APPLICATIONS OF COMPUTER ALGEBRA TO ALGEBRAIC GEOMETRY, SINGULARITY THEORY AND SYMBOLIC–NUMERICAL SOLVING

GERT-MARTIN GREUEL

1. Introduction

Although computer algebra is a research field in its own, its main driving force comes from various fields of applications. These applications range from mathematics and computer science over physics and engineering up to applications to technical and industrial problems.

In this article I should like to show, by means of examples, the impact of computer algebra on two branches of pure mathematics: algebraic geometry and singularity theory. Today, algorithms, programmes and systems in computer algebra have reached a stage where it is possible to compute highly sophisticated mathematical objects such as moduli spaces and objects related to mixed Hodge structures (Sections 2 and 3). Moreover, computer algebra has been and is still successfully used in testing or disproving conjectures, or in computing interesting examples (Section 4). Finally, I shall report on recent experiments where different methods of computer algebra have been applied to symbolic–numerical solving of polynomial equations (Section 5), an important application of computer algebra to real life problems. Each of the sections contains a few unsolved problems, respectively projects, for further research.

The examples were chosen either from diploma theses of some of my students (T. Bayer, M. Schulze, M. Wenk), respectively from joint research projects together with C. Lossen and E. Shustin. All algorithms are implemented in the computer algebra system Singular [GPS]. They are mainly based on Gröbner basis methods which were foundationally developed by Buchberger [Bu1], [Bu2] for polynomial rings. Subsequently, they have been extended to local and "mixed" rings in [GP2] for use in singularity theory.

Gröbner basis computations are, nowadays, implemented in all major general purpose computer algebra systems such as the big–M–systems (`Magma, Maple, Mathematica, MuPad`) but also in special systems designed for use in commutative algebra and algebraic geometry (`CoCoA, Macaulay,` Singular). However, having just the possibility to compute Gröbner bases (w.r.t. a few monomial orderings) is, for applications to mathematical research problems, not much more than having the elementary numerical operations on a calculator for applications to engineering problems. Hence, I should like to emphasise the necessity to further develop packages and libraries to make the systems still more useful for the "working mathematician". Today, new and advanced algorithms can be built on already existing powerful procedures for computing, for example, free resolutions, Ext and Tor groups, sheaf cohomology, primary decomposition, ring normalisation, versal deformations, and many more (cf. [GPS]). The development of new algorithms provides, in addition, a better understanding and often even produces new theoretical insight, as has been the case, just to mention one example, for primary decomposition (cf. [GTZ], [EHV]). This has also been the case for some topics treated in the present article.

For more applications of computer algebra to algebraic geometry and singularity theory see [Gr2].

We assume the reader is familiar with the main notions of Gröbner bases (cf. [CLO1], [BW]).

I should like to thank T. Bayer, C. Lossen und M. Schulze for helping to prepare this article and M. Möller for useful comments concerning symbolic–numerical solving.

Development and implementation of algorithms on which this paper is based were supported by the DFG–Schwerpunkt "Effiziente Algorithmen für diskrete Probleme und ihre Anwendungen" and by the "Stiftung Rheinland–Pfalz für Innovation", which we kindly acknowledge.

2. Monodromy and Gauss–Manin connection

The monodromy of a morphism $f : X \to S$ between complex spaces or algebraic schemes/$\mathbb{C}$, which we suppose to be a differentiable fibre bundle outside the discriminant $\Delta \subset S$, describes the action of the fundamental group of $S \setminus \Delta$ on the cohomology $H^*(X_t, \mathbb{C})$ of the general fibre. The Gauß–Manin connection may be considered as an algebraic description of the monodromy action by means of differential forms. Finally, the mixed Hodge structure is an analytic structure on $H^*(X_t, \mathbb{C})$ generalising the Hodge decomposition of compact, smooth algebraic varieties. These concepts have many applications and were widely studied in the global situation for proper maps as well as in the local situation for isolated singularities, for a survey see [Ku]. Here we shall only consider the local case.

Let $f \in \langle x \rangle \subset \mathbb{C}\{x_0, \dots, x_n\}$ be a convergent power series (in practice a polynomial) with isolated singularity at 0 and $\mu = \dim_{\mathbb{C}} \mathbb{C}\{x\}/\langle f_{x_0}, \dots, f_{x_n} \rangle$ the Milnor number of f. Then f defines in an ε-ball B_ε around 0 a holomorphic function $f : B_\varepsilon \to \mathbb{C}$, and, by a theorem of Milnor, there exists a small δ–disc S_δ in $\mathbb{C}$ around 0 such that $f : B_\varepsilon \setminus X_0 \to S_\delta \setminus \{0\}$ is a C^∞–fibre bundle so that the general fibre $X_t = f^{-1}(t)$, $t \neq 0$, is homotopy equivalent to a bouquet of μ n–dimensional spheres.

The simple, counterclockwise path γ in S_δ around 0 induces a C^∞–diffeomorphism of X_t $(t \neq 0)$ and an automorphism T of the singular cohomology group $H^n(X_t, \mathbb{C})$ which is a μ–dimensional $\mathbb{C}$–vector space. The automorphism T is called the local **Picard–Lefschetz monodromy** of f. We address the problem of computing the eigenvalues and the Jordan normal form of T.

The first important theorem is the monodromy theorem, due to Deligne in the global and to Brieskorn in the local situation which says that the eigenvalues of T are roots of unity, that is, we have $T = e^{2\pi i M}$, where M is a complex matrix with eigenvalues in $\mathbb{Q}$.

Hence, we are left with the problem of computing the eigenvalues and the Jordan normal form of M. Since X_t is a complex Stein manifold, its complex cohomology can be computed, via the holomorphic de Rham theorem, by using holomorphic differential forms, which is the starting point of Brieskorn's algorithm for computing the monodromy. To cut a long story short, we just mention that the Brieskorn lattices (cf. [Br])

$$H' = \Omega^n / \left(df \wedge \Omega^{n-1} + d\Omega^{n-1} \right), \quad H'' = \Omega^{n+1}/df \wedge d\Omega^{n-1}$$

are free $\mathbb{C}\{t\}$–modules of rank μ. Here $(\Omega^\bullet, d)$ denotes the complex of holomorphic differential forms in $(\mathbb{C}^n, 0)$. We define the local **Gauß–Manin connection** of f as

$$\triangledown : df \wedge H' = df \wedge \Omega^n / df \wedge d\Omega^{n-1} \longrightarrow H'', \quad [df \wedge \omega] \longmapsto [d\omega].$$

Extending $\triangledown$ to an endomorphism of $H'' \otimes_{\mathbb{C}\{t\}} \mathbb{C}(t)$ and describing it with respect to a basis, we see immediately that the kernel of $\triangledown$, together with a basis of H'', is the same as the solutions of a rank μ system of ordinary differential equations

$$\frac{dy}{dt} = -Ay, \qquad A = (a_{ij}) = \sum_{i \geq -p} A_i t^i \in \mathrm{Mat}\big(\mu \times \mu, \mathbb{C}(t)\big),$$

in a neighbourhood of 0 in $\mathbb{C}$. The connection matrix A has a pole at $t = 0$ and is holomorphic for $t \neq 0$. If $\phi_t = (\phi_1, \ldots, \phi_\mu)$ is a fundamental system of solutions at a point $t \neq 0$, then the analytic continuation of ϕ_t along the path γ transforms ϕ_t into another fundamental system ϕ_t' which satisfies $\phi_t' = T_\triangledown \phi_t$ for some matrix $T_\triangledown \in \mathrm{GL}(\mu, \mathbb{C})$.

It is a fundamental fact that the Picard–Lefschetz monodromy T coincides with the monodromy $T_\triangledown$ of the Gauß–Manin connection.

Brieskorn [Br] used this fact to describe the essential steps for an algorithm to compute the characteristic polynomial of T. Results of Gerard and Levelt [GL] allowed the extension of this algorithm to compute the Jordan normal form of T. An implementation of Schulze in Singular is able to compute interesting examples (including the uni– and bimodal singularities, [Sch]).

The algorithm uses the regularity theorem which says that there exists a basis of some lattice in $H'' \otimes \mathbb{C}(t)$ such that the connection matrix A has a pole of order 1.

Basically, if $A = A_{-1}t^{-1} + A_0 + A_1 t + \ldots$ has a simple pole, then $T = e^{2\pi i A_{-1}}$ is the monodromy (this holds if the eigenvalues of A_{-1} do not differ by integers which can be achieved algorithmically).

Singular example for computing the monodromy (omitting the output):

```
> LIB "mondromy.lib";
> ring R = 0,(x,y),ds;
> poly f = x2y2+x6+y6;     //example of A'Campo (monodromy is not
> matrix M = monodromy(f); //diagonalisable)
> print(jordanform(M));    //prints Jordan normalform of monodromy
```

Ingredients for the implementation of Brieskorn's algorithm:

1. Computation of standard bases and normal forms for local orderings;
2. find k so that $f^k \in \langle f_{x_0}, \ldots, f_{x_n} \rangle$ and express f^k as linear combination of $f_{x_0}, \ldots, f_{x_n}$;
3. computation of the connection matrix on increasing lattices in $H'' \otimes \mathbb{C}(t)$ up to sufficiently high order (until saturation) by linear algebra over $\mathbb{Q}$;
4. computation of the transformation matrix to a simple pole by linear algebra over $\mathbb{Q}$.

The most expensive parts are certain normal form computations for a local ordering and the linear algebra part because here one has to deal iteratively with matrices with several thousand rows and columns.

In the remaining part of this section we describe a new algorithm, developed by M. Schulze, based on the theory of $\mathscr{D}$–modules ($\mathscr{D} = \mathbb{C}\{t\}[\partial_t]$): the complex $\Omega^\bullet[D]$ with differential $\mathbf{d}$ defined by

$$\mathbf{d}(\omega D^k) := d\omega D^k - df \wedge \omega D^{k+1}$$

is a complex of $\mathscr{D}$-modules with $\mathscr{D}$–action

$$\partial_t \omega D^k = \omega D^{k+1}\,, \qquad t\omega D^k = f\omega D^k - k\omega D^{k-1}.$$

The $\mathscr{D}$–module $\mathscr{H} := H^{n+1}(\Omega^\bullet[D], \mathbf{d}) = \Omega^{n+1}[D]/\mathbf{d}\Omega^n[D]$ is called the **Gauß–Manin system** of f. The operator ∂_t is invertible on $\mathscr{H}$. For $k \geq 0$, let

$$F_k\Omega^{n+1}[D] := \bigoplus_{i=0}^{k} \Omega^{n+1}D^i$$

and $F_k\mathscr{H}$ be the image of $F_k\Omega^{n+1}[D]$ under the canonical map $\Omega^{n+1}[D] \to \mathscr{H}$. This defines a filtration F on $\mathscr{H}$ called the **Hodge filtration**. By the De Rham and Poincaré lemma, $df \wedge H' = \partial_t^{-1}F_0\mathscr{H} \subset F_0\mathscr{H} = H''$. We denote by

$$\mathbb{C}\{\{\partial_t^{-1}\}\} := \left\{ \sum_{i\geq 0} a_i \partial_t^{-i} \in \mathbb{C}[[\partial_t^{-1}]] \;\middle|\; \sum_{i \geq 0} \frac{a_i}{i!} t^i \in \mathbb{C}\{t\} \right\},$$

the ring of **micro-differential operators** with constant coefficients and abbreviate $s := \partial_t^{-1}$. Then H'' is a free $\mathbb{C}\{\{s\}\}$–module of rank μ. By definition, $df \wedge \Omega^n \subset \Omega^{n+1}$ is isomorphic to the Jacobian ideal of f, and $\Omega_f = \Omega^{n+1}/df \wedge \Omega^n$ to the Milnor algebra. Using Gröbner basis methods, one can compute a monomial $\mathbb{C}$–basis $m = \{m_1, \ldots, m_\mu\}$ of Ω_f, inducing a section $v \in \mathrm{Hom}_{\mathbb{C}}(\Omega_f, H'')$ of the projection π and an isomorphism $\mathbb{C}\{\{s\}\}^\mu \cong H''$, by Nakayama's lemma. We define the matrix

$$B = \sum_{k \geq 0} B_k s^k \in \mathrm{Mat}\big(\mu \times \mu, \mathbb{C}\{\{s\}\}\big)$$

of multiplication by t this respect to m, i.e., $Bm := tm$. An easy computation shows that $B + s^2\partial_s$ is the basis representation of t with respect to m.

By definition of the differential $\mathbf{d}$, computing B up to order $k-1$ amounts to expressing k times an element of Ω^{n+1} in the basis m and $df \wedge \Omega^n$, which is the Jacobian ideal of f. This can be done using Gröbner basis methods. To do the k–th step in the computation of the saturation H''_∞ of H'', one has to compute B up to order k. To compute the residue of ∂_t on H''_∞, whose eigenvalues are the eigenvalues of monodromy, one has to compute B up to sufficiently high order and compute a $\mathbb{C}\{\{s\}\}$–basis of H''_∞ as well as the basis representation of the images of this basis under $\partial_t t$ with respect to this basis. This can also be done using Gröbner basis methods.

Compared to the Brieskorn algorithm, we have interchanged the roles of ∂_t^{-1} and t. The ∂_t^{-1}–structure of H'' is much more natural and there are many advantages of this new algorithm: There are no problems with estimations, no huge linear algebra problems, we need not to lift a power of f in the Jacobian ideal, the basis of H'' is easier to compute, and so on. The main point is that we can continue the computation when we have to increase the order of B. In the Brieskorn algorithm, we have to start again almost from the beginning. Nevertheless, the three components of this new algorithm explained above also require difficult computations, especially the first one. The new algorithm can be extended to compute the Jordan normal form of the monodromy in a similar way as it was done in [Sch].

Problems:

1. Generalise the algorithm of M. Schulze to isolated complete intersection singularities.
2. Find an algorithm to compute the V–filtration of the mixed Hodge structure of an isolated hypersurface singularity.
3. Compute the spectrum, resp. the spectral pairs, of an isolated hypersurface singularity.

The last problem was solved (and implemented in Singular) by S. Endraß for nondegenerate singularities. M. Schulze has made progress in attacking 2. and 3.

3. Moduli spaces and invariants

When classifying objects in algebraic geometry, one usually fixes discrete invariants, such as the genus of a projective curve, and then one would like to have a distinct view on the set of objects with fixed invariants with respect to some equivalence relation. For small invariants it is sometimes possible to enumerate the equivalence classes and to provide normal forms. For bigger invariants this usually fails and a way to describe the objects is to construct a classifying space such that each point of this space corresponds to a unique equivalence class. In algebraic geometry this classifying space should again be an algebraic variety, together with certain functorial properties. These ideas lead to the notion of a fine, respectively coarse, **moduli space** ([MuF], [Ne]).

Classically, moduli spaces have been constructed for global algebraic objects such as projective varieties, or for vector bundles on a fixed projective variety. During the past years there has also been some progress in constructing moduli spaces for singularities (cf. [GHP]) and for Cohen–Macaulay modules on a fixed local ring of a curve singularity ([GP3], see also [Gr1] for a survey). Indeed, the methods of proof are constructible and can be transferred to algorithms and finally to programmes.

In the following, we describe an algorithm to compute a moduli space for isolated hypersurface singularities, following [GHP]. The algorithm has been developed and implemented in Singular by T. Bayer ([Ba]).

Let $\mathbf{w} = (w_1, \dots, w_n) \in \mathbb{Z}^n$, $w_i > 0$, be a weight vector and $f \in \mathbb{C}\{x_1, \dots, x_n\}$ a semiquasihomogeneous power series, i.e.,

$$f = f_0 + \sum_{\langle \mathbf{w}, \alpha \rangle > d} c_\alpha x^\alpha, \quad f_0 = \sum_{\langle \mathbf{w}, \alpha \rangle = d} c_\alpha x^\alpha$$

such that the quasi-homogeneous (or weighted homogeneous) principal part f_0 has an isolated singularity at the origin. We denote the class of all such power series (resp. singularities) by $\mathcal{C}_{f_0}$.

Two power series f, g are called **right equivalent**, $f \overset{r}{\sim} g$, if there exists a holomorphic coordinate change $\phi^\# : (\mathbb{C}^n, 0) \to (\mathbb{C}^n, 0)$ such that $f = g \circ \phi^\#$, or, equivalently, $f = \phi(g)$, where $\phi \in \mathrm{Aut}(\mathbb{C}\{x_1, \dots, x_n\})$ is the algebra automorphism corresponding to $\phi^\#$.

In a series of papers, V.I. Arnold classified all isolated hypersurface singularities w.r.t. right equivalence up to modality 2, by giving normal forms [AGV].

Here we should like to present an algorithm to compute a moduli space for semiquasihomogeneous power series with fixed principal part w.r.t. right equivalence.

In [GHP], also a moduli space for contact equivalence was constructed, but that construction is more involved and not treated here.

To start with we need an algebraic variety which parametrises all semiquasihomogeneous power series (up to right equivalence) and then to identify equivalent objects. Indeed, equivalent objects belong to the same orbit of an algebraic group action and the aim is to compute explicitly the group, the action of the group and, finally, the quotient space.

Giving f_0, we compute the set of exponents $B \subset \mathbb{N}^n$ so that $\{x^\alpha \mid \alpha \in B\}$ is a monomial basis of the Milnor algebra $M_{f_0} = \mathbb{C}\{x_1, \ldots, x_n\}/\langle f_{0,x_1}, \ldots, f_{0,x_n}\rangle$. This requires a standard basis computation for a local ordering (cf. [GP1]). Then we select $B_- = \{\alpha \in B | \langle \mathbf{w}, \alpha\rangle > d\}$ and set $T_- = \mathbb{C}^k$ with $k = |B_-|$.

The polynomial

$$F_t(x) = f_0(x) + \sum_{\alpha \in B_-} t_\alpha x^\alpha$$

is the miniversal μ–constant unfolding of f_0. By a theorem of Arnold ([AGV]), for any $f \in \mathcal{C}_{f_0}$ there exists a $t \in T_-$ such that $f \overset{r}{\sim} F_t$. The next step in the algorithm is to compute, for a given $f \in \mathcal{C}_{f_0}$, a coordinate change ϕ and a $t \in T_-$ such that $\phi(f) = F_t$. The computation follows Arnold's proof, constructing ϕ degree by degree until the maximal weighted degree $\langle \mathbf{w}, \alpha\rangle$, $\alpha \in B_-$.

Usually there exist $t \neq t' \in T_-$ such that $F_t \overset{r}{\sim} F_{t'}$. However, we have the following fact (proved in [GHP] by using the Gauß–Manin connection): let $f, g \in \mathcal{C}_{f_0}$, $\varphi \in \operatorname{Aut} \mathbb{C}\{x\}$ and assume $\varphi(f) = g$. Then $\operatorname{ord}_{\mathbf{w}}(\varphi) \geq 0$, that is $\operatorname{ord}_{\mathbf{w}}\big(\varphi(x_i) - x_i\big) \geq w_i$ for $i = 1, \ldots, n$.

In the theorem of Arnold, $\operatorname{ord}_{\mathbf{w}}(\phi) > 0$, which implies that $t \in T_-$ is unique. Moreover, $\operatorname{Aut}_{>0} \mathbb{C}\{x\} = \{\varphi \in \operatorname{Aut} \mathbb{C}\{x\} \mid \operatorname{ord}_{\mathbf{w}}(\varphi) > 0\}$ is a normal subgroup of $\operatorname{Aut}_{\geq 0}\mathbb{C}\{x\} = \{\varphi \in \operatorname{Aut} \mathbb{C}\{x\} \mid \operatorname{ord}_{\mathbf{w}}(\varphi) \geq 0\}$, and the quotient

$$G^{\mathbf{w}} = \operatorname{Aut}_{\geq 0} \mathbb{C}\{x\} / \operatorname{Aut}_{>0} \mathbb{C}\{x\}$$

acts algebraically on T_-. Let $G^{\mathbf{w}}_{f_0} \subset G^{\mathbf{w}}$ denote the subgroup which fixes f_0 and denote by $E_{f_0} \subset \operatorname{Aut}(T_-)$ the image of $G^{\mathbf{w}}_{f_0}$. Then E_{f_0} is a finite group acting algebraically on T_- and the geometric quotient T_-/E_{f_0} is the desired coarse moduli space for unfoldings in $\mathcal{C}_{f_0}$ modulo right equivalence (cf. [GHP]).

The following steps are needed for computing the moduli space:

0. Compute miniversal μ–constant unfolding,
1. compute $G^{\mathbf{w}}_{f_0}$,
2. compute the action of $G^{\mathbf{w}}_{f_0}$ on T_- using Arnold's theorem,
3. compute E_{f_0} and linearise to get E'_{f_0} acting linearly on some $\mathbb{C}^\ell$, $\ell \geq k$, and compute an equivariant embedding $i : T_- \hookrightarrow \mathbb{C}^\ell$,
4. determine generating invariant polynomials for E'_{f_0},
5. determine the relations between the invariants to obtain the equations for $i(T_-)/E'_{f_0} \cong T_-/E_{f_0}$, which is the desired moduli space.

Singular example for computing the moduli space (we omit intermediate commands):

```
> LIB "qhmoduli.lib";
> ring R = 0, (x,y,z), ls;       // define a local ring
> poly f = x2y + x2z + y5 - z5; // principal part
```

Step 0: compute a basis for the semi-universal unfolding

```
> ideal B = UpperMonomials(f); B;
B[1]=y3z3, B[2]=x2y3, B[3]=x2y2
```

Hence, $F = f + t_1y^3z^3 + t_2x^2y^3 + t_3x^2y^2$ is the miniversal μ–constant unfolding. The dimension of the moduli space is 3.

Step 1, 2 and 3: compute the equations of the stabilizer of f, compute the induced action on $T_- = \mathbb{C}^3$, linearise the action with equivariant embedding $T_- \hookrightarrow \mathbb{C}^4$

```
> list stab = StabEqn(f);  ....  // commands omitted
> actionid;            //linearised action of E'_f on C^4 ⊃ T_- = C^3
actionid[1]=s(1)*t(1), actionid[2]=-s(3)*t(2)+s(3)*t(4)+s(5)*t(2)
actionid[3]=s(4)*t(3), actionid[4]=s(5)*t(4)
```

Step 4: compute generators for the invariant ring of E'_f

```
> def T = InvariantRing(groupid, actionid); setring T;
> invars;            //there are 21 invariants of degree 3 to 10
invars[1]=t(1)*t(4)^2, invars[2]=t(2)*t(3)*t(4)-t(3)*t(4)^2, ...
```

Step 5: compute equations of the moduli space

```
> def R4 = ImageVariety(V, invars);  //V is the ideal of T_- ⊂ C^4
> setring R4; imageid;       //simplified equation of moduli space
imageid[1]=Y(5)^2-Y(4)*Y(6), imageid[2]=Y(3)*Y(5)-Y(2)*Y(6), ...
imageid[55]=9*Y(1)^5+2816*Y(2)^2*Y(6)*Y(9)+296*Y(6)*Y(7)^2
  -152*Y(2)*Y(6)*Y(8)-960*Y(6)*Y(7)*Y(9)-9*Y(6)*Y(10)
```

Hence, the moduli space for $x^2y + x^2z + y^5 - z^5$ is a 3–dimensional affine subvariety of $\mathbb{C}^{10}$ defined by 55 equations of degrees between 2 and 5.

This shows already that moduli spaces have a complicated structure, even for relatively small examples.

Problems:

1. Extend the algorithms to construct moduli spaces for singularities with respect to contact equivalence. This will contain completely new parts since we need not only handle finite groups but unipotent groups.
2. Moduli spaces for torsion free modules on curve singularities have been constructed in [GP3] with constructive proofs. Again unipotent group actions come into play. It would be desirable to develop and implement algorithms and test conjectures related to the structure of these moduli spaces.

4. Curves with prescribed singularities

It is a classical and interesting problem, which is still in the centre of theoretical research, to study the variety $V = V_d(S_1, \ldots, S_r)$ of (irreducible) curves $C \subset \mathbb{P}^2_{\mathbb{C}}$ of degree d having exactly r singularities of prescribed (topological or analytical) types $S_1, \ldots, S_r$. Among the most important questions are:

- Is $V \neq \emptyset$ (existence problem)?
- Is V irreducible (irreducibility problem)?
- Is V smooth of expected dimension (T–smoothness problem)?

A complete answer is only known in the special case of nodal curves, that is, for $V_d(r) = V_d(S_1, \ldots, S_r)$ with S_i ordinary nodes (A_1–singularities): $V_d(r) \neq \emptyset$ and T–smooth $\Longleftrightarrow r \leq \frac{(d-1)(d-2)}{2}$ (Severi, 1921), $V_d(r)$ is irreducible (if $\neq \emptyset$) (Harris, 1985). Even for cuspidal curves there is no sufficient and necessary answer to any of the above questions and one can hardly expect such an answer.

Clearly, one can easily give an upper bound for the number of singular points that may occur on a plane irreducible curve C of degree d: by the genus formula C can have, at most, $(d-1)(d-2)/2$ singularities. Another upper bound for the (weighted) number of singularities arises from applying Bézout's Theorem to the intersection of two generic polars of C:

$$\sum_{z\in C} \mu(C,z) \leq (d-1)^2,$$

$\mu(C,z) = \dim_{\mathbb{C}} \mathbb{C}\{x,y\}/(f_x, f_y)$ the Milnor number of C at z.

On the other hand, in the case of arbitrary topological types S_i, we have the following existence theorem, which is asymptotically optimal (with respect to the occurring invariants and the exponent of d)

Theorem: [GLS, Lo]. *$V_d(S_1,\dots,S_r) \neq \emptyset$ if $\sum_{i=1}^r \mu(S_i) \leq \frac{1}{46}(d+2)^2$ and two additional conditions for the five "worst" singularities hold true.*

In case of only one singularity we have the slightly better sufficient condition for existence, $\mu(S_1) \leq \frac{1}{29}(d-5)^2$.

The theorem is just an existence statement, the proof gives no hint how to produce any equation. To produce explicit equations one needs some constructive method. Then the computer can be used in order to check the construction, or even, to improve the results. The following is a prominent example (actually, it belongs to a series of "world record" examples):

Example: [GN] *The irreducible curve with affine equation*

$$y^2 - 2y\left(x^{28} + 2x^{21}y^{16} - 2x^{14}y^{32} + 4x^7y^{48} - 10y^{64}\right) + x^{56} + 4x^{49}y^{16} = 0$$

has degree 65 and an A_{2260}–singularity ($x^2 - y^{2261} = 0$) and a semiquasihomogeneous singularity $S_{9,16}$ with principal part $f_0 = x^9 + y^{16}$ as only singularities. In particular, it is an element of the variety $V_{65}(A_{2260}, S_{9,16})$ which has negative expected dimension (hence is not T–smooth).

In order to verify this, one may proceed, using Singular, as follows:

```
> ring s = 0,(x,y),ds;
> poly f = y2-2x28y-4x21y17+4x14y33-8x7y49+20y65+x56+4x49y16;
> matrix Hess = jacob(jacob(f));     //the Hessian matrix of f
> vdim(std(jacob(f)));               //the Milnor number of f
2260
```

Since the rank of the Hessian at 0 is checked to be 1, f has an A_k singularity at 0; it is an A_{2260}–singularity since the Milnor number is 2260. In the following we show that the projective curve defined by f has no further singularities in the affine part. This follows from

$$\dim_{\mathbb{C}}(\mathbb{C}[x,y]_{\langle x,y\rangle}/\langle \text{jacob}(f), f\rangle = \dim_{\mathbb{C}}(\mathbb{C}[x,y]/\langle \text{jacob}(f), f\rangle,$$

confirmed by Singular:

```
> vdim(std(jacob(f)+f));
2260           // multiplicity of Sing(C) at 0 (local ordering)
> ring r = 0,(x,y),dp;
> poly f = fetch(s,f);
> vdim(std(jacob(f)+f));
```

```
2260            // multiplicity of Sing(C)      (global ordering)
```

Finally, we have to consider the singularities at infinity:

```
> ring sh = 0,(x,y,z),dp;
> poly f = fetch(s,f);
> poly F = homog(f,z); F;   // homogeneous polynomial defining C
4x49y16+20y65+x56z9-8x7y49z9+4x14y33z18-4x21y17z27-2x28yz36+y2z63
> ring r1 = 0,(y,z),dp;
> map phi = sh,1,y,z;
> poly g = phi(F);       // F in affine chart (x=1)
> vdim(std(jacob(g)+g));
120
> ring r2 = 0,(y,z),ds;  // local ring at (1:0:0)
> poly g = fetch(r1,g); g;
z9+4y16-2yz36-4y17z27+4y33z18-8y49z9+20y65+y2z63
> vdim(std(jacob(g)+g));
120
```

As before, we can conclude that there is precisely one singularity of C on the line at infinity, situated at $(1:0:0)$, being semiquasihomogeneous of type $S_{9,16}$. (Note that in our computation we have considered all points at infinity except $(0:1:0)$. The latter is obviously not a point of C).

In the following we should like to mention a few **problems and conjectures** which are currently in the centre of research in connection with singular curves in $\mathbb{P}^2_{\mathbb{C}}$.

Computing zero–dimensional ideals

Many of the questions concerning plane projective curves with prescribed singularities can be translated to properties of zero–dimensional (homogeneous) ideals $I \subset \mathbb{C}[x,y,z]$, e.g.,

- existence of curves with (ordinary) multiple points *in prescribed position*, or, more generally, existence of curves with prescribed position of infinitely near points (clusters),
- T–smoothness of the varieties $V_d(S_1,\dots,S_r)$,
- existence of (global) deformations of projective curves.

For instance, consider the following problem: given points $p_1,\dots,p_n \in \mathbb{P}^2_{\mathbb{C}}$ and positive integers $m_1,\dots,m_n$. Determine the dimension of the variety of curves of any degree d passing through each of the points p_i with multiplicity (at least) m_i, $i = 1,\dots,n$. The equivalent formulation would be: determine the ideal

$$I = \mathfrak{m}_{p_1}^{m_1} \cap \ldots \cap \mathfrak{m}_{p_n}^{m_n} \subset \mathbb{C}[x,y,z]\,,$$

$\mathfrak{m}_{p_i}$ the maximal ideal at p_i, and compute the Hilbert function H_I of I.

Harbourne–Hirschowitz conjecture: *Let $n > 9$, $p_1,\dots,p_n \in \mathbb{P}^2_{\mathbb{C}}$ in general position, m a positive integer, and let $I = \mathfrak{m}_{p_1}^{m} \cap \ldots \cap \mathfrak{m}_{p_n}^{m}$. Then the Hilbert function satisfies*

$$H_I(d) \;=\; \max\left\{0,\; \frac{(d+1)(d+2)}{2} - n\cdot\frac{m(m+1)}{2}\right\}$$

for all $d > 0$. In other words, the variety of curves with n singular points of multiplicity (at least) m at the prescribed (generic) points has the expected dimension.

There are several special cases where this conjecture is known to hold true; in particular, C. Ciliberto and R. Miranda [CM] have proven that it always holds for $m \leq 12$. Nevertheless the general conjecture is still far from being proven.

Nagata conjecture: *Let $n > 9$, $p_1, \dots, p_n \in \mathbb{P}^2_{\mathbb{C}}$ in general position, $m_1, \dots, m_n$ positive integers, and let $a(m_1, \dots, m_n)$ denote the minimal degree of a curve passing through each of the points p_i with multiplicity (at least) m_i, $i = 1, \dots, n$. Then*

$$a(m_1, \dots, m_n) > \frac{m_1 + \ldots + m_n}{\sqrt{n}}.$$

N. Nagata [Na] has proven the statement to be true for any $n > 9$ being a square. There are many people working to prove this conjecture for other integers n ([Ra]), or, at least, to improve the known lower bounds for $a(m_1, \dots, m_n)$ (the best known general bound is probably given in [Ro]). But the general question is still widely open.

Computer algebra could be used to provide evidence for such conjectures (or, to produce counter examples) provided one can solve the following **problems:**

1. find algorithms to compute the 0–dimensional ideals (related to the above problems). In many cases this is easy but for others this is unknown (e.g., to compute the equisingularity ideal for a sufficiently general singularity);
2. find *fast* algorithms to compute the intersection of zero–dimensional ideals. The general method for computing intersections via syzygies or elimination is too slow, due to the high complexity of the algorithms involved. There is already some considerable progress made, by the so–called Buchberger–Möller algorithm and further generalisations (cf. [AKR]), but certainly this is not yet sufficient.

5. Symbolic–numerical polynomial solving

Algebraic geometry is concerned with investigating the structure of the set of solutions of finitely many polynomial equations. Solving such a system is considered to be part of numerical analysis rather than of algebraic geometry. However, knowing something about the structure of the solution set can actually help in finding the solutions.

Given polynomials $f_1, \dots, f_k \in K[x_1, \dots, x_n]$, K being $\mathbb{R}$ or $\mathbb{C}$, numerical solving means determining the coordinates $(p_1, \dots, p_n)$, up to a given precision of all (respectively some, respectively one) points of the variety

$$V = \{p = (p_1, \dots, p_n) \in K^n \mid f_1(p) = \ldots = f_k(p) = 0\}.$$

Algebraically, we are interested in describing the structure of V, in computing its dimension, in the number of solutions (if finite), in the decomposition into irreducible varieties (e.g., by primary decomposition), in the radical of the ideal I generated by $f_1, \dots, f_k$ or the normalisation of the ring $K[x_1, \dots, x_n]/I$. Since V, the set of solutions of $f_1 = \ldots = f_k = 0$ depends only on I, even only on the radical of I, all the above mentioned methods can be used in preparing the given polynomial system for better numerical solving.

Hence, algebraic geometry and computer algebra may be used as symbolic preprocessing for easy numerical postprocessing. In particular, the following algorithms and methods may be applied (all being implemented in Singular).

- Find other generators of I, or of ideals with the same solution set, for example, triangular sets, based on Gröbner basis computations (see below), which allow better numerical solving: the numerical algorithms become more stable, we can solve overdetermined systems and find all solutions.
- Create other ideals, having the same solution set, for example, the *radical* for obtaining only simple zeros, *primary decomposition* for splitting the system into several smaller ones.
- Compute a parametrisation of the solution set V, which is only possible for rational V, sometimes it is achieved by the *normalisation* of $K[x_1, \ldots, x_n]/I$.
- Reduce higher dimensional solving to 0–dimensional solving by applying a *Noether normalisation.*

Pure numerical solving has the advantage of being fast, flexible in accuracy by using iterative methods and being applicable not only to polynomial systems. Indeed, the big success of numerical methods during the past years seems to show that symbolic methods are of little use in solving systems coming from real life problems. However, due to rounding errors, numerical methods are principally uncertain, often unstable in an unpredictable way, sometimes do not find all solutions and have problems with overdetermined systems. Moreover, they can hardly treat underdetermined systems (sometimes curves, at most surfaces, as solution sets) and certainly get into trouble near singularities.

On the other hand, symbolic methods are principally exact and stable. However they have a high complexity, are, therefore, slow, and, in practice, are applicable only to small systems (this is the case, in particular, for radical computation, primary decomposition and normalisation). Nevertheless, they are applicable to any polynomial system of any dimension and for zero–dimensional systems they can predict precisely the number of complex solutions (counted with multiplicities) . Moreover, as is well–known, symbolic preprocessing of a system of polynomials (even of ordinary and partial differential equations) may not only lead to better conditions for the system to be solved numerically but can help to find all solutions or even make numerical solving possible (see below).

There is continuous progress in applying symbolic methods to numerical solving, cf. the various articles in the ISSAC Proceedings, the survey article by Möller [Mo], the textbook by Cox, Little and O'Shea [CLO2] or the recent paper by Verschelde [Ve]. Besides Gröbner basis many other methods have been used. Recently, resultant methods have been re–popularised, in particular in connection with numerical solving (cf. [CE], [WEM]), partly due to the new sparse resultants by Gelfand, Kapranov and Zelevinsky [GKZ]. I should also mention the work of Stetter (cf. [Ste1, Ste2]), which is not discussed in this paper.

In the following I shall describe roughly how Gröbner bases and resultants can be applied to prepare for numerical solving of zero–dimensional systems. Moreover, I shall present experimental material for comparing the performance of the two methods, which seems to be the first practical comparison of resultants and Gröbner bases in connection with numerical solving under equal conditions. The motivation for doing this came from a collaboration with electrical engineers, aiming at symbolic analysing and sizing micro-electric analog circuits.

Let $f_1, \ldots, f_k \in K[x_1, \ldots, x_n]$, and assume that the system $f_1 = \ldots = f_k = 0$ has only finitely many complex solutions. The problem is to find all solutions up to a given precision. We present two methods, one by computing a lexicographical

Gröbner basis and then splitting this into triangular sets, the second by computing the sparse u–resultants and the determinants of the partly evaluated resultant matrix. Both methods end up with the problem of solving univariate polynomial equations for which we use Laguerre's method.

Solving polynomial systems using Gröbner bases and triangular sets:

Input: Zero–dimensional system $f_1, \ldots, f_k \in K[x_1, \ldots, x_n]$, $k \geq n$.
Output: Complex roots of $f_1 = \ldots = f_k = 0$ in $\mathbb{C}^n$.

- Compute a reduced lexicographical Gröbner basis $G = \{g_1, \ldots, g_s\}$ of the ideal $I = \langle f_1, \ldots, f_k \rangle$ with $s \geq n$.
- Compute a triangular system: a *triangular basis* is a reduced lexicographical Gröbner basis $G = \{g_1, \ldots, g_n\}$ (as many polynomials as variables) with g_i of the form $g_i = x_i^{p_i} + g_i'(x_i, \ldots, x_n)$ with $\deg_{x_i} g_i' < p_i$. A *triangular system* for I consists of triangular bases $T_1, \ldots, T_s$ such that $V(I) = V(T_1) \cup \ldots \cup V(T_s)$. Triangular systems can be computed effectively, basically by two different methods, one due to Lazard [La], [D5], the other due to Möller [Mo]. Choose any of these methods to compute a triangular system $T_1, \ldots, T_s$ for I.
- Use a numerical solver (e.g. Laguerre's method) to find all zeros of T_i, $i = 1, \ldots, s$. The union of these zero–sets is the desired solution set.

There are several variations on how to compute triangular sets. The $V(T_i)$ need not be disjoint (but can be made disjoint). The T_i need not define maximal ideals (but this can be achieved), we may use the factorising Gröbner, etc. Some of these have been implemented in Singular by D. Hillebrand, a former student of M. Möller (cf. [Hi]).

Singular example (the output has been changed to save space):

```
> ring s = 0,(x,y,z),lp;
> ideal i = x2+y+z-1,x+y2+z-1,x+y+z2-1;
> option(redSB);  //option for computing a reduced Groebner basis
> ideal j = groebner(i); j;
j[1]=z6-4z4+4z3-z2, j[2]=2yz2+z4-z2, j[3]=y2-y-z2+z, j[4]=x+y+z2-1
> LIB "triang.lib";
> triangMH(j);    //triangular system with Moeller's method
>                 //(fast, but not necessarily disjoint)
[1]:                    [2]:
   _[1]=z                  _[1]=z4-4z2+4z-1
   _[2]=y2-y+z             _[2]=2y+z2-1
   _[3]=x+y-1              _[3]=2x+z2-1
> triangMH(j,2);  //triangular system (with Moeller's method,
>                 //improved by Hillebrand) and factorisation
[1]:             [2]:           [3]:               [4]:
   _[1]=z           _[1]=z         _[1]=z2+2z-1       _[1]=z-1
   _[2]=y           _[2]=y-1       _[2]=y-z           _[2]=y
   _[3]=x-1         _[3]=x         _[3]=x-z           _[3]=x
```

We can now solve the system easily by recursively finding roots of univariate polynomials, Singular commands are:

```
> LIB "solve.lib";
> triang_solve(triangMH(j,2),30);   //accuracy of 30 digits
```

or applying `triangLf_solve(i);` directly to i.

Resultant methods have recently become popular again, due to new sparse resultants invented by Gelfand, Kapranov and Zelevinsky [GKZ]. Indeed, they beat by far the classical Macaulay resultants as is shown, for example, in [We]. The following algorithm to use sparse resultants for polynomial solving is due to Canny and Emiris [CE], it has been implemented in SINGULAR by M. Wenk [We].

For computing resultants, we have to start with a zero–dimensional polynomial system with as many equations as variables, and have to compute the u–resultant (named so by Van der Waerden) where u_i are new variables which have to be specialised later. The construction of the sparse resultant matrix uses a mixed polyhedral subdivision of the Minkowski sum of the Newton polytopes. Specialisations of the u–coordinates are used then to reduce the problem to the univariate case. The determinants of the specialised u–resultant matrices are univariate polynomials, the roots are determined by Laguerre's algorithm.

The main advantage of the sparse resultants against Macaulay resultants is that the size of the resultant matrices depend on the Newton polytopes and not just on the degrees of the input polynomials, hence is much smaller. Note that by the resultant method we can only determine roots in $(\mathbb{C} \setminus \{0\})^n$.

Here is a more detailed description of the algorithm (for details see [CE] and [We]):

Solving polynomial systems using resultants (after Gelfand, Kapranov, Zelevinsky (1994) and Canny, Emiris (1997)):

Input: Zero–dimensional system $f_1, \ldots, f_n \in \mathbb{C}[x_1, \ldots, x_n]$.
Output: Complex roots of $f_1 = \ldots = f_n = 0$ in $(\mathbb{C} \setminus \{0\})^n$.

- Add $f_0 = u_0 + u_1 x_1 + \ldots + u_n x_n \in \mathbb{K}[u_0, \ldots, u_n, x_1, \ldots, x_n]$ and compute the Newton polytopes $Q_i \subset \mathbb{R}^n$ of f_i, $i = 0, \ldots, n$.
- Compute, using linear programming, a polyhedral subdivision Δ of the Minkowski sum $Q = Q_0 + \ldots + Q_n$. Translate Δ by a small vector in $\mathbb{R}^n$ such that lattice points are interior points.
- Construct from Δ the square resultant matrix $M(u_0, \ldots, u_n)$, which has as entries either a number or a variable u_i.
- Set $M^i(u_0) := M(u_0, 0, \ldots, 0, -1, 0, \ldots, 0)$, -1 at the i–th place. The set L_i of all roots of $\det\big(M^i(u_0)\big)$, $i = 1, \ldots, n$, contains the i–th components of all complex solutions of the system $f_1 = \ldots = f_n = 0$ (in an unordered manner).
- For each solution of $f_1 = \ldots = f_n = 0$ identify the components which were computed in the previous step. This is done by substituting $u_1, \ldots, u_i$ by random numbers and $u_{i+1}, \ldots, u_n$ by 0 in $M(u_0, u_1, \ldots, u_n)$, computing the determinant and solve this for u_0.

Most of the time is spent in the second last and, in particular, in the last step.

As an example, we show the computation of the complex zero–set of the ideals $I_1, \ldots, I_5$ (with precision of 30 digits, see below) which represent more than 60 examples. On average our resultant solver could manage the same examples as Mathematica and MAPLE (but MAPLE found fewer roots). The problems for the resultant solver occurred either because of too big matrices or because of numerical problems for the subsequent Laguerre solver (no convergence). Sometimes not all solutions were found, sometimes the system returned too many solutions because multiple solutions were interpreted as different ones.

Our experiments do not confirm the claim made in [WEM] that resultant methods are best suited to polynomial systems solving, at least for bigger examples. On the contrary, Gröbner bases and triangular sets showed the best performance, identified most precisely the correct number of simple roots, could treat many more examples and had the least numerical difficulties. The most expensive part was usually the computation of a lexicographical Gröbner basis (computed through FGLM); the triangular set computation was less expensive and the numerical solving depended very much on the example.

Of course, the Gröbner bases in SINGULAR are highly tuned and the resultant computations can certainly be improved. The examples show that resultant methods are a good alternative for small examples. But for many variables even the sparse resultants become huge and we have to compute several determinants of these matrices. This is the main bottleneck for the resultant method. Nevertheless, still more research has to be done.

Interpretation of the table: vars: number of variables, mult: number of complex solutions with multiplicity, # roots: number of different roots without multiplicity, time: total time, degree of res.: number of (not necessarily simple roots) in $(\mathbb{C} \setminus \{0\})^n$, matrix size: number of rows of (square) resultant matrix.
Commands used:

- `triang_solve(I);` and `ures_solve(I);` (SINGULAR),
- `NSolve(eqns,vars,30);` (Mathematica),
- `evalf(solve(eqns,vars),30);` (MAPLE),
- `GROESOLVE(eqns,vars);`, respectively `solve(I,var);` with options `on complex; on rounded; precision 30;` (REDUCE).

No	vars	mult	#roots	SINGULAR 1-3-7				
				Triang. systems		Resultant method		
				roots found	time	degree of res.	time	matrix size
1	3	53	32	32	24 sec	31	$\sim$ 40 sec	$\sim$ 160
2	4	16	16	16	2 sec	16	$\sim$ 5 sec	$\sim$ 70
3	5	70	70	70	5 sec	70	$\sim$ 50 sec	$\sim$ 880
4	6	156	156	156	22 sec	156	$>$ 5000 sec	$\sim$ 5460
5	10	27	27	27	66 sec	30	$>$ 5000 sec	$\sim$ 10000

No	Mathematica 4.0		MAPLE V.5		REDUCE 3.5	
	roots found	time	roots found	time	roots found	time
1	37	100 sec	—	$>$ 5000 sec	—	$>$ 5000 sec
2	16	9 sec	1	1 sec	—	$>$ 5000 sec

$I_1 = \langle x^3z+6x^2-2xz^3+y^4+yz,\ 5x^2y^2+y^3z+3yz^3+z^3,\ -x^2z-2xyz^2+4y^4+2y^2z\rangle \subset \mathbb{C}[x,y,z]$,

$I_2 =$ (cf. [WEM]) $\langle 5x^2+6x+3y+6z+2w+3, 4x+4y^2+3z+2w+4, x-11z^2+3z+7w-9, x+3y+2z-3w^2+13\rangle \subset \mathbb{C}[x,y,z,w]$,

$I_3 =$ (cyclic 5) $\langle a+b+c+d+e,\ ab+ae+bc+cd+de,\ abc+abe+ade+bcd+cde,\ abcd+abce+abde+acde+bcde,\ abcde-1\rangle \subset \mathbb{C}[a,b,c,d,e]$,

$I_4 =$ (Arnborg 6) $\langle a+b+c+d+e+f,\ ab+af+bc+cd+de+ef,\ abc+abf+aef+bcd+cde+def,\ abcd+abcf+abef+adef+bcde+cdef,\ abcde+abcdf+abcef+abdef+acdef+bcdef,\ abcdef-1\rangle \subset \mathbb{C}[a,b,c,d,e,f]$,

$I_5 =$ (POSSO, Methan6_1) $\langle -10ai - 320000a + 64bh + 10hj + 11hk,$ $160000a - 32bh - 5bi - 5bk - 5000b,\ -ci + gi + 210g + jk + 1300000,\ -ei + 700000,\ -2f + k^2,\ -gi - 210g + hj,\ 320000a - 64bh - 10hj - 11hk - 16h + 7000000,\ ei - hj - jk - 410j,\ -10ai - 10bi + 10bk + 20000b - 10ci - 10ei + 14f - 10gi + 11hk,\ 10bi - 10bk + 10ci + 10gi + 1400g + 10hj - 11hk - 10jk - 10k^2 - 4200k\rangle \subset \mathbb{C}[a,b,c,d,e,f,g,h,i,j,k]$.

The computations, with precision of 30 digits, were performed on a Pentium Pro 200 with 128 MB. MAPLE and REDUCE could not solve examples 3 – 5 within our time limit of 5000 seconds, Mathematica stopped with an error.

MAPLE offers also the opportunity to preprocess a set of ideal generators in view of solving, by using the command `gsolve;`. But within our time limit of 5000 seconds, this also lead only to a result for example 2. However, in this case, MAPLE is able to compute all roots, by successively applying `fsolve(eqn,var,complex);` and substituting.

REFERENCES

[AGV] V.I. Arnold, S.M. Gusein–Zade and A.N. Varchenko, *Singularities of Differential Maps*, Volume I. Birkhäuser 1985.

[AKR] J. Abbott, M. Kreuzer and L. Robbiano, *Computing zero–dimensional schemes*, Preprint in preparation (2000).

[Ba] T. Bayer, *Computation of moduli spaces for semiquasihomogeneous singularities and an implementation in* SINGULAR, Diplomarbeit, Kaiserslautern, 2000.

[Bo] A. Borel, *Linear Algebraic Groups*, 2nd Edition, Springer 1991.

[Br] E. Brieskorn, *Die Monodromie der isolierten Singularitäten von Hyperflächen*, Manuscripta Math., **2**, 103–161 (1970).

[Bu1] B. Buchberger, *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal*, PhD Thesis, University of Innsbruck, Austria, 1965.

[Bu2] B. Buchberger, *Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems*, Äqu. Math., **4**, 374–383 (1970).

[Bu3] B. Buchberger, *Gröbner bases: an algorithmic method in polynomial ideal theory*, in: Recent trends in multidimensional system theory, N.B. Bose, ed., Reidel (1985).

[BW] B. Buchberger and F. Winkler, *Gröbner Bases and Applications*, LNS **251**, 109–143, CUP (1998).

[BWe] T. Becker and V. Weispfenning, *Gröbner Bases, A Computational Approach to commutative Algebra*, Graduate Texts in Mathematics 141, Springer 1993.

[CE] J.F. Canny and I.Z. Emiris, *A Subdivision–Based Algorithm for the Sparse Resultant*, Preprint, Berkeley, 1997.

[CLO1] D. Cox, J. Little and D. O'Shea, *Ideals, Varieties and Algorithms*, Springer 1992.

[CLO2] D. Cox, J. Little and D. O'Shea, *Using Algebraic Geometry*, Springer 1998.

[CM] C. Ciliberto and R. Miranda, *Linear Systems of Plane Curves with Base Points of Equal Multiplicity*, Duke preprint, no. math.AG/9804018 (1998).

[De] H. Derksen, *Constructive Invariant Theory and the Linearization Problem*, PhD Thesis, University of Basel, 1997.

[DGJP] W. Decker, G.-M. Greuel, T. de Jong and G. Pfister, *The normalisation: a new algorithm, implementation and comparisons*, in: Proc. EUROCONFERENCE Computational Methods for Representations of Groups and Algebras (1.4.–5.4.1997), 177–185, Birkhäuser 1998.

[DGP] W. Decker, G.-M. Greuel and G. Pfister, *Primary decomposition: algorithms and comparisons*, in: G.-M. Greuel, B.H. Matzat and G. Hiß (Eds.), *Algorithmic Algebra and Number Theory*, 187–220, Springer 1998.

[D5] J. Della Dorca, C. Dicrescenzo and D. Duval, *About a new method for computing in algebraic number fields*, EUROCAL 1985, LN in CS 204, 289–290 (1985).

[Ei] D. Eisenbud, *Commutative Algebra with a view toward Algebraic Geometry*. Springer 1995.

[EHV] D. Eisenbud, C. Huneke and W. Vasconcelos, *Direct methods for primary decomposition*, Invent. Math., **110**, 207–235 (1992).

[GHP] G.-M. Greuel, C. Hertling and G. Pfister, *Moduli Spaces of Semiquasihomogeneous Singularities with fixed Principal Part.* J. of Alg. Geom. **6**, no. 1, 169–199 (1997).

[GKZ] I. Gelfand, M. Kapranov and A. Zelevinski, *Discriminants, Resultants and Multidimensional Determinants*, Birkhäuser 1994.

[GL] R. Gerard and A.H.M. Levelt, *Invariants mesurant l'irrégularité en un point singulier des systèmes d'équations différentielles linéaires*, Ann. Inst. Fourier Grenoble, **23**, 157–195 (1973).

[GLS] G.-M. Greuel, C. Lossen and E. Shustin, *Plane curves of minimal degree with prescribed singularities*, Invent. Math. **133**, 539–580 (1998).

[GMH] G.-M. Greuel, B.H. Matzat and G. Hiß (Eds.), *Algorithmic Algebra and Number Theory*, 187–220, Springer 1998.

[GN] S.M. Gusein–Zade and N.N. Nekhoroshev, *On A_k–singularity on a plane curve of fixed degree*, Duke preprint, no. math.AG/9906147 (1999).

[GP1] G.-M. Greuel and G. Pfister, *Advances and improvements in the theory of standard bases and syzygies*, Arch. Math. **66**, 163–176 (1996).

[GP2] G.-M. Greuel and G. Pfister, *Gröbner bases and algebraic geometry*, in: B. Buchberger and F. Winkler (Eds.): Gröbner Bases and Applications, LNS **251**, 109–143. CUP (1998).

[GP3] G.-M. Greuel and G. Pfister, *Moduli spaces for torsion free modules on curve singularities I*, J. Algebraic Geometry **2**, 81–135 (1993).

[GPS] G.-M. Greuel, G. Pfister and H. Schönemann. SINGULAR, *A System for Polynomial Computations*, version 1.2 User Manual, in: Reports On Computer Algebra, number 21. Centre for Computer Algebra, University of Kaiserslautern, June 1998. Available via `http://www.singular.uni-kl.de`.

[Gr1] G.-M. Greuel, *Deformation und Klassifikation von Singularitäten und Moduln*, Jahresber. Deutsche Math.–Verein Jubiläumstagung 1990, 177–238 (1992).

[Gr2] G.-M. Greuel, *Computer algebra and algebraic geometry — achievements and perspectives*, to appear in JSC, 2000.

[GTZ] P. Gianni, B. Trager and G. Zacharias, *Gröbner Bases and Primary Decomposition of Polynomial Ideals*, J. Symbolic Computation **6**, 149–167 (1988).

[Hi] D. Hillebrand, *Triangulierung nulldimensionaler Ideale — Implementierung und Vergleich zweier Algorithmen*, Diplomarbeit, Dortmund, 1999.

[Ku] V. Kulikov, *Mixed Hodge Structures and Singularities*, Cambridge Tracts in Mathematics **132**, Cambridge University Press 1998.

[La] D. Lazard, *Solving Zero–dimensional Algebraic Systems*, J. Symbolic Computation **13**, 117–131, (1992).

[Lo] C. Lossen, *The geometry of equisingular and equianalytic families of curves on a surface*, PhD Thesis, University of Kaiserslautern, 1998.

[Mil] J. Milnor, *Singular Points of Complex Hypersurfaces*, Ann. of Math. Studies **61**, Princeton (1968).

[Mo] H.M. Möller, *Gröbner Bases and Numerical Analysis*, in: B. Buchberger and F. Winkler (Eds.): *Gröbner Bases and Applications.* LNS 251, 159–178, CUP (1998).

[MuF] D. Mumford and J. Fogarty, *Geometric Invariant Theory*, Springer 1982.

[Na] N. Nagata, *On the fourteenth problem of Hilbert*, Amer. J. Math. **81**, 766–772, (1959).

[Ne] P.E. Newstead, *Introduction to moduli problems and orbit spaces*, Lecture Notes, Tata Institute of Fundamental Research, Springer 1978.

[Ra] Z. Ran, *On the Nagata Problem*, Duke preprint, no. math.AG/9809101 (1998).

[Ro] J. Roé, *On the existence of plane curves with prescribed multiple points*, Duke preprint, no. math.AG/9807066 (1998).

[Sch] M. Schulze, *Computation of the Monodromy of an Isolated Hypersurface Singularity*, Diploma Thesis, Kaiserslautern 1999.

[St] B. Sturmfels, *Algorithms in Invariant Theory*, Springer 1993.

[Ste1] H. Stetter, *Matrix eigenproblems are at the heart of polynomial systems solving*, Sigsam Bulletin **30**, 22–25 (1996).

[Ste2] H. Stetter, *Stabilization of polynomial systems solving with Groebner bases*, in Proceedings of ISSAC 97, 117–124 (1997).

[Ve] J. Verschelde, *Polynomial homotopies for dense, sparse and determinantal systems*, Duke preprint, no. math.NA/9907060 (1999).

[We] M. Wenk, *Resultantenmethoden zur Lösung algebraischer Gleichungssysteme implementiert in* SINGULAR, Diplomarbeit, Kaiserslautern, 1999.

[WEM] A. Wallack, I.Z. Emiris and D. Monacho, *MARS: A Maple/Matlab/C Resultant–Based Solver*, in Proceedings of ISSAC 98, 244–251 (1998).

DEPARTMENT OF MATHEMATICS,, UNIVERSITY OF KAISERSLAUTERN,, ERWIN-SCHRÖDINGER-STRASSE, D-67663 KAISERSLAUTERN

E-mail address: greuel@mathematik.uni-kl.de