SIGSAM Bulletin

Communications in Computer Algebra

SIGSAM BULLETIN

Volume 37, Number 3 September, 2003 Issue 145

A Quarterly Publication of the

Special Interest Group on Symbolic and Algebraic Manipulation

SIGSAM BULLETIN

Communications in Computer Algebra

A Quarterly Publication of the ACM Special Interest Group on Symbolic and Algebraic Manipulation.

Submission of Articles Any material that is of interest to SIGSAM members can be submitted for publication. Submissions to the SIGSAM BULLETIN should be written in LaTeX2ε and should be sent to the editor or one of the associate editors.

The BULLETIN publishes both reviewed an non-reviewed articles. A formally reviewed article is indicated by an imprint on its first page.

The ISSN number for the BULLETIN is given below.

ISSN: 0163-5824

International Symposium on Symbolic and Algebraic Computation Poster Abstracts 2003

Communicated by Olga Caprotti
RISC-Linz Research Institute for Symbolic Computation, Austria
olga.caprotti@risc.uni-linz.ac.at

ISSAC 2003 was held on July 3–6, 2003 at Drexel University in Philadelphia, USA. Below are the abstracts of posters that were accepted and presented at the conference. The *best poster award* committee, consisting of J. Davenport, D. Jeffrey, M. Monagan, T. Shaska, T. Sturm, N. Takayama, E. Volcheck and members of the poster committee, assigned the prize to B. Jüttler, J. Schicho, and M. Shalaby for their poster C^1 *Implicitization of Planar Curves*. All posters are archived and are available in electronic form from
http://www.acm.org/sigs/sigsam/issac/2003.

C^1 Spline Implicitization of Planar Curves

B. Jüttler
Institute of Analysis, Dept. of Applied Geometry
Johannes Kepler University, Linz, Austria
bert.juettler@jku.at

J. Schicho and M. Shalaby
Research Institute for Symbolic Computation
Johannes Kepler University, Linz, Austria
[jschicho,shalaby]@risc.uni-linz.ac.at

We present a new method for constructing a low degree C^1 implicit spline representation of a given parametric planar curve. To ensure the low degree condition, quadratic B-splines are used to approximate the given curve via orthogonal projection in Sobolev spaces. Adaptive knot removal, which is based on spline wavelets, is used to reduce the number of segments. The B-spline segments are implicitized. After multiplying the implicit B–spline segments by suitable polynomial factors the resulting bivariate functions are joined along suitable transversal lines. This yields to a globally C^1 bivariate function.

References

B. Jüttler, J. Schicho and M. Shalaby, *Spline Implicitization of Planar Curves*, Curves and Surfaces 2002, St. Malo, France.

M. Shalaby, B. Jüttler and J. Schicho, C^1 *Spline Implicitization of Planar Curves*, accpeted for ADG 2002, Hagenberg, Austria.

Interpolation in Symbolic Summation

Fabrizio Caruso
Dipartimento di Matematica "Tonelli", University of Pisa, Italy
caruso@dm.unipi.it

We present two software packages for computing the null space of a matrix with rational function coefficients. We use two approaches based on classical interpolation algorithms. The packages are written in the Mathematica language.

In the first package we use polynomial interpolation and Cramer's rule to reconstruct the components of the solutions. This package works under some assumptions on the matrix that, in most cases, are true for systems involved in Zeilberger's algorithm for hypergeometric definite summation.

In the second package we use a hybrid method based on heuristic rational function interpolation on some low degree components of the solutions and "symbolic" Gaussian elimination on the other components. The second package works under very general conditions and can be customized for systems related to symbolic summation.

On the relationship between the Dixon-based resultant construction and the supports of polynomial systems

Arthur D. Chtcherba
Department of Computer Science
University of Wyoming
Laramie, WY 82072
artas@cs.uwyo.edu

Deepak Kapur
Department of Computer Science
University of New Mexico
Albuquerque, NM 87131
kapur@cs.unm.edu

Different matrix based resultant formulations use the support of the polynomials in a polynomial system in various ways for setting up resultant matrices for computing resultants. Every formulation suffers, however, from the fact that for most polynomial systems, the output is not a resultant, but rather a nontrivial multiple of the resultant, called a *projection operator*. It is shown that for the Dixon-based resultant methods, the degree of the projection operator of unmixed polynomial systems is determined by the *support hull* of the support of the polynomial system. This is similar to the property that the Newton polytope of a support determines the degree of the resultant for toric zeros.

The support hull of a given support is similar to its convex hull (Newton polytope) except that instead of the Euclidean distance, the support hull is defined using relative quadrant (octant) position of points. The concept of a *support hull interior* point with respect to a support is defined. It is shown that for unmixed polynomial systems, generic inclusion of terms corresponding to support hull interior points does not change the size of the Dixon matrix (hence, the degree of the projection operator). The support hull of a support is the closure of the support with respect to support-interior points.

The above results are shown to hold both for the generalized Dixon formulation as well as for Sylvester-type Dixon dialytic matrices constructed using the Dixon formulation.

It is proved that for an unmixed polynomial system, the size of the Dixon matrix is less than or equal to the Minkowski sum of the alternating sums of the successive projections of the support of the polynomial system. This is a refinement of the result in Kapur and Saxena 1996 about the size of the Dixon matrix of a polynomial system, where it was shown that for the unmixed polynomial system, the size of the Dixon matrix is less than or equal to the Minkowski sum of the successive projection of the support.

Many other combinatorial properties of the size of the Dixon matrix and the structure of the Dixon polynomial of a given polynomial system are related to the support hull of the polynomial system and their projections along different dimensions.

This research is supported in part by NSF grant nos. CCR-0203051, CDA-9503064 and a grant from the Computer Science Research Institute at Sandia National Labs.

Approximate Computation of Pseudovarieties

Robert M. Corless, Hiroshi Kai and Stephen M. Watt
Ontario Research Centre for Computer Algebra
University of Western Ontario
London, Ontario, N6A 5B7 CANADA
[rmc,kai,watt]@orcca.on.ca

1 Introduction

This poster is concerned with a generalization of the notion of a variety of an ideal I, that we call a *pseudovariety* in analogy with the standard terminologies "pseudozero"[6, 7, 8, 9] and "pseuodspectrum"[9, 10]. Other works use "pseudozero set"[3] or "root neighborhoods"[4] to describe this concept.

"To state the main idea as directly as possible, the pseudozero set of f is the union of the zero sets of all systems $\hat{f}$ that are "acceptable approximations" of f in the sense that they come from f by small changes in the coefficients."[3]

The first thing to note about pseudovarieties (a technical definition will come later) is that they depend on the specified generators (basis) of the ideal I. They are not, therefore, true algebraic objects like the variety $V(I)$, which is independent of the basis for the ideal I. However, we prefer to retain the term "variety" as a subword of the definition because other features of varieties are preserved, and association of the words may help us to think about these things.

Let us first consider the simplest kind of pseudovariety. For multivariate polynomials $p_i \in \mathbb{C}[x_1, \cdots, x_s]$, define:

Definition 1 (Straightforward Pseudovariety).

$$PV\left(I_{\epsilon;p}\right) = \bigcup_{||\Delta p_i|| \leq \epsilon} V\left(\langle p_1 + \Delta p_1, \cdots, p_m + \Delta p_m \rangle\right)$$

where $\Delta p_i \in \mathbb{C}[x_1, \cdots, x_s]$ and the norm $|| \cdot ||$ means the 2-norm of the vector of polynomial coefficients.

One of the goals of this poster is to explore methods for visualization of the pseudovariety $PV(I_{\epsilon;p})$. We will use this notation to describe the stability of roots and a monodromy condition for systems.

2 Properties for pseudovarieties

Some basic properties, but helpful also for visualization, are shown for the affine varieties $V = V(f_1, \cdots, f_m)$ and $W = V(g_1, \cdots, g_n)$ in the text book [2] as follows.

Lemma 1. *If $V, W \subset \mathbb{C}^s$ are affine varieties, then so are $V \cap W$ and $V \cup W$.*

We would like to show these properties will be preserved for pseudovarieties $V = PV(I_{\epsilon;f_1,\cdots,f_m})$ and $W = PV(I_{\epsilon;g_1,\cdots,g_n})$ in some sense.

Lemma 2.

- *If $V, W \subset \mathbb{C}^s$ are pseudovarieties w.r.t. ϵ, then so is $V \cap W$.*
- *If unit polynomials $||f_i|| = 1, ||g_j|| = 1$ are assumed, then $V \cup W$ is a subset of a pseudovariety w.r.t. $\sqrt{2\pi}\epsilon$.*

Proof. We claim that

$$V \cap W = PV(I_{\epsilon; f_1, \cdots, f_m, g_1, \cdots, g_n})$$

and

$$V \cup W \subset PV(I_{\sqrt{2\pi}\epsilon; f_i g_j : 1 \le i \le m, 1 \le j \le n}).$$

The first equality is trivial to prove, since the coefficients of each polynomials are perturbating independently. The second one is proved as follows.

If $(a_1, \cdots, a_s) \in V$, then, at least, one of the neighborhoods $\tilde{f}_i, 1 \le i \le m$ must vanish at this point, which implies that all of the $\tilde{f}_i g_j$ for $1 \le j \le n$ also vanish at $(a_1, \cdots, a_s)$. By the way,

$$\begin{aligned} ||fg||_2^2 &= \frac{1}{2\pi} \int_0^{2\pi} f(z)\overline{f(z)} \cdot g(z)\overline{g(z)} dz \\ &\le \left[\frac{1}{2\pi} \int_0^{2\pi} f(z)\overline{f(z)} dz\right] \cdot \left[\int_0^{2\pi} g(z)\overline{g(z)} dz\right] \\ &= 2\pi \cdot ||f||^2 \cdot ||g||^2. \end{aligned}$$

Therefore,

$$||\tilde{f}_i g_j - f_i g_j|| \le \sqrt{2\pi} \cdot ||\tilde{f}_i - f_i|| \cdot ||g_j|| = \sqrt{2\pi}\epsilon.$$

Thus, $V \subset PV(I_{\sqrt{2\pi}\epsilon; f_i g_j : 1 \le i \le m, 1 \le j \le n})$, and $W \subset PV(I_{\sqrt{2\pi}\epsilon; f_i g_j : 1 \le i \le m, 1 \le j \le n})$ follows similarly. This proves that $V \cup W \subset PV(I_{\sqrt{2\pi}\epsilon; f_i g_j : 1 \le i \le m, 1 \le j \le n})$. □

The first property about $V \cap W$ may give a solution for visualization of pseudovarieties of overdetermined system, i.e. $m > s$.

3 Pseudozero

A useful theorem for the problem is known as a pseudozero criterion for systems of multivariate polynomials. This has been described by Stetter in [8]. We would like to give a brief explanation for the 2-norm case, and then we will show later how it works for visualization of pseudovarieties.

A polynomial in s variables with support $J \subset \mathbb{N}_0^s$ may be written as $p(x) = \sum_{j \in J} a_j x^j \in \mathbb{C}[x_1, \cdots, x_s]$. The tolerance associated with p is defined by a nonnegative vector $e \in \mathbb{R}^{|J|}$ whose components $e_j \ge 0$ correspond to the coefficients a_j, $j \in J$. Let

$$J' := \{j \in J : e_j > 0\} \ne \emptyset, \text{ and } |J'| \le |J|,$$

then the following defines the concept of e-neighborhood.

Definition 2 (e-neighborhood). *The e-neighborhood $N(p, e)$ of the polynomial p with tolerance e consists of those polynomials $\tilde{p} \in \mathbb{C}[x_1, \cdots, x_s]$, $\tilde{p}(x) = \sum \tilde{a}_j x^j$, which satisfy*

$$||(\cdots, \frac{|\tilde{a}_j - a_j|}{e_j}, \cdots)|| \le 1, \tag{1}$$

for $\tilde{a}_j = 0,\ j \notin J$ and $\tilde{a}_j = a_j, j \in J \backslash J'$.

Using the definition, the pseudovariety may be described as follows.

$$PV\left(I_{e;p}\right) = \bigcup_{p_i+\Delta p_i \in N(p_i,e_i)} V\left(\langle p_1+\Delta p_1, \cdots, p_m+\Delta p_m\rangle\right) \tag{2}$$

If $z \in PV(I_{e;p})$, then z is called a *pseudozero* of a system. A pseudozero criterion is described as follows.

Theorem 1 (Pseudozero criterion). *$z \in \mathbb{C}^s$ is in $PV(I_{e;p})$ if and only if*

$$|p_i(z)| \leq \left\| \begin{pmatrix} \vdots \\ e_j|z|^j \\ \vdots \end{pmatrix} \right\|, \quad \textit{for } \ i = 1, \cdots, m.$$

The theorem can be proved by using the Hölder inequality.

4 Method for visualization

The method we use to visualize a pseudovariety for the case $m = s$ is to find an algebraic characterization of the boundary of the pseudovariety, that we then display by a numerical parameterization. That is, we find some points on the pseudovariety by a method such as Newton's method, and then use numerical parameterization (i.e. solving a differential equation numerically) with these as initial points. Figures 1, 2, and 3 show projections of these parameterizations for the example

$$x^2 + y^2 - 1 = 0, \ \ 25xy - 12 = 0\,.$$

We see from these figures that the isolated roots of the original system can merge to a double root if the perturbation is as large as $\epsilon = 0.05$.

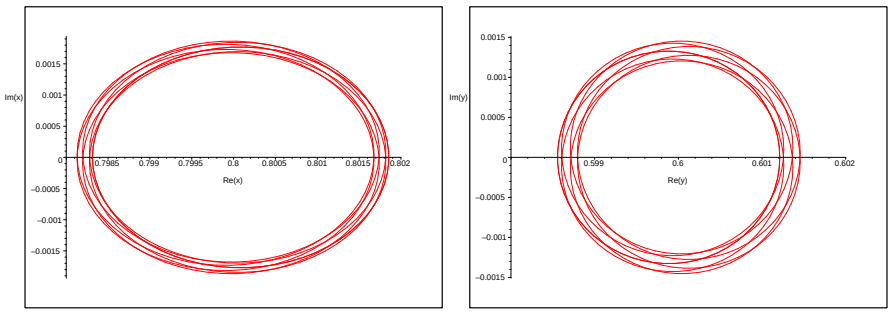

Figure 1: $\epsilon = 0.001$

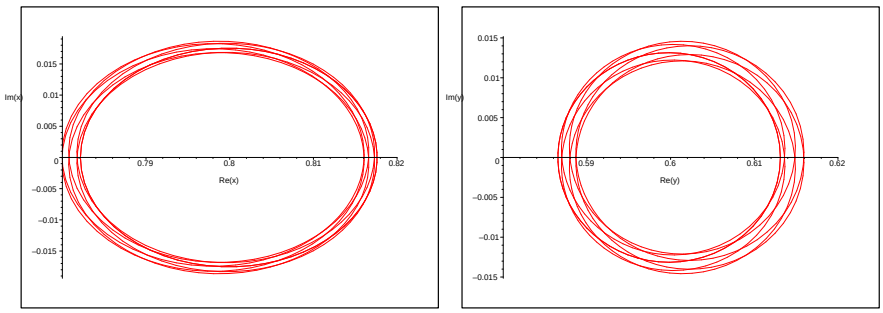

Figure 2: $\epsilon = 0.01$

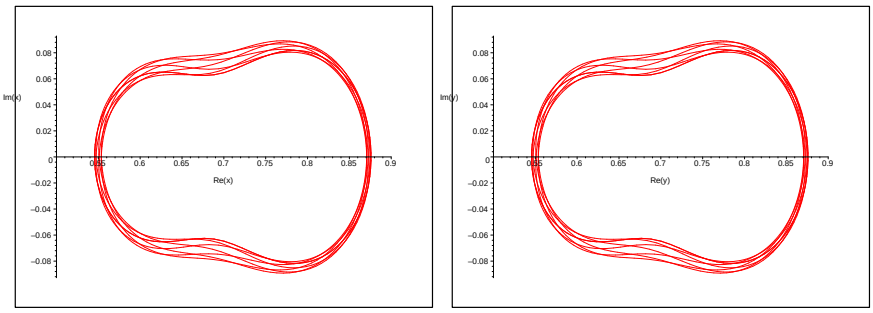

Figure 3: $\epsilon = 0.05$

5 Stability of roots

The condition number for a zero of systems of polynomials was defined in [5] by using linearization. It should be a useful tool for small enough perturbations. Here we would like to show stability of roots from our point of view by using our visualization tools.

If all pseudozeros in $PV(I_{\epsilon;p})$ are isolated, e.g. Figure 1 and Figure 2, the conditioning of roots may be defined as

$$\max_{t_1,\cdots,t_s} \{|x_1(t_1,\cdots,t_s) - x_1^0|,\cdots,|x_s(t_1,\cdots,t_s) - x_s^0|\} \tag{3}$$

where $(x_1^0,\cdots,x_s^0)$ is a root of original system $p_1,\cdots,p_s$. (min also taken, to ensure roots correspond) This may be directly obtained from the solution of the boundaries $x_1(t_1,\cdots,t_s),\cdots,x_s(t_1,\cdots,t_s)$.

For example, about the example in the section **3**, the maximum perturbation of roots is bounded by

$$\begin{aligned} \max\{|x-x^0|,|y-y^0|\} &= 0.001881 \quad \text{for } \epsilon = 0.001 \\ \max\{|x-x^0|,|y-y^0|\} &= 0.020089 \quad \text{for } \epsilon = 0.01 \end{aligned}$$

6 Monodromy

The role monodromy plays here will be explained more fully in the poster. We observed, e.g. in Figure 3, that ϵ may change the monodromy group[1]. This asks for the sizes and values of ϵ and coefficients of Δp_i such that roots interchange. We may find these values as follows.

$$\text{minimize } ||\Delta p_1||^2 + \cdots + ||\Delta p_s||^2$$

subject to

$$p_i + \Delta p_i = 0,\ i = 1,\cdots,s,$$
$$\det(J(p_1+\Delta p_1,\cdots,p_s+\Delta p_s)) = 0,$$

where J is the Jacobian matrix w.r.t. polynomials $p_1+\Delta p_1,\cdots,p_s+\Delta p_s$. We may solve this problem by practical methods of optimization, e.g. Lagrange multipliers.

7 Conclusion

We described about pseudovarieties for systems of polynomials on the following issues :

- A method for visualizing low-dimensional projections of pseudovarieties,
- Understanding stability of roots of nearby systems of polynomials,
- Decide if nearby systems have multiple roots with respect to ϵ.

There are several remaining works concerning about the visualization method :

- Verify the backward error of the solutions,
- Compare visualization methods with the one given in [3].

Furthermore, related works may arise about finding nearest singularities of polynomials :

- Find nearest singularities of algebraic curves of bivariate polynomials,
- Find nearest positive-dimensional system of a zero-dimensional system.

For univariate polynomials, there are some works to compute nearest singular polynomials, e.g. [11].

References

[1] Robert M. Corless, André Galligo, Ilias S. Kotsireas, and Stephen M. Watt. A geometric-numeric algorithm for absolute factorization of multivariate polynomials. In *ISSAC'2002*, pages 37–45. ACM, 2002.

[2] David Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms, An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, second edition, 1996.

[3] J. William Hoffman, James J. Madden, and Hong Zhang. Pseudozeros of multivariate polynomials. *Math. Comp*, 72:975–1002, 2003.

[4] Ronald G. Mosier. Root neighborhoods of a polynomial. *Math. Comp.*, 47(175):265–273, July 1986.

[5] Steve Smale. Complexity theory and numerical analysis. *Acta Numerica*, 6:523–551, 1996.

[6] Hans J. Stetter. Analysis of zero clusters in multivariate polynomial systems. In *ISSAC'96*, pages 127–136. ACM, 1996.

[7] Hans J. Stetter. The nearest polynomial with a given zero, and similar problems. *SIGSAM Bull.*, 33(4):2–4, 1999.

[8] Hans J. Stetter. Polynomials with coefficients of limited accuracy. In *Computer Algebra in Scientific Computing (Eds. V.G.Ganzha, E.W.Mayr, E.V.Vorozhtsov)*, pages 409–430. Springer, 1999.

[9] Kim-Chuan Toh and Lloyd N. Trefethen. Pseudozeros of polynomials and pseudospectra of companion matrices. *Numer. Math.*, 68:403–425, 1994.

[10] Lloyd N. Trefethen. Computation of pseudospectra. *Acta Numerica*, 8:247–295, 1999.

[11] Lihong Zhi and Wenda Wu. Nearest singular polynomials. *Journal of Symbolic Computation*, 26(6):667–676, 1998.

Polynomial Solutions for First Order ODEs with Constant Coefficients

Ruyong Feng and Xiao-Shan Gao
Institute of System Science, Academia Sinica
Beijing 100080, China

Most work on finding elementary function solutions for differential equations focussed on linear equations [4, 2, 6, 1, 3]. In this paper, we try to find polynomial solutions to non-linear differential equations. Instead of finding arbitrary polynomial solutions, we will find the polynomial general solutions. For example, the general solution for $(\frac{dy}{dx})^2 - 4y = 0$ is $y = (x + c)^2$, where c is an arbitrary constant. We give a necessary and sufficient condition for an ODE with constant coefficients to have polynomial general solutions. For a first order ODE of degree n and with constant coefficients, we give an algorithm of complexity $O(n^9)$ to decide if it has a polynomial general solution and to compute the solution if it exists.

References

[1] M. Bronstein, On solutions of linear ordinary differential equations in their coefficient field, *JSC*, **13**, 413-440, 1992.

[2] J.J. Kovacic, An algorithm for solving second order linear homogeneous differential equations, *JSC*, **2**, 3-43, 1986.

[3] Z.M. Li and F. Schwarz, Rational solutions of Ricatti-like PDEs, *JSC*, **31**, 691-716, 2001.

[4] R.H. Risch, The problem of integration in finite terms, *Trans. AMS*, **139**, 167-189, 1969.

[5] J.F. Ritt, *Differential algebra*, Amer. Math. Sco. Colloquium, New York, 1950.

[6] M. Singer, Liouillian solutions of linear differential equations with Liouillian coefficients. *JSC*, **11**, 251-273, 1991.

[7] W.T. Wu, *Mathematics mechanization*, Science Press/Kluwer, Beijing, 2000.

Validated Numerical Methods for Systems and Control Engineering

Masaaki Kanno and Malcolm C. Smith
Control Group, Department of Engineering, University of Cambridge
Trumpington Street, Cambridge, CB2 1PZ, United Kingdom
Tel: +44-1223-339222/2745, Fax: +44-1223-332662
[mk303,mcs]@eng.cam.ac.uk

The aim of this research is to investigate the development of numerical methods for systems and control which have a guarantee on accuracy. An end-product of such research is an algorithm which could be described as "infallible" in the following sense: the user would specify *a priori* a tolerance as small as desired, and the computer would provide an answer which was guaranteed to be accurate to the specified tolerance. Though this is an established subject within Computer Science [5], as well as a few application areas in science and engineering (see [1, Part III]), the direction appears to be quite new in the control systems area. A characteristic feature of previous work is the application of computer algebra tools and the avoidance of floating-point arithmetic.

A starting point for this research has been to consider the computation of certain quantities which are widely used in the modern analysis methods for control system performance, e.g. the H_2 and H_∞ norms of a stable transfer function and the induced L_∞ norm of a linear system. Beyond that, some common algorithms for controller computation have been considered, such as H_2 and H_∞ optimisation. The data input for such problems is typically a rational transfer function or a state-space realisation of the system. As such, the data consists of real number coefficients of polynomials or a finite set of real matrices.

If the computation of a scalar quantity is sought a "validated numerical method" will produce an interval of a user-specified width within which the numerical answer is guaranteed to lie. Such an algorithm to compute the H_∞/L_∞ norm has been implemented using computer algebra methods making use of the result below. All numerical quantities are treated as rational numbers, so that rounding errors inherent in floating-point arithmetic are avoided.

Let $G(s)$ be a matrix of rational functions of the complex variable s. The H_2, H_∞ and L_∞ norms of $G(s)$ are the standard Hardy and Lebesgue space norms, for example,

$$\|G(s)\|_\infty := \sup_{\mathrm{Re}(s)>0} \overline{\sigma}(G(s))$$

and $\|G(s)\|_\infty := \sup_{\omega\in\mathbf{R}} \overline{\sigma}(G(j\omega))$ are the H_∞ and L_∞ norms respectively, where $\overline{\sigma}(\cdot)$ is the largest singular value of a matrix.

Theorem 1 ([4]). *Let $G(s) \in L_\infty$ be rational and write its L_∞ norm $\|G(s)\|_\infty$ as γ_∞. Let $\Phi_\gamma(s) = \gamma^2 I - G^T(-s)G(s)$ and denote $g_\gamma(s^2) = \det \Phi_\gamma(s)$. Moreover, write $g_\gamma(x) = \frac{n_\gamma(x)}{d_\gamma(x)}$. Let $h_\gamma(x)$ be the square-free part of $n_\gamma(x)$ considered as a polynomial in x and γ. Then, $\gamma > \gamma_\infty$ if and only if $\gamma > \overline{\sigma}(G(j\infty))$ and $h_\gamma(x)$ has no roots in $-\infty < x \leq 0$. Further, if γ_∞ is achieved in $0 < \omega < \infty$, then $h_{\gamma_\infty}(x)$ has a multiple root in $-\infty < x < 0$.*

The above theorem reduces the L_∞ norm computation of $G(s)$ to a set of tests which can be carried out using computer algebra tools. Specifically we see that the norm is equal to $\overline{\sigma}(G(0))$, $\overline{\sigma}(G(j\infty))$ or a real root of the discriminant of $h_\gamma(x)$. Each of these quantities is a real root of a univariate polynomial equation. Hence, intervals with arbitrary widths which contain these candidates can be found by means of standard real root localisation methods, for instance, Descartes' rule of signs. To determine which interval contains the true L_∞ norm is then a decision test which amounts to examining whether $\gamma > \overline{\sigma}(G(j\infty))$ and also the existence of roots of $h_\gamma(x)$ in $-\infty < x \leq 0$, both of which can be carried out by way of the Sturm test. Finally, to determine whether $G(s)$ belongs to H_∞ is simply a matter to determine whether the elements of $G(s)$ are analytic in the right half-plane—which is easily done by applying the Routh-Hurwitz test to the least common denominator. We emphasise that the original real number data, and each step of the test, makes use of rational number arithmetic only. Hence, rounding errors are avoided and the method counts as a validated numerical method. This contrasts with all current implementations of the H_∞/L_∞ norm computation [2], which use floating-point arithmetic, and are prone to numerical error.

The development of validated numerical methods for the computation of *controllers* using some standard synthesis procedures, e.g. H_∞ optimisation, presents a number of interesting challenges. Firstly, it may not always be desirable to specify "guaranteed accuracy" in terms of the coefficients of the controller's transfer function or state-space realisation. Other more useful possibilities might be to use one of the standard metrics on dynamical systems, such as the gap or ν-gap metric [7]. In addition, basic questions relating to continuity of solution need to be answered, as in [3], along with algorithm development. There is also the issue that many synthesis algorithms consist of multiple stages where it seems that interval methods [6] will play a vital role in order to give a user-specified width for the tolerance interval at the final stage. Progress in this direction has been reported in [4].

For the currently investigated problems, numerical algorithms based on floating-point arithmetic are commercially available. Our further target is to tackle some problems which do not allow satisfactory algorithms to be implemented using ordinary floating-point arithmetic. Lack of reliable computational tools has been preventing some theoretical developments from being used in practice. Investigation of these problems is crucial and could have a significant impact on various aspects, e.g., practical applications, methodologies of computation of numerical solution, further development of theories.

References

[1] E. Adams and U. Kulisch, editors. *Scientific Computing with Automatic Result Verification*, volume 189 of *Mathematics in Science and Engineering*, London, 1993. Academic Press.

[2] S. Boyd and V. Balakrishnan. A regularity result for the singular values of a transfer matrix and a quadratically convergent algorithm for computing its L_∞-norm. *Systems and Control Letters*, 15(1):1–7, July 1990.

[3] M. Green and M. C. Smith. Continuity properties of LQG optimal controllers. *Systems and Control Letters*, 26(1):33–39, September 1995.

[4] M. Kanno. *Guaranteed Accuracy Computations in Systems and Control*. PhD thesis, University of Cambridge, 2003. In preparation.

[5] W. Krandick and S. Rump, editors. *Journal of Symbolic Computation: Special Issue on Validated Numerical Methods and Computer Algebra*, volume 24, number 6. Academic Press, December 1997.

[6] A. Neumaier. *Interval Methods for Systems of Equations*. Cambridge University Press, Cambridge, 1990.

[7] G. Vinnicombe. *Uncertainty and Feedback — $\mathcal{H}_\infty$ loop-shaping and the ν-gap metric*. Imperial College Press, 2001.

Computing Limits of Sequences

Manuel Kauers[1]
Research Institute for Symbolic Computation
Johannes Kepler Universität Linz
4040 Linz, Austria
manuel@kauers.de

1 Introduction

Automated asymptotic methods only recently became subject of symbolic computation. Contributions like [3, 12, 5, 10, 13] discuss a variety of aspects concerning the asymptotic analysis of continuous, real-valued functions. First approaches employed *generalized series expansion* [5, 3], later methods are based on the theory of *Hardy fields* [6].

So far, it seems that more emphasis was laid on the treatment of continuous functions, and although some mathematical fundaments are already available [1, 2], algorithmic approaches for the discrete case seem rare.

Our poster focuses on limit computation of *sequences* in $\mathbb{Q}$, more specifically, of sequences that can be defined by $\Pi\Sigma$-expressions [7]. We will present a rather simple approach which avoids the use of heavy theory but whose first applications already give promising fine results.

2 A discrete Analogon to l'Hospital's Rule

As is pointed out, e.g., in [4], the difference operator Δ, defined by $\Delta a_n := a_{n+1} - a_n$ can be viewed as a discrete analogon to the differential operator $D\colon f \mapsto f'$ of the continuous world. For instance, the discrete product rule

$$\Delta(a_n b_n) := (\Delta a_n) b_n + a_{n+1} \Delta b_n$$

closely resembles the product rule for differentiation.

Analogously, the summation operator Σ, defined by $\Sigma a_n := \sum_{k=1}^{n} a_k$, corresponds to indefinite integration in the continuous setting.

This correspondence between discrete and continuous operators has been exploited in algorithms for difference equations and indefinite summation (see, e.g., [7, 11]). Following this spirit, the following theorem may be regarded as a discrete analogon to l'Hospital's rule.

Theorem (Discrete L'Hospital's Rule) Let $(a_n)_{n=1}^{\infty}, (b_n)_{n=1}^{\infty}$ be real sequences, both tending to infinity as $n \to \infty$ and suppose that (b_n) is asymptotically strictly monotonous. Then,

$$\lim_{n\to\infty} \frac{a_n}{b_n} = \lim_{n\to\infty} \frac{\Delta a_n}{\Delta b_n},$$

provided that the limit on the right-hand side exists. □

The theorem follows immediately from statement I.1.70 in [9] or II §8, ex. 5 in [8] (see also Lemma 8.7 of [2]).

As an example,

$$\lim_{n\to\infty} \frac{\binom{2n}{n}}{\sum_{k=1}^{n} \binom{2k}{k}} = \lim_{n\to\infty} \frac{\binom{2(n+1)}{n+1} - \binom{2n}{n}}{\binom{2(n+1)}{n+1}} = \lim_{n\to\infty} \left(1 - \frac{n+1}{2(2n+1)}\right) = \tfrac{3}{4}.$$

Note that this limit is found neither by Maple nor by Mathematica because standard limit computation techniques cannot deal with the summation sign in the denominator.

3 Towards an Algorithm

We consider terms that are built from rational functions, indefinite sums, and indefinite products over an indeterminate n. Such expressions are called $\Pi\Sigma$-expressions. A $\Pi\Sigma$-expression is called *admissible,* if the scopes of all occuring product quantifiers are asymptotically postive. It can be shown that every admissible term is asymptotically monotonous, so we need not worry about a decision procedure for monotonicity.

Examples for admissible terms include expressions involving exponentials a^n ($a \geq 0$), factorials $n!$, binomial coefficients $\binom{n}{k}$, etc. while terms like $(-1)^n$ are excluded.

We illustrate our method with by example. In order to compute, for instance, $\lim_{n\to\infty} \frac{\mathrm{H}_n \sum_{k=1}^{n} \mathrm{H}_k}{n\mathrm{H}_{2n}^2}$, where $\mathrm{H}_n := \sum_{k=1}^{n} \frac{1}{k}$ denotes the nth harmonic number, we first apply l'Hospital's rule obtaining

$$\lim_{n\to\infty} \frac{p_n}{q_n} \tag{4}$$

with

$$\begin{aligned} p_n = {} & 4n^3\mathrm{H}_n\mathrm{H}_{2n} + 4n^2 \sum_{k=1}^{n} \mathrm{H}_k + 10n^2\mathrm{H}_n\mathrm{H}_{2n} + 7n \sum_{k=1}^{n} \mathrm{H}_k + 4n^2\mathrm{H}_n \\ & + 4n^2\mathrm{H}_{2n} + 8n\mathrm{H}_n\mathrm{H}_{2n} + 7n\mathrm{H}_n + 3\sum_{k=1}^{n} \mathrm{H}_k + 4n + 6n\mathrm{H}_{2n} \\ & + 2\mathrm{H}_n\mathrm{H}_{2n} + 3\mathrm{H}_n + 2\mathrm{H}_{2n} + 3 \quad \text{and} \\ q_n = {} & 4n^3\mathrm{H}_n^2 + 8n^3\mathrm{H}_n + 10n^2\mathrm{H}_n^2 + 20n^2\mathrm{H}_n + 4n^2 + 8n\mathrm{H}_n^2 + 16n\mathrm{H}_n \\ & + 6n + 2\mathrm{H}_n^2 + 4\mathrm{H}_n + 2 \end{aligned}$$

after expanding products and clearing double denominators. Subsequent applications of l'Hospital's rule would lead to more and more complicated expressions and would most likely never come to a conclusion.

Instead, we determine in the next step a *dominant term* of both numerator and denominator and consider only the quotient of these "leading terms" — the other terms are negligible asymptotically. This is known from calculus exercises like

$$\lim_{n\to\infty} \frac{5n^3 + 3n^2 + 4}{3n^3 + 4n - 5} = \tfrac{5}{3},$$

which we now extend from univariate to multivariate rational functions. Taking the point of view of $\Pi\Sigma$-fields [7], we may regard the expressions p_n and q_n of (4) as polynomials in the ring

$$R = \mathbb{Q}[n, \mathrm{H}_n, \mathrm{H}_{2n}, \sum_{k=1}^{n} \mathrm{H}_k].$$

This view regards n, H_n, etc. no longer as sequences but as pure algebraic variables. For monomials m_1, m_2 in R corresponding to admissible sequences $m_n^{(1)}, m_n^{(2)}$, respectively, let

$$m_1 \prec m_2 \quad :\Longleftrightarrow \quad \lim_{n\to\infty} \frac{m_n^{(1)}}{m_n^{(2)}} = 0.$$

This defines a strict partial ordering $\prec$ on the monomials of R.

Continuing the example of (4), we determine the leading term of both p_n and q_n with respect to $\prec$ by calling the algorithm recursively. In general, since $\prec$ is only a partial ordering, a unique greatest summand need not exists.

This case is more complicated to deal with, but in the situation of (4) the leading terms are unique, and the problem reduces to

$$\lim_{n\to\infty}\frac{p_n}{q_n}=\lim_{n\to\infty}\frac{4n^3\mathrm{H}_n\mathrm{H}_{2n}}{4n^3\mathrm{H}_n^2}=\lim_{n\to\infty}\frac{\mathrm{H}_{2n}}{\mathrm{H}_n}\overset{\text{l'H}}{=}\lim_{n\to\infty}\frac{4n+3}{4n+2}=1.$$

Putting things together, we obtain

$$\lim_{n\to\infty}\frac{\mathrm{H}_n\sum\limits_{k=1}^{n}\mathrm{H}_k}{n\mathrm{H}_{2n}^2}=1.$$

4 Open Problems

Although a Maple implementation of the method outlined in the previous section works quite nice (see Section 5 for examples), some questions remain open and will be subject of further investigations.

Currently, we deal with monotonicity by restricting the admissible sequences to a class where everything is asymptotically monotonous, and we employ a set of ad-hoc criterions for checking that numerator and denominator go to infinity. Naturally, future work will focus on enlarging the class of admissible terms and on a more systematic treatment for deciding unboundedness.

The method described in Section 3 is a semi-decision procedure in the sense that every computed limit is correct but the method may fail or it may run forever. Refined versions should focus on the treatment of sequences which currently cause a failure of the method, and an a priory criteria for termination would be interesting.

5 Some Examples

For giving a flavor of the problems that can be treated by our method, we provide some examples, none of which can be found by the standard general purpose CA-systems.

- $\displaystyle\lim_{n\to\infty}\frac{\sum\limits_{k=1}^{n}\left(k^2\sum\limits_{i=1}^{k}\frac{2^i}{i}\right)}{2^n n}=4,$
- $\displaystyle\lim_{n\to\infty}\frac{\sum\limits_{k=1}^{n}\binom{3k}{k}\binom{5k}{k}}{\binom{3n}{n}\binom{5n}{n}}=\frac{84375}{83351},$
- $\displaystyle\lim_{n\to\infty}\frac{\sum\limits_{k=1}^{n}\frac{2^k}{k^2}\binom{2k}{k}}{\left(\sum\limits_{k=1}^{n}\frac{2^k}{k^2}\right)\sum\limits_{k=1}^{n}\binom{2k}{k}}=\frac{3}{7},$
- $\displaystyle\lim_{n\to\infty}\frac{(\mathrm{H}_n)^3\sum\limits_{k=1}^{n}\frac{1}{\mathrm{H}_k}}{n\sum\limits_{k=1}^{n}\frac{\mathrm{H}_k}{k}}=2$
- $\displaystyle\lim_{n\to\infty}\frac{n^3\left(\sum\limits_{k=1}^{n}\frac{2^k}{k^2}\right)^2}{2^n\sum\limits_{k=1}^{n}\frac{2^k}{k}}=2,$
- $\displaystyle\lim_{n\to\infty}\frac{\mathrm{H}_n\sum\limits_{k=1}^{n}\frac{2^k}{k}}{n\sum\limits_{k=1}^{n}\frac{2^k\mathrm{H}_k}{k^2}}=1,$
- $\displaystyle\lim_{n\to\infty}\frac{\sum\limits_{k=1}^{2n}\frac{2^k k!}{k^2}}{\left(\sum\limits_{k=1}^{n}\frac{4^k}{k^2}\right)\sum\limits_{k=1}^{2n}k!}=\frac{3}{16},$
- $\displaystyle\lim_{n\to\infty}\frac{\sum\limits_{k=1}^{2n}\frac{(\mathrm{H}_k)^2}{k}}{(\mathrm{H}_n)^3}=\frac{1}{3}.$

References

[1] Michael Boshernitzan. Orders of infinity generated by difference equations. In *American Journal of Mathematics*, 106:1067–1089, 1984.

[2] Michael Boshernitzan. Discrete Orders of Infinity. In *American Journal of Mathematics,* 106:1147–1198, 1984.

[3] Keith O. Geddes, Gaston H. Gonnet. A New Algorithm for Computing Symbolic Limits Using Generalized Hierarchical Series. In *Proceedings of ISSAC'88,* 1988

[4] Ronald L. Graham, Donald E. Knuth, Oren Patashnik. *Concrete Mathematics.* Addison-Wesley, second edition, 1994.

[5] Dominik Gruntz. *On Computing Limits in a Symbolic Manipulation System.* ETH Diss 11432, 1996.

[6] Maxwell Rosenlicht. Hardy fields. In *Journal of Mathematical Analysis and Applications,* 93:286–311, 1983.

[7] Michael Karr. Summation in Finite Terms. In *Journal of the ACM,* 28:305–350, 1981

[8] Konrad Knopp. *Theorie und Anwendung der unendlichen Reihen.* Springer, fifth edition, 1964.

[9] George Pólya, Gabor Szegő. *Problems and Theorems in Analysis I.* Springer, 1978.

[10] Bruno Salvy, John Shackell. Symbolic Asymptotics: Functions of Two Variables, Implicit Functions. In *Journal of Symbolic Computation,* 25:329–349, 1998.

[11] Carsten Schneider. *Symbolic Summation in Difference Fields.* PhD Thesis RISC, 2001.

[12] John Shackell. Growth Estimate of exp-log Functions. In *Journal of Symbolic Computation,* 10:611–632, 1990.

[13] John Shackell, Bruno Salvy. Symbolic Asymptotics: Multiseries of Inverse Functions. In *Journal of Symbolic Computation,* 27:543–563, 1999.

Using computer algebra system SINGULAR:PLURAL for computations in noncommutative polynomial algebras

Viktor Levandovskyy and Hans Schönemann
Center for Computer Algebra
Fachbereich Mathematik, Universität Kaiserslautern
Postfach 3049 D-67653 Germany
[levandov,hannes]@mathematik.uni-kl.de

We give a detailed account of algorithms and applications provided with SINGULAR:PLURAL (we call it PLURAL for short). The poster is done in form of (big) one-page introduction to the capabilities of the system.

Computational objects

We give examples of construction of several important algebras, arising from various applications, highlighting the comfortability of PLURAL's interface for such tasks. We illustrate several requirements, which we impose on objects (algebras and modules over them) to be handled within PLURAL. In particular, the data type `ideal` (resp. `module`) correspond to left structures but there are special procedures, interpreting the input as a set of two–sided generators of an ideal.

Algorithms and applications

We stress the fact, that with our previous works we can use the same terminology and, conceptually, the same algorithms as in the commutative case. Of course, the background and motivation of computations are sometimes very different, like for the elimination, computations modulo an ideal and even for Gröbner basis itself.

On the other side, there are algorithms, appearing only in the noncommutative case, like the two–sided Gröbner bases for ideals, computation of the central subalgebra and highly nontrivial test for a maximality of a given left ideal and so on. We discuss especially hard computational problems, like the elimination, coming from D–modules theory and algebraic dependence of central elements in quantum algebras.

Combined computations

SINGULAR:PLURAL is a kernel extension of a well–known computer algebra system SINGULAR. Thus PLURAL inherits the same user programming language, many SINGULAR libraries, help system and general principles. Moreover, we are able to perform *combined*, that is, both commutative and noncommutative computations (e.g. of Gröbner bases) at the same time, where the commutative part of such a computation is done by fast and functionally rich SINGULAR engine. This feature is especially useful in applications like the classification problems, arising in noncommutative algebras, where investigations of many properties could be reduced to the purely commutative problems. We demonstrate advantages of combined computations by computing the one–dimensional representations of quantum algebras $U_q'(\mathfrak{so}_3)$ in three cases, obtained by specializing the deformation parameter q at different values. As far as we know, SINGULAR:PLURAL is the only modern system allowing user to do such combined computations. We will enhance this interplay further.

Example: Magic numbers in $U(\mathfrak{g}_2)$ over $\mathbb{Q}$

It is an important but also entertaining example, showing the whole palette of PLURAL abilities, especially useful for investigating concrete problems.

- number of algebra generators $x_1, \ldots, x_n$: $n = 14$
- let $I = \langle x_1^3 \rangle$ be a two–sided ideal (x_1 the image of the shortest positive root of $\mathfrak{g}_2$)
- left Gröbner basis of I consists of: 106 generators
- $\mathbb{Q}$–vector space dimension of $U(\mathfrak{g}_2)/I$ is finite: 50
- 1^{st} syzygy module of I consists of: 3244 generators
- we compute both generators Z_2, Z_6 of the center of $U(\mathfrak{g}_2)$, including previously unpublished Z_6 of degree 6, which consists of 754 monomials
- we compute the intersection of I with the center $\langle Z_2^2 - 6Z_2, Z_6 \rangle \subset \mathbb{K}[Z_2, Z_6]$
- finally, we are able to compute the center of $U(\mathfrak{g}_2)/I$, whose non-constant part consists on one polynomial only.

You can find more information and downloads for SINGULAR:PLURAL at our website

`http://www.singular.uni-kl.de/plural`

Exact Implicitization of Polynomial Curves and Surfaces

Ilias Kotsireas, Edmond Lau, Richard Voino
Department of Computing
Wilfrid Laurier University
75 University Avenue
Waterloo Ontario N2L 3C5 Canada
ikotsire@wlu.ca

Recent advances in the eigenvalue method for implicitization include an efficient C/GMP implementation for polynomial curves and surfaces and investigation of structural properties of implicitization matrices. These advances will be illustrated with numerous examples. CAGD applications will be discussed as well.

MAD: a flexible system for authoring mathematical documents

Ludovic Meunier
Algorithms Project, INRIA Rocquencourt
Ludovic.Meunier@inria.fr

1 Presentation and motivation

MAD (*Mathematical Abstract Document*) is a document preparation system integrated with MAPLE. MAD provides a simplified model for describing a collection of logically structured documents with mathematical content. MAD also provides routines for manipulating the data structure that represents such documents and for exporting this data structure into standard formats, such as LaTeX, PostScript, PDF and HTML.

Integrating MAD into a computer algebra system such as MAPLE brings the following original features: 1) the automation of the computation and of the typesetting of mathematical formulæ, graphs and tables; 2) the automation of the building of documents thanks to the underlying programming language.

The system MAD was created in order to generate the interface of the Encyclopedia of Special Functions (ESF, `http://algo.inria.fr/esf`) [2]. The ESF is an *automatically* generated encyclopedia, where all mathematical formulæ and graphs are computed by generic algorithms. The ESF can be compared up to some extent with the Abramowitz and Stegun's *Handbook of Mathematical Functions* [1] and the DLMF project[2]. This kind of mathematical document sets the following requirements for it to be easily used: pretty display (human readable format) and semantics (machine understandable format) of the mathematical formulæ, cross references for browsing and quoting, paper version in several formats. The system MAD is designed to automate the edition and the production of such mathematical documents.

2 Overview of the system

2.1 Abstract document

MAD allows to concentrate on the semantics of the content and its logical architecture within the document.

An abstract document is represented by a recursive data structure of documents, chapters, sections and paragraphs. The paragraphs are the leaves of the data structure and they are meant to hold the content of the document. This content is made up of plain text, mathematical formulæ (inline and 2D), graphs, cross-references, lists, tables, ...

Allowing the nesting of sectioning descriptors into each other simplifies the syntax. For instance, the single MAD command `Section` is mapped to the LaTeX counterparts `\section`, `\subsection`, `\subsubsection`, ... according to the nesting depth of the descriptor. Allowing nested documents is another facility for expressing a *collection* of abstract documents that are actually spread over several physical files as a single data structure. Note here that the physical mapping onto files and the logical structure of a collection of documents are totally independent.

MAD also supports cross-references (hyperlinks). The system of cross-references can be incrementally updated, which allows to refer to documents that are generated in different MAPLE sessions.

2.2 Mathematical formulæ

The semantics of a mathematical object is kept in the MAPLE format, from which other formats, such as MathML and OpenMath can be derived. The display, however, is encoded in a simple pretty printing format that describes

[2] `http://dlmf.nist.gov`

the relative positioning of the operands within the mathematical expression. This format supports the automatic line-breaking of large expressions and the adjustment of the height of sized symbols (*e.g.* parenthesis). This format involves a reduced set of 14 primitive instructions and it is easily mapped to LaTeX.

2.3 Translators

A translator to a target format allows to translate the MAD format into the target format. A translator is an *independent* piece of software that *only* provides the format-specific pieces of code. The design of splitting the implementation between generic code and format-specific code makes the entire system MAD easily maintainable and also extendable, since the development of new translators requires relatively little work.

The way the documents look (*i.e.* their *styles*) is to be set at translator level. Two translators are currently available.

– *Translator to LaTeX.* In order to have more control over spacing and line-breaking, this translator produces AMS-LaTeX. The most important part of the translator lies in its implementation of the mathematical format. A recursive computation of approximate widths and heights of TeX boxes determines the appropriate sizes for parentheses and brackets, as well as the appropriate locations of line-breaks. The formats PostScript and PDF are produced from the LaTeX source by standard tools (`dvips` and `pdflatex` respectively).

– *Translator to* HTML. This translator produces HTML 4.0. Mathematical formulæ are displayed as GIF or PNG pictures, which are produced by the script `htmx`[3] from the output of the previous translator. Note that, instead of pictures, mathematical formulæ could be rendered in other formats (see Section 2.2). Note that HTML could also be produced directly from the LaTeX output by tools, such as `hevea`[4] and `latex2html`[5].

2.4 Authoring documents

An abstract document is represented by a recursive MAPLE data structure. MAD provides a suite of routines for building up and manipulate such a data structure. In particular, MAPLE expressions are automatically converted into the pretty printing format described in Section 2.2. A translator is then used to export the abstract document into some other format.

3 Software

The system MAD is implemented in MAPLE as the collection of the following packages: `DocumentGenerator` (main), `LaTeX` and `HTMX` (translators), `CommonLib` (tools). These packages form a standalone software that runs on MAPLE 8 on UNIX-like operating systems. The software is distributed under GNU Lesser General Public Licence and is available at `http://algo.inria.fr/meunier`.

The software has been designed to be easily customized (several options and parameters are available) and extended (new translators can be added).

Possible extensions to the system MAD are 1) a translator to the MWS (MAPLE worksheet) format for keeping help files up to date automatically; 2) a front-end to MAPLE for simplifying the syntax and the use of MAD with simple documents.

4 Frequently Asked Questions

– *What is* MAD *for?* MAD is designed to *automate* the edition and the conversion of mathematical documents into standard formats.

[3] `http://www.math.uic.edu/~fields/htmx/`
[4] `http://pauillac.inria.fr/~maranget/hevea/`
[5] `http://www.latex2html.org/`

– *What about the Web?* MAD can produce HTML and it is being tuned towards a CGI-based server for exchanging mathematical knowledge over the Internet.

– *What about LaTeX?* MAD can produce LaTeX output and relies on the program `latex` for producing PDF and PostScript.

– *What about* OpenMath*,* MathML*,* XML*, ...?* MAD can integrate *any* of these formats, provided that the corresponding translators exist. Again, MAPLE features a translator towards MathML that can be used.

– *Does* MAD *work?* Yes ! See the ESF at `http://algo.inria.fr/esf`.

References

[1] Milton Abramowitz and Irene A. Stegun, editors. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Dover Publications Inc., New York, 1992. Reprint of the 1972 edition.

[2] Ludovic Meunier and Bruno Salvy. ESF: An automatically generated Encyclopedia of Special Functions. In *ISSAC'03*, 2003.

Implementation of Prime Decomposition of Polynomial Ideals over Small Finite Fields

Masayuki Noro
Department of Mathematics, Kobe University
noro@math.sci.kobe-u.ac.jp

Kazuhiro Yokoyama
Graduate School of Mathematics,
Kyushu University

An algorithm for the prime decomposition of polynomial ideals over small finite fields is proposed and implemented. Here "small" means that the order of a finite field is small enough to fit in a single machine word. In particular, we are interested in cases where the order is so small that we cannot apply usual methods which is successful in cases of characteristic 0. To overcome this difficulty, [8] introduced the notion of "separable ideals" and "separable closure of ideals", and we propose a precise algorithm for the prime decomposition of polynomial ideals over small finite fields based on [8]. Our final goal is to develop a practical algorithm for the primary decomposition of a polynomial ideal over a finite field. For this purpose we can apply the "localization technique" of [7], which enables us to extract primary components from prime divisors. This does not depend on the characteristic of the coefficient field. Therefore primary decomposition computations can be efficiently reduced to prime decomposition computations.

For computing the prime decomposition we employ the well-established strategy of [3] with modification necessary for treating difficulties in small characteristic cases. For an inseparable ideal H, we compute its separable closure $\mathrm{sc}(H)$, which is the inverse image of H by "Frobenius map" ϕ. Then we compute $\{P_1, \ldots, P_k\}$, the prime divisors of $\mathrm{sc}(H)$, from which we obtain those of H as $\{\sqrt{\phi(P_1)}, \ldots, \sqrt{\phi(P_k)}\}$. Each $\sqrt{\phi(P_i)}$ can be computed by applying the efficient method of [5] based on "inverse Frobenius map computation".

To achieve the most efficient implementation, the implementation of sub-procedures, including fundamental tools such as minimal polynomial computation and polynomial factorization over finite fields, is done as follows:

- As we want to compute the prime divisors of a given ideal I, we can make use of "partial decompositions" [2, 7]. This is done by first applying the "pre-decomposition" suggested by [7], and involves factoring all elements in a Gröbner basis of I. For each computed ideal, we then apply an algorithm based on [8].

- A partial decomposition of the given ideal called "intermediate decomposition" is computed using factorization of the minimal polynomial of each variable. Components of the intermediate decomposition tend to be prime, and so this decomposition improves computational efficiency.

- The whole computation can be done without computing the radical of the given ideal I. But in some cases it is necessary to compute the intersection of computed prime ideals in order to improve efficiency. In these cases, the radical ideal of a given ideal is computed as a by-product.
- Primitive root representation is employed to allow efficient arithmetic over extension fields $GF(q)$, as it is well suited to polynomial factorization over $GF(q)$ (see [6]). The most recent algorithms of [1, 6] are used for polynomial factorization over $GF(q)$.
- The minimal polynomial of each variable is computed by an "FGLM-type" method if it is considered to be more efficient than an elimination ideal computation by the Buchberger algorithm. Practically speaking it is hard to predict the superior algorithm for a given input ideal. OpenXM [4] provides a machinery to execute "competitive computation" and we apply it for minimal polynomial computation.

All procedures were implemented in Risa/Asir, with computational tests of a number of examples conducted. Experimentally, the algorithm was found to work very well with a set of examples. Although testing using only a limited number of examples cannot fully verify the efficiency of the implementation, it does give an indication of the quality of the algorithm and its implementation.

References

[1] Bernardin, L., Monagan, M.B. (1997). Efficient multivariate factorization over finite fields. in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC-12). Lecture Notes in Computer Science* **1255**, 15–28.

[2] Caboara, M., Conti, P., Traverso, C. (1997) Yet another ideal decomposition algorithm. in Proceedings of AAECC 12, L.N.C.S. **1255**, Springer, 39-54.

[3] Gianni, P., Trager, B., Zacharias, G. (1988). Gröbner bases and primary decomposition of polynomial ideals. *J. Symb. Comp.* **6**, 149-167.

[4] Maekawa, M., Noro, M., Ohara, K., Takayama, N., Tamura, Y. (2001). The Design and Implementation of OpenXM-RFC 100 and 101. in Proceedings of ASCM2001, World Scientific, 102–111.

[5] Matsumoto, R. (2001) Computing the radical of an ideal in positive characteristic. *J. Symb. Comp.* **32**, 263-271.

[6] Noro, M., Yokoyama, K. (2002). Yet another practical implementation of polynomial factorization over finite fields. in Proceedings of ISSAC'02, ACM Press, 200-206.

[7] Shimoyama, T., Yokoyama, K. (1996). Localization and primary decomposition of polynomial ideals. *J. Symb. Comp.* **22**, 247-277.

[8] Yokoyama, K. (2002). Prime decomposition of polynomial ideals over finite fields. in Mathematical Software Proceedings of ICMS2002, World Scientific, 217-227.

LU based algorithms for characteristic polynomial over a finite field

Clément Pernet
LMC-IMAG, BP 53
51 av. des Mathématiques.
38041 Grenoble, France.
Clement.Pernet@imag.fr

Zhendong Wan
Dept of CIS.,
University of Delaware
Newark, Delaware 19716
wan@cis.udel.edu

Linear algebra computation over finite fields is important because many symbolic computation problems directly or indirectly rely on linear algebra using efficient finite fields representation.

From a theoretical point of view, the time complexity of many linear algebra algorithms has been drastically reduced during the last decade. For some of them, such as characteristic polynomial computation [3], efficiency has been achieved thanks to the use of block algorithms, grouping operations into matrix multiplications. Thus, the time complexity of this problem is reduced to the one of matrix multiplication times a logarithmic factor: $M(n)log(n)$, where n is the dimension of the square matrix given as entry.

From a practical point of view, the use of block algorithms also permits one to benefit from the high performance of LEVEL3 BLAS matrix multiplications, as presented in [1].

The poster will focus on algorithms for the computation of the characteristic polynomial of matrices over a finite field, based on a block LU factorization algorithm, which is the LSP factorization of [2]. LSP is a generalization of LUP factorization for singular matrices.

We give a new algorithm computing the minimal polynomial of the Krylov sequence of a dense matrix and a vector, using a LU factorization. Its time complexity is $O(M(n)log(n))$. This algorithm is also Monte-Carlo for the computation of the minimal polynomial of a dense matrix.

We also present two algorithms for the computation of the characteristic polynomial, both using LSP factorizations and the previous minimal polynomial computation routine:

1. A variation of the Keller-Gehrig algorithm [3], having time complexity $O(M(n)log(n))$. Our contribution consists in replacing Keller-Gehrig's step-form elimination by the LSP factorization. Although the asymptotic complexity remains the same this approach appears to be well suited for this application, and enables high practical efficiency.

2. A variation of the Krylov method, having time complexity $O(n^3)$. It is only based on the two following routines: LUP factorization and triangular matrix system solver. This algorithm is also of high practical efficiency.

Lastly, we present some experimentats results for our implementation of these algorithms. It will enables one to see that these two algorithms are complementary, and the number of invariant factors of the entry matrix determines which one should be used: `Krylov-LU` for a few blocks, and `Keller-Gehrig` for many blocks. These routines have been integrated in a specific package designed for efficient dense linear algebra over finite fields: FFLAS_FFLAP available on the web at [4]

References

[1] J.-G. Dumas, T. Gautier, and C. Pernet. Finite field linear algebra subroutines. In T. Mora, editor, *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, Lille, France*. ACM Press, New York, July 2002.

[2] O. H. Ibarra, S. Moran, and R. Hui. A generalization of the fast LUP matrix decomposition algorithm and applications. *Journal of Algorithms*, 3(1):45–56, Mar. 1982.

[3] W. Keller-Gehrig. Fast algorithms for the characteristic polynomial. *Theoretical computer science*, 36:309–317, 1985.

[4] C. Pernet. Finite field linear algebra (subroutine-package), 2003. `www-lmc.imag.fr/Jean-Guillaume.Dumas/FFLAS`.

Symbolic Solution of Simple BVPs on the Operator Level: A New Approach

Markus Rosenkranz[6]
Research Institute for Symbolic Computation
Johannes Kepler Universität Linz
4040 Linz, Austria
`Markus.Rosenkranz@risc.uni-linz.ac.at`

1 Introduction

Computer algebra entred the territory of *differential equations* around 1970, when various human-imitating heuristics were superseded by truly algorithmic methods supported by a rapidly growing algebraic theory of integration [11, 12]. The well-known results by Risch [15] established a satisfactory basis for integrating elementary transcendental functions, also including important bounds for soving linear first-order ODEs. Since then, the research on symbolic solutions of ODEs and PDEs expands vigorously along manifold lines, e.g. the Prelle-Singer approach using first integrals [14, 10], the holonomic paradigm [5], differential resultants and Gröbner bases [8], and symmetry analysis [1].

Along the same lines, one might expect that computer algebra has something to say also about *boundary value problems* (BVPs), all the more so since they are at least as important in applications as plain ODEs and PDEs. However, this field—a traditional territory of analysis—seems to be somewhat of a terra incognita for the computer-algebra community. Some people fail to see the fundamental difference (as seen from a symbolic viewpoint!) between ODEs/PDEs and BVPs. Let us therefore start out with a careful specification of the problem that will highlight this difference. (At the same time, we will restrict ourselves to the modest case covered by our algorithm, but this is immaterial for the issue just touched. In fact, we see many possibilities of extending our approach, but we will not discuss this here.)

2 Problem Specification

Let $[a,b]$ be a finite interval in $\mathbb{R}$, and let T be a linear differential operator with constant coefficients given by

$$T\,u = c_0\,u^{(n)} + \ldots + c_{n-1}\,u' + c_n\,u,$$

where c_0 is nonzero. We view T as a linear operator on the Banach space $(C[a,b], ||\cdot||_\infty)$ with dense domain of definition $\mathfrak{D}(T) = C^n[a,b]$. The boundary operators $B_1,\ldots,B_n$ are defined on the same domain; for each $i = 1,\ldots,n$ we have

$$\begin{aligned} B_i\,u = {} & p_{i,0}\,u^{(n-1)}(a) + \ldots + p_{i,n-1}\,u'(a) + p_{i,n}\,u(a) \\ & + q_{i,0}\,u^{(n-1)}(b) + \ldots + q_{i,n-1}\,u'(b) + q_{i,n}\,u(b), \end{aligned}$$

where the coefficients $p_{i,j}, q_{i,j}$ are real numbers. Now the *boundary value problem induced by* T *and* $B_1,\ldots,B_n$ is to find for each right-hand side $f \in C[a,b]$ a function $u \in C^n[a,b]$ (we assume its existence and uniqueness

throughout this note, restricting ourselves to regular BVPs) such that

$$\begin{aligned} &T\,u = f, \\ &B_1\,u = \ldots = B_n\,u = 0. \end{aligned} \tag{5}$$

Now we *could* view this as a differential equation parametrized by the forcing function f, but this is a rather artifical interpretation in the sense of computer algebra: Solution algorithms for ODEs are typically specialized to a certain class of functions —like Liouvillian extensions—for exploiting all the structural information available. However, in the BVP (5), we view f as ranging over the nonalgorithmic domain $C[a,b]$. Therefore the natural interpretation of (5) is to search not for a *function* $u \in C^n[a,b]$ for the infinitely many instantiations of f but for an *operator* $G : C[a,b] \to C^n[a,b]$, mapping each $f \in C[a,b]$ to "its" solution $u \in C^n[a,b]$. The operator G is known as the *Green's operator* [18].

The Green's operator can be defined analogously for many other types of BVPs for ODEs and PDEs, and it can often be described as an integral operator having a so-called *Green's function* g as its kernel. In the case of (5), this is indeed possible [6], leading to the Green's operator

$$G\,f(x) = \int_a^b g(x,\xi)\,f(\xi)\,d\xi. \tag{6}$$

Thus one *can* reduce the search for the operator G to the search of the (bivariate!) function g, and there is a solution method going along these lines [9]. However, working directly on the *operator level* seems more natural to us, and we will now outline a new method for computing G in a suitable polynomial setting [16, 17]. The Green's function g can be extracted from G in a trivial postprocessing step, if this is desired. Apart from the conceptual advantages, our method may also be superior to [9] on efficiency grounds, but this should be analyzed in detail later.

3 The Green's Polynomials

Our approach models the involved differential, integral and boundary operators as *noncommutative polynomials*: The differentiation $u \mapsto u'$ is represented by the indeterminate D, the antiderivative operator $u \mapsto (x \mapsto \int_a^x u(\xi)\,d\xi)$ by A, its dual $u \mapsto (x \mapsto \int_x^b u(\xi)\,d\xi)$ by B, the left boundary operator $u \mapsto (x \mapsto u(a))$ by L, and the right counterpart $u \mapsto (x \mapsto u(b))$ by R. Moreover, we have a parametrized family of multiplication operators M_f representing $u \mapsto (x \mapsto f(x)\,u(x))$. The functions f are assumed to range over some algebra $\mathfrak{F}$ of functions that should fulfill suitable closure axioms coming out as a natural extension of the well-known axioms for a differential algebra. We call such an $\mathfrak{F}$ an *analytic algebra*; see [17] for details. For the results presented here, it is sufficient to take $\{x \mapsto x^k\,e^{\lambda x} \mid k \in \mathbb{N} \wedge \lambda \in \mathbb{C}\}$ for $\mathfrak{F}$. The noncommutative polynomial ring

$$\mathfrak{An}(\mathfrak{F}) = \mathbb{C}\langle A, B, D, L, R, M_f \mid f \in \mathfrak{F}\rangle$$

will be called *analytic polynomials*.

The *algorithm for solving* a BVP of the type (5) proceeds in three phases. First we compute a projector $P \in \mathfrak{An}(\mathfrak{F})$ onto the nullspace of T by using some trivial linear algebra on the fundamental system of T (the latter is typically presupposed when solving a BVP). Second we employ the Moore-Penrose theory [13] for reducing (5) to the right-inversion problem $GT = 1 - P$, which can be solved immediately by factoring the characteristic polynomial of T. Third we reduce the resulting expression $(1-P)\,T^\blacklozenge$ (with $T^\blacklozenge \in \mathfrak{An}$ being the right inverse) with respect to a carefully selected system of 36 polynomial equations. The result is a polynomial in $\mathfrak{An}(\mathfrak{F})$ in a normal form that allows to read off the Green's function (6) immediately.

The polynomial equation system is the core idea for the algebraization employed in our approach. It captures the essential interactions between the indeterminates of $\mathfrak{An}(\mathfrak{F})$. For example, we have the equation $D\,A = 1$, expressing the Fundamental Theorem of Calculus, the product rule, and integration by parts. We have proved that the rewrite system induced by these equations is noetherian and confluent; thus one can regard it as a noncommutative Gröbner

basis [2, 3]. The ideal generated by them identifies all those operators that should be equal on analytic grounds; we call the corresponding factor algebra the *Green's polynomials*. The algorithm computes the Green's polyomial associated with the Green's operator by providing its canonical representative.

4 An Example

The algorithm outlined above has been implemented in Mathematica, using the framework of the Theorema system, an integrated working environment for proving, solving and computing in various domains [4]. The following call solves the boundary value problem for the differential operator $T = D^2 + 2D + 1$ for the boundary conditions $L\,u = 0$ and $R\,u = 0$ on the interval $[0, \pi]$. We give the input and output verbatim:

$$\begin{aligned}
\texttt{In[13]:=}\quad & \mathrm{Compute}[\mathrm{Green}[D^2 + 2D + 1, \langle L, R\rangle, \mathrm{by} \to \mathrm{GreenEvaluator}] \\
\texttt{Out[13]=}\quad & (1 - \pi^{-1})\lceil e^{-x}x\rceil A\lceil e^{x}\rceil - \lceil e^{-x}\rceil A\lceil e^{x}x\rceil + \pi^{-1}\lceil e^{-x}x\rceil A\lceil e^{x}x\rceil \\
& - \pi^{-1}\lceil e^{-x}x\rceil B\lceil e^{x}\rceil + \pi^{-1}\lceil e^{-x}x\rceil B\lceil e^{x}x\rceil
\end{aligned}$$

The multiplication operators M_f are denoted by $\lceil f\rceil$ for the sake of readability. Note that one can immediately read off the corresponding term $g(x, \xi)$ for the Green's function (6), which is typically defined by a case distiction on $\xi < x$ and $\xi > x$: The summands with A go into the first case, those with B into the second; the multiplication operators before A and B yield terms in x, those after yield terms in ξ.

References

[1] G. W. Bluman, S. Kumei. *Symmetries and Differential Equations*. Springer, New York, 1989.

[2] B. Buchberger. *Introduction to Gröbner Bases*. In [3]. Pages 3–31.

[3] B. Buchberger, F. Winkler. *Gröbner Bases and Applications*, Cambridge University Press, Cambridge (UK), 1998.

[4] B. Buchberger, W. Windsteiger. *The Theorema Language: Implementing Object- and Meta-Level Usage of Symbols*. In: Proceedings of Calculemus 98 (Eindhoven, Netherlands), 1998.

[5] F. Chyzak. *Gröbner Bases, Symbolic Summation and Symbolic Integration*. In [3]. Pages 32–60.

[6] E. A. Coddington, N. Levinson. *Theory of Ordinary Differential Equations*, McGraw-Hill Book Company, New York, 1955.

[7] J. H. Davenport. *The Equation $y' = fy = g$*. Proceedings of EUROSAM 84. Springer Lecture Notes of Computer Science 174, Springer, Berlin, 1984. Pages 341–350.

[8] G. Carrá-Ferro. Differential Gröbner Bases in One Variable and in the Partial Case. *Mathematical Computer Modelling* **25**, Pergamon Press, 1997. Pages 1–10.

[9] E. Kamke. *Differentialgleichungen und Lösungsmethoden (Volume 1)*, Teubner, Stuttgart, tenth edition, 1983. Pages 190–193.

[10] Y.-K. Man. Computing Closed Form Solutions of First Order ODEs Using the Prelle-Singer Procedure. *Journ. Symb. Comp.* **16**, 1993. Pages 423–443.

[11] J. Moses. *Symbolic Integration*. PhD Thesis & Project MAC TR-47, Massachusetts Institute of Technology, 1967.

[12] J. Moses. Symbolic Integration: The Stormy Decade. *Communications of the ACM* **14**, 1971. Pages 527–537.

[13] M. Z. Nashed (Ed.). *Generalized Inverses and Applications*, Proceedings of an Advanced Seminar Sponsored by the Mathematics Research Center, Academic Press, New York, 1976. Pages 57–61.

[14] M. J. Prelle, M. F. Singer. Elementary First Integrals of Differential Equations. *Trans. Amer. Math. Soc.* **279**(1), 1983. Pages 215–229.

[15] R. H. Risch. The Problem of Integration in Finite Terms. *Transactions of the ACM* **139**, 1969. Pages 167–189.

[16] M. Rosenkranz, B. Buchberger, H. W. Engl. Solving Linear BVP via Non-Commutative Gröbner Bases, accepted to *Applicable Analysis*, 2003.

[17] M. Rosenkranz. *A Polynomial Approach to Linear Boundary Value Problems*. PhD thesis, to be submitted in 2003.

[18] I. Stakgold. *Green's Functions and Boundary Value Problems*, John Wiley & Sons, New York, 1979.

On Optimization Problems

Wu Tian-Jiao
Academy of Mathematics & Systems Sciences
Chinese Academy of Sciences, Beijing, 100080
tjwu@mx.amss.ac.cn

Let $\mathbf{R}$ = real field, $X = (x_1, x_2, \ldots\ldots, x_n)$, D a domain in $\mathbf{R}^n(X)$, f, g_i, h_j in $\mathbf{R}[X]$.

Optimization Problem O: Find optimal (i.e. greastest or least) value of $f(X)$ under constraints $h_j(X) = 0$, $g_i(X) \neq 0$ with X in D.

Let us cite below some practical examples of optimization.

Ex. 1. Some nonlinear programming problem, cf. [1] :

$$f(x_1, x_2) = (x_1 - 2)^2 + (x_2 - 1)^2 \tag{1}$$

$$\text{Constraints} : \begin{cases} x_1 - 2 * x_2 + 1 = 0 \\ x_1^2 + 4 * x_2^2 - 4 \leq 0 \end{cases}$$

Question: Determine the least value of f with corresponding x_1 and x_2.

Answer: *Least (f)* $= 9 - \frac{23}{8} * \sqrt{7} \approx 1.393464981$ with $(x_1, x_2) = (\frac{-1+\sqrt{7}}{2}, \frac{1+\sqrt{7}}{4})$.

Ex. 2. Motztkin Polynomial, cf. [2]:

$$f = 1 + x_2^2 * x_2^4 + x_1^4 * x_2^2 - 3 * x_1^2 * x_2^2 \tag{2}$$

Question: Determine definiteness of f.

Answer: f is positive semidefinite.

Optimization problems are usually formulated for f, g_i, h_j to be arbitrary differenatiable functions and then solved by means of Lagrangian Multipliers in using various kinds of numerical methods. However, this will give only solutions in the form of *local* extremal (i.e. maximum or minimum) values but usually not *global* optimal (i.e. greatest or least) one. On the other hand, if f, g_i, h_j are all *polynomials* (i.e. $f, g_i, h_j \in \mathbf{R}[\mathbf{X}]$), then we have a method which will give the *global* optimal value, so far it is already known to exist. Our method is based on the following

Finite Kernel Theorem. If f, g_i, h_j are all *polynomials* with $HS = \{h_j\}$, $GS = \{g_i\}$, then there is a *finite* set $K = K\{f, GS, D, HS\}$ of *real* values such that the $global$ optimal value of Problem O is equal to the optimal value of the finite set K, so far it is already known to exist.

For a proof of the theorem cf. [3], Chap. 5, § 5.5.

We call the finite real value set $K = K\{f, GS, D, HS\}$ in the theorem the **Finite Kernel Set** of the Optimization Problem O. There is a package $e\_val(HS, Ord, NZ)$ due to D.K.Wang which permits us to determine the set $K = K\{f, GS, D, HS\}$, cf. [4].

Format of e_val software package is $e\_val(HS, Ord, NZ)$, in which $Ord = [x_n, x_{n-1}, \ldots, x_1]$ is the set of variables arranged in natural descending order and NZ is a set of polynomials designated to be non-zero, introduced for simplifying the computations.

Our method is different from the usual known ones in that the computations are *symbolic* instead of numerical ones. Theoretically it is based on computer algebra and algebraic geometry. The author had applied the above method to get global optimal solutions of various kinds of practical problems: nonlinear programming, traffic control, inequalities proving, definiteness of polynomials, bilevel programming, etc.

Below we will apply the **Finite Kernel Theorem** and software package e_val to the solving of previous problems.

Solving of Ex. 1.

It is seen that least value of f in [1] should exist.

To decide the $least(f)$ let x_0 be a new variable and set $h_0 = f - x_0$.

Set $HS = \{h_0\}$, $Ord = [x_3, x_2, x_1, x_0]$ and $NZ = \emptyset$.

Our problem is equivalent to the determination of optimal value of x_0 in $\mathbf{R}^{n+1}$ with coordinates $(x_0, x_1, \ldots, x_n)$. Remark that the notations will undergo naturally proper changes. Then $AS := e\_val(HS, Ord, NZ)$ gives us immediately a set of 3 asc-sets AS_s given below:

$$\begin{aligned} AS := \{&[-49 + 5 * x_3^2 + 46 * x_1 + 8 * x_0,\ x_1 - 2 * x_2 + 1,\ 5 * x_1^2 - 18 * x_1 + 17 - 4x_0], \\ &[x_3, -95 + 92 * x_2 + 8 * x_0, -49 + 46 * x_1 + 8 * x_0, 1481 - 1152 * x_0 + 64 * x_0^2], \\ &[177 + 25 * x_3^2,\ -7 + 5 * x_2, 5 * x_1 - 9, -1 + 5 * x_0]\}. \end{aligned}$$

Each of the 3 asc-sets AS_s in AS will furnish us a finite set K_s given by $K_1 = \emptyset$, $K_2 = \{9 \pm \frac{23}{8} * \sqrt{7}\}$, $K_3 = \emptyset$. Then the Finite Kernel Set $K = K\{x_0, GS, \mathbf{R}^{n+1}, \{h_0\}\}$ of the problem is the union of K_s:

$$K = K_1 \cup K_2 \cup K_3 = \{9 \pm \frac{23}{8} * \sqrt{7}\}.$$

We have then $Least(f) = Least(x_0) = Least(K) = 9 - \frac{23}{8} * \sqrt{7} \approx 1.393464981$ at point $(x_1, x_2) = (\frac{-1+\sqrt{7}}{2}, \frac{1+\sqrt{7}}{4})$.

Solving of Ex. 2.

In the same way as in $Ex.1.$ we get by e_val a set of 10 asc-sets AS_s, $s = 1, 2, \ldots, 10$, furnishing a finite set of real values K_s either $\{0\}$ or $\{1\}$. The *Finite Kernel Set* $K = K\{x_0, GS, \mathbf{R}^{n+1}, \{h_0\}\}$ is the union of these K_s and is equal to $\{0, 1\}$. Therefore the least value is $x_0 = 0$ at 4 points $(x_1, x_2) = (\pm 1, \pm 1)$. According to definition f in (2) is positive semidefinite.

References

[1] Wu Tianjiao, Some Test Problems on Applications of Wu's Method in Nonlinear Programming Problems, Chinese Quarterly Journal of Mathematics, Vol.9, No.2, 8-17, June, (1994).

[2] Wu Tianjiao, On the Decision of Definiteness of Polynomials, Proceedings of the 3rd ASCM, 281-283, Aug. 6-8, 1998. Lanzhou

[3] Wu, W.T., Mathematics Mechanization (Mathematical Geometry Theorem-Proving, Mechanical Geometry Problem-Solving and Polynomial Equations-Solving), Science Press/Kluwer Academic Problishers, 2000.

[4] Wang, D.K., Polynomial equations solving & mechanical geometry theorem proving, Doctoral Thesis, Institute of Systems Science, (1993).

A Sister Celine Type Algorithm for Definite Summation and Integration

Burkhard Zimmermann
Research Institute for Symbolic Computation
Johannes Kepler University, Linz, Austria
Burkhard.Zimmermann@risc.uni-linz.ac.at

Sister Celine's method [2] computes a recurrence for a sum or multisum $\sum_{k=0}^{\infty} f(n,k)$ where $f(n,k)$ is a given hypergeometric term. It was generalized to the q-hypergeometric case and to integrals [3].

The poster presents a generalization of Sister Celine's method to a wider class of special functions, which we call "level-2-functions". They are ∂-finite functions [1] satisfying certain technical conditions. Level-2-functions are closed under addition, multiplication, derivations, shifts and certain substitutions. They incude: polynomials, exponential and trigonometric functions, functions from mathematical physics such as Bessel functions, orthogonal polynomials; combinatorial sequences such as factorials, binomial coefficients, Fibonacci numbers, harmonic numbers, and functions from q-analysis.

Just as Sister Celine's method, our new algorithm makes an ansatz for a k-free annihilating operator for the summand $f(n,k)$ and subsequently solves a linear equation system for finding the unknown coefficients of that operator. Sister Celine's method works only if the shiftquotients $f(n+1,k)/f(n,k)$ and $f(n,k)/f(n,k+1)$ are rational functions in n and k; their rationality is the key for the reduction to a problem about polynomials in $Q(n)[k]$. Since our algorithm has to deal with nonhypergeometric summands as well, a different strategy for reducing to a polynomial problem is needed. The algorithm views the summand as an element of a suitably constructed ring of polynomials equipped with shift operators. The shifts are given by their action on the indeterminates of that polynomial ring.

A new theorem proves the existence of nontrivial annihilating operators and thus implies termination of the algorithm. It is a generalization of the fundamental theorem of hypergeometric summation [3].

The performance of the algorithm can be improved by the method of homomorphic images. A prototype implementation, the Mathematica package `mdf.m`, is available upon request from the author.

References

[1] F. Chyzak and B. Salvy. Non-commutative elimination in ore algebras proves multivariate identities. *J. Symbolic Comput.*, 26(2):187–227, 1998.

[2] Sister Mary Celine Fasenmyer. Some generalized hypergeometric polynomials. *Bull. Amer. Math. Soc.*, 53:806–812, 1947.

[3] H. Wilf and D. Zeilberger. An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities. *Invent. Math.*, 108:575–633, 1992.

[4] D. Zeilberger. A holonomic systems approach to special functions identities. *J. Comput. Appl. Math.*, 32:321–368, 1990.

Polynomial Transformations of Tschirnhaus, Bring and Jerrard

Victor S. Adamchik
Department of Computer Science,
Carnegie Mellon University, Pittsburgh, PA, USA
adamchik@cs.cmu.edu

David J. Jeffrey
Department of Applied Mathematics,
The University of Western Ontario, London, Ontario, Canada
djeffrey@uwo.ca

Abstract

Tschirnhaus gave transformations for the elimination of some of the intermediate terms in a polynomial. His transformations were developed further by Bring and Jerrard, and here we describe all these transformations in modern notation. We also discuss their possible utility for polynomial solving, particularly with respect to the Mathematica poster on the solution of the quintic.

1 Introduction

A recent issue of the BULLETIN contained a translation of the 1683 paper by Tschirnhaus [10], in which he proposed a method for solving a polynomial equation $P_n(x)$ of degree n by transforming it into a polynomial $Q_n(y)$ which has a simpler form (meaning that it has fewer terms). Specifically, he extended the idea (which he attributed to Decartes) in which a polynomial of degree n is *reduced* or *depressed* (lovely word!) by removing its term in degree $n-1$. Tschirnhaus's transformation is a polynomial substitution $y = T_k(x)$, in which the degree of the transformation $k < n$ can be selected. Tschirnhaus demonstrated the utility of his transformation by apparently solving the cubic equation in a way different from Cardano.

Although Tschirnhaus's work is described in modern books [9], later works by Bring [2, 3] and Jerrard [7, 6] have been largely forgotten. Here, we present the transformations in modern notation and make some comments on their utility. The investigations described here were stimulated by work on the Quintic poster [11] and therefore the discussion is directed towards the quintic. We shall deal with the following forms of a quintic:

$$x^5 + a_4\,x^4 + a_3\,x^3 + a_2\,x^2 + a_1\,x + a_0 = 0\,, \quad \text{General quintic form} \tag{1}$$

$$x^5 + b_3\,x^3 + b_2\,x^2 + b_1\,x + b_0 = 0\,, \quad \text{Reduced quintic form} \tag{2}$$

$$x^5 + c_2\,x^2 + c_1\,x + c_0 = 0\,, \quad \text{Principal quintic form} \tag{3}$$

$$x^5 + d_1\,x + d_0 = 0\,. \quad \text{Bring—Jerrard quintic form} \tag{4}$$

2 Tschirnhaus's solution of the cubic

Before considering the quintic, we look at the cubic. Tschirnhaus transformed the depressed cubic $x^3 + px + q = 0$ into the binomial $y^3 + r = 0$. This can be done efficiently using the resultant, which is a tool from after Tschirnhaus's

time, of course. Let $P_3(x)$ be the cubic and let $T_2(x,y)$ be Tschirnhaus's transform, then we have

$$\begin{aligned} P_3(x) &= x^3 + px + q = 0\,, & (5)\\ T_2(x,y) &= x^2 + \alpha x + \tfrac{2}{3}p + y = 0\,, & (6)\\ \mathrm{res}_x(P_3,T) &= y^3 + \left(p\alpha^2 - \tfrac{1}{3}p^2 + 3\alpha q\right) y + \tfrac{2}{3}\alpha^2 p^2 - \alpha^3 q + q^2 + \alpha pq + \tfrac{2}{27}p^3\,. & (7)\end{aligned}$$

Setting the coefficient of y in (7) to zero, we obtain a quadratic in α. Either root for α yields the equation

$$y^3 = 9q^3\alpha/p^2 + \tfrac{4}{3}\alpha pq - \tfrac{8}{27}p^3 - 2q^2\,,$$

and so 3 values for y are obtained. Tschirnhaus substituted the value(s) for y into (6) and solved for x. However, he failed to point out that this results in 6 values for x. The equation obeyed by these 6 values can be obtained by eliminating y between (7) and (6), and it is

$$(x^3 + px + q)(x^3 + 3\,\alpha\,x^2 + x[2p^2 - 9\alpha q]/p + \tfrac{4}{3}\,\alpha\,p + 9\,\alpha\,q^2/p^2 - 2\,q) = 0\,. \quad (8)$$

After each root has been computed, it must be tested to see which factor it satisfies.

3 General quintic to principal quintic

To apply Tschirnhaus's idea to the quintic, we use the formulae for the power-sums of the quintic roots. This method is more interesting than simply using the resultant again, and in addition is actually easier to work with. Denote the roots of (1) by $x_i,\ i = 1, ..., 5$, and let

$$S_n = S_n(x_k) = \sum_{k=1}^{5} x_k^n$$

be the sum of the nth powers. Following Newton [5], a general representation for S_n for $n \in \mathbb{N}$ is

$$S_n = -n\,a_{5-n} - \sum_{j=1}^{n-1} S_{n-j}\,a_{5-j}\,,$$

with $a_j = 0$ for $j < 0$. Special cases are $S_1 = -a_4$, $S_2 = a_4^2 - 2a_3$, $S_3 = -a_4^3 + 3a_3a_4 - 3a_2$,

$$\begin{aligned} S_4 &= a_4^4 - 4\,a_3\,a_4^2 + 4\,a_2\,a_4 + 2\,a_3^2 - 4\,a_1\,, & (9)\\ S_5 &= -a_4^5 + 5\,\left(a_3\,a_4^3 - a_2\,a_4^2 - a_3^2\,a_4 + a_1\,a_4 - a_0 + a_2\,a_3\right)\,. & (10)\end{aligned}$$

Supposing that the roots x_k of (1) are related to roots y_k of (3) by a quadratic Tschirnhaus transformation

$$y_k = {x_k}^2 + \alpha\,x_k + \beta \quad (11)$$

we wish to determine α and β subject to the requirement that they are expressed algebraically in terms of the coefficients. The power sums for (3) are $S_1(y_k) = S_2(y_k) = 0$,

$$S_3(y_k) = -3b_2\,, \qquad S_4(y_k) = -4b_1\,, \qquad S_5(y_k) = -5b_0\,. \quad (12)$$

Evaluating $S_1(y_k)$ and $S_2(y_k)$ using (11), we obtain equations for α and β. After some simplification,

$$\begin{aligned} \alpha\,a_4 - 5\,\beta + 2\,a_3 - a_4^2 &= 0\,, & (13)\\ \alpha^2\,a_3 - 10\,\beta^2 + \alpha\,(3\,a_2 - a_3\,a_4) + 2\,a_1 - 2\,a_2\,a_4 + a_3^2 &= 0\,. & (14)\end{aligned}$$

This system is quadratic with respect to α and β, and thus will produce two sets of coefficients. We are free to choose either of these. After finding α and β, we use the last three equations, those in (12), to obtain $b_0,\ b_1,\ b_2$.

As Tschirnhaus pointed out, transformation (11) can be applied to a general polynomial of degree n to remove terms in x^{n-1} and x^{n-2}.

4 Transformation to a Bring-Jerrard form

Removing, from a general quintic, the three terms in x^4, x^3 and x^2, brings it to Bring—Jerrard form. Tschirnhaus clearly thought that he would be able to do this by using the cubic transformation

$$z_k = {x_k}^3 + \alpha\, {x_k}^2 + \beta\, x_k + \gamma\,. \tag{15}$$

This section shows that it is not always possible to solve for the coefficients α, β, γ in terms of radicals, which are the quantities that Tschirnhaus would have been expecting to use.

To reduce the size of the equations, we shall start with a principal form cubic $x^5 + c_2\, x^2 + c_1\, x + c_0 = 0$ and eliminate the x^2 term. Extending the approach of the previous section in the obvious way, we use the power sums for the new quintic

$$\begin{aligned} S_1(z_k) &= S_2(z_k) = S_2(z_k) = 0, \\ S_4(z_k) &= -4d_1, \\ S_5(z_k) &= -5d_0. \end{aligned} \tag{16}$$

From the first three equations in (16) we determine α, β, γ. The remaining two equations will determine the new coefficients. Using (15) to evaluate S_1, S_2, S_3, we obtain three equations for the parameters:

$$5\,\gamma - 3\,c_2 = 0, \tag{17}$$

$$5\,\gamma^2 - 10\,\alpha\,c_0 - 4\,\alpha^2\,c_1 - 8\,\beta\,c_1 - 6\,\alpha\,\beta\,c_2 - 6\,\gamma\,c_2 + 3\,c_2^2 = 0, \tag{18}$$

$$\begin{aligned} 5\,\gamma^3 - 15\,\alpha^2\,\beta\,c_0 - 15\,\beta^2\,c_0 - 30\,\alpha\,\gamma\,c_0 - 12\,\alpha\,\beta^2\,c_1 - 12\,\alpha^2\,\gamma\,c_1 - 24\,\beta\,\gamma\,c_1 + 9\,c_0\,c_1 + \\ 12\,\alpha\,c_1^2 - 3\,\beta^3\,c_2 - 18\,\alpha\,\beta\,\gamma\,c_2 - 9\,\gamma^2\,c_2 + 24\,\alpha\,c_0\,c_2 + 21\,\alpha^2\,c_1\,c_2 + \\ 21\,\beta\,c_1\,c_2 + 3\,\alpha^3\,c_2^2 + 18\,\alpha\,\beta\,c_2^2 + 9\,\gamma\,c_2^2 - 3\,c_2^3 = 0\,. \end{aligned} \tag{19}$$

In general this system is not solvable by radicals. We can easily see this by eliminating β and γ. This will produce a sixth degree polynomial equation in α.

In 1796, Bring [2, 3] found a way around this problem. He considered a quartic transformation,

$$z_k = x_k^4 + \alpha\, x_k^3 + \beta\, x_k^2 + \gamma\, x_k + \delta, \tag{20}$$

that offers an extra parameter. Varying this parameter Bring was able to decrease the degrees of the equations for the rest of parameters. Substituting (20) into (16), we get a system of five equations with six unknown variables. From the equation

$$S_1(z_k) = 5\,\delta - 4\,c_1 - 3\,\alpha\,c_2 = 0\,,$$

we find $\delta = \frac{4}{5}c_1 + \frac{3}{5}c_2\alpha$. The second equation

$$\begin{aligned} S_2(z_k) = {} & -10\alpha\,\beta\,c_0 - 4\,\beta^2\,c_1 + \tfrac{4}{5}\,c_1^2 + 8\,c_0\,c_2 + \tfrac{46}{5}\,\alpha\,c_1\,c_2 \\ & + \tfrac{6}{5}\,\alpha^2\,c_2^2 + 6\,\beta\,c_2^2 - 2\,\gamma\,(5\,c_0 + 4\,\alpha\,c_1 + 3\,\beta\,c_2) = 0 \end{aligned} \tag{21}$$

relates β and γ. The trick is to choose a β that will make the coefficient of γ in (21) vanish. The value is

$$\beta = -\frac{5\,c_0}{3\,c_2} - \frac{4\,c_1}{3\,c_2}\,\alpha\,. \tag{22}$$

Therefore, equation (21) now depends only on α and is a quadratic.

$$\alpha^2(27c_2^4 - 160c_1^3 + 300c_0\,c_1\,c_2) + \alpha\,(27c_1\,c_2^3 - 400c_0\,c_1^2 + 375c_0^2\,c_2) + 18c_1^2\,c_2^2 - 45c_0\,c_2^3 - 250c_0^2\,c_1 = 0\,. \tag{23}$$

Finally, setting the sum of the cubes of (20) to zero, $S_3(z_k) = 0$, we obtain a cubic equation for γ. The equation will not be shown. Therefore all of the intermediate quantities can be found in terms of radicals.

This work was generalized by Jerrard [7, 6] (it is most likely that Jerrard was not aware of Bring's research) to show that the transformation (20) could be applied to a polynomial of degree n to remove the terms in x^{n-1}, x^{n-2} and x^{n-3}. In particular, Jerrard claimed that he developed a method for solving a general quintic. Hamilton was asked to report on this. In his detailed report, Hamilton showed that Jerrard had not completely solved the general quintic equation by radicals. The report is available at
`http://www.maths.tcd.ie/pub/HistMath/People/Hamilton/Jerrard/`.

5 Discussion

Starting with a general quintic (1), the transformation (20) produces the Bring–Jerrard form. In the case of a solvable quintic, the roots can be found using the 1771 formulae of Malfatti [8], who was the first to "solve" the quintic using a resolvent of sixth degree. In 1948, G. Watson [1] gave a public lecture in which he outlined step-by-step a procedure for solving a (solvable) quintic in radicals. In 1991, Dummit [4] developed exact formulas for the roots of solvable quintics.

The irreducible quintic can be solved in terms of Jacobi theta functions, as was first done by Hermite in 1858. Once we have the 5 solutions, we must invert the Tschirnhaus-Bring transformation to obtain the solutions to the original quintic. Thus we shall in general obtain 20 candidates for five solutions. There is no way of knowing which ones are correct without numerical testing. It is interesting to note that if one used Tschirnhaus's cubic transformation to solve a quintic (using something other than radicals), then one would obtain 15 solution candidates. By using a quartic transformation, Bring and Jerrard simplified the intermediate expressions at the price of now generating 20 solution candidates. Clearly, as symbolic solutions, any of these methods have their limitations, since we (or a computer system) can present only a list of solution candidates, rather than guaranteed solutions.

There is a Mathematica implementation of the Tschirnhaus-Bring transformation available from the Wolfram Research web cite at `http://library.wolfram.com/infocenter/TechNotes/158/`.
On this page you will find a set of notebooks that demonstrate various approaches to the solution of the quintic.

Acknowledgements. We would like to thank M. Trott for helpful discussions and comments. We would like to thank Luigi Pepe and Henrik Kragh Soerensen for help locating Malfatti's original paper. The first-named author was supported by grant CCR-0204003 from the National Science Foundation.

References

[1] B. Berndt, B. K. Spearman, K. S. Williams, Commentary on an unpublished lecture by G. N. Watson on solving the quintic, *Math. Intell*, **24** (2002), 15-33.

[2] E. S. Bring, *Meletemata quaedam mathematematica circa transformationem aequationum algebraicarum*, Lund, 1786.

[3] E. S. Bring, *Quart. J. Math.*, **6** (1864).

[4] D. S. Dummit, Solving solvable quintics, *Mathematics of Computation* , **57** (1991), 387–401. (corrigenda, *Math. Comp.*, **59** (1992), 309)

[5] J. A. Eidswick, A proof of Newton's power sum formula, *American Mathematical Monthly*, **75** (1968), 396–397.

[6] G. B. Jerrard, *An essay on the resolution of equations*, Taylor and Francis, 1859.

[7] G. B. Jerrard, On the possibility of solving equations of any degree however elevated, *Phil. Mag. Ser.*, 4, **3** (1852), 457–460.

[8] Gianfrancesco Malfatti, De aequationibus quadrato-cubicis dissertatio analitica. Atti dell'Accademia delle scienze di Siena, vol. IV (1771). pp. 129–185. (Contained in: Gianfrancesco Malfatti, Opere, volume I, Roma, Edizioni Cremonese, 1981, pp. 33–91.)

[9] Jean-Pierre Tignol, Galois' theory of algebraic equations, Longman, New York : Wiley, 1988.

[10] E. Tschirnhaus, A method for removing all intermediate terms from a given equation, THIS BULLETIN, Vol 37 (1), No. 143, (2003) 1–3. [Original: Methodus auferendi omnes terminos intermedios ex data equatione, *Acta Eruditorium*, **2** (1683), 204-207]

[11] Poster: *Solving the Quintic*, Wolfram Research, Inc. Champaign, IL, 1995. `http://library.wolfram.com/examples/quintic/`.

Announcement: Call for Presentations for Maple Summer Workshop 2004

July 11th — 14th, Wilfrid Laurier University

Waterloo, Ontario, Canada

The call for presentations for MSW 2004 is now open. Deadline for abstract submission is January 16th, 2004. The Maple Summer Workshop is an annual international meeting that brings together members of the Maple community to share techniques and experiences with the Maplesoft suite of products. The program combines guest speakers, interactive tutorials, contributed sessions, and Q&A with Maple developers and executives - all with a shared focus on innovative usage of Maple in education, research, and industry.

Guidelines for Abstract Submission

The program committee welcomes abstracts on:

Maple in education (all levels and disciplines); Maple, MapleNet, or Maple T.A. in distance education and/or Web-based learning; Applications of MapleNet or Maple T.A.; Maple in mathematics research; Applications of Maple in engineering and science.

Proposals must consist of a title, an abstract of $\frac{1}{2}$ – 1 page, a list of key words, and a specification of the intended audience. All submissions will be considered for one of two forums:

(1) Presentation during a formal contributed session (2) Poster session

Participants selected to present during the formal contributed session are invited to "posterize" their presentations, and participate in both forums. MSW registration fees will be waived for participants selected to present as part of the contributed sessions only. Other participants will be charged regular registration rate.

MSW Proceedings

All accepted abstracts will be included in the final proceedings. Submission of a final paper is optional, and all such papers will appear in the proceedings. Presenters opting to submit a final paper for inclusion in the final proceedings must do so by May 21st, 2004. The final proceedings will be published to a CD. Formatting guidelines for paper submission are available at `http://www.maplesoft.com/msw/abstracts.shtml`.

Submit proposals for presentations to `mswabstracts@maplesoft.com`, or by mail to: Maplesoft, C/O Robert J. Lopez, MSW Program Chair, 615 Kumpf Drive, Waterloo, ON , Canada N2V 1K8

Key Dates:

Deadline for abstract submission: Friday, January 16th, 2004

Notification of Acceptance: Friday, February 27th, 2004

Deadline for final paper submission: Friday, May 21st, 2004

For more details, visit `www.maplesoft.com/msw`.

ECCAD'2004 EAST COAST COMPUTER ALGEBRA DAY 2004
May 8, 2004
Wilfrid Laurier University
Waterloo ON, Canada
First announcement and call for posters and participation

ECCAD'2004 Invited Speakers:

- Prof. Jonathan M. Borwein, FRSC Dalhousie University, CANADA.
  Title: Mathematics by Experiment: Plausible Reasoning in the 21st Century.
- Prof. David A. Cox, Amherst College, USA.
  Title: Implicitization and Commutative Algebra.
- Prof. Daniel Lazard, Université Paris 6, FRANCE.
  Title: Solving zero-dimensional systems of equations and inequations, depending on parameters.

ECCAD'2004 conference e-mail: `eccad2004@wlu.ca`

ECCAD'2004 conference web site: Information about the conference can be found at:

`http://www.cargo.wlu.ca/eccad2004/`

ECCAD'2004 Organizer: Ilias S. Kotsireas, Wilfrid Laurier University, Waterloo, ON, Canada
ECCAD'2004 Local Arrangements:

- Edmond Lau, Wilfrid Laurier University, Waterloo, ON, Canada
- Lorna Schmalz, Wilfrid Laurier University, Waterloo, ON, Canada

ECCAD'2004 Advisory Committee:

- Christopher W. Brown, United States Naval Academy, Annapolis, MD, USA
- Bruce Char, Drexel University, Philadelphia, USA
- Robert M. Corless, University of Western Ontario, Ontario, Canada
- Shuhong Gao, Clemson University, Clemson, South Carolina, USA
- Keith O. Geddes, University of Waterloo, Waterloo, Ontario, Canada
- Mark W. Giesbrecht, University of Waterloo, Ontario, Canada
- Erich Kaltofen, North Carolina State University, Raleigh, NC, USA
- B. David Saunders, University of Delaware, Newark, DE, USA
- William Sit, City University of New York, New York, NY, USA
- Stephen M. Watt, University of Western Ontario, Ontario, Canada

ECCAD'2004 Sponsors: Wilfrid Laurier University Research Office, Wilfrid Laurier University Faculty of Science, Maplesoft, NSF.

ECCAD'2004 Poster Sessions: In keeping with tradition, there will be two poster sessions offering all participants an opportunity to present timely research in an informal environment. If you wish to submit a poster, please send a title and abstract to eccad2004@wlu.ca If you would like to set up a demo at the conference, please send space and other requirements to eccad2004@wlu.ca All submissions are due by April 15, 2002. Recent post-docs and graduate students are particularly encouraged to submit posters and arrange to set up a demo of their work. Poster abstracts may be up to 2 pages. Poster abstracts received before the submisison deadline will be inlcuded in the ECCAD'2004 book of abstracts which will be available at the conference.

NINTH RHINE WORKSHOP ON COMPUTER ALGEBRA

First Announcement and Call for Papers

March 25-26, 2004

University of Nijmegen, Nijmegen, The Netherlands

Topics.

The topics of the workshop include all aspects of Computer Algebra, from theory to applications and systems.

Purposes.

This will be the ninth edition of a workshop initiated in Strasbourg in 1988 and held every second year since. To avoid competition with well-established conferences in the field, the workshop is kept as informal as possible. Its two main purposes are to offer an opportunity for newcomers in the field to present their work and to be a forum for, and aimed at, (but not restricted to) European researchers.

Submissions.

Full papers or extended abstracts can be submitted to the Workshop Chair listed below. As submissions will appear in locally printed proceedings, intended for attendees only, there will be a light refereeing procedure. This also makes submission for formal publication elsewhere possible. The Program Committee, chaired by Prof. A. M. Cohen (Eindhoven), will be announced in full at a later stage.

Submission by E-mail (preferable):

send a LaTeX file and a postscript-file (for comparison) to: `bosma@math.kun.nl`

Submission by regular mail (not recommended):

send *two* copies to Wieb Bosma, Mathematisch Instituut, Universiteit Nijmegen, Postbus 9010, 6500 GL Nijmegen, the Netherlands.

Please state the name, address, and E-mail address for the author in full. Submissions of plans for the demonstration of software are also welcome.

Important Dates

January 10, 2004: Submitted papers must be received

January 31, 2004: Notification of acceptance

March 5, 2004: Final versions of papers (or extended abstracts) must be received

March 26-27, 2004: Workshop

Registration

The registration procedure will be announced in the second announcement.

Organization

Workshop Chair: Wieb Bosma, University of Nijmegen, `bosma@math.kun.nl`

Program Committee Chair: Arjeh Cohen, University of Eindhoven, `amc@win.tue.nl`

Information

Updated information will be available on the web page `http://www-math.sci.kun.nl/~bosma/RWCA04`