + \cdots + g_nT_n$ and $G' = g'_1T'_1 + g'_2T'_2 + \cdots + g'_nT'_n$, where T_i and T'_i $(i \geq 1)$ are power products satisfying $S_i = TT_i$ and $S'_i = T'T'_i$ for some power products T and T', and $g_1g'_1 \neq 0$ (some of g_j or g'_j $(j > 1)$ may be 0). We consider the case that both F and F' are reduced k times by G and then reduced k' times by G': $F \xrightarrow{G} \cdots \xrightarrow{G}\xrightarrow{G'} \cdots \xrightarrow{G'} \tilde{F}$ and $F' \xrightarrow{G} \cdots \xrightarrow{G}\xrightarrow{G'} \cdots \xrightarrow{G'} \tilde{F}'$, hence $\tilde{F}$ and $\tilde{F}'$ are double clones of G and G'. The next lemma is well known; we can easily prove it by mathematical inductions on k and k' (cf. [3]).

Lemma 1 (well known). *Let F, G and G' be defined as above. Suppose F is reduced k times by G then reduced k' times by G' (only the leading terms are reduced), then the resulting polynomial $\tilde{F}$ can be expressed as (we discard a numerical multiplier)*

$$\tilde{F} \;=\; \mathrm{assP}\begin{pmatrix} f_1 & f_2 & \cdots & f_n & f_{n+1} & \cdots & \cdots \\ g_1 & g_2 & \cdots & g_n & & & \\ & \ddots & \ddots & \cdots & \ddots & & \\ & & g'_1 & g'_2 & \cdots & g'_n & \\ & & & \ddots & \ddots & \cdots & \ddots \end{pmatrix}, \tag{3.2}$$

where the numbers of $(\cdots g_1 \cdots g_n \cdots)$-rows and $(\cdots g'_1 \cdots g'_n \cdots)$-rows are k and k', respectively. Here, polynomials F, G and G' are padded suitably by zero-coefficient terms so that the elements in each column of the matrix (3.2) correspond to the same term, as in (3.1).

Theorem 1. *Let F, F', $\tilde{F}$ and $\tilde{F}'$ be as above, and assume that $\mathrm{lt}(\tilde{F})/\mathrm{lc}(\tilde{F}) = S\,\mathrm{lt}(\tilde{F}')/\mathrm{lc}(\tilde{F}')$, with S a power product. Let $\tilde{F}$ and $\tilde{F}'$ be expressed as in* (3.2) *(for $\tilde{F}'$, we must replace the top row by $(f'_1\ f'_2 \cdots f'_n \cdots)$). Then, $\mathrm{lc}(\tilde{F}')\tilde{F} - \mathrm{lc}(\tilde{F})S\tilde{F}'$ can be factored as*

$$\begin{vmatrix} g_1 & \cdots & g_k & \cdots & g_{k+k'} \\ & \ddots & \ddots & \cdots & \vdots \\ & & g'_1 & \cdots & g'_{k'} \\ & & & \ddots & \vdots \\ & & & & g'_1 \end{vmatrix} \times \mathrm{assP}\begin{pmatrix} f_1 & f_2 & \cdots & f_n & f_{n+1} & \cdots & \cdots \\ f'_1 & f'_2 & \cdots & f'_n & f'_{n+1} & \cdots & \cdots \\ g_1 & g_2 & \cdots & g_n & & & \\ & \ddots & \ddots & \cdots & \ddots & & \\ & & g'_1 & g'_2 & \cdots & g'_n & \\ & & & \ddots & \ddots & \cdots & \ddots \end{pmatrix}, \tag{3.3}$$

where the numbers of $(\cdots\ g_1\ \cdots\ g_n\ \cdots)$-rows and $(\cdots\ g'_1\ \cdots\ g'_n\ \cdots)$-rows in the above matrix are k and k', respectively.

Proof . The coefficient of $S_{k+k'+i}$ term $(i \geq 2)$ in $\mathrm{lc}(\tilde{F}')\tilde{F} - \mathrm{lc}(\tilde{F})S\tilde{F}'$ is

$$\begin{vmatrix} f'_1 & \cdots & f'_k & \cdots & f'_{k+k'} & f'_{k+k'+1} \\ g_1 & \cdots & g_k & \cdots & g_{k+k'} & g_{k+k'+1} \\ & \ddots & \vdots & \ddots & \vdots & \\ & & g'_1 & \cdots & g'_{k'} & g'_{k'+1} \\ & & & \ddots & \vdots & \\ & & & & g'_1 & g'_{1+1} \end{vmatrix} \cdot \begin{vmatrix} f_1 & \cdots & f_k & \cdots & f_{k+k'} & f_{k+k'+i} \\ g_1 & \cdots & g_k & \cdots & g_{k+k'} & g_{k+k'+i} \\ & \ddots & \vdots & \ddots & \vdots & \\ & & g'_1 & \cdots & g'_{k'} & g'_{k'+i} \\ & & & \ddots & \vdots & \\ & & & & g'_1 & g'_{1+i} \end{vmatrix} \tag{3.4}$$

$$- \begin{vmatrix} f_1 & \cdots & f_k & \cdots & f_{k+k'} & f_{k+k'+1} \\ g_1 & \cdots & g_k & \cdots & g_{k+k'} & g_{k+k'+1} \\ & \ddots & \vdots & \ddots & \vdots & \\ & & g'_1 & \cdots & g'_{k'} & g'_{k'+1} \\ & & & \ddots & \vdots & \\ & & & & g'_1 & g'_{1+1} \end{vmatrix} \cdot \begin{vmatrix} f'_1 & \cdots & f'_k & \cdots & f'_{k+k'} & f'_{k+k'+i} \\ g_1 & \cdots & g_k & \cdots & g_{k+k'} & g_{k+k'+i} \\ & \ddots & \vdots & \ddots & \vdots & \\ & & g'_1 & \cdots & g'_{k'} & g'_{k'+i} \\ & & & \ddots & \vdots & \\ & & & & g'_1 & g'_{1+i} \end{vmatrix}.$$

The Sylvester identity allows us to factor the above expression as

$$\Rightarrow \begin{vmatrix} g_1 & \cdots & g_k & \cdots & g_{k+k'} \\ & \ddots & \vdots & \ddots & \vdots \\ & & g'_1 & \cdots & g'_{k'} \\ & & & \ddots & \vdots \\ & & & & g'_1 \end{vmatrix} \cdot \begin{vmatrix} f_1 & \cdots & f_k & \cdots & f_{k+k'} & f_{k+k'+1} & f_{k+k'+i} \\ f'_1 & \cdots & f'_k & \cdots & f'_{k+k'} & f'_{k+k'+1} & f'_{k+k'+i} \\ g_1 & \cdots & g_k & \cdots & g_{k+k'} & g_{k+k'+1} & g_{k+k'+i} \\ & \ddots & \vdots & \ddots & \vdots & \vdots & \\ & & g'_1 & \cdots & g'_{k'} & g'_{k'+1} & g'_{k'+i} \\ & & & \ddots & \vdots & \vdots & \\ & & & & g'_1 & g'_{1+1} & g'_{1+i} \end{vmatrix}. \tag{3.5}$$

This proves the theorem. □

Remark 1. Consider the case that F is reduced k_1 times by G then reduced k'_1 times by G' and F' is reduced k_2 times by G then reduced k'_2 times by G'. If $k_1 > k_2$, for example, then we put $k = k_2$ and treat the result of $k_1 - k_2$ reductions of F as a new F. If $k'_1 \neq k'_2$ then $\tilde{F}$ and $\tilde{F}'$ are not double clones but we must treat them as single clones of G'. □

Remark 2 . Theorem 1 can be generalized easily to the case of multiple clones: $F \xrightarrow{G_1} \cdots \xrightarrow{G_1} \cdots \xrightarrow{G_j} \cdots \xrightarrow{G_j} \tilde{F}$ and $F' \xrightarrow{G_1} \cdots \xrightarrow{G_1} \cdots \xrightarrow{G_j} \cdots \xrightarrow{G_j} \tilde{F}'$, where $\tilde{F} = \mathrm{clone}(G_1, \ldots, G_j)$ and $\tilde{F}' = \mathrm{clone}(G_1, \ldots, G_j)$. □

The above theorem is valid for any G and G', regardless of the magnitudes of leading terms of G and G'. The theorem tells us that term cancellations occur frequently: all the terms that are not proportional to $g_1^k {g'_1}^{k'}$ cancel one another. This cancellation does not cause large errors usually. If $|\mathrm{lc}(G)| \ll \|G\|$ and/or $|\mathrm{lc}(G')| \ll \|G'\|$, however, the term cancellation is the main-term cancellation and it causes large errors. Below, we order-estimate the amount of term cancellation occurring in $\mathrm{lc}(\tilde{F}')\tilde{F} - \mathrm{lc}(\tilde{F})S\tilde{F}'$ in a simple case.

3.2 Estimation of Amount of Main-Term Cancellation

By $\widetilde{D}_1, \widetilde{D}'_1, \widetilde{D}_i$ and $\widetilde{D}'_i$, we denote the determinants representing $\mathrm{lc}(\tilde{F})$, $\mathrm{lc}(\tilde{F}')$, the coefficient of $S_{k+k'+i}$ term of $\tilde{F}$, and the coefficient of $S_{k+k'+i}$ term of $S\tilde{F}'$, respectively, hence the first expression in the proof of Theorem 1 is $\widetilde{D}'_1\widetilde{D}_i - \widetilde{D}_1\widetilde{D}'_i$. Furthermore, by $\widetilde{D}_{1i}$, we denote the determinant of order $k+k'+2$ in the r.h.s. of (3.5). The magnitudes of $\widetilde{D}_1$ etc. change complicatedly as the situation changes, so we assume that the coefficients of F and F' are as follows.

$$f_1 = f'_1 = 1, \qquad f_i = 0 \text{ or } O(1), \quad f'_i = 0 \text{ or } O(1) \quad (i \geq 2). \tag{3.6}$$

Corollary 1. *Let the coefficients of F and F' be as in* (3.6)*. Let reducers G and G' be polynomials with coefficients such that*

$$\begin{aligned} &|g_1| \ll 1, \quad g_2 = \cdots = g_{l-1} = 0, \ |g_l| = O(1), \quad |g_{l+i}| = O(1) \text{ or } 0, \\ &|g'_1| \ll 1, \quad g'_2 = \cdots = g'_{l'-1} = 0, \ |g'_{l'}| = O(1), \quad |g'_{l'+i}| = O(1) \text{ or } 0. \end{aligned} \tag{3.7}$$

Claim 1: when $l = l' = 2$ (hence $g_2 = O(1)$ and $g'_2 = O(1)$), there occurs cancellation of amount $O((1/g_1)^k(1/g'_1)^{k'})$ in the computation of $\mathrm{lc}(\tilde{F}')\tilde{F} - \mathrm{lc}(\tilde{F})S\tilde{F}'$. Claim 2: when $l \geq 3$ and/or $l' \geq 3$ (hence $g_2 = 0$ and/or $g'_2 = 0$), let $|\widetilde{D}_1| = O((g_1)^{\kappa_1}(g'_1)^{\kappa'_1})$, $|\widetilde{D}_i| = O((g_1)^{\kappa_i}(g'_1)^{\kappa'_i})$ and $|\widetilde{D}_{1i}| = O((g_1)^{\tilde{\kappa}}(g'_1)^{\tilde{\kappa}'})$, then there occurs cancellation of amount $O((1/g_1)^{k-\kappa_1-\kappa_i+\tilde{\kappa}}(1/g'_1)^{k'-\kappa'_1-\kappa'_i+\tilde{\kappa}'})$ in the computation of $\mathrm{lc}(\tilde{F}')\tilde{F} - \mathrm{lc}(\tilde{F})S\tilde{F}'$.

Proof. When $l = l' = 2$, consider $\widetilde{D}_1$ for example, which is the determinant constructed from the leftmost $k+k'+1$ columns of matrix in (3.2). The product of diagonal elements gives the main term of $\widetilde{D}_1$, because other terms contain at least one g_1 or g'_1. Similarly, if we consider those $\widetilde{D}_{1i}$ for which $f'_{k+k'+i} \neq 0$, we see that $\widetilde{D}_{1i} = O(1)$. Then, determinants in (3.5) lead us to Claim 1. The determinants also lead us to Claim 2, because the main terms of $\widetilde{D}'_1\widetilde{D}$ and $\widetilde{D}_1\widetilde{D}'_i$ must be of the same order. □

Determination of $\tilde{\kappa}_1, \tilde{\kappa}'_1, \tilde{\kappa}_i, \tilde{\kappa}'_i, \tilde{\kappa}$ and $\tilde{\kappa}'$ in the general case of $l \geq 3$ and/or $l' \geq 3$ is messy. Because of the page limit, we omit the determination.

Theorem 1 allows us to analyze self-reduction caused by polynomials with large leading terms and paired self-reduction, too. For the case of large leading terms, we put $F_1 = F$, $F_2 = F'$ and $G' = G$, and assume that the leading terms of F and F' are large. Then, estimating the magnitudes of determinants in (3.5), we obtain the following corollary which can be easily generalized to the case that F_1 and/or F_2 contain several large terms at their heads.

Corollary 2. *Let F and F' be polynomials with large leading terms and G be a normal polynomial. Put $\tilde{F} = \mathrm{Lred}(F, G)$ and $\tilde{F}' = \mathrm{Lred}(F', G)$. Then, in the computation of $\mathrm{Spol}(\tilde{F}_1, \tilde{F}_2)$, there occurs main-term cancellation of magnitude $\min(|\mathrm{lc}(F)|/\|\mathrm{rt}(F)\|, |\mathrm{lc}(F')|/\|\mathrm{rt}(F')\|)$.*

4 New Method of Stabilization

We consider that the coefficients of input system of polynomials are inexact. If the largest relative error in the coefficients is ε then we say the *accuracy* of the system is ε. Below, by ϵ_{m} we denote the *machine epsilon* (= the difference between 1 and the smallest representable number greater than 1) of double precision floating-point numbers (double-floats). If the coefficients of input system are given by double-floats, we have $\varepsilon \geq \epsilon_{\mathrm{m}}$.

4.1 Supporting Theorem

We will compute the Gröbner basis by converting each input coefficient into a multiple precision floating-point number (big-float). We assume that each big-float is a p-digit decimal number satisfying $10^{-p} \ll \varepsilon$, and put $\epsilon_{\mathrm{M}} = 10^{-p}$.

Theorem 2. *As far as self-reductions treated in Sect. 3 are concerned, the main-term cancellation due to self-reduction ruins only tail figures of the coefficients concerned.*

Proof. We note that, although the big-floats in our case contain relative errors which are much larger than ϵ_{M}, *the errors are introduced initially and the coefficients are treated as definite numbers of the full precision throughout the computation*. On the other hand, Theorem 1 implies that, in the self-reductions considered, the coefficients of main terms cancel exactly within the precision. Hence the self-reductions ruin only tail figures of the coefficients. □

Remark 3. One may think that all the cancellation errors can be avoided if we increase the precision. This is, however, wrong as Example 1 shows. In the self-reductions we have considered, the main terms cancel exactly, which is the key of Theorem 2. □

4.2 Effective Floating-Point Numbers

In actual computation, we must remove the fully erroneous terms and estimate the amount of accuracy loss. Thus, we utilize *multiple precision effective floating-point numbers* (big-efloats), instead of big-floats.

We explain the efloats briefly. The efloat was proposed by the present authors in 1997 [7] so as to detect the cancellation errors automatically. The efloat is a pair of two floating-point numbers and expressed as $\#\mathrm{E}[f, e]$; we call f and e *value-part* and *error-part*, respectively. The arithmetic of efloats is as follows.

$$\begin{aligned}
\#\mathrm{E}[f_a, e_a] + \#\mathrm{E}[f_b, e_b] &\Longrightarrow \#\mathrm{E}[f_a + f_b,\ \max\{e_a, e_b\}],\\
\#\mathrm{E}[f_a, e_a] - \#\mathrm{E}[f_b, e_b] &\Longrightarrow \#\mathrm{E}[f_a - f_b,\ \max\{e_a, e_b\}],\\
\#\mathrm{E}[f_a, e_a] \times \#\mathrm{E}[f_b, e_b] &\Longrightarrow \#\mathrm{E}[f_a \times f_b,\ \max\{|f_b e_a|, |f_a e_b|\}],\\
\#\mathrm{E}[f_a, e_a] \div \#\mathrm{E}[f_b, e_b] &\Longrightarrow \#\mathrm{E}[f_a \div f_b,\ \max\{|e_a/f_b|, |f_a e_b/f_b^2|\}].
\end{aligned} \tag{4.1}$$

Thus, the value-part of efloat is nothing but the conventional floating-point value. On the other hand, the error-part of efloat represents the cancellation error approximately; the rounding errors are neglected in determining the error-part. Similarly, we neglect the rounding errors throughout the following arguments.

The big-efloat is expressed as $\#\mathrm{BE}[f, e]$, where f is a big-float, and it is processed by the same arithmetic as efloat. We set the error-part e to $10^{-p+2}\,|f|$.

We explain how the fully erroneous terms are removed (we explain only for the case of efloats). We set the error-part of each efloat coefficient to $5\varepsilon|f|$ (in the examples, we set to about $5\epsilon_{\mathrm{m}}|f|$). In our algebra system named GAL, the efloat $\#\mathrm{E}[f, e]$ with $|f| < e$ is automatically set to 0 (not $\#\mathrm{E}[0, 0]$). Therefore, GAL removes fully erroneous terms unless the rounding errors accumulate to $5\epsilon_{\mathrm{m}}$ or more, which is extremely rare in practice.

Example 3. Check Theorem 2 by the system in Example 2.

We convert the coefficients into double-floats, and compute a Gröbner basis with big-efloats of 30 decimal precision. For reference, we show the initial polynomials; if all the figures from 17th to the last decimal places are 0, our system outputs only one 0. Note that the rounding errors appear at the 17th decimal places.

$$\begin{cases} P_1 = + \#\mathrm{BE}[3.33333333333333310\mathrm{e}{-2},\ 2.0\mathrm{e}{-28}]\, x^3 + x^2 y \\ \qquad + \#\mathrm{BE}[3.33333333333333310\mathrm{e}{-1},\ 3.2\mathrm{e}{-27}]\, y^2, \\ P_2 = + \#\mathrm{BE}[3.33333333333333310\mathrm{e}{-1},\ 3.2\mathrm{e}{-27}]\, x^2 y^2 - x y^2, \\ \qquad - \#\mathrm{BE}[3.33333333333333310\mathrm{e}{-1},\ 3.2\mathrm{e}{-27}]\, xy \\ P_3 = + \#\mathrm{BE}[5.0000000000000000\mathrm{e}{-2},\ 3.9\mathrm{e}{-28}]\, y^3 + x^2. \end{cases}$$

The $\mathrm{Spol}(P_3, P_1)$, for example, is reduced and normalized as follows; we see that 17th to 30th figures of xy^3 term are contaminated by rounding errors.

$$x^4 + \#\mathrm{BE}[1.50000000000000001665334536937720\mathrm{e}{-1},\ 3.9\mathrm{e}{-28}]\, xy^3 \\ + \#\mathrm{BE}[5.0000000000000000\mathrm{e}{-2},\ 2.0\mathrm{e}{-28}]\, xy^2.$$

We obtain the following unreduced Gröbner base.

$$\begin{cases} P_2'' = y^2, \\ P_4' = xy + \#\mathrm{BE}[\underline{8.44022550452195}867628931165460 0\mathrm{e}{-2},\ 3.3\mathrm{e}{-21}]\, y^2, \\ P_5 = x^2 + \#\mathrm{BE}[\underline{7.14849689746270}7006365493318940,\ 4.2\mathrm{e}{-19}]\, xy \\ \qquad + \#\mathrm{BE}[\underline{5.73716139524645}7225742044589410\mathrm{e}{-1},\ 2.6\mathrm{e}{-20}]\, y^2. \end{cases}$$

Here, underlines show correct figures. We see that, although large cancellations have occurred, the accuracy loss in the Gröbner basis is only slight. □

4.3 Description of New Method

Now, we describe our new method which is based on Theorem 2 crucially. The method is composed of the following three devices.

Device 1: Convert the numeric coefficients of input polynomials into big-efloats of a suitably determined initial precision, say $p = 30$, and compute the Gröbner basis by using only the leading-term reduction and the S-polynomial construction.

Device 2: Monitor the error-parts of big-efloat coefficients during the computation, and if the amount of the largest cancellation accumulated, let it be C, becomes large satisfying $\epsilon_{\mathrm{M}} C > 10^{-5}\varepsilon$, say, then increase the precision of big-efloats and retry the computation.

Device 3: Monitor the clone generation of likeness greater than 5, say, and self-reduction by such clones. We explain the device for single reduction; for multiple reductions, see Sect. 5. Suppose that self-reduction occurs in the subtraction $\tilde{F}_1 - \tilde{F}_2$ in computing $\mathrm{Spol}(\mathrm{Lred}(F_1, G), \mathrm{Lred}(F_2, G))$ etc. Here, $\tilde{F}_1 = \mathrm{clone}(G)$ and $\tilde{F}_2 = \mathrm{clone}(G)$, hence $\tilde{F}_i = -c_i T \mathrm{rt}(G) + (\text{small-terms})$ $(i = 1, 2)$, where c_1 and c_2 are numbers such that $c_1 \approx c_2$ and T is a power product. Then, we "subtract" $-\mathrm{rt}(G)$ from both $\tilde{F}_1$ and $\tilde{F}_2$ as $\tilde{F}_1' := \tilde{F}_1 + cT\mathrm{rt}(G)$ and $\tilde{F}_2' := \tilde{F}_2 + cT\mathrm{rt}(G)$, where c will be determined in Subsect. 5.2 ($c \approx c_1 \approx c_2$). We call this operation *reducer subtraction*. Regard the possible cancellation occurring in $\tilde{F}_1' - \tilde{F}_2'$ as the intrinsic cancellation.

The number 5 in Device 3 is determined from the following reason: Example 2 shows that we must monitor clones of likeness 10 or more, and it is impractical to monitor clones of likeness 2 or less. With Devices 1 and 2, we can protect the accuracy of the system from self-reduction completely; the number 5 for specifying the clone in Device 3 is irrelevant to this protection. Device 3 is for estimating the intrinsic cancellation; we explain the details in Sect. 5.

As for the intrinsic cancellation, authors of [2] and [10] defined the cancellation in terms of syzygies. The computation of syzygies is quite costly in practice. On the other hand, the reducer subtraction is not a costly operation (see Sect. 5 for implementation), hence our method is practical.

5 Implementation Details

Although our ideas given above are simple, actual implementation of the Device 3 requires various detailed considerations.

5.1 Representation of Clones

In our current program, each input polynomial or S-polynomial generated is numbered uniquely, say F_i $(i \in \mathbb{N})$, and the numbering is not changed if the polynomial is reduced; if F_i is reduced to 0 then F_i is removed from the memory. Suppose a polynomial F_i is reduced by G_j to become a clone of G_j. It is not enough to save the index j to specify the clone; we must save the current G_j because G_j itself might change later during the computation. Let the reduction be $\tilde{F}_i := F_i - c_j T_j G_j$, where $c_j \in \mathbb{C}$ and T_j is a power product. The multiplier c_j changes from the reduction to reduction, hence we must save the multipliers, too. Therefore, we represent clones generated from G_j as follows.

1. Normalize G_j so that its leading coefficient is 1.
2. Represent each clone by a triplet $\langle j, c_j, T_j G_j \rangle$ which we call *clone-triplet*. Construct a clone-triplet each time F_i is reduced.

3. Save the clone-triplets for F_i into a list and attach the list to F_i. For example, if $F_i \xrightarrow{G_j} \xrightarrow{G_{j'}} \cdots$, then the list is $(\cdots \langle j', c'_{j'}, T'_{j'}G_{j'}\rangle \langle j, c_j, T_jG_j\rangle)$.

We normalize not only clones but also each polynomial appearing in the computation so that its leading coefficient is 1, which makes the programming easy. The normalization is made after each reduction (and S-polynomial generation): $\tilde{F}_i := F_i - c_jT_jG_j \longrightarrow \tilde{F}_i := \tilde{F}_i/\mathrm{lc}(\tilde{F}_i)$. Just after this normalization, all the multipliers in the clone-triplet list for F_i must be changed as $\langle j, c_j, T_jG_j\rangle \to \langle j, c_j/\mathrm{lc}(\tilde{F}_i), T_jG_j\rangle \ \ (j=1,2,\ldots)$.

5.2 Reducer Subtraction

The reducer subtraction is performed as follows. Let $\tilde{F}_i = F_i - c_iTG$ $(i = 1, 2)$, and suppose that self-reduction occurs in $\tilde{F}_1 - \tilde{F}_2$, as in Device 3 (self-reduction occurs actually in $\mathrm{Spol}(\tilde{F}_1, \tilde{F}_2)$ or $\mathrm{Lred}(\tilde{F}_1, \tilde{F}_2)$, but we simplified the situation by multiplying suitable power products to $\tilde{F}_1$ and $\tilde{F}_2$). By computing c as

$$c = \begin{cases} c_1 & \text{if } |c_1| \le |c_2|, \\ c_2 & \text{if } |c_1| > |c_2|, \end{cases} \tag{5.1}$$

we subtract $-\mathrm{rt}(G)$ from $\tilde{F}_i$ $(i = 1, 2)$ as $\tilde{F}_i := \tilde{F}_i + cT\mathrm{rt}(G)$.

The above subtraction is for the reducer used at the last reduction (the leftmost reducer in the clone-triplet list of F_i). For other reducers, the subtraction is made as follows; we explain only for the case that equalities in (2.5) hold. Suppose F_1 is reduced by $G_1, \ldots, G_j$ as $F_1 \xrightarrow{G_1} \cdots \xrightarrow{G_j} \tilde{F}_1$ and we have

$$\tilde{F}_1 := (\cdots(F_1 - c_{11}T_1G_1)\cdots) - c_{1j}T_jG_j.$$

In this case, the leading terms of $\mathrm{rt}(G_1)$ may be eliminated and $\tilde{F}_1$ may contain only $\mathrm{rt}(\cdots\mathrm{rt}(G_1)\cdots)$. Therefore, we scan terms of $\tilde{F}_1$ and $\mathrm{rt}(G_1)$ from highest to lowest order, and determine which $\mathrm{rt}(\cdots\mathrm{rt}(G_1)\cdots)$ is contained in $\tilde{F}_1$. Then, we subtract a suitable multiple of that $\mathrm{rt}(\cdots\mathrm{rt}(G_1)\cdots)$ from $\tilde{F}_1$ (and $\tilde{F}_2$).

5.3 Estimating the Amount of Intrinsic Cancellation

The actual term cancellation is the sum of the intrinsic cancellations and cancellations due to self-reductions. Therefore, if we remove all the cancellations due to self-reductions, then the rest must be the sum of intrinsic cancellations. It should be mentioned that Device 3 will fail to remove small amounts of cancellations due to small self-reductions, because we neglect the clones of likeness < 5. Therefore, the method explained in Device 3 will over-estimate the amount of intrinsic cancellation.

To illustrate our technique we show the estimation of the intrinsic cancellation in Example 2, in particular at the reduction step $\mathrm{Spol}(P_3, P'_2) \xrightarrow{P_3} \xrightarrow{P_1} \xrightarrow{P_4} \cdots$, where self-reduction by double clones occurs and we encounter main-term cancellation of $O(10^{10})$.

Example 4. Intrinsic cancellation in Step 5 of Example 2.

Put $Q_1 = \mathrm{Lred}(\mathrm{Lred}(\mathrm{Lred}(\mathrm{Spol}(P_3, P_2'), P_3), P_1), P_4)$ and let $\mathrm{Lred}(Q_1, P_2') = Q_1 - Q_2$, where $P_2' = \mathrm{clone}(P_1, P_4)$. Below, underlines show figures which are same in both Q_1 and Q_2 (or Q_1' and Q_2', or Q_1'' and Q_2'').

$$\begin{aligned} Q_1 = &+ \#\mathrm{BE}[\underline{1.1152418136}7893093558453405171\mathrm{e}{-1},\ 8.6\mathrm{e}{-28}]\, y^3 \\ &+ \#\mathrm{BE}[\underline{3.345725}3711642806415804801040\mathrm{e}{-1},\ 3.7\mathrm{e}{-27}]\, xy \\ &- \#\mathrm{BE}[\underline{4.1613}5062896641687828404499 50\mathrm{e}{-5},\ 1.1\mathrm{e}{-28}]\, y^2, \\ Q_2 = &+ \#\mathrm{BE}[\underline{1.1152418132}0025535431698179200\mathrm{e}{-1},\ 8.6\mathrm{e}{-28}]\, y^3 \\ &+ \#\mathrm{BE}[\underline{3.345725}4396007606295094537600\mathrm{e}{-1},\ 3.7\mathrm{e}{-27}]\, xy \\ &- \#\mathrm{BE}[\underline{4.1612}9570397496130897476309 08\mathrm{e}{-5},\ 1.1\mathrm{e}{-28}]\, y^2. \end{aligned}$$

A multiple of $-\mathrm{rt}(P_4)$ is subtracted from Q_1 and Q_2; $Q_1' \leftarrow Q_1$ and $Q_2' \leftarrow Q_2$:

$$\begin{aligned} Q_1' = &+ \#\mathrm{BE}[\underline{2.3}2908374086518471491083791 54\mathrm{e}{-9},\ 8.6\mathrm{e}{-28}]\, y^3 \\ &- \#\mathrm{BE}[\underline{1.1194031}4101705996407177741 30\mathrm{e}{-2},\ 1.1\mathrm{e}{-28}]\, y^2, \\ Q_2' = &+ \#\mathrm{BE}[\underline{2.2}8121599959763245520130227 54\mathrm{e}{-9},\ 8.6\mathrm{e}{-28}]\, y^3 \\ &+ \#\mathrm{BE}[6.8436479987928973656039068264\mathrm{e}{-9},\ 3.7\mathrm{e}{-27}]\, xy \\ &+ \#\mathrm{BE}[\underline{1.1194030}8609206850850246813 10\mathrm{e}{-2},\ -1.1\mathrm{e}{-28}]\, y^2. \end{aligned}$$

A multiple of $-\mathrm{rt}(P_1)$ is subtracted from Q_1' and Q_2'; $Q_1'' \leftarrow Q_1'$ and $Q_2'' \leftarrow Q_2'$:

$$\begin{aligned} Q_1'' = &+ \#\mathrm{BE}[\underline{2.3}2908374086518471491083791 54\mathrm{e}{-9},\ 8.6\mathrm{e}{-28}]\, y^3, \\ Q_2'' = &+ \#\mathrm{BE}[\underline{2.2}8121599959763245520130227 54\mathrm{e}{-9},\ 8.6\mathrm{e}{-28}]\, y^3 \\ &+ \#\mathrm{BE}[6.8436479987928973656039068264\mathrm{e}{-9},\ 3.7\mathrm{e}{-27}]\, xy \\ &+ \#\mathrm{BE}[5.4924991455569309281904242148\mathrm{e}{-10},\ 1.1\mathrm{e}{-28}]\, y^2. \end{aligned}$$

We see $O(10^2)$ cancellation occurs in $Q_1'' - Q_2''$ which we regard as the intrinsic cancellation. □

6 Concluding Remarks

For the page limit of LNCS, several parts were omitted in this paper. See [1] for the omitted parts.

We showed that, restricting the reductions to leading-term reductions, we are able to describe local steps of Gröbner basis computation by matrices and analyze self-reduction and intrinsic cancellation in terms of determinants (Theorem 1). Furthermore, we showed that the main-term cancellation due to self-reduction causes no problem if we utilize big-efloats, as far as the self-reductions investigated in Sect. 2 are concerned (Theorem 2). We are now trying to prove that any self-reduction causes no problem.

Our analysis suggests us that the cancellation errors will be decreased largely if self-reduction is avoided as far as possible. We are now developing a program package based on this suggestion.

Finally, the authors acknowledge anonymous referees for valuable comments.

References

1. Sasaki, T., Kako, F.: Floating-point Gröber Basis Computation with Ill-conditionedness Estimation. Technical Report of Univ. of Tsukuba, in (December 2007), http://www.math.tsukuba.ac.jp/~sasaki/papers/ASCM2007
2. Bodrato, M., Zanoni, A.: Intervals, syzygies, numerical Gröbner bases: a mixed study. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2006. LNCS, vol. 4194, pp. 64–76. Springer, Heidelberg (2006)
3. Collins, J.E.: Subresultant and reduced polynomial remainder sequence. J. ACM 14, 128–142 (1967)
4. Cox, D., Little, J., O'Shea, D.: Ideals, Varieties, and Algorithms. Springer, New York (1997)
5. Fortuna, E., Gianni, P., Trager, B.: Degree reduction under specialization. J. Pure Appl. Algebra 164, 153–164 (2001)
6. Gonzalez-Vega, L., Traverso, C., Zanoni, A.: Hilbert stratification and parametric Gröber bases. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2005. LNCS, vol. 3718, pp. 220–235. Springer, Heidelberg (2005)
7. Kako, F., Sasaki, T.: Proposal of "effective" floating-point number. Preprint of Univ. Tsukuba (May 1997) (unpublished)
8. Kondratyev, A., Stetter, H.J., Winkler, S.: Numerical computation of Gröbner bases. In: Proceedings of CASC 2004 (Computer Algebra in Scientific Computing), St. Petersburg, Russia, pp. 295–306 (2004)
9. Mourrain, B.: Pythagore's dilemma, symbolic-numeric computation, and the border basis method. In: Symbolic-Numeric Computations (Trends in Mathematics), pp. 223–243. Birkhäuser Verlag, Basel (2007)
10. Sasaki, T., Kako, F.: Computing floating-point Gröbner base stably. In: Proceedings of SNC 2007 (Symbolic Numeric Computation), London, Canada, pp. 180–189 (2007)
11. Shirayanagi, K.: An algorithm to compute floating-point Gröbner bases. In: Mathematical Computation with Maple V. Ideas and Applications, pp. 95–106. Birkhäuser, Basel (1993)
12. Shirayanagi, K.: Floating point Gröbner bases. Mathematics and Computers in Simulation 42, 509–528 (1996)
13. Shirayanagi, K., Sweedler, M.: Remarks on automatic algorithm stabilization. J. Symb. Comput. 26, 761–765 (1998)
14. Stetter, H.J.: Stabilization of polynomial systems solving with Gröbner bases. In: Proceedings of ISSAC 1997 (Intern'l Symposium on Symbolic and Algebraic Computation), pp. 117–124. ACM Press, New York (1997)
15. Stetter, H.J.: Numerical Polynomial Algebra. SIAM Publ., Philadelphia (2004)
16. Stetter, H.J.: Approximate Gröbner bases – an impossible concept? In: Proceedings of SNC 2005 (Symbolic-Numeric Computation), Xi'an, China, pp. 235–236 (2005)
17. Traverso, C.: Syzygies, and the stabilization of numerical Buchberger algorithm. In: Proceedings of LMCS 2002 (Logic, Mathematics and Computer Science), RISC-Linz, Austria, pp. 244–255 (2002)
18. Traverso, C., Zanoni, A.: Numerical stability and stabilization of Gröbner basis computation. In: Proceedings of ISSAC 2002 (Intern'l Symposium on Symbolic and Algebraic Computation), pp. 262–269. ACM Press, New York (2002)
19. Weispfenning, V.: Gröbner bases for inexact input data. In: Proceedings of CASC 2003 (Computer Algebra in Scientific Computing), Passau, Germany, pp. 403–411 (2003)

The Maximality of the Dixon Matrix on Corner-Cut Monomial Supports

Eng-Wee Chionh

School of Computing, National University of Singapore
Computing 1, #03-68, Law Link, Singapore 117590
chionhew@comp.nus.edu.sg
http://www.comp.nus.edu.sg/~chionhew

Abstract. It has been established that the bivariate Dixon matrix persists to be the exact resultant when there are at most two exposed points at each corner of a corner-cut support; but it becomes singular when there are four or more exposed points at any of the corners. For the remaining case of three or fewer exposed points at each of the corners, it is observed that the Dixon matrix is maximal but its determinant is a multiple of the resultant with a priori known bracket powers as the extraneous factors. The maximality of the Dixon matrix for the three-or-fewer exposed points case has been established mechanically for the special situation in which the excess degree is unity when there are three exposed points at a corner. This paper presents a greatly simplified mechanical proof so that its validity can be easily verified.

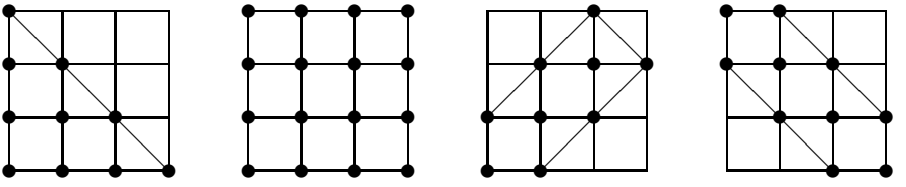
Keywords: Dixon matrix, corner-cut monomial supports, maximality, mechanical proof.

1 Introduction

The resultant is a remarkable tool for eliminating variables from a system of polynomial equations [11]. It distinguishes itself among the elimination methods such as Groebner bases and Wu-Ritt triangular systems in two important aspects: availability of an explicit compact formula and computational efficiency. The geometry of the monomial support of the system of polynomial equations determines the construction of the resultant. The Macaulay quotient resultant works for total degree polynomial systems with a triangular monomial support. The Dixon determinant applies for multi-degree polynomial systems with a rectangular monomial support. In computer geometry modeling and processing, resultants offer themselves as a natural means for converting the parametric shape representation to the implicit shape representation, a process known as implicitization. In the context of implicitization, the role of support geometry is all the more conspicuous — the Macaulay quotient for triangular patches and the Dixon determinant for rectangular patches. However, for multi-sided toric patches [9], both the Macaulay quotient and the Dixon determinant usually fail because in general toric patches are built on corner-cut monomial supports (rectangular supports with some monomial points removed around the support corners). These

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 293–306, 2008.
© Springer-Verlag Berlin Heidelberg 2008

missing monomial points introduce base points that are not accounted for by the Dixon construction. But fortunately the Dixon method adapts to fit corner-cut monomial supports if the cutting is not too severe. The cutting severity can be quantified by counting what is called exposed points of the corner-cut monomial support at each of its four corners. The following diagrams show respectively a total degree 3 triangular support, a bicubic 3×3 rectangular support, a corner-cut 3×3 pentagonal support with 1, 3, 2, 3 exposed points at the corners, and a corner-cut 3×3 hexagonal support with 3, 1, 3, 1 exposed points at the corners.



Currently the findings and observations of the effect of exposed points are as follows.

1. If the number of exposed points is at most two at each of the corners of the monomial support, the Dixon determinant is the exact resultant [2].
2. If the number of exposed points is four or more at any of the corners of the monomial support, the Dixon matrix is singular so its determinant vanishes [12].
3. If the number of exposed points is at most three at any of the corners of the bidegree $m \times n$ monomial support, it is conjectured [6] that the resultant is

Conjecture 1

$$\frac{|D|}{B_{00}^{\epsilon_{00}} B_{m0}^{\epsilon_{m0}} B_{mn}^{\epsilon_{mn}} B_{0n}^{\epsilon_{0n}}}, \tag{1}$$

where $B_{\mu\nu}$ is the bracket corresponding to the three exposed points at corner $(\mu, \nu) \in \{(0,0), (m,0), (m,n), (0,n)\}$, otherwise it is unity; $\epsilon_{\mu\nu}$ is the excess degree at corner (μ, ν).

The conjecture is proved partially [7,8] — it is true when the excess degree $\epsilon_{\mu\nu}$ is unity at corner (μ, ν). A crucial step in these proofs is establishing the maximality of the Dixon matrix. The proofs given in [7,8] are mechanical. A Maple program [1] is run to verify that certain rows or columns are independent under various at-most-three exposed point configurations. The contribution of this paper is to replace the complicated exhaustive verification mechanical proof with a greatly simplified mechanical proof which can be easily verified analytically. The simplification is made possible with the use of some very simple entry formulas for rows and columns of the Dixon matrix corresponding to exposed points in the monomial support.

The rest of the paper develops as follows. Section 2 discusses monomial support structures and defines exterior points and exposed points. Section 3 reviews the Dixon construction in the setting of monomial support. Section 4 presents the simplification effects of exterior and exposed points on the Dixon matrix.

Section 5 explains excess degrees and their role in formulating quotient Dixon resultants. Section 6 gives the maximality proof for corner-cut monomial supports with at most three exposed points at each of the four corners. Section 7 concludes the paper with some discussions.

2 The Structure of Cornet-Cut Monomial Supports

This section defines rectangular lattice sets, monomial supports and their bidegree hulls, exterior points, and exposed points of a monomial support.

2.1 Rectangular and Arbitrary Lattice Sets

Let $\mathbf{Z}$ be the set of integers and $\mathbf{R}$ be the set of reals. The set of lattice points is $\mathbf{Z} \times \mathbf{Z}$, a subset of the Euclidean plane $\mathbf{R} \times \mathbf{R}$.

It is very convenient to denote a rectangular lattice set whose sides are parallel to the axes with any pair of diagonally opposite vertices:

$$\{(x, y) \in \mathbf{Z} \times \mathbf{Z} : i \le x \le k, j \le y \le l\} = (i, j)..(k, l) = (k, l)..(i, j) \quad (2)$$
$$= (i, l)..(k, j) = (k, j)..(i, l) \quad (3)$$

where (i, j) is the bottom left corner and (k, l) is the top right corner. In particular, the rectangular lattice set $(0, 0)..(m, n)$ is denoted S_{mn}; that is,

$$S_{mn} = (0, 0)..(m, n) = (m, n)..(0, 0) = (0, n)..(m, 0) = (m, 0)..(0, n). \quad (4)$$

We denote the set of four corners of S_{mn} as

$$K_{mn} = \{(0, 0), (m, 0), (m, n), (0, n)\}. \quad (5)$$

2.2 Bidegree Hulls

For a lattice set $S \subseteq \mathbf{Z} \times \mathbf{Z}$, the bidegree hull of S is $(x_0, y_0)..(x_1, y_1)$ in which

$$x_0 = \min\{x : (x, y) \in S\}, \qquad y_0 = \min\{y : (x, y) \in S\}, \quad (6)$$
$$x_1 = \max\{x : (x, y) \in S\}, \qquad y_1 = \max\{y : (x, y) \in S\}. \quad (7)$$

For the rest of the paper, we shall write

$$S \sqsubseteq S_{mn} \quad (8)$$

to emphasize that S_{mn} is the bidegree hull of the lattice set S.

2.3 Exterior Points and Exposed Points

Consider a lattice set $S \sqsubseteq S_{mn}$. A point $(x, y) \in S_{mn} \setminus S$ is an exterior point of S with respect to the corner $(\mu, \nu) \in K_{mn}$ if

$$((x, y)..(\mu, \nu)) \cap S = \emptyset. \quad (9)$$

A point $(x, y) \in S$ is an exposed point of S with respect to the corner (μ, ν) if

$$((x, y)..(\mu, \nu)) \cap S = \{(x, y)\}. \tag{10}$$

The sets of exterior and exposed points of $S \sqsubseteq S_{mn}$ with respect to the corner $(\mu, \nu) \in K_{mn}$ will be denoted as $E_{\mu\nu}$ and $X_{\mu\nu}$ respectively. Clearly, we have

$$X_{00} \cup X_{m0} \cup X_{mn} \cup X_{0n} \subseteq S \subseteq S_{mn} \setminus (E_{00} \cup E_{m0} \cup E_{mn} \cup E_{0n}). \tag{11}$$

The structure of S with respect to its bidegree hull S_{mn} is also discussed in [3].

2.4 Presentation Convention

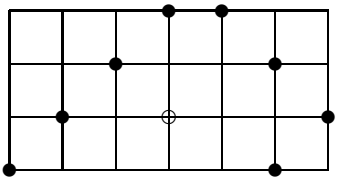
In illustrations exposed points will be drawn as "•" and points of S that are not exposed will be drawn as "○". For ease of association to the corners, the sets of exterior and exposed points of a support will be presented respectively in the form

$$\begin{array}{|c|c|} \hline E_{0n} & E_{mn} \\ \hline E_{00} & E_{m0} \\ \hline \end{array}, \quad \begin{array}{|c|c|} \hline X_{0n} & X_{mn} \\ \hline X_{00} & X_{m0} \\ \hline \end{array}. \tag{12}$$

2.5 An Example

For example, consider the lattice set

$$S = \{(0,0), (5,0), (1,1), (3,1), (6,1), (2,2), (5,2), (3,3), (4,3)\} \sqsubseteq S_{63}. \tag{13}$$



The sets of exterior points are

$$\begin{array}{|c|c|} \hline \{(0,1), (0,2), (1,2), (0,3), (1,3), (2,3)\} & \{(6,2), (5,3), (6,3)\} \\ \hline \emptyset & \{(6,0)\} \\ \hline \end{array}. \tag{14}$$

The sets of exposed points are

$$\begin{array}{|c|c|} \hline \{(0,0), (1,1), (2,2), (3,3)\} & \{(6,1), (5,2), (4,3)\} \\ \hline \{(0,0)\} & \{(5,0), (6,1)\} \\ \hline \end{array}. \tag{15}$$

3 The Dixon Construction

The original Dixon construction [5] is meant for rectangular supports S_{mn}. But it is straightforward to adapt it to any lattice set $S \sqsubseteq S_{mn}$.

3.1 The Dixon Polynomial $P(S)$ for a Support S

Given a lattice set $S \sqsubseteq S_{mn}$, the Dixon polynomial for S is

$$P(S) = \frac{\begin{vmatrix} f(s,t) & g(s,t) & h(s,t) \\ f(\alpha,t) & g(\alpha,t) & h(\alpha,t) \\ f(\alpha,\beta) & g(\alpha,\beta) & h(\alpha,\beta) \end{vmatrix}}{(s-\alpha)(t-\beta)} \tag{16}$$

$$= \frac{\sum_{(i,j),(k,l),(p,q)\in S}(i,j,k,l,p,q)s^i t^{j+l}\alpha^{k+p}\beta^q}{(s-\alpha)(t-\beta)} \tag{17}$$

where f, g, h are polynomials defined on S:

$$f(s,t) = \sum_{(i,j)\in S} f_{ij}s^i t^j, \quad g(s,t) = \sum_{(i,j)\in S} g_{ij}s^i t^j, \quad h(s,t) = \sum_{(i,j)\in S} h_{ij}s^i t^j; \tag{18}$$

and the 3×3 determinant (i,j,k,l,p,q) consisting of coefficients of f, g, h, is called a bracket:

$$(i,j,k,l,p,q) = \begin{vmatrix} f_{ij} & g_{ij} & h_{ij} \\ f_{kl} & g_{kl} & h_{kl} \\ f_{pq} & g_{pq} & h_{pq} \end{vmatrix}. \tag{19}$$

When there is no risk of confusion we further abbreviate a bracket as

$$ijklpq = (i,j,k,l,p,q). \tag{20}$$

3.2 The Dixon Matrix $D(S)$ for a Support S

The Dixon matrix for S is the coefficient matrix $D(S)$ of $P(S)$ with row indices comprising monomials in s, t and column indices comprising monomials in α, β:

$$P(S) = \begin{bmatrix} \cdots & s^\sigma t^\tau & \cdots \end{bmatrix} D(S) \begin{bmatrix} \cdots & \alpha^a \beta^t & \cdots \end{bmatrix}^T \tag{21}$$

3.3 The Row Support $R(S)$ and Column Support $C(S)$ for the Dixon Matrix $D(S)$

The set of exponents (σ, τ) in the row indices, $R(S)$, is called the row support of $D(S)$ and the set of exponents (a, b) in the column indices, $C(S)$, is called the column support of $D(S)$. In general, we have

$$R(S_{mn}) = S_{m-1,2n-1}, \tag{22}$$
$$C(S_{mn}) = S_{2m-1,n-1}. \tag{23}$$

3.4 An Example

For example, we have

$$P(S_{11}) = \begin{bmatrix} 1 & t \end{bmatrix} \begin{bmatrix} 001001 & 001011 \\ 001101 & 011011 \end{bmatrix} \begin{bmatrix} 1 & \alpha \end{bmatrix}^T. \tag{24}$$

Furthermore, the row and column supports are

$$R(S_{11}) = S_{01}, \tag{25}$$
$$C(S_{11}) = S_{10}. \tag{26}$$

4 Exterior Points Simplify the Dixon Matrix

When exterior points of a support $S \sqsubseteq S_{mn}$ exist, $D(S)$ is obtained from $D(S_{mn})$ by removing some of its rows and columns and reducing some brackets of its entries to zero. The precise simplification effects of exterior points are described in this section.

4.1 Translating Points of S to Points of $R(S)$ and $C(S)$

First we introduce the notations $(x, y)'_{\mu\nu}$ and $(x, y)''_{\mu\nu}$ concerning translating a point (x, y) with respect to a corner $(\mu, \nu) \in K_{mn}$. Let

$(x, y)'_{0n} = (x, y + n - 1)$	$(x, y)'_{mn} = (x - 1, y + n - 1)$
$(x, y)'_{00} = (x, y)$	$(x, y)'_{m0} = (x - 1, y)$

, (27)

and

$(x, y)''_{0n} = (x, y - 1)$	$(x, y)''_{mn} = (x + m - 1, y - 1)$
$(x, y)''_{00} = (x, y)$	$(x, y)''_{m0} = (x + m - 1, y)$

. (28)

A set $A \subseteq \mathbf{Z} \times \mathbf{Z}$ is translated similarly element-wise with the corner subscript and $'$, $''$ notations:

$$A'_{\mu\nu} = \{(x, y)'_{\mu\nu} : (x, y) \in A\}, \quad A''_{\mu\nu} = \{(x, y)''_{\mu\nu} : (x, y) \in A\}. \tag{29}$$

4.2 Exterior Points Eliminate Rows and Columns

The following theorem [12] says that exterior points simplifies the Dixon matrix by eliminating rows and columns corresponding to exterior points.

Theorem 1. *Let $E_{\mu\nu}$, $(\mu,\nu) \in K_{mn}$, be the sets of exterior points of the support $S \sqsubseteq S_{mn}$. Then*

$$R(S) = S_{m-1,2n-1} \setminus (E'_{00} \cup E'_{m0} \cup E'_{mn} \cup E'_{0n}) \tag{30}$$

$$C(S) = S_{2m-1,n-1} \setminus (E''_{00} \cup E''_{m0} \cup E''_{mn} \cup E''_{0n}) \tag{31}$$

Furthermore,

$$|R(S)| = |C(S)| = 2mn - |E_{00}| - |E_{m0}| - |E_{mn}| - |E_{0n}|. \tag{32}$$

4.3 Exterior Points Eliminate Brackets

The following theorem [12,13] gives explicit formulas for the entries of rows and columns of $D(S)$ corresponding to exposed points.

Theorem 2. *Let $X_{\mu\nu}$, $(\mu,\nu) \in K_{mn}$, be the sets of exposed points of the support $S \sqsubseteq S_{mn}$. Let the Dixon polynomial on S be*

$$P(S) = \sum_{(\sigma,\tau)\in R(S),(a,b)\in C(S)} \Delta_{\sigma,\tau,a,b} s^{\sigma} t^{\tau} \alpha^{a} \beta^{b} \tag{33}$$

and $(x,y) \in X_{\mu\nu}$.

For $(x',y') = (x,y)'_{\mu\nu}$, the entry $\Delta_{x',y',a,b}$, for any $(a,b) \in C(S)$ is respectively

$$\begin{array}{|c|c|}\hline -\sum_{k=0}^{m}(x,y,k,n,a+1-k,b) & \sum_{k=0}^{m}(x,y,k,n,a-k,b) \\ \hline \sum_{k=0}^{m}(x,y,k,0,a+1-k,b+1) & -\sum_{k=0}^{m}(x,y,k,0,a-k,b+1) \\ \hline \end{array} \tag{34}$$

For $(x'',y'') = (x,y)''_{\mu\nu}$, the entry $\Delta_{\sigma,\tau,x'',y''}$, for any $(\sigma,\tau) \in R(S)$ is respectively

$$\begin{array}{|c|c|}\hline -\sum_{l=0}^{n}(\sigma+1,\tau-l,0,l,x,y) & \sum_{l=0}^{n}(\sigma,\tau-l,m,l,x,y) \\ \hline \sum_{l=0}^{n}(\sigma+1,\tau+1-l,0,l,x,y) & -\sum_{l=0}^{n}(\sigma,\tau+1-l,m,l,x,y) \\ \hline \end{array} \tag{35}$$

4.4 Examples

The Dixon matrix $D(S_{22})$ is an 8×8 matrix containing a total of 144 brackets. Let

$$S_1 = S_{22} \setminus \{(0,0),(0,2),(2,0),(2,2)\}. \tag{36}$$

The Dixon matrix $D(S_1)$ is a 4×4 matrix containing a total of 12 brackets. Let

$$S_2 = S_1 \setminus \{(1,1)\}. \tag{37}$$

The Dixon matrix $D(S_2)$ is a 4×4 matrix containing a total of 8 brackets. Note that removing the point $(1,1)$ from S_1 does not decimate rows nor columns, it only reduces the number of brackets.

5 Excess Degrees and Quotient Dixon Resultants

The theories of sparse resultants and BKK degree bound [4] tell that the degree d of the sparse resultant in the coefficients of each of the polynomials f, g, h defined on S is $\mathrm{NV}(\widehat{S})$, where $\widehat{S}$ is the convex hull of S and $\mathrm{NV}(X)$ is the normalized volume (twice the Euclidean area) of the set X. Thus we have

$$d = 2mn - \sum_{(\mu,\nu)\in K_{mn}} \delta_{\mu\nu} \tag{38}$$

where $\delta_{\mu\nu}$ is the normalized volume chipped off from S_{mn} by the boundary of $\widehat{S}$ at corner $(\mu,\nu) \in K_{mn}$. But

$$\dim(D(S)) = 2mn - \sum_{(\mu,\nu)\in K_{mn}} |E_{\mu\nu}|. \tag{39}$$

Consequently,

$$d = \dim(D(S)) - \sum_{(\mu,\nu)\in K_{mn}} (\delta_{\mu\nu} - |E_{\mu\nu}|), \tag{40}$$

$$= \dim(D(S)) - \sum_{(\mu,\nu)\in K_{mn}} \epsilon_{\mu\nu}, \tag{41}$$

where the difference

$$\epsilon_{\mu\nu} = \delta_{\mu\nu} - |E_{\mu\nu}| \tag{42}$$

is called the excess degree at corner (μ,ν).

The following theorems state the value of $\epsilon_{\mu\nu}$ for $|X_{\mu\nu}| \le 2$ and $|X_{\mu\nu}| = 3$. The proof can be found in [2] and [12] respectively.

Theorem 3. *For any corner $(\mu,\nu) \in K_{mn}$, if $|X_{\mu\nu}| \le 2$, then $\epsilon_{\mu\nu} = 0$.*

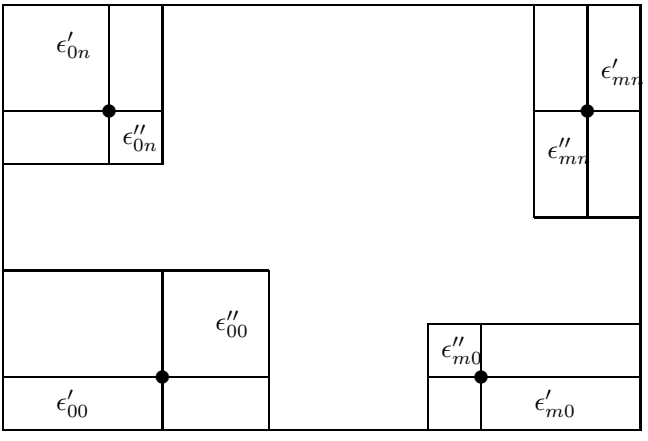
Theorem 4. *Let the three exposed points at corner $(\mu, \nu) \in K_{mn}$ be respectively*

$$\begin{array}{|c|c|}\hline (0,y),(e,f),(x,n) & (m,y),(e,f),(x,n) \\ \hline (0,y),(e,f),(x,0) & (m,y),(e,f),(x,0) \\ \hline \end{array}. \tag{43}$$

The excess degree at corner $(\mu, \nu) \in K_{mn}$ is

$$\epsilon_{\mu\nu} = \frac{\min(NV((\mu,\nu)..(e,f)), NV((e,f)..(x,y)))}{2}. \tag{44}$$

The following diagram shows the possible (e, f) positions and the attending rectangular sets $\epsilon'_{\mu\nu} = (\mu, \nu)..(e, f)$, $\epsilon''_{\mu\nu} = (e, f)..(x, y)$ at the four corners.



Note that for any $(\mu, \nu) \in K_{mn}$, $|X_{\mu\nu}| = 1$ when $(x, y) = (\mu, \nu)$ and $|X_{\mu\nu}| = 2$ when $(x, y) = (e, f)$. In either case, $\epsilon_{\mu\nu} = 0$ by Theorem 4. Thus Theorem 3 is a special case of Theorem 4.

6 Supports with Three or Fewer Exposed Points Preserve Maximality

Conjecture 1 guesses that when $|X_{\mu\nu}| \leq 3$ for all $(\mu, \nu) \in K_{mn}$, $D(S)$ is maximal, that is, $|D(S)| \neq 0$. The conjecture is trivially true when $\epsilon_{\mu\nu} = 0$ for all $(\mu, \nu) \in K_{mn}$. This is because in this case we have $|X_{\mu\nu}| \leq 2$ for all $(\mu, \nu) \in K_{mn}$ [2]. This section proves that the conjecture is true for the special case when $\epsilon_{\mu\nu} \leq 1$ for all $(\mu, \nu) \in K_{mn}$.

In the following discussions, we adopt the convention that, for any set X,

$$X^1 = X, \qquad \text{and} \qquad X^0 = \emptyset. \tag{45}$$

6.1 All Possible Exposed Points

Given that $|X_{\mu\nu}| \leq 3$ and $\epsilon_{\mu\nu} \leq 1$ for all $(\mu, \nu) \in K_{mn}$, by Theorems 3 and 4, the exposed points of $S \sqsubseteq S_{mn}$ must be as follows (the notation "|" means "or"):

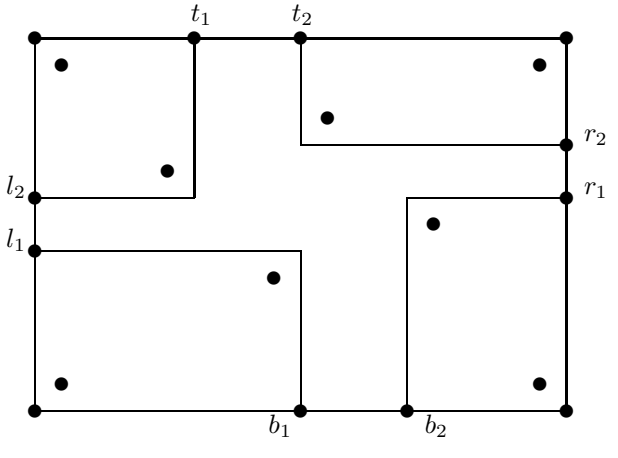
$$
\begin{aligned}
X_{00} &= \{(0,0)\} && (46)\\
&| \quad \{(b_1,0),(0,l_1)\} && (47)\\
&| \quad \{(b_1,0),(1,1),(0,l_1)\} && (48)\\
&| \quad \{(b_1,0),(b_1-1,l_1-1),(0,l_1)\}; && (49)
\end{aligned}
$$

$$
\begin{aligned}
X_{m0} &= \{(m,0)\} && (50)\\
&| \quad \{(b_2,0),(m,r_1)\} && (51)\\
&| \quad \{(b_2,0),(m-1,1),(m,r_1)\} && (52)\\
&| \quad \{(b_2,0),(b_2+1,r_1-1),(m,r_1)\}; && (53)
\end{aligned}
$$

$$
\begin{aligned}
X_{mn} &= \{(m,n)\} && (54)\\
&| \quad \{(t_2,n),(m,r_2)\} && (55)\\
&| \quad \{(t_2,n),(m-1,n-1),(m,r_2)\} && (56)\\
&| \quad \{(t_2,n),(t_2+1,r_2+1),(m,r_2)\}; && (57)
\end{aligned}
$$

$$
\begin{aligned}
X_{0n} &= \{(0,n)\} && (58)\\
&| \quad \{(0,l_2),(t_1,n)\} && (59)\\
&| \quad \{(t_1,n),(1,n-1),(0,l_2)\} && (60)\\
&| \quad \{(t_1,n),(t_1-1,l_2+1),(0,l_2)\}. && (61)
\end{aligned}
$$

The following figure displays all these possible exposed point configurations.



6.2 Columns Indexed by Exposed Points in the Column Support

To prove that $D(S)$ is maximal it suffices to prove that columns of $D(S)$ indexed by the following indices are linearly independent.

$$C_X = ((X''_{00})^{\epsilon_{00}} \cup (X''_{m0})^{\epsilon_{m0}} \cup (X''_{mn})^{\epsilon_{mn}} \cup (X''_{0n})^{\epsilon_{0n}}) \cap C(S), \tag{62}$$

This is because the columns of $D(S)$ indexed by

$$X''_{\mu\nu} \cap C(S) \tag{63}$$

where $|X_{\mu\nu}| = 3$ produces the bracket factor $B_{\mu\nu}^{\epsilon_{\mu\nu}} = B_{\mu\nu}$ corresponding to the three exposed points at corner (μ, ν) [14]. Thus a maximal minor M of $D(S)$ containing these columns produces the factor $B_{00}^{\epsilon_{00}} B_{m0}^{\epsilon_{m0}} B_{mn}^{\epsilon_{mn}} B_{0n}^{\epsilon_{0n}}$. But a maximal minor is a multiple of the resultant for the support S [10], by Equation (41) we have

$$\dim(M) - \sum_{(\mu,\nu)\in K_{mn}} \epsilon_{\mu\nu} \geq \dim(D(S)) - \sum_{(\mu,\nu)\in K_{mn}} \epsilon_{\mu\nu}. \tag{64}$$

Consequently $\dim(M) = \dim(D(S))$ and $D(S)$ is maximal.

6.3 Corresponding Row Indices

The four sets of corresponding row indices are given as follows [6].

Row Indices R_{00}:

$$R_{00} = (P, (b_1 - 1, l_1 - 1), Q) \tag{65}$$

where

$$P = \begin{cases} (0, l_1), & (1,1) \in X_{00}; \\ (b_1 - 2, 2l_1 - 2), & (b_1 - 1, l_1 - 1) \in X_{00}; \end{cases} \tag{66}$$

and

$$Q = \begin{cases} (b_1 - 1, n - 1), & |X_{0n}| = 1; \\ (t_1 - 1, n + l_2 - 1), & |X_{0n}| > 1. \end{cases} \tag{67}$$

Row Indices R_{m0}:

$$R_{m0} = \{P, (b_2, r_1 - 1), Q\} \tag{68}$$

where

$$P = \begin{cases} (m - 1, r_1), & (m - 1, 1) \in X_{m0}; \\ (b_2 + 1, 2r_1 - 2), & (b_2 + 1, r_1 - 1) \in X_{m0}; \end{cases} \tag{69}$$

and

$$Q = \begin{cases} (b_2, n - 1), & |X_{mn}| = 1; \\ (t_2, n + r_2 - 1), & |X_{mn}| > 1. \end{cases} \tag{70}$$

Row Indices R_{mn}:

$$R_{mn} = \{P, (t_2, r_2 + n), Q\} \tag{71}$$

where

$$P = \begin{cases} (m-1, n+r_2-1), & (m-1, n-1) \in X_{mn}; \\ (t_2+1, 2r_2+1), & (t_2+1, r_2+1) \in X_{mn}; \end{cases} \tag{72}$$

and

$$Q = \begin{cases} (t_2, n), & |X_{m0}| = 1; \\ (b_2, r_1), & |X_{m0}| > 1. \end{cases} \tag{73}$$

Row Indices R_{0n}:

$$R_{0n} = \{P, (t_1 - 1, l_2 + n), Q\} \tag{74}$$

where

$$P = \begin{cases} (0, n+l_2-1), & (1, n-1) \in X_{0n}; \\ (t_1-2, 2l_2+1), & (t_1-1, l_2+1) \in X_{0n}; \end{cases} \tag{75}$$

and

$$Q = \begin{cases} (t_1 - 1, n), & |X_{00}| = 1; \\ (b_1 - 1, l_1), & |X_{00}| > 1. \end{cases} \tag{76}$$

Let the union of these sets of row indices be

$$R_X = (R_{00}^{\epsilon_{00}} \cup R_{m0}^{\epsilon_{m0}} \cup R_{mn}^{\epsilon_{m0}} \cup R_{0n}^{\epsilon_{0n}}) \cap R(S). \tag{77}$$

6.4 Proving the Maximality of $D(S)$

As discussed previously, to establish the maximality of $D(S)$ we need only show that in general the columns of $D(S)$ indexed by C_X of Equation (62) are linearly independent under all possibilities of $X_{\mu\nu}$, $(\mu, \nu) \in K_{mn}$. This is so if a square submatrix of these columns is non-singular. To this end consider the submatrix M indexed by row indices R_X and column indices C_X. The submatrix M can be obtained by computing its entries with the column entry formulas (35). It is then readily verified that either the submatrix M is triangular with non-zero diagonal entries or its determinant is non-zero. Consequently M is non-singular.

That is, the columns indexed by C_X are linearly independent in general and thus $D(S)$ is maximal in general.

6.5 Provision for Degeneracies

Using column entry formulas (35) or otherwise, it is easy to check that the column corresponding to an exposed point (x, y) is zero if (x, y) is the only exposed point of $S \sqsubseteq S_{mn}$ on a vertical edge of S_{mn}. For example, if the sets of exposed points are

$\{(t_1, n), (1, n-1), (0, l)\}$	$\{(t_1, n), (m-1, n-1), (m, r)\}$
$\{(b_1, 0), (1, 1), (0, l)\}$	$\{(b_2, 0), (m-1, 1), (m, r)\}$

, (78)

then $(0, l)$ is the only exposed point on the edge $x = 0$ and (m, r) is the only exposed point on the edge $x = m$. In this case $|C_X| = 8$ and the 8×8 submatrix indexed by R_X (without the 3rd, 6th, 9th, and 12th rows) and C_X is an diagonal matrix with diagonal entries

$$B_1, -B_1, -B_2, B_2, B_3, -B_3, -B_4, B_4 \tag{79}$$

where the brackets are

$$B_1 = (b_1, 0, 1, 1, 0, l), \tag{80}$$
$$B_2 = (b_2, 0, m-1, 1, m, r), \tag{81}$$
$$B_3 = (t_2, n, m-1, n-1, m, r), \tag{82}$$
$$B_4 = (t_1, n, 1, n-1, 0, l). \tag{83}$$

That is, for any $(\mu, \nu) \in K_{mn}$, if $|X_{\mu\nu}| = 3$ but $|X''_{\mu\nu} \cap C(S)| = 2$, there is degeneracy and the submatrix should be formed by omitting the point Q from the row indices in $R_{\mu\nu}$

7 Conclusions

With the aid of column entry formulas (35), it is straightforward to show that the columns listed in (62) are linearly independent in general because a square submatrix of these columns is non-singular. Consequently, arguing using the degree of the sparse resultant and the fact that a maximal minor of $D(S)$ is a multiple of the sparse resultant, we are able to show that $D(S)$ is maximal in general when $|X_{\mu\nu}| \leq 3$ and $\epsilon_{\mu\nu} \leq 1$ for any $(\mu, \nu) \in K_{mn}$.

In the proof the theories of sparse resultants and BKK bounds, and the fact that a maximal minor is a multiple of the resultant are needed. It would be nice if the maximality of $D(S)$ can be established directly using elementary and constructive means in future.

References

1. Char, B.W., Geddes, K.O., Gonnet, G.H., Leong, B.L., Monogan, M.B., Watt, S.M.: Maple V language reference manual. Springer, New York (1991)
2. Chionh, E.-W.: Rectangular corner-cutting and Dixon $\mathcal{A}$-resultants. Journal of Symbolic Computation 31, 651–669 (2001)
3. Chtcherba, A.D., Kapur, D.: Resultants for unmixed bivariate polynomial systems using the Dixon formulation. Journal of Symbolic Computation 38, 915–958 (2004)
4. Cox, D., John, L., Donal, O.: Using Algebraic Geometry. Springer, New York (1998)
5. Dixon, A.L.: The eliminant of three quantics in two independent variables. Proc. London Math. 6, 49–96, 473–492 (1908)
6. Foo, M.-C.: Dixon $\mathcal{A}$-resultant formulas. Master thesis, National University of Singapore (2003)
7. Foo, M.-C., Chionh, E.-W.: Corner point pasting and Dixon $\mathcal{A}$-resultant quotients. In: Asian Symposium on Computer Mathematics, pp. 114–127 (2003)
8. Foo, M.-C., Chionh, E.-W.: Corner edge cutting and Dixon $\mathcal{A}$-resultant quotients. Journal of Symbolic Computation 37, 101–119 (2004)
9. Krasauska, R.: Toric surface patches. Advances in Computational Mathematics 17, 89–113 (2002)
10. Emiris, I.Z., Mourrain, B.: Matrices in elimination theory. Journal of Symbolic Computation 28, 3–44 (1999)
11. van der Waerden, B.L.: Modern Algebra, ch. XI, vol. II. Ungar Pub. Co., New York (1950)
12. Xiao, W.: Loose entry formulas and the reduction of Dixon determinant entries. Master thesis, National University of Singapore (2004)
13. Xiao, W., Chionh, E.W.: Formal Power Series and Loose Entry Formulas for the Dixon Matrix. In: Computer Algebra and Geometric Algebra with Applications. Springer, Heidelberg (2005)
14. Xiao, W., Chionh, E.W.: Bracket Producing Rows and Columns of the Dixon Determinant. In: Proceedings of the 8th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2006), pp. 164–170. IEEE, Los Alamitos (2006)

Properties of Ascending Chains for Partial Difference Polynomial Systems*

Gui-Lin Zhang and Xiao-Shan Gao

Key Laboratory of Mathematics Mechanization
Institute of Systems Science, AMSS, Chinese Academy of Sciences
xgao@mmrc.iss.ac.cn, zhangguil@amss.ac.cn

Abstract. A characteristic set theory for partial difference polynomial systems is proposed. We introduce the concept of coherent and regular ascending chains and prove that a partial difference ascending chain is the characteristic set of its saturation ideal if and only if it is coherent and regular. This gives a method to decide whether a polynomial belongs to the saturation ideal of an ascending chain. We introduce the concept of strongly irreducible ascending chains and prove that a partial difference ascending chain is the characteristic set of a reflexive prime ideal if and only if it is strongly irreducible. This gives a simple and precise representation for reflexive prime ideals.

Keywords: Ascending chain, characteristic set, coherent chain, regular chain, irreducible chain, partial difference polynomial.

1 Introduction

The characteristic set method is a fundamental tool for studying systems of algebraic or algebraic differential equations. The method could be used to transform an equation system into so-called characteristic sets, which are systems of equations in certain triangular form also called ascending chains, or simply chains. This allows people to give the dimension, the order, and the degree of a solution set over an algebraically or differentially closed field. Also, triangular equation systems are ready for symbolic and numerical solutions.

The characteristic set method was introduced by Ritt in the 1930s as an algebraic tool to study differential equations [17, 19]. However, the algorithmic study of the characteristic set was in stagnation for quite a long time until Wu's work on zero decomposition for polynomial equations and automated geometry theorem proving appeared in the late 1970s [23,24,25]. Since then, many efficient algorithms and new properties for characteristic sets were proposed for algebraic equation systems and differential equation systems [1,2,4,5,6,7,8,9,11,12,16,22, 27]. In [13, 14, 15], a characteristic set method was also introduced for ordinary difference equation systems and ordinary differential-difference equation systems.

In this paper, we develop a characteristic set theory for partial difference polynomial systems. We obtain three main results. First, we introduce the concept

* Supported by a National Key Basic Research Project of China (2004CB318000).

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 307–321, 2008.
© Springer-Verlag Berlin Heidelberg 2008

of coherent chains and this leads to a normal representation of the difference polynomials in the saturation ideal of a coherent chain. Second, we introduce the concept of regular chains and prove that a partial difference chain is the characteristic set of its saturation ideal if and only if it is coherent and regular. We also prove that the saturation ideal of a partial difference regular and coherent chain is the union of some algebraic saturation ideals. This gives a method to decide whether a polynomial belongs to the saturation ideal of a chain. Third, we introduce the concept of strongly irreducible chains and prove that a partial difference chain is the characteristic set of a reflexive prime ideal if and only if it is strongly irreducible. This gives a simple and precise representation for prime and reflexive prime ideals. These results are generalizations of similar results about algebraic polynomial systems [2], differential systems [6,16], and ordinary difference systems [13,14]. Due to the complicated structure of partial difference polynomials, our generalization is nontrivial and there still exist many problems unsolved in the partial difference case. The major open problem is to give a constructive criterion for regular and non-trivial chains. For details, please see Section 4.

In [18,20,26], the characteristic set of partial differential and difference polynomial systems was defined and used to prove the Noetherian property of the partial differential and difference polynomial ring. Dimension polynomials for differential and difference polynomial ideals were also studied in [18,26]. But, the results presented in this paper for regular and irreducible chains were not given in these papers.

The rest of this paper is organized as follows. In Section 2, we present the notations and known results needed in this paper. In Sections 3, 4, and 5, we prove the properties of coherent, regular, and strongly irreducible chains respectively. In Section 6, we give the zero decomposition theorem and algorithm. In Section 7, we conclude the paper.

2 Preliminaries

We will introduce the notions and preliminary properties needed in this paper. For the general theory of difference algebra, please refer to [3,10,18].

2.1 Difference Polynomials and Difference Chains

Let $\mathcal{K}$ be a field of characteristic zero. We say that $\mathcal{K}$ is an inversive partial difference field with transforming operators $\{\sigma_1, \ldots, \sigma_m\}$ over $\mathcal{K}$, if $\{\sigma_1, \ldots, \sigma_m\}$ are automorphisms of $\mathcal{K}$ onto $\mathcal{K}$ which commute pairwise on the elements of $\mathcal{K}$. Let

$$T_\sigma = \{\sigma_1^{o_1} \ldots \sigma_m^{o_m} | j = 1, \ldots, m, o_j \geq 0\}.$$

We regard T_σ as a free commutative monoid. The *order* of an element $\eta = \sigma_1^{o_1} \ldots \sigma_m^{o_m}$ of T_σ is $\mathrm{ord}(\eta) = \Sigma_{j=1}^m o_j$. η is proper if $\mathrm{ord}(\eta) \neq 0$. The *vector* of an element $\eta \in T_\sigma$ is $\mathrm{vec}(\eta) = (i_1, \ldots, i_m)$ if $\eta = \sigma_1^{i_1} \ldots \sigma_m^{i_m}$. For $\eta_1, \eta_2 \in T_\sigma$, η_1 is a

(proper) multiple transform of η_2 or η_2 is a *(proper) factor transform* of η_1 if $\exists$ $\eta \in T_\sigma$ such that $\eta_1 = \eta\eta_2$ ($\mathrm{ord}(\eta) \neq 0$). It is denoted by $\eta_1 \gg \eta_2 (\eta_1 \succ \eta_2)$.

Let $\mathbb{X} = \{x_1, \ldots, x_n\}$ be a finite set of difference indeterminates over $\mathcal{K}$ and

$$T_\sigma \mathbb{X} = \{\eta x_i | \eta \in T_\sigma, i = 1, \ldots, n\}.$$

$\mathcal{R} = \mathcal{K}\{x_1, \ldots, x_n\} = \mathcal{K}[T_\sigma \mathbb{X}]$ denotes the ring of partial difference polynomials in the indeterminates $\mathbb{X}$ with coefficients in $\mathcal{K}$. For convenience, in this paper, when we say polynomials, we mean partial difference polynomials, otherwise we will point out clearly.

Let $<$ be a *total ordering* over $T_\sigma \mathbb{X}$ defined as follows: $\forall \eta, \theta \in T_\sigma, 1 \leq i, j \leq n$, $\eta x_i > \theta x_j$ if $i > j$ or $i = j$ and $\mathrm{ord}(\eta) > \mathrm{ord}(\theta)$ or else $\mathrm{ord}(\eta) = \mathrm{ord}(\theta)$ and the first nonzero element of $\mathrm{vec}(\eta) - \mathrm{vec}(\theta)$ is greater than zero. Let f be a polynomial not in $\mathcal{K}$, the *leader* of f is the highest element of $T_\sigma \mathbb{X}$ (*w.r.t.* $<$) that appears in f, and we denote it by $\mathbf{u}_f$. We write f as a univariate polynomial in $\mathbf{u}_f$:

$$f = I_d \mathbf{u}_f^d + \cdots + I_0.$$

$I_d = \mathrm{init}(f)$ is called the *initial* of f. Let $\mathbf{u}_f = \eta x_i$. Then i and x_i are called the *class* and *leading variable* of f, denoted as $\mathrm{class}(f)$ and $\mathrm{lvar}(f)$ respectively. We define $\mathrm{vec}(\eta x_i) = \mathrm{vec}(\eta)$ and $\mathrm{vec}(f, x_j) = \mathrm{vec}(\eta)$, if $\eta x_j = \max\{\tau x_j$ appears in $f\}$, $\mathrm{vec}(f) = \mathrm{vec}(f, \mathrm{lvar}(f))$.

An *n-tuple* over $\mathcal{K}$ is of the form $\mathbf{a} = (a_1, \ldots, a_n)$, where the a_i are selected from some difference extension field of $\mathcal{K}$. Let $f \in \mathcal{K}\{\mathbb{X}\}$. To substitute an n-tuple $\mathbf{a}$ into f means to replace each of the ηx_i occurring in f with the corresponding ηa_i. Let $\mathbb{P}$ be a set of polynomials in $\mathcal{K}\{\mathbb{X}\}$. An n-tuple over $\mathcal{K}$ is called a *solution* of the equation set $\mathbb{P}$=0 if the result of substituting the n-tuple into each polynomial of $\mathbb{P}$ is zero. We use $\mathrm{Zero}(\mathbb{P})$ to denote the set of solutions of $\mathbb{P} = 0$. Let $f \in \mathcal{K}\{\mathbb{X}\}$. It is easy to check that $\mathrm{Zero}(f) = \mathrm{Zero}(\eta f)$ $\forall \eta \in T_\sigma$. For the sets of polynomials $\mathbb{P}$ and $\mathbb{D}$, $\mathrm{Zero}(\mathbb{P}/\mathbb{D})$ denotes the set of solutions of $\mathbb{P} = 0$ which do not annihilate any polynomial of $\mathbb{D}$.

Let g be a polynomial not in $\mathcal{K}$. A polynomial f is said to be of *less than* g, denoted as $f < g$, if $\mathbf{u}_f < \mathbf{u}_g$ or $(\mathbf{u}_f = \mathbf{u}_g) = u$ and $\deg(f, u) < \deg(g, u)$. If neither $f < g$ nor $g < f$, we say that f and g are equivalent and we write $f \equiv g$. A polynomial f is said *reduced* w.r.t. g if $\deg(f, \eta \mathbf{u}_g) < \deg(g, \mathbf{u}_g), \forall \eta \in T_\sigma$.

A subset $\mathcal{A}$ of $\mathcal{R} \backslash \mathcal{K}$, where every element is reduced w.r.t. all the others, is called an *autoreduced set*. A *chain* is an autoreduced set where the polynomials are listed in the ascending ordering: $\mathcal{A} = A_1 < A_2 < \cdots < A_p$. It is easy to show that every chain $\mathcal{A}$ in $\mathcal{R} = \mathcal{K}\{\mathbb{X}\}$ is a finite set.

If $\mathcal{A} = A_1, \ldots, A_p$ and $\mathcal{B} = B_1, \ldots, B_q$ are two chains, we say that $\mathcal{A} < \mathcal{B}$ if either there is some $j \leq \min(p, q)$ such that $A_i \equiv B_i$ for $i < j$ and $A_j < B_j$, or $q < p$ and $A_i \equiv B_i$ for $i \leq q$. If neither $\mathcal{A} < \mathcal{B}$ nor $\mathcal{B} < \mathcal{A}$, we say that $\mathcal{A}$ and $\mathcal{B}$ are of the same order and we denote $\mathcal{A} \equiv \mathcal{B}$. The following result is a basic property of chains (page 147 in [18]).

Lemma 1. *A strictly decreasing sequence of chains* $\mathcal{A}_1 > \mathcal{A}_2 > \mathcal{A}_3 > \cdots$ *is finite.*

If $\mathcal{F} \subseteq \mathcal{R}$, then the set of all the chains contained in $\mathcal{F}$ has a minimal element according to Lemma 1, which is called a *characteristic set* of $\mathcal{F}$ and it is denoted by $\mathrm{CS}(\mathcal{F})$. A polynomial f is *reduced* w.r.t. a chain if it is reduced to every polynomial in the chain. The following results are easy to prove.

Lemma 2. *If $f \neq 0$ is reduced w.r.t. $\mathrm{CS}(\mathcal{F})$, then $\mathrm{CS}(\mathcal{F} \cup \{f\}) < \mathrm{CS}(\mathcal{F})$.*

Lemma 3. *A chain $\mathcal{A} \subset \mathbb{P}$ is a characteristic set of $\mathbb{P}$ if and only if there is no nonzero polynomial in $\mathbb{P}$ which are reduced w.r.t. $\mathcal{A}$.*

A *difference ideal* is a subset $\mathcal{I}$ of $\mathcal{R} = \mathcal{K}\{x_1, \ldots, x_n\}$, which is an algebraic ideal in $\mathcal{R}$ and is closed under transforming. A difference ideal $\mathcal{I}$ is called *reflexive* if $\eta f \in \mathcal{I}$ implies $f \in \mathcal{I}$ for all $\eta \in T_\sigma$. Let S be a set of elements of $\mathcal{R}$. The difference ideal generated by S is denoted by $[S]$. Obviously, $[S]$ is the set of all linear combinations of the polynomials in S and their transforms. The ordinary or algebraic ideal generated by S is denoted as (S). A difference ideal $\mathcal{I}$ of $\mathcal{R}$ is called *perfect* if the presence in $\mathcal{I}$ of a product of transforms of an element f of $\mathcal{R}$ implies $f \in \mathcal{I}$. The perfect difference ideal generated by S is denoted as $\{S\}$. A perfect ideal is always reflexive. A difference ideal $\mathcal{I}$ is called a *prime difference ideal* if it is prime as an algebraic ideal.

Let $\mathcal{A}$ be a chain and $\mathbb{I}_\mathcal{A}$ the set of products of the initials of the polynomials in $\mathcal{A}$ and their transforms. The *saturation ideal* of $\mathcal{A}$ is defined as follows

$$\mathbf{sat}(\mathcal{A}) = \{f \in \mathcal{K}\{\mathbb{X}\} \,|\, \exists J \in \mathbb{I}_\mathcal{A}, s.t.\, Jf \in [\mathcal{A}]\}.$$

2.2 Invertibility of Algebraic Polynomials

We will introduce some notations and results about invertibility of algebraic polynomials w.r.t. an algebraic ascending chain. These results are given in [1, 2, 5, 6].

Let $\mathcal{A} = A_1, \ldots, A_m$ be a nontrivial triangular set in $\mathcal{K}[x_1, \ldots, x_n]$ over a field $\mathcal{K}$ of characteristic zero. Let y_i be the leading variable of A_i, $y = \{y_1, \ldots, y_p\}$ and $u = \{x_1, \ldots, x_n\} \setminus y$. u is called the parameter set of $\mathcal{A}$. We can denote $\mathcal{K}[x_1, \ldots, x_n]$ as $\mathcal{K}[u, y]$. I_i is the initial of A_i. For a triangular set $\mathcal{A}$, let $I_\mathcal{A}$ be the set of products of the initials of the polynomials in $\mathcal{A}$. The *algebraic saturation ideal* of a triangular set $\mathcal{A}$ is defined as follows

$$\mathbf{a\text{-}sat}(\mathcal{A}) = \{f \in \mathcal{K}[x_1, \ldots, x_n] \,|\, \exists J \in I_\mathcal{A}, s.t. Jf \in (\mathcal{A})\}.$$

Definition 4. *Let $\mathcal{A} = A_1, A_2, \ldots, A_m$ be a nontrivial triangular set in $\mathcal{K}[u, y]$ with u as the parameter set, and $f \in \mathcal{K}[u, y]$. f is said to be invertible w.r.t. $\mathcal{A}$ if $(f, A_1, \ldots, A_s) \cap \mathcal{K}[u] \neq \{0\}$ where $s = \mathrm{class}(f)$. $\mathcal{A}$ is called regular if the initials of A_i are invertible w.r.t. $A_1, \ldots, A_{i-1}$.*

Theorem 5. *[2, 6] Let $\mathcal{A}$ be a triangular set. Then $\mathcal{A}$ is a characteristic set of* $\mathbf{a\text{-}sat}(\mathcal{A})$ *iff $\mathcal{A}$ is regular.*

Lemma 6. *[6] A finite product of polynomials which are invertible w.r.t. $\mathcal{A}$ is also invertible w.r.t. $\mathcal{A}$.*

Lemma 7. *[6] A polynomial g is not invertible w.r.t. a regular triangular set $\mathcal{A}$ iff there is a nonzero f in $\mathcal{K}[u, y]$ such that $fg \in (\mathcal{A})$ and g is reduced w.r.t. $\mathcal{A}$.*

Lemma 8. *[25] Let $\mathcal{A}$ be an irreducible triangular set. Then a polynomial g is invertible w.r.t. $\mathcal{A}$ iff $g \notin$ **a-sat**(A).*

3 Coherent Chains

For any chain $\mathcal{A}$, after a proper renaming of variables, we could write it as the following form:

$$\mathcal{A} = \begin{cases} A_{1,1}(u, y_1), \ldots, A_{1,k_1}(u, y_1) \\ A_{2,1}(u, y_1, y_2), \ldots, A_{2,k_2}(u, y_1, y_2) \\ \ldots \\ A_{p,1}(u, y_1, \ldots, y_p), \ldots, A_{p,k_p}(u, y_1, \ldots, y_p) \end{cases} \tag{1}$$

where $\mathrm{lvar}(A_{i,j}) = y_i$, $u = \{u_1, \ldots, u_q\}$ such that $p+q = n$, $\mathbb{X} = u \bigcup \{y_1, \ldots, y_p\}$. For $c = 1, \ldots, p$, let

$$\mathcal{A}_c = A_{c,1}(u, y_1, \ldots, y_c), \ldots, A_{c,k_c}(u, y_1, \ldots, y_c) \tag{2}$$

3.1 Prolongation of Chains

We will now introduce the prolongation of a chain, which is a key concept in our theory. For instance, we will use this concept to define the pseudo-remainder of a polynomials w.r.t. a chain.

For a set of polynomials $\mathbb{P}$, we use $\mathbb{L}_{\mathbb{P}}$ to denote the set of leaders of the polynomials in $\mathbb{P}$. For $\eta x_c \in T_\sigma$, we use $\mathbb{D}_{\eta x_c}$ to denote the set of θx_c such that θ is a factor of η. More precisely, we have:

$$\begin{aligned} \mathbb{L}_{\mathbb{P}} &= \{\eta x_c \in T_\sigma \mathbb{X} \quad s.t. \quad \exists\, P \in \mathbb{P}, \mathbf{u}_P = \eta x_c\}. \\ \mathbb{D}_{\eta x_c} &= \{\theta x_c \in T_\sigma \mathbb{X} \quad s.t. \quad \eta \gg \theta\}. \end{aligned}$$

We define the *main variables* and *parameters* of a chain $\mathcal{A}$ as follows.

$$\begin{aligned} \mathbb{MV}_{\mathcal{A}} &= \{\eta x_c \in T_\sigma \mathbb{X} \quad s.t. \quad \exists\, A \in \mathcal{A}, \mathbf{u}_A = \theta x_c, \text{ and } \eta \gg \theta\}. \\ \mathbb{PA}_{\mathcal{A}} &= T_\sigma \mathbb{X} \setminus \mathbb{MV}_{\mathcal{A}}. \end{aligned}$$

For any finite set of polynomials $\mathbb{P}$ and a chain $\mathcal{A}$, we say that $\mathcal{A}_{\mathbb{P}}$ is a *prolongation of* $\mathcal{A}$ w.r.t. $\mathbb{P}$ if it satisfies the following properties:

- $\mathcal{A}_{\mathbb{P}} \supseteq \mathcal{A}$ is an algebraic triangular set under the ordering $\leq$ when all $\eta x_i \in T_\sigma \mathbb{X}$ are considered as independent variables.
- If $A \in \mathcal{A}_{\mathbb{P}}$, then there exist a $B \in \mathcal{A}$ and an $\eta \in T_\sigma$ such that $A = \eta B$ and B has the lowest degree among all elements in $\{C | \mathbf{u}_A = \mathbf{u}_{\theta C},\ C \in \mathcal{A}\}$.

– For any ηx_c occurring in $\mathbb{P} \cup \mathcal{A}_{\mathbb{P}}$, either $\eta x_c \in \mathbb{PA}_{\mathcal{A}}$ or there exists an $A \in \mathcal{A}_{\mathbb{P}}$ such that $\mathbf{u}_A = \eta x_c$.

Intuitively speaking, $\mathcal{A}_{\mathbb{P}}$ is a finite subset of $T_\sigma A$ such that each ηx_i occurring in $\mathbb{P}$ is either in $\mathbb{PA}_{\mathcal{A}}$ or a leader of a polynomial in $\mathcal{A}_{\mathbb{P}}$. It is easy to show that $\mathcal{A}_{\mathbb{P}}$ satisfies the following properties.

– The parameters of $\mathcal{A}_{\mathbb{P}}$ as an algebraic triangular set are all in $\mathbb{PA}_{\mathcal{A}}$.
– A polynomial f is reduced w.r.t. $\mathcal{A}$ if and only if f is reduced w.r.t. $\mathcal{A}_f$ in the algebraic sense, where $\mathcal{A}_f = \mathcal{A}_{\{f\}}$.

The following algorithm can be used to compute a prolongation $\mathcal{A}_{\mathbb{P}}$, for a given chain $\mathcal{A}$ and a polynomial set $\mathbb{P}$.

Algorithm 9. Prolongation$(\mathcal{A}, \mathbb{P})$

– **Input**:*A chain* $\mathcal{A}$ *of form (1) and a finite set of polynomials* $\mathbb{P}$.
– **Output**: *A prolongation* $\mathcal{A}_{\mathbb{P}}$ *of* $\mathcal{A}$ *w.r.t.* $\mathbb{P}$.

Begin
  $\mathcal{A}_{\mathbb{P}} := \mathcal{A}$
  For i=p to 1
    $\Omega_i := \{\eta \mid \eta y_i$ *appears in* $\mathcal{A}_i, \mathcal{A}_{i+1}, \ldots, \mathcal{A}_p$ *or* $\mathbb{P}\}$;
    $\tau :=$ *LCM*(Ω_i)
    For all $\eta \ll \tau$
      $\Omega_\eta := \{A \mid A \in \mathcal{A}_i, \exists\ \theta \in T_\sigma, \theta \mathbf{u}_A = \eta y_i\}$,
      choose an element A *of* Ω_η *with the least degree s.t.* $\theta \mathbf{u}_A = \eta y_i$.
      $\mathcal{A}_{\mathbb{P}} := \mathcal{A}_{\mathbb{P}} \cup \theta A$
    $\Lambda := \{\eta \mid \eta y_i$ *occurring in* $\mathcal{A}_{\mathbb{P}}, \eta y_i \notin \mathbb{PA}_{\mathcal{A}}$ *and* $\forall\ A \in \mathcal{A}_{\mathbb{P}}$, $\mathbf{u}_A \neq \eta y_i\}$
    While $(\Lambda \neq \emptyset)$
      $\theta :=$ *max* Λ
      For all $\bar{\theta} \in \Lambda$ *and* $\bar{\theta} \ll \theta$
        choose $A \in \mathcal{A}$ *with the least degree, s.t.* $\exists\ \theta' \in T_\sigma$, $\theta' \mathbf{u}_A = \bar{\theta} y_i$
        $\mathcal{A}_{\mathbb{P}} := \mathcal{A}_{\mathbb{P}} \cup \theta' A$
      $\Lambda := \{\eta \mid \eta y_i$ *occurring in* $\mathcal{A}_{\mathbb{P}}, \eta y_i \notin \mathbb{PA}_{\mathcal{A}}$ *and* $\forall\ A \in \mathcal{A}_{\mathbb{P}}$
        $\mathbf{u}_A \neq \eta y_i\}$
    End While
  $i := i - 1$
End.

The termination of the algorithm is apparent if we notice that the sequence of elements of $\theta := \max \Lambda$ is strictly decreasing.

Example 1. Consider the chain $\mathcal{A} = \{A_1, A_2, A_3\} \subseteq \mathcal{K}\{y\}$. The transforming operators are $\{\sigma_1, \sigma_2\}$.

$$A_1 = \sigma_2^2 \sigma_1 y^2, A_2 = \sigma_2 \sigma_1^3 y + \sigma_2 y, A_3 = \sigma_2^3 \sigma_1^2 y + \sigma_2^4 y \tag{3}$$

Let vec$(\mathcal{A})$ denotes all the vec$(\mathcal{A}_i)$ for a chain $\mathcal{A}$. Then, elements of vec$(\mathcal{A})$ are represented by circles in Figure 1. We have $\mathbb{PA}_{\mathcal{A}} = \{y, \sigma_2\sigma_1 y, \sigma_2\sigma_1^2 y, \sigma_1^i y, \sigma_2^j y \mid i, j \in \mathbb{N}\}$.

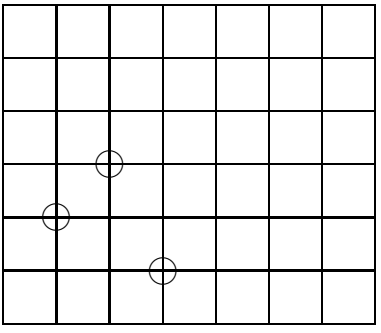


Fig. 1. The $\mathrm{vec}(\mathcal{A})$ for $\mathcal{A}$

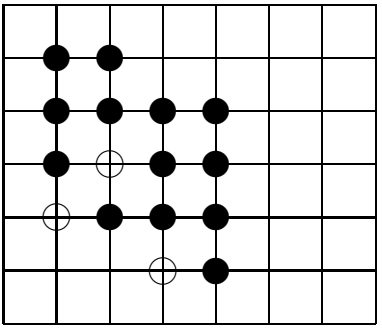


Fig. 2. The $\mathrm{vec}(\mathcal{A}_P)$ for $\mathcal{A}_P$

For $P = \sigma_2^3\sigma_1^4 y + \sigma_2^4 y$, we have $\mathcal{A}_P = \{A_1, \sigma_2 A_1, \sigma_1 A_1, A_2, \sigma_2^2 A_1, A_3, \sigma_2 A_2, \sigma_1 A_2, \sigma_2^3 A_1, \sigma_2 A_3, \sigma_1 A_3, \sigma_2\sigma_1 A_2, \sigma_2^2 A_3, \sigma_2\sigma_1 A_3, \sigma_1^2 A_3, \sigma_2\sigma_1^2 A_3\}$. The elements of $\mathrm{vec}(\mathcal{A}_P)$ are given in Figure 2. The new elements of $\mathrm{vec}(\mathcal{A}_P)$ are represented by black dots.

We use $\mathrm{prem}(f, g, x)$ to denote the algebraic pseudo-remainder of f w.r.t. g relative to variable x, $\mathrm{prem}(f, g)$ is $\mathrm{prem}(f, g, x)$ where x is the leading variable of g.

With these notations, we define the *difference pseudo remainder* of f w.r.t. $\mathcal{A}$ to be: $\mathrm{rprem}(f, \mathcal{A}) = \mathrm{prem}(f, \mathcal{A}_f)$ where the variables and their transforms in f and $\mathcal{A}$ are treated as independent algebraic variables. The following lemma is clear.

Lemma 10. *Let $f, \mathcal{A}$ be as above and $r = \mathrm{rprem}(f, \mathcal{A})$. Then, there is a $J \in \mathbb{I}_{\mathcal{A}}$ such that $\mathbf{u}_J < \mathbf{u}_f$,*

$$Jf \equiv r \bmod [\mathcal{A}] \tag{4}$$

and r is reduced w.r.t. $\mathcal{A}$. Equation (4) is called the *remainder formula*.

3.2 Coherent Chains

It is clear that the prolongation of a chain is not unique since for some $A \in \mathcal{A}_{\mathbb{P}}$ we may choose different A_1 and A_2 in $\mathcal{A}$ to generate $A : \mathbf{u}_A = \mathbf{u}_{\theta_1 A_1} = \mathbf{u}_{\theta_2 A_2}$. The concept of coherent chain is to guarantee that all these different prolongations of a chain are equivalent in certain sense.

Definition 11. *Let $\mathcal{A} = A_1, \ldots, A_l$ be a chain in $\mathcal{K}\{\mathbb{X}\}$ and $v_i = \mathrm{vec}(\mathbf{u}_{A_i})$, $i = 1, \ldots, l$. For any $1 \leq i < j \leq m$, if $\mathrm{class}(A_i) = \mathrm{class}(A_j) = t$, let the least common multiple transform of $\mathbf{u}_{A_i}$ and $\mathbf{u}_{A_j}$ be $\eta_{i,j}\mathbf{u}_{A_i} = \eta_{j,i}\mathbf{u}_{A_j}$. We define the Δ-*polynomials *of A_i and A_j as $\Delta_{j,i} = \eta_{j,i}A_j$ and $\Delta_{i,j} = \eta_{i,j}A_i$. If $\mathrm{rprem}(\Delta_{i,j}, \mathcal{A}) = 0$ and $\mathrm{rprem}(\Delta_{j,i}, \mathcal{A}) = 0$, we call $\mathcal{A}$ a* coherent chain. *Let $\Delta(\mathcal{A})$ be the set of all the Δ-polynomials of $\mathcal{A}$.*

Let $\mathcal{A} = A_1, \ldots, A_l$ be a chain. A representation $g = \sum_{i,j} g_{i,j}\eta_{i,j}A_i$ is called *canonical representation* if $\eta_{i,j}A_i$ in the expression are distinct elements in $\mathcal{A}_f$ for some polynomial f. In other words, $g \in (\mathcal{A}_f)$.

Let $\mathcal{A}^* = \mathcal{A}_{\mathcal{A}}$.

Lemma 12. *With the notation of Definition 11. Then the initials appeared in* $\mathrm{rprem}(\Delta_{j,i}, \mathcal{A})$ *are all in* $I_{\mathcal{A}^*}$.

Proof: It is apparent due to the definition of the coherent chain. ∎

Lemma 13. *Let* $\mathcal{A}$ *be a coherent chain of form (1),* $A \in \mathcal{A}$, *and* $\eta \in T_\sigma$. *Then there is a* $J \in \mathbb{I}_{\mathcal{A}}$ *such that* $\mathbf{u}_J < \mathbf{u}_{\eta A}$ *and* $J\eta A$ *has a canonical representation.*

Proof: Let $c = \mathrm{class}(A)$. The polynomials in $\mathcal{A}$ with class c are $A_{c,1}, \ldots, A_{c,i-1}$, $A_{c,i} = A, \ldots, A_{c,k_c}$.

First, if $\mathbf{u}_{\eta A}$ is not the multiple transform of any one of $\mathbf{u}_{A_1}, \ldots, \mathbf{u}_{A_{i-1}}$, $\mathbf{u}_{A_{i+1}} \ldots, \mathbf{u}_{A_{c,k_c}}$, then $\eta A \in \mathcal{A}_{\eta A}$. Second, suppose that $\mathbf{u}_{\eta A}$ is the multiple transform of $\mathbf{u}_{A_{c,k}}$, but $\eta A \in \mathcal{A}_{\eta A}$.

Otherwise, we will prove this by induction on the ordering of $\mathbf{u}_{\eta A}$. Let the least common transform of $\mathbf{u}_A$ and $\mathbf{u}_{A_{c,k}}$ be $\mathbf{u}_{\eta_i A} = \mathbf{u}_{\eta_k A_{c,k}}$, $\Delta_{i,k} = \eta_i A$, $\bar{\eta}\eta_i = \eta$, so $\eta A = \bar{\eta}\Delta_{i,k}$. Since $\mathcal{A}$ is a coherent chain, $\mathrm{rprem}(\Delta_{i,k}, \mathcal{A}) = 0$. We have

$$\bar{J}\Delta_{i,k} = g_1\tau_1 B_1 + g_2\tau_2 B_2 + \cdots$$

where $B_j \in \mathcal{A}$, $\tau_j B_j \in \mathcal{A}_{\Delta_{i,j}}$, and $\mathbf{u}_{\bar{J}} < \mathbf{u}_{\Delta_{i,k}}$, $\mathrm{degree}(\Delta_{i,k}, \mathbf{u}_{\Delta_{i,k}}) \geq \mathrm{degree}(\tau_1 B_1, \mathbf{u}_{\tau_1 B_1})$, $\mathbf{u}_{\Delta_{i,k}} = \mathbf{u}_{\tau_1 B_1} > \mathbf{u}_{\tau_2 B_2} > \cdots$. Let $\bar{\eta}$ act on the two sides of the above equation and we get

$$\bar{\eta}\bar{J} \cdot \bar{\eta}\Delta_{i,j} = \bar{\eta}g_1 \cdot \bar{\eta}\tau_1 B_1 + \bar{\eta}g_2 \cdot \bar{\eta}\tau_2 B_2 + \cdots$$

We denote it by

$$J_1\eta A = \bar{g}_1 \cdot \rho_1 B_1 + \bar{g}_2 \cdot \rho_2 B_2 + \cdots$$

where $J_1 = \bar{\eta}\bar{J}$, $\mathbf{u}_{J_1} < \eta A$, $\rho_j = \bar{\eta}_j\tau_j$. If $\rho_1 B_1$ is not of the first two cases, we continue the above process on $\rho_1 B_1$ until we get (after rearrange the symbols properly)

$$J_2\eta A = f_1 \cdot \theta_1 C_1 + f_2 \cdot \theta_2 C_2 + \cdots$$

where $C_j \in \mathcal{A}$ $\theta_j \in T_\sigma$ $\mathbf{u}_{\eta A} = \mathbf{u}_{\theta_1 C_1} > \mathbf{u}_{\theta_2 C_2} > \cdots$ and $\theta_1 C_1$ is of the first two cases, any $\theta_2 C_2, \theta_3 C_3, \ldots$ satisfy the induction hypothesis. Then there is a $J \in \mathbb{I}_{\mathcal{A}}$ such that $J\eta A$ has a canonical representation.

The following is the main property of the coherent chain.

Theorem 14. *If* $\mathcal{A} = A_1, \ldots, A_l$ *is a coherent chain, then for any* $f = \sum g_{i,j}\eta_j A_i$, *there is a* $J \in \mathbb{I}_{\mathcal{A}}$ *such that* $J \cdot f$ *has a canonical representation and* $\mathbf{u}_J < \max\{\mathbf{u}_{\eta_j A_i}\}$.

Proof: This is a direct consequence of Lemma 13.

Canonical representations are useful because in a canonical representation $\sum g_{i,j}\eta_j A_i$ the polynomial $\eta_i A_i$ with the largest leader is unique and can be eliminated under certain conditions.

4 Regular Chains

Let $\mathcal{A}$ be a chain of form (1), f a polynomial. f is said to be *partial difference invertible, (or invertible)* w.r.t. $\mathcal{A}$ if it is invertible w.r.t. $\mathcal{A}_f$ when f and $\mathcal{A}_f$ are treated as algebraic polynomials.

Definition 15. *Let $\mathcal{A} = A_1, \ldots, A_m$ be a chain and $I_i = \mathrm{init}(A_i)$. $\mathcal{A}$ is said to be (difference) regular if ηI_j is invertible w.r.t. $\mathcal{A}$ for any $\eta \in T_\sigma$ and $1 \leq j \leq m$.*

Lemma 16. *Let $\mathcal{A}$ be a characteristic set of an ideal I. If a polynomial f is invertible w.r.t. $\mathcal{A}$, then $f \notin I$.*

Proof: Let $\mathbb{U}$ be the algebraic parameter set of $\mathcal{A}$. Since f is invertible w.r.t. $\mathcal{A}$, there exists a polynomial g and a nonzero $r \in \mathcal{K}[\mathbb{U}]$ such that $gf = r \mod [\mathcal{A}]$. If $f \in I$, we have $r \in I$. Since r is reduced w.r.t. $\mathcal{A}$, by Lemma 3, we have $r = 0$, a contradiction. ∎

Lemma 17. *If $\mathcal{A}$ is a regular chain of form (1), then $\mathcal{A}_f$ is a regular algebraic triangular set for any polynomial f.*

Proof: If $\mathcal{A}$ is difference regular, then by Definition 15, all ηI_j are invertible w.r.t. $\mathcal{A}$. The initials of the polynomials in $\mathcal{A}_f$ are all of the form ηI_j and they are of ordering lower than the highest ordering of the polynomials in $\mathcal{A}_f$. Then, by Definition 4, $\mathcal{A}_f$ is a regular algebraic triangular set. ∎

Lemma 18. *If a chain $\mathcal{A}$ of form (1) is the characteristic set of $\mathbf{sat}(\mathcal{A})$, then for any polynomial f, $\mathcal{A}_f$ is a regular algebraic triangular set.*

Proof. By Lemma 5, we need only to prove that $\mathcal{B} = \mathcal{A}_f$ is the characteristic set of $\mathbf{a\text{-}sat}(\mathcal{B})$. Let W be the set of all the ηy_j such that ηy_j is of lower or equal ordering than an $\bar{\eta} y_j$ occurring in $\mathcal{B}$. Then $\mathcal{B} \subset \mathcal{K}[W]$. If $\mathcal{B}$ is not the characteristic set of $\mathbf{a\text{-}sat}(\mathcal{B})$, then there is a $g \in \mathbf{a\text{-}sat}(\mathcal{B}) \cap \mathcal{K}[W]$ which is reduced w.r.t. $\mathcal{B}$ and is not zero. g does not contain ηy_i which is of higher ordering than those in W. As a consequence, g is also reduced w.r.t. $\mathcal{A}$. Since $g \in \mathbf{a\text{-}sat}(\mathcal{B}) \subset \mathbf{sat}(\mathcal{A})$ and $\mathcal{A}$ is the characteristic set of $\mathbf{sat}(\mathcal{A})$, g must be zero, a contradiction. ∎

As pointed out in [13], the Rosenfeld Lemma [21] for differential equations can not be extended to difference case. Correspondingly, we have:

Lemma 19. *Let $\mathcal{A}$ be a coherent and regular chain, and r a polynomial reduced w.r.t. $\mathcal{A}$. If $r \in \mathbf{sat}(\mathcal{A})$, then $r = 0$.*

Proof. Let $\mathcal{A} = A_1, A_2, \ldots, A_l$. Since $r \in \mathbf{sat}(\mathcal{A})$, there is a $J_1 \in \mathbb{I}_\mathcal{A}$ such that $J_1 \cdot r \equiv 0 \bmod [\mathcal{A}]$. By Lemma 6, J_1 is difference invertible w.r.t. $\mathcal{A}$, i.e. there is a polynomial $\bar{J}_1$ and a nonzero $N \in \mathcal{K}[V]$ such that

$$\bar{J}_1 \cdot J_1 \equiv N \bmod [\mathcal{A}]$$

where V is the set of parameters of A_{J_1} as an algebraic triangular set. Hence,

$$Nr \equiv \bar{J}_1 \cdot J_1 \cdot r \equiv 0 \bmod [\mathcal{A}].$$

Or equivalently,

$$N \cdot r = \sum g_{i,j}\eta_{i,j}A_j. \tag{5}$$

Since $\mathcal{A}$ is a coherent chain, by Theorem 14, there is a $J_2 \in \mathbb{I}_{\mathcal{A}}$ such that $J_2 \cdot N \cdot r$ has a canonical representation in $[\mathcal{A}]$, where $\mathbf{u}_{J_2} < \max\{\mathbf{u}_{\eta_{i,j}A_j}\}$ in (5). That is

$$J_2 \cdot N \cdot r = \sum_{ij} \bar{g}_{i,j}\rho_{i,j}A_j, \tag{6}$$

where, $\mathbf{u}\rho_{i,j}A_j$ are pairwise different. If $\max\{\mathbf{u}_{\rho_{i,j}A_j}\}$ in (6) is lower in ordering than $\max\{\mathbf{u}_{\eta_{i,j}A_j}\}$ in (5), we have already reduced the highest ordering of $\mathbf{u}_{\eta_{i,j}A_j}$ in (5). Otherwise, assume $\mathbf{u}_{\rho_a A_b} = \max\{\mathbf{u}_{\rho_{i,j}A_j}\}$ and $A_b = I_b \cdot \mathbf{u}_{A_b}^{d_b} + R_b$. Substituting $\mathbf{u}_{\rho_a A_b}^{d_b}$ by $-\frac{\rho_a R_b}{\rho_a I_b}$ in (6), the left side keeps unchanged since $\mathbf{u}_{J_2} < \mathbf{u}_{\rho_a A_b}$, N is free of $\mathbf{u}_{\rho_a A_b}$ and $\deg(r, \mathbf{u}_{\rho_a A_b}) < \deg(\rho_a A_b, \mathbf{u}_{\rho_a A_b})$. In the right side, the $\rho_a A_b$ becomes zero, i.e. $\max\{\mathbf{u}_{\rho_{i,j}A_j}\}$ decreases. Clearing denominators of the substituted formula of (6), we obtain a new equation:

$$(\rho_a I_b)^t \cdot J_2 \cdot N \cdot r = \sum f_{ij}\tau_{i,j}A_j. \tag{7}$$

Note that in the right side of (7), the highest ordering of $\tau_{i,j}A_j$ is less than $\mathbf{u}_{\rho_a A_b}$ and $(\rho_a I_b)^t \cdot J_2$ is invertible w.r.t. $\mathcal{A}$. Then after multiplying a polynomial which is invertible w.r.t. $\mathcal{A}$ and can be represented as a linear combination of $\tau_{i,j}A_j$ all of which is strictly lower than $\mathbf{u}_{\rho_a A_b}$. Repeating the above process, we can obtain a nonzero $\bar{N}$, such that $\bar{N} \cdot r = 0$. Then $r = 0$. By Lemma 3, $\mathcal{A}$ is the characteristic set of $\mathbf{sat}(\mathcal{A})$. ∎

The following is one of the main results in this paper.

Theorem 20. *A chain $\mathcal{A}$ is the characteristic set of $\mathbf{sat}(\mathcal{A})$ iff $\mathcal{A}$ is coherent and difference regular.*

Proof: If $\mathcal{A}$ is coherent and difference regular, then by Lemma 19, any polynomial in $\mathbf{sat}(\mathcal{A})$ which is difference reduced w.r.t. $\mathcal{A}$ is zero. So $\mathcal{A}$ is a characteristic set of $\mathbf{sat}(\mathcal{A})$. Conversely, let $\mathcal{A} = A_1, A_2, \ldots, A_l$ be a characteristic set of the saturation ideal $\mathbf{sat}(\mathcal{A})$ and $I_i = \mathrm{init}(A_i)$. For any $1 \leq i < j \leq l$, let $r = \mathrm{rprem}(\Delta_{i,j}, \mathcal{A})$ as in Definition 11. Then r is in $\mathbf{sat}(\mathcal{A})$ and is difference reduced w.r.t. $\mathcal{A}$. Since $\mathcal{A}$ is the characteristic set of $\mathbf{sat}(\mathcal{A})$, $r = 0$. Then $\mathcal{A}$ is coherent. To prove that $\mathcal{A}$ is regular, for any $0 \leq i \leq l$, $\eta \in T_\sigma$ we need to prove that $f = \eta I_i$ is invertible w.r.t. $\mathcal{A}$. Assume this is not true. By definition, f is not invertible w.r.t. $\mathcal{A}_f$ when they are treated as algebraic equations. By Lemma 18, $\mathcal{A}_f$ is a regular algebraic triangular set. By Lemma 7, there is a $g \neq 0$ which is reduced w.r.t. $\mathcal{A}_g$ (and hence $\mathcal{A}$) such that $f \cdot g \in (A_f) \subset [A]$. Since $f = \eta I_i \in \mathbb{I}_{\mathcal{A}}$, $g \in \mathbf{sat}(\mathcal{A})$ and g is reduced w.r.t. $\mathcal{A}$. Since $\mathcal{A}$ is the characteristic set of $\mathbf{sat}(\mathcal{A})$, we have $g = 0$, a contradiction. Hence, $f = \eta I_i$ is invertible w.r.t. $\mathcal{A}$ and $\mathcal{A}$ is difference regular. ∎

Theorem 21. *If $\mathcal{A}$ is a coherent and difference regular chain of form (1),then*

$$\mathbf{sat}(\mathcal{A}) = \cup_{f \in \mathcal{K}\{\mathbb{X}\}} \mathbf{a\text{-}sat}(\mathcal{A}_f).$$

Proof: It is easy to see that $\mathbf{sat}(\mathcal{A}) \supset \bigcup_{f \in \mathcal{K}\{\mathbb{X}\}} \mathbf{a\text{-}sat}(\mathcal{A}_f)$. If $f \in \mathbf{sat}(\mathcal{A})$, since $\mathcal{A}$ is coherent and difference regular chain, and $\mathcal{A}$ is the characteristic set of $\mathbf{sat}(\mathcal{A})$, we have $\mathrm{rprem}(f, \mathcal{A}) = 0$, or $\mathrm{prem}(f, \mathcal{A}_f) = 0$, that is $f \in \mathbf{a\text{-}sat}(\mathcal{A}_f)$. Hence $\mathbf{sat}(\mathcal{A}) \subset \bigcup_{f \in \mathcal{K}\{\mathbb{X}\}} \mathbf{a\text{-}sat}(\mathcal{A}_f)$. ∎

Note that we cannot check whether a chain is regular directly due to the reason that T_σ contains an infinite number of elements. To give a complete zero decomposition algorithm like the one in [13,15], we need to define a type of chains such that we have a constructive criterion to check whether it is regular and the chain $\mathcal{A}$ is non-trivial in the sense that $\mathrm{Zero}(\mathbf{a\text{-}sat}(\mathcal{A})) \neq \emptyset$. These problems are the major open ones for the characteristic set method of partial difference polynomial systems.

5 Characteristic Set of Reflexive Prime Difference Ideals

In the algebraic and differential cases, prime ideals can be described by irreducible chains. In this section, we will extend this result to the partial difference case. In order to do that, we need to introduce the concept of strongly irreducible chains.

A chain $\mathcal{A}$ is called *strongly irreducible* if

- $\mathcal{A}_f$ is an irreducible algebraic triangular set for any $f \in \mathcal{K}\{\mathbb{X}\}$, and
- For $\eta \in T_\sigma$ and $h \in \mathcal{K}\{\mathbb{X}\}$, if $\eta h \in \mathbf{a\text{-}sat}(\mathcal{A}_f)$ then $h \in \mathbf{a\text{-}sat}(\mathcal{A}_f)$.

Theorem 22. *Let $\mathcal{A}$ be a coherent and strongly irreducible difference chain. Then $\mathbf{sat}(\mathcal{A})$ is a reflexive prime difference ideal.*

Proof: Let f, g be two r-pols such that $fg \in \mathbf{sat}(\mathcal{A})$. By Lemma 21, there exists a polynomial h such that $fg \in D = \mathbf{a\text{-}sat}(\mathcal{A}_h)$. Since $\mathcal{A}$ is strongly irreducible, $\mathcal{A}_h$ is an irreducible algebraic triangular set and hence D is a prime ideal. We thus have $f \in D$ or $g \in D$. In other words, $f \in \mathbf{sat}(\mathcal{A})$ or $g \in \mathbf{sat}(\mathcal{A})$. Hence, $\mathbf{sat}(\mathcal{A})$ is a prime ideal. We still need to show that $\mathbf{sat}(\mathcal{A})$ is reflexive. If $\sigma_i f \in \mathbf{sat}(\mathcal{A})$ then $\exists\ h \in \mathcal{K}\{\mathbb{X}\}$, $\sigma_i f \in \mathbf{a\text{-}sat}(\mathcal{A}_h)$. $f \in \mathbf{a\text{-}sat}(\mathcal{A}_h)$ according to the definition of strongly irreducible chain. Then $f \in \mathbf{sat}(\mathcal{A})$. ∎

To prove that the characteristic set of any prime ideal is strongly irreducible, we need the following lemmas.

Lemma 23. *Let $\mathcal{I}$ be a prime difference ideal, $\mathcal{A}$ its characteristic set. Then $\mathcal{I} = \mathbf{sat}(\mathcal{A})$.*

Proof: It is clear that $\mathcal{I} \subset \mathbf{sat}(\mathcal{A})$. Let $f \in \mathbf{sat}(\mathcal{A})$. Then there is a $J \in \mathbb{I}_\mathcal{A}$ such that $Jf \in [A] \subset \mathcal{I}$. By Theorem 20, J is invertible w.r.t. $\mathcal{A}$. Hence J is not in $\mathcal{I}$ by Lemma 16. Since $\mathcal{I}$ is a prime ideal, $f \in \mathcal{I}$. ∎

Lemma 24. *Let $\mathcal{I}$ be a reflexive prime difference ideal, $\mathcal{A}$ its characteristic set. Then $\forall\ h \in f \in \mathcal{K}\{\mathbb{X}\}$, $\mathcal{A}_h$ is algebraic irreducible.*

Proof: Otherwise, there exists an $h \in f \in \mathcal{K}\{\mathbb{X}\}$, such that $\mathcal{A}_h$ is a reducible algebraic triangular set. By definition, there exist polynomials f and g which are reduced w.r.t. $\mathcal{A}_h$ such that $fg \in \mathcal{A}_h \subset \mathbf{sat}(\mathcal{A}) = \mathcal{I}$. From this, we have $f \in \mathcal{I}$ or $g \in \mathcal{I}$, which is impossible since f and g are reduced w.r.t. $\mathcal{A}$. ∎

Theorem 25. *Let $\mathcal{I}$ be a reflexive prime difference ideal, $\mathcal{A}$ a characteristic set of $\mathcal{I}$. Then $\mathcal{A}$ is coherent, strongly irreducible, and $\mathcal{I} = \mathbf{sat}(\mathcal{A})$.*

Proof: By Lemma 23, for any characteristic set $\mathcal{A}$ of $\mathcal{I}$, we have $\mathcal{I} = \mathbf{sat}(\mathcal{A})$. By Theorem 20, $\mathcal{A}$ is coherent. By Lemma 24, we have for any $h \in \mathcal{K}\{\mathbb{X}\}$, $\mathcal{A}_h$ is algebraic irreducible. Also, if $\sigma_i g \in \mathbf{a\text{-}sat}(\mathcal{A}_h)$, then $\sigma_i g \in \mathcal{I}$. Since $\mathcal{I}$ is reflexive, $g \in \mathcal{I}$. Then $g \in \mathbf{a\text{-}sat}(\mathcal{A}_h)$. ∎

The following example shows that it is difficult to decide whether a chain is strongly irreducible. Even in the the ordinary case, deciding whether a chain is strongly irreducible is a major difficult problem in difference algebra.

Example 2. [10] Let $\mathcal{K} = Q(t)$. The transforming operators over $\mathcal{K}$ is σ such that $\sigma t = (t+1)$. $\mathcal{A} \subseteq \mathcal{K}\{x_1, x_2\}$ and $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2\}$ where $A_1 = x_1^2 + t, A_2 = x_2^2 + t + k$. If $k > 1$, $A_2 - \sigma^k A_1 = (x_2 - \sigma^k x_1)(x_2 + \sigma^k x_1)$, $x_2 - \sigma^k x_1 \notin \mathbf{sat}(\mathcal{A})$, $x_2 + \sigma^k x_1 \notin \mathbf{sat}(\mathcal{A})$. $\mathbf{sat}(\mathcal{A})$ is not a prime difference ideal, and $\mathcal{A}$ is not strongly irreducible.

6 Algorithms of Zero Decomposition

In this section, we will present an algorithm which can be used to decompose the zero set of a general polynomial set into the zero sets of coherent chains.

Lemma 26. *Let $\mathbb{P}$ be a finite set of polynomials, $\mathcal{A} = A_1, \ldots, A_m$ a characteristic set of $\mathbb{P}$, $I_i = \mathrm{init}(A_i)$, and $J = \prod_{i=1}^m I_i$. If $\mathrm{prem}(P, \mathcal{A}) = 0$ for all $P \in \mathbb{P}$, then*

$$\mathrm{Zero}(\mathbb{P}) = \mathrm{Zero}(\mathcal{A}/J) \bigcup \cup_{i=1}^m \mathrm{Zero}(\mathbb{P} \cup \{I_i\})$$
$$\mathrm{Zero}(\mathbb{P}) = \mathrm{Zero}(\mathbf{sat}(\mathcal{A})) \bigcup \cup_{i=1}^m \mathrm{Zero}(\mathbb{P} \cup \{I_i\})$$

Proof: This is direct consequence of the remainder formula (4). ∎

Now, we can give the *zero decomposition theorem.*

Theorem 27. *Let $\mathbb{P}$ be a finite set of polynomials in $\mathcal{K}\{y_1, \ldots, y_n\}$, then we can obtain a sequence of coherent chains $\mathcal{A}_i$, $i = 1, \ldots, k$ such that*

$$\mathrm{Zero}(\mathbb{P}) = \bigcup_{i=1}^{k} \mathrm{Zero}(\mathcal{A}_i / I_{\mathcal{A}_i}) = \bigcup_{i=1}^{k} \mathrm{Zero}(\mathbf{sat}(\mathcal{A}_i)) \tag{8}$$

We first give the following algorithm to find the decomposition.

Algorithm 28. ZDT($\mathbb{P}$)

- **Input:** *a finite set* $\mathbb{P}$ *of polynomials.*
- **Output:** $W = \{\mathcal{A}_1, \ldots, \mathcal{A}_k\}$ *s.t.* $\mathcal{A}_i$ *is coherent and* $\mathrm{Zero}(\mathbb{P}) = \bigcup_{i=1}^{k} \mathrm{Zero}(\mathbf{sat}(\mathcal{A}_i))$.

Begin
  $\mathcal{B} = CS(\mathbb{P})$ *//This gives the characteristic set of* $\mathbb{P}$.
  If $\mathcal{B} = 1$ *then* $W = \{\}$
  Else
    $\mathbb{R} = \{\mathrm{prem}(f, \mathcal{B}) \neq 0 \mid f \in (\mathbb{P} \setminus \mathcal{B}) \cup \triangle(\mathcal{B})\}$
    If $\mathbb{R} = \emptyset$ *then* $W = \{\mathcal{B}\} \cup \cup_i \mathbf{ZDT}(\mathbb{P} \cup \{I_i\})$
    Else $W = \mathbf{ZDT}(\mathbb{P} \cup \mathbb{R})$
      where I_j *are the initials of the polynomials in* $\mathcal{B}$.
End.

Proof of Correctness of Algorithm 28. If $\mathbb{R} = \emptyset$, by Lemma 26, we obtain a chain. Since I_j is reduced w.r.t. $\mathcal{B}$, by Lemma 2, the characteristic set of $\mathbb{P} \cup \{I_i\}$ is of lower ordering than that of $\mathcal{B}$. Similarly, the characteristic set of $\mathbb{P} \cup \mathbb{R}$ is of lower ordering than that of $\mathcal{B}$. By Lemma 1, the algorithm will end and give the decomposition. ∎

Example 3. Let $\mathcal{A} = \{A_1, A_2\}$ where $A_1 = \sigma_1^2 y_3^2 + \sigma_2 y_2$, $A_2 = \sigma_2 y_3 + \sigma_1^2 y_1$. $\mathcal{A}$ is not coherent, since the remainder $A_3 = \mathrm{rprem}(\sigma_2 A_1, \mathcal{A}) = \sigma_2^2 y_2 + \sigma_1^4 y_1^2$ is reduced w.r.t. $\mathcal{A}$. If we do the zero decomposition for $\mathcal{A}$ we obtain $\{A_3, A_1, A_2\}$ which is a coherent chain.

7 Conclusion

In this paper, we extend some of the main properties of chains to the partial difference polynomial systems. We prove that a partial difference chain is the characteristic set of its saturation ideal if and only if it is coherent and regular. We also prove that a partial difference ascending chain is the characteristic set of a reflexive prime ideal if and only if it is strongly irreducible. Finally, we give the zero decomposition algorithm.

Comparing to the algebraic, differential, and ordinary difference cases, there still exist major problems unsolved in the partial difference case. These include to give a constructive criterion for a chain to be regular and non-trivial and to solve the perfect ideal membership problem.

References

1. Aubry, P.: Ensembles Triangulaires de polynômes et Résolution de Systèmes Algébriques, Implantation en Axiom. Thèse de l'université Pierre et Marie Curie (1999)
2. Aubry, P., Lazard, D., Moreno Maza, M.: On the Theory of Triangular Sets. Journal of Symbolic Computation 28, 105–124 (1999)

3. Bentsen, I.: The Existence of Solutions of Abstract Partial Difference Polynomial. Trans. of AMS 158, 373–397 (1971)
4. Boulier, F., Lazard, D., Ollivier, F., Petitot, M.: Representation for the Radical of a Finitely Generated Differential Ideal. In: Proc. of ISSAC 1995, pp. 158–166. ACM Press, New York (1995)
5. Boulier, F., Lemaire, F., Moreno Maza, M.: Well Known Theorems on Triangular Systems and the D5 Principle. In: Proc. of Transgressive Computing 2006, pp. 79–91 (2006)
6. Bouziane, D., Kandri Rody, A., Mârouf, H.: Unmixed-dimensional Decomposition of a Finitely Generated Perfect Differential Ideal. Journal of Symbolic Computation 31, 631–649 (2001)
7. Cheng, J.S., Gao, X.S., Yap, C.K.: Complete Numerical Isolation of Real Zeros in Zero-dimensional Triangular Systems. In: Prof. ISSAC 2007, pp. 92–99. ACM Press, New York (2007)
8. Chou, S.C.: Mechanical Geometry Theorem Proving. Kluwer Academic Publishers, Norwell (1987)
9. Chou, S.C., Gao, X.S.: Ritt-Wu's Decomposition Algorithm and Geometry Theorem Proving. In: Stickel, M.E. (ed.) CADE 1990. LNCS, vol. 449, pp. 207–220. Springer, Heidelberg (1990)
10. Cohn, R.M.: Difference Algebra. Interscience Publishers (1965)
11. Dahan, X., Moreno Maza, M., Schost, E., Wu, W., Xie, Y.: Lifting Techniques for Triangular Decompositions. In: Proc. ISSAC 2005, pp. 108–115. ACM Press, New York (2005)
12. Gao, X.S., Chou, S.C.: A Zero Structure Theorem for Differential Parametric Systems. Journal of Symbolic Computation 16, 585–595 (1994)
13. Gao, X.S., Luo, Y.: A Characteristic Set Method for Difference Polynomial Systems. In: International Conference on Polynomial System Solving, November 24-26 (2004); Submitted to JSC
14. Gao, X.S., Luo, Y., Zhang, G.: A Characteristic Set Method For Ordinary Difference Polynomial Systems. MM-Preprints 25, 84–102 (2006)
15. Gao, X.S., van der Hoeven, J., Yuan, C., Zhang, G.: A Characteristic Set Method for Differential-Difference Polynomial Systems. In: MEGA 2007, Strobl, Austria (July 2007)
16. Hubert, E.: Factorization-free Decomposition Algorithms in Differential Algebra. Journal Symbolic Computation 29, 641–662 (2000)
17. Kolchin, E.: Differential Algebra and Algebraic Groups. Academic Press, New York (1973)
18. Kondratieva, M.V., Levin, A.B., Mikhalev, A.V., Pankratiev, E.V.: Differential and Difference Dimension Polynomials. Kluwer Academic Publishers, Dordrecht (1999)
19. Ritt, J.F.: Differential Algebra, Amer. Math. Soc. Colloquium (1950)
20. Ritt, J.F., Raudenbush Jr., H.W.: Ideal Theory and Algebraic Difference Equations. Trans. of AMS 46, 445–452 (1939)
21. Rosenfeld, A.: Specialization in Differential Algebra. Trans. Am. Math. Soc 90, 394–407 (1959)
22. Wang, D.: Elimination Methods. Springer, Berlin (2000)
23. Wu, W.T.: On the Decision Problem and the Mechanization of Theorem in Elementary Geometry. Scientia Sinica 21, 159–172 (1978)
24. Wu, W.T.: A Constructive Theorey of Differential Algebraic Geometry. Lect. Notes in Math, vol. 1255, pp. 173–189. Springer, Heidelberg (1987)

25. Wu, W.T.: Basic Principle of Mechanical Theorem Proving in Geometries (in Chinese). Science Press, Beijing (1984); English Edition. Springer, Wien (1994)
26. van der Hoeven, J.: Differential and Mixed Differential-Difference Equations from the Effective Viewpoint (preprints, 1996)
27. Yang, L., Zhang, J.Z., Hou, X.R.: Non-linear Algebraic Equations and Automated Theorem Proving (in Chinese). ShangHai Science and Education Pub., ShangHai (1996)

Some Mathematical Problems in Cryptanalysis

Xiaoyun Wang[1,2]

[1] Tsinghua University, Beijing, China
xywang@sdu.edu.cn
[2] Shandong University, Jinan, China

Abstract. The talk will recall some basic mathematical problems on analyzing both symmetric and asymmetric ciphers. A special focus will be on the advances in asymmetric ciphers based on the research of some fundamentally hard mathematical problems. Some fundamental statistical problems and cryptanalysis techniques for symmetric cipher will be reviewed as well.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, p. 322, 2008.
© Springer-Verlag Berlin Heidelberg 2008

A Reduction Attack on Algebraic Surface Public-Key Cryptosystems

Maki Iwami

Osaka University of Economics and Law, Japan
maki@keiho-u.ac.jp

Abstract. An algebraic surface public-key cryptosystem was developed by Akiyama and Goto. Its security is based on a decision randomizing polynomial problem which is related to a problem of finding sections on fibered algebraic surfaces which can be reduced to solving a multivariate equation system known to be NP-complete. In the case that the defining equation of a surface used for public-key is in a certain form, Uchiyama and Tokunaga succeeded in attacking in the sense of getting plain texts from corresponding ciphertexts using reductions efficiently without solving section finding problem. In this paper, two algorithms applicable to all cases are suggested. One is the generalization of Uchiyama-Tokunaga's attack from polynomial ring over $\mathbb{F}_p$ to polynomial ring over rational function field, and the other takes advantages of Gröbner base techniques so as to deal with in the polynomial ring over $\mathbb{F}_p$. [1]

1 Introduction

Public key cryptography is widely used because it enables secure communication with a party accessing the site for the first time. But the current one is time consuming, uses a lot of electricity so unsuitable for mobile applications, and will be decrypted when quantum computers become available. To address these issues, a Public-key Cryptosystem using Algebraic Surfaces was developed by Akiyama and Goto, and opened to general public at website as a research news of Toshiba corporation in 2005 [2], and some papers are published [1, 3, 4]. Its brief survey is given in **section2**. Its security is based on a decision randomizing polynomial problem which is related to a problem of finding sections on fibered algebraic surfaces. This problem can be reduced to solving a multivariate equation system and it is known to be NP-complete. In 2007, in the case that the defining equation of a surface used for public-key is in a certain form, Uchiyama and Tokunaga successed to attack in the sense of getting plain text from cipher text using reductions efficiently without solving section finding problem [5]. And the abstract is introduced in CRYPTREC report 2006 [6] in the Appendix Chapter, which is known to be the important report related to e-Government recommended ciphers in Japan. Its brief survey is given in **section3**. Note that, at this point, the cryptosystem can be used safely if only we

[1] A part of this work was supported by JSPS. Grant-in-Aid for Scientific Research.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 323–332, 2008.
© Springer-Verlag Berlin Heidelberg 2008

avoid to adopt using public key with vulnerability in the key generation step. Therefore, in this paper, two algorithms applicable to all cases are suggested in **section4**, i.e. the cryptosystem is completely broken. One is the generalization of Uchiyama-Tokunaga's attack from polynomial ring over $\mathbb{F}_p$ to polynomial ring over rational function field (which is written in [8] without proofs). And the other takes advantages of Gröbner base techniques so as to deal with in the polynomial ring over $\mathbb{F}_p$.

2 Algebraic Surface Public-Key Cryptosystem [1, 3, 4]

[**key generation**]
Secret key: Choose two distinct curves of the form $D_1 : (x, y, t) = (u_x(t), u_y(t), t)$, $D_2 : (x, y, t) = (v_x(t), v_y(t), t)$ satisfying $\deg u_x(t) \neq \deg v_x(t)$ or $\deg u_y(t) \neq \deg v_y(t)$ for the uniqueness of decryption, and satisfying $(u_x(t) - v_x(t))|(u_y(t) - v_y(t))$ for $c_{10}(t) \in \mathbb{F}_p[t]$ (not $\mathbb{F}_p(t)$) in step (a) in Pulic-key generation.

Public key:
(a) Construct algebraic surface (public key) $X(x, y, t) = \sum_{i,j} c_{ij}(t)x^i y^j = 0$ over $\mathbb{F}_p$ containing two curves (secret key), i.e. it satisfy $X(u_x(t), u_y(t), t) = X(v_x(t), v_y(t), t) = 0$. First, ramdomly choose $c_{ij}(t)$ with $(i, j) \neq (0, 0), (1, 0)$ and then calculate $c_{10}(t)$ and $c_{00}(t) \in \mathbb{F}_p(t)$ as follows.
$c_{10}(t) := -\sum_{(i,j)\neq(0,0),(1,0)} c_{i,j}(t)\{u_x(t)^i u_y(t)^j - v_x(t)^i v_y(t)^j\}/(u_x(t) - v_x(t))$
$c_{00}(t) := -\sum_{(i,j)\neq(0,0)} c_{i,j}(t)u_x(t)^i u_y(t)^j$.

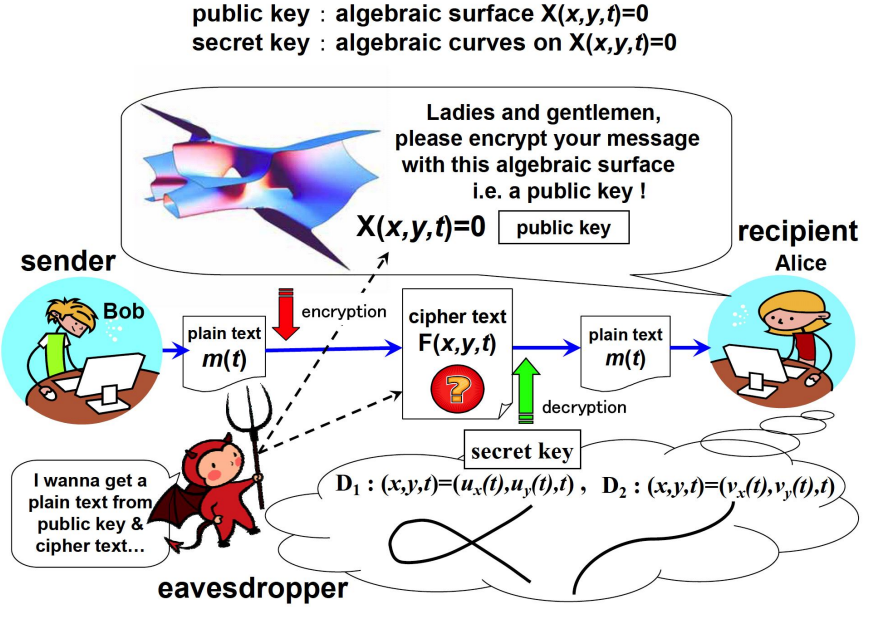


Fig. 1. Akiyama-Goto Algebraic Surface Pulic-Key Cryptosystems

(b) Choose $\ell \in \mathbb{N}$ as a lower bound for the degree of a monic irreducible polynomial $f(t) \in \mathbb{F}_p[t]$ chosen in the encryption step. For reasons of security (see 5.3 in [4]), we impose $\deg_t X(x,y,t) < \ell$.
(c) Choose $d \in \mathbb{N}$ satisfying $d \geq \max\{\deg u_x(t), \deg u_y(t), \deg v_x(t), \deg v_y(t)\}$.

By taking a large ℓ or d, the characteristic p of the ground field can be chosen as small as possible (e.g. at most 4 bits). The estimation of the key size are discussed in section 7 in [4].

[**encryption**] Let m be a plain text, and divide m into small blocks as $m = m_0||m_1||\cdots||m_{\ell-1}$ where each m_i is chosen $0 \leq m_i \leq p-1$.

1. Embed m into a plain text polynomial as $m(t) = m_{\ell-1}t^{\ell-1} + \cdots + m_1 t + m_0$
2. Choose a random polynomial $s(x,y,t)$ containing a term $x^\alpha y^\beta$ with $\alpha > \deg_x X(x,y,t)$ and $\beta > \deg_y X(x,y,t)$ and satisfying $(\deg_x s(x,y,t) + \deg_y s(x,y,t))d + \deg_t s(x,y,t) < \ell$. (This implies $\deg(s(u_x(t),u_y(t),t) - s(v_x(t),v_y(t),t)) < \ell$, therefore we can extract $f(t)$ in the decryption step.)
3. Choose a random polynomial $r(x,y,t)$ satisfying $\deg_t r(x,y,t) < \ell$, and a random monic irreducible polynomial $f(t)$ with $\deg f(t) \geqq \ell$
4. Compute the cipher polynomial $F(x,y,t) = m(t) + f(t)s(x,y,t) + X(x,y,t)\, r(x,y,t)$.

[**decryption**] As D_1, D_2 are on X, $X(u_x(t),u_y(t),t) = X(v_x(t),v_y(t),t) = 0$.

1. Substitute sections D_1 and D_2 into $F(x,y,t)$:
   $h_1(t) = F(u_x(t),u_y(t),t) = m(t) + f(t)s(u_x(t),u_y(t),t)$
   $h_2(t) = F(v_x(t),v_y(t),t) = m(t) + f(t)s(v_x(t),v_y(t),t)$
2. Compute $h_1(t) - h_2(t) (= f(t)\{s(u_x(t),u_y(t),t) - s(v_x(t),v_y(t),t))\}$.
3. Factorize and find $f(t)$ as a monic irreducible polynomial with maximum degree.
4. Compute $m(t)$ by reducing $h_1(t)$ by $f(t)$. ($\deg m(t) < \deg f(t)$.)
5. Extract m from $m(t)$.

Example 1 (key generation and encryption). Now we consider in $\mathbb{F}_2$.
[Secret key] We define secret keys as two distinct curves as
$D_1 : (u_x(t),u_y(t),t) = (t^2+t, t^3+t^2+t+1, t)$, $D_2 : (v_x(t),v_y(t),t) = (1+t, 1+t^2, t)$.
[Public key] As one example, we obtain the following algebraic surface.
$X_B(x,y,t) := t^2 + t^8 + t^{12} + t^{14} + t^{21} + t^{22} + t^{23} + t^{24} + t^{26} + t^{27} + t^{28} + t^{29} + y + t^2y + y^2 + t^3y^2 + y^3 + t^2y^3 + t^3y^3 + ty^4 + t^2y^4 + t^4y^4 + t^5y^4 + y^5 + t^4y^5 + x(t + t^3 + t^5 + t^{10} + t^{11} + t^{16} + t^{17} + t^{19} + t^{21} + t^{23} + t^{26} + t^{28} + t^3y + t^4y + y^2 + ty^2 + t^3y^2 + y^3 + ty^3 + t^3y^3 + t^5y^3 + y^4 + ty^4 + t^2y^4 + t^3y^4 + t^4y^4 + y^5 + t^2y^5) + x^4(1 + t^2 + t^4 + t^5 + t^3y + t^4y + t^5y + y^2 + ty^2 + t^2y^2 + t^3y^2 + t^4y^2 + ty^3 + t^2y^3 + t^4y^3 + y^4 + t^2y^4 + t^3y^4 + t^5y^4 + y^5 + t^4y^5) + x^3(t + t^3 + t^4 + t^5 + y + ty + t^3y + t^5y + t^2y^2 + t^4y^2 + y^3 + t^2y^3 + t^3y^3 + t^5y^3 + t^2y^4 + t^3y^4 + y^5 + t^5y^5) + x^2(t + t^5 + t^2y + t^4y + y^2 + ty^2 + ty^3 + t^2y^3 + t^3y^3 + t^4y^3 + t^5y^3 + y^4 + ty^4 + t^3y^4 + t^4y^4 + y^5 + ty^5 + t^5y^5) + x^5(1 + t^3 + ty + t^2y + t^3y + t^4y + t^4y^2 + t^5y^2 + ty^3 + t^2y^3 + t^4y^3 + y^4 + t^2y^4 + t^4y^4 + ty^5 + t^2y^5 + t^5y^5)$
[encryption] $m(t) := 1 + t + t^2 + t^3 + t^4 + t^6 + t^8 + t^9 + t^{14} + t^{17} + t^{19} + t^{20} + t^{23} + t^{26} + t^{28} + t^{29} + t^{30} + t^{32} + t^{34} + t^{35} + t^{36} + t^{37} + t^{39}$,
$f(t) := 1+t+t^2+t^4+t^7+t^9+t^{10}+t^{11}+t^{14}+t^{17}+t^{22}+t^{23}+t^{25}+t^{26}+t^{27}+t^{28}+t^{32}+$

$t^{34}+t^{36}+t^{38}+t^{40}$, $s(x,y,t) := t+t^3+x^3+y^2+x^6y^6$, $r(x,y,t) := 1+t^3+t^4+xy+y^2$. Then the cipher polynomial $F_B(x,y,t) := t^2+t^8+t^{12}+t^{14}+t^{21}+t^{22}+t^{23}+t^{24}+t^{26}+t^{27}+t^{28}+t^{29}+y+t^2y+y^2+t^3y^2+y^3+t^2y^3+t^3y^3+ty^4+t^2y^4+t^4y^4+t^5y^4+y^5+t^4y^5+x(t+t^3+t^5+t^{10}+t^{11}+t^{16}+t^{17}+t^{19}+t^{21}+t^{23}+t^{26}+t^{28}+t^3y+t^4y+y^2+ty^2+t^3y^2+y^3+ty^3+t^3y^3+t^5y^3+y^4+ty^4+t^2y^4+t^3y^4+t^4y^4+y^5+t^2y^5)+x^4(1+t^2+t^4+t^5+t^3y+t^4y+t^5y+y^2+ty^2+t^2y^2+t^3y^2+t^4y^2+ty^3+t^2y^3+t^4y^3+y^4+t^2y^4+t^3y^4+t^5y^4+y^5+t^4y^5)+x^3(t+t^3+t^4+t^5+y+ty+t^3y+t^5y+t^2y^2+t^4y^2+y^3+t^2y^3+t^3y^3+t^5y^3+t^2y^4+t^3y^4+y^5+t^5y^5)+x^2(t+t^5+t^2y+t^4y+y^2+ty^2+ty^3+t^2y^3+t^3y^3+t^4y^3+t^5y^3+y^4+ty^4+t^3y^4+t^4y^4+y^5+ty^5+t^5y^5)+x^5(1+t^3+ty+t^2y+t^3y+t^4y+t^4y^2+t^5y^2+ty^3+t^2y^3+t^4y^3+y^4+t^2y^4+t^4y^4+ty^5+t^2y^5+t^5y^5)$.

3 Attack on Algebraic Surface Public-Key Cryptosystem under the Assumption [5]

Assumption 1. For the defining equation of the algebraic surface X which will be used as the public key, the leading term is in the form as $\mathrm{LT}(X)cx^\alpha y^\beta$ where $c \in \mathbb{F}_p$ and $(\alpha,\beta) \neq (0,0)$ w.r.t. a monomial order $\hat{R}$.

Algorithm 1 (Uchiyama-Tokunaga's attack).
Input : Akiyama-Goto's public key $X(x,y,t) \in \mathbb{F}_p[x,y,t]$ satisfying **Assumption1**, cipher polynomial $F(x,y,t) \in \mathbb{F}_p[x,y,t]$.
Ountput : Plaintext m which corresponds to the cipher polynomial $F(x,y,t)$.

1. Calculate normal form $R_1(x,y,t)$ of the reduction of $F(x,y,t)$ by $X(x,y,t)$.
2. Among the temrs of R_1, randomly choose the term satisfying x^iy^j $((i,j) \neq (0,0))$, and its coefficient not being in $\mathbb{F}_p$, then let its coefficient be C.
3. Calculate factors in $\mathbb{F}_p[t]$ of C, and let the set consisting of irreducible fators whose degree is greater than or equal to ℓ be $\hat{G}$. Choose the element $g \in \hat{G}$ and the normal form n of R_1 becomes an element in $\mathbb{F}_p[t]$.($g(t)$ is $f(t)$)
4. Compute a polynomial $n(t) = n_{k-1}t^{k-1} + \cdots + n_1t + n_0 \in \mathbb{F}_p[t]$, outputs $m = n_0||n_1||\cdots||n_{k-1}$ and end. ($n(t)$ is $m(t)$)

If $c_{\alpha\beta}(t)$ of $\mathrm{LT}(X) = c_{\alpha\beta}(t)x^\alpha y^\beta$ is not constant then the normal form of $F(x,y,t)$ by $X(x,y,t)$ does not necessarily stop in the form as $F(x,y,t) = m(t)+f(t)R_2(x,y,t)+X(x,y,t)(f(t)G_2(x,y,t)+q(x,y,t)) \underset{X}{\longrightarrow} m(t)+f(t)R_2(x,y,t)$, i.e. some terms might be reduced and disappear, so we might fail to detect $f(t)$.

Example 2 (unworkable case : Uchiyama-Tokunaga's attack to Example 1). The normal form of the reduction of $F_B(x,y,t)$ by $X_B(x,y,t)$ is as follows. Note that denominators are caused by $t(1+t+t^4)$ from $\mathrm{LT}(X_B) = t(1+t+t^4)x^5y^5$.
$R_B(x,y,t) := (1+t^3+t^7+\cdots+t^{44}y^{16}+t^{46}y^{16}+t^{48}y^{16}+x^2(t^4y^6+t^8y^6+t^{12}y^6+\cdots+t^{42}y^{16}+t^{45}y^{16}+t^{49}y^{16})+x(ty^6+t^4y^6+t^5y^6+\cdots+t^{45}y^{16}+t^{47}y^{16}+t^{49}y^{16})+x^4(t^6+t^4y^6+t^6y^6+\cdots+t^{46}y^{16}+t^47y^{16}+t^{50y^{16}})+x^3(1+t+t^2+\cdots+t^{48}y^{16}+t^{49}y^{16}+t^{50}y^{16}))/(1+t^3+ty+t^2y+t^3y+t^4y+t^4y^2+t^5y^2+ty^3+t^2y^3+t^4y^3+y^4+t^2y^4+t^4y^4+ty^5+t^2y^5+t^5y^5)^2$. Even if we focus attention on numerator, GCD of $c_{ij}(t)$ except $c_{00}(t)$ is 1 i.e. we cannot detect $f(t)$. In the

reduction process, leading coefficient $t(1+t+t^4)$ of $\mathrm{LT}(X_B)$ result in breaking such an important form $m(t)+f(t)R_2$, therefore even if we focus attention on $c_{ij}(ij \neq (0,0))$ (i.e. coefficients of x, y), $f(t)$ cannot be detected.

4 Attack on Algebraic Surface Public-Key Cryptosystem

In this section, two algorithms applicable to all cases are suggested. The idea is to transform $X(x,y,t)$ into monic w.r.t. x and y, because t in $\mathrm{LT}(X)$ drives detecting $f(t)$ failure in the reduction process. First one is **Algorithm 2** which we generalize the Uchiyama-Tokunaga's attack in a straightforward way, working in the polynomial ring w.r.t. x, y over rational function field w.r.t. t. We denote it by $\mathbb{F}_p(t)[x,y]$. Second one is **Algorithm 3** which we use Gröbner base techniques, working in the polynomial ring $\mathbb{F}_p[x,y,t,\mathcal{A}]$ introducing a new parameter $\mathcal{A}$. We can say that two algorithms are essentially the same.

4.1 Attack by Straightforward Generalization in $\mathbb{F}_p(t)[x,y]$

Algorithm 2. [straightforward generalization in $\mathbb{F}_p(t)[x,y]$]
Input : Akiyama-Goto's public key $X(x,y,t) \in \mathbb{F}_p[x,y,t]$,
cipher polynomial $F(x,y,t) \in \mathbb{F}_p[x,y,t]$.
Ountput : Plaintext m which corresponds to the cipher polynomial $F(x,y,t)$.

0. Transform public key X to be monic as $\widetilde{X}{:=}X/\mathrm{LC}(X)$.
1. Calculate the normal form $R_1(x,y,t) \in \mathbb{F}_p(t)[x,y]$ by reduction of F by $\widetilde{X}$.
2. Randomly choose the term satisfying $c_{ij}(t)x^iy^j$ $((i,j) \neq (0,0))$ and $c_{ij}(t)$ not being in $\mathbb{F}_p$, changing $c_{ij}(t)(\in \mathbb{F}_p(t)[x,y])$ to equivalent fractions with a common denominator, and let the numerator be $C(\in \mathbb{F}_p[t])$.
3. Factorize C in $\mathbb{F}_p[t]$, and let the set consisting of irreducible factors whose degree is greater than or equal to ℓ be $\hat{G}$. Choose $g \in \hat{G}$ and calculate the the normal form n by reduction of R_1 by g becomes an element in $\mathbb{F}_p[t]$.
4. Compute a polynomial $n(t) = n_{k-1}t^{k-1} + \cdots + n_1t + n_0 \in \mathbb{F}_p[t]$, outputs $m = n_0||n_1||\cdots||n_{k-1}$ and end.

During the reduction step, to obtain $m(t)$ in $\mathbb{F}_p[x,y,t]$ (not in $\mathbb{F}_p(t)[x,y]$), we must not combine appearing rational functions and the lower polynomial terms. This algorithm needs basic proposition and theorem as follows.

Proposition 1 (Proposition1 in pp.79-80 in [7]). *Let $G = \{g_1, \cdots, g_t\}$ be a Gröbner basis for an ideal $I \subset k[x_1, \cdots, x_n]$ and let $f \in k[x_1, \cdots, x_n]$ (which denotes the polynomial ring over the field k where $x_1, \cdots, x_n$ are variables). Then there is a unique $r \in k[x_1, \cdots, x_n]$ with the following two properties:*
(i) There is $g \in I$ such that $f = g + r$.
(ii) No term of r is reduced by any of $\mathrm{LT}(g_1), \cdots, \mathrm{LT}(g_t)$.
In particular, r is the normal form of the reduction of f by G no matter how the elements of G are listed when using the reduction algorithm.

Theorem 1. *Generated polynomials $g(t)$ and $n(t)$ in* **Algorithm 2** *are equal to the polynomial $f(t)$ used in encryption/decryption step and $m(t)$ obtained from plain text polynomial in Akiyama-Goto cryptosystem, respectively.*

Proof. We transform public key X to be monic as $\widetilde{X}{:=}X/\mathrm{LC}(X) \in \mathbb{F}_p(t)[x,y]$. Let $I := \left\langle \widetilde{X} \right\rangle \subset \mathbb{F}_p(t)[x,y]$ be a principal ideal generated by $\widetilde{X}$. Then $\{\widetilde{X}\}$ is a Gröbner basis of I. Note that we are not working in $\mathbb{F}_p[x,y,t]$ but in $\mathbb{F}_p(t)[x,y]$. As I is a principal ideal, any elements of I is uniquely denoted by $G\widetilde{X}$ ($G \in \mathbb{F}_p(t)[x,y]$). Therefore, $\exists G_1, R_1 \in \mathbb{F}_p(t)[x,y]$ *s.t.* $F = G_1\widetilde{X} + R_1$ are uniquely determined by **Proposition 1**. R_1 is equivalent to R_1 in Step 1 in **Algorithm 2**. Note that R_1 satisfies the condition in **Proposision 1**(ii) i.e. no term of R_1 is reduced by $\mathrm{LT}(\widetilde{X})$. Similarly, for $s(x,y,t)$, $\exists G_2, R_2 \in \mathbb{F}_p(t)[x,y]$ *s.t.* $s = G_2\widetilde{X} + R_2$ are uniquely determined, and no term of R_2 is reduced by $\mathrm{LT}(\widetilde{X})$. Substitute $X = \mathrm{LC}(X)\widetilde{X}$ and $s = G_2\widetilde{X}+R_2$ for the cipher polynomial generated as $F = m(t) + f(t)s(x,y,t) + X(x,y,t)r(x,y,t)$, then we obtain

$$\begin{aligned} F(x,y,t) &= m(t) + f(t)(G_2(x,y,t)\widetilde{X}(x,y,t) + R_2(x,y,t)) + \mathrm{LC}(X)\widetilde{X}(x,y,t)r(x,y,t) \\ &= m(t) + f(t)R_2(x,y,t) + \widetilde{X}(x,y,t)(f(t)G_2(x,y,t) + \mathrm{LC}(X)r(x,y,t)). \end{aligned}$$

where $m(t), f(t), \mathrm{LC}(X) \in \mathbb{F}_p[t]$, $r(x,y,t) \in \mathbb{F}_p[x,y,t]$, and $R_2(x,y,t), \widetilde{X}(x,y,t)$, $G_2(x,y,t) \in \mathbb{F}_p(t)[x,y]$. Now $\mathrm{LT}(X)$, leading term of $X(x,y,t)$ is monic w.r.t. x and y, therefore it satisfies **Assumption 1**, then each term of $m(t) + f(t)R_2$ cannot be reduced by $\mathrm{LT}(\widetilde{X})$, therefore the uniqueness of R_1 in **Proposition 1** leads to the equality $m(t) + f(t)R_2 = R_1$. Now we assume that $R_2 = R_2(t) \in \mathbb{F}_p(t)$ i.e. free from x and y. If we substitute $D_1 = (u_x(t), u_y(t), t)$ and $D_2 = (u_x(t), u_y(t), t)$ for cipher polynomial F, respectively, we obtain the following equations in the decryption step of Akiyama-Goto cryptosystem.

$$\begin{aligned} h_1(t) &= F(u_x(t), u_y(t), t) = m(t) + f(t)s(u_x(t), u_y(t), t) \\ &= m(t) + f(t)\left(G_2(u_x(t), u_y(t), t)X(u_x(t), u_y(t), t) + R_2(t)\right) = m(t) + f(t)R_2(t) \end{aligned}$$

by substituting $s = G_2\widetilde{X} + R_2$. Similary, we obtain $h_2(t) = m(t) + f(t)R_2(t)$, too. This means $h_1 = h_2$ then it cannot be decrypted. Therefore, R_2 has terms as $x^iy^jt^k$ ($(i,j) \neq (0,0), k \geq 0$) in its numerator, and then $R_1 = m(t) + f(t)R_2$ has terms as $x^iy^jt^k$ ($(i,j) \neq (0,0), k \geq 0$) in its numerator, too. Let C be the coefficient satisfying the condition of Step 2 in all terms of R_1. Then $f \in \hat{G}$ as C is reducible by $f(t)$. If there exists $g \in \hat{G}$ differ from f such that the normal form of the reduction of R_1 by g becomes an element of $\mathbb{F}_p[t]$, then $\exists G_3, n \in \mathbb{F}_p(t)[x,y]$ *s.t.* $R_1 = G_3g + n$ are uniquely determined and $n \in \mathbb{F}_p(t)$ by **Proposition 1**. Then, by the decryption step, we have the equality as $h_1(t) - h_2(t) = g(t)(G_3(u_x(t), u_y(t), t) - G_3(v_x(t), v_y(t), t))$. Then we have $g(t) = f(t)$, and it comes to a contradiction. Therefore, the generated polynomial g in Step 3 is equal to f. We utilize the above equation, we have the equality as $R_1 = n(t) + f(t)G_3 = m(t) + f(t)R_2$ If $m \neq 0$ then using the fact that each term of m is not reducible by the term of $f(t)$ and **Proposition 1**, we obtain $n(t) = m(t)$. Moreover $n(t) \in \mathbb{F}_p[t]$(not $\mathbb{F}_p(t)$), because, in the above equation, we can see all higher order terms have a factor $f(t)$ which reduce remaining all factors (which

are in $\mathbb{F}_p(t)[x,y]$) to be 0, i.e. only $m(t) \in \mathbb{F}_p[t]$ remains. If $m = 0$ then $n(t)$ is reducible by $f(t)$, we obtain $n(t) = m(t) \in \mathbb{F}_p[t]$.

Example 3. $\widetilde{X}_B(x,y,t) := X_B(x,y,t)/\mathrm{LC}(X_B)$ where $\mathrm{LC}(X_B) = t(1+t+t^4)$. Note that we are working in $k(t)[x,y]$ (polynomial ring w.r.t. x,y over rational function field w.r.t. t). $F_B(x,y,t) \underset{\tilde{X}_B(x,y,t)}{\longrightarrow} R_1(x,y,t)(\in \mathbb{F}_2(t)[x,y])$, where $t^2(1+t+t^4)^3$ appears at denominators. The following list is the set of numerators, which are factorized over $\mathbb{F}_2$, of all nonzero terms of $R_1(x,y,t)$ except for c_{00}, after changing them to equivalent fractions with a common denominator.

$\{(1+t)^5(1+t+t^2)(1+t+t^4)(1+t^3+t^4+t^5+t^6+t^7+t^8+t^{10}+t^{12})(1+t^2+t^3+t^4+t^5+t^6+t^8+t^9+t^{11}+t^{13}+t^{14})(1+t+t^2+t^4+t^7+t^9+t^{10}+t^{11}+t^{14}+t^{17}+t^{22}+t^{23}+t^{25}+t^{26}+t^{27}+t^{28}+t^{32}+t^{34}+t^{36}+t^{38}+t^{40})$, $\cdots\cdots\cdots\cdots$, $(1+t)(1+t+t^5+t^6+t^8)(1+t+t^2+t^4+t^7+t^9+t^{10}+t^{11}+t^{14}+t^{17}+t^{22}+t^{23}+t^{25}+t^{26}+t^{27}+t^{28}+t^{32}+t^{34}+t^{36}+t^{38}+t^{40})\}$.

We can see all nonzero elements (except c_{00}) have the same factor $1+t+t^2+t^4+t^7+t^9+t^{10}+t^{11}+t^{14}+t^{17}+t^{22}+t^{23}+t^{25}+t^{26}+t^{27}+t^{28}+t^{32}+t^{34}+t^{36}+t^{38}+t^{40}) := g(t)$ (i.e. $f(t)$ is obtained). In the actual computation, we may only to calculate GCD of any two elements (except c_{00}). Finally, we reduce $R(x,y,t)$ by $g(t)(= f(t))$ and obtain the plain text polynomial $m(t)$.

4.2 Attack by Utilizing Gröbner Base Techniques in $\mathbb{F}_p[x,y,t,\mathcal{A}]$

In this section, we utilize Gröbner base techniques introducing a new parameter. This algorithm enables us not to work via rational function field but to keep staying in the polynomial ring. We need the following Corollary.

Corollary 1 (Corollary2 in pp.80 in [7]). *Let $G = \{g_1, \cdots, g_t\}$ be a Gröbner basis for an ideal $I \subset k[x_1, \cdots, x_n]$ and let $f \in k[x_1, \cdots, x_n]$. Then $f \in I$ if and only if the normalform of the reduction of f by G is zero.*

Algorithm 3. [utilizing Gröbner base techniques in $\mathbb{F}_p[x,y,t,\mathcal{A}]$]
Input : Akiyama-Goto's public key $X(x,y,t) \in \mathbb{F}_p[x,y,t]$,
cipher polynomial $F(x,y,t) \in \mathbb{F}_p[x,y,t]$.
Ountput : Plaintext m which corresponds to the cipher polynomial $F(x,y,t)$.

0. Calculate Gröbner base GB_X for an ideal $I_X := \langle \mathcal{A} \cdot X(x,y,t), \mathcal{A} \cdot \mathrm{LC}(X) - 1 \rangle \subset \mathbb{F}_p[x,y,t,\mathcal{A}]$, introducing a new parameter $\mathcal{A}$, using the order $x \succ y \succ \mathcal{A} \succ t$ in $\mathbb{F}_p[x,y,t,\mathcal{A}]$.
1. Calculate the normal form $R(x,y,t,\mathcal{A}) \in \mathbb{F}_p[x,y,t,\mathcal{A}]$ of the reduction of $F(x,y,t)$ by GB_X.
2. Randomly choose the term satisfying $c_{ij}(t,\mathcal{A})x^iy^j$ $((i,j) \neq (0,0))$ where $c_{ij}(t,\mathcal{A})$ not being in $\mathbb{F}_p$, then let $c_{ij}(t,\mathcal{A})$ be C.
3. To perform desired factorization for detecting $f(t)$, we factor out powers of $\mathcal{A}$. Therefore we transform each term of C by using the relation $\mathcal{A} \cdot \mathrm{LC}(X) = 1$ as $\mathcal{A}^0 \mapsto (\mathcal{A} \cdot \mathrm{LC}(X))^2 = \mathcal{A}^2 \cdot \mathrm{LC}(X)^2, \mathcal{A}^1 \mapsto \mathcal{A}(\mathcal{A} \cdot \mathrm{LC}(X))^1 = \mathcal{A}^2 \cdot \mathrm{LC}(X), \cdots$, so as to make the powers of $\mathcal{A}$ of each term equal. Then perform factorization

in $\mathbb{F}_p[t, \mathcal{A}]$, and let the set consisting of irreducible fators whose degree is greater than or equal to ℓ be $\hat{G}$. Choose the element $g(t) \in \hat{G}$. Calculate Gröbner base GB_g for an ideal $I_g := \langle g(t), \mathcal{A} \cdot \mathrm{LC}(X) - 1 \rangle \subset \mathbb{F}_p[t, \mathcal{A}]$, and calculate the normal form $n \in \mathbb{F}_p[t]$ reducing $R(0, 0, t, \mathcal{A})$ by GB_g.

4. Compute a polynomial $n(t) = n_{k-1}t^{k-1} + \cdots + n_1 t + n_0 \in \mathbb{F}_p[t]$, outputs $m = n_0||n_1|| \cdots ||n_{k-1}$ and end.

Theorem 2. *Generated polynomials $g(t)$ and $n(t)$ in* **Algorithm 3** *are equal to the polynomial $f(t)$ used in encryption/decryption step and $m(t)$ obtained from plain text polynomial in Akiyama-Goto cryptosystem, respectively.a*

Proof. Reduction by $\widetilde{X} := X/\mathrm{LC}(X) \in \mathbb{F}_p(t)[x, y]$ in **Algorithm2** is equivalent to reduction by Gröbner base GB_X of an ideal $I_X = \langle \mathcal{A} \cdot X(x, y, t), \mathcal{A} \cdot \mathrm{LC}(X) - 1 \rangle \subset \mathbb{F}_p[x, y, t, \mathcal{A}]$. The technique of dealing with rational function by Gröbner base in the polynomial ring is well-known in computer algebra. We assume $\mathrm{GB}_X := \{g_1, \cdots, g_w\} \subset \mathbb{F}_p[x, y, t, \mathcal{A}]$ $(g_1 \succ \cdots \succ g_w)$. Note that g_1 is monic w.r.t. x, y as generators of GB_X are AX and $A \cdot \mathrm{LC}(X) - 1$. As can be seen in the proofs in **Algorithm1, 2**, in the word of Gröbner base theory, the strategies is : (1) Reduce a cipher polynomial $F(x, y, t)$ by GB_X, (2) Detect $f(t)$, (3) Continue reducing by GB_f, (4) Obtained normal form is a plain text polynomial $m(t)$.

Let $\widetilde{R_1}(x, y, t, \mathcal{A})$ be the normal form of $F(x, y, t)$ reduced by GB_X, i.e.

$$F(x, y, t) = \sum_{i=1}^{w} a_i(x, y, t, \mathcal{A}) g_i(x, y, t, \mathcal{A}) + \widetilde{R_1}(x, y, t, \mathcal{A}) \underset{\mathrm{GB}_X}{\longrightarrow} \widetilde{R_1}(x, y, t, \mathcal{A})$$

where $g_i(x, y, t, \mathcal{A}) \succ \widetilde{R_1}(x, y, t, \mathcal{A})$, $a_i(x, y, t, \mathcal{A}), \widetilde{R_1}(x, y, t, \mathcal{A}) \in \mathbb{F}_p[x, y, t, \mathcal{A}]$, $g_i(x, y, t, \mathcal{A}) \in \mathrm{GB}_X \subset I_X \in \mathbb{F}_p[x, y, t, \mathcal{A}]$. Whereas, F is constructed as $F(x, y, t) = m(t) + f(t)s(x, y, t) + X(x, y, t)r(x, y, t)$. Let $\widetilde{R_2}(x, y, t, \mathcal{A})$ be the normal of $s(x, y, t)$ reduced by GB_X, i.e. $s(x, y, t) = \sum_{i=1}^{w} b_i g_i + \widetilde{R_1} \underset{\mathrm{GB}_X}{\longrightarrow} \widetilde{R_2}$ where $g_i \succ \widetilde{R_2}$, $b_i, \widetilde{R_1} \in \mathbb{F}_p[x, y, t, \mathcal{A}]$, $g_i \in GB_X \subset \widetilde{R_1} \in \mathbb{F}_p[x, y, t, \mathcal{A}]$. $\mathcal{A}X$, $\mathrm{LC}(X) \cdot \mathcal{A} - 1 \in I_X$ implies $\mathcal{A}X \underset{\mathrm{GB}_X}{\longrightarrow} 0$ and $\mathrm{LC}(X) \cdot \mathcal{A} - 1 \underset{\mathrm{GB}_X}{\longrightarrow} 0$ by **Corollary 1**. Then $X(x, y, t) = \mathrm{LC}(X) \cdot (\mathcal{A} \cdot X) - X \cdot (\mathrm{LC}(X) \cdot \mathcal{A} - 1) \underset{\mathrm{GB}_X}{\longrightarrow} 0$. Therefore, from the uniqueness of the normal form by Gröbner base, we obtain $F(x, y, t) \underset{\mathrm{GB}_X}{\longrightarrow} m(t) + f(t)\widetilde{R_2}(x, y, t, \mathcal{A}) = \widetilde{R_1}(x, y, t, \mathcal{A}) \in \mathbb{F}_p[x, y, t, \mathcal{A}]$

Note that if we substitute $\mathcal{A} = 1/\mathrm{LC}(X)$ into $\widetilde{R_i}(x, y, t, \mathcal{A}) \in \mathbb{F}_p[x, y, t, \mathcal{A}]$ in **Algorithm 3**, then we obtain $R_i(x, y, t)$ in **Algorithm 2** as $\widetilde{R_i}(x, y, t, 1/\mathrm{LC}(X)) = R_i(x, y, t) \in \mathbb{F}_p(t)[x, y]$ $(i = 1, 2)$. The proof that $\widetilde{R_2}$ has terms as $x^i y^j t^k$, $((i, j) \neq (0, 0), k \geq 0)$, and the proof can be done in the same way as we see in **Algorithm 1,2**. Therefore focusing attention on coefficients of $x^i y^j$ $(i, j) \neq (0, 0)$, $f(t)$ can be detected if we factor out power of $\mathcal{A}$ utilizing the relation $\mathcal{A} \cdot \mathrm{LC}(X) = 1$ over $\mathbb{F}_p$ (or substitute $1/\mathrm{LC}(X)$ into $\mathcal{A}$ as in **Algorithm 2**. It depends on the software system which method we should use).

To reduce $\widetilde{R_1}(x, y, t, \mathcal{A})$ by $f(t)$, we use the Gröbner base GB_f of an ideal $I_f = \langle f(t),\ \mathcal{A} \cdot \mathrm{LC}(X) - 1 \rangle,\ x \succ y \succ \mathcal{A} \succ t$. As $m(t) \in \mathbb{F}_p[t]$ and $f(t) \succ m(t)$, we can calculate as $\widetilde{R_1}(0, 0, t, \mathcal{A}) = m(t) + f(t)\widetilde{R_2}(0, 0, t, \mathcal{A}) \underset{\mathrm{GB}_f}{\longrightarrow} m(t)$.

Example 4. We cryptanalyze the cipher polynomial $F_B(x, y, t)$ and obtain a plain text m by **Algorithm 3** using the order $x \succ y \succ \mathcal{A} \succ t$ in $\mathbb{F}_2[x, y, t, \mathcal{A}]$. Let GB_X be a Gröbner Basis for ideal $I_X := \langle \mathcal{A} \cdot X(x, y, t), \mathcal{A} \cdot \mathrm{LC}(X) - 1 \rangle$ which is calculated as follows.

$\mathrm{GB}_X = \{1+\mathcal{A}t+\mathcal{A}t^2+\mathcal{A}t^5, 1+\mathcal{A}t+t^2+t^3+\mathcal{A}t^3+t^6+t^7+\cdots\cdots+\mathcal{A}t^2xy^5+x^2y^5+\mathcal{A}x^2y^5+\mathcal{A}t^2x^2y^5+x^3y^5+\mathcal{A}x^3y^5+\mathcal{A}tx^3y^5+\mathcal{A}t^2x^3y^5\mathcal{A}x^4y^5+\mathcal{A}t^4x^4y^5+x^5y^5\}$
Calculate the normal form of the reduction of $F_B(x, y, t)$ by GB_X, we obtain $R(x, y, t, \mathcal{A}) = \sum c_{ij}(t, \mathcal{A})x^iy^j \in \mathbb{F}_2[x, y, t, \alpha]$. Take some terms $c_{ij}(t, \mathcal{A})x^iy^j$ except including c_{00}, calculate GCD of $\{c_{ij}(t, \mathcal{A})\}$, by extracting powers of $\mathcal{A}$ by using the relation $\mathcal{A} \cdot \mathrm{LT} = 1$. For example, if we take $1 + t + t^2 + \mathcal{A}t^2 + t^3 + \mathcal{A}^2t^3 + t^6 + t^9 + t^{11} + t^{12} + t^{14} + t^{18} + t^{20} + t^{27} + t^{28} + t^{34} + t^{40}$, then we transform by $\mathcal{A}^0 \to (\mathcal{A} \cdot \mathrm{LC})^2 = \mathcal{A}^2 \cdot \mathrm{LC}^2$, $\mathcal{A}^1 \to \mathcal{A}(\mathcal{A} \cdot \mathrm{LC})^1 = \mathcal{A}^2 \cdot \mathrm{LC}$ to factor out $\mathcal{A}^2$. Then we obtain $\mathcal{A}^2(t^2(1 + t^3 + t^4)^2(1 + t + t^2 + t^4 + t^7 + t^9 + t^{10} + t^{11} + t^{14} + t^{17} + t^{22} + t^{23} + t^{25} + t^{26} + t^{27} + t^{28} + t^{32} + t^{34} + t^{36} + t^{38} + t^{40})$ and detect $f(t)$ having maximum degree. Finally, we reduce $R(0, 0, t, \mathcal{A})$ by GB_f as follows.
$R(0, 0, t, \mathcal{A}) = 1 + \mathcal{A} + \mathcal{A}^2t + \mathcal{A}t^3 + \mathcal{A}^2t^3 + \mathcal{A}t^4 + \mathcal{A}^2t^4 + t^5 + t^6 + t^7 + t^{13} + t^{15} + t^{16} + t^{18} + t^{19} + t^{22} + t^{23} + t^{25} + t^{30} + t^{33} + t^{34} + t^{37} + t^{40} + t^{41} + t^{44} + t^{46} + t^{58} + t^{60} + t^{62} + t^{63} + t^{64}$, $\mathrm{GB}_f = \{1 + t + t^2 + t^4 + t^7 + t^9 + t^{10} + t^{11} + t^{14} + t^{17} + t^{22} + t^{23} + t^{25} + t^{26} + t^{27} + t^{28} + t^{32} + t^{34} + t^{36} + t^{38} + t^{40}, A + t + t^2 + t^5 + t^6 + t^7 + t^8 + t^{10} + t^{12} + t^{13} + t^{15} + t^{16} + t^{18} + t^{21} + t^{27} + t^{30} + t^{32} + t^{33} + t^{34} + t^{39}\}$,
$R(0, 0, t, \mathcal{A}) \underset{\mathrm{GB}_f}{\longrightarrow} m(t)$ (We succeed in obtaining the plain text polynomial.)

5 Conclusion

In this paper, two algorithms to attack Akiyama-Goto Algebraic Surface Public-key Cryptosystem (2005) are suggested. They are applicable to all cases, i.e. it shows that the cryptosystem is useless. One is a straightforward generalization of Uchiyama-Tokunaga's attack in $\mathbb{F}_p[x, y, t]$, by working via polynomial ring over rational function field $\mathbb{F}_p(t)[x, y]$. And the second one takes advantages of Gröbner base techniques so as to work in the polynomial ring $\mathbb{F}_p[x, y, t, \mathcal{A}]$, introducing a new parameter $\mathcal{A}$.

References

[1] Akiyama, A., Goto, Y.: A Construction of an Algebraic Surface Public-key Cryptosystem. In: CD-ROM 2E4-3, Symposium on Cryptography and Information Security (SCIS 2005), January 2005, pp. 925–930 (2005)
[2] Algebraic surface public key cryptosystem. opened to the general public at website (February 2005), `http://www.toshiba.co.jp/rdc/rd/topics_e_05.htm#050206`, `http://www.toshiba.co.jp/rdc/rd/detail_j/0502_06.htm`
[3] Akiyama, A., Goto, Y.: A Security Analysis for a Public-key Cryptosystem using Algebraic Surfaces. In: CD-ROM 2A3-1, SCIS 2006 (January 2006)
[4] Akiyama, K., Goto, Y.: A Public-key Cryptosystem using Algebraic Surfaces. In: Workshop Record of the International Workshop on Post-Quantum Cryptography (PQCrypto 2006), May 2006, pp. 119–138 (2006)

[5] Uchiyama, S., Tokunaga, H.: On the Security of the Algebraic Surface Public-Key Cryptosystems. In: CD-ROM 2C1-2, SCIS 2007 (January 2007) (written in Japanese)
[6] Cryptography Research and Evaluation Committees : CRYPTREC Report 2006, Report of the Cryptographic Technique Monitoring Subcommittee (March 2007), http://www2.nict.go.jp/y/y213/cryptrec_publicity/c06_wat_final.pdf
[7] Cox, D., Little, J., O'Shea, D.: Ideals, Varieties, and Algoriths: An Introduction to Computational Algebraic Geometry and Commutative Algebra, 2nd edn. Springer, Heidelberg
[8] Iwami, M.: A Reduction Attack on Algebraic Surface Public-Key Cryptosystems. Workshop of Research Institute for Mathematical Sciences (RIMS) Kyoto University, New development of research on Computer Algebra (held on July 4-6 2007) RIMS Kokyuroku 1572, pp.114–123 (November 2007) (written in Japanese)

The Four Colour Theorem: Engineering of a Formal Proof

Georges Gonthier

Microsoft Research, Cambridge, England

Abstract. The 150 year old Four Colour Theorem is the first famous result with a proof that requires large computer calculations. Such proofs are still controversial: It is thought that computer programs cannot be reviewed with mathematical rigor.

To overturn this belief, we have created a fully computer-checked proof of the Four Colour Theorem. Using the Coq proof assistant, we wrote an extended program that specifies both the calculations and their mathematical justification. Only the interface of the program – the statement of the theorem – needs to be reviewed. The rest (99.8%) is self-checking: Coq verifies that it strictly follows the rules of logic. Thus, our proof is more rigorous than a traditional one.

Our effort turned out to be more than just an exercise in verification; having to definine rigorously all key concepts provided new mathematical insight into the concept of planarity. Planarity has topological and combinatorial characterizations, which are often confused in arguments that are both pictorially appealing and logically incomplete. The rigor of our computer proof imposed a strict separation between the two.

We developed a purely combinatorial theory of planarity based on a symmetrical presentation of hypermaps, which greatly simplified the proof. The theory supplies an elegant analogue of the Jordan Curve property, which allowed us to prove the Theorem under minimal topological assumptions, without appealing to Jordan Curve theorem.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, p. 333, 2008.
© Springer-Verlag Berlin Heidelberg 2008

On the Computation of Elimination Ideals of Boolean Polynomial Rings

Yosuke Sato[1], Akira Nagai[2], and Shutaro Inoue[3]

[1] Tokyo University of Science, 1-3, Kagurazaka, Shinjuku-ku, Tokyo, Japan
ysato@rs.kagu.tus.ac.jp
[2] NTT Information Sharing Platform Laboratories,
3-9-11, Midorimachi, Musashino, Tokyo, Japan
nagai.akira@lab.ntt.co.jp
[3] Tokyo Universityy of Science, 1-3, Kagurazaka, Shinjuku-ku, Tokyo, Japan
inoue@mi.kagu.tus.ac.jp

Abstract. In order to compute an eliminate portion of a given polynomial ideal by a Gröbner basis computation, we usually need to compute a Gröbner basis of the whole ideal with respect to some proper term order. In a boolean polynomial ring, we show that we can compute an eliminate portion by computing Gröbner bases in the boolean polynomial ring with the same coefficient ring that has the only variables which we want to eliminate. We also check the efficiency of our method through our implementation.

Keywords: Boolean Gröbner Bases.

1 Introduction

For solving polynomial equations, Gröbner bases computation is a powerful tool. Though Gröbner bases were originally introduced by B.Buchberger in polynomial rings over fields([1]), there also have been done many works concerning Gröbner bases of polynomial rings with coefficient rings that are not fields. Among them Gröbner bases of boolean polynomial rings (*boolean Gröbner bases*) introduced in [8,9] have a nice property.

In computation of a Gröbner basis, selection of the term order dramatically affects the computation cost both for time and space. In a polynomial ring over a field, a total degree reverse lexicographic term order is usually the least expensive one. In most cases, a purely lexicographic term order is much more expensive. In a boolean polynomial ring, however, according to the data we got through many computation experiments of boolean Gröbner bases, computation costs of a purely lexicographic term order and other term orders such as a total degree reverse lexicographic term order are not much different as far as we keep the order of each variable. This property is extremely pleasant, since a purely lexicographic term order is most convenient when we compute an eliminate portion of a given ideal. In a boolean polynomial ring, we also have a nice classical result, namely *boolean extension theorem*, that is we can always extend a zero of the elimination ideal to a zero of the whole ideal. (See Theorem 6 for more details.) These

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 334–348, 2008.
© Springer-Verlag Berlin Heidelberg 2008

properties enable us to handle boolean equations completely with computation of elimination ideals by boolean Gröbner bases. In fact, "Set Constraint Solvers" that is a free software developed by Y.Sato([10,12]), employs a naive but efficient algorithm for the computation of the eliminate portion of a given ideal. For an ideal I of a boolean polynomial ring $\mathbf{B}(X_1, \ldots, X_n)$(precise definition is given in Definition 4), the program computes a boolean Gröbner basis of I w.r.t. the purely lexicographic term order such that $X_n > \cdots > X_1$ in order to compute the elimination ideals $I \cap \mathbf{B}(X_1, \ldots, X_i)$ for all $i = 1, \ldots, n-1$ simultaneously.

Besides the naive algorithm, we have an another algorithm for elimination ideals which is based on the computation of comprehensive Gröbner bases. Unlike in polynomial rings over fields, construction of comprehensive Gröbner bases is very simple in boolean polynomial rings. Since a boolean polynomial ring is also a boolean ring, a boolean polynomial ring $\mathbf{B}(\bar{A}, \bar{X})$ with variables $\bar{A}$ and $\bar{X}$ can be considered as a boolean polynomial ring $(\mathbf{B}(\bar{A}))(\bar{X})$ over the coefficient boolean ring $\mathbf{B}(\bar{A})$. Given an ideal I of $\mathbf{B}(\bar{A}, \bar{X})$ and a term order of variables $\bar{X}$. Let G be a boolean Gröbner basis of I in $(\mathbf{B}(\bar{A}))(\bar{X})$, then G becomes a comprehensive Gröbner basis of I with parameters $\bar{A}$. (See Theorem 24 for more details.) In the view of construction of elimination ideals, the important fact is that $G \cap \mathbf{B}(\bar{A})$ is either an empty set or a singleton of a boolean polynomial $f(\bar{A})$ of $\mathbf{B}(\bar{A})$. In case of an empty set, the eliminate portion $I \cap \mathbf{B}(\bar{A})$ is equal to $\{0\}$, otherwise it is equal to the ideal $\langle f(\bar{A}) \rangle$ in $\mathbf{B}(\bar{A})$. Unfortunately, in most cases, a naive algorithm to construct the above G is much more expensive than the computation of boolean Gröbner basis of I in $\mathbf{B}(\bar{A}, \bar{X})$ w.r.t. a purely lexicographic term order such that $\bar{X} > \bar{A}$. Though there is a work concerning efficient computations of comprehensive Gröbner bases([7]), it is based on the computation of boolean Gröbner bases in a boolean polynomial ring $\mathbf{B}(\bar{A}, \bar{X})$, and it does not give us a more efficient algorithm for computation of elimination ideals than the naive algorithm employed in "Set Constraint Solvers".

In this paper, we show that for a given ideal $I = \langle f_1(\bar{A}, \bar{X}), \ldots, f_l(\bar{A}, \bar{X}) \rangle$ in a boolean polynomial ring $\mathbf{B}(\bar{A}, \bar{X})$, we can construct the elimination ideal $I \cap \mathbf{B}(\bar{A})$ by computing a boolean Gröbner basis of the ideal $\langle f_1(\bar{c}, \bar{X}), \ldots, f_l(\bar{c}, \bar{X}) \rangle$ in the boolean polynomial ring $\mathbf{B}(\bar{X})$ for each $0, 1$ specialization $\bar{c}$ of $\bar{A}$. We also give some example of our computation experiments to show that our method is quite effective in case we do not have many parameters.

Our plan is as follows. In section 2, we show two classical results of boolean algebra which we need for understanding our work.

We give a quick review of boolean Gröbner bases and comprehensive boolean Gröbner bases in section 3 and 4. Section 5 is devoted to our new algorithm for the computation of elimination ideals.

2 Boolean Polynomial Ring

In this section, we show two classical results of boolean algebra in terms of boolean polynomial rings. More details can be found in many text books of boolean algebra such as [6] for example.

Definition 1. *A commutative ring* $\mathbf{B}$ *with an identity* 1 *is called* a boolean ring *if every element* a *of* $\mathbf{B}$ *is idempotent, i.e.* $a^2 = a$.

$\langle \mathbf{B}, \vee, \wedge, \neg \rangle$ becomes a boolean algebra with the boolean operations $\vee, \wedge, \neg$ defined by $a \vee b = a + b + a \cdot b, a \wedge b = a \cdot b, \neg a = 1 + a$. Conversely, for a boolean algebra $\langle \mathbf{B}, \vee, \wedge, \neg \rangle$, if we define $+$ and $\cdot$ by $a + b = (\neg a \wedge b) \vee (a \wedge \neg b)$ and $a \cdot b = a \wedge b$, $\langle \mathbf{B}, +, \cdot \rangle$ becomes a boolean ring. We use the symbol $\succeq$ to denote a partial order of a boolean ring, that is $a \succeq b$ if and only if $ab = b$ for elements a, b of a boolean ring $\mathbf{B}$.

Since $-a = a$ in a boolean ring, we do not need to use the symbol '$-$', however, we also use $-$ when we want to stress its meaning.

Definition 2. *A non-zero element* e *of a boolean ring* $\mathbf{B}$ *is said to be* atomic*, if there does not exist a non-zero element* c *such that* $ce = c$ *except for* $c = e$. (*An atomic element is nothing but a non-zero minimal element w.r.t.* $\succeq$.)

Lemma 3. *If* $\mathbf{B}$ *is a finite boolean ring, it has at least one atomic element. Let* $e_1, \ldots, e_k$ *be all the atomic elements of* $\mathbf{B}$*, then* $e_i e_j = 0$ *for any* $i \neq j$ *and* $e_1 + \cdots + e_k = 1$.

proof. We show the last equation, the rests are obvious. If $e_1 + \cdots + e_k \neq 1$, $e_1 + \cdots + e_k + 1 \neq 0$. Let c be a minimal element(an atomic element) of $\mathbf{B}$ such that $e_1 + \cdots + e_k + 1 \succeq c$, i.e. $c(e_1 + \cdots + e_k + 1) = c$. It follows that $c(e_1 + \cdots + e_k) = 0$. Since c is a minimal element, $c = e_i$ for some e_i, which leads us to a contradiction $e_i = e_i(e_1 + \cdots + e_k) = 0$. □

Definition 4. *Let* $\mathbf{B}$ *be a boolean ring. A quotient ring* $\mathbf{B}[X_1, \ldots, X_n]/\langle X_1^2 - X_1, \ldots, X_n^2 - X_n \rangle$ *with an ideal* $\langle X_1^2 - X_1, \ldots, X_n^2 - X_n \rangle$ *becomes a boolean ring. It is called* a boolean polynomial ring *and denoted by* $\mathbf{B}(X_1, \ldots, X_n)$*, its element is called* a boolean polynomial.

Note that a boolean polynomial of $\mathbf{B}(X_1, \ldots, X_n)$ is uniquely represented by a polynomial of $\mathbf{B}[X_1, \ldots, X_n]$ that has at most degree 1 for each variable X_i. In what follows, we identify a boolean polynomial with such a representation.

Multiple variables such as $X_1, \ldots, X_n$ or $Y_1, \ldots, Y_m$ are abbreviated to $\bar{X}$ or $\bar{Y}$ respectively. Lower small Greek letters such as a, b, c are usually used for elements of a boolean ring $\mathbf{B}$. The symbol $\bar{a}$ denotes an n-tuple of element of $\mathbf{B}$ for some n. For $\bar{a} = (a_1, \ldots, a_n)$ and $\bar{b} = (b_1, \ldots, b_m)$, $(\bar{a}, \bar{b})$ denotes an $n + m$-tuple $(a_1, \ldots, a_n, b_1, \ldots, b_m)$. For a boolean polynomial $f(\bar{X}, \bar{Y})$ with variables $\bar{X}$ and $\bar{Y}$, $f(\bar{a}, \bar{Y})$ denote a boolean polynomial in $\mathbf{B}(\bar{Y})$ obtained by specializing $\bar{X}$ with $\bar{a}$.

Definition 5. *Let* I *be an ideal of* $\mathbf{B}(X_1, \ldots, X_n)$*. For a subset* S *of* $\mathbf{B}$*,* $V_S(I)$ *denotes a subset* $\{\bar{a} \in S^n | \forall f \in I f(\bar{a}) = 0\}$*. When* $S = \mathbf{B}$*,* $V_{\mathbf{B}}(I)$ *is simply denoted by* $V(I)$ *and called a* variety *of* I*. We say* I *is* satisfiable *in* S *if* $V_S(I)$ *is not empty. When* $S = \mathbf{B}$*, we simply say* I *is* satisfiable.

Theorem 6 (boolean extension theorem). *Let* I *be a finitely generated ideal in a boolean polynomial ring* $\mathbf{B}(Y_1, \ldots, Y_m, X_1, \ldots, X_n)$.

For any $\bar{b} \in V(I \cap \mathbf{B}(\bar{Y}))$*, there exist* $\bar{c} \in \mathbf{B}^n$ *such that* $(\bar{b}, \bar{c}) \in V(I)$.

Proof. It suffices to show the theorem for $n = 1$. Note first that any finitely generated ideal is principal in a boolean ring, that is an ideal $\langle f_1, \ldots, f_s \rangle$ is equal to the principal ideal $\langle f_1 \vee \cdots \vee f_s \rangle$. Let $I = \langle fX_1 + g \rangle$ for some $f, g \in \mathbf{B}(\bar{Y})$. We claim that $I \cap \mathbf{B}(\bar{Y}) = \langle fg+g \rangle$. Since $(f+1)(fX_1+g) = fg+g$, $fg+g \in I \cap \mathbf{B}(\bar{Y})$. Conversely, suppose that $h \in I \cap \mathbf{B}(\bar{Y})$, i.e. there exist $p, q \in \mathbf{B}(\bar{Y})$ such that $h = (pX_1 + q)(fX_1 + g)$. Then, $h = (pf + pg + qf)X_1 + qg$. Since $h \in \mathbf{B}(\bar{Y})$, we must have $pf + pg + qf = 0$, from which we have $h = qg = fqg + (f + 1)qg = g(pf+pg)+(f+1)qg = gp(f+1)+(f+1)qg = (p+q)(f+1)g \in \langle fg+g \rangle$. Suppose now that $\bar{b} \in V(\langle fg + g \rangle)$, that is $f(\bar{b})g(\bar{b}) + g(\bar{b}) = 0$. Let $c = (f(\bar{b}) + 1)d + g(\bar{b})$ where d can be any element of $\mathbf{B}$. Then $f(\bar{b})c + g(\bar{b}) = f(\bar{b})g(\bar{b}) + g(\bar{b}) = 0$. That is $(\bar{b}, c) \in V(I)$. □

Corollary 7 (boolean weak Nullstellensatz). *For any finitely generated ideal I of a boolean polynomial ring $\mathbf{B}(X_1, \ldots, X_n)$, the variety $V(I)(\subseteq \mathbf{B}^n)$ of I is an empty set if and only if there exists a non-zero constant element of $\mathbf{B}$ in I.*

Proof. If $I \cap \mathbf{B} = \{0\}$, the above proof also works to show that $V(I) \neq \emptyset$. The converse is trivial. □

Theorem 8 (boolean strong Nullstellensatz). *Let I be a finitely generated ideal of a boolean polynomial ring $\mathbf{B}(X_1, \ldots, X_n)$ such that $V(I) \neq \emptyset$.*

Then, for any boolean polynomial $h(\bar{X}) \in \mathbf{B}(\bar{X})$,

$$h(\bar{X}) \in I \quad \text{if and only if} \quad \forall(\bar{b}) \in V(I)\ h(\bar{b}) = 0.$$

Proof. Let $I = \langle f(\bar{X}) \rangle$ and $\mathbf{B}'$ be a boolean subring of $\mathbf{B}$ generated by all coefficients of $f(\bar{X})$ and $h(\bar{X})$, i.e. $\mathbf{B}'$ is the smallest boolean subring of $\mathbf{B}$ which includes all coefficients of $f(\bar{X})$ and $h(\bar{X})$. First note that I is also satisfiable in $\mathbf{B}'$ by boolean weak Nullstellensatz. Secondly note that $\mathbf{B}'$ is finite, because each element of $\mathbf{B}'$ is a sum of finite elements which have a form $a_1^{n_1} a_2^{n_2} \cdots a_l^{n_l}$ where $a_1, a_2, \ldots, a_l$ are coefficients of $f(\bar{X})$ and each n_i is either 0 or 1. By Lemma 3, $\mathbf{B}'$ has atomic elements $e_1, \ldots, e_k$ such that $e_i e_j = 0$ for any $i \neq j$ and $e_1 + \cdots + e_k = 1$. Suppose now that $\forall \bar{b} \in V(I)\ h(\bar{b}) = 0$. We certainly have the property:

$$\forall \bar{b} \in \mathbf{B}'^n (f(\bar{b}) = 0 \Rightarrow h(\bar{b}) = 0) \tag{1}$$

In order to show $h(\bar{X}) \in I$, we prove the following claims.

Claim 1: $f(b_1, \ldots, b_n) = 0 \Leftrightarrow e_i f(e_i b_1, \ldots, e_i b_n) = 0$ for each $i = 1, \ldots, k$.

Proof of Claim1. We clearly have $f(b_1, \ldots, b_n) = 0 \Leftrightarrow e_i f(b_1, \ldots, b_n) = 0$ for each $i = 1, \ldots, k$.We also have the equation $e_i f(b_1, \ldots, b_n) = e_i f(e_i b_1, \ldots, e_i b_n)$. The assertion follows from them. ⊠

Claim 2: $\forall (b_1, \ldots, b_n) \in \mathbf{B}'^n (\, e_i f(e_i b_1, \ldots, e_i b_n) = 0 \Rightarrow e_i h(e_i b_1, \ldots, e_i b_n) = 0\,)$ for each $i = 1, \ldots, k$.

Proof of Claim2. Let i be fixed and suppose $e_i f(e_i b_1, \ldots, e_i b_n) = 0$ for elements $b_1, \ldots, b_n$ in $\mathbf{B}'$. Since I is satisfiable in $\mathbf{B}'$, we have elements $c_1, \ldots, c_n$

in $\mathbf{B}'$ such that $f(c_1, \ldots, c_n) = 0$. Let $a_j = e_i b_j + (1 + e_i)c_j$ for each $j = 1, \ldots, n$. Then, we have $e_i a_j = e_i b_j$ and $e_t a_j = e_t c_j$ for each $t \neq i$. By Claim 1, we have $f(a_1, \ldots, a_n) = 0$. By the property (1), we have $h(a_1, \ldots, a_n) = 0$. By Claim 1 again, we have $e_i h(e_i a_1, \ldots, e_i a_n) = 0$ which is equivalent to $e_i h(e_i b_1, \ldots, e_i b_n) = 0$. ⊠

Claim 3: The ideal $\langle e_i f(\bar{X}), e_i(Uh(\bar{X}) + 1)\rangle \subseteq \mathbf{B}'(U, \bar{X})$ is unsatisfiable in $\mathbf{B}'$ for each $i = 1, \ldots, k$, where U is a new variable.

Proof of Claim3. Assume that $e_i f(b_1, \ldots, b_n) = 0$ for some $(b_1, \ldots, b_n) \in \mathbf{B}'^n$. By Claim 1, we have $e_i f(e_i b_1, \ldots, e_i b_n) = 0$. By Claim 2, we have $e_i h(e_i b_1, \ldots, e_i b_n) = 0$. By Claim 1 again, we have $e_i h(b_1, \ldots, b_n) = 0$.
Therefore $e_i(Uh(b_1, \ldots, b_n) + 1) = e_i \neq 0$. ⊠

By the last claim and boolean weak Nullstellensatz, we can see the ideal $\langle e_i f(\bar{X}), e_i(Uh(\bar{X})+1)\rangle$ contains a non-zero element of $\mathbf{B}'$. Since e_i is an atomic element of $\mathbf{B}'$, it must contain e_i. So, there exist boolean polynomials $p(U, \bar{X})$ and $q(U, \bar{X})$ of $\mathbf{B}'(U, \bar{X})$ such that $e_i = e_i f(\bar{X})p(U, \bar{X}) + e_i(Uh(\bar{X}) + 1)q(U, \bar{X})$.

Multiplying $h(\bar{X})$ from both sides and substituting U by 1, we have $e_i h(\bar{X}) = e_i f(\bar{X})p(1, \bar{X})h(\bar{X})$, which shows that $e_i h(\bar{X}) \in I$. So, $h(\bar{X}) = e_1 h(\bar{X}) + \cdots + e_k h(\bar{X}) \in I$.

The converse is trivial. □

3 Boolean Gröbner Bases

A boolean Gröbner basis is defined as a natural modification of a Gröbner basis in a polynomial ring over a boolean ring. Though it was introduced in [8,9] together with a computation algorithm using a special monomial reduction, the same notion was independently discovered by V.Weispfenning in a polynomial ring over a more general coefficient ring, namely, a commutative von Neumann regular ring([15]). In this section, we give a quick review of boolean Gröbner bases. For the proofs and more detailed descriptions, refer to [15] or [11].

We concentrate on boolean rings. In what follows, we assume that some term order on a set of power products of variables is given. For a polynomial f in a polynomial ring $\mathbf{B}[X_1, \ldots, X_n]$ $(= \mathbf{B}[\bar{X}])$ over a boolean ring $\mathbf{B}$, we use the notations $LT(f)$, $LM(f)$ and $LC(f)$ to denote the leading power product, the leading monomial and leading coefficient of f respectively. $f - LM(f)$ is also denoted by $Rd(f)$. We also use the notations $LT(F)$ and $LM(F)$ to denote the sets $\{LT(f)|f \in F\}$ and $\{LM(f)|f \in F\}$ for a (possibly infinite) subset F of $\mathbf{B}[\bar{X}]$. $T(\bar{X})$ denotes the set of power products consisting of variables $\bar{X}$.

Definition 9. *For an ideal I of a polynomial ring $\mathbf{B}[\bar{X}]$, a finite subset G of I is called a* Gröbner basis *of I if $\langle LM(I)\rangle = \langle LM(G)\rangle$.*

Definition 10. *For a polynomial $f \in \mathbf{B}[\bar{X}]$, let $a = LC(f)$, $t = LT(f)$ and $h = Rd(f)$. A* monomial reduction $\rightarrow_f$ *by f is defined as follows:*

$$bts + p \rightarrow_f (1 - a)bts + absh + p.$$

(*Note that* $(bts + p) - ((1 - a)bts + absh + p) = bs(af)$.)

Where s is a term of $T(\bar{X})$, b is an element of $\mathbf{B}$ such that $ab \neq 0$ and p is any polynomial of $\mathbf{B}[\bar{X}]$. For a set $F \subseteq \mathbf{B}[\bar{X}]$, we write $g \to_F g'$ if and only if $g \to_f g'$ for some $f \in F$. A recursive closure of $\to_F$ is denoted by $\overset{*}{\to}_F$, i.e. $g \overset{*}{\to}_F g'$ if and only if $g = g'$ or there exist a sequence of monomial reductions $g \to_F g_1 \to_F \cdots \to_F g_n \to_F g'$.

Theorem 11. *When F is finite, $\to_F$ is noetherian, that is there is no infinite sequence of polynomials $g_1, g_2, \ldots$ such that $g_i \to_F g_{i+1}$ for each $i = 1, 2, \ldots$.*

Theorem 12. *Let I be an ideal of a polynomial ring $\mathbf{B}[\bar{X}]$.*
A finite subset G of I is a Gröbner basis of I if and only if $\forall h \in I\ h \overset{*}{\to}_G 0$.

Using our monomial reductions, a reduced Gröbner basis is defined exactly same as in a polynomial ring over a field. A Gröbner basis G is *reduced* if each polynomial of G is not reducible by a monomial reduction of any other polynomial of G. In a polynomial ring over a field, a reduced Gröbner basis is uniquely determined. In our case, however, this property does not hold.

Example 1. *Let $\mathbf{B} = \mathbb{GF}_2 \times \mathbb{GF}_2$. In a polynomial ring $\mathbf{B}[X]$, $\{(1,0)X, (0,1)X\}$ and $\{(1,1)X\}$ are both reduced Gröbner bases of the same ideal.*

In order to have a unique Gröbner basis, we need one more definition.

Definition 13. *A reduced Gröbner basis G is said to be* stratified *if G does not contain two polynomials which have the same leading power product.*

Theorem 14. *If G and G' are stratified Gröbner bases of the same ideal w.r.t. some term order, then $G = G'$.*

In the above example, $\{(1,1)X\}$ is the stratified Gröbner basis, but the other is not.

Definition 15. *For a polynomial f, $LC(f)f$ is called a* boolean closure *of f, and denoted by $bc(f)$. If $f = bc(f)$, f is said to be* boolean closed.

Theorem 16. *Let G be a Gröbner basis of an ideal I, then $\{bc(g) | g \in G\} \setminus \{0\}$ is also a Gröbner basis of an ideal I.*

S-polynomial is also defined similarly as in a polynomial ring over a field.

Definition 17. *Let $f = atr + f'$ and $g = bsr + g'$ be polynomials where $a = LC(f)$, $b = LC(g)$, $tr = LT(f)$ and $sr = LT(g)$ for some power product t, s, r such that $GCD(t, s) = 1$, i.e. t and s do not contain a common variable. The polynomial $bsf + atg = bsf' + atg'$ is called an S-*polynomial *of f and g and denoted by $S(f, g)$.*

As in a polynomial ring over a field, the following property is crucial for the construction of Gröbner bases.

Theorem 18. *Let G be a finite set of polynomials such that each element of G is boolean closed. Then, G is a Gröbner basis if and only if $S(f, g) \overset{*}{\to}_G 0$ for any pair f, g of G.*

For any given finite set F, using our monomial reductions, we can always construct a Gröbner basis of $\langle F \rangle$ with computing boolean closures and S-polynomials by the following algorithms. It is also easy to construct a stratified Gröbner basis from a Gröbner basis.

Algorithm `BC`
Input: F a finite subset of $\mathbf{B}[\bar{X}]$
Output: F' a set of boolean closed polynomials such that $\langle F' \rangle = \langle F \rangle$
`begin`
$F' = \emptyset$
`while there exists a polynomial` $f \in F$ `which is not boolean closed`
$F = F \cup \{bc(f) - f\} \setminus \{f\}$, $F' = F' \cup \{bc(f)\}$
`end.`

Algorithm `GBasis`
Input: F a finite subset of $\mathbf{B}[\bar{X}]$, $>$ a term order of $T(\bar{X})$
Output: G a Gröbner basis of $\langle F \rangle$ w.r.t. $>$
`begin`
G `= BC(`F`)`
`while there exists two polynomials` $p, q \in G$ `such that` $S(p,q) \xrightarrow{*}_G h$
`for some non-zero polynomial` h `which is irreducible by` $\rightarrow_G$
G `=` $G \cup$ `BC(`$\{h\}$`)`
`end.`

Since any element of a boolean ring is idempotent, a boolean polynomial ring is more natural to work on. We can also define Gröbner bases in boolean polynomial rings.

A power product $X_1^{l_1} \cdots X_n^{l_n}$ is called a *boolean power product* if each l_i is either 0 or 1. The set of all boolean power products consisting of variables $\bar{X}$ is denoted by $BT(\bar{X})$. A boolean polynomial $f(\bar{X})$ in $\mathbf{B}(\bar{X})$ is uniquely represented by $b_1 t_1 + \cdots + b_k t_k$ with elements $b_1, \ldots, b_k$ of $\mathbf{B}$ and distinct boolean power products $t_1, \ldots, t_k$. We call $b_1 t_1 + \cdots + b_k t_k$ the *canonical representation* of $f(\bar{X})$. Since $BT(\bar{X})$ is a subset of $T(\bar{X})$, a term order $\geq$ on $T(\bar{X})$ is also defined on $BT(\bar{X})$. Given such a term order $\geq$, we use the same notations $LT(f), LM(f), LC(f)$ and $Rd(f)$ as before, which are defined by using its canonical representation. We also use the same notations $LT(F)$ and $LM(F)$ for a set F of boolean polynomials as before.

Definition 19. *For an ideal I of a boolean polynomial ring $\mathbf{B}(\bar{X})$, a finite subset G of I is called a* boolean Gröbner basis *of I if $\langle LM(I) \rangle = \langle LM(G) \rangle$ in $\mathbf{B}(\bar{X})$.*

Using canonical representations of boolean polynomials, we can also define monomial reductions for boolean polynomials as Definition 10 and have the same property of Theorem 12. The boolean closure of a boolean polynomial is also similarly defined as Definition 15 and the same property of Theorem 16 holds.

We can also define a stratified boolean Gröbner basis as in Definition 13, which is unique w.r.t. a term order. Construction of a boolean Gröbner basis is very simple. Given a finite set of boolean polynomials $F \subseteq \mathbf{B}(\bar{X})$. Compute a Gröbner basis G of the ideal $\langle F \cup \{X_1^2 - X_1, \ldots, X_n^2 - X_n\}\rangle$ in $\mathbf{B}[\bar{X}]$ w.r.t. the same term order. Then, $G \setminus \{X_1^2 - X_1, \ldots, X_n^2 - X_n\}$ is a boolean Gröbner basis of $\langle F\rangle$ in $\mathbf{B}(\bar{X})$. If G is stratified, then $G \setminus \{X_1^2 - X_1, \ldots, X_n^2 - X_n\}$ is also stratified.

Example 2. *The following left constraint with unknown set variables X and Y and an unknown element variable a is equivalent to the right system of equations of a boolean polynomial ring $\mathbf{B}(X, Y, A)$, where $\mathbf{B}$ is a boolean ring of sets and the variable A stands for the singleton $\{a\}$.*

$$\begin{cases} X \cup Y \subseteq \{1,2\} \\ 1 \in X \\ a \in Y \\ X \cap Y = \emptyset \end{cases} \iff \begin{cases} (XY + X + Y) + \{1,2\}(XY + X + Y) = 0 \\ \{1\}X + \{1\} = 0 \\ AY + A = 0 \\ XY = 0 \end{cases}$$

The stratified boolean Gröbner basis G of the ideal

$$I = \langle (XY + X + Y) + \{1,2\}(XY + X + Y), \{1\}X + \{1\}, AY + A, XY\rangle$$

w.r.t. a lexicographic term order $X > Y > A$ has the following form:

$$G = \{\{2\}XY, \{2\}YA + \{2\}A, (1+\{2\})Y, \{2\}XA, (1+\{2\})X + \{1\}, (1+\{2\})A\}.$$

From this we can get the elimination ideal $I \cap \mathbf{B}(A) = \langle (1 + \{2\})A\rangle$. By boolean extension theorem, we can see that the given constraint is satisfiable if and only if the element variable a satisfies the equation $(1 + \{2\})\{a\} = 0$ that is $a = 2$.

We conclude this section with the following theorem, which is essentially a special instance of Theorem 2.3 of [15].

Definition 20. *Let $\mathbf{B}$ be a boolean ring and k be a natural number. $\mathbf{B}^k$ denotes a direct product, i.e. the set of all k-tuples of elements of $\mathbf{B}$. For an element p of $\mathbf{B}^k$, $p_i \in \mathbf{B}$ denotes the i-th element of p for each $i = 1, \ldots, k$. If we define $p+q$ and $p \cdot q$ for $p, q \in \mathbf{B}^k$ by $(p+q)_i = p_i + q_i$ and $(p \cdot q)_i = p_i \cdot q_i$ for each $i = 1, \ldots, k$, $\mathbf{B}^k$ also becomes a boolean ring. For a polynomial $f(\bar{X})$ in $\mathbf{B}^k[\bar{X}]$ $f_i (i = 1, \ldots, k)$ denotes the polynomial in $\mathbf{B}[\bar{X}]$ obtained by replacing each coefficient p of f by p_i. For a boolean polynomial $f(\bar{X})$ in $\mathbf{B}^k(\bar{X})$, a boolean polynomial f_i in $\mathbf{B}(\bar{X})$ is defined similarly.*

Theorem 21. *In a polynomial ring $\mathbf{B}^k[\bar{X}]$, let G be a finite set of boolean closed polynomials. Then, G is a (reduced) Gröbner basis of an ideal I if and only if $G_i = \{g_i | g \in G\} \setminus \{0\}$ is a (reduced) Gröbner basis of the ideal $I_i = \{f_i | f \in I\}$ in $\mathbf{B}[\bar{X}]$ for each $i = 1, \ldots, k$.*

Corollary 22. *In a boolean polynomial ring $\mathbf{B}^i(\bar{X})$, let G be a finite set of boolean closed boolean polynomials. Then, G is a (reduced) boolean Gröbner basis of an ideal I if and only if $G_i = \{g_i | g \in G\} \setminus \{0\}$ is a (reduced) Gröbner basis of the ideal $I_i = \{f_i | f \in I\}$ in $\mathbf{B}(\bar{X})$ for each $i = 1, \ldots, k$.*

4 Comprehensive Boolean Gröbner Bases

In a polynomial ring over a field, construction of a comprehensive Gröbner basis is not so simple in general. In order to get a uniform (with respect to parameters) representation of reduced Gröbner bases, we need to divide a parameter space into several partitions according to the conditions that parameters satisfy. (See [2,4,5,13,14,16].)

In our boolean polynomial ring, however, we can always construct a stratified comprehensive boolean Gröbner basis. We do not even need to divide a parameter space.

In this section, we present a naive method to construct comprehensive boolean Gröbner bases. In what follows, we use variables $\bar{A} = A_1, \ldots, A_m$ for parameters and variables $\bar{X} = X_1, \ldots, X_n$ for main variables. We also assume that some term order on $T(\bar{X})$ is given.

Definition 23. *Let $F = \{f_1(\bar{A}, \bar{X}), \ldots, f_l(\bar{A}, \bar{X})\}$ be a finite subset of a boolean polynomial ring $\mathbf{B}(\bar{A}, \bar{X})$. A finite subset $G = \{g_1(\bar{A}, \bar{X}), \ldots, g_k(\bar{A}, \bar{X})\}$ of $\mathbf{B}(\bar{A}, \bar{X})$ is called a* comprehensive boolean Gröbner basis *of F, if $G(\bar{a}) = \{g_1(\bar{a}, \bar{X}), \ldots, g_k(\bar{a}, \bar{X})\} \setminus \{0\}$ is a boolean Gröbner basis of the ideal $\langle F(\bar{a})\rangle = \langle f_1(\bar{a}, \bar{X}), \ldots, f_l(\bar{a}, \bar{X})\rangle$ in $\mathbf{B}'(\bar{X})$ for any boolean extension $\mathbf{B}'$ of $\mathbf{B}$, i.e. a boolean ring which includes $\mathbf{B}$ as a subring, and any $\bar{a} = (a_1, \ldots, a_m) \in \mathbf{B}'^m$. G is also said to be* stratified *if $G(\bar{a})$ is stratified for any $\bar{a} = (a_1, \ldots, a_m) \in \mathbf{B}'^m$.*

Theorem 24. *Let $F = \{f_1(\bar{A}, \bar{X}), \ldots, f_l(\bar{A}, \bar{X})\}$ be a finite subset of a boolean polynomial ring $\mathbf{B}(\bar{A}, \bar{X})$. Considering $\mathbf{B}(\bar{A}, \bar{X})$ as a boolean polynomial ring $(\mathbf{B}(\bar{A}))(\bar{X})$ with the coefficient boolean ring $\mathbf{B}(\bar{A})$, let $G = \{g_1(\bar{A}, \bar{X}), \ldots, g_k(\bar{A}, \bar{X})\}$ be a (stratified) boolean Gröbner basis of the ideal $\langle F\rangle$ in this polynomial ring. Then G becomes a (stratified) comprehensive boolean Gröbner basis of F.*

Proof. Let $\bar{a} = a_1, \ldots, a_m$ be an arbitrary m-tuple of elements of $\mathbf{B}'$. Note that the specialization of parameters $\bar{A}$ with $\bar{a}$ induces a homomorphism from $\mathbf{B}(\bar{A}, \bar{X})$ to $\mathbf{B}'(\bar{X})$. We clearly have $\langle F(\bar{a})\rangle = \langle G(\bar{a})\rangle$ in $\mathbf{B}'(\bar{X})$.

If $f(\bar{A}, \bar{X}) \to_{g(\bar{A},\bar{X})} h(\bar{A}, \bar{X})$ in $(\mathbf{B}(\bar{A}))(\bar{X})$, then $f(\bar{A}, \bar{X}) = p(\bar{A})ts + f'(\bar{A}, \bar{X})$, $g(\bar{A}, \bar{X}) = q(\bar{A})t + g'(\bar{A}, \bar{X})$ and $h(\bar{A}, \bar{X}) = (1 - q(\bar{A}))p(\bar{A})ts + q(\bar{A})p(\bar{A})sg'(\bar{A}, \bar{X}) + f'(\bar{A}, \bar{X})$ for some $t, s \in T(\bar{X})$ and $p(\bar{A}), q(\bar{A}) \in \mathbf{B}(\bar{A})$ and $f'(\bar{A}, \bar{X})$, $g'(\bar{A}, \bar{X}) \in \mathbf{B}(\bar{A}, \bar{X})$, where $q(\bar{A})t$ is the boolean leading monomial of $g(\bar{A}, \bar{X})$. In case $q(\bar{a})p(\bar{a}) \neq 0$, certainly $q(\bar{a}) \neq 0$ and $p(\bar{a}) \neq 0$, so $q(\bar{a})t$ is the boolean leading monomial of $g(\bar{a}, \bar{X})$ and $p(\bar{a})ts$ is a monomial of $f(\bar{A}, \bar{X})$ and $f(\bar{a}, \bar{X}) \to_{g(\bar{a},\bar{X})} h(\bar{a}, \bar{X})$. Otherwise, $h(\bar{a}, \bar{X}) = f(\bar{a}, \bar{X})$. In either case, we have $f(\bar{a}, \bar{X}) \overset{*}{\to}_{g(\bar{a},\bar{X})} h(\bar{a}, \bar{X})$. Therefore, if $f(\bar{A}, \bar{X}) \to_G h(\bar{A}, \bar{X})$ in $(\mathbf{B}(\bar{A}))(\bar{X})$, then we have $f(\bar{a}, \bar{X}) \overset{*}{\to}_{G(\bar{a})} h(\bar{a}, \bar{X})$ in $\mathbf{B}'(\bar{X})$. Any boolean polynomial in the ideal $\langle F(\bar{a})\rangle$ is equal to $f(\bar{a}, \bar{X})$ for some boolean polynomial $f(\bar{A}, \bar{X})$ in the ideal $\langle F\rangle$ of $(\mathbf{B}(\bar{A}))(\bar{X})$. Since G is a boolean Gröbner basis of $\langle F\rangle$, we have $f(\bar{A}, \bar{X}) \overset{*}{\to}_G 0$. By the above observation, we have $f(\bar{a}, \bar{X}) \overset{*}{\to}_{G(\bar{a})} 0$. This shows that G is a comprehensive boolean Gröbner basis of F.

Suppose G is stratified, then any element g of G is boolean closed.

So, if $LC(g)(\bar{a}) = 0$, then $g(\bar{a}, \bar{X})$ must be equal to 0. Therefore, unless $g(\bar{a}, \bar{X}) = 0$, we have $LT(g(\bar{a}, \bar{X})) = LT(g(\bar{A}, \bar{X}))$. Now it is clear that $G(\bar{a})$ is stratified. □

Example 3. *For the same example of Example 2, the stratified boolean Gröbner basis of I in the boolean polynomial ring $(\mathbf{B}(A))(X, Y)$ has the following form: $\{(\{2\}A+\{2\})XY, (1+A+\{2\})X+\{1\}A+\{1\}, (1+A+\{2\})Y+\{2\}A, (1+\{2\})A\}$. From this, we can get the elimination ideal $I \cap \mathbf{B}(A) = \langle(1 + \{2\})A\rangle$. Moreover, if we specialize the variable A with $\{a\}$, it becomes the stratified boolean Gröbner basis $\{X + \{1\}, Y + \{2\}\}$.*

5 New Algorithm for Elimination Ideals

In section 2, we saw the importance of ideals in boolean polynomial rings. For a system of boolean equations given in a form of

$$\begin{cases} f_1(X_1, X_2, \ldots, X_n) = 0 \\ \qquad\qquad\qquad \vdots \qquad\qquad\qquad \cdots \qquad (1) \\ f_l(X_1, X_2, \ldots, X_n) = 0 \end{cases}$$

with boolean polynomials $f_1(\bar{X}), f_2(\bar{X}), \ldots, f_l(\bar{X})$ of $\mathbf{B}(\bar{X})$, we can solve it by computing a stratified boolean Gröbner basis of the ideal $I = \langle f_1(\bar{X}), f_2(\bar{X}), \ldots, f_l(\bar{X})\rangle$ w.r.t. a certain term order. We can also solve many problems concerning it. For example, if we want to decide whether a given boolean polynomial $h(\bar{X})$ vanishes on every solutions, what we have to do is computing a boolean Gröbner basis G of I(w.r.t. any term order) and checking the normal form of $h(\bar{X})$ by $\rightarrow_G$ is 0 or not. In any case, as long as we can compute a boolean Gröbner basis of I, we are almost done. Unfortunately, however, Gröbner bases computations are getting heavier when the number of variables increases. If we are interested in only solutions of some restricted variables, say X_1, X_2, X_3 for example, we do not necessarily need a boolean Gröbner basis of the whole ideal I, what we need is a boolean Gröbner basis of the elimination ideal $\langle f_1(\bar{X}), f_2(\bar{X}), \ldots, f_l(\bar{X})\rangle \cap \mathbf{B}(X_1, X_2, X_3)$. If we want to know whether a given polynomial $h(X_1, X_2, X_3)$ consisting of only three variables X_1, X_2, X_3 vanishes on every solution of (1), what we need is not a boolean Gröbner basis of the whole ideal but a boolean Gröbner basis of the above elimination ideal. A Gröbner basis of such an elimination ideal is usually obtained by computing a Gröbner basis of the whole ideal w.r.t. a block order $X_1, X_2, X_3 << X_4, \ldots, X_n$, i.e. a term order such that each variable X_1, X_2, X_3 is lexicographically less than the other variables.

In this section, we give an algorithm to compute a boolean Gröbner basis of an elimination ideal without computing a boolean Gröbner basis of the whole ideal. The next lemma is an easy but important key fact for our algorithm.

Lemma 25. *Let I be an ideal of a boolean polynomial ring $\mathbf{B}(\bar{A}, \bar{X})$ with variables $\bar{A}$ and $\bar{X}$, G be a stratified boolean Gröbner basis of I in a boolean polynomial ring $(\mathbf{B}(\bar{A}))(\bar{X})$ w.r.t. some term order of $T(\bar{X})$. Then $G \cap \mathbf{B}(\bar{A})$ is either an empty set or a singleton $\{h(\bar{A})\}$ of a boolean polynomial of $\mathbf{B}(\bar{A})$. In the latter case, the elimination ideal $I \cap \mathbf{B}(\bar{A})$ is equal to $\langle h(\bar{A}) \rangle$, otherwise it is equal to the trivial ideal $\{0\}$ in $\mathbf{B}(\bar{A})$.*

If the computation of such a G were faster than the computation of a boolean Gröbner basis of I in the boolean polynomial ring $\mathbf{B}(\bar{A}, \bar{X})$ w.r.t. a block order $\bar{A} << \bar{X}$, it would give us a more efficient algorithm to compute a boolean Gröbner basis of the elimination ideal. Unfortunately, however, a naive method described in section 4 is much slower than the computation of a boolean Gröbner basis of I w.r.t. such a block order in general. Though there is a work concerning an efficient computation algorithm for such a G, it is based on the computation of a boolean Gröbner basis of the whole ideal w.r.t. a block order([7]).

In this section, we give a new approach to compute an elimination ideal $I \cap \mathbf{B}(\bar{A})$. The key fact is the following lemma concerning the structure of a boolean polynomial ring $\mathbf{B}(\bar{A})$.

Lemma 26. *A boolean polynomial ring $\mathbf{B}(A_1, \ldots, A_m)$ is isomorphic to the direct product $\mathbf{B}^{2^m}$. An isomorphism ϕ from $\mathbf{B}(A_1, \ldots, A_m)$ to $\mathbf{B}^{2^m}$ is given by $\phi(f(A_1, \ldots, A_m))_i = f(c_1^i, \ldots, c_m^i)$ for each $i = 1, \ldots, 2^m$, where $c_1^i \cdots c_m^i$ is a binary number representation of $i-1$. The inverse of ϕ is given by $\phi^{-1}((a_1, a_2, \ldots, a_{2^m})) = \sum_{i=1}^{2^m} a_i(A_1 + c_1^i + 1)(A_2 + c_2^i + 1) \cdots (A_m + c_m^i + 1)$. (Note that $c_k^i + 1 = 1$ if $c_k^i = 0$ and $c_k^i + 1 = 0$ if $c_k^i = 1$.)*

Proof. Proof is by induction on m. We first show the lemma for $m = 1$. Any boolean polynomial of $\mathbf{B}(A_1)$ has a form $aA_1 + b$ for some elements a and b of $\mathbf{B}$. By the definition, $\phi(aA_1 + b))_1 = b$ and $\phi(aA_1 + b))_2 = a + b$, that is $\phi(aA_1 + b) = (b, a + b)$. It is easy to check that ϕ is a homomorphism. It is also obvious that ϕ is a bijection. Let $m > 1$ and assume that the lemma holds for $m - 1$. Note that any element of $\mathbf{B}(A_1, \ldots, A_m)$ is uniquely represented as $f(A_2, \ldots, A_m)A_1 + g(A_2, \ldots, A_m)$ with some elements $f(A_2, \ldots, A_m)$ and $g(A_2, \ldots, A_m)$ of $\mathbf{B}(A_2, \ldots, A_m)$. Considering a boolean polynomial ring $\mathbf{B}(A_1, \ldots, A_m)$ as a boolean polynomial ring $(\mathbf{B}(A_2, \ldots, A_m))(A_1)$, it is isomorphic to $\mathbf{B}(A_2, \ldots, A_m)^2$ with an isomorphism θ such that $\theta(f(A_2, \ldots, A_m)A_1 + g(A_2, \ldots, A_m)) = (g(A_2, \ldots, A_m), f(A_2, \ldots, A_m) + g(A_2, \ldots, A_m)$ by the first assertion we have shown above. By the assumption, we have an isomorphism ψ from $\mathbf{B}(A_2, \ldots, A_m)$ to $\mathbf{B}^{2^{m-1}}$ given by $\psi(f(A_2, \ldots, A_m))_i = f(c_2, \ldots, c_m)$ for each $i = 1, \ldots, 2^{m-1}$, where $c_2 \cdots c_m$ is a binary number representation of $i - 1$. Now, a map ϕ from $\mathbf{B}(A_1, \ldots, A_m)$ to $\mathbf{B}^{2^m}$ defined by $\phi(f(\bar{A})) = (\psi(\theta(f(\bar{A}))_1), \psi(\theta(f(\bar{A}))_2))$ (the concatenation of two 2^{m-1}-tuples $\psi(\theta(f(\bar{A}))_1)$ and $\psi(\theta(f(\bar{A}))_2)$) satisfies the property of the lemma.

The last assertion is obvious. □

This lemma together with Corollary 22 gives us a new algorithm to compute a boolean Gröbner basis of an elimination ideal.

Algorithm `ElimBGB`
Input: F a finite subset of $\mathbf{B}(\bar{A}, \bar{X})$,
$\bar{A}$ parameter variables, $>$ a term order of $T(\bar{A})$
Output: G a boolean Gröbner basis of the elimination ideal $\langle F\rangle \cap \mathbf{B}(\bar{A})$ w.r.t. $>$.

`begin`
F' = $\{\phi(f)|f \in F\}$
For $i = 1$ to 2^m, compute the stratified boolean Gröbner basis G_i of the ideal generated by $F'_i = \{f_i \in \mathbf{B}(\bar{A})|f \in F'\}$ w.r.t. any term order of $T(\bar{X})$.
For $i = 1$ to 2^m, if G_i contains a non-zero element of $\mathbf{B}$ then b_i = such an element, else $b_i = 0$.
G = a boolean Gröbner basis of the ideal $\langle \phi^{-1}((b_1, \ldots, b_{2^m}))\rangle$ w.r.t. $>$.
`end.`

In the above, ϕ is an isomorphism from $(\mathbf{B}(A_1, \ldots, A_m))(\bar{X})$ to $\mathbf{B}^{2^m}(\bar{X})$ obtained as an extension of the isomorphism defined in Lemma 26.

We implemented the algorithm for a boolean ring $\mathbf{B} = \{S \subseteq \mathcal{S}t | S$ is a finite or co-finite set$\}$, where $\mathcal{S}t$ denotes a countable set of all strings.

We show how our algorithm works using an example of our computation. In the following example, we have 30 variables $X_1, \ldots, X_{30}$. $a, b, \ldots, t$ denote strings, so $\{d, p\}, \{a, b\}, \ldots$ are elements of our $\mathbf{B}$.

Example 4. *Compute the eliminate portion* $\langle F\rangle \cap \mathbf{B}(X_1, X_2, X_3)$ *for* $F = \{f_1(\bar{X}), f_2(\bar{X}), \ldots, f_{18}(\bar{X})\}$ *with*

$f_1(\bar{X}) = X_1X_3X_{18} + \{d, p\}X_4X_{18}X_{20} + X_{11}X_{13} + \{a, b\}X_6X_{10} + X_5X_{18} + X_4,$
$f_2(\bar{X}) = X_2X_5 + X_4X_5 + \{k, q\}X_6X_8 + X_1X_{26}X_{27} + \{c, k\}X_4X_8 + X_{10} + X_6X_{10},$
$f_3(\bar{X}) = X_1X_2X_4 + X_3X_5X_{10} + \{d, i\}X_{11} + X_1 + X_5 + X_{12}X_{24}X_{28},$
$f_4(\bar{X}) = X_2X_4 + \{e, g\}X_3X_4 + X_1X_{12} + X_{12}X_{15} + X_5X_{10}X_{12} + X_3X_{11},$
$f_5(\bar{X}) = X_1X_3 + X_1X_5 + \{j, l\}X_2X_5X_{16} + X_{11}X_{12} + X_{11}X_{23} + X_{16} + 1,$
$f_6(\bar{X}) = X_6X_{17} + X_5X_9X_{30} + \{a, c\}X_8X_{10} + X_1X_{12} + X_{25}X_{29},$
$f_7(\bar{X}) = X_1X_{11} + X_{12} + X_2X_8 + X_3X_{11}X_{12} + X_{11}X_{12} + X_4X_6 + \{b, e\},$
$f_8(\bar{X}) = X_2 + X_4X_7 + \{c, f\}X_{12}X_{17}X_{21} + X_2X_3X_{12} + X_6X_7 + X_{12} + X_4X_{25} + X_1X_{11},$
$f_9(\bar{X}) = X_3 + X_3X_4 + \{k, m\}X_1X_3 + X_5X_6 + \{h, i\}X_7X_{24},$
$f_{10}(\bar{X}) = X_1 + \{g, i\}X_4X_5X_{11} + \{m, r\}X_1X_9X_{11} + X_2X_6 + X_{11} + 1,$
$f_{11}(\bar{X}) = X_3X_{20} + X_5 + X_7X_5 + X_{11} + \{l, o, s\}X_{13}X_{30} + X_{11}X_{18}X_{23},$
$f_{12}(\bar{X}) = X_3X_{14} + \{f, n, t\}X_1X_2 + X_2 + X_{11} + X_{11}X_{15} + X_{19}X_{22},$
$f_{13}(\bar{X}) = X_2X_7 + \{f, j\}X_{11} + X_2X_3 + X_{11}X_{12} + X_9X_{13} + X_{13},$
$f_{14}(\bar{X}) = X_3X_7 + X_8 + \{d, o\}X_8X_{13} + \{c, t\}X_2X_{23} + X_3X_{20}X_{22} + 1,$
$f_{15}(\bar{X}) = X_4X_9 + X_7X_{20} + \{b, l\}X_8X_{19} + X_{20},$
$f_{16}(\bar{X}) = \{a, e, n\}X_7X_9 + X_3X_5 + X_6X_{22} + \{e, r\}X_{18}X_{29} + X_{19}X_{21},$
$f_{17}(\bar{X}) = X_3 + X_{14} + X_{17}X_{18} + X_3X_4X_{19},$
$f_{18}(\bar{X}) = X_7 + X_7X_{21} + X_{23}X_{24}\}.$

We apply the algorithm `ElimBGB` *for* F *where* X_1, X_2, X_3 *are parameters and we use a purely lexicographic term order such that* $X_1 < X_2 < X_3$.

The isomorphism ϕ *from* $\mathbf{B}(X_1, X_2, X_3)$ *to* $\mathbf{B}^8$ *is given by* $\phi(f(X_1, X_2, X_3)) = (f(0,0,0), f(0,0,1), f(0,1,0), f(0,1,1), f(1,0,0), f(1,0,1), f(1,1,0), f(1,1,1))$.

$$
\begin{aligned}
F'_1 &= \{f_1(0,0,0,X_4,\ldots,X_{30}),\ldots,f_{18}(0,0,0,X_4,\ldots,X_{30})\}\\
F'_2 &= \{f_1(0,0,1,X_4,\ldots,X_{30}),\ldots,f_{18}(0,0,1,X_4,\ldots,X_{30})\}\\
&\vdots\\
F'_8 &= \{f_1(1,1,1,X_4,\ldots,X_{30}),\ldots,f_{18}(1,1,1,X_4,\ldots,X_{30})\}
\end{aligned}
$$

Computation of a stratified boolean Gröbner basis for each F'_i *yields*

$$b_1 = 0, b_2 = \{i\}, b_3 = \{k,q\}, b_4 = 0, b_5 = 0, b_6 = 0, b_7 = 0, b_8 = 0.$$

$$
\begin{aligned}
&\phi^{-1}((0,\{i\},\{k,q\},0,0,0,0,0)) =\\
&\quad \{i\}(X_1+1)(X_2+1)X_3 + \{k,q\}(X_1+1)X_2(X_3+1).
\end{aligned}
$$

We finally have a desired stratified boolean Gröbner basis

$$
\begin{aligned}
G = \{&\{i,k,q\}X_1X_2X_3 + \{i,k,q\}X_2X_3 + \{i\}X_1X_3 + \{i\}X_3\\
&+\{k,q\}X_1X_2 + \{k,q\}X_2\}.
\end{aligned}
$$

Total computation time is 274seconds by a PC with 1.7GHZ pentium-M CPU and 2GB SDRAM. Whereas, a boolean Gröbner basis computation w.r.t. any term order did not terminate in hours, a comprehensive Gröbner basis computation with parameters X_1, X_2, X_3 *did not either terminate in hours.*

In the algorithm `ElimBGB`, if we use a proper term order of $T(\bar{X})$ we can also get other eliminate portions. In the above example, we used a purely lexicographic term order such that $X_4 < X_5 < X_6 < \cdots < X_{30}$. From $G_1, G_2, \ldots, G_8$, for example, if we use $G_i \cap \mathbf{B}(X_4, X_5)$ instead of using a constant part b_i, we can get an elimination ideal $\langle F\rangle \cap \mathbf{B}(X_1, \ldots, X_5)$. From which, we can compute the stratified boolean Gröbner basis G of the elimination ideal $\langle F\rangle \cap \mathbf{B}(X_3, X_4, X_5)$ w.r.t. a purely lexicographic term order $X_3 < X_4 < X_5$.

$$
\begin{aligned}
G = \{&\{s,b,i,k,c,e,f,t,m,l\}X_3X_4X_5 + \{i\}X_4X_5 + \{s,b,k,c,e,f,t,m,l\}X_3X_5\\
&+\{s,b,i,k,c,e,f,t,m,l\}X_3X_4 + \{i\}X_4 + \{s,b,k,c,e,f,t,m,l\}X_3,\\
&\{o\}X_5X_4 + \{o\}X_3X_5 + \{o\}X_4 + \{o\}X_3,\\
&(1+\{s,b,i,k,c,h,e,f,t,m,l\})X_3X_4 + (1+\{s,b,i,k,c,h,e,f,t,m,l\})X_3\}
\end{aligned}
$$

6 Conclusions and Remarks

What the algorithm `ElimBG` computes is essentially a comprehensive boolean Gröbner basis of $\langle F\rangle$ with parameters $\bar{A}$. The possible values for each parameter are not only 0 and 1 but also all elements of $\mathbf{B}$. In the example of the last section, possible values for each parameter X_1, X_2, X_3 are all the subsets of $\{a, b, c, \ldots, s, t\}$ and their complements. So, there are $(2^{21})^3 = 2^{63}$-many possible cases for all specializations, where as there are only 8 cases for 0 and 1

specializations. In this sense, our algorithm is efficient. For m-many parameters, however, our algorithm needs computations of 2^m-many boolean Gröbner bases. This is of course infeasible when m is big, our method is effective only when m is small.

Complexity of boolean Gröbner bases computation is exponential in the number of variables in the worst case for both time and spaces. (So, the method based on computations of boolean Gröbner bases is not quite unreasonable, since it is an NP-hard problem to solve boolean equations.) Therefore, the method based on the computation of boolean Gröbner bases w.r.t. some block order or the naive method to compute comprehensive boolean Gröbner bases described in section 4 should be more efficient than our method at least from the theoretical point of view. (In case we have a parallel computation environment with enough computer resources, it does not apply.) Nevertheless, our (sequential) computation experiments show the efficiency of our method. In the experiments, we used randomly generated 32 sets of boolean polynomials with 20 to 30 variables with 5 parameters. For 15 examples, either a boolean Gröbner basis computation w.r.t. a block order or a naive comprehensive boolean Gröbner basis computation did not terminate in hours, whereas we successfully computed the elimination ideals by our method for all examples.

The most important reason we are working on boolean Gröbner bases is that they give us a *canonical form* of an ideal in a boolean polynomial ring. (See [11] for a canonical boolean Gröbner basis besides a stratified boolean Gröbner basis.) There are many other methods for solving boolean equations. Though we introduced our method for the computation of boolean Gröbner bases, it can be applied even for other computation methods for boolean equations such as an algorithm in [3].

The results shown in Section 2 are very old classical results. The proof of boolean strong Nullstellensatz is usually given by using Löwenheim's formula. We give a simple proof in this paper.

References

1. Buchberger, B.: Ein Algorithms zum Auffinden der Basiselemente des Restklassenrings bach einem nulldimensionalen Polynomial. Doctoral Dissertation Math. Inst. University of Innsbruck, Austria (1965)
2. Kapur, D.: An Approach for Solving Systems of Parametric Polynomial Equations. In: Saraswat, Van Hentenryck (eds.) Principles and Practices of Constraint Programming, pp. 217–244. MIT Press, Cambridge (1995)
3. Menju, S., Sakai, K., Sato, Y., Aiba, A.: A Study on Boolean Constraint Solvers. Constraint Logic Programming Selected Research, pp. 253–267. MIT Press, Cambridge (1993)
4. Montes, A.: A new algorithm for discussing Gröbner bases with parameters. J. Symb. Comp. 33(2), 183–208 (2002)
5. Manubens, M., Montes, A.: Improving DISPGB algorithm using the discriminant ideal. J. Symb. Comp. 41, 1245–1263 (2006)
6. Rudeanu, S.: Boolean functions and equations. North-Holland Publishing Co., Amsterdam. American Elsevier Publishing Co., Inc., New York (1974)

7. Sato, Y., Inoue, S.: On the Construction of Comprehensive Boolean Gröbner Bases. In: Proceedings of the Seventh Asian Symposium on Computer Mathematics (ASCM 2005), pp. 145–148 (2005)
8. Sakai, K., Sato, Y.: Boolean Gröbner bases. ICOT Technical Momorandum 488 (1988), http://www.icot.or.jp/ARCHIVE/Museum/TRTM/tm-list-E.html
9. Sakai, K., Sato, Y., Menju, S.: Boolean Gröbner bases (revised). ICOT Technical Report 613 (1991), http://www.icot.or.jp/ARCHIVE/Museum/TRTM/tr-list-E.html
10. Sato, Y., et al.: Set Constrains Solvers (Prolog version) (1996), http://www.icot.or.jp/ARCHIVE/Museum/FUNDING/funding-96-E.html
11. Sato, Y.: A new type of canonical Gröbner bases in polynomial rings over Von Neumann regular rings. In: Proceedings of ISSAC 1998, pp. 317–332. ACM Press, New York (1998)
12. Sato, Y., et al.: Set Constrains Solvers (Klic version) (1998), http://www.icot.or.jp/ARCHIVE/Museum/FUNDING/funding-98-E.html
13. Suzuki, A., Sato, Y.: An Alternative approach to Comprehensive Gröbner Bases. J. Symb. Comp. 36(3-4), 649–667 (2003)
14. Suzuki, A., Sato, Y.: A Simple Algorithm to Compute Comprehensive Gröbner Bases Using Gröbner Bases. In: International Symposium on Symbolic and Algebraic Computation (ISSAC 2006), Proceedings, pp. 326–331 (2006)
15. Weispfenning, V.: Gröbner bases in polynomial ideals over commutative regular rings. In: Davenport, J.H. (ed.) ISSAC 1987 and EUROCAL 1987. LNCS, vol. 378, pp. 336–347. Springer, Heidelberg (1989)
16. Weispfenning, V.: Comprehensive Gröbner bases. J. Symb. Comp. 14(1), 1–29 (1992)

Computer Search for Large Sets of Idempotent Quasigroups*

Feifei Ma[1,2] and Jian Zhang[1]

[1] State Key Laboratory of Computer Science
Institute of Software, Chinese Academy of Sciences
[2] Graduate University, Chinese Academy of Sciences
{maff,zj}@ios.ac.cn

Abstract. A collection of $n-2$ idempotent quasigroups of order n is called a large set if any two of them are disjoint, denoted by $LIQ(n)$. While the existence of ordinary $LIQ(n)$ has been extensively studied, the spectrums of large sets of idempotent quasigroups with various identities remain open, for example, large set of Steiner pentagon quasigroups of order 11 which is denoted by $LSPQ(11)$. This paper describes some computer searching efforts seeking to solve such problems. A series of results are obtained, including the non-existence of $LSPQ(11)$.

1 Introduction

The quasigroup problem has long been the focus of much interest in combinatorics. Many classes of finite quasigroups attract such attention partly because they are very natural objects in their own right and partly because they are correlative to design theory. A number of hard combinatorial problems are also raised by quasigroups. Over the last decade, the study of quasigroups has largely benefited from the improvement of automated reasoning techniques. Many open problems, to which the conventional mathematical methods are hard to apply, have been solved by means of computer search. For example, a series of open problems of the type from QG2 to QG9 were settled by several model generators such as *MGTP,FINDER,SEM* and the propositional satisfiability prover *SATO,DDPP* respectively [4,8,12,10,11]. Later, the non-existence of QG2(10), which used to be quite difficult, was established by Dubois et al. with their specific-purpose program *qgs* [3]. In fact, quasigroups with certain identities are highly structured. It is natural to translate the constraints for a quasigroup to logic formulae or model it as a constraint satisfaction problem, which can be effectively handled by these automatic tools.

A more challenging problem arising in this field is the large set problem for various idempotent quasigroups. Unlike the above problems, we need to find a set of quasigroups satisfying certain constraints rather than only one. Apparently this is more difficult since the search space to be explored is exponentially

* This work is supported by the National Natural Science Foundation of China (NSFC) under grant No. 60673044.

D. Kapur (Ed.): ASCM 2007, LNAI 5081, pp. 349–358, 2008.
© Springer-Verlag Berlin Heidelberg 2008

more huge. Lie Zhu summarized some open cases of such problems in [13] and [14]. Using the first order model generator SEM together with a novel searching strategy, we are able to solve some of the problems. This paper gives a summary of the experimental results and a brief description of the search method.

2 Preliminaries

2.1 Definitions

Firstly, let us recall some notations:

Definition 1 (Quasigroup). *A* quasigroup *is an ordered pair* $(Q, \oplus)$*, where* Q *is a set and* $\oplus$ *is a binary operation on* Q *such that the equations* $a \oplus x = b$ *and* $y \oplus a = b$ *are uniquely solvable for every pair of elements* a*,* b *in* Q*.*

For a finite set Q, the order of the quasigroup $(Q, \oplus)$ is denoted as $|Q|$.

A quasigroup $(Q, \cdot)$ is *idempotent* if the identity $x \cdot x = x$ (briefly $x^2 = x$) holds for all x in Q. We denote an idempotent quasigroup of order n as $IQ(n)$.

Two idempotent quasigroups $(Q, \oplus)$ and $(Q, \cdot)$ are said to be *disjoint* if for any $x, y \in Q$, $x \oplus y \neq x \cdot y$ whenever $x \neq y$.

Definition 2 (Large Set). *A collection of idempotent quasigroups* $(Q, \oplus_1)$*,* $(Q, \oplus_2)$*, ...,* $(Q, \oplus_{n-2})$*, where* $n = |Q|$*, is called a* large set *if any two of the idempotent quasigroups are disjoint.*

A large set of idempotent quasigroups of order n is denoted by $LIQ(n)$.

Definition 3 (Steiner Pentagon Quasigroup). *A quasigroup of order* n *is called a* Steiner pentagon quasigroup *if it satisfies the identities* $\{x^2 = x, (yx)x = y, x(yx) = y(xy)\}$*, denoted by* $SPQ(n)$*.*

Obviously a $SPQ(n)$ is a particular kind of $IQ(n)$. A large set of Steiner pentagon quasigroups of order n is denoted by $LSPQ(n)$.

2.2 The Problems

The existence of $LIQ(n)$ has already been established by Teirlinck and Lindner [9], and Chang [1]. In [1], Chang concluded that there exists an $LIQ(n)$ for any $n \geq 3$ with the exception $n = 6$. However, for IQs with certain identities, the spectrum of large sets has not been explored extensively so far. Generally there are two classes of IQs with which we are concerned in the paper. The first class includes 7 kinds of idempotent quasigroups whose existence has been studied systematically. These quasigroups satisfy following identities, respectively:

1. $xy \cdot yx = x$ Schröder quasigroup
2. $yx \cdot xy = x$ Stein's third law
3. $(xy \cdot y)y = x$ C_3-quasigroup
4. $x \cdot xy = yx$ Stein's first law; Stein quasigroup
5. $(yx \cdot y)y = x$

6. $yx \cdot y = x \cdot yx$ Stein's second law
7. $xy \cdot y = x \cdot xy$ Schröder's first law

Any short conjugate-orthogonal identity, if nontrivial, is conjugate-equivalent to one of them[2]. The large set of idempotent quasigroups satisfying property (i) of order n is denoted by $LIQ^{(i)}(n)$. Since for some orders, there is no IQ, Lie Zhu [14] summarized the open cases in the following list. They are of the moderate orders, therefore may be suitable for computer search.

Table 1. Open Cases for LIQ of Moderate Sizes

1. $LIQ^{(1)}(8)$ $LIQ^{(1)}(12)$ $LIQ^{(1)}(13)$
2. $LIQ^{(2)}(5)$ $LIQ^{(2)}(9)$ $LIQ^{(2)}(12)$
3. $LIQ^{(3)}(4)$ $LIQ^{(3)}(7)$ $LIQ^{(3)}(10)$ $LIQ^{(3)}(13)$
4. $LIQ^{(4)}(4)$ $LIQ^{(4)}(5)$ $LIQ^{(4)}(9)$ $LIQ^{(4)}(11)$
5. $LIQ^{(5)}(5)$ $LIQ^{(5)}(7)$ $LIQ^{(5)}(8)$ $LIQ^{(5)}(11)$
6. $LIQ^{(6)}(5)$ $LIQ^{(6)}(9)$ $LIQ^{(6)}(13)$
7. $LIQ^{(7)}(8)$ $LIQ^{(7)}(9)$ $LIQ^{(7)}(13)$

The second class of IQs is SPQ, which we already mentioned. It is known [6,13] that the spectrum for Steiner pentagon quasigroups is precisely the set of all positive integers $n \equiv 1$ or 5 (mod 10), except for $n = 15$. Zhu pointed out that the smallest unknown order for LSPQ is 11.

3 Search for LIQs of Small Orders

SEM is a general-purpose search program for finding finite models. It accepts a set of first order formulae as input and tries to find one or a specified number of models. The size of the model should be given by the user. Since the search is exhaustive, when the program terminates without finding any model, it means that there is no model of the given size.

With the help of SEM, we are able to perform the search for $LIQ^{(i)}(n)$. Without loss of generality, we assume the domain Q to be the set $\{0,1,\ldots,n-1\}$. A quasigroup is actually a function $f : Q \times Q \mapsto Q$ satisfying the constraints

$$\forall x \forall y \forall z (f(x,y) = f(x,z) \rightarrow y = z)$$

and

$$\forall x \forall y \forall z (f(x,y) = f(z,y) \rightarrow x = z)$$

Hence to find a large set of quasigroups with some identity is to determine $n-2$ such functions, namely $f_1, f_2, \ldots, f_{n-2}$, with extra constraint of the identity, and what's more, the disjoint constraints:

$$\forall x \forall y (f_i(x,y) = f_j(x,y) \rightarrow x = y)$$

where $i \neq j$. All of these constraints, together with the functions, form the input file for SEM.

We completed all the searches for $LIQ^{(i)}(n)$s with n no more than 8. The experiments were performed on an Intel 1.86GHZ 2CPU PC with Fedora 7 OS. The results are listed in Table 2. For each of the order 8 cases, the execution time of SEM ranges from 3 seconds to 3 minutes; for all other cases, the execution time is much less than a second. For each case where there is no LIQ, we further obtained the maximum number of disjoint quasigroups, denoted by $D^{(i)}(n)$, which is also useful in mathematics. When $D^{(i)}(n) = 1$, there are no disjoint $IQ^{(i)}(n)$s while $IQ^{(i)}(n)$s do exist. We have used Mace4 [7] to double check the results.

Table 2. Search Results for $LIQ^{(i)}(n)$s with $n \leq 8$

Identity i	Order n	Existence of LIQ	$D^{(i)}(n)$
1	8	YES	-
2	5	NO	2
3	4	YES	-
	7	NO	2
4	4	YES	-
	5	NO	1
5	5	NO	1
	7	NO	1
	8	NO	3
6	5	NO	2
7	8	YES	-

4 Search for $LSPQ(11)$

Intuitively, it seems infeasible to complete the computer search for $LSPQ(11)$ using the direct method. The highest order of $LSPQ(n)$ that has been completed by SEM is 10, but for order 11 it is much more difficult. Our experience is that, even searching for a partial model of 4 disjoint $SPQ(11)$s would take as long as 4 hours without any result. Nevertheless, via a tactical utilization of SEM, we are able to conclude the non-existence of $LSPQ(11)$. The effectiveness of our approach resides essentially in two strategies:

1. Constraint weakening.
2. Isomorphism elimination.

We know a **permutation** of a Domain Q is a one to one mapping (bijection) from Q onto itself. Let us denote a first order theory, i.e., a set of first order formulae by Σ. For two models M_1 and M_2 of Σ on Q, if there exists a permutation P that maps one of them to the other, then M_1 and M_2 are **isomorphic**.

Lemma 1. *Given a first order theory Σ and a finite domain Q. If there is no element in Q appearing in Σ as a constant, then for any model M of Σ on Q and any permutation P on Q, the new interpretation $P(M)$ obtained by performing P on M is also a model of Σ on Q.*

The lemma is straightforward since the assumption guarantees the interchangeablity of all elements in Q. Consequently, we have the following theorem which serves as the foundation in our approach:

Theorem 1. *Let S_n be the set of non-isomorphic $SPQ(n)$s. An $LSPQ(n)$, if there exists any, is isomorphic to an $LSPQ(n)$ containing at least one $SPQ(n)$ in S_n.*

Proof. First of all let us represent all the constraints for $LSPQ(n)$ by first order formulae and collect them in the set $\Sigma_{LSPQ(n)}$. $\Sigma_{LSPQ(n)}$ should be of the following form:

$$\begin{aligned}\Sigma_{LSPQ(n)} = \{ & \bigwedge_{1\le i\le n-2} \forall x\forall y\forall z(f_i(x,y)=f_i(x,z)\rightarrow y=z),\\ & \bigwedge_{1\le i\le n-2} \forall x\forall y\forall z(f_i(x,y)=f_i(z,y)\rightarrow x=z),\\ & \bigwedge_{1\le i\le n-2} \forall x f_i(x,x)=x,\\ & \bigwedge_{1\le i\le n-2} \forall x\forall y f_i(f_i(y,x),x)=y,\\ & \bigwedge_{1\le i\le n-2} \forall x\forall y f_i(x,f_i(y,x))=f_i(y,f_i(x,y)),\\ & \bigwedge_{1\le i<j\le n-2} \forall x\forall y(f_i(x,y)=f_j(x,y)\rightarrow x=y)\}\end{aligned}$$

Apparently $\Sigma_{LSPQ(n)}$ is a first order theory with no constant appearing in its formulae. Suppose there exists an $LSPQ(n)$, namely L. So L is a model of $\Sigma_{LSPQ(n)}$. Arbitrarily choose an $SPQ(n)$ from L, it must be isomorphic to some $SPQ(n)$ A in S_n since all non-isomorphic $SPQ(n)$s are included in S_n. Denote the permutation which maps the chosen one in L to A by P. Perform the permutation P on L and denote the result by $P(L)$, by lemma 1 $P(L)$ is a model of $\Sigma_{LSPQ(n)}$, i.e., an $LSPQ(n)$. What's more, A is contained in $P(L)$ because it is obtained by performing P on the originally chosen $SPQ(n)$, which is part of L. By definition L is isomorphic to $P(L)$, hence the theorem holds. □

Furthermore, it's observed that a large set of idempotent quasigroups has the following property:

Lemma 2. *Suppose an $LIQ(n)$=$\{(Q,\oplus_1), (Q,\oplus_2), \ldots, (Q,\oplus_{n-2})\}$, where $n = |Q|$. For any $x, y \in Q$, if $x \neq y$, then the collection of $x \oplus_i y$ is exactly the set Q excluding x and y, or formally, $\{x \oplus_i y | 1 \le i \le n-2\}$=$Q$-$\{x,y\}$.*

Proof. For any $1 \leq i \leq n-2$, from the idempotent identity we have $x \oplus_i x = x$ and $y \oplus_i y = y$. Since $(Q, \oplus_i)$ is a quasigroup and $x \neq y$, we have $x \oplus_i y \neq x$, otherwise the equation $x \oplus_i z = x$ with the unknown z is not uniquely solvable. Similarly, we get $x \oplus_i y \neq y$. Therefore, $x \oplus_i y \in Q - \{x, y\}$. Also, the disjoint property of large set implies that for any $i \neq j$, we have $x \oplus_i y \neq x \oplus_j y$. So the cardinality of $\{x \oplus_i y | 1 \leq i \leq n-2\}$ is $n-2$, equaling to the cardinality of Q-{x,y}. The two sets are equal. □

Now we explain the basic idea of our method. Suppose there are m non-isomorphic $SPQ(n)$s in total, namely $A_1, A_2, \ldots, A_m$. By Theorem 1 we know that $LSPQ(n)$ exists if and only if there is an $LSPQ(n)$ containing at least one such A_i. Therefore without losing any non-isomorphic solution, we can safely divide the search space into m (maybe intersecting) sub-spaces, each of which corresponds to an $A_i \in LSPQ(n)$. Now let's consider any of the m situations. While searching in the ith sub-space, the first $SPQ(n)$ is prefixed to be A_i. We are to determine the multiplication tables of the rest $n-3$ $SPQ(n)$s in the large set. We denote them by $f_2^i, f_3^i, \ldots, f_{n-2}^i$. For these $n-3$ SPQs, it doesn't make any difference how they are ordered. We can choose a cell which is not on the diagonal, for example, $cell(0,1)$, and sort them by the cell value. Lemma 2 implies that any $f_j^i(0,1)$ and $f_k^i(0,1)$ are different for $j \neq k$, thus we can fix

$$f_2^i(0,1) < f_3^i(0,1) < \ldots < f_{n-2}^i(0,1) \tag{1}$$

and $(n-3)! - 1$ isomorphisms are then eliminated. Also, if $n-3$ $SPQ(n)$s in the large set are determined, the candidate for the last one is unique and can be worked out immediately. This is because by Lemma 2, there is only one candidate value left for each $cell(x, y)$ where $x \neq y$, and the idempotent property forces each $cell(x, x)$ to be assigned x. What we need to do is to check if the candidate satisfies the identities of SPQ and it is quite an easy job. So the task is reduced to finding $n-4$ disjoint $SPQ(n)$s which are all disjoint with A_i, and satisfying (1).

Although the problem is much simplified, it remains intractable for the open case of order 11. We once tried one such subcase. SEM did not complete the search after running a week, and finally the process was killed. However, it is noticed that search for only one $SPQ(11)$ that is disjoint with A_i could be completed quite fast by SEM and it seems feasible to find all the solutions. If we weaken the disjoint constraints for f_j^i to be disjoint with A_i only, we can obtain the candidate set for f_j^i. Once all the candidates for $f_2^i, \ldots, f_{n-3}^i$ are found out, we can check if a large set can be formed.

Our method can be summarized as the following procedure:

1. Find the set of all non-isomorphic $SPQ(n)$s: $S_n = \{A_1, A_2, \ldots, A_m\}$. Currently we use SEMD [5].
2. For each A_i in S_n, do the following:
   2a Fix f_1^i to be A_i. For each f_j^i $(2 \leq j \leq n-3)$ use SEM to find all the candidates satisfying the following constraints:
      i. The SPQ(n) identities.
      ii. f_j^i(0,1) equals to the (j–1)th smallest value in the set Q–$\{0, 1, f_1^i(0,1)\}$.

iii. Disjoint with f_1^i.
and denote the candidate set by C_j^i.

2b Call the function `SCAN(i,n)` to find all the sets of $n-4$ SPQs which are disjoint with each other. If there are no such SPQs, try next A_i.

2c For each set of the disjoint SPQs calculate the possible solution for f_{n-2}^i and check if it is an $SPQ(n)$. If so, an $LSPQ(n)$ is discovered. If all the sets fail to produce an $LSPQ(n)$, try next A_i.

The program `SCAN(i,n)` is illustrated in Figure 1. It is a depth-first search program to find all combinations of $n-4$ disjoint SPQs. The $n-4$ SPQs are from $C_2^i, C_3^i, \ldots, C_{n-3}^i$ separately. The program works by scanning the SPQs in C_j^i $(2 \leq j \leq n-3)$ in sequence so as to pair up disjoint SPQs. The variable `cur` is the current search depth, i.e, the number of the candidate set being scanned. The current search path is kept in the array `pnt`. `C_i_j` is the array to store the SPQs in the candidate set C_j^i and `size[j]` is the cardinality of C_j^i. Initially `cur` is set to 3 and `pnt[j]` is set to 1.

If all $SPQ(n)$s in S_n have been tried and no $LSPQ(n)$ is found, the process terminates. Since the process is exhaustive, the non-existence of $LSPQ(n)$ can be established.

```
bool SCAN(i,n){
  sol_num=0;
  cur=3;
  while(TRUE){
    if(there exists some j, 1<j<cur, such that C_i_j[pnt[j]]
       and C_i_cur[pnt[cur]] are not disjoint)
       pnt[cur]++;
    else {if(cur==n-3){
             record pnt;
             sol_num++;
             pnt[cur]++;
          }
          else {cur++;
                pnt[cur]=1;
               }
     }
    while(pnt[cur]==size[cur]+1){
       cur--;
       if(cur<2){
          if(sol_num==0) return FALSE;
          return TRUE;
       }
       pnt[cur]++;
     }
  }
}
```

Fig. 1. The program `SCAN(i,n)`

A1	0	1	2	3	4	5	6	7	8	9	10
0	0	2	6	5	10	9	8	1	7	4	3
1	3	1	5	8	9	6	4	0	10	2	7
2	4	0	2	6	7	3	10	9	5	1	8
3	1	6	7	3	5	2	9	8	4	10	0
4	2	7	8	9	4	10	1	5	3	0	6
5	6	10	1	0	3	5	7	4	2	8	9
6	5	3	0	2	8	1	6	10	9	7	4
7	9	4	3	10	2	8	5	7	0	6	1
8	10	9	4	1	6	7	0	3	8	5	2
9	7	8	10	4	1	0	3	2	6	9	5
10	8	5	9	7	0	4	2	6	1	3	10

A2	0	1	2	3	4	5	6	7	8	9	10
0	0	2	7	5	6	8	9	10	1	3	4
1	3	1	5	6	8	9	2	4	0	10	7
2	4	0	2	8	5	3	1	6	10	7	9
3	1	4	9	3	10	2	8	5	7	0	6
4	2	3	8	9	4	10	7	1	6	5	0
5	6	7	1	0	2	5	10	3	9	4	8
6	5	9	10	1	0	7	6	2	4	8	3
7	8	5	0	10	9	6	4	7	3	2	1
8	7	10	4	2	1	0	3	9	8	6	5
9	10	6	3	4	7	1	0	8	5	9	2
10	9	8	6	7	3	4	5	0	2	1	10

Fig. 2. Two non-isomorphic $SPQ(11)$s

Following the instructions above, we performed the search for $LSPQ(11)$. Firstly, there are only two non-isomorphic $SPQ(11)$s found by SEM, denoted by A_1 and A_2 respectively. They are shown in Fig. 2.

The search for C_j^i, $1 \leq i \leq 2$, $2 \leq j \leq 8$ is performed on an IBM BladeCenter with 8 2.5GHz PowerPC 970 2CPU processors. The search results are listed in Table 3. The times are also given in seconds. From Table 3 we can see that the

Table 3. Experimental Result for C_j^i

C_j^i	j:	2	3	4	5	6	7	8
$i=1$	Solution Number	1055	940	980	979	1055	929	924
	Running time	3593.13	3046.69	3362.07	3448.14	3590.43	3207.82	2994.13
$i=2$	Solution Number	830	732	758	732	726	867	804
	Running time	3614.09	2893.40	3418.53	3254.60	3374.20	3362.34	3167.44

cardinality of each set C_j^i is about 1000 and the running time doesn't exceed one hour in general.

For each of the two subspaces, we use the program in Fig. 1 to find 7 disjoint $SPQ(11)$s from the candidate sets. After running about 1 minute on the PC, the program returned `FALSE` for both cases, implying that $LSPQ(11)$ doesn't exist.

5 Discussion

It is interesting to notice that adding some redundant lemmas can greatly influence the running time of SEM. A remarkable evidence is the self-orthogonal property of SPQ. An idempotent quasigroup $(Q, \oplus)$ is called **self-orthogonal** if

$$\{(x \oplus y, y \oplus x) | x, y \in Q, x \neq y\} = \{(u, v) | u, v \in Q, u \neq v\}.$$

The property can be represented by the conjunction of the following two first order formulae.

$$\forall x \forall y (x \neq y \rightarrow f(x, y) \neq f(y, x)) \tag{2}$$

$$\forall x\forall y\forall z\forall w(x \neq y \wedge z \neq w \wedge (x \neq z \vee y \neq w) \rightarrow f(x,y) \neq f(z,w) \vee f(y,z) \neq f(w,z)) \quad (3)$$

It is known that an $SPQ(n)$ is self-orthogonal. Formula 2 can reduce the running time greatly in our experiments. We once tried to search for $LSPQ(9)$ directly by SEM. It lasted for about 10 hours and still could not give a result. By contrast, when formula 2 is added to the input file, the search is completed within a second, confirming the non-existence of $LSPQ(9)$. It is also helpful to the generating process of C_j^i, reducing the running time from almost 24 hours to less than an hour. However, when formula 3 is also added, in both cases the search would slow down dramatically, almost as slow as when there are no lemmas. While short lemmas are helpful, long ones may play a negative role. This issue may deserve further investigation.

Although the method in section 4 is designed for LSPQ, it can be generalized to $LIQ^i(n)$s or any other LIQs with similar identities. In addition, it can be easily modified to find out the maximum disjoint IQs if necessary. The basic idea is also suggestive to the automated search for other algebraic structures. We have just applied the method to $LIQ^2(9)$, $LIQ^4(9)$, $LIQ^6(9)$ and $LIQ^7(9)$ and established their nonexistences. It is hopeful that more open problems regarding the existence of large set can be solved in the future.

6 Conclusion

Searching for large set of idempotent quasigroups (LIQs) presents new challenges to computer scientists and mathematicians. The search space is much larger in general. This paper presents some search results and techniques which appear to be effective on the problem. The preliminary results are quite interesting to Lie Zhu and provided useful information for his research.

To reduce the search time, we have used three techniques:(1) adding redundant formulae to the original set of constraints; (2) dividing the set of constraints into several subsets, and constructing a large set gradually by considering various pairs of quasigroups; (3) employing the symmetry among the quasigroups in the large set, in addition to the symmetry among domain elements.

While the above techniques have helped us to obtain some results on previously open cases, we are also developing other techniques and investigating more difficult cases. Moreover, we will use other model finding tools in the future.

Acknowledgement

The authors are very thankful to Lie Zhu for his introduction to the problems.

References

1. Chang, Y.: The Spectrum for Large Sets of Idempotent Quasigroups. Journal of Combinatorial Designs (2000)
2. Colbourn, C.J., Dinitz, J.H.: Handbook of Combinatorial Designs, 2nd edn. CRC Press, Boca Raton (2007)

3. Dubois, O., Dequen, G.: The Non-existence of (3,1,2)-Conjugate Orthogonal Idempotent Latin Square of Order 10. In: Proc. of the 7th International Conference on Principles and Practice of Constraint Programming (2001)
4. Fujita, M., Slaney, J.K., Bennett, F.: Automatic Generation of Some Results in Finite Algebra. In: Proc. IJCAI, pp. 52–59 (1993)
5. Jia, X., Zhang, J.: A Powerful Technique to Eliminate Isomorphism in Finite Model Search. In: Proc. IJCAR, pp. 318–331 (2006)
6. Lindner, C.C., Stinson, D.R.: Steiner Pentagon Systems. Discrete Math. 52, 67–74 (1984)
7. McCune, W.: Mace4 Reference Manual and Guide. Technical Memorandum 264, Argonne National Laboratory, Argonne, IL, USA (2003)
8. Slaney, J., Fujita, M., Stickel, M.: Automated Reasoning and Exhaustive Search: Quasigroup Existence Problems. Computers and Mathematics with Applications 29, 115–132 (1995)
9. Teirlinck, L., Lindner, C.C.: The Construction of Large Sets of Idempotent Quasigroups. Eur. J. of Combin. 9, 83–89 (1988)
10. Zhang, H.: SATO: An Efficient Propositional Prover. In: Proc. of CADE, pp. 272–275 (1997)
11. Zhang, H., Stickel, M.: Implementing the Davis-Putnam Method. Journal of Automated Reasoning 24(1/2), 277–296 (2000)
12. Zhang, J., Zhang, H.: SEM: a System for Enumerating Models. In: Proc. of International Joint Conference on Artificial Intelligence, pp. 11–18 (1995)
13. Zhu, L.: Large Set Problems for Various Idempotent Quasigroups (July 2006)
14. Zhu, L.: Personal Communication (September 2007)

Author Index