LNCS 6885

Vladimir P. Gerdt
Wolfram Koepf
Ernst W. Mayr
Evgenii V. Vorozhtsov (Eds.)

Computer Algebra in Scientific Computing

13th International Workshop, CASC 2011
Kassel, Germany, September 2011
Proceedings

Springer

Vladimir P. Gerdt   Wolfram Koepf
Ernst W. Mayr   Evgenii V. Vorozhtsov (Eds.)

Computer Algebra in Scientific Computing

13th International Workshop, CASC 2011
Kassel, Germany, September 5-9, 2011
Proceedings

Springer

Volume Editors

Vladimir P. Gerdt
Joint Institute for Nuclear Research (JINR)
Laboratory of Information Technologies (LIT)
141980 Dubna, Russia
E-mail: gerdt@jinr.ru

Wolfram Koepf
Universität Kassel, Institut für Mathematik
Heinrich-Plett-Straße 40, 34132 Kassel, Germany
E-mail: koepf@mathematik.uni-kassel.de

Ernst W. Mayr
Technische Universität München, Institut für Informatik
Lehrstuhl für Effiziente Algorithmen
Boltzmannstraße 3, 85748 Garching, Germany
E-mail: mayr@in.tum.de

Evgenii V. Vorozhtsov
Russian Academy of Sciences, Institute of Theoretical and Applied Mechanics
Novosibirsk, 630090, Russia
E-mail: vorozh@itam.nsc.ru

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-23567-2 e-ISBN 978-3-642-23568-9
DOI 10.1007/978-3-642-23568-9
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011934868

CR Subject Classification (1998): I.1, G.2, I.3.5, F.2, G.1, F.2.2

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

It is hard to imagine the present-day progress in computer algebra and its applications to scientific computing without the numerous contributions of the German specialists in this area. In particular, research in the area of computer algebra and its applications has been actively conducted at the Institute of Mathematics of the University of Kassel (www.mathematik.uni-kassel.de) during the past 20 years. Thus, the educational use of computer algebra systems has been promoted via two book series for a calculus course by Wolfram Koepf and for a course on "Mathematics for Engineers" by Walter Strampp in the 1990s. Research topics vary from power series and summation (Wolfram Koepf), cryptography (Hans-Georg Rück), PDEs and commutative algebra (Werner Seiler), to integrable systems and symmetry (Walter Strampp). Numerous PhD and habilitation theses in the areas of non-commutative polynomial algorithms, algebraic algorithms for q-special functions, Fourier series, PDEs, differential Galois theory, polycyclic groups, number fields, and Tamagawa numbers have been developed by the Computational Mathematics group.

As spokesperson of the German computer algebra working group Fachgruppe Computeralgebra (www.fachgruppe-computeralgebra.de), Wolfram Koepf has organized several conferences in Kassel to give young academics the opportunity to present their research. The *Computeralgebra Rundbrief*, the biannual magazine of the Fachgruppe, has been edited for the last decade by Markus Wessler from Kassel. And for some years now, Werner Seiler and Wolfram Koepf have been offering special computer algebra courses for gifted high school students. Therefore, it should not surprising at all that CASC 2011, the 13th Workshop on Computer Algebra in Scientific Computing, took place in Kassel.

The twelve earlier CASC conferences, CASC 1998, CASC 1999, CASC 2000, CASC 2001, CASC 2002, CASC 2003, CASC 2004, CASC 2005, CASC 2006, CASC 2007, CASC 2009, and CASC 2010 were held, respectively, in St. Petersburg (Russia), in Munich (Germany), in Samarkand (Uzbekistan), in Konstanz (Germany), in Yalta (Ukraine), in Passau (Germany), in St. Petersburg (Russia), in Kalamata (Greece), in Chişinău (Moldova), in Bonn (Germany), in Kobe (Japan), and in Tsakhkadzor (Armenia), and they all proved to be very successful.

This volume contains twenty six full papers submitted to the workshop by the participants and accepted by the Program Committee after a thorough reviewing process, and two extended abstracts of the invited talks.

One of the traditional topics of the CASC workshops, which is a cornerstone of symbolic algebraic computation, polynomial algebra, is represented by contributions devoted to the development of object-oriented computer algebra software for the modeling of algebraic structures as typed objects, the methods of deciding whether a multivariate polynomial is regular (i.e., not a zero divisor) modulo regular differential chains, generation of a new involutive division by antigraded monomial ordering, construction of irreducible polynomials over finite fields, divide-and-conquer algorithms for univariate polynomial arithmetic, and contributions to polynomial computations in non-commutative algebras.

Two papers deal with matrix algorithms: the knowledge-based automatic generation of partitioned matrix expressions and acceleration of the inversion of triangular Toeplitz matrices.

Several papers are devoted to the investigation with the aid of computer algebra of various topics related to the ordinary differential equations (ODEs): solution of boundary-value problems for ODEs with the aid of Maple, equilibrium and bifurcation analysis of the systems of autonomous ODEs, the determination of all Laurent-series solutions of a linear ODE system, investigation of a 2D system of ODEs by studying its polynomial ideals, finding a region of attraction to an equilibrium of a nonlinear ODE system.

One topic which is especially important for applications in scientific computing is the development of symbolic-numerical algorithms. This topic is represented by three papers. The first of them shows how one can apply a specialized symbolic-numeric cylindrical algebraic decomposition for computing the exact optimal value function. Another paper deals with the symbolic-numerical solution of the 2D Schrödinger equation to study the quantum tunneling problem for a coupled pair of ions. The third paper presents symbolic-numeric stability analysis of satellite dynamics under the influence of gravitational and aerodynamic forces.

Several papers deal with the application of symbolic computations in applied problems of physics, mechanics, social science, and engineering: the handling of complex nonlinear analog circuits with the aid of the *Mathematica* toolbox Analog Insydes 2011, the determination of the Hilbert-space metric rendering a given Hamiltonian self-adjoint in quantum mechanics, beams line design for solving the problems in the design of particle beam accelerators, modeling of convection in porous media in polar coordinates, the stability investigation of equilibrium position in the four-body problem of celestial mechanics, application of CAS GAP in quantum physics, the analysis of decision-making and coalition formation in social and political life using the CAS RelView, the application of CAS *Mathematica* for obtaining invariant manifolds of Lagrange systems, the use of quantifier elimination for studying some systems arising in the life sciences.

Our particular thanks are due to the CASC 2011 local Organizing Committee in Kassel, i.e., W.M. Seiler (University of Kassel), who has ably handled all the local arrangements in Kassel. Furthermore, we want to thank all the members of the Program Committee for their invaluable work. And last but not least we are extremely grateful to W. Meixner for his extensive help in the preparation of the camera-ready manuscript for this volume.

July 2011

V.P. Gerdt
W. Koepf
E.W. Mayr
E.V. Vorozhtsov

Organization

CASC 2011 was organized jointly by the Department of Informatics at the Technische Universität München, Germany, and the Institute for Mathematics at Kassel, Germany.

Workshop General Chairs

Vladimir P. Gerdt	JINR, Dubna
Ernst W. Mayr	TU, München

Program Committee Chairs

Wolfram Koepf (Kassel, Co-chair)
Evgenii V. Vorozhtsov (Novosibirsk, Co-chair)

Program Committee

Sergei Abramov (Moscow)
Alkis Akritas (Volos)
Hans-Joachim Bungartz (Munich)
Andreas Dolzmann (Saarbrücken)
Victor F. Edneral (Moscow)
Ioannis Z. Emiris (Athens)
Jaime Gutierrez (Santander)
Richard Liska (Prague)
Marc Moreno Maza (London, Ontario)
Alexander Prokopenya (Brest, BLR)
Eugenio Roanes-Lozano (Madrid)
Valery Romanovski (Maribor)
Markus Rosenkranz (Canterbury)
Mohab Safey El Din (Paris)
Yosuke Sato (Tokyo)
Werner M. Seiler (Kassel)
Doru Stefanescu (Bucharest)
Stephen M. Watt (W. Ontario, CAN)
Andreas Weber (Bonn)
Eva Zerz (Aachen)

Local Organization

Werner M. Seiler (Kassel)

Website

`http://wwwmayr.in.tum.de/CASC2011`

Table of Contents

A Recurrent Method for Constructing Irreducible Polynomials over Finite Fields

Sergey Abrahamyan[1] and Melsik Kyureghyan[2]

[1] Institute of Informatics and Automation Problems,
P. Sevak street 1, Yerevan 0014, Armenia
serj.abrahamyan@gmail.com

[2] Institute of Informatics and Automation Problems,
P. Sevak street 1, Yerevan 0014, Armenia
melsik@ipia.sci.am

Abstract. In this paper we consider irreducibility of the polynomial composition of the form $(x^p - x + \delta_2)^n P\left(\frac{x^p-x+\delta_1}{x^p-x+\delta_2}\right)$ over $\mathbb{F}_q$ under certain conditions. Furthermore, a computationally simple and explicit method of constructing recursive sequences of irreducible polynomials of degree np^k $(k = 1, 2, 3, \cdots)$ over $\mathbb{F}_q$ is given.

Keywords: finite field, recurrent method, polynomial composition.

1 Introduction

The subject of irreducible polynomials over a finite field along with numerous construction techniques has been of considerable interest in recent years. Such polynomials have important applications, e.g. in construction of devices for the arithmetic in finite fields and in applications to coding theory and cryptography [7,9]. It is of interest, both from theoretical and practical point of view, and has been the topic of an active line of research to construct sequences of irreducible polynomials of increasing degree over the base field $\mathbb{F}_q$. An effective method for generating classes of such polynomials over finite fields is considered to be the polynomial composition method. The problem of irreducibility of polynomial composition has been studied in literature by several authors including Varshamov [13], Cohen [1], Kyuregyan [2]-[5], Meyn [10], Wan [14], who have approached this problem from different aspects.

Let $\mathbb{F}_q$ be a finite field of order $q = p^s$, where p is a prime and s is a natural number; $x^p - x + \delta_1$ and $x^p - x + \delta_2$ be relatively prime polynomials over $\mathbb{F}_q$ and $P(x) = \sum_{i=0}^{n} c_i x^i$ is an irreducible polynomial of degree n over $\mathbb{F}_q$; $\mathbb{F}_{q^n}$ denotes a finite extension field over $\mathbb{F}_q$.

The question of interest in this work is to examine the irreducibility of the polynomial composition

$$F(x) = (x^p - x + \delta_2)^n P\left(\frac{x^p - x + \delta_1}{x^p - x + \delta_2}\right) = \sum_{i=0}^{n} c_i \left(x^p - x + \delta_1\right)^i \cdot \left(x^p - x + \delta_2\right)^{n-i}$$

over $\mathbb{F}_q$ under the condition of non-zero trace.

Based on this irreducibility result we have proposed a recurrent construction in section 4 that basically employs trace computing techniques. The proposed method allows some explicit and computationally easy construction of irreducible polynomials of increasing degree over base field $\mathbb{F}_q$ starting from an irreducible polynomial satisfying conditions in Theorem 3. Some auxiliary results on irreducibility of polynomials over finite fields associated with trace are considered in [15,16,11,12].

Theoretical computations show that the expense of constructing an irreducible polynomial of degree np^k from a given irreducible polynomial of degree np^{k-1} by the proposed method is $O\left(n^2p^{2k-1}\log np^k\right)$.

2 Preliminaries

In the present section, we consider some standard preliminary results on irreducibility of polynomials.

The following theorem due to Cohen [8] establishes the conditions under which the composition construction of polynomials $F(x) = g^n(x)P\big(f(x)/g(x)\big)$ is irreducible.

Theorem 1 (Theorem 3.7, [8]). *Let $f(x), g(x) \in \mathbb{F}_q[x]$ be relatively prime polynomials and let $P(x) \in \mathbb{F}_q[x]$ be an irreducible polynomial of degree n. Then the composition*

$$F(x) = g^n(x)P\big(f(x)/g(x)\big)$$

is irreducible over $\mathbb{F}_q$ if and only if $f(x) - \alpha g(x)$ is irreducible over $\mathbb{F}_{q^n}$ for some root $\alpha \in \mathbb{F}_{q^n}$ of $P(x)$.

Definition 1. *For $\alpha \in \mathbb{F}_{q^n}$ the trace $Tr_{\mathbb{F}_{q^n}/\mathbb{F}_q}(\alpha)$ of α over $\mathbb{F}_q$ is defined by*

$$Tr_{\mathbb{F}_{q^n}/\mathbb{F}_q}(\alpha) = \alpha + \alpha^q + \cdots \alpha^{q^{n-2}} + \alpha^{q^{n-1}}.$$

For convenience, we denote $Tr_{\mathbb{F}_{q^n}/\mathbb{F}_q} = Tr_{q^n/q}$.

Proposition 1 ([6], Theorem 3.78). *Let $\alpha \in \mathbb{F}_q$ and let p be the characteristic of $\mathbb{F}_q$. Then the trinomial $x^p - x - \alpha$ is irreducible in $\mathbb{F}_q[x]$ if and only if it has no root in $\mathbb{F}_q$.*

Proposition 2 ([6], Corollary 3.79). *With the notation of Proposition 1, the trinomial $x^p - x - \alpha$ is irreducible in $\mathbb{F}_q[x]$ if and only if $Tr_{\mathbb{F}_q}(\alpha) \neq 0$.*

With these preliminaries we state a theorem in the next section that yields irreducible polynomials of degree pn over $\mathbb{F}_q$.

3 Irreducibility of the Polynomial Composition

In this section, we shall obtain some result on the irreducibility of the composition of irreducible polynomials $F(x) = g^n(x)P\big(f(x)/g(x)\big)$ under condition of non-zero trace.

Theorem 2. *Let $P(x) = \sum_{i=0}^{n} c_i x^i$ be an irreducible polynomial of degree n over $\mathbb{F}_q$ and let $x^p - x + \delta_1$ and $x^p - x + \delta_2$ be relatively prime polynomials in $\mathbb{F}_q[x]$. Then*

$$F(x) = (x^p - x + \delta_2)^n P\left(\frac{x^p - x + \delta_1}{x^p - x + \delta_2}\right)$$

is an irreducible polynomial of degree pn over $\mathbb{F}_q$ if and only if

$$Tr_{q/p}\left((\delta_2 - \delta_1)\frac{P'(1)}{P(1)} - \delta_2 n\right) \neq 0.$$

Proof. Irreducibility of $P(x)$ over $\mathbb{F}_q$ implies that it can be represented over $\mathbb{F}_{q^n}$ as

$$P(x) = a\prod_{u=0}^{n-1}\left(x - \alpha^{q^u}\right), \quad a \in \mathbb{F}_q \tag{1}$$

where α is a root of $P(x)$. By substituting $\frac{x^p-x+\delta_1}{x^p-x+\delta_2}$ for x in (1) and multiplying its both sides by $(x^p - x + \delta_2)^n$ we obtain

$$\begin{aligned}
F(x) &= (x^p - x + \delta_2)^n P\left(\frac{x^p - x + \delta_1}{x^p - x + \delta_2}\right) = a\prod_{u=0}^{n-1}\Big((x^p - x + \delta_1) - \alpha^{q^u}(x^p - x + \delta_2)\Big) \\
&= a\prod_{u=0}^{n-1}\left(\left(1 - \alpha^{q^u}\right)x^p - \left(1 - \alpha^{q^u}\right)x + \left(\delta_1 - \alpha^{q^u}\delta_2\right)\right) = \\
&= a\prod_{u=0}^{n-1}\left(1 - \alpha^{q^u}\right)\left(x^p - x + \frac{\delta_1 - \alpha^{q^u}\delta_2}{1 - \alpha^{q^u}}\right) = \\
&= a(1-\alpha)^{\frac{q^n-1}{q-1}}\prod_{u=0}^{n-1}\left(x^p - x + \frac{\delta_1 - \alpha^{q^u}\delta_2}{1 - \alpha^{q^u}}\right).
\end{aligned}$$

By Theorem 1 $F(x)$ is irreducible over $\mathbb{F}_q$ if and only if the polynomial $(x^p - x + \delta_1) - \alpha(x^p - x + \delta_2)$ or equivalently the polynomial $x^p - x + \frac{\delta_1 - \alpha\delta_2}{1-\alpha}$ is irreducible over $\mathbb{F}_{q^n}$. On the other hand, the polynomial $x^p - x - \frac{\delta_1 - \alpha\delta_2}{\alpha - 1}$ is irreducible over $\mathbb{F}_{q^n}$ if and only if $Tr_{q^n/p}\left(\frac{\delta_1 - \alpha\delta_2}{\alpha - 1}\right) \neq 0$ by Proposition 2.

To calculate the trace we shall make use of the basic algebraic properties of trace. Then it can be written as follows:

$$\begin{aligned}
Tr_{q^n/p}\left(\frac{\delta_1 - \alpha\delta_2}{\alpha - 1}\right) &= Tr_{q/p}\left(Tr_{q^n/q}\left(\frac{\delta_1 - \alpha\delta_2}{\alpha - 1}\right)\right) \\
&= Tr_{q/p}\left(Tr_{q^n/q}\left(\frac{\delta_1}{\alpha - 1} - \frac{\alpha\delta_2 + \delta_2 - \delta_2}{\alpha - 1}\right)\right) \\
&= Tr_{q/p}\left((\delta_1 - \delta_2)\,Tr_{q^n/q}\left(\frac{1}{\alpha - 1}\right) - \delta_2 n\right).
\end{aligned} \tag{2}$$

Thus, to calculate the trace $Tr_{q^n/p}\left(\frac{\delta_1-\alpha\delta_2}{\alpha-1}\right)$ we first need to calculate the value of trace $Tr_{q^n/q}\left(\frac{1}{\alpha-1}\right)$. To do this we introduce the following notation:

$$P\left(x+1\right)=\sum_{i=0}^{n}c_i\left(x+1\right)^i=\sum_{i=0}^{n}d_ix^i=D(x).$$

Because α is a root of $P(x)$, $(\alpha-1)$ will be a root of $D(x)$, and, therefore, $\frac{1}{\alpha-1}$ will be a root of its reciprocal polynomial $D^*(x)$[1]. Hence

$$Tr_{q^n/q}\left(\frac{1}{\alpha-1}\right)=-\frac{d_1}{d_0}.$$

Compute the coefficients d_1 and d_0

$$d_0=D(0)=\sum_{i=0}^{n}c_i=P(1)$$

$$d_1=D'(0)=P'\left(x+1\right)\Big|_{x=0}=\sum_{i=1}^{n}i\;c_i=P'(1).$$

Consequently,

$$Tr_{q^n/q}\left(\frac{1}{\alpha-1}\right)=-\frac{P'(1)}{P(1)}.$$

Substitution of the latter formula in expression (2) gives

$$Tr_{q^n/p}\left(\frac{\delta_1-\alpha\delta_2}{\alpha-1}\right)=Tr_{q/p}\left(\left(\delta_2-\delta_1\right)\frac{P'(1)}{P(1)}-\delta_2n\right).$$

This yields that $F(x)$ is irreducible of degree np over $\mathbb{F}_q$ if and only if $Tr_{q/p}\left(\left(\delta_2-\delta_1\right)\frac{P'(1)}{P(1)}-\delta_2n\right)$ is non-zero. □

4 Recurrent Method

Using the results obtained above and those in section 2 we shall describe a method that allows recursive sequences of irreducible polynomials of degree np^k $(k=1,2,3,\cdots)$ over $\mathbb{F}_q$. The construction technique used to generate irreducible polynomials of degree np^k from a given irreducible polynomial of degree np^{k-1} over $\mathbb{F}_q$ is based on computations of trace.

Theorem 3. *Let* $P(x)=\sum_{i=0}^{n}c_ix^i$ *be an irreducible polynomial of degree* n *over* $\mathbb{F}_q$ *and let* $x^p-x+\delta_1$ *and* $x^p-x+\delta_2$ *be relatively prime polynomials in* $\mathbb{F}_p[x]$. *Define*

[1] Recall that for a polynomial $F(x)$ of degree n, its monic reciprocal is defined by $F^*(x)=\frac{1}{F(0)}x^nF\left(1/x\right)$.

$$F_0(x) = P(x)$$
$$F_k(x) = (x^p - x + \delta_2)^{n_{k-1}} F_{k-1}\left(\frac{x^p - x + \delta_1}{x^p - x + \delta_2}\right), \tag{3}$$

where $n_k = np^k$ denotes the degree of $F_k(x)$. Suppose

$$Tr_{q/p}\left((\delta_2 - \delta_1)\frac{P'(1)}{P(1)} - \delta_2 n\right) \neq 0 \quad Tr_{q/p}\left(\frac{(\delta_1 - \delta_2)\, P'\left(\frac{\delta_1}{\delta_2}\right) - \delta_2 n P\left(\frac{\delta_1}{\delta_2}\right)}{P\left(\frac{\delta_1}{\delta_2}\right)}\right) \neq 0. \tag{4}$$

Then $F_k(x)$ is an irreducible polynomial of degree np^k for every $k \geq 1$ over $\mathbb{F}_q$.

Proof. Our proof is by induction. For $k = 1$ we have that $F_1(x)$ is an np degree irreducible polynomial over $\mathbb{F}_q$ by the given condition (4) (see also Theorem 2). Consider the step $k = 2$. According to Theorem 2 $F_2(x)$ is irreducible if and only if

$$Tr_{q/p}\left((\delta_2 - \delta_1)\frac{F_1'(1)}{F_1(1)} - np\delta_2\right) = (\delta_2 - \delta_1)\, Tr_{q/p}\left(\frac{F_1'(1)}{F_1(1)}\right) \neq 0. \tag{5}$$

To show that $F_2(x)$ is irreducible it suffices to show that $Tr_{q/p}\left(\frac{F_1'(1)}{F_1(1)}\right) \neq 0$.
To this aim we compute the values of $F_1(x)$ and $F_1'(x)$ at point 1.
Since by (3)

$$F_1(x) = \sum_{i=0}^{n} c_i\,(x^p - x + \delta_1)^i\,(x^p - x + \delta_2)^{(n-i)}, \tag{6}$$

then

$$F_1(1) = \sum_{i=0}^{n} c_i\; \delta_1^i\; \delta_2^{(n-i)} = \delta_2^n P\left(\frac{\delta_1}{\delta_2}\right). \tag{7}$$

Grouping the terms in (6) and computing its first derivative we obtain

$$F_1'(x) =$$
$$= \left(c_n\,(x^p - x + \delta_1)^n + \sum_{i=1}^{n-1} c_i (x^p - x + \delta_1)^i (x^p - x + \delta_2)^{n-i} + c_0\,(x^p - x + \delta_2)^n\right)'$$
$$= -nc_n\,(x^p - x + \delta_1)^{n-1} - nc_0\,(x^p - x + \delta_2)^{n-1} +$$
$$+ \sum_{i=1}^{n-1} c_i \Big[-i\,(x^p - x + \delta_1)^{i-1}\,(x^p - x + \delta_2)^{n-i} -$$
$$- (n-i)\,(x^p - x + \delta_1)^i\,(x^p - x + \delta_2)^{n-i-1}\Big]$$

and removing the expression $(x^p - x + \delta_1)^{i-1}\,(x^p - x + \delta_2)^{n-1-i}$ out of the square brackets we have

$$F_1'(x) =$$
$$- nc_n \left(x^p - x + \delta_1\right)^{n-1} - nc_0 \left(x^p - x + \delta_2\right)^{n-1}$$
$$- \sum_{i=1}^{n-1} c_i \left[nx^p - nx + n\delta_1 - i(\delta_1 - \delta_2)\right] \left(x^p - x + \delta_1\right)^{i-1} \left(x^p - x + \delta_2\right)^{n-i-1}$$
$$(\textit{splitting the latter sum into two sums we get})$$
$$= (\delta_1 - \delta_2) \sum_{i=1}^{n-1} i\ c_i \left(x^p - x + \delta_1\right)^{i-1} \left(x^p - x + \delta_2\right)^{n-i-1} - nc_n \left(x^p - x + \delta_1\right)^{n-1}$$
$$- n \sum_{i=1}^{n-1} c_i \left(x^p - x + \delta_1\right)^{i} \left(x^p - x + \delta_2\right)^{n-i-1} - nc_0 \left(x^p - x + \delta_2\right)^{n-1}.$$

Next we compute the derivative of $F_1(x)$ at the point $x = 1$

$$F_1'(1) =$$
$$- nc_n\delta_1^{n-1} - nc_0\delta_2^{n-1} + (\delta_1 - \delta_2) \sum_{i=1}^{n-1} i\ c_i\ \delta_1^{i-1}\delta_2^{n-i-1} - n \sum_{i=1}^{n-1} c_i\ \delta_1^{i}\delta_2^{n-i-1}$$
$$= -nc_n\delta_1^{n-1} - nc_0\delta_2^{n-1} + (\delta_1 - \delta_2)\,\delta_2^{n-2} \sum_{i=1}^{n-1} i\ c_i \left(\frac{\delta_1}{\delta_2}\right)^{i-1} - n\delta_2^{n-1} \sum_{i=1}^{n-1} c_i \left(\frac{\delta_1}{\delta_2}\right)^{i}$$

since $\sum_{i=0}^{n} c_i \left(\frac{\delta_1}{\delta_2}\right)^i = \sum_{i=1}^{n-1} c_i \left(\frac{\delta_1}{\delta_2}\right)^i - c_n \left(\frac{\delta_1}{\delta_2}\right)^n - c_0$ and $\sum_{i=1}^{n} ic_i \left(\frac{\delta_1}{\delta_2}\right)^{i-1} = \sum_{i=1}^{n-1} ic_i \left(\frac{\delta_1}{\delta_2}\right)^i - nc_n \left(\frac{\delta_1}{\delta_2}\right)^{n-1}$ we get

$$F_1'(1) =$$
$$- nc_n\delta_1^{n-1} + (\delta_1 - \delta_2)\,\delta_2^{n-2} \sum_{i=1}^{n} i\ c_i \left(\frac{\delta_1}{\delta_2}\right)^{i-1} - nc_n (\delta_1 - \delta_2)\,\delta_2^{n-2} \left(\frac{\delta_1}{\delta_2}\right)^{n-1}$$
$$- n\delta_2^{n-1} \sum_{i=0}^{n} c_i \left(\frac{\delta_1}{\delta_2}\right)^{i} + n\delta_2^{n-1} c_n \left(\frac{\delta_1}{\delta_2}\right)^{n} = -nc_n\delta_1^{n-1} - nc_n \left(\frac{\delta_1}{\delta_2}\right)^{n}$$
$$+ nc_n\delta_1^{n-1} + nc_n \left(\frac{\delta_1}{\delta_2}\right)^{n} + (\delta_1 - \delta_2)\,\delta_2^{n-2} P' \left(\frac{\delta_1}{\delta_2}\right) - n\delta_2^{n-1} P \left(\frac{\delta_1}{\delta_2}\right)$$
$$= \delta_2^{n-2} \left((\delta_1 - \delta_2)\, P' \left(\frac{\delta_1}{\delta_2}\right) - n\delta_2 P \left(\frac{\delta_1}{\delta_2}\right) \right). \tag{8}$$

Substituting (7) and (8) in expression (5) we obtain

$$Tr_{q/p} \left(\frac{F_1'(1)}{F_1(1)}\right) = \frac{1}{\delta_2^2} Tr_{q/p} \left(\frac{(\delta_1 - \delta_2)\, P' \left(\frac{\delta_1}{\delta_2}\right) - n\delta_2 P \left(\frac{\delta_1}{\delta_2}\right)}{P \left(\frac{\delta_1}{\delta_2}\right)} \right)$$

which is not equal to 0 because of condition (4).

Now suppose that $F_k(x)$ is irreducible over $\mathbb{F}_q$ by induction. We prove that $F_{k+1}(x)$ is also irreducible over $\mathbb{F}_q$, or equivalently

$$Tr_{q/p}\left(\frac{F_k'(1)}{F_k(1)}\right) \neq 0. \tag{9}$$

From the assumption above on the irreducibility of $F_k(x)$ over $\mathbb{F}_q$ it follows that

$$Tr_{q/p}\left(\frac{F_{k-1}'(1)}{F_{k-1}(1)}\right) \neq 0. \tag{10}$$

We compute $F_k(1)$ and $F_k'(1)$ as in the case for $k = 2$.
Let $F_{k-1}(x) = \sum_{i=0}^{n_{k-1}} c_i^{(k-1)} x^i$. By (3)

$$F_k(x) = \sum_{i=0}^{n_{k-1}} c_i^{(k-1)} \left(x^p - x + \delta_1\right)^i \left(x^p - x + \delta_2\right)^{(n_{k-1}-i)} \tag{11}$$

and at the point $x = 1$

$$\begin{aligned} F_k(1) &= \sum_{i=0}^{n_{k-1}} c_i^{(k-1)} \delta_1^i \delta_2^{(n_{k-1}-i)} \\ &= \delta_2^{n_k-1} \sum_{i=0}^{n_{k-1}} c_i^{(k-1)} \left(\frac{\delta_1}{\delta_2}\right)^i = \delta_2^{n_k-1} F_{k-1}\left(\frac{\delta_1}{\delta_2}\right), \text{ for } k > 1. \end{aligned}$$

It is easy to see from (11) that

$$F_{k-1}\left(\frac{\delta_1}{\delta_2}\right) = F_{k-1}(1).$$

Hence

$$F_k(1) = \delta_2^{n_k-1} F_{k-1}(1), \quad k \geq 2 \tag{12}$$

Now we compute the first derivative of $F_k'(x)$

$$\begin{aligned} F_k'(x) = &\Bigg(c_{n_{k-1}}^{(k-1)} \left(x^p - x + \delta_1\right)^{n_{k-1}} + \\ &+ \sum_{i=1}^{n_{k-1}-1} c_i^{(k-1)} \left(x^p - x + \delta_1\right)^i \left(x^p - x + \delta_2\right)^{n_{k-1}-i} + c_0^{(k-1)} \left(x^p + \delta_1\right)^{n_{k-1}}\Bigg)' \\ &= \sum_{i=1}^{n_{k-1}-1} c_i^{(k-1)} \Big[-i\left(x^p - x + \delta_1\right)^{i-1} \left(x^p - x + \delta_2\right)^{n_{k-1}-i} - \\ &- \left(n_{k-1} - i\right)\left(x^p - x + \delta_2\right)^{n_{k-1}-i-1} \left(x^p - x + \delta_1\right)^i\Big] = \\ &= - \sum_{i=1}^{n_{k-1}-1} c_i^{(k-1)} \left[i\delta_2 - i\delta_1\right] \left(x^p - x + \delta_1\right)^{i-1} \left(x^p - x + \delta_2\right)^{n_{k-1}-1} \\ &= (\delta_1 - \delta_2) \sum_{i=1}^{n_{k-1}-1} i\, c_i^{(k-1)} \left(x^p - x + \delta_1\right)^{i-1} \left(x^p - x + \delta_2\right)^{n_{k-1}-i-1} \end{aligned}$$

and at the point $x = 1$

$$\begin{aligned}
F_k'(1) &= (\delta_1 - \delta_2) \sum_{i=1}^{n_{k-1}-1} i\, c_i^{(k-1)} \delta_1^{(i-1)} \delta_2^{(n_{k-1}-i)-1} \\
&= (\delta_1 - \delta_2)\, \delta_2^{n_k-1} \sum_{i=1}^{n_{k-1}-1} i\, c_i^{(k-1)} \left(\frac{\delta_1}{\delta_2}\right)^{i-1} \\
&= (\delta_1 - \delta_2)\, \delta_2^{n_{k-1}} F_{k-1}' \left(\frac{\delta_1}{\delta_2}\right), \ k > 2.
\end{aligned}$$

Obviously,

$$F_{k-1}' \left(\frac{\delta_1}{\delta_2}\right) = F_{k-1}'(1), \quad k > 2.$$

Hence

$$F_k'(1) = (\delta_1 - \delta_2)\, \delta_2^{n_k-1} F_{k-1}'(1), \quad k > 2. \tag{13}$$

Substituting (12) and (13) in (9) we get

$$Tr_{q/p}\left(\frac{F_k'(1)}{F_k(1)}\right) = Tr_{q/p}\left(\frac{(\delta_1 - \delta_2)\, \delta_2^{n_k-1} F_{k-1}'(1)}{\delta_2^{n_k-1} F_{k-1}(1)}\right) = (\delta_1 - \delta_2)\, Tr_{q/p}\left(\frac{F_{k-1}'(1)}{F_{k-1}(1)}\right)$$

which is non zero by (10). □

The complexity of the method we have described is evaluated in the terms of the number of necessary elementary operations. Theoretical computations show that the complexity of constructing an irreducible polynomial of degree np^k from a given irreducible polynomial of degree np^{k-1} with the above-given method is, thus, $O\left(n^2 p^{2k-1} \log np^k\right)$ elementary operations.

Acknowledgments. The authors wish to thank the anonymous referees for many helpful suggestions and corrections that allowed to improve the presentation of the paper.

This study was supported by the grant ('GRASP-10-05') of the National Academy of Sciences of RA, the National Foundation of Science and Advanced Technologies (RA) and Civilian Research and Development Foundation (US).

References

1. Cohen, S.: On irreducible polynomials of certain types in finite fields. Proc. Cambridge Philos. Soc. 66, 335–344 (1969)
2. Kyuregyan, M.: Recurrent methods for constructing irreducible polynomials over $\mathbb{F}_q$ of odd characteristics. Finite Fields Appl. 9, 39–58 (2003)
3. Kyuregyan, M.: Iterated constructions of irreducible polynomials over finite fields with linearly independent roots. Finite Fields Appl. 10, 323–431 (2004)
4. Kyuregyan, M.: Recurrent methods for constructing irreducible polynomials over $\mathbb{F}_q$ of odd characteristics II. Finite Fields Appl. 12, 357–378 (2006)
5. Kyuregyan, M., Kyuregyan, G.: Irreducible Compositions of Polynomials over Finite Fields, Design, Codes and Cryptography (2010), doi:10.1007/s10623-010-9478-5

6. Lidl, R., Niederreiter, H.: Finite Fields. Cambridge University Press, Cambridge (1987)
7. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. North-Holland, New York (1977)
8. Menezes, A., Blake, I.F., Gao, X., Mullin, R.C., Vanstone, S.A., Yaghoobian, T.: Applications of Finite Fields. Kluwer Academic Publishers, Boston (1993)
9. Menezes, A., Van Oorschoot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1997)
10. Meyn, H.: On the construction of irreducible self-reciprocal polynomials over finite fields. Appl. Algebrain Eng. Commun. Comput. 1, 43–53 (1990)
11. Moisio, M.: Kloosterman sums, elliptic curves, and irreducible polynomials with prescribed trace and norm, arXiv:0706.2112v5 [math.NT] (November 21, 2007); Comment: 21 pages; revised version with somewhat more clearer proofs; to appear in Acta Arithmetica
12. Ree, R.: Proof of a conjecture of S. Chowla. J. Number Theory 3, 210–212 (1971)
13. Varshamov, R.: A general method of synthesizing irreducible polynomials over Galois fields. Soviet Math. Dokl. 29, 334–336 (1984)
14. Wan, D.: Generators and irreducible polynomials over finite fields. Math. Comp. 219, 1195–1212 (1997)
15. Yucas, J.L.: Irreducible polynomials over finite fields with prescribed trace/prescribed constant term. Finite Fields and Their Applications 12, 211–221 (2006)
16. Yucas, J.L.: Generalized reciprocals, factors of Dickson polynomials and generalized cyclotomic polynomials over finite fields. Finite Fields and Their Applications 13(3), 492–515 (2007)

Higher-Order Linear Differential Systems with Truncated Coefficients

S.A. Abramov[1,⋆], M.A. Barkatou[2], and E. Pflügel[3]

[1] Computing Centre of the Russian Academy of Sciences, Vavilova, 40, Moscow 119333, Russia
sergeyabramov@mail.ru

[2] Institut XLIM, Département Mathématiques et Informatique, Université de Limoges ; CNRS, 123, Av. A. Thomas, 87060 Limoges cedex, France
moulay.barkatou@unilim.fr

[3] Faculty of CISM, Kingston University, Penrhyn Road, Kingston upon Thames, Surrey KT1 2EE, United Kingdom
E.Pfluegel@kingston.ac.uk

Abstract. We consider the following problem: given a linear differential system with formal Laurent series coefficients, we want to decide whether the system has non-zero Laurent series solutions, and find all such solutions if they exist. Let us also assume we need only a given positive integer number l of initial terms of these series solutions. How many initial terms of the coefficients of the original system should we use to construct what we need?

Supposing that the series coefficients of the original systems are represented algorithmically, we show that these questions are undecidable in general. However, they are decidable in the scalar case and in the case when we know in advance that a given system has an invertible leading matrix. We use our results in order to improve some functionality of the Maple [17] package ISOLDE [11].

1 Introduction

Linear differential systems with variable (e.g. power series) matrix coefficients appear in many areas of mathematics. Laurent series solutions of such systems form a building block for other types of solutions, and more generally, algorithms for finding Laurent series solutions may be a part of various computer algebra algorithms (see e.g. [1,6]).

Let a linear differential system with formal Laurent series coefficients be given. First of all, we want to decide whether or not the given system has non-zero Laurent series solutions. Suppose that such solutions exist, and we need only a given positive integer number l of initial terms of each of them. Then the following question arises: how many initial terms of the coefficients of the original system should we use to find what we need? Is it possible to compute at least

⋆ Supported by RFBR grant 10-01-00249-a.

an upper bound for the number of such terms? If we have such bound, then we can truncate the series involved into the original system before finding the truncated solutions. Such truncation leads to a system whose matrix coefficients are polynomials. In many cases it is much easier to work with such kind of systems than with systems with series matrix coefficients.

Let k be a field of characteristic 0. We denote by $k[[x]]$ the ring of formal power series with coefficients in k and $k((x)) = k[[x]][x^{-1}]$ its quotient field. If $i \in \mathbb{Z}, u \in k((x))$ then the notation $[x^i]u$ is used for the coefficient of x^i in u. For a nonzero element $u = \sum u_i x^i$ of $k((x))$ we denote by $\operatorname{val}_x u$ the *$x-$adic valuation* of u defined by $\operatorname{val}_x u = \min\{i \text{ such that } u_i \neq 0\}$. By convention $\operatorname{val}_x 0 = \infty$. For $M(x) \in \operatorname{Mat}_m(k((x)))$ we define $\operatorname{val}_x M(x)$ as the minimum of the valuations of the entries of $M(x)$.

We shall write θ for $x\frac{d}{dx}$ and consider differential systems of the form

$$A_r(x)\theta^r y + A_{r-1}(x)\theta^{r-1}y + \cdots + A_0(x)y = 0 \tag{1}$$

where $y = (y_1, y_2, \ldots, y_m)^T$ is a column vector of unknown functions of x. For the coefficient matrices

$$A_0(x), A_1(x), \ldots, A_r(x) \tag{2}$$

we have $A_i(x) \in \operatorname{Mat}_m(k[[x]])$, $A_0(x), A_r(x)$ are non-zero and $\min_i\{\operatorname{val}_x(A_i)\} = 0$.

Let a system S be of the form (1) and define the *l-truncation* $S^{\langle l\rangle}$ which is obtained by omitting all the terms of degree larger than or equal to l in the coefficients of S.

In this paper, we are concerned with two problems:

Problem 1 (Existence Problem). Given a system S of the form (1), decide whether or not this system has a solution in $k((x))^m \setminus \{0\}$.

Problem 2 (Truncation Problem). Given a system S of the form (1),

1. Decide whether or not there exists a non-negative integer sequence $(a_l)_{1\le l<\infty}$ such that for any $e \in \mathbb{Z}$, $l \in \mathbb{Z}^+$ and column vectors $c_e, c_{e+1}, \ldots, c_{e+l-1} \in k^m$, the system S possesses a solution $y(x) \in k((x))^m$ of the form

$$y(x) = c_e x^e + c_{e+1}x^{e+1} + \ldots + c_{e+l-1}x^{e+l-1} + O(x^{e+l}),$$

iff the system $S^{\langle a_l\rangle}$ possesses a solution $\tilde{y}(x) \in k((x))^m$ such that

$$\tilde{y}(x) - y(x) = O(x^{e+l}).$$

2. If such sequences (a_l) exist, then find at least one of them.

We suppose that the entries of the matrices (2) are represented *algorithmically*: for any entry $u(x)$ an algorithm Ξ_u (a procedure, terminating in finitely many steps) such that $u(x) = \sum_{i=0}^{\infty} \Xi_u(i)x^i$ is given. This is factually a model of computation. Our results can be represented without using this model (skipping the undecidability questions then), see Remark 1.

We will show that the existence and truncation problems are algorithmically undecidable in the general case (Section 3.2). However, both problems can be solved algorithmically for scalar equations (Section 2), and for first order systems (Section 3.2.1). The problems are also solvable for systems of the form (1) with a leading matrix invertible in $\mathrm{Mat}_m(k((x)))$ (Section 3.2). Note that we are not able to check algorithmically whether or not a given matrix is invertible. However, if we know *in advance* that the matrix $A_r(x)$ in a given system of the form (1) is invertible and the matrix $A_0(x)$ is non-zero then our algorithm completely solves the existence and truncation problems.

The output of our algorithm for the scalar case can be represented as an integer $d \geq -1$ such that

– a solution of S in $k((x)) \setminus \{0\}$ exists iff $d \geq 0$,
– if $d \geq 0$ then a solution of the truncation problem for S is represented by the sequence

$$a_l = \max\{d,\, l\} \quad (l = 1, 2, \ldots). \tag{3}$$

It follows from (3) that if in the scalar case the truncation problem has a solution then a sequence (a_l) can be taken such that $a_l = l$ for all l large enough. In the case of system the situation can be more complicated. However, when systems have invertible leading matrices, we can organise our algorithm in such a way that the output is again an integer $d \geq -1$ such that

– a solution of S in $k((x))^m \setminus \{0\}$ exists iff $d \geq 0$,
– if $d \geq 0$ then a solution of the truncation problem for S is represented by the sequence

$$a_l = d + l \quad (l = 1, 2, \ldots). \tag{4}$$

A sequence (a_l) giving a slightly more accurate bound is proposed as well.

At the end of Section 3, we mention another problem that can also be solved using the results of this paper.

To our knowledge, finding sequences (a_l) for the types of systems considered in this paper does not seem to have been done elsewhere in the literature.

2 The Case of Scalar Equations

If $m = 1$ in the system (1), then this system is a scalar equation. In this particular case, both the existence and truncation problem have an algorithmic solution. The crucial point is that for any such equation we can determine algorithmically the *indicial polynomial* [12, Ch. IV, § 8] $I_S(\lambda) \in k[\lambda] \setminus \{0\}$ such that

(a) if $y(x) \in k((x))^m$ is a nonzero solution of S then $I_S(\mathrm{val}_x\, y(x)) = 0$, and
(b) if e^* is the maximal integer root of $I_S(\lambda)$ then S has a solution $y(x) \in k((x))^m$ of valuation e^*.

The indicial polynomial can be constructed algorithmically, and computer algebra methods for computing its integer roots exist. Supposing in the scalar case that $\min_i \operatorname{val}_x A_i(x) = 0$ it is sufficient to know only the coefficients $[x^0]A_0(x)$, $[x^0]A_1(x)$, ... , $[x^0]A_r(x)$ for this construction.

Proposition 1. *Let $m = 1$ in S of the form (1). Let $I_S(\lambda)$ be the indicial polynomial of the scalar equation S. In this case*

(i) If $I_S(\lambda)$ has no integer root, then S has no solution in $k((x)) \setminus \{0\}$.

(ii) Otherwise a solution of the truncation problem is given by the sequence

$$a_l = \max\{e^* - e_* + 1,\, l\} \quad (l = 1, 2, \dots), \tag{5}$$

where e_*, e^* are the minimal and maximal integer roots of $I_S(\lambda)$, respectively.

Proof. (i) Follows from the property (a) of the indicial equation.

(ii) For an arbitrary solution $y(x) \in k((x)) \setminus \{0\}$ we have $\operatorname{val}_x y(x) = e$, where e is an integer root of $I_S(x)$. Let a solution have the form $c_e x^e + c_{e+1}x^{e+1} + \dots$ Then for any $l > 0$ the coefficients $c_e, c_{e+1}, \dots, c_{e+l}$ satisfy a relation $a_{l,0}c_e + a_{l,1}c_{e+1} + \dots + a_{l,l}c_{e+l} = 0$ where $a_{l,l} = I_S(e+l)$ and the constants $a_{l,0}, a_{l,1}, \dots, a_{l,l}$ can be computed from

$$[x^i]A_j(x) \;\; (i = 0, 1, \dots, l, \;\; j = 0, 1, \dots, r). \tag{6}$$

Thus, if $l > e^* - e$ then c_{e+l} is defined uniquely by $c_e, c_{e+1}, \dots, c_{e+l-1}$, and the values (6) define all the values $[e]y(x), [e+1]y(x), \dots, [e+l]y(x)$ for all belonging to $k((x))$ solutions of S (the Frobenius method [12, Ch. IV, § 8]). Observe that the values (6) coincide for S and $S^{\langle l \rangle}$. The claim follows. □

The output of the algorithm can be represented in the form of an integer $d \geq -1$ as was explained in the end of Section 1: if $I_S(\lambda)$ has no integer root then we set $d = -1$; and $d = e^* - e_* + 1$ otherwise.

Note that this algorithm needs only $[x^0]A_0(x)$, $[x^0]A_1(x)$, ... , $[x^0]A_r(x)$ for computing this d.

Example 1. *Let $k = \mathbb{Q}$, and S be the scalar equation*

$$(1 - x)\theta^2 y + (-2 + 4x)\theta y + (-x + 2x^2 + 2x^3 + 2x^4 + \dots)\, y = 0,$$

which has solutions $f(x) = 1 - x$, and $g(x) = x^2 - x^3$. The indicial polynomial is $\lambda^2 - 2\lambda$, its roots are 0 and 2, thus $d = e^* - e_* + 1 = 3$, $a_1 = a_2 = a_3 = 3$. For finding three terms of each of series solutions we construct $S^{\langle 3 \rangle}$

$$(1 - x)\theta^2 y + (-2 + 4x)\theta y + (-x + 2x^2)y = 0.$$

The latter equation has two independent solutions:

$$-1 + x + 0 \cdot x^2 - x^3 - \frac{1}{4}x^4 + O(x^5), \quad -x^2 + x^3 + 0 \cdot x^4 + O(x^5).$$

We get three wanted terms of solutions of S:

$$-1 + x + 0 \cdot x^2, \quad -x^2 + x^3 + 0 \cdot x^4.$$

These polynomials coincide with exact solutions of S. Note that we cannot take $S^{\langle 2 \rangle}$ for computing two first terms of each of solutions of S, since $S^{\langle 2 \rangle}$ is

$$(1 - x)\theta^2 y + (-2 + 4x)\theta y - xy = 0,$$

and the space of Laurent solutions of this equation has dimension 1. We get the solution $-x^2 + x^3 + \frac{1}{4}x^4 + O(x^5)$ which gives us two first terms of $g(x)$, but we do not obtain the corresponding truncation of $f(x)$. This confirms that the sequence (3) is a correct solution of the truncation problem, while the sequence $a_l = l$, $l = 1, 2, \ldots$, is not in general.

Concerning finding integer roots of polynomials over k we can remark the following. If $k = \mathbb{Q}$ then the algorithm is well known. It is also known that if k_0 is a field of characteristic 0 such that an algorithm for finding integer roots of a polynomials over k_0 is given then one can find integer roots of polynomials over any simple extension (algebraic or transcendental) of k_0. This can be used recursively.

3 The System Case

As we mentioned in Section 1 we will show that the existence and truncation problems are algorithmically undecidable in the general system case. We will first introduce some auxiliary facts and notions.

3.1 Undecidability in the General Case

Many algorithmic problems related to systems are undecidable. To prove this for some of such problems we will use the undecidability of one specific known problem.

We will call a *signal* any algorithm Ω computing for each $i = 0, 1, \ldots$ a value $\Omega(i)$ belonging to $\{0, 1\}$ and such that

- $\Omega(0) = 1$,
- if $\Omega(i) = 0$ for some $i \geq 1$ then $\Omega(i + 1) = 0$.

A signal Ω is *infinite* if $\Omega(i) = 1$ for all $i \geq 0$ and *finite* otherwise.

We will use the following fact:

The problem of recognizing whether or not a given signal is finite is undecidable.

This is a consequence of classical Turing's result on undecidability of the problem of terminating of an algorithm [16].

Using this fact we can prove that the problem of recognizing whether or not a given non-zero square matrix with entries in $k((x))$ is invertible over the field $k((x))$. Indeed, let $k = \mathbb{Q}$, and Ω be an arbitrary signal. The matrix

$$\begin{pmatrix} 1-x & 1 \\ 1 & \sum_{i=0}^{\infty} \Omega(i)x^i \end{pmatrix} \tag{7}$$

is invertible iff the signal Ω is finite.

If Ω is a signal and $s(x) = s_0 + s_1x + s_2x^2 + \ldots \in k[[x]]$ then we set $\Omega * s(x) = \Omega(0)s_0 + \Omega(1)s_1x + \Omega(2)s_2x^2 + \ldots$; given a system S of the form (1), we can replace any entry $s(x)$ of the matrix coefficients of S by $\Omega * s(x)$, the obtained new system will be denoted $\Omega * S$.

Proposition 2. *(i) The algorithmic problem of recognizing whether or not a given system of the form (1) has a solution in $k((x))^m \setminus \{0\}$ is undecidable.*

(ii) There exist systems of the form (1) for which the truncation problem has no solution (no sequence (a_i) exists); the algorithmic problem of recognizing whether or not the truncation problem for a given system of the form (1) has a solution (a sequence (a_i)) is undecidable.

Proof. (i) Let S be the system

$$A(x)x\theta y + A(x)y = 0 \tag{8}$$

where $A(x)$ is the matrix (7). Then the system $\Omega * S$ has a solution in $k((x))^2 \setminus \{0\}$ iff the signal Ω is infinite. Indeed, if the matrix (7) is invertible then the system is equivalent to $\begin{pmatrix} x & 0 \\ 0 & x \end{pmatrix} \theta y + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} y = 0$ which has no non-zero solution in $k((x))^2$. But if (7) is not invertible then the system has solutions in $k((x))^m \setminus \{0\}$ since the system $\begin{pmatrix} x & 0 \\ 0 & x \end{pmatrix} \theta y + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} y = \begin{pmatrix} s_1 \\ s_2 \end{pmatrix}$ with $s_1(x), s_2(x) \in k((x))$ has a solution in $k((x))^2 \setminus \{0\}$ if at least one of the series $s_1(x), s_2(x)$ is non-zero.

(ii) It follows from the proof of (i) that no sequence (a_i) exists for the system S of the form (8). However, if Ω is a finite signal then any sequence (a_l) can be used for the system $\Omega * S$ because neither $\Omega * S$ nor its truncations have solutions in $k((x)) \setminus \{0\}$. So for the system $\Omega * S$ a sequence (a_l) exists iff Ω is finite. □

Remark 1. *As it was mentioned in the Introduction, our results can be formulated without a special supposition on the form of series representation. It follows from the proof of Proposition 2 that for some systems of the form (1) no sequence (a_l) exists. In Section 3.2 we will show that if the leading matrix is invertible in* $\mathrm{Mat}_m(k((x)))$*, then after a finite number of steps we can recognize whether or not this system has a solution in $k((x))^m \setminus \{0\}$. If the answer is affirmative then there is a guarantee that a sequence (a_l) exists and can be presented.*

3.2 Some Particular Decidable Cases

We will now show that in some particular cases the existence and truncation problems are decidable.

3.2.1 First Order Systems

In this section, we consider first order systems of the form

$$\theta y = A(x)y, \tag{9}$$

where $A(x) \in \mathrm{Mat}_m(k((x)))$. We will show that the existence and truncations problems are decidable for systems of this form. Our approach is based on the concept of *simple systems* [3] (which is related to the notion of super-irreducible forms of linear differential systems [13]).

A system $\theta y = A(x)y$ can always be rewritten as a system of the form

$$D(x)\theta y = N(x)y \tag{10}$$

where $D(x), N(x) \in \mathrm{Mat}_m(k[[x]])$ with $\min\{\mathrm{val}_x D(x), \mathrm{val}_x N(x)\} = 0$.

For this take $N(x) = D(x)A(x)$ and $D(x) = \mathrm{diag}(x^{\alpha_1}, x^{\alpha_2}, \ldots, x^{\alpha_m})$ where $\alpha_i = \max\{0, -\mathrm{val}_x A_{i.}(x)\}$, here $A_{i.}(x)$ denotes the ith row of the matrix $A(x)$.

Now consider a general system S of the form (10) with coefficients $D(x) = \sum_{i=0}^{\infty} D_i x^i$, $N(x) = \sum_0^{\infty} N_i x^i$. With S we associate the following polynomial in λ

$$I_S(\lambda) = \det(D_0\lambda - N_0). \tag{11}$$

If this polynomial $I_S(\lambda)$ is non-zero then we shall say that the system S is *simple* and refer to the polynomial $I_S(\lambda)$ as the *indicial polynomial* of S as in [7, Definition 2.1]. By extension, a system of the form (9) will be called simple if the corresponding system (10) is simple.

The following result shows that for simple systems, the existence and truncation problems can be solved very similarly as in the scalar case.

Proposition 3. *Given a simple system of the form (10) with indicial polynomial $I_S(\lambda)$ given by (11), we have*

(i) If $I_S(\lambda)$ has no integer root, then S has no solution in $k((x)) \setminus \{0\}$.

(ii) Otherwise a solution of the truncation problem is given by the sequence

$$a_l = \max\{e^* - e_* + 1,\, l\} \quad (l = 1, 2, \ldots), \tag{12}$$

where e_* and e^* are the minimal and maximal integer roots of $I_S(\lambda)$, respectively.

Proof. (i) The statements of this proposition follow from the results in [7]. The algorithm presented therein computes regular formal solutions (i.e. in particular, Laurent series solutions) by finding successive terms. The equation that determines a coefficient of the monomial x^j $(j \geq 0)$, part of a solution $y = x^e \sum y_j x^j$ $(y_0 \neq 0)$, is

$$((e+j)D_0 - N_0)\, y_j = -b_j \tag{13}$$

where $b_0 = 0$ and

$$b_j = ((e+j-1)D_1 - N_1)\, y_{j-1} + \cdots + (eD_j - N_j)y_0. \quad (j \geq 1)$$

Putting $j = 0$, we see why the nonzero polynomial $I_S(\lambda) = \det(\lambda D_0 - N_0)$ plays a similar role as the indicial polynomial in the scalar case (see Section 2): We have $I_S(\mathrm{val}_x y) = 0$ so the existence of integer roots of I_S is a necessary condition for the existence of solutions in $k((x))^m \setminus \{0\}$.

(ii) Let $x^e \sum y_j x^j$ be a solution of $D^{\langle a_l \rangle}(x)\theta y = N^{\langle a_l \rangle}(x)$ where $a_l = \max\{e^* - e_* + 1,\, l\}$. This means that equations (13) are satisfied for $j = 0, 1, \ldots, l$. In order to extend this to a solution of the untruncated system, we need to compute the coefficient y_j for $j > l$, we note that the matrix $(e+j)D_0 - N_0$ is invertible since $e + j > e + l \geq e^*$. This means that we can determine y_j uniquely. □

Example 2. *Let $k = \mathbb{Q}$ and S be the first-order system*

$$\theta y = \begin{pmatrix} 0 & x^3 \\ 0 & -3 \end{pmatrix} y$$

which has as a basis of solutions

$$y_1(x) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad y_2(x) = \begin{pmatrix} \ln(x) \\ x^{-3} \end{pmatrix}.$$

Hence the space of solution of S in $k((x))^2$ has dimension 1. This system is simple: it is already in the form (10) with $D(x) = I_2$ and $N(x) = \begin{pmatrix} 0 & x^3 \\ 0 & -3 \end{pmatrix}$. Its indicial polynomial is

$$I_S(\lambda) = \begin{vmatrix} \lambda & 0 \\ 0 & \lambda + 3 \end{vmatrix} = \lambda(\lambda + 3).$$

The roots of $I_S(\lambda)$ are -3 and 0, thus $d = e^* - e_* + 1 = 4$, $a_1 = a_2 = a_3 = 4$. For $l = 1, 2, 3$ the l-truncation of S is

$$\theta y = \begin{pmatrix} 0 & 0 \\ 0 & -3 \end{pmatrix} y.$$

The latter system has two independent solutions in $k((x))^2$:

$$\tilde{y}_1(x) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \tilde{y}_2(x) = \begin{pmatrix} 0 \\ x^{-3} \end{pmatrix}.$$

This confirms that the sequence (12) is a correct solution of the truncation problem, while the sequence $a_l = l$, $l = 1, 2, \ldots$, is not in general.

Remark 2. *The results in Proposition 3 are valid for more general class of systems, namely simple systems of higher order [4], [5]. Recall that a system S of the form (1) is simple if the matrix polynomial defined by*

$$L_S(\lambda) = A_r(0)\,\lambda^r + A_{r-1}(0)\,\lambda^{r-1} + \cdots + A_0(0),$$

is regular, i.e., $\det(L_S(\lambda)) \not\equiv 0$. For a simple system S we define its indicial polynomial as $I_S(\lambda) = \det(L_S(\lambda))$.

Having clarified the situation if a system is simple, we now solve Problem 1 and Problem 2 for the case of a general first order system of the form (9). Define the *span* of an invertible matrix $T(x) \in \mathrm{Mat}_m(k((x)))$ by

$$\sigma(T(x)) = -\mathrm{val}_x\, T(x) - \mathrm{val}_x\, T^{-1}(x).$$

We need the following well-known technical lemma, which has been stated in [14], see also its use in [15].

Lemma 1. *([14]) Let $A(x), T(x) \in \mathrm{Mat}_m(k((x)))$ with $T(x)$ invertible and let $\tilde{A}(x) = T^{-1}(x)(A(x)T(x) - \theta(T(x)))$. Then the coefficient $\tilde{A}_j$ depends only on the A_i with $i \le j + \sigma(T(x))$.*

Proposition 4. *Consider a system $\theta y = A(x)y$ of the form (9), and let $q = \max\{0, -\mathrm{val}_x\, A(x)\}$.*

(i) There exists an algorithm, using only the first mq terms of the entries of $x^q A(x)$, that computes an invertible matrix $T(x) \in \mathrm{Mat}_m(k[x])$ with $\det T(x) = cx^\nu$ for some nonzero constant $c \in k$ and some nonnegative integer ν, with span $\sigma(T(x)) \le (m-1)\,q$ such that the substitution $y = T(x)z$ yields a system

$$\theta z = B(x)z, \tag{14}$$

which is simple. Let $\tilde{I}(\lambda)$ denote the indicial polynomial of the corresponding simple system.

(ii) If $\tilde{I}(\lambda)$ has no integer root then (9) has no solution in $k((x))^m \setminus \{0\}$.

(iii) Otherwise a solution of the truncation problem (for the input system (9)), is given by the sequence

$$a_l = mq + \max\{e^* - e_* + 1,\, l + (m-1)q\}, \quad (l = 1, 2, \dots) \tag{15}$$

where e_*, e^* are the minimal and maximal integer roots of $\tilde{I}(\lambda)$, respectively.

Proof. (i) The algorithm from [13] computes the so-called super-irreducible form of a given system (9). It was shown in [3] that if a system has the super-irreducible form then it can be written as a simple system. The algorithm from [13] needs at the most $(m-1)q$ reduction steps (see, for example, the proof of Proposition 2.2 in [10]). At each step, a transformation matrix with span 1 is computed. Overall, this shows the estimate on the span of $T(x)$.

(ii) Compute an invertible matrix $T(x) \in \mathrm{Mat}_m(k[x])$ such that the matrix

$$B(x) = T^{-1}(x)A(x)T(x) - T^{-1}(x)\theta T(x)$$

defines the system (14). Write (14) as a simple system $D(x)\theta z = N(x)z$, and let $\tilde{I}(\lambda) = \det(N_0 - \lambda D_0)$ be its indicial polynomial. If $\tilde{I}(\lambda)$ does not have integer roots, the simple system does not have any solutions in $k((x))^m \setminus \{0\}$. Hence the original system (9) cannot have solutions of this form either.

(iii) In order to solve the truncation problem for the input system, note that due to the relationship $y = T(x)z$ between solutions y of the input system and

z of the simple system, we have to compute at the most $l + \sigma(T(x))$ terms of z, if we need l terms of y. Using Proposition 3 to first solve the truncation problem for the simple system $D(x)\theta z = N(x)z$, we obtain $\tilde{a}_l = \max\{e^* - e_* + 1, l + \sigma(T(x))\}$. It then remains to show how many terms of the input system are required in order to ensure that we have $\tilde{a}_l$ terms of the simple system. This can be seen as follows: for any $j \geq 0$, the coefficients D_j and N_j of the simple system depend on the coefficients $B_0, \ldots, B_{q(B)-1+j}$ of the matrix $B(x)$, due to the construction of $D(x)$. Here, $q(B) = \max(0, -\mathrm{val}_x(B(x))$. Using Lemma 1, the coefficient $B_{q(B)-1+j}$ depends only on the coefficients A_i with $i \leq j + q(B) - 1 + \sigma(T(x))$ of $A(x)$. The proof is completed by the fact that $\sigma(T(x)) \leq (m-1)q$ for the transformation matrix $T(x)$ computed by the super-reduction algorithm as shown in (i), and that $q(B) \leq q$. □

Example 3. *Let $k = \mathbb{Q}$, q be a positive integer and S be the first-order system*

$$\theta y = A(x)y \quad where \quad A(x) = \begin{pmatrix} x^4 & x^{3-q} \\ -x^{q+5} - 4x^{q+1} & -x^4 + (q-3) \end{pmatrix},$$

which has as a basis of solutions

$$y_1(x) = \begin{pmatrix} 1 \\ -x^{q+1} \end{pmatrix} \quad y_2(x) = \begin{pmatrix} \ln(x) \\ -x^{q+1}\ln(x) - x^{q-3} \end{pmatrix}.$$

Hence the space of solution of S in $k((x))^2$ has dimension 1. This system S is not simple. Let $T(x) = \begin{pmatrix} 1 & 0 \\ 0 & x^q \end{pmatrix}$ of span q. Then the substitution $y = T(x)z$ yields the equivalent system

$$\theta z = B(x)z \quad where \quad B(x) = \begin{pmatrix} x^4 & x^3 \\ -x^5 - 4x & -x^4 - 3 \end{pmatrix}.$$

The latter system is simple, and its indicial polynomial is

$$\tilde{I}(\lambda) = \begin{vmatrix} \lambda & 0 \\ 0 & \lambda + 3 \end{vmatrix} = \lambda(\lambda + 3).$$

The roots of $\tilde{I}(\lambda)$ are -3 and 0, thus $e^* - e_* + 1 = 4$, $a_1 = a_2 = a_3 = 2q + 4$ and $a_l = 2q + l$ for $l \geq 4$. Take, for example, $q = 3$. Then for $l = 4, 5, 6, 7$ the l-truncation of S is

$$\theta y = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} y.$$

The latter system has one independent solution in $k((x))^2$, namely $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$.

3.2.2 Extension to Higher Order Systems

The results from the previous section can be easily extended to a system of the form

$$\theta^r y = -U_{r-1}(x)\theta^{r-1}y - \cdots - U_0(x)y \tag{16}$$

where $U_0(x), \ldots, U_{r-1}(x) \in \mathrm{Mat}_m(k((x)))$. The idea is that there exists a linear first order system $\theta Y(x) = U(x)Y(x)$ with companion block matrix $U(x)$ that corresponds to (16). This matrix belongs to $\mathrm{Mat}_n(k((x)))$, $n = rm$:

$$U(x) = \begin{pmatrix} 0 & I_m & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & I_m \\ -U_0(x) & -U_1(x) & \ldots & -U_{r-1}(x) \end{pmatrix}. \tag{17}$$

Applying Proposition 4 to system $\theta Y(x) = U(x)Y(x)$ we obtain the following proposition:

Proposition 5. *Let $q = \max\{0, -\mathrm{val}_x U_{r-1}(x), \ldots, -\mathrm{val}_x U_0(x)\}$. There exists an algorithm, that uses only the first rmq terms (i.e., terms of degree less than rmq) of the matrices $x^q U_0(x), \ldots, x^q U_{r-1}(x)$, and computes a nonzero polynomial $\tilde{I}(\lambda)$ such that:*

– *if $\tilde{I}(\lambda)$ has no integer root then (16) has no solution in $k((x))^m \setminus \{0\}$,*
– *otherwise a solution of the truncation problem is given by the sequence*

$$a_l = rmq + \max\{e^* - e_* + 1,\, l + (rm-1)q\} \quad (l = 1, 2, \ldots), \tag{18}$$

where e_*, e^* are the minimal and maximal integer roots of $\tilde{I}(\lambda)$, respectively.

3.2.3 Systems with Invertible Leading Matrices

Let the leading matrix of a system S of the form (1) be invertible. In this case S can be rewritten as the system $\bar{S}$ of the form

$$\theta^r y = -A_r^{-1}(x)A_{r-1}(x)\theta^{r-1}y - \cdots - A_r^{-1}(x)A_0(x)y. \tag{19}$$

Let

$$\gamma = \min_i \mathrm{val}_x \left(A_r^{-1}(x)A_i(x)\right), \tag{20}$$

and $q = \max\{-\gamma, 0\}$. The question to be answered is: given a non-negative integer v, how many first terms of the entries of S do we need to compute v first terms of

$$x^q A_r^{-1}(x)A_{r-1}(x), \ \ x^q A_r^{-1}(x)A_{r-2}(x), \ \ \ldots, \ \ x^q A_r^{-1}(x)A_0(x)? \tag{21}$$

Before answering this question we formulate a few facts related to the operations which we use to transform S to $\bar{S}$. As before, we suppose that all power series are represented algorithmically.

(A) Let it be known in advance that amongst the given series

$$s_1(x), s_2(x), \ldots, s_p(x) \in k[[x]], \quad p \geq 1,$$

there is at least one non-zero. Then we can compute

$$\nu = \min_i \mathrm{val}_x s_i(x).$$

To do this we consider the series $s_1(x), s_2(x), \ldots, s_p(x)$ "in parallel": we generate algorithmically the sequence

$$[x^0]s_1(x), \dots, [x^0]s_p(x), [x^1]s_1(x), \dots, [x^1]s_p(x), \dots$$

until we find i such that $[x^i]s_j(x) \neq 0$ for some $1 \leq j \leq p$. Then $\nu = i$.

(B) Let it be known in advance that among given matrices

$$M_1(x), M_2(x), \dots, M_p(x) \in \mathrm{Mat}(k[[x]]), \quad p \geq 1,$$

there is at least one non-zero. Then we can compute $\min_i \mathrm{val}_x M_i(x)$. To do this we consider the entries of all the matrices "in parallel" (as in (A)).

(C) Let it be known in advance that a matrix $M(x) \in \mathrm{Mat}(k[[x]])$ is invertible. We can compute $\mathrm{val}_x \det M(x)$, using $\mathrm{val}_x \det M(x) + 1$ initial entries of the matrix M. We can also compute $\mathrm{val}_x M^{-1}(x)$ which is equal to the difference of the minimum of the valuation of all co-factors of $M(x)$ and $\mathrm{val}_x \det M(x)$. This difference is non-positive, thus, we use $\mathrm{val}_x \det M(x) + 1$ initial terms of the entries of the matrix $M(x)$.

Every time when below in (A′), (B′), (C′) and in Proposition 6 we tell about the first w terms (where w is a positive integer) of entries of some matrices belonging to $k[[x]]$, we have in mind the terms of degree less than w.

We get from (A), (B), (C) the following.

(A′) We use $\mathrm{val}_x \det A_r(x) + 1$ first terms of the entries of the matrix $A_r(x)$ to compute $\mathrm{val}_x \det A_r(x)$ and $\mathrm{val}_x A_r^{-1}(x)$.

(B′) We use no more than $\mathrm{val}_x \det A_r(x) + \gamma + 1$ first terms of the entries of the matrices $A_0(x), A_1(x), \dots, A_r(x)$ to compute γ (see (20)).

(C′) We use no more than $\mathrm{val}_x \det A_r(x) + \gamma + v$ first terms of the entries of the matrices $A_0(x), A_1(x), \dots, A_r(x)$ to compute the first v terms of (21).

This and Proposition 5 imply the following statement related to systems of the form (1) with invertible $A_r(x)$.

Proposition 6. *Let γ be as in (20) and $q = \max\{-\gamma, 0\}$. There exists an algorithm, that uses only the first*

$$rmq + \gamma + \mathrm{val}_x \det A_r(x) + 1$$

terms of the entries of the matrices $A_0(x), A_1(x), \dots, A_r(x)$, and computes a nonzero polynomial $\tilde{I}(\lambda)$ such that:

- *if $\tilde{I}(\lambda)$ has no integer root then (1) has no solution in $k((x))^m \setminus \{0\}$,*
- *otherwise a solution of the truncation problem is given by the sequence*

$$a_l = rmq + \gamma + \mathrm{val}_x \det A_r(x) + \max\{e^* - e_* + 1,\, l + (rm-1)q\} \quad (l = 1, 2, \dots), \tag{22}$$

where e_*, e^* are the minimal and maximal integer roots of $\tilde{I}(\lambda)$, respectively.

Finally we can formulate a consequence of the latter proposition:

Proposition 7. *For a given system S of the form (1) with invertible $A_r(x)$ we can compute algorithmically an integer $d \geq -1$ such that*

– *a solution of S in $k((x))^m \setminus \{0\}$ exists iff $d \geq 0$,*
– *if $d \geq 0$ then a solution of the truncation problem for S is represented by the sequence $a_l = d + l,\ l = 1, 2, \ldots$*

Proof. Indeed, we set $d = -1$ if the polynomial $\tilde{I}(\lambda)$ has no integer root and $d = 2rmq - q + \gamma + \mathrm{val}_x \det A_r(x) + e^* - e_* + 1$ otherwise, where $q, \gamma,\ \tilde{I}_S(\lambda)$, e^*, e_* are as in Proposition 6. □

The following example shows that unlike the scalar case in the case of system we cannot in general take a sequence (a_l) such that $a_l = l$ at least for all l large enough.

Example 4. *Consider the system $x\theta y = A(x)y$ where*

$$A(x) = \begin{pmatrix} 0 & 1 \\ x^2 u(x) & 0 \end{pmatrix},$$

$u(x) = x + x^2 + x^3 + \ldots,\ y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$. *It is easy to show that y_1 satisfies the equation*

$$\theta^2 y_1 + \theta y_1 - u(x) y_1 = 0. \tag{23}$$

For the latter equation the sequence $b_l = l,\ l = 1, 2, \ldots$, is a solution of the truncation problem. For any other solution (b'_l) of this problem we will have $b'_l \geq b_l,\ l = 1, 2, \ldots$. Note that l-truncation of the original system $x\theta y = A(x)y$ induces $(l-2)$-truncation of (23). However, the sequence $c_l = l - 2,\ l = 1, 2, \ldots$, is not a solution of the truncation problem for (23). Thus, a sequence (a_l) which is a solution of the truncated problem for the original system must be such that $a_l \geq l + 2,\ l = 1, 2, \ldots$ If we replace in the original system x^2 by x^{2q} with integer $q > 1$, then we will obtain $a_l \geq l + 2q,\ l = 1, 2, \ldots$

Our results can be also used for solving the following problem. Suppose that for a system S of the form (1) only a finite number of terms of the entries of $A_0(x), A_1(x), \ldots, A_r(x)$ is known. So we know not the system S itself but the system $S^{\langle v \rangle}$ for some non-negative integer v. Suppose that we also know that $A_r(x)$ is invertible and that S has solutions in $k((x))^m \setminus \{0\}$. How many terms of these solutions can be determined from the given "approximate" system $S^{\langle v \rangle}$? Some non-trivial lower bound can be obtained from Propositions 1, 5, and 6.

4 Implementation

We have used the results obtained in this paper to improve some functionality contained in the Maple package ISOLDE [11]. The *RegularSolutions* function computes formal regular solutions of first order linear functional systems such as systems of linear differential, difference, and q-difference equations. In particular, it can be used for computing truncated Laurent series solutions of first order linear differential systems.

The old implementation used the sequence $a_l = l$ and did hence not always compute the accurate space of truncated Laurent series solutions, as the following example shows:

```
> A := linalg[matrix](2,2,[1,x^1,x^2*sin(x),3/x]);
```

$$A := \begin{bmatrix} 1 & x \\ x^2 \sin(x) & 3\,x^{-1} \end{bmatrix}$$

```
> L := LocalLinearDifferentialSystem(A,x,0);
```

$$L := L1$$

```
> RegularSolutions(L,x,2);
```

$$[[[\_C_1 + x\_C_1, 0], \{\}]]$$

Here, for $l = 2$, the function returns only one truncated Laurent series. We have added the new option *'allSolutions'* which ensures that a complete basis of the regular solutions space is computed, by taking into account formula (15). The sequence is then $a_1 = a_2 = 3$ and $a_l = l$ for $l \geq 3$, since the indicial polynomial of the system has roots 0 and 3. The output is then

```
> RegularSolutions(L,x,2,'allSolutions');
```

$$[[[\_C_1 + x\_C_1, x^3\_C_2], \{\}]]$$

This new feature will be available in the upcoming new release of ISOLDE.

5 Conclusion

In this paper, we have investigated the existence and truncation problem for higher-order linear differential systems. We have shown that they are undecidable in the general case but they can be solved in the case of the system's leading matrix being invertible. In the decidable cases, this means that we can reduce the problem of finding Laurent series solutions of systems with power series coefficients to that of finding the same type of solutions for systems with polynomial coefficients. A number of methods exist to do this task efficiently (e.g., [1,6]).

The mathematical techniques we employ in this paper use the algebra of polynomials and matrices, and we give explicit formulae for finding a_l for a given l. An implementation of our results can be done easily in any computer algebra system, as demonstrated in the previous section for the Maple package ISOLDE, and this equally applies to the implementations of the algorithms from [1,6]. We hope that this paper hence also makes a practical contribution to the scientific computing community, wishing to use computer algebra for handling systems of linear differential equations.

From our work, new questions arise. For example, can we solve the existence and truncation problem when we know in advance that the equations of a given system are independent over $k((x))[\theta]$ while the leading matrix is not invertible? Can our results be extended to more general classes of equations, such as difference and q-difference systems? We will continue to investigate this line of enquiry.

References

1. Abramov, S., Bronstein, M., Khmelnov, D.: On regular and logarithmic solutions of ordinary linear differential systems. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2005. LNCS, vol. 3718, pp. 1–12. Springer, Heidelberg (2005)
2. Barkatou, M.A.: A rational version of Moser's Algorithm. In: ISSAC 1995 Proceedings, pp. 297–302. ACM Press, New York (1995)
3. Barkatou, M.A.: On rational solutions of systems of linear differential equations. J. Symbolic Computation 28, 547–567 (1999)
4. Barkatou, M.A., Cluzeau, T., El Bacha, C.: Algorithms for regular solutions of higher-order linear differential systems. In: Johnson, J.R., Park, H., Kaltofen, E. (eds.) ISSAC 2009 Proceedings, pp. 7–14. ACM Press, New York (2009)
5. Barkatou, M.A., Cluzeau, T., El Bacha, C.: Simple forms of higher-order linear differential systems and their applications to computing regular solutions. J. Symbolic Computation 46, 633–658 (2011)
6. Barkatou, M.A., El Bacha, C., Pflügel, E.: Simultaneously row- and column-reduced higher-order linear differential systems. In: Koepf, W. (ed.) ISSAC 2010 Proceedings, pp. 45–52. ACM Press, New York (2010)
7. Barkatou, M.A., Pflügel, E.: An algorithm computing the regular formal solutions of a system of linear differential equations. J. Symbolic Computation 28, 569–588 (1999)
8. Barkatou, M.A., Pflügel, E.: On the equivalence problem of linear differential systems and its application for factoring completely reducible systems. In: Gloor, O. (ed.) ISSAC 1998 Proceedings, pp. 268–275. ACM Press, New York (1998)
9. Barkatou, M.A., Pflügel, E.: Computing super-irreducible forms of systems of linear differential equations via Moser-reduction: A new approach. In: Dongming, W. (ed.) ISSAC 2007 Proceedings, pp. 1–8. ACM Press, New York (2007)
10. Barkatou, M.A., Pflügel, E.: On the Moser- and super-reduction algorithms of systems of linear differential equations and their complexity. J. Symbolic Computation 44, 1017–1036 (2009)
11. Barkatou, M.A., Pflügel, E.: The ISOLDE package. A SourceForge Open Source project (2006), http://isolde.sourceforge.net
12. Coddington, E., Levinson, N.: Theory of Ordinary Differential Equations. McGraw-Hill, New York (1955)
13. Hilali, A., Wazner, A.: Formes super–irréductibles des systèmes différentiels linéaires. Numer. Math. 50, 429–449 (1987)
14. Lutz, D.A., Schäfke, R.: On the identification and stability of formal invariants for singular differential equations. Linear Algebra and Its Applications 72, 1–46 (1985)
15. Pflügel, E.: Effective formal reduction of linear differential systems. Applicable Algebra in Engineering, Communication and Computation 10, 153–187 (2000)
16. Turing, A.: On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society, Series 2 42, 230–265 (1936)
17. Maple online help: http://www.maplesoft.com/support/help/

Topology of Families of Implicit Algebraic Surfaces Depending on a Parameter

Juan Gerardo Alcázar[⋆]

Departamento de Matemáticas, Universidad de Alcalá, E-28871 Madrid, Spain

Abstract. Given a family of algebraic surfaces, implicitly defined, depending on a parameter t, here we provide an algorithm for computing the different shapes arising in the family. The algorithm decomposes the real line into finitely many pieces (points and intervals) so that over each interval the shape is invariant, in the sense that the topology of the family can be described by means of the same simplicial complex. As a consequence, by applying known algorithms ([1], [6], [7], [11]) the different shapes in the family can be computed. The algorithm is due to a generalization of the ideas in [2] to the surface case.

1 Introduction

Families of surfaces depending on parameters are common in the context of CAGD. Examples like offset surfaces (where the parameter is the offsetting distance), canal or tubular surfaces (where the parameter is the "thickness" of the "tube") are well-known; other examples include surfaces with shape parameters, that are chosen in order to produce results with certain topological features. Now in this paper we consider a problem involving families of algebraic surfaces depending on one parameter. More precisely, we address the following question: given a family of algebraic surfaces depending on a parameter, implicitly defined by a real polynomial $F(x, y, z, t)$ (where t is the parameter), how can we determine the different shapes arising in the family? In our case, for "shape" we understand a simplicial complex topologically describing the surface (i.e. isotopic to it). We do not address here the problem of computing this simplicial complex; for this purpose we refer to existing papers in the literature ([1], [5], [6], [7], [11]). Otherwise, our problem here is the decomposition of the parameter space ($\mathbb{R}$, in this case) into finitely many pieces which are either points or intervals, such that over each piece, the topology of the surface is invariant. The main ingredient is the notion of *delineability*, used also in [5], [8], among others, for computing a triangulation of a semi-algebraic set in arbitrary dimension. However, the case of varieties depending on parameters, which is our main concern here and our contribution, is apparently not discussed in these references.

The ideas in the paper essentially correspond to the generalization of the results in [2], where an algorithm for computing the different shapes arising in

⋆ Supported by the Spanish " Ministerio de Ciencia e Innovacion" under the Project MTM2008-04699-C03-01.

a family of implicit algebraic curves is provided, to the surface case. The ideas in that paper have already been applied in papers like [3] or [4]: in the first one, an algorithm for computing the different shapes arising in a one parameter family of rational curves is provided; in the second one, the results in [2] are used for algorithmically determining certain topological features of an algebraic surface. So, we believe that the ideas here could be applied in a similar way to algorithmically solve, in appropriate cases, similar questions on families of rational surfaces, or implicit hypersurfaces.

We also want to mention that the ideas in the paper could be seen from the point of view of Cylindrical Algebraic Decomposition. In particular, the polynomial in the main result of the paper (namely, Theorem 1) coincides with Brown's projection set (see [8]) of the polynomial $F(x, y, z, t)$, whenever F fulfills the hypotheses in Section 2. However, since C.A.D. is usually presented in the context of quantifier elimination, the connection of this theory with our problem is, in our opinion, not obvious (although, perhaps, expectable). Hence, a contribution of this paper is to make this connection clearer.

We have structured the paper in four sections. The main result in the paper is provided in Section 2, together with the necessary hypotheses and some examples of application. The proof of this result is provided in Section 3. The paper ends with a brief section on open questions.

2 Statement of the Main Result

In order to state the main result of the paper, we need to introduce some notation and hypotheses, first. So, let $F \in \mathbb{R}[x, y, z, t]$ be a polynomial in the variables x, y, z, t. Moreover, we assume that the following hypotheses hold:

(i) *$F = F(x, y, z, t)$ is square-free, and it explicitly depends on the variable z* (otherwise the problem is reducible to that in [2]). Under this hypothesis, and denoting the derivative of F with respect to the variable z as F_z, it holds that the resultant $\text{Res}_z(F, F_z)$ does not identically vanish: indeed, since F depends on z, F_z is not identically 0. Then if $\text{Res}_z(F, F_z) = 0$ we deduce that F, F_z have some common factor depending on z. But this implies that F is not square-free, which cannot happen by hypothesis.

(ii) *The leading coefficient of F w.r.t. z does not depend on x, y.* Notice that this hypothesis can always be achieved by applying a linear change of coordinates of the type

$$X = x, Y = y + \alpha z, Z = x + \beta y + \gamma z$$

(iii) *The leading coefficient of $\text{Res}_z(F, F_z)$ does not depend on x.* Observe that for a fixed t, $H(x, y, t) = \text{Res}_z(F, F_z)$ is the projection onto the xy-plane of the space curve defined by $F(x, y, z, t), F_z(x, y, t, z)$. As a consequence, any rotation

$$X = ax + by, Y = -bx + ay, Z = z$$

applied on the variety $V(F, F_z)$ causes $H(x, y, t)$ to be rotated in the same way. Since almost all planar rotations transform $H(x, y, t)$ so that $\mathrm{lcoeff}_y(H(x, y, t))$ does not depend on y, we have that almost all the above rotations transform F in the desired way.

As a result, we have that the above hypotheses can be fulfilled by applying a linear change of coordinates; since this does not change the topology of the family members, in the sequel we will assume the above hypotheses hold. Now regarding t as a real parameter, for different instances of $t_0 \in \mathbb{R}$ we have different surfaces S_{t_0}, implicitly defined by $F(x, y, z, t_0)$. So, we will say that F defines a family of algebraic surfaces depending on the parameter t. We consider also the following polynomials; here, $\sqrt{\cdot}$ stands for the square-free part of a polynomial, and $D_w(G)$ represents the resultant $\mathrm{Res}_w(G, G_w)$:

$$M(x,y,t) = \sqrt{D_z(F)};\ \ N(x,t) = \sqrt{D_y(M)};\ \ R(t) = D_x(N)$$

By the above reasonings $D_z(F)$ is not identically zero, and hence $M \neq 0$; however, N or R might be identically 0. So, first we consider the case when $N, R \neq 0$. The special cases $N = 0, R = 0$ will be considered later (see Subsection 2.1). In this situation, we define

$$\mathcal{A} := \{t \in \mathbb{R} | R(t) = 0\}$$

That is, $\mathcal{A}$ is the set of real roots of $R(t)$. Then, we have the following result.

Theorem 1. *Let $t_i, t_{i+1} \in \mathbb{R}$ satisfy $(t_i, t_{i+1}) \cap \mathcal{A} = \emptyset$. Then, the shapes of S_{t_i}, S_{t_i+1} are the same (in the sense that there is a simplicial complex describing both the shapes of S_{t_i}, S_{t_i+1}).*

Hence, Theorem 1 implies that the only values of the parameter where the shape of the surfaces in the family may change, are those in $\mathcal{A}$. This way, the elements of $\mathcal{A}$ induce a finite partition of the real line so that each element of the partition (which is either a real interval, or a real number) gives rise to a same shape in the family. The proof of this theorem is provided in the next section. Notice that the decomposition of the parameter space that arises from the application of Theorem 1 is not necessarily optimal, in the sense that it can happen that the topology of the family stays invariant when t crosses an element of $\mathcal{A}$.

Remark 1. If the degree of F is $\mathcal{O}(n)$, then in the worst case the degrees of $M(x, y, d)$, $N(x, d)$ and $R(d)$ are $\mathcal{O}(n^2)$, $\mathcal{O}(n^4)$, and $\mathcal{O}(n^{16})$, respectively. If F is sparse, then the discriminants $D_z(F)$, $D_y(M)$, $D_x(N)$ are typically non-square-free, and therefore after taking out multiple factors, the degree is generally reduced. Nevertheless, it is clear that as the degree of F grows, the practical application of Theorem 1 encounters more difficulties. So, at the moment the algorithm is useful in practice just for small degrees, generally with spare inputs.

Example. Let us consider the family of quartics defined by

$$(x^2 - 1)^2 + (y^2 - 1)^2 + (z^2 - 1)^2 = d^2$$

In this case, one may check that $\mathcal{A} = \{-\sqrt{3}, -\sqrt{2}, -1, 0, 1, \sqrt{2}, \sqrt{3}\}$ (the whole computation, in the Maple 13 system, takes 0.297 seconds). By symmetry, it is enough to consider the behavior for $d \geq 0$. In this sense, we distinguish the following cases: (I) $d = 0$; (II) $d \in (0, 1)$; (III) $d = 1$; (IV) $d \in (1, \sqrt{2})$; (V) $d = \sqrt{2}$; (VI) $d \in (\sqrt{2}, \sqrt{3})$; (VII) $d = \sqrt{3}$; (VIII) $d > \sqrt{3}$. Pictures corresponding to these cases are shown in Figure 1; (I), (II), (III) are shown from left to right in the first row; (IV), (V) are also shown from left to right in the second row; furthermore, the right-most picture in the second row corresponds to a view of the interior part in the case (V); finally, in the last row pictures corresponding to the interior part of the surface in (VI), (VII) and (VIII) are provided (in all these cases the exterior part coincides with that of (V), and changes occur only within the surface).

Other Examples. We provide a table with other examples. In this table, we provide (from left to right): a brief description of the family; the total degree of the implicit equation of the family; the number of terms of this equation; the time consumed by the computation; and the size of the set $\mathcal{A}$ (i.e. the number of elements of $\mathcal{A}$). In all the cases, we have used an Intel Core 2 Duo processor with speeds revving up to 1.83 GHz and operating system Windows XP, and an implementation running on the Computer Algebra System Maple 13.

Table 1. Other Examples

Example	Description	Degree	n.terms	time	Size
1	Family of Cubics	3	5	0.218	3
2	Offset to Circular Cone	6	19	0.406	1
3	Offset to Ellipsoid	8	60	2.313	13
4	Offset to Two-Sheeted Hyperb.	8	60	3.328	3
5	Offset to One-Sheeted Hyperb.	6	19	0.391	1
6	Canal Surface around a Circle	4	15	0.297	3
7	Family of Quintics	5	6	5.047	8
8	Family of Septics	7	5	0.250	3
9	Family of Sextics	6	6	2531.671	47
10	Offset to Paraboloid	6	41	0.484	3

The timing in Example 9 illustrates the complexity of the computations in the cases when the degrees of the intermediate polynomials are not significantly reduced when taking out multiple factors.

2.1 Special Cases

We consider the following special cases: (1) $N \neq 0$, $R = 0$; (2) $N = 0$. One may see that in the first case, the only possibility is that $N(x, t)$ does not explicitly depend on x. Thus, denoting $\mathcal{B} = \{t \in \mathbb{R} | N(t) = 0\}$, the following result holds:

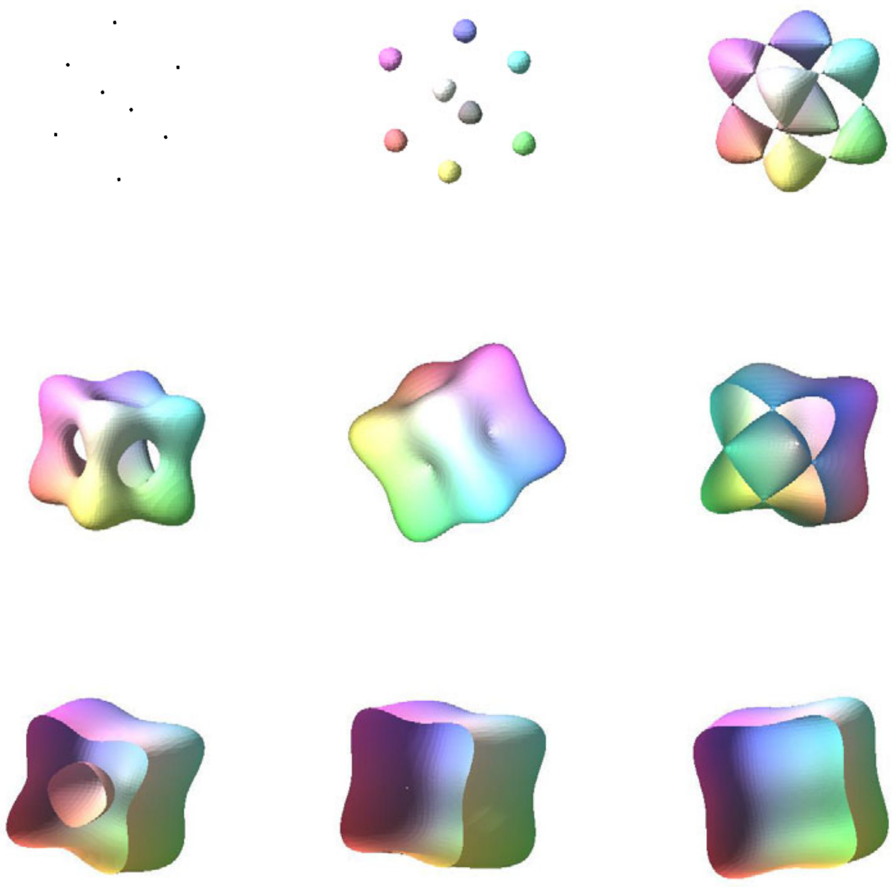


Fig. 1. The family $(x^2-1)^2+(y^2-1)^2+(z^2-1)^2=d^2$. From the first row to the last one, and from left to right, we have: (first row) cases $d=0$, $d\in(0,1)$, $d=1$; (second row) cases $d\in(1,\sqrt{2})$, $d=\sqrt{2}$ (outer part), $d=\sqrt{2}$ (inside); (third row) cases $d\in(\sqrt{2},\sqrt{3})$ (inside), $d=\sqrt{3}$ (inside), $d>\sqrt{3}$ (inside).

Theorem 2. *Assume that $N \neq 0$ but $R = 0$, and let $t_i, t_{i+1} \in \mathbb{R}$ satisfy that $(t_i, t_{i+1}) \cap \mathcal{B} = \emptyset$. Then, the shapes of S_{t_i}, S_{t_i+1} are the same (in the sense that there is a simplicial complex describing both the shapes of S_{t_i}, S_{t_i+1}).*

In the second case, we deduce that $M = M(x,t)$. In this situation, if $M = M(t)$ then we set $\mathcal{C} = \{t \in \mathbb{R} | M(t) = 0\}$, otherwise we set $\mathcal{C} = \{t \in \mathbb{R} | D_x(M)(t) = 0\}$. In any case, the following result holds.

Theorem 3. *Assume that $N = 0$, and let $t_i, t_{i+1} \in \mathbb{R}$ satisfy that $(t_i, t_{i+1}) \cap \mathcal{C} = \emptyset$. Then, the shapes of S_{t_i}, S_{t_i+1} are the same (in the sense that there is a simplicial complex describing both the shapes of S_{t_i}, S_{t_i+1}).*

3 Proof of the Main Result

In the sequel, we will prove Theorem 1; the proofs of Theorem 2 and Theorem 3 are analogous and in fact simpler, and are omitted here. For this purpose, we write

$$\mathcal{A} = \{a_1, \dots, a_n\}$$

where $a_1 < \cdots < a_n$; furthermore, we denote $a_0 = -\infty$, $a_{n+1} = \infty$. Then, our goal is to prove that for $t \in (a_i, a_{i+1})$, where $i = 0, \dots, n$, all the surfaces of the family can be described by the same simplicial complex; hence, that the shape of the family is invariant along (a_i, a_{i+1}).

Now the key of this proof is the notion of *delineability*. The reader may see [10] for further reading on this notion, or revise Section 3 in [2], for a brief review on it. Here we will simply recall the following formal definition:

Definition 1. *Let $\breve{x}$ denote the $(r-1)$-tuple $(x_1, \dots, x_{r-1})$. An r-variate polynomial $f(\breve{x}, x_r)$ over the reals is said to be* (analytic) delineable *on a submanifold $\mathcal{T}$ of $\mathbb{R}^{r-1}$, if it holds that:*

1. *the portion of the real variety of f that lies in the cylinder $\mathcal{T} \times \mathbb{R}$ over $\mathcal{T}$ consists of the union of the function graphs of some $k \geq 0$ analytic functions $\vartheta_1 < \cdots < \vartheta_k$ from $\mathcal{T}$ into $\mathbb{R}$,*
2. *there exist positive integers $m_1, \dots, m_k$ such that for every $a \in \mathcal{T}$, the multiplicity of the root $\vartheta_i(a)$ of $f(a, x_r)$ (considered as a polynomial in x_r alone) is m_i.*

Furthermore, the ϑ_i in the condition 1 of the definition above are called real root functions *(or simply* real roots*) of f on $\mathcal{T}$.*

Intuitively speaking, if a polynomial $G(x,y)$ is delineable on a subset $T \subset \mathbb{R}$, this means that over that subset, the real part of the curve defined by G consists of the union of finitely many *non-intersecting* curves, which correspond to the analytic functions ϑ_i of Definition 1. Similarly, if $F(x,y,z)$ is delineable on a subset $R \subset \mathbb{R}^2$, this implies that the real part of the surface defined by F over this subset is the union of finitely many *non-intersecting surfaces*, corresponding to the ϑ_i. A sufficient condition for a polynomial to be delineable over a given subset is provided in Theorem 2 of [10], pp. 246. By using this result, one can prove (see Section 4 of [2] for more details) that:

(1) For $i = 0, \ldots, n$, the polynomial $N(x,t)$ is delineable over $I_i = (a_i, a_{i+1})$. The real roots of N over I_i are represented as $X_{1,i}, \ldots, X_{r,i}$; also, we write $X_{0,i} = -\infty$, $X_{r,i} = \infty$. Moreover, the graphs of these functions are denoted as $\mathcal{X}_{1,i}, \ldots, \mathcal{X}_{r,i}$. Observe that these are 1-dimensional subsets of $\mathbb{R}^2$. Furthermore, for $k = 0, \ldots, r$ we also denote

$$S_{k,i} = \{(x,t) \in \mathbb{R}^2 | X_{k,i}(t) < x < X_{k+1,i}(t), a_i < t < a_{i+1}\}$$

These are 2-dimensional subsets of $\mathbb{R}^2$ (in fact, regions of the plane lying in between two consecutive $\mathcal{X}_{j,i}$'s, with $t \in I_i$)

(2) For $j = 1, \ldots, r$, the polynomial $M(x,y,t)$ is delineable over $\mathcal{X}_{j,i}$. The real roots of $M(x,y,t)$ over $\mathcal{X}_{j,i}$ are denoted as $Y_{1,j}, \ldots, Y_{\ell,j}$; the graphs of these functions are denoted $\mathcal{Y}_{1,j}, \ldots, \mathcal{Y}_{\ell,j}$. Observe that these are 1-dimensional subsets of $\mathbb{R}^3$.

(3) For $k = 0, \ldots, r$, the polynomial $M(x,y,t)$ is delineable over $S_{k,i}$. The real roots of $M(x,y,t)$ over $S_{k,i}$ are denoted as $V_{1,k}, \ldots, V_{s,k}$; the graphs of these functions are denoted $\mathcal{V}_{1,k}, \ldots, \mathcal{V}_{s,k}$. These are 2-dimensional subsets of $\mathbb{R}^3$.

Figure 2 illustrates the above functions. Also in [2] it is shown that the $\mathcal{V}_{q,k}$'s and the $\mathcal{Y}_{p,j}$'s "join properly", in the sense that there exists just one $\mathcal{Y}_{p,j}$ in the topological closure of each $\mathcal{V}_{q,k}$ (see Lemma 12 in [2]); in other words, that the situation suggested in Figure 2, right, is topologically correct.

In our case, we need to prove also the following result. In this sense, we need to introduce the following notation: given $V_{q,k}, V_{q+1,k}$, real roots of $M(x,y,t)$ over $S_{k,i}$, let

$$T_{q,k} = \{(x,y,t) \in \mathbb{R}^3 | V_{q,k}(x,t) < y < V_{q+1,k}(x,t), (x,t) \in S_{k,i}\}$$

Then the following lemma, which is proven in a similar way to Theorem 7 in [2], holds. In the proof of the result we will use the notions of *degree-invariance*

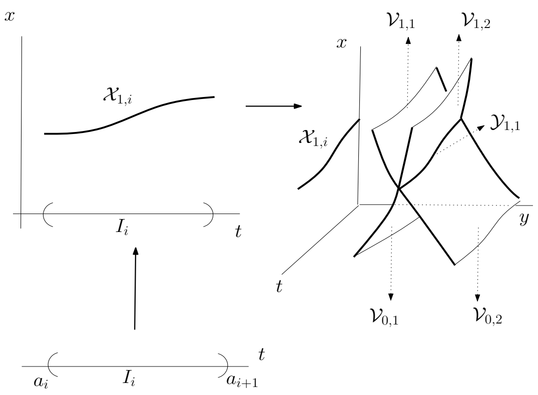


Fig. 2. Real Roots of Certain Functions

and *order-invariance*, introduced in [10]; the reader can also find them in [2] (see Definition 2 therein).

Lemma 1. *The following statements are true:*

(1) F is delineable over each $T_{q,k}$.
(2) F is delineable over each $\mathcal{Y}_{p,j}$.
(3) F is delineable over each $\mathcal{V}_{q,k}$.

Proof. The proof of (1) is analogous to that of statement (i), Theorem 7 in [2]. So, let us see (2), (3). For this purpose, let us check the conditions in the sufficient condition for delineability. Now by hypothesis $\mathrm{lcoeff}_z(F)$ depends only on the variable t, i.e. $\mathrm{lcoeff}_z(F) = h(t)$; moreover, by elementary properties of resultants $h(t)$ divides $R(t)$. So, whenever $t \in (a_i, a_{i+1})$ we have that $R(t)$ does not vanish, and therefore $h(t)$ does not vanish, either. As a consequence, F is degree-invariant along each $\mathcal{Y}_{p,j}$, and along each $\mathcal{V}_{q,k}$. On the other hand, $M = \sqrt{D_z(F)}$ is order invariant both over each $\mathcal{Y}_{p,j}$ and over each $\mathcal{V}_{q,k}$, because of Theorem 2 in [10]. Now let us see that $H := D_z(F)$ is also order invariant on each $\mathcal{Y}_{p,j}$ and on each $\mathcal{V}_{q,k}$. For this purpose, let $M = M_1 \cdots M_s$, where the M_i's are different and relatively prime, and let $H = M_1^{m_1} \cdots M_s^{m_s}$. We focus on $\mathcal{Y}_{p,j}$, first. Now let us see that $\mathcal{Y}_{p,j}$ must be fully contained in one of the M_i's. Indeed, if the order of M at any point of $\mathcal{Y}_{p,j}$ is 1 then $\mathcal{Y}_{p,j}$ does not contain any singular point of M, and therefore it must be fully contained in one of the M_i's. Otherwise, $\mathcal{Y}_{p,j}$ consists of singular points of M, and therefore it is contained in (see Exercise 11, p. 464 in [9])

$$\mathrm{Sing}(M_1) \cup \cdots \cup \mathrm{Sing}(M_s) \cup (M_1 \cap M_2) \cup \cdots \cup (M_{s-1} \cap M_s)$$

(where $\mathrm{Sing}(M_i)$ denotes the singular locus of M_i). Now assume by contradiction that $\mathcal{Y}_{p,j}$ has some points that belong to some M_i and other points that belong to some other M_ℓ, $\ell \neq i$, and not to M_i. So, $\mathcal{Y}_{p,j}$ has points in $\mathrm{Sing}(M_i)$, $\mathrm{Sing}(M_\ell)$ and $(M_i \cap M_\ell)$; moreover, these three sets do not coincide. Since $\mathcal{Y}_{p,j}$ is the image of a connected subset, namely $\mathcal{X}_{j,i}$, trough an analytic function, namely $Y_{p,j}$, it follows that $\mathcal{Y}_{p,j}$ is also connected. Now if $\mathrm{Sing}(M_i)$, $\mathrm{Sing}(M_\ell)$, $(M_i \cap M_\ell)$ project onto the same $\mathcal{X}_{j,i}$, since these sets do not coincide and the real roots of M over $\mathcal{X}_{j,i}$ are non-intersecting, we would deduce that $\mathcal{Y}_{p,j}$ is not connected, which cannot happen. Then they must project onto different $\mathcal{X}_{j,i}$'s; but this is again a contradiction because by definition $\mathcal{Y}_{p,j}$ projects just onto one $\mathcal{X}_{j,i}$. So, let us assume w.l.o.g. that $\mathcal{Y}_{p,j}$ is included in the zero set of, say, M_1. In that case, if the order of M at any point of $\mathcal{Y}_{p,j}$ is α, it is easy to see that the order of H at any point of $\mathcal{Y}_{p,j}$ is $\alpha \cdot m_1$. So, H is order invariant on $\mathcal{Y}_{p,j}$. Now let us consider $\mathcal{V}_{q,k}$. Notice that $\mathcal{V}_{q,k}$ is a two-dimensional subset of $\mathbb{R}^3$, and the singular locus of M is at most 1-dimensional. Hence, since M is order invariant over $\mathcal{V}_{q,k}$ then its order at any point of $\mathcal{V}_{q,k}$ must be 1 (because almost all points in $\mathcal{V}_{q,k}$ are non-singular). As a consequence, $\mathcal{V}_{q,k}$ must be completely contained in the zero-set of one of the M_i's (otherwise it would contain singular points and the order of M would be greater than 1 at those points). So if $\mathcal{V}_{q,k}$ is contained in the zero-set of M_i, the order of H on $\mathcal{V}_{q,k}$ is m_i. $\square$

Now we can introduce the following functions:

– Let $S_{1,q},\ldots,S_{\beta,q}$ be the real roots of F over $T_{q,k}$; let $\mathcal{S}_{1,q},\ldots,\mathcal{S}_{\beta,q}$ be the graphs of these functions.
– Let $T_{1,j},\ldots,T_{\gamma,j}$ be the real roots of F over $\mathcal{Y}_{p,j}$; let $\mathcal{T}_{1,j},\ldots,\mathcal{T}_{\gamma,j}$ be the graphs of these functions.
– Let $W_{1,k},\ldots,W_{\mu,k}$ be the real roots of F over $\mathcal{V}_{q,k}$; let $\mathcal{W}_{1,k},\ldots,\mathcal{W}_{\mu,k}$ be the graphs of these functions.

Although the images of the above functions lie in $\mathbb{R}^4$, let us try to visualize the intersections of their graphs with the hyperplane $\{t=t_0\}$. For this purpose, first we show, in Fig. 3, the intersection of Figure 2 with $t=t_0$; then, in Fig. 4, we show the intersections of the $\mathcal{S}_{a,q}$'s, the $\mathcal{T}_{b,j}$'s and the $\mathcal{W}_{c,k}$'s with $t=t_0$. In the case of Fig. 4, we have that S_{t_0} is homeomorphic to a cone. Furthermore, in Fig. 4 we just have one $\mathcal{T}_{b,j}$, which is responsible for the vertex of the cone, two $\mathcal{W}_{c,k}$'s, responsible for the two intersecting lines of the cone marked in thick line, and four $\mathcal{S}_{a,q}$'s responsible for the surface of the cone (with the exception of the vertex and the two intersecting lines).

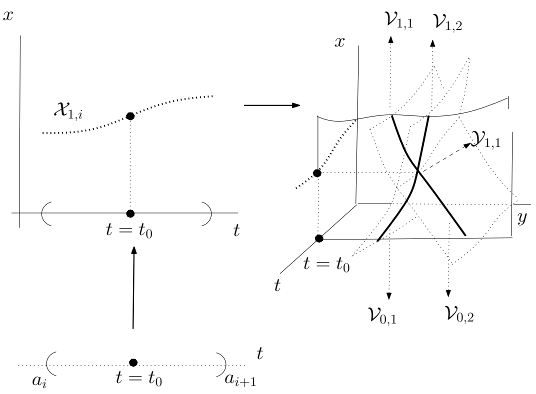


Fig. 3. Intersection of Fig. 2 with $\{t=t_0\}$

Moreover we also have the following result, which shows that the $\mathcal{S}_{a,q}$'s, the $\mathcal{T}_{b,j}$'s and the $\mathcal{W}_{c,k}$'s "join properly", in the sense that the adjacencies between them are kept invariant for $t\in(a_i,a_{i+1})$. The proof of this result is analogous to Phase 2 in [2] (see pp. 684-686). In this proof we will need the following subsets. First, $\mathrm{Cyl}(\mathcal{Y}_{p,j})$ is the cylinder over $\mathcal{Y}_{p,j}$, i.e. $\mathrm{Cyl}(\mathcal{Y}_{p,j})=\mathcal{Y}_{p,j}\times\mathbb{R}$; moreover, given $[t_a,t_b]\subset I_i$, we denote $\mathcal{Z}=\{(x,y,z,t)|t\in[t_a,t_b]\}$; finally,

$$T_i^\star=\mathrm{Cyl}(\mathcal{Y}_{p,j})\cap\mathcal{Z}\cap\overline{\mathcal{W}_{c,k}},$$

where $\overline{\mathcal{W}_{c,k}}$ stands for the topological closure of $\mathcal{W}_{c,k}$. Similarly,

$$\tilde{T}_i=\mathrm{Cyl}(\mathcal{V}_{q,k})\cap\mathcal{Z}\cap\overline{\mathcal{S}_{a,q}}$$

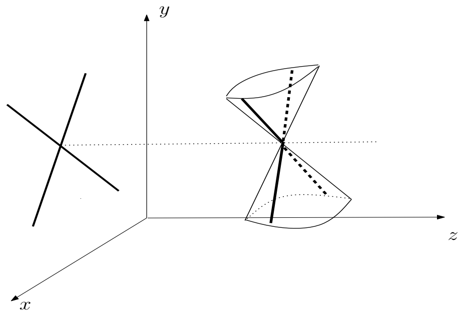


Fig. 4. The surface S_{t_0}

Then the following lemma holds.

Lemma 2. *Let $S_{k,i}$ be fixed, let the $\mathcal{V}_{q,k}$'s be the real roots of M over $S_{k,i}$, and let the $\mathcal{Y}_{p,j}$'s be the real roots of M over $\mathcal{X}_{k,i}$. Then the following statements are true:*

(i) For each $\mathcal{W}_{c,k}$, real root of F over $\mathcal{V}_{q,k}$ (resp. $\mathcal{V}_{q+1,k}$), there is just one $\mathcal{T}_{b,j}$, real root of F over $\mathcal{Y}_{p,j}$, such that $\mathcal{T}_{b,j} \subset \overline{\mathcal{W}_{c,k}}$.
(ii) For each $\mathcal{S}_{a,q}$, real root of F over $T_{q,k}$ (resp. $T_{q+1,k}$), there is just one $\mathcal{W}_{c,k}$, real root of F over $\mathcal{V}_{q,k}$ (resp. $\mathcal{V}_{q+1,k}$), such that $\mathcal{W}_{c,k} \subset \overline{\mathcal{S}_{a,q}}$.

Proof. In order to prove (i), proceed as in Phase 2 of [2] (see pp. 684-686), with two modifications: one must use the set $h_n^\star = V_{q,k}(h_n)$ instead of h_n, where h_n is the set defined in the proof of Lemma 10 of [2], and one must also use the set $T_i^\star$ (see above) instead of the set T_i introduced in Phase 2 of [2]. In order to prove (ii), the strategy is the same but now the modifications are as follows: $\tilde{h}_n = \{(x,y,t) \in \mathbb{R}^3 | (x,t) \in S_{k,i}, V_{q,k}(x,t) - \dfrac{1}{n} < y < V_{q,k}(x,t)\}$ plays the role of h_n, and $\tilde{T}_i$ (see above) plays the role of the set T_i. □

Remark 2. A result analogous to Lemma 2 can be proven for the real roots of M over $\mathcal{X}_{k+1,i}$, instead of $\mathcal{X}_{k,i}$.

Now let us provide a construction for the surface of the family corresponding to $t = t_0 \in (a_i, a_{i+1})$. For this purpose, by using the analytic functions that we have introduced, a cylindrical algebraic decomposition of S_{t_0} can be obtained. Afterwards, in order to prove Theorem 1 we will show that this decomposition is topologically the same for any $t_0 \in (a_i, a_{i+1})$. So, let $t_0 \in (a_i, a_{i+1})$; we first need a description of the zero set of $M(x, y, t_0)$ (which is an algebraic curve). This is done by using the functions $X_{a,i}$'s, $Y_{b,j}$'s and $V_{c,k}$'s introduced at the beginning of the section (see also Fig.3):

– Let $X_{1,i}(t_0), \ldots, X_{r,i}(t_0)$. These numbers are the x-coordinates of the critical points of $M(x, y, t_0)$ (see Lemma 14 in [2]). We can also say that this numbers correspond to the intersections

$$\mathcal{X}_{1,i} \cap \{t = t_0\}, \ldots, \mathcal{X}_{r,i} \cap \{t = t_0\}$$

– Let j be fixed. Then for $j = 1, \ldots, r$, $Y_{1,j}(X_{j,i}(t_0), t_0), \ldots, Y_{\ell,j}(X_{j,i}(t_0), t_0)$ are the y-coordinates of the points of $M(x, y, t_0)$ belonging to the critical (vertical) line $x = X_{j,i}(t_0)$. We can also say that these numbers correspond to the intersections

$$\mathcal{Y}_{1,j} \cap \{t = t_0\}, \ldots, \mathcal{Y}_{\ell,j} \cap \{t = t_0\}$$

– The branches of $M(x, y, t_0)$ with $X_{j,i}(t_0) < x < X_{j+1,i}(t_0)$ are the intersections

$$\mathcal{V}_{1,k} \cap \{t = t_0\}, \ldots, \mathcal{V}_{s,k} \cap \{t = t_0\}$$

Hence, let us provide a description of S_{t_0}. For this purpose, we need first the following observation.

Lemma 3. *Let $t_0 \in (a_i, a_{i+1})$. The xy-projection of the singular locus of S_{t_0} is contained in the zero-set of $M(x, y, t_0)$.*

Proof. Since $t_0 \in (a_i, a_{i+1})$, it follows that $\text{lcoeff}_z(F)$ does not vanish at $t = t_0$ (because otherwise by elementary properties of the resultant, $R(t_0) = 0$). Hence, the resultant $H(x, y, t) = \text{Res}_z(F, F_z)$ specializes well at $t = t_0$ (see Lemma 4.3.1 in [12]). Since the singular locus of S_{t_0} is contained in the variety defined by $F(x, y, z, t_0), F_z(x, y, z, t_0)$, the result follows. □

Then we are ready to show how a cylindrical algebraic decomposition of S_{t_0} can be computed; for this purpose, one proceeds as follows:

– *0-dimensional and 1-dimensional parts:* here we provide a decomposition of the part of the surface projecting onto $M(x, y, t_0)$). According to Lemma 3, this subset of S_{t_0} contains the singular locus of S_{t_0}. Hence, a cylindrical algebraic decomposition of it can be obtained in the following way:
  - (0-dimensional part) For $j = 1, \ldots, r$, compute the intersections of $\mathcal{T}_{1,j}$, $\ldots$, $\mathcal{T}_{\gamma,j}$ with $\{t = t_0\}$
  - (1-dimensional part) For $k = 0, \ldots, r$, compute the intersections of $\mathcal{W}_{1,k}$, $\ldots$, $\mathcal{W}_{\mu,k}$ with $\{t = t_0\}$.

– *2-dimensional part:* For $q = 0, \ldots, s$, compute the intersections of $\mathcal{S}_{1,q}$, $\ldots$, $\mathcal{S}_{\beta,q}$ with $\{t = t_0\}$.

We refer to the above decomposition as **Sup**. Finally, we can prove Theorem 1.

Proof of Theorem 1: By definition, the real roots of a polynomial over a certain subset are non-intersecting. So, the number of intersections of the $\mathcal{S}_{a,q}$'s, the $\mathcal{T}_{b,j}$'s and the $\mathcal{W}_{c,k}$'s with $\{t = t_0\}$ is invariant for $t_0 \in (a_i, a_{i+1})$. Moreover, since the $\mathcal{S}_{a,q}$'s are non-intersecting, the relative positions of the intersections of the $\mathcal{S}_{a,q}$'s with $\{t = t_0\}$ are invariant for $t \in (a_i, a_{i+1})$; similarly for the $\mathcal{T}_{b,j}$'s and the $\mathcal{W}_{c,k}$'s. Moreover, from Lemma 2 we deduce that the adjacencies of these intersections are also invariant for $t \in (a_i, a_{i+1})$. Hence, the above decomposition **Sup** is topologically invariant for $t \in (a_i, a_{i+1})$. Therefore, one can associate with it a simplicial complex which is the same for every $t \in (a_i, a_{i+1})$. □

4 Open Questions

The generalization of the main result here to one-parameter families of algebraic hypersurfaces of arbitrary dimension seems plausible, but at the moment we are unaware of a complete, formal proof. This generalization, and also the generalization to families with an arbitrary number of parameters, is therefore left here as an open problem.

References

1. Alberti, L., Mourrain, B., Tecourt, J.P.: Isotopic Triangulation of a Real Algebraic Surface. Journal of Symbolic Computation 44(9), 1291–1310 (2009)
2. Alcazar, J.G., Schicho, J., Sendra, R.: A Delineability-based Method for Computing Critical Sets of Algebraic Surfaces. Journal of Symbolic Computation 42, 678–691 (2007)
3. Alcazar, J.G.: Applications of Level Curves to Some Problems on Algebraic Surfaces. In: Lambán, L., Romero, A., Rubio, J. (eds.) Contribuciones Científicas en honor de Mirian Andrés Gómez, pp. 105–122. Univ. La Rioja (2010)
4. Alcazar, J.G.: On the Different Shapes Arising in a Family of Rational Curves Depending on a Parameter. Computer Aided Geometric Design 27(2), 162–178 (2009)
5. Basu, S., Pollack, R., Roy, M.F.: Algorithms in Real Algebraic Geometry. Springer, Heidelberg (2003)
6. Berberich, E., Sagraloff, M.: A generic and flexible framework for the geometrical and topological analysis of (algebraic) surfaces. Computer Aided Geometric Design 26(6), 627–647 (2009)
7. Berberich, E., Kerber, M., Sagraloff, M.: An Efficient Algorithm for the Stratification and Triangulation of Algebraic Surfaces. Computational Geometry: Theory and Applications 43(3), 257–278 (2009); Special Issue on 24th Annual Symposium on Computational Geometry
8. Brown, C.W.: Improved Projection for Cylindrical Algebraic Decomposition. Journal of Symbolic Computation 32(5), 447–465 (2001)
9. Cox, D., Little, J., O'Shea, D.: Ideals, Varieties and Algorithms. Springer, Heidelberg (1992)
10. MacCallum, S.: An Improved Projection Operation for Cylindrical Algebraic Decomposition. In: Caviness, B.F., Johnson, J.R. (eds.) Quantifier Elimination and Cylindrical Algebraic Decomposition, pp. 242–268. Springer, Heidelberg (1998)
11. Mourrain, B., Tecourt, J.P.: Isotopic Meshing of a Real Algebraic Surface, Rapport de recherche, n 5508. Unite de Recherche INRIA Sophia Antipolis (2005)
12. Winkler, F.: Polynomial Algorithms in Computer Algebra. Springer Verlag, ACM Press (1996)

A Modular Approach for Beam Lines Design

Serge N. Andrianov

Faculty of Applied Mathematics and Control Processes,
Saint Petersburg State University,
198504, Saint Petersburg, Russian Federation
`sandrianov@yandex.ru`

Abstract. We discuss advantages of numerical simulation based on symbolic presentations of beam line dynamical models. In some previous papers, some of these features were discussed. In this paper, we demonstrate how the symbolic presentation of necessary information can provide an in-depth study of different features of complex systems. For this purpose, we suggest a modular principle for all levels of the modeling and optimization procedures. This principle is based on so-called LEGO objects, which have both symbolic and numerical representation. For beam line design, it is necessary to support three types of similar objects. The first of them contains all necessary objects for beam line components description, the second contains all objects which correspond to particle beam models, and the third contains all objects corresponding to a transfer map ("a beam propagator"). In the suggested approach, the beam propagator is presented as a set of two-dimensional matrices describing different kinds of beam or beam line properties up to some approximation order. These matrices can be computed both in symbolic and numerical forms up to the necessary approximation order of the nonlinear effects. An example of practical application is demonstrated.

Keywords: Symbolic algebra, LEGO object, Lie algebraic methods, beam physics.

1 Introduction

Computational tools play a very important role in the design and operation of modern accelerators. The corresponding environment ensures the fulfillment of necessary computational procedures oriented both for beam lines design (on the development stage) and for accelerator system control (at the operation stage, for example, using EPICS[1]). During the design stage, the computer programs are plainly used to match the lattice design (select beam optics, compensate for the different effects, and analyze the stability of a beam and so on). In

[1] EPICS is a set of Open Source software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as a particle accelerators, telescopes, and other large scientific experiments. http://www.aps.anl.gov/epics/

commissioning and operation, these codes are used to build realistic accelerator models and control the correct beam behavior.

The LEGO concept (named by analogy to the well known children's game) – one of corresponding approaches for necessary environment design is proposed by J. Irwin with coauthors [1]. The concept is based on modular presentation for beam line structure description. For every module (*LEGO module*), they used numerical methods for corresponding integration of particles motion equations and some set of additional operations. In this paper, we consider the approach based on LEGO objects for all levels of the modeling process: from an initial design stage up to searching for optimal variants of required structures. The hierarchy of LEGO element classes suggested in [1] is maintained on the whole. But here we change some principles for presentation and computation of the corresponding LEGO objects. The mathematical core of this approach is based on the so-called *matrix presentation* for Lie algebraic tools. In general case, these problems can be described in terms of Lie algebraic methods (the most full description of these tools can be found in [2], see also [3,4]). According to this approach we have to constrain an evolution operator (propagator) in the form of a sequence of propagators describing partial propagators appropriate to a "time" step or another of an evolution sequence process. From algebraic point of view, one should consider a one-parameter Lie group based on some Lie algebra. The mathematical background for this well-known approach allows us performing necessary theoretical and practical manipulations successfully. In some previous works (see, e.g., [5]–[10]), the author considered the procedures, which are necessary for an evolution operator (propagator) evaluation according to the LEGO concept. In these works, both symbolic and numerical aspects of required calculations are considered. This approach allows the researcher to study an evolution of the phase manifold describing objects under study. Especially it is necessary to mention that the suggested approach can also be applied for problems of cosmic dynamics and molecular dynamics. Indeed, in all similar problems, we have to study an evolution of a family of objects (they are considered as "particles").

2 Matrix Presentation for LEGO Objects

In our approach, we describe the possibility of describing both the beam propagator and beam properties in terms of two-dimensional matrices. Here we have to particularize the following two problems.

The first problem is connected with presentation forms for the beam propagator – *beam propagator presentation*, and the second problem is related to the forms of *particle beam description*. It is obvious that the corresponding presentations have to match with each other.

2.1 The Basic Concepts – The Ordinary Differential Equations

In this paper, we consider two forms of time evolution equations [9]. The first type is based on the traditional form of ordinary differential equations

$$\frac{d\mathbf{X}}{dt} = \mathbf{F}(\mathbf{X}, t) = \sum_{k=0}^{\infty} \mathbb{P}^{1k}(t)\mathbf{X}^{[k]}, \tag{1}$$

where $\mathbf{F} = (F_1, \dots, F_n)^{\mathrm{T}}$ is a vector function, which generates a vector field

$$\mathcal{L}_{\mathbf{F}} = \mathbf{F}^{\mathrm{T}}(\mathbf{X}, t)\frac{\partial}{\partial \mathbf{X}}.$$

The operator $\mathcal{L}_{\mathbf{F}}$ is a Lie operator, and there is an expansion

$$\mathcal{L}_{\mathbf{F}} = \sum_{k=0}^{\infty} \left(\mathbf{X}^{[k]}\right)^{\mathrm{T}} (\mathbb{P}^{1k})^{\mathrm{T}}(t)\frac{\partial}{\partial \mathbf{X}} = \sum_{k=0}^{\infty} \mathcal{L}_{\mathbf{F}_k}(t),$$

where $\mathbf{X}^{[k]}$ is the Kronecker power for the vector $\mathbf{X}$, $\mathbb{P}_k$ is a matrix function with the matrix size $n \times \binom{n}{k}$ and

$$\mathcal{L}_{\mathbf{F}_k} = \left(\mathbf{X}^{[k]}\right)^{\mathrm{T}} \left(\mathbb{P}^{1k}(t)\right)^{\mathrm{T}} \frac{\partial}{\partial \mathbf{X}}, \quad \mathcal{L}_{\mathbf{F}_0} = \mathcal{I}d, \qquad \mathbf{F}_k(\mathbf{X}, t) = \mathbb{P}^{1k}(t)\mathbf{X}^{[k]}. \tag{2}$$

One can check the equality $\mathcal{L}_{\mathbf{F}_k} \circ \mathbf{X}^l = \mathbb{P}^l_k \mathbf{X}^{[k+l-1]}$, where $\mathbb{P}^l_k$ is a $\binom{n}{l} \times \binom{n}{k+l-1}$ matrix. Let $\mathbb{A}$ be an $n \times m$ matrix, then one can define a l-multiple Kronecker sum $\mathbb{A}^{\oplus l}$ according to the following rule:

$$\mathbb{A}^{\oplus l} = \mathbb{A}^{\oplus (l-1)} \otimes \mathbb{E}_n + \mathbb{E}_n^{[(l-1)]} \otimes \mathbb{A}, \quad \mathbb{A}^{\oplus 0} = \mathbb{O}, \quad \mathbb{E}_n^{[0]} = 1,$$

where $\mathbb{E}_n$ is the identity matrix. We can formulate the next theorem. In the book [9], one can find the following propositions.

Theorem 1 [9] *The matrix $\mathbb{P}^l_k$ for the operator $\mathcal{L}_{\mathbf{F}_k}$ in the space of homogeneous polynomials of lth order $\mathfrak{X}^{[l]}$ has the form*

$$\mathbb{P}^l_k = \left(\mathbb{P}^{1k}\right)^{\oplus l}.$$

Lemma 1 [9] *For any vector $\mathbf{X}$ ($\dim \mathbf{X} = n$), we have*

$$\frac{1}{(k-1)!}\frac{\partial^{k-1}}{\left(\partial \mathbf{X}^{\mathrm{T}}\right)^{k-1}}\mathbf{X}^{[k]} = \mathbb{X}^{\oplus k},$$

where $\mathbb{X}^{\oplus k}$ is a $\binom{n}{k} \times n^{(k-1)}$ matrix.

2.2 The Basic Concepts – The Hamiltonian Formalism

It is known that a wide class of physical systems are governed by Hamiltonian systems. In this case, for the right-hand side of eq. (1), we have

$$\mathbf{F}(\mathbf{X}, t) = \mathbb{J}\frac{\partial \mathcal{H}(\mathbf{X}, t)}{\partial \mathbf{X}}, \tag{3}$$

where $\mathcal{H}(\mathbf{X},t)$ is a Hamiltonian function (a Hamiltonian) for the dynamical system under study, $\mathbb{J}$ is a symplectic matrix and $\dim \mathbf{X} = 2n$. It is not difficult to note that the infinitesimal operator (Lie operator) for eq. (1) can be written as

$$\mathcal{L}_{\mathbf{F}} = \left(\mathbb{J}\frac{\partial \mathcal{H}(\mathbf{X},t)}{\partial \mathbf{X}}\right)^{\mathrm{T}} \frac{\partial}{\partial \mathbf{X}} = -\left[\mathcal{H}, \circ\right] = -\sum_{k=1}^{n}\left(\frac{\partial \mathcal{H}}{\partial Q_k}\frac{\partial}{\partial P_k} - \frac{\partial \mathcal{H}}{\partial P_k}\frac{\partial}{\partial Q_k}\right).$$

Let us present the Hamiltonian $\mathcal{H}$ as a series

$$\mathcal{H}(\mathbf{X},t) = \sum_{k=0}^{\infty} \mathbf{H}_k^{\mathrm{T}}(t)\mathbf{X}^{[k]},$$

then we can write

$$\mathbf{F}(\mathcal{H}(\mathbf{X},t)) = \mathbb{J}_0\frac{\partial \mathcal{H}}{\partial \mathbf{X}} = \mathbb{J}\sum_{k=2}^{\infty}\frac{\partial \mathbf{H}_k^{\mathrm{T}}(t)\mathbf{X}^{[k]}}{\partial \mathbf{X}},$$

from where according to [9] we have

$$\mathbf{F}(\mathcal{H}(\mathbf{X},t)) = \mathbb{J}_0\sum_{k=2}^{\infty}\left(\mathbb{E}_{2n}\otimes \mathbf{H}_k^{\mathrm{T}}\right)\frac{\partial \mathbf{X}^{[k]}}{\partial \mathbf{X}}.$$

From eq. (2) and above propositions, one can write

$$\mathbf{F}(\mathbf{X},t) = \mathbf{F}(\mathcal{H}(\mathbf{X},t)) = \sum_{k=2}^{\infty}\mathbb{P}^{1k}(t)(\mathcal{H})\mathbf{X}^{[k]},$$

where

$$\mathbb{P}^{1k}(t) = \mathbb{J}_0\left(\mathbb{E}_{2n}\otimes \mathbf{H}_{k+1}^{\mathrm{T}}\right)\mathrm{colon}\left(\left(\mathbf{E}^1\right)^{\oplus(k+1)}, \ldots, \left(\mathbf{E}^{2n}\right)^{\oplus(k+1)}\right),\ k \geq 1, \quad (4)$$

where $\mathbf{E}^i$ are the unit vectors ($\mathbf{E}_k^i = \delta_{ik}$), and $\mathbb{E}_{2n}$ is the $2n \times 2n$ identity matrix. The above described formulae allow us to compute the matrices $\mathbb{P}^{1k}$ for both forms of beam motion equations: in the form of ODE's and in the form of Hamiltonian equations. Just these matrices generate the LEGO objects of the first type enabling a description of the desired beam line.

2.3 Particle Beam Presentation

There are several approaches for beam manifold description. There are three of them, which are interesting for our purpose.

Phase Manifold Envelope Equations. In some beam physics problems, it is useful to know the "time evolution" for boundaries of a set occupied by beam particles. Let the boundary be described by the following vector equation

$$\mathbf{G}_0(\mathbf{X}) = \mathbf{G}(\mathbf{X}, t=0) = 0, \quad \dim \mathbf{G}_0 = 2n.$$

In the case of more popular representation using elliptical manifold, we have $n = 1$ and $G_0(\mathbf{X}) = \mathbf{X}^{\mathrm{T}}\mathbb{A}_0\mathbf{X} - 1$, where, for example, the matrix $\mathbb{S}_0 = \mathbb{A}_0^{-1}$ is an envelope matrix. This matrix can be defined from different point of view. For example, using statistical description (the corresponding manifold is considered as a point manifold). In this case, we use the term *root-mean-square envelope* (see, for example, [2]). In the case of representation (2.3) we can write the following expansion

$$\mathbf{G}_0(\mathbf{X}) = \sum_{k=0}^{\infty} \mathbb{G}_0^k \mathbf{X}^{[k]}.$$

The matrices $\mathbb{A}_0$ and $\mathbb{G}_0^k$, $k \geq 0$ describe the form of an initial phase manifold (for $t = 0$) in linear and nonlinear approximations.

The Phase Distribution Function Description. The second type is based on a phase distribution function $f_0(\mathbf{X})$ determining the distribution of particles in the initial manifold $\mathfrak{M}_0$. The function $f_0(\mathbf{X})$ can be written as

$$f_0(\mathbf{X}) = \sum_{k=0}^{\infty} \left(\mathbf{F}_0^k\right)^{\mathrm{T}} \mathbf{X}^{[k]}. \tag{5}$$

In (5), the vectors (the column matrices) $\mathbf{F}_0^k$, $k \geq 0$ describe the initial particle distribution on the initial manifold $\mathfrak{M}_0$.

The Pointwise Description. As the third type of description of particles ensembles here we consider the matrix $\mathbb{M}_0^N = \left\{\mathbf{X}_0^1, \mathbf{X}_0^2, \ldots, \mathbf{X}_0^N\right\}$, where N is the number of particles occupied in the initial phase manifold.

In all these cases, the following matrix objects are considered as LEGO objects for the initial phase manifold: 1) the matrices $\mathbb{A}_0$ or $\mathbb{G}_0^k$; 2) the column matrices $\mathbf{F}_0^k$ and the matrix $\mathbb{M}_0^N$.

It is obvious that for practical problems we have to cut off the corresponding sets. For the first two types, the truncation procedure is connected with our knowledge about control electromagnetic field. For the third case, the number of "points“ N is limited by computational limitation.

3 The Time Evolution of LEGO Objects

In this section, we consider the new class of LEGO objects describing the time evolution of our objects under study – a set of charged particles forming the beam. For this purpose, we use the matrix formalism for Lie algebraic tools suggested in [5]. This approach is based on constructive formulae for special matrices $\mathbb{R}^{1k}$, which present the new class of LEGO objects. In the case of the particle motion equations in the usual form of ODE's (see eqs. (1)), the desired solution can be written in the form [5]

$$\mathbf{X}(t) = \sum_{k=0}^{\infty} \mathbb{R}^{1k}(t|t_0)\mathbf{X}_0^{[k]}, \tag{6}$$

where matrices $\mathbb{R}^{1k}(t|t_0)$ accumulate the influence of all aberration effects (aberrations) in the kth order of the solution expansion in (6). For ODE's the corresponding formulae for evaluation of these matrices can be written in the following form of the matrix equations

$$\frac{d\mathbb{R}^{jk}(t|t_0)}{dt} = \sum_{i=0}^{\infty} \mathbb{P}^{ji}(t)\mathbb{R}^{ik}(t|t_0), \quad \mathbb{P}^{ij} = \mathbb{P}^{1(j-i+1)}\mathbb{P}^{(i-1)(j-1)}, \tag{7}$$

with the following initial data

$$\mathbb{R}^{kk}(t_0|t_0) = \mathbb{E}_{2n}^{[k]}, \;\; \mathbb{R}^{jk}(t_0|t_0) = \mathbb{O}, \forall\, j, k. \tag{8}$$

Here it is necessary to mention that in the case of piecewise presentation for the problem time interval $[t_0, t] = \bigcup_{k=\overline{1,m}} [t_{k-1}, t_k]$ (where $t_m = t$) there is a concatenation formula

$$\mathbb{R}^{ik}(\tau|t_0) = \sum_{j=i}^{k} \mathbb{R}^{ij}(\tau|t_1) R^{jk}(t_1|t_0), \; \forall\, \tau \in [t_0, t], \text{ and } \mathbb{R}^{\mathrm{kk}} = \left(\mathbb{R}^{11}\right)^{[\mathrm{k}]}.$$

The solution of eqs. (7) and (8) can be founded (up to necessary truncation order) both in the numerical and symbolic modes. In the case of numerical computation, some of numerical methods of ODE's solution (for example, the most popular Runge–Kutta methods) are used. In the symbolic mode, the different types of formulae can be used, see [9]. In the case of usual form of differential equations, the following simplest presentation can be used

$$\mathbb{R}^{ik}(t|t_0) = \sum_{j=i+1}^{k} \int_{t_0}^{t} \mathbb{R}^{ii}(t|\tau)\mathbb{P}^{ij}\mathbb{R}^{jk}(\tau|t_0)d\tau. \tag{9}$$

In the case of Hamiltonian motion equation, we can evaluate the matrices $\mathbb{P}^{ij}$ for Hamiltonian equations (see eq. (4)) and then integrate according to (9). For some applications it is useful to know the matrices $\mathbb{T}^{ik}(t_0|t)$ generated by the inverse map $\mathcal{T} = \mathcal{M}^{-1}$. These matrices can be evaluated using the generalized Gauss algorithm [9].

The second approach is based on the Lie map presentation for the corresponding Hamiltonian equations. According to this approach we write

$$\mathbf{X}(t) = \mathcal{M}(t|t_0) \circ \mathbf{X}_0,$$

where $\mathcal{M}(t|t_0)$ is the Lie operator (see [2]) for eq. (3). Thus, we have the following operator equation

$$\frac{d\mathcal{M}(t|t_0)}{dt} = \mathcal{L}_{\mathcal{H}} \circ \mathcal{M}(t|t_0), \tag{10}$$

where $\mathcal{L}_{\mathcal{H}} = -\left(\partial\mathcal{H}/\partial\mathbf{X}\right)^{\mathrm{T}} \mathbb{J}_0 \partial/\partial\mathbf{X}$.

Using the matrix formalism tools (see details in [5] and [9]) we can write

$$\mathcal{M}(t|t_0) \circ \mathbf{X}_0 = \sum_{k=1}^{\infty} \mathbb{M}^{1k}(t|t_0)\mathbf{X}_0^{[k]}.$$

The algorithm for the matrix $\mathbb{M}^{1k}(t|t_0)$ evaluation is described in detail in the book [9]. For necessary evaluations one can use some of factorization formulae (for example, the well known Dragt–Finn factorization [11]). Let $\mathcal{M}_{\leq k}$ denote the following Lie map $\mathcal{M}_{\leq k} = \mathcal{M}_k \circ \cdots \circ \mathcal{M}_2 \circ \mathcal{M}_1$ and $\mathcal{M}_k(t|t_0) = \exp \mathcal{L}_{\mathbf{G}_k(t|t_0)}$ and vector functions $\mathbf{G}_k$ can be evaluated according to the Dragt–Finn factorization and [9]. Then, for example, we can write

$$\mathcal{M}_{\leq 3} \circ \mathbf{X} = \mathbb{M}^{11} \left(\mathbf{X} + \sum_{m=2}^{3} \sum_{k=1}^{\infty} \frac{\mathbb{P}_m^{k1}}{k!} \mathbf{X}^{[k(m-1)+1]} + \right. \\ \left. + \sum_{l=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{k!l!} \mathbb{P}_2^{kl}\, \mathbb{P}_3^{l\,(k+1)}\, \mathbf{X}^{[2l+k+1]} \right),$$

where

$$\mathbb{P}_m^{k\,l} = \prod_{j=1}^{k} \mathbb{G}_m^{\oplus((j-1)(m-1)+l)}, \quad \mathbf{G}_k = \mathbb{G}_k \mathbf{X}^{[k]}.$$

So, we can write the following truncated formulae for $\mathbb{M}^{12}$ and $\mathbb{M}^{13}$:

$$\mathcal{M}_{\leq 2} \circ \mathbf{X} \approx \mathbb{M}^{11} \left(\mathbf{X} + \mathbb{P}_2^{11}\, \mathbf{X}^{[2]} \right) = \mathbb{M}^{11}\mathbf{X} + \mathbb{M}^{12}\mathbf{X}^{[2]}, \quad \mathbb{M}^{12} = \mathbb{M}^{11}\, \mathbb{P}_2^{11},$$

$$\mathcal{M}_{\leq 3} \circ \mathbf{X} \approx \mathbb{M}^{11} \left(\mathbf{X} + \mathbb{P}_2^{11}\, \mathbf{X}^{[2]} + \left(\mathbb{P}_3^{11} + \frac{1}{2!}\mathbb{P}_2^{21} \right) \mathbf{X}^{[3]} \right) = \\ = \mathbb{M}^{11}\mathbf{X} + \mathbb{M}^{12}\mathbf{X}^{[2]} + \mathbb{M}^{13}\mathbf{X}^{[3]}.$$

$$\mathbb{M}^{13} = \mathbb{M}^{11} \left(\mathbb{P}_3^{11} + \frac{1}{2!}\mathbb{P}_2^{21} \right).$$

The above described approach gives us the new LEGO objects – the matrices $\mathbb{R}^{1k}$ (for usual ODE's) or $\mathbb{M}^{1k}$ (for Hamiltonian differential equations) up to necessary truncation order. The corresponding symbolic computation can be evaluated up to necessary order of truncation (now some formulae up to fifth order for several types of control elements, such as dipoles, quadrupoles, and so on are calculated).

4 Auxiliary LEGO Objects

In the previous sections, we have described the basic classes of LEGO objects and their mathematical presentation in the matrix formalism. But often it is necessary to define some additional class of LEGO objects for introducing some additional demands to models. In this section, we describe (as an example of similar objects) the matrices describing nonlinear aberrations up to some order K.

The *first class of matrix LEGO objects* consists of some auxiliary matrices needed for our computational procedures. For example, "inverse" variants $\mathbb{T}^{1k}$ (and $\mathbb{T}^{ik}$, $i \leq k$) for matrices $\mathbb{R}^{1k}$ $\mathbb{M}^{1k}$ are often used. These matrices are evaluated in a symbolic mode (using generalized Gauss's algorithm) and kept in abstract forms, which are integrated to a special data base for control elements of different nature.

The *second class of "abstract"* matrices is composed of matrices $\mathbb{Q}^{1k}$, where $\mathbb{M}^{1k}(t|t_0) = \mathbb{M}^{11}(t|t_0)\mathbb{Q}^{1k}(t|t_0)$. The elements of these matrices are evaluated to provide the symplecticity condition.

Indeed, the transfer maps generated by Hamiltonian systems obey the symplectic condition, which can be in the following form

$$\mathbb{M}^*(s|s_0)\mathbb{J}_0\mathbb{M}(s|s_0) = \mathbb{J}_0, \quad \forall\, s \geq s_0, \tag{11}$$

where $\mathbb{J}_0 = \begin{pmatrix} \mathbb{O} & \mathbb{E} \\ -\mathbb{E} & \mathbb{O} \end{pmatrix}$ is the so-called canonical symplectic matrix, and $\mathbb{M}$ denotes the Jacobian matrix:

$$\mathbb{M}(s|s_0) = \frac{\partial}{\partial \mathbf{X}_0^{\mathrm{T}}}\Big(\mathcal{M}(s|s_0) \circ \mathbf{X}_0\Big),$$

The condition (11) is disturbed when we use a truncated propagator $\mathcal{M}^N$ instead of full propagator $\mathcal{M}$. Several symplectic integration methods have been proposed in the literature (see, for example, [13]–[15]).

In the matrix form, the action of the truncated map $\mathcal{M}^N$ up to the Nth order of nonlinearities obeys the following equation

$$\mathcal{M}_N(t|t_0) \circ \mathbf{X}_0 = \sum_{k=1}^{N} \mathbb{M}^{1k}(t|t_0)\mathbf{X}_0^{[k]}. \tag{12}$$

In contrast to full series, the symplectic condition (11) applied to a truncated map $\mathcal{M}_N$ will be broken. Using the matrices $\mathbb{Q}^{1k}$ we can write that eq. (12) in the form up to the Nth order of nonlinearities obeys the following equation

$$\mathcal{M}_N(s|s_0) \circ \mathbf{X}_0 = \mathbb{M}^{11}(s|s_0)\mathbb{Q}^{N}(s|s_0) = \mathbb{M}^{11}(s|s_0)\sum_{k=1}^{N} \mathbb{Q}^{1k}(s|s_0)\mathbf{X}_0^{[k]}. \tag{13}$$

Using eq. (9) we can write

$$\mathbb{R}^{1k}(s|s_0) = \mathbb{R}^{11}(s|s_0)\sum_{j=2}^{k}\int\limits_{s_0}^{s} \left(\mathbb{R}^{ii}(\tau|s_0)\right)^{-1}\mathbb{P}^{ij}(\tau)\mathbb{R}^{jk}(\tau|s_0)d\tau. \tag{14}$$

Using eqs. (13) and (14) one can evaluate

$$\mathbb{Q}^{1k}(s|s_0) = \sum_{j=2}^{k} \int_{s_0}^{s} \left(\mathbb{R}^{ii}(\tau|s_0)\right)^{-1} \mathbb{P}^{ij}(\tau)\mathbb{R}^{jk}(\tau|s_0)d\tau,$$

As the matrix $\mathbb{M}^{11}(s|s_0)$ satisfies corresponding symplecticity

$$\left(\mathbb{M}^{11}\right)^{\mathrm{T}}(s|s_0)\mathbb{J}_0\mathbb{M}^{11}(s|s_0) = \mathbb{J}_0, \quad \forall\, s \geq s_0,$$

then (11) constrains some conditions on elements of the matrices $\mathbb{Q}^{1k}$. In works [7]–[10], the necessary transformations are demonstrated. As an example let us consider one dimensional motion equations with second-order nonlinearities

$$\mathbf{X}(t) = \mathbb{M}^{11}\mathbf{X}_0 + \mathbb{M}^{12}\mathbf{X}_0^{[2]} = \mathbb{M}^{11}\left(\mathbf{X}_0 + \mathbb{Q}^{12}\mathbf{X}_0^{[2]}\right), \ \dim \mathbf{X} = 2.$$

The symplectic condition leads to linear algebraic homogeneous equations for matrix elements of $\mathbb{Q}^{12} = \{q_{ij}\}_{i=\overline{1,4},j=\overline{1,10}}$:

$$\mathbb{Q}^{12} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} & q_{15} & q_{16} & q_{17} & q_{18} & q_{19} & q_{110} \\ q_{21} & q_{22} & q_{23} & q_{15} & q_{25} & q_{17} & q_{27} & q_{19}/2 & 2\,q_{110} & q_{210} \\ q_{31} & q_{32} & q_{33} & -2\,q_{11} & -2\,q_{21} & -q_{12} & -q_{22} & q_{14}/2 & -q_{15} & q_{25}/2 \\ q_{32}/2 & 2\,q_{33} & q_{43} & -q_{12} & -q_{22} & q_{32}/2 & 2\,q_{33} & q_{43} & -q_{12} & -q_{22} \end{bmatrix}. \qquad (15)$$

Similar formulas can be obtained for any order of nonlinearities N and dimension of the phase vector $\mathbf{X}$. It should be noted that similar relations decrease the computational costs. Indeed in the case of $\dim \mathbf{X} = 4$ for 40 elements of $\mathbb{Q}^{12}$ we obtain 24 restrictions, and for 80 elements for $\mathbb{Q}^{13}$ – 60 restrictions of type (15). The analogous matrices can be calculated in symbolic forms for other aberration orders, can be kept in a special data base, and used according to the LEGO object paradigm. We also note that the corresponding conditions are right for all control elements.

Numerical integration algorithms play an essential role in problems of long-term beam evolution, stability of similar process, nonlinear non-integrable Hamiltonian systems, and so on. Unfortunately, standard numerical integration methods are not symplectic, and this violation of the symplectic condition (11) can lead to some false effects, for example, spurious chaotic, dissipative behavior and so on.

There is a set of numerical methods preserving some qualitative structure, which adheres to the dynamical system under study. These schemes will be noted as conservative integration schemes. It is necessary to distinguish two similar types of integration schemes.

The first type of beam propagator evaluation is based on the universal exponential identities or relations among Lie algebra because these maps have all requested properties. But its numerical realization loses these properties, and it is necessary to restore desired properties for a numerical variant of this map too. In this case, the second approach is used, which is based on the universal schemes such as different symplectic variants for traditional numerical integration schemes (see, for example, works by J.M. San-Serna [12] and others).

So we have two approaches for map evaluations. The first is based on symbolic presentation for $\mathbb{M}^{1k}$, which can be found for some restricted models for functional s-dependence of matrices $\mathbb{P}^{1k}(s)$. This set of models is defined by functions family, for which there are symbolic solutions appropriate for fringe field distribution. Among them the most familiar is the class of piecewise constant functions. The symbolic presentation for aberration matrices $\mathbb{M}^{1k}$ permits one to realize a parametrical investigation of the beam propagator. This feature is especially useful for optimization problems (see, for example, [16]). The second approach is based on numerical solution of differential equations for aberration matrices, which can be evaluated for arbitrary forms of fringe fields in the control elements. Here we lose not only the flexibility symbolic-specific, but possibility to keep corresponding solutions in data knowledge as LEGO objects.

5 Computational Experiments – Mass-to-Charge Ratio Separator Design Problem

The LEGO object paradigm (shortly described above) was applied to some beam physics problems and demonstrated enough flexibility and effectiveness. In this paper, we consider the problem of mass-separator design with high solid angle. The described approach allows a researcher to form an extensible sequence of approximating models for beam systems under study.

Let us formulate basic demands made on mass-to-charge ratio m/q separator systems.

1. To provide the *transport of an initial set of particles beams*, which corresponds to a separate sort of particles (in our case – with different mass-to-charge ratio. This requirement corresponds to aperture restrictions and is closely connected with the so-called luminosity of the facility.
2. To provide the *focusing procedure for each of beams* having the same mass-to-charge ratio. This condition leads us to improved particle beam registration.
3. To provide *sufficient separation of beams* with different m/q. The requirement corresponds to high quality separation of beam fractions with different m/q.

With the help of a special software (here we used the so-called modular programming technique) a researcher can design the system under study on a "*virtual work board*" for a designer (see fig. 1).

This designer's board allows us to determine all necessary parameters for beam line under study. Every visual element of this virtual builder corresponds to some LEGO objects. At the next step, the beam line designer evaluates the full propagator in the linear approximation according to

$$\mathbb{R}^{11}(s_{\mathrm{tot}}|s_0) = \mathbb{R}^{11}(s_N|s_{N-1}) \cdot \ldots \cdot \mathbb{R}^{11}(s_2|s_1) \cdot \mathbb{R}^{11}(s_1|s_0).$$

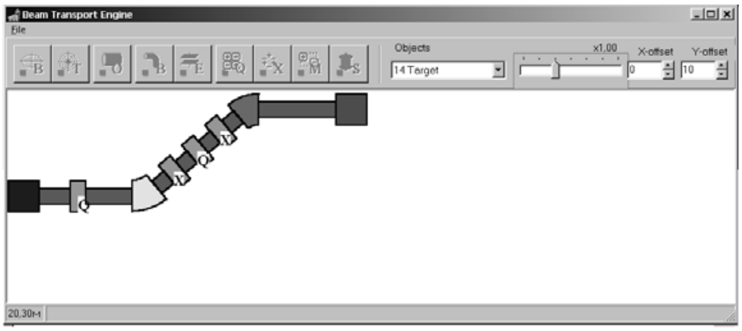
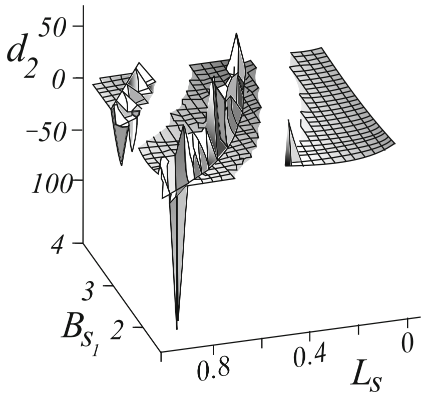
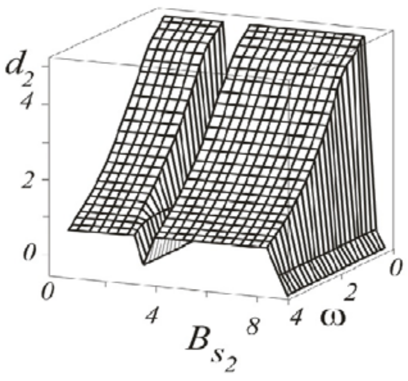


Fig. 1. A screen-shot of designer's "*virtual work board*"



(a) Dependence of distance between electrical and magnetic dipoles d_2 from initial solenoid magnetic field B_{s_1} and their length L_s



(b) Dependence of distance between electrical and magnetic dipoles d_2 from second solenoid magnetic field B_{s_1} and reduced gradients of the central quadrupole ω

Fig. 2. Examples of 3D Maple-images for a high solid angle mass-separator

Here every matrix $\mathbb{R}^{11}(s_{k+1}|s_k)$ correlates with the corresponding beam control elements as provided by the beam line. At the next steps, the designer investigates proposed variants. All described analytical calculations were realized in different computer algebra codes (such as Maxima, Maple or Mathematica). This permits to select more appropriate codes for problems under study. The particular attention is paid there to visualization problems. Just usage of 2D- and 3D-representation (including animate forms) of computing results (see, for example, Fig. 2) allows the designer to study a great number of variants of the corresponding beam line structure and to select the more appropriate variants of structure [9]. These graphical examples and their animating variants demonstrate the effectiveness of the above-described LEGO paradigm based on symbolic objects and corresponding manipulation procedures. This approach permits to realize the design of the solenoids-based mass-separator [9] with high resolution.

Acknowledgements. This work has been supported in part by Saint Petersburg State University – Subject Plan 9.0.111.2009 (Russian Federation). The author would like to thank prof. N.S. Edamenko for several enjoyable discussions. The author also thanks his postgraduate students in the Faculty of Applied Mathematics and Control Processes Saint Petersburg State University for several effective contribution to our collaboration. Their theoretical and programming skills for practical realization help us to achieve very interesting results.

References

1. Cai, Y., Donald, M., Irwin, J., Yan, J.: LEGO: A Modular Accelerator Design Code. SLAC-PUB-7642 (August 1997)
2. Dragt, A.J.: Lie Methods for Nonlinear Dynamics with Applications to Accelerator Physics, p. 1805. University of Maryland, College Park (2011), www.physics.umd.edu/dsat/
3. Dragt, A.J.: Lectures on nonlinear orbit dynamics. In: AIP Conf. Proc., vol. (87), pp. 147–313 (1987)
4. Dragt, A.J.: Lie Algebraic Treatment of Linear and Nonlinear Beam Dynamics. In: Annual Review of Nuclear and Particle Science, vol. 38, pp. 455–496 (1988)
5. Andrianov, S.N.: The explicit form for Lie transformations. In: Proc. Fifth European Particle Accelerator Conference EPAC 1996, SITGES (Barcelona, Spain), pp. 998–1000. Barselona (1996)
6. Andrianov, S.N.: Matrix representation of the Lie algebraic methods for design of nonlinear beam lines. In: AIP Conf. Proc., N.Y, vol. (391), pp. 355–360 (1997)
7. Andrianov, S.N.: Symbolic computation of approximate symmetries for ordinary differential equations. Mathematics and Computers in Simulation 57(3-5), 147–154 (2001)
8. Andrianov, S.N.: Lego-Technology Approach for Beam Line Design. In: Proc. EPAC 2002, Paris, France, pp. 1667–1669 (2002)
9. Andrianov, S.N.: Dynamical Modeling of Control Systems for Particle Beams. SPbSU, Saint Petersburg (2004) (in Russian)
10. Andrianov, S.N.: A role of symbolic computations in beam physics. In: Gerdt, V.P., Koepf, W., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2010. LNCS, vol. 6244, pp. 19–30. Springer, Heidelberg (2010)
11. Dragt, A.J., Finn, J.M.: Lie series and invariant functions for analytic symplectic maps. J. Math. Phys. 17(12), 2215–2227 (1976)
12. Sanz-Serna, J.M.: Symplectic integrators for Hamiltonian problems: an overview. Acta Numerica 1, 243–286 (1992)
13. Ruth, R.D.: A canonical integration technique. IEEE Trans. Nucl. Sci. 30, 2669 (1983)
14. Yoshida, H.: Construction of higher order symplectic integrators. Phys. Lett. A 150, 262 (1990)
15. Forest, E.: Canonical integrators as tracking codes. In: AIP Conf. Proc., vol. 184, pp. 1106–1136. American Institute of Physics, New York (1989)
16. Andrianov, S., Edamenko, N., Podzivalov, E.: Some problems of global optimization for beam lines. In: Proc. PHYSCON 2009, Catania, Italy, September 1-4 (2009), http://lib.physcon.ru/download/p1998.pdf

Computations on Simple Games Using RelView

Rudolf Berghammer[1], Agnieszka Rusinowska[2], and Harrie de Swart[3]

[1] Institut für Informatik, Universität Kiel, 24098 Kiel, Germany
[2] Centre d'Economie de la Sorbonne, CNRS – Université Paris I, 75647 Paris, France
[3] Department of Philosophy, Tilburg University, 5000 LE Tilburg, The Netherlands

Abstract. Simple games are a powerful tool to analyze decision-making and coalition formation in social and political life. In this paper we present relational models of simple games and develop relational algorithms for solving some game-theoretic basic problems. The algorithms immediately can be transformed into the language of the Computer Algebra system RelView and, therefore, the system can be used to solve the problems and to visualize the results of the computations.

1 Introduction

In game theory (starting with [12]) a distinction is made between non-cooperative games and cooperative games. In non-cooperative games each player (agent, party etc.) must decide individually, while in cooperative games players are allowed to act jointly and to decide within a group what strategy will be followed.

Cooperative games (see e.g. [13, 14] for an introduction) are a very useful tool for modeling the cooperation of players and for measuring the outcome caused by this. Simple games (also called 0/1-games) are a special class of cooperative games. Here the numerical payoff of a coalition (a set of players) is either 1 or 0. Hence, only two classes of coalitions are possible. The winning ones (their payoff is 1) take all and the losing ones (with payoff 0) receive nothing. This kind of games is very important, e.g., in social choice theory, for the comparison and measurement of influence and power of agents in decision-making processes and for the analysis of social and political situations. In respect of the latter application domain we refer to [8, 9, 16, 18], for example.

As demonstrated, for instance, in [10, 15], a lot of important problems on simple games are known to be intractable in terms of complexity theory. In the recent years we successfully have combined relation algebra (cf. [17]) and the BDD-based specific purpose Computer Algebra system RelView (cf. [4, 5]) for the formation of coalitions and alliances and to measure the strength of agents in social and political networks. See [6, 7] for details. In [3] this approach is extended to the solution of some standard problems on simple games like the detection of some key players, the test of some fundamental properties of simple games, and the computation of some power indices. All relation-algebraic solutions of [3] and, hence, also the corresponding RelView-programs, base on the so-called relational vector-model of simple games. An alternative model, called relational membership-model, is only used to facilitate the input of the RelView-programs and to visualize their computed results.

The present paper is a continuation of [3]. Besides the vector-model and the membership-model we study a third relational model of simple games, viz. the seat-distribution-model of the important sub-class of voting games. We also show how some standard problems on simple games can be solved using the membership-model instead of the vector-model. There are situations where the membership-model is more appropriate than the vector-model. Because RelView has a very efficient BDD implementation of relations, the tool is able to deal with non-trivial simple games that, for instance, appear in practical political life. In addition, the tool has visualization facilities which are not easily found in other software tools and which are most helpful for fully comprehending difficult concepts and for understanding and testing the programs. We will demonstrate the visualization of results by an example from the present practical political life, viz. the game that models the German parliament after the 2009 election.

2 Relational Preliminaries

We denote the set (type) of all relations with source X and target Y (i.e., the powerset $2^{X\times Y}$) by $[X \leftrightarrow Y]$ and write $R : X \leftrightarrow Y$ instead of $R \in [X \leftrightarrow Y]$. If X and Y are finite sets, then we may consider R also as a Boolean matrix. This interpretation is well suited for many purposes and Boolean matrices are also used as one of the graphical representations of relations within RelView. Therefore, in this paper we often use Boolean matrix terminology and notation. In particular, we speak of rows, columns and components/entries of relations and write $R_{x,y}$ instead of $\langle x, y\rangle \in R$ or $x\,R\,y$. We will employ the following basic operations on relations: $\overline{R}$ (*complement*), $R \cup S$ (*union*), $R \cap S$ (*intersection*), R^{T} (*transposition*) and RS (*composition*). Furthermore, we will use the special relations O (*empty relation*), L (*universal relation*) and I (*identity relation*). Here we overload the symbols, i.e., avoid the binding of types to them.

By $syq(R, S) = \overline{R^{\mathsf{T}}\,\overline{S}} \cap \overline{\overline{R}^{\mathsf{T}} S}$ the *symmetric quotient* of $R : X \leftrightarrow Y$ and $S : X \leftrightarrow Z$ is defined. The type of $syq(R, S)$ is $[Y \leftrightarrow Z]$, and transforming its definition into a component-wise notation, we have for all $y \in Y$ and $z \in Z$ that $syq(R, S)_{y,z}$ iff for all $x \in X$ it holds $R_{x,y}$ iff $S_{x,z}$.

A *vector* is a relation v with the specific set $\mathbf{1} := \{\bot\}$ as target. Since in $v_{x,\bot}$ the second index $\bot$ is irrelevant, we write in the following v_x instead of $v_{x,\bot}$. Vectors correspond to Boolean column vectors. We say that $v : X \leftrightarrow \mathbf{1}$ *describes* the subset Y of X if for all $x \in X$ we have $x \in Y$ iff v_x. In such a case $inj(v) : Y \leftrightarrow X$ denotes the *embedding-relation* of Y into X. This means that for all $y \in Y$ and $x \in X$ we have $inj(v)_{y,x}$ iff $y = x$. To model sets we also will use the relation-level equivalents of the set-theoretic symbol "$\in$", i.e., *membership-relations* $\mathsf{E} : X \leftrightarrow 2^X$ defined by $\mathsf{E}_{x,Y}$ iff $x \in Y$, for all $x \in X$ and $Y \in 2^X$. A combination of embedding-relations and membership-relations allows a *column-wise enumeration* of a subset of a powerset. If $v : 2^X \leftrightarrow \mathbf{1}$ describes a subset $\mathfrak{S}$ of 2^X in the sense defined above, then for all $x \in X$ and $Y \in \mathfrak{S}$ we have $(\mathsf{E}\,inj(v)^{\mathsf{T}})_{x,Y}$ iff $x \in Y$. Using Boolean matrix terminology this means that the elements of $\mathfrak{S}$ are described precisely by the columns of $\mathsf{E}\,inj(v)^{\mathsf{T}} : X \leftrightarrow \mathfrak{S}$.

A non-empty vector $v : X \leftrightarrow \mathbf{1}$ is a *point* if $vv^{\mathsf{T}} \subseteq \mathsf{I}$. This means that it describes a singleton subset of X or an element from X if we identify a singleton set with the only element it contains. In the Boolean matrix model, hence, a point $p : X \leftrightarrow \mathbf{1}$ is a Boolean column vector in which exactly one entry (component) is 1. If it describes $x \in X$, then for all $y \in X$ it holds p_y iff $x = y$.

For a direct product $X \times Y$ there are the projections which decompose a pair $u = \langle u_1, u_2 \rangle$ into its first component u_1 and its second component u_2. Within relation algebra it is very useful to consider instead of projections the corresponding *projection relations* $\pi : X \times Y \leftrightarrow X$ and $\rho : X \times Y \leftrightarrow Y$ such that, given any $u \in X \times Y$, it holds $\pi_{u,x}$ iff $u_1 = x$ and $\rho_{u,y}$ iff $u_2 = y$. Projection relations enable us to describe the well-known pairing operation of functional programming relation-algebraically as follows: For given relations $R : Z \leftrightarrow X$ and $S : Z \leftrightarrow Y$ we define their *pairing* (frequently also called *fork* or *tupling*) $[R, S] : Z \leftrightarrow X \times Y$ by $[R, S] := R\pi^{\mathsf{T}} \cap S\rho^{\mathsf{T}}$. Then for all $z \in Z$ and pairs $u \in X \times Y$ a simple reflection shows that $[R, S]_{z,u}$ iff R_{z,u_1} and S_{z,u_2}.

We also will employ a function *rel* (in the usual mathematical sense) which establishes a Boolean lattice isomorphism between the types $[X \times Y \leftrightarrow \mathbf{1}]$ and $[X \leftrightarrow Y]$. It is defined by $rel(v) = \pi^{\mathsf{T}}(\rho \cap v\mathsf{L}^{\mathsf{T}})$ for all vectors $v : X \times Y \leftrightarrow \mathbf{1}$, where $\pi : X \times Y \leftrightarrow X$ and $\rho : X \times Y \leftrightarrow Y$ are the projection relations of $X \times Y$ and L is a universal vector of type $[Y \leftrightarrow \mathbf{1}]$. Using a component-wise notation, the definition says that for all $x \in X$ and $y \in Y$ we have $v_{\langle x,y \rangle}$ iff $rel(v)_{x,y}$.

3 The Computer Algebra System RelView

RelView (see [1, 5] is a specific purpose Computer Algebra system for the visualization and manipulation of relations. In it all data are represented as relations, which the tool visualizes in different ways. It offers several algorithms for pretty-printing a relation for which source and target coincide as a directed graph. Alternatively, an arbitrary relation may be displayed as a Boolean matrix which is very useful for visual editing and also for discovering structural properties that are not evident from a graphical presentation. Because RelView often works on (very) large data, it uses a very efficient implementation of relations based on reduced ordered binary decision diagrams (see [11]). E.g., a membership-relation $\mathsf{E} : X \leftrightarrow 2^X$ requires $\mathcal{O}(|X|)$ BDD-vertices only. Besides it, we will also use a vector *cardfilter*$(Q) : 2^X \leftrightarrow \mathbf{1}$ that describes the subset $\{Y \in 2^X \mid |Y| < Q\}$ of 2^X. Its BDD-implementation requires $\mathcal{O}(|X|^2)$ vertices.

The main purpose of RelView is the evaluation of relation-algebraic expressions. These are constructed from the relations of its workspace using pre-defined operations and tests and user-defined functions and programs. A RelView-program is much like a function procedure in Modula 2, except that it only uses relations as data type. It starts with a head line containing the program name and the formal parameters. Then the declaration of the local relational domains, functions and variables follows. Declarations of product domains allow to introduce projection relations and pairings. The main part of a program is the body, a while-program over relations. As a program computes a value, it contains a return-clause, which is a relation-algebraic expression whose value after the execution of the body is the result. For instance, the RelView-programs corresponding to the definition of *rel*(v) in Sect. 2 looks as follows:

```
rel(v,S)
  DECL XY = PROD(S*S^,S^*S);
       pi, rho, L
  BEG  pi = p-1(XY); rho = p-2(XY); L = L1n(rho)^
       RETURN pi^*(rho & v*L^)
  END.
```

In this program the declarations introduce `XY` as name for the direct product $X \times Y$ and three variables `pi`, `rho` and `L`. Since polymorphism is not part of the present version of RelView, the type $[X \leftrightarrow Y]$ is made available via a second argument S of this type. Using the product domain `XY`, in the body the projection relations $\pi : X \times Y \leftrightarrow X$ and $\rho : X \times Y \leftrightarrow Y$ and the universal vector $\mathsf{L} : Y \leftrightarrow \mathbf{1}$ are computed and stored in `pi`, `rho` and `L`, respectively. The return-clause is a direct translation of the definition of *rel*(v) into RelView-code.

4 Relational Models of Simple Games

A *cooperative game* is a pair (N, f), where $N = \{1, \dots, n\}$ is the set of *players* and $f : 2^N \rightarrow \mathbb{R}$ is the game's *characteristic function*. A subset C of N is called a *coalition* and $f(C)$ represents its payoff. The game (N, f) is *simple* if $f(C) \in \{0, 1\}$ for all $C \in 2^N$. In this case, a coalition C with $f(C) = 1$ is *winning* and one with $f(C) = 0$ is *losing*. A function from 2^N into $\{0, 1\}$ can be seen as a vector of type $[2^N \leftrightarrow \mathbf{1}]$ such that the function maps $C \in 2^N$ to 1 iff the entry of the vector in the row corresponding to C is 1. Hence, the above definition immediately leads to a first relational model of simple games.

Definition 4.1. *Given a simple game (N, f), a vector $v : 2^N \leftrightarrow \mathbf{1}$ is called its* relational vector-model *if for all $C \in 2^N$ it holds $f(C) = 1$ iff v_C.*

The vector-model $v : 2^N \leftrightarrow \mathbf{1}$ of a simple game (N, f) describes the set $\mathcal{W}$ of the game's winning coalitions as subset of 2^N in the sense of Sect. 2. Characteristic functions are not the only possibility do define simple games. Another natural way to introduce them is to use pairs $(N, \mathcal{W})$ with $N = \{1, \dots, n\}$ again as set of players and $\mathcal{W}$ as subset of 2^N that specifies the set of winning coalitions. If we enumerate the set $\mathcal{W}$ via the columns of a relation as described in Sect. 2, we obtain another relational model of simple games.

Definition 4.2. *Let (N, f) be a simple game and $\mathcal{W}$ denote the set of its winning coalitions. Then $M : N \leftrightarrow \mathcal{W}$ is called the game's* relational membership-model *if for all $k \in N$ and $C \in \mathcal{W}$ it holds $M_{k,C}$ iff $k \in \mathcal{W}$.*

Since the columns of the membership-model enumerate the set of winning coalitions, with regard to the use of RelView this model is in particular appropriate for input and output purposes. Which coalitions are winning can hardly be seen from the vector-model. Concerning the efficiency of algorithms. experiments with RelView have shown that in the case of a high percentage of winning coalitions typically the vector-model is superior and for games with smaller sets of winning coalitions prevalently the membership-model wins.

From Sect. 2 we already know how to get the membership-model from the vector-model. This transformation is described again in the "$\Rightarrow$"-part of the following theorem. In its "$\Leftarrow$"-part it is shown how to obtain the vector-model back from the membership-model. For a proof of this implication, see [3].

Theorem 4.1. *Let (N, f) be a simple game with set $\mathcal{W}$ of winning coalitions and $\mathsf{E} : N \leftrightarrow 2^N$ be a membership-relation. If $v : 2^N \leftrightarrow \mathbf{1}$ is the game's vector-model, then its membership-model is $M := \mathsf{E}\, inj(v)^{\mathsf{T}} : N \leftrightarrow \mathcal{W}$, and if $M : N \leftrightarrow \mathcal{W}$ is the game's membership-model, then its vector-model is $v := syq(\mathsf{E}, M)\mathsf{L} : 2^N \leftrightarrow \mathbf{1}$.*

Assume (N, f) to be a simple game and let $w_1, \ldots, w_n, Q$ be natural numbers. In this context, Q is called the *quota* and w_k is called the *weight* of player k. Then the linear list $[Q; w_1, \ldots, w_n]$ constitutes a *weighted realization* of the game if for all coalitions $C \in 2^N$ it holds that C is winning iff $\sum_{k \in C} w_k \geq Q$. A simple game is called a *weighted voting game* or a *weighted majority game* if it has a weighted realization. This type of simple games plays a prominent role if game theory is used to model and analyze real political situations. See [8, 9], for example.

To obtain a specification of weighted voting games within relation algebra, the players are interpreted as the parties of a parliament and the weights as the number of the parliament seats the parties hold, i.e., in the very same way as in real political life. This leads to the following seat-distribution-model.

Definition 4.3. *If (N, f) is a weighted voting game with the weighted realization $[Q; w_1, \ldots, w_n]$, a relation $D : S \leftrightarrow N$ models the game's* seat-distribution *if it is a mapping (in terms of relation algebra this may be specified by $D^{\mathsf{T}} D \subseteq \mathsf{I}$ and $D\mathsf{L} = \mathsf{L}$; cf. [17]), and for all $k \in N$ it holds $w_k = |\{s \in S \mid D_{s,k}\}|$.*

In the concrete case of real political parties and parliaments, for all $s \in S$ and $k \in N$ the relationship $D_{s,k}$ (or $D(s) = k$ in conventional notation) is interpreted as "seat s is owned by party k". I.e., the weight of a party equals the number of its seats. In Theorem 4.2 we show how to obtain from the seat-distribution-model the vector-model and, hence, via Theorem 4.1 also the membership model.

Theorem 4.2. *Assume (N, f) to be a weighted voting game with the weighted realization $[Q; w_1, \ldots, w_n]$ and let the mapping $D : S \leftrightarrow N$ model its seat-distribution. If $\mathsf{E} : N \leftrightarrow 2^N$ and $\mathsf{E}' : S \leftrightarrow 2^S$ are membership-relations, then the vector-model of the game is $v := syq(D\mathsf{E}, \mathsf{E}')\, \overline{cardfilter(Q)} : 2^N \leftrightarrow \mathbf{1}$.*

Proof. Let c abbreviate *cardfilter*(Q). From the component-wise descriptions of c and of symmetric quotients (see Sect. 3 and 2) we get for all $C \in 2^N$ that

$$\begin{aligned}
v_C &\Leftrightarrow (syq(D\mathsf{E},\mathsf{E}')\,\overline{c}\,)_C \\
&\Leftrightarrow \exists\, X \in 2^S : syq(D\mathsf{E},\mathsf{E}')_{C,X} \wedge \overline{c}_X \\
&\Leftrightarrow \exists\, X \in 2^S : syq(D\mathsf{E},\mathsf{E}')_{C,X} \wedge |X| \geq Q
\end{aligned}$$

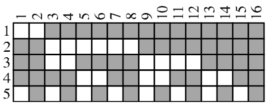
$$\begin{aligned}
&\Leftrightarrow \exists\, X \in 2^S : (\forall\, s \in S : (D\mathsf{E})_{s,C} \leftrightarrow \mathsf{E}'_{s,X}) \wedge |X| \geq Q \\
&\Leftrightarrow \exists\, X \in 2^S : (\forall\, s \in S : (\exists\, k \in N : D_{s,k} \wedge \mathsf{E}_{k,C}) \leftrightarrow \mathsf{E}'_{s,X}) \wedge |X| \geq Q \\
&\Leftrightarrow \exists\, X \in 2^S : (\forall\, s \in S : (\exists\, k \in N : D_{s,k} \wedge k \in C) \leftrightarrow s \in X) \wedge |X| \geq Q \\
&\Leftrightarrow \exists\, X \in 2^S : X = \{s \in S \mid \exists\, k \in C : D_{s,k}\} \wedge |X| \geq Q \\
&\Leftrightarrow \exists\, X \in 2^S : X = \textstyle\bigcup_{k \in C}\{s \in S \mid D_{s,k}\} \wedge |X| \geq Q \\
&\Leftrightarrow |\textstyle\bigcup_{k \in C}\{s \in S \mid D_{s,k}\}| \geq Q \\
&\Leftrightarrow \textstyle\sum_{k \in C} |\{s \in S \mid D_{s,k}\}| \geq Q \\
&\Leftrightarrow \textstyle\sum_{k \in C} w_k \geq Q. \qquad \square
\end{aligned}$$

If the weights are small, then it is easy to obtain the seat-distribution relation D of a weighted voting game using RelView's facilities for interactively constructing relations on the system's screen using command buttons and the mouse. In the case of larger weights such a procedure may become cumbersome. Here it is advantageous to employ the ASCII file-format of RelView in order to load a relation W into the system that consists of the pairs $\langle 1, w_1\rangle, \ldots, \langle n, w_n\rangle$ and then to apply a simple RelView-program that transforms W into D.

In practical political live the number of winning coalitions of a simple game that models a certain situation can grow rapidly with the number of players. Therefore, in the following example we deal with a rather small game.

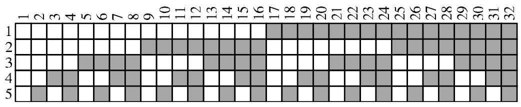
Example 4.1. We consider a weighted voting game with five players, that models the parliament of Germany (the German Bundestag) after the September 2009 election. Its weighted realization is $[312; 239, 146, 93, 76, 68]$, with 312 as quota (for absolute majority; the number of seats of the present German parliament is 622) and then the numbers of seats of the five parties. These are, from left to right, labeled with 1, 2, 3, 4 and 5 and correspond (in the same order) to the parties CDU/CSU, SPD, FDP, Die Linke and Bündnis 90 / Die Grünen. All data are taken from the official web site `www.bundeswahlleiter.de`.

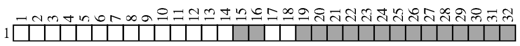
Depicted by RelView, the membership-model $M : N \leftrightarrow \mathcal{W}$ of this game looks as follows; in this Boolean 5×16 matrix a black square means a 1-entry and a white square means a 0-entry. The row labels of M denote the players.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	□	□	■	■	■	■	■	■	■	■	■	■	■	■	■	■
2	■	■	□	□	□	□	□	□	■	■	■	■	■	■	■	■
3	■	■	□	□	■	■	■	■	□	□	□	□	■	■	■	■
4	■	■	■	■	□	□	■	■	□	□	■	■	□	□	■	■
5	□	■	□	■	□	■	□	■	□	■	□	■	□	■	□	■

E.g., column 1 represents the coalition consisting of SPD, FDP and Die Linke and column 5 represents the coalition of CDU/CSU and FDP that forms the present German government. If we transform M into the vector-model vector $v : 2^N \leftrightarrow \mathbf{1}$, we obtain in RelView a Boolean 32×1 matrix in which exactly 16 entries are 1. The following two pictures show the membership-relation $\mathsf{E} : N \leftrightarrow 2^N$ and, below it, the transpose of v, that is, the row vector $v^{\mathsf{T}} : \mathbf{1} \leftrightarrow 2^N$.





The 32 columns of E describe all coalitions. A comparison of the pictures (here the transposition of v is adequate) shows that the 1-entries of the vector-model precisely designate the columns of E that belong to the membership-model.

5 Three Applications Concerning Power of Players

As already mentioned in the introduction, simple games are very useful for the comparison and measurement of power in decision-making processes. In this section we consider three different possibilities to describe power and show how they can be specified using relation algebra and the membership-model. We also demonstrate how RelView can used to evaluate the relation-algebraic specifications. Our starting point is the following notion, first introduced in [12].

Definition 5.1. *A winning coalition of a simple game is* minimal winning *if every proper subset is losing.*

In the following theorem we show how to specify the set $\mathcal{W}^\diamond$ of all minimal winning coalitions as a subset of the set $\mathcal{W}$ of all winning ones via a vector of type $[\mathcal{W} \leftrightarrow \mathbf{1}]$. An immediate consequence is the column-wise enumeration of $\mathcal{W}^\diamond$.

Theorem 5.1. *If* $M : N \leftrightarrow \mathcal{W}$ *is the membership-model of a simple game, then* $m := \overline{(\bar{\mathsf{I}} \cap \overline{\overline{M}^\mathsf{T} M})\mathsf{L}} : \mathcal{W} \leftrightarrow \mathbf{1}$ *describes the minimal winning coalitions* $\mathcal{W}^\diamond$ *as subset of* $\mathcal{W}$ *and* $M^\diamond := M\mathit{inj}(m)^\mathsf{T} : N \leftrightarrow \mathcal{W}^\diamond$ *column-wisely enumerates* $\mathcal{W}^\diamond$.

Proof. In the specification of the vector m the type of L is $[\mathcal{W} \leftrightarrow \mathbf{1}]$ and the type of I is $[\mathcal{W} \leftrightarrow \mathcal{W}]$. Now, we obtain for all $C \in \mathcal{W}$ that

$$\begin{aligned} m_C &\Leftrightarrow \overline{((\bar{\mathsf{I}} \cap \overline{\overline{M}^\mathsf{T} M})\mathsf{L})}_C \\ &\Leftrightarrow \neg\exists\, D \in \mathcal{W} : \bar{\mathsf{I}}_{C,D} \wedge (\overline{\overline{M}^\mathsf{T} M})_{C,D} \wedge \mathsf{L}_D \\ &\Leftrightarrow \neg\exists\, D \in \mathcal{W} : C \neq D \wedge (\overline{\overline{M}^\mathsf{T} M})_{C,D} \\ &\Leftrightarrow \neg\exists\, D \in \mathcal{W} : C \neq D \wedge \neg\exists\, i \in N : \overline{M}_{i,C} \wedge M_{i,D} \\ &\Leftrightarrow \neg\exists\, D \in \mathcal{W} : C \neq D \wedge \forall\, i \in N : M_{i,D} \rightarrow M_{i,C} \\ &\Leftrightarrow \neg\exists\, D \in \mathcal{W} : C \neq D \wedge \forall\, i \in N : i \in D \rightarrow i \in C \\ &\Leftrightarrow \neg\exists\, D \in \mathcal{W} : D \subset C. \end{aligned}$$

This shows that m describes $\mathcal{W}^\diamond$ as subset of $\mathcal{W}$. The second claim follows from the fact that for all players $k \in N$ and minimal winning coalitions $C \in M^\diamond$ we have (due to the component-wise description of embedding-relations in Sect. 2)

$$\begin{aligned} M^\diamond_{k,C} &\Leftrightarrow (M\mathit{inj}(m)^\mathsf{T})_{k,C} \\ &\Leftrightarrow \exists\, D \in \mathcal{W} : M_{k,D} \wedge \mathit{inj}(m)_{C,D} \\ &\Leftrightarrow \exists\, D \in \mathcal{W} : k \in D \wedge C = D \\ &\Leftrightarrow k \in C. \end{aligned}$$

□

When investigating the power of players, one possibility is to identify some key players having different strength and then to classify the set of players accordingly. In this paper we consider the following key players.

Definition 5.2. *Player k of a simple game is a* dictator *if $\{k\}$ is the only minimal winning coalition, a* veto player *if it belongs to all minimal winning coalitions, and a* dummy player *if it does not belong to any minimal winning coalition.*

A dictator is the most powerful player of a simple game. He can enforce any decision without help of the other players. There exists at most one dictator. A veto player is needed to win, but he cannot win on his own. If, however, any coalition that contains this player is winning, then he is a dictator. Finally, a dummy player is a player without any power. Next, we present relation-algebraic specifications of these key players.

Theorem 5.2. *Let the relation $M^\diamond : N \leftrightarrow \mathcal{W}^\diamond$ be as in Theorem 5.1 If we define $a := syq(\mathsf{I}, M^\diamond)\,\overline{\overline{\mathsf{I}}\,\mathsf{L}} : N \leftrightarrow \mathbf{1}$, $b := \overline{\overline{M^\diamond}\,\mathsf{L}} : N \leftrightarrow \mathbf{1}$ and $c := \overline{M^\diamond \mathsf{L}} : N \leftrightarrow \mathbf{1}$, then for all players $k \in N$ it holds that it is a dictator iff a_k, a veto player iff b_k, and a dummy player iff c_k.*

Proof. Notice, that in the specification of a besides $\mathsf{L} : \mathcal{W}^\diamond \leftrightarrow \mathbf{1}$ two different identity relations appear, viz. $\mathsf{I} : N \leftrightarrow N$ in $syq(\mathsf{I}, M^\diamond)$ and $\mathsf{I} : \mathcal{W}^\diamond \leftrightarrow \mathcal{W}^\diamond$ in $\overline{\overline{\mathsf{I}}\,\mathsf{L}}$. Now, the first claim follows from the calculation

$$\begin{aligned}
a_k &\Leftrightarrow (syq(\mathsf{I}, M^\diamond)\,\overline{\overline{\mathsf{I}}\,\mathsf{L}}\,)_k \\
&\Leftrightarrow \exists\, C \in \mathcal{W}^\diamond : syq(\mathsf{I}, M^\diamond)_{k,C} \wedge (\,\overline{\overline{\mathsf{I}}\,\mathsf{L}}\,)_C \\
&\Leftrightarrow \exists\, C \in \mathcal{W}^\diamond : (\forall\, i \in N : \mathsf{I}_{i,k} \leftrightarrow M^\diamond_{i,C}) \wedge \neg\exists\, D \in \mathcal{W}^\diamond : \overline{\mathsf{I}}_{C,D} \wedge \mathsf{L}_D \\
&\Leftrightarrow \exists\, C \in \mathcal{W}^\diamond : (\forall\, i \in N : i = k \leftrightarrow i \in C) \wedge \neg\exists\, D \in \mathcal{W}^\diamond : C \neq D \\
&\Leftrightarrow \exists\, C \in \mathcal{W}^\diamond : C = \{k\} \wedge \neg\exists\, D \in \mathcal{W}^\diamond : C \neq D \\
&\Leftrightarrow k \text{ is a dictator}
\end{aligned}$$

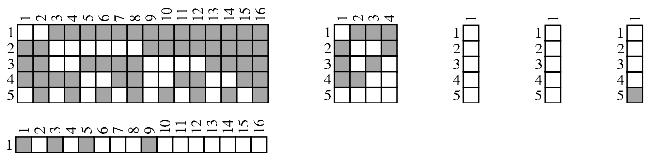
that uses the component-wise description of symmetric quotients given in Sect. 2. The relation L in the specification of b has type $[\mathcal{W}^\diamond \leftrightarrow \mathbf{1}]$, too, and

$$\begin{aligned}
b_k &\Leftrightarrow (\,\overline{\overline{M^\diamond}\,\mathsf{L}}\,)_k \\
&\Leftrightarrow \neg\exists\, C \in \mathcal{W}^\diamond : \neg M^\diamond_{k,C} \wedge \mathsf{L}_C \\
&\Leftrightarrow \forall\, C \in \mathcal{W}^\diamond : M^\diamond_{k,C} \\
&\Leftrightarrow \forall\, C \in \mathcal{W}^\diamond : k \in C \\
&\Leftrightarrow k \text{ is a veto player}
\end{aligned}$$

is a proof of the second claim. The third claim can be shown in a similar way. □

We have translated the relation-algebraic specifications given in the last two theorems into RelView-code. Then we have applied the RelView-programs to the relations of the game of Example 4.1. Here are the results.

Example 5.1. For the parliament of Germany RelView computed the results given in the following pictures. The two leftmost pictures show again the membership-model and, below it, the transpose of the vector $m : \mathcal{W} \leftrightarrow \mathbf{1}$.



The four 1-entries of the row vector $m^{\mathsf{T}} : \mathbf{1} \leftrightarrow \mathcal{W}$ precisely designate those rows of M which represent minimal winning coalitions. If we assemble these four rows in form of a new matrix, we obtain the column-wise enumeration $M^{\diamond} : N \leftrightarrow \mathcal{W}^{\diamond}$ given in the next picture as Boolean 5×4 matrix. The remaining three vectors of type $[N \leftrightarrow \mathbf{1}]$ describe, from left to right, the sets of dictators, veto players and dummy players, respectively. Hence, in the present German parliament there exists neither a dictator nor a veto player. But there is a dummy player, viz. Bündnis 90 / Die Grünen.

Another means for measuring power is the desirability relation. It directly compares two players with regard to the strength of forming winning coalitions.

Definition 5.3. *Let (N, f) be a simple game with $\mathcal{W}$ as set of winning coalitions. Then $j \in N$ is* at least as desirable as *$i \in N$, denoted by $i \preceq_D j$, if for all $C \in 2^N$ from $i, j \notin C$ and $C \cup \{i\} \in \mathcal{W}$ it follows $C \cup \{j\} \in \mathcal{W}$.*

In words $i \preceq_D j$ says that if i can form a winning coalition with some further players, then j can do either. The desirability relation $\preceq_D$ is a pre-order on the players. With its help a lot of problems on simple games easily can be solved. E.g., $\preceq_D$ is linear iff the game is swap-robust, which means that a one-for-one exchange of players between two winning coalitions leaves at least one of them winning. Next, we show how to specify $i \preceq_D j$ by means of relation algebra.

Theorem 5.3. *Let $M : N \leftrightarrow \mathcal{W}$ be the membership model of a simple game and assume the players $i, j \in N$ to be described by the points $p, q : N \leftrightarrow \mathbf{1}$, respectively. If $\underline{\mathsf{E} : N \leftrightarrow 2^N}$ is a membership-relation, then $i \preceq_D j$ is equivalent to the inclusion $\overline{\mathsf{E}^{\mathsf{T}} p} \cap \overline{\mathsf{E}^{\mathsf{T}} q} \cap syq(\mathsf{E} \cup p\mathsf{L}, M)\mathsf{L} \subseteq syq(\mathsf{E} \cup q\mathsf{L}, M)\mathsf{L}$.*

Proof. Since the point p describes player i, we have for all $C \in 2^N$ that

$$(\,\overline{\mathsf{E}^{\mathsf{T}} p}\,)_C \;\Leftrightarrow\; \neg\exists\, k \in N : \mathsf{E}_{k,C} \wedge p_k \;\Leftrightarrow\; \neg\exists\, k \in N : k \in C \wedge k = i \;\Leftrightarrow\; i \notin C$$

(for the second step, see Sect. 2) and also that (the L in $p\mathsf{L}$ has type $[\mathbf{1} \leftrightarrow 2^N]$ and the L composed from the right to the symmetric quotient has type $[\mathcal{W} \leftrightarrow \mathbf{1}]$)

$$\begin{aligned}
(syq(\mathsf{E} \cup p\mathsf{L}, M)\mathsf{L})_C &\Leftrightarrow \exists\, D \in \mathcal{W} : syq(\mathsf{E} \cup p\mathsf{L}, M)_{C,D} \wedge \mathsf{L}_D \\
&\Leftrightarrow \exists\, D \in \mathcal{W} : \forall\, k \in N : (\mathsf{E}_{k,C} \vee p_k) \leftrightarrow M_{k,D} \\
&\Leftrightarrow \exists\, D \in \mathcal{W} : \forall\, k \in N : (k \in C \vee k = i) \leftrightarrow k \in D \\
&\Leftrightarrow \exists\, D \in \mathcal{W} : C \cup \{i\} = D \\
&\Leftrightarrow C \cup \{i\} \in \mathcal{W}.
\end{aligned}$$

The latter derivation employs the component-wise description of symmetric quotients given in Sect. 2. In the same way for all $C \in 2^N$ we get the equivalence of

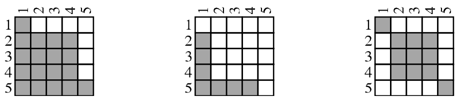
$(\overline{\mathsf{E}^\mathsf{T} q})_C$ and $j \notin C$ and of $(syq(\mathsf{E} \cup q\mathsf{L}, M)\mathsf{L})_C$ and $C \cup \{j\} \in \mathcal{W}$ from the fact that the point q describes player j. Using the just shown equivalences in

$$\begin{aligned} i \preceq_D j &\Leftrightarrow \forall C \in 2^N : i \notin C \wedge j \notin C \wedge C \cup \{i\} \in \mathcal{W} \rightarrow C \cup \{j\} \in \mathcal{W} \\ &\Leftrightarrow \forall C \in 2^N : (\overline{\mathsf{E}^\mathsf{T} p} \cap \overline{\mathsf{E}^\mathsf{T} q} \cap syq(\mathsf{E} \cup p\mathsf{L}, M)\mathsf{L})_C \rightarrow (syq(\mathsf{E} \cup q\mathsf{L}, M)\mathsf{L})_C \\ &\Leftrightarrow \overline{\mathsf{E}^\mathsf{T} p} \cap \overline{\mathsf{E}^\mathsf{T} q} \cap syq(\mathsf{E} \cup p\mathsf{L}, M)\mathsf{L} \subseteq syq(\mathsf{E} \cup q\mathsf{L}, M)\mathsf{L}, \end{aligned}$$

we obtain the claimed result. □

As a consequence we get the desirability relation $\preceq_D : N \leftrightarrow N$ as the union of all compositions pq^T, where p and q range over all points from $[N \leftrightarrow \mathbf{1}]$ such that the right-hand side of Theorem 5.3 holds. This algorithm easily can be implemented in RelView via two nested loops. From $\preceq_D$ two further relations on players are derived, viz. the *more desirability relation* $\prec_D$ as intersection of $\preceq_D$ and the complement of its transpose, and the *equal desirability relation* $\equiv_D$ as intersection of $\preceq_D$ and its transpose.

Example 5.2. Let us consider again the simple game introduced in Example 4.1. The following pictures show, from left to right, the relations $\preceq_D$, $\prec_D$ and $\equiv_D$ as computed by RelView:



Even though the players 2, 3 and 4 have different weights, they are equally desirable. Such players are also called *symmetric*.

Indices are a third means to measure power. A well-established index goes back to [2]. Under the assumption that all coalitions are equally likely and that each player votes 'yes' or 'no' with probability $\frac{1}{2}$, the power of k is defined as the probability that k is decisive for the outcome. If all indices are normalized in such a way that their sum equals to 1, this leads to the following specification.

Definition 5.4. *Let (N, f) be a simple game with set of winning coalitions $\mathcal{W}$. Then the pair $\langle k, C\rangle \in N \times \mathcal{W}$ is called a* swing *if $k \in C$ and $C \setminus \{k\} \notin \mathcal{W}$. The* Banzhaf power index *of $k \in N$ is defined as $\frac{\eta_k}{\eta}$, where η_k is the number of swings with first component k and η is the number of all swings.*

If we have a relation $B : N \leftrightarrow \mathcal{W}$ at hand that precisely contains the swings of a simple game, then η_k equals the number of 1-entries of row k of B and η equals the total number of 1-entries of B. The next theorem presents a relation-algebraic specification of this relation.

Theorem 5.4. *Assume $M : N \leftrightarrow \mathcal{W}$ to be the membership model of a simple game. If we define $B := M \cap \overline{rel(syq([\overline{\mathsf{I}}, M], M)\mathsf{L})} : N \leftrightarrow \mathcal{W}$, then for all $k \in N$ and $C \in \mathcal{W}$ the pair $\langle k, C\rangle$ is a swing iff $B_{k,C}$.*

Proof. Notice that in the definition of B the relations I and L have the types $[N \leftrightarrow N]$ and $[\mathcal{W} \leftrightarrow \mathbf{1}]$, respectively. We start the proof with

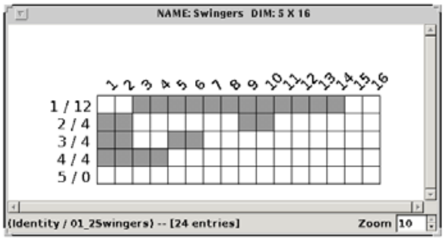
$$\begin{aligned}(syq([\overline{\mathsf{I}}, M], M)\mathsf{L})_{\langle k,C\rangle} &\Leftrightarrow \exists D \in \mathcal{W} : syq([\overline{\mathsf{I}}, M], M)_{\langle k,C\rangle,D} \wedge \mathsf{L}_D \\ &\Leftrightarrow \exists D \in \mathcal{W} : \forall i \in N : [\overline{\mathsf{I}}, M]_{i,\langle k,C\rangle} \leftrightarrow M_{i,D} \\ &\Leftrightarrow \exists D \in \mathcal{W} : \forall i \in N : (\overline{\mathsf{I}}_{i,k} \wedge M_{i,C}) \leftrightarrow M_{i,D} \\ &\Leftrightarrow \exists D \in \mathcal{W} : \forall i \in N : (i \neq k \wedge i \in C) \leftrightarrow i \in D \\ &\Leftrightarrow \exists D \in \mathcal{W} : C \setminus \{k\} = D \\ &\Leftrightarrow C \setminus \{k\} \in \mathcal{W},\end{aligned}$$

using the component-wise descriptions of symmetric quotients and pairings given in Sect. 2. If we combine this result with the component-wise description of the function *rel* given in Sect. 2, too, we can complete the proof by

$$\begin{aligned}B_{k,c} &\Leftrightarrow (M \cap \overline{rel(syq([\overline{\mathsf{I}}, M], M)\mathsf{L})}\,)_{k,C} \\ &\Leftrightarrow M_{k,C} \wedge \neg rel(syq([\overline{\mathsf{I}}, M], M)\mathsf{L})_{k,C} \\ &\Leftrightarrow k \in C \wedge \neg(syq([\overline{\mathsf{I}}, M], M)\mathsf{L})_{\langle k,C\rangle} \\ &\Leftrightarrow k \in C \wedge C \setminus \{k\} \notin \mathcal{W} \\ &\Leftrightarrow \langle k, C\rangle \text{ is a swing .} \qquad \square\end{aligned}$$

If the RelView tool depicts B as a Boolean matrix in the relation-window, then in the window's status bar the number of 1-entries of B is shown. Furthermore, it is able to mark the rows and columns of B for explanatory purposes. So far, we have only shown the possibility to attach consecutive row and column numbers to relations. But also the numbers of 1-entries can be attached as labels. This immediately allows to compute Banzhaf power indices using the tool.

Example 5.3. The following picture shows the relation-window of RelView, where the swing-relation $B : N \leftrightarrow \mathcal{W}$ for the parliament of Germany is depicted.


NAME: Swingers DIM: 5 X 16

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1 / 12			▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨		
2 / 4	▨	▨							▨	▨						
3 / 4	▨	▨			▨	▨										
4 / 4	▨	▨	▨	▨												
5 / 0																

(Identity / 01_2Swingers) -- [24 entries]

From the message `[24 entries]` at the bottom of this window and the second components of the row labels we obtain the following Banzhaf power indices: CDU/CSU $\frac{12}{24}$, SPD $\frac{4}{24}$, FDP $\frac{4}{24}$, Die Linke $\frac{4}{24}$ and Bündnis 90 / Die Grünen $\frac{0}{24}$.

6 Conclusions

In spite of the fact that RelView implements relations very efficiently, due of its general approach it cannot compete with special algorithms tailored for hard game-theoretic problems. Such algorithms even can tackle games like the US Federal System game with 537 players. We believe that the real attraction of RelView lies in its flexibility: New types and properties of games are introduced all the time and RelView proved to be ideal for experimenting with new concepts while avoiding unnecessary overhead. By now, systematic experiments are accepted as means for obtaining new scientific insights and results, and tools for this purpose become increasingly important as one proceeds in investigations.

In those cases where the present RelView tool is not effective, all hope is not lost. By our experiments we have noticed that in many cases the BDDs of the results are relatively small. This led to the insight that BDDs are an excellent means for solving efficiently game-theoretic problems, especially if they are manipulable in full generality and not only indirectly via the language of RelView. Consequently, the current direction in the development of RelView is to make it more extensible by expanding its interface in such a way that it is possible to outsource program logic into plug-ins. By specific game-theoretic plug-ins we hope to be able to treat successfully in the future also large problems.

References

[1] http://www.informatik.uni-kiel.de/~relview.shtml
[2] Banzhaf, J.F.: Weighted voting doesn't work: A mathematical analysis. Rutgers Law Review 19, 317–343 (1965)
[3] Berghammer, R., Bolus, S., Rusinowska, A., de Swart, H.: A relation-algebraic approach to simple games. Europ. J. Operat. Res. 210, 68–80 (2011)
[4] Berghammer, R., Braßel, B.: Computing and visualizing closure objects using relation algebra and relView. In: Gerdt, V.P., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2009. LNCS, vol. 5743, pp. 29–44. Springer, Heidelberg (2009)
[5] Berghammer, R., Neumann, F.: RelView – An OBDD-Based Computer Algebra System for Relations. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V., et al. (eds.) CASC 2005. LNCS, vol. 3718, pp. 40–51. Springer, Heidelberg (2005)
[6] Berghammer, R., Rusinowska, A., de Swart, H.: Applying relational algebra and RelView to coalition formation. Europ. J. Operat. Res. 178, 530–542 (2007)
[7] Berghammer, R., Rusinowska, A., de Swart, H.: An interdisciplinary approach to coalition formation. Europ. J. Operat. Res. 195, 487–496 (2009)
[8] van Deemen, A.: Dominant players and minimum size coalitions. Europ. J. Polit. Res. 17, 313–332 (1989)
[9] van Deemen, A.: Coalition formation in centralized policy games. J. Theoret. Polit. 3, 139–161 (1991)
[10] Elkind, E., Goldberg, L.A., Goldberg, P.W., Wooldridge, M.: On the computational complexity of weighted voting games. Ann. Math. Artif. Intell. 56, 109–131 (2009)
[11] Leoniuk, B.: ROBDD-basierte Implementierung von Relationen und relationalen Operationen mit Anwendungen. Diss., Univ. Kiel (2001)
[12] von Neumann, J., Morgenstern, O.: Theory of games and economic behaviour. Princeton University Press, Princeton (1944)
[13] Peleg, B., Sudhölter, P.: Introduction to the theory of cooperative games. Springer, Heidelberg (2003)
[14] Peters, H.: Game theory: A Multi-leveled approach. Springer, Heidelberg (2008)
[15] Prasad, K., Kelly, J.S.: NP-completeness of some problems concerning voting games. Int. J. Game Theory 19, 1–9 (1990)
[16] van Roozendaal, P.: Centre parties and coalition cabinet formations: a game theoretic approach. Europ. J. Polit. Res. 18, 325–348 (1990)
[17] Schmidt, G., Ströhlein, T.: Relations and graphs. Springer, Heidelberg (1993)
[18] Taylor, A.D.: Mathematics and politics. Springer, Heidelberg (1995)

On the Regularity Property of Differential Polynomials Modulo Regular Differential Chains*

François Boulier[1], François Lemaire[1], and Alexandre Sedoglavic[1]

Université Lille I, LIFL, 59655 Villeneuve d'Ascq, France
Francois.Boulier@univ-lille1.fr,
{Francois.Lemaire,Alexandre.Sedoglavic}@lifl.fr

Abstract. This paper provides an algorithm which computes the normal form of a rational differential fraction modulo a regular differential chain if, and only if, this normal form exists. A regularity test for polynomials modulo regular chains is revisited in the nondifferential setting and lifted to differential algebra. A new characterization of regular chains is provided.

1 Introduction

This paper is concerned by methods for deciding whether a polynomial f (multivariate, over a field, say, $\mathbb{Q}$) is regular (i.e. not a zerodivisor) modulo a polynomial ideal defined by a regular chain C, which is a set of polynomials. For casual readers, this regularity property may seem quite exotic, compared to (say) the membership property to polynomial ideals. It is however very important and is pretty much related to the problem of computing the solutions of the system of polynomial equations $C = 0$. For instance, if f is proved to be a zerodivisor, then a factorization of some element of C is exhibited, which permits to split the set of equations to be solved, into two simpler sets. Moreover, as we shall see, regularity testing is strongly related to the problem of computing normal forms of polynomials modulo the ideal defined by the regular chain C, which are canonical representatives of the residue class ring defined by C. These comments are stated in the nondifferential case, for simplicity. However, they all have a counterpart for polynomial differential equations, i.e. in differential algebra.

Normal forms have many applications. In differential algebra, they make it easier to compute power series solutions, as pointed out in [2]. In both nondifferential and differential algebra, they permit to search linear dependencies between rational fractions modulo regular chains, by searching linear dependencies between their normal forms, modulo "nothing" (one of the key ideas of [10],

* This work has benefited from the support of the French ANR (decision number ANR-2010-BLAN-0109-03). It benefited also of many exchanges with Markus Rosenkranz and Georg Regensburger.

developed in the differential case in [3]). The very same principle, applied on the derivatives of rational differential fractions, may help to find first integrals.

The motivation for this paper comes from very fruitful remarks of a few reviewers of [2]. In that paper, a normal form algorithm is given for rational differential fractions modulo regular differential chains [2, Figure 2, Algorithm NF]. This normal form algorithm ultimately relies on an algorithm for computing the inverse of a nondifferential polynomial, modulo the ideal defined by a nondifferential regular chain. However, the algorithm provided in [2] may fail to compute the inverse, even if the inverse does exist [2, last comments of section 4]. A few reviewers of [2] then asked if it is possible to provide a complete algorithm, based on regular chains related methods[1], for computing normal forms if, and only if, these normal forms exist. In this paper, we provide the following results:

1. a complete normal form algorithm (Figure 1 and Theorem 4) ;
2. a revisited algorithmic characterization of the polynomials which are regular modulo the ideal defined by a nondifferential regular chain (Theorem 1) and its generalization in differential algebra (Theorem 3) ;
3. a new characterization of regular chains (Theorem 2).

The first result is an answer to the reviewers request. The second one improves former results of [18] and [8] in the nondifferential setting. It completes the proof of [14, Theorem 2.4] and extends this theorem in differential algebra. The third one permits to generalize [14, Theorem 2.4] and [1, Theorem 6.1].

2 Basics of Differential Algebra

The reference books are [16] and [13]. A *differential ring* R is a ring endowed with finitely many, say m, abstract *derivations* $\delta_1, \ldots, \delta_m$ i.e. unary operations which satisfy the following axioms, for each derivation δ:

$$\delta(a+b) = \delta(a) + \delta(b), \quad \delta(a\,b) = \delta(a)\,b + a\delta(b), \qquad (\forall\ a,\, b \in R)$$

and which are assumed to commute pairwise. This paper is mostly concerned with a differential polynomial ring R in n *differential indeterminates* $u_1, \ldots, u_n$ with coefficients in a commutative differential field K of characteristic zero, say $K = \mathbb{Q}$. Letting $U = \{u_1, \ldots, u_n\}$, one denotes $R = K\{U\}$, following Ritt and Kolchin. The set of derivations generates a commutative monoid w.r.t. the composition operation. It is denoted:

$$\Theta = \{\delta_1^{a_1} \cdots \delta_m^{a_m} \mid a_1, \ldots, a_m \in \mathbb{N}\}$$

where $\mathbb{N}$ stands for the set of the nonnegative integers. The elements of Θ are the *derivation operators*. The monoid Θ acts multiplicatively on U, giving the infinite set ΘU of the *derivatives*.

[1] Observe that, in principle, each required inverse could be easily obtained by using Rabinowitsch's trick and by running the Buchberger algorithm. However, Gröbner bases are not regular chains related methods. Moreover, the method could be costly.

If A is a finite subset of R, one denotes (A) the smallest ideal containing A w.r.t. the inclusion relation and $[A]$ the smallest differential ideal containing A. Let $\mathfrak{A}$ be an ideal and $S = \{s_1, \ldots, s_t\}$ be a finite subset of R, not containing zero. Then

$$\mathfrak{A} : S^{\infty} = \{p \in R \mid \exists\ a_1, \ldots, a_t \in \mathbb{N},\ s_1^{a_1} \cdots s_t^{a_t}\, p \in \mathfrak{A}\}$$

is called the *saturation* of $\mathfrak{A}$ by the multiplicative family generated by S. The saturation of a (differential) ideal is a (differential) ideal [13, chapter I, Corollary to Lemma 1].

Fix a *ranking*, i.e. a total ordering over ΘU satisfying some properties [13, chapter I, section 8]. Consider some differential polynomial $p \notin K$. The highest derivative v w.r.t. the ranking such that $\deg(p, v) > 0$ is called the *leading derivative* of p. It is denoted $\mathrm{ld}\, p$. The leading coefficient of p w.r.t. v is called the *initial* of p. The differential polynomial $\partial p / \partial v$ is called the *separant* of p. If C is a finite subset of $R \setminus K$ then I_C denotes its set of initials, S_C denotes its set of separants and $H_C = I_C \cup S_C$.

A differential polynomial q is said to be *partially reduced* w.r.t. p if it does not depend on any proper derivative of the leading derivative v of p. It is said to be *reduced* w.r.t. p if it is partially reduced w.r.t. p and $\deg(q, v) < \deg(p, v)$. A set of differential polynomials of $R \setminus K$ is said to be *autoreduced* if its elements are pairwise reduced. Autoreduced sets are necessarily finite [13, chapter I, section 9]. To each autoreduced set C, one may associate the set $L = \mathrm{ld}\, C$ of the leading derivatives of C and the set $N = \Theta U \setminus \Theta L$ of the derivatives which are not derivatives of any element of L (the derivatives “under the stairs” defined by C).

The following definition is borrowed from [2, Definition 3.1].

Definition 1. *The set $C = \{c_1, \ldots, c_n\}$ is a* regular differential chain *if it satisfies the following conditions:*

a *the elements of C are pairwise partially reduced and have distinct leading derivatives ;*

b *for each $2 \le k \le n$, the initial i_k of c_k is regular in $K[N \cup L]/(c_1, \ldots, c_{k-1}) : (i_1 \cdots i_{k-1})^{\infty}$;*

c *for each $1 \le k \le n$, the separant s_k of c_k is regular in $K[N \cup L]/(c_1, \ldots, c_k) : (i_1 \cdots i_k)^{\infty}$;*

d *for any pair $\{c_k, c_\ell\}$ of elements of C, whose leading derivatives $\theta_k u$ and $\theta_\ell u$ are derivatives of some same differential indeterminate u, the Δ-polynomial*

$$\Delta(c_k, c_\ell) = s_\ell \frac{\theta_{k\ell}}{\theta_k} c_k - s_k \frac{\theta_{k\ell}}{\theta_\ell} c_\ell\,,$$

where $\theta_{k\ell}$ denotes the least common multiple of θ_k and θ_ℓ, is reduced to zero by C, using Ritt's reduction algorithm [13, chapter I, section 9].

Triangularity plus condition **b** is the *regular chain* condition of [1]. Autoreduced regular differential chains are the same objects as Ritt characteristic sets. See [2, Proposition 3.2].

3 The Normal Form of a Rational Differential Fraction

All the results of this section are borrowed from [2]. Let C be a regular differential chain of R, defining a differential ideal $\mathfrak{A} = [C] : H_C^\infty$. Let $L = \operatorname{ld} C$ and $N = \Theta U \setminus \Theta L$. The normal form of a rational differential fraction is introduced in [2, Definition 5.1 and Proposition 5.2], recalled below.

Definition 2. *Let a/b be a rational differential fraction, with b regular modulo $\mathfrak{A}$. A* normal form *of a/b modulo C is any rational differential fraction f/g such that*

1 *f is reduced with respect to C ;*
2 *g belongs to $K[N]$ (and is thus regular modulo $\mathfrak{A}$),*
3 *a/b and f/g are equivalent modulo $\mathfrak{A}$ (in the sense that $a\,g - b\,f \in \mathfrak{A}$).*

Proposition 1. *Let a/b be a rational differential fraction, with b regular modulo $\mathfrak{A}$. The normal form f/g of a/b exists and is unique. In particular,*

4 *a belongs to $\mathfrak{A}$ if and only if its normal form is zero ;*
5 *f/g is a canonical representative of the residue class of a/b in the total fraction ring of $R/\mathfrak{A}$.*

Moreover,

6 *each irreducible factor of g divides the denominator of an inverse of b, or of some initial or separant of C .*

Recall that the normal form algorithm relies on the computation of inverses of differential polynomials, defined below.

Definition 3. *Let f be a nonzero differential polynomial of R. An* inverse *of f is any fraction p/q of nonzero differential polynomials such that $p \in K[N \cup L]$ and $q \in K[N]$ and $f\,p \equiv q \mod \mathfrak{A}$.*

4 On the Regularity Property of Polynomials

Though this section only addresses algebraic (i.e. nondifferential) questions, we state it with the terminology of the differential algebra. Consider a triangular set C in the polynomial ring $S = K[N \cup L]$. The ideal defined by C, in S, is $\mathfrak{B} = (C) : I_C^\infty$. Assume that $C = \{c_1, \ldots, c_n\}$, that the leading derivative (leading variable) of c_k is x_k and that $x_1 < \cdots < x_n$. It is possible to define the *iterated resultant* of any polynomial f w.r.t. C as follows. See [18, Definition 5.2] or [19, Definition 1.2]. See [8, Definition 4] or [15, Definition 1] for a close definition. See [7] for a definition in a more general setting.

$$\operatorname{res}(f,\, C) = \operatorname{res}(\cdots \operatorname{res}(f,\, c_n,\, x_n), \ldots, c_1,\, x_1) \tag{1}$$

where $\operatorname{res}(f,\, c_k,\, x_k)$ denotes the usual resultant of f and c_k w.r.t. x_k. The next lemma is borrowed from [18, Lemma 5.2]. Together with the two following ones, it prepares the proof of Theorem 1.

Lemma 1. *Let f be any polynomial. There exist polynomials p, $q_1, \ldots, q_n$ such that*

$$p\, f = q_1\, c_1 + q_2\, c_2 + \cdots + q_n\, c_n + \operatorname{res}(f,\, C)\,. \tag{2}$$

Proof. By [17, section 5.8, identity (5.21)], there exist two polynomials p_n and g_n such that

$$p_n\, f = g_n\, c_n + \operatorname{res}(f,\, c_n,\, x_n)\,. \tag{3}$$

There exist two polynomials p_{n-1} and g_{n-1} such that

$$p_{n-1}\, \operatorname{res}(f,\, c_n,\, x_n) = g_{n-1}\, c_{n-1} + \operatorname{res}(\operatorname{res}(f,\, c_n,\, x_n),\, c_{n-1},\, x_{n-1}) \tag{4}$$

hence such that

$$p_{n-1}\, p_n\, f = p_{n-1}\, g_n\, c_n + g_{n-1}\, c_{n-1} + \operatorname{res}(\operatorname{res}(f,\, c_n,\, x_n),\, c_{n-1},\, x_{n-1})\,. \tag{5}$$

Continuing, we obtain (2).

The two following lemmas are easy.

Lemma 2. *Let f, g be two polynomials. Then* $\operatorname{res}(f\, g,\, C) = \operatorname{res}(f,\, C)\, \operatorname{res}(g,\, C)$*.*

Proof. By induction on the number n of elements of C. If $n = 1$ then the lemma is the well-known multiplicativity property of resultants. See [9, section 3.1, exercises 3, 8 and 10] or [7, page 349]. If $n > 1$, assume inductively that the lemma holds for $C_{n-1} = \{c_1, \ldots, c_{n-1}\}$. Then $\operatorname{res}(f\, g,\, C) = \operatorname{res}(\operatorname{res}(f\, g,\, c_n,\, x_n),\, C_{n-1})$. Then, by the induction hypothesis and the multiplicativity property of resultants, $\operatorname{res}(f\, g,\, C)$ is equal to $\operatorname{res}(\operatorname{res}(f,\, c_n,\, x_n),\, C_{n-1})\, \operatorname{res}(\operatorname{res}(g,\, c_n,\, x_n),\, C_{n-1})$, which, in turn, is equal to $\operatorname{res}(f,\, C)\, \operatorname{res}(g,\, C)$.

Lemma 3. *Let $2 \leq k < n$ be an index and f be any polynomial such that $\deg(f, x_\ell) = 0$, for $k < \ell \leq n$. There exists a positive integer m such that $\operatorname{res}(f,\, C) = \operatorname{res}(f,\, C_k)^m$.*

Proof. It is an easy consequence of Lemma 2 and of the fact that, if $\deg(f, x) = 0$ and $\deg(g, x) > 0$ then $\operatorname{res}(f,\, g,\, x) = f^{\deg(g,\, x)}$.

In the sequel, a polynomial $f \in S$ is said to be regular modulo $\mathfrak{B}$ (recall $\mathfrak{B} = (C) : I_C^\infty$) if it is a regular element of the ring $S/\mathfrak{B}$. Regular elements and zerodivisors of a ring are defined as in [21, chapter I, § 5].

Theorem 1. *Assume C is a regular chain. A polynomial f is regular modulo $\mathfrak{B}$ if, and only if,* $\operatorname{res}(f,\, C) \neq 0$*. Together with the iterated resultant* $q = \operatorname{res}(f,\, C)$*, one can compute a polynomial p such that*

$$p\, f = q \mod \mathfrak{B}$$

If f is a zerodivisor modulo $\mathfrak{B}$ then $q = 0$, else p/q is an inverse of f modulo $\mathfrak{B}$.

Proof. The triangularity of C ensures that $\mathrm{res}(f, C) \in K[N]$. Thus, if the first part of the Theorem is proved, the second one follows immediately by Lemma 1.

In order to prove the first part of the Theorem, we first show that we can reduce our problem to the zerodimensional case[2]. Denote $S_0 = K(N)[L]$, and $\mathfrak{B}_0 = (C) : I_C^\infty$ in the ring S_0. By [5, Theorem 1.6], the multiplicative family of the nonzero elements of $K[N]$, is regular modulo $\mathfrak{B}$. Thus the ring $S_0/\mathfrak{B}_0$ is a subring of the total ring of fractions of $S/\mathfrak{B}$ [21, chapter IV, § 9]. Thus, f is regular modulo $\mathfrak{B}$ in S if, and only if, $f/1$ is regular modulo $\mathfrak{B}_0$ in S_0 [21, chapter I, § 19, Corollary 1].

Assume C is a regular chain in S. Then it is a zerodimensional regular chain in S_0. By [8, Lemma 4], an element $f/1$ is regular modulo $\mathfrak{B}_0$ in S_0 if, and only if, $\mathrm{res}(f, C) \neq 0$. Therefore, a polynomial f is regular modulo $\mathfrak{B}$ if, and only if, $\mathrm{res}(f, C) \neq 0$.

The next three lemmas prepare Theorem 2, which gives a necessary and sufficient condition that a triangular set C needs to satisfy in order to be a regular chain. Thus, recall that C is only supposed to be a triangular set.

Lemma 4. *Assume $\mathfrak{B}$ is proper. Let f be any polynomial. If* $\mathrm{res}(f, C) \neq 0$ *then f is regular modulo $\mathfrak{B}$.*

Proof. Let $\mathfrak{p}$ be any associated prime ideal of $\mathfrak{B}$ (such a $\mathfrak{p}$ exists for $\mathfrak{B}$ is proper). Take Formula (2) modulo $\mathfrak{p}$. The triangularity of C implies that $\mathrm{res}(f, C) \in K[N]$. By [5, Theorem 1.6] and the hypothesis, $\mathrm{res}(f, C) \neq 0 \mod \mathfrak{p}$. Since the elements of C are zero modulo $\mathfrak{p}$, the polynomial f is nonzero modulo $\mathfrak{p}$, i.e. is regular modulo $\mathfrak{B}$ by [21, chapter IV, § 6, Corollary 3].

The following lemma is new.

Lemma 5. *Assume $\mathfrak{B}$ is proper. Assume that, for any polynomial f which is regular modulo $\mathfrak{B}$, we have* $\mathrm{res}(f, C) \neq 0$. *Then C is a regular chain.*

Proof. The initials of the elements of $C = \{c_1, \ldots, c_n\}$ are regular modulo $\mathfrak{B}$ by [21, chapter IV, § 6, Corollary 3, and § 10, Theorem 17] and the fact that $\mathfrak{B}$ is proper. Thus, by assumption, for each $1 \leq k \leq n$, we have $\mathrm{res}(i_k, C) \neq 0$, where i_k denotes the initial of c_k. Thus, by Lemma 3 and the fact that $\deg(i_k, x_\ell) = 0$ for $k \leq \ell \leq n$, we have $\mathrm{res}(i_k, C_{k-1}) \neq 0$, where $C_{k-1} = \{c_1, \ldots, c_{k-1}\}$. Then, by Lemma 4, the initial i_k is regular modulo $(C_{k-1}) : I_{C_{k-1}}^\infty$. Thus C is a regular chain.

The following lemma is part of [8, Theorem 1].

Lemma 6. *Let h denote the product of the initials of the elements $c_2, \ldots, c_n$ of C. If* $\mathrm{res}(h, C) \neq 0$ *then C is a regular chain.*

Proof. Denote $C_k = \{c_1, \ldots, c_k\}$, for $1 \leq k \leq n$. By Lemma 2, Lemma 3 and the hypothesis, $\mathrm{res}(i_k, C_{k-1}) \neq 0$, for all $2 \leq k \leq n$. The ideal $(C_1) : I_{C_1}^\infty$ is

[2] We are actually proving a very close variant of [5, Theorem 1.1].

proper. By Lemma 4, and the fact that $\operatorname{res}(i_2, C_1) \neq 0$, the initial i_2 is regular modulo $(C_1) : I_{C_1}^{\infty}$. The set C_2 is thus a regular chain and $(C_2) : I_{C_2}^{\infty}$ is proper. By Lemma 4, and the fact that $\operatorname{res}(i_3, C_2) \neq 0$, the initial i_3 is regular modulo $(C_2) : I_{C_2}^{\infty}$. Continuing, one concludes that C is a regular chain.

In the following theorem, the implication $2 \Rightarrow 1$ is new. The equivalence between the other points is a consequence of [8, Theorem 1] and [14, Theorem 2.4].

Theorem 2. *Let C be a triangular set. The three following properties are equivalent.*

1. *C is a regular chain ;*
2. *a polynomial f is regular modulo $\mathfrak{B}$ if, and only if,* $\operatorname{res}(f, C) \neq 0$ *;*
3. $\operatorname{res}(h, C) \neq 0$*, where h denotes the product of the initials of the elements $c_2, \ldots, c_n$ of C.*

Proof. The implication $1 \Rightarrow 2$ is a corollary to Theorem 1. The implication $2 \Rightarrow 1$: since $\operatorname{res}(1, C) \neq 0$ for any triangular set C, Property 2 implies that $\mathfrak{B}$ is proper ; the implication is thus a corollary to Lemma 5. The implication $3 \Rightarrow 1$ is Lemma 6. The implication $2 \Rightarrow 3$: Property 2 implies that $\mathfrak{B}$ is proper ; thus h is regular modulo $\mathfrak{B}$ by [21, chapter IV, § 6, Corollary 3, and § 10, Theorem 17]. Thus $\operatorname{res}(h, C) \neq 0$.

Comparison of Theorem 1 with other works. Inspecting the proof of Lemma 6, we see that Property 3 is equivalent to [19, Definition 1.3 (normal ascending chains)], which refers to [20], i.e. that $\operatorname{res}(i_k, C) \neq 0$ for $2 \leq k \leq n$, where i_k denotes the initial of c_k. Therefore, normal ascending chains and regular chains are exactly the same objects.

An algorithm for computing the inverse of a polynomial modulo a regular chain can be found in [15, Algorithm 3]. This algorithm relies on the hypothesis that the polynomial to be inverted is regular modulo the ideal defined by the chain. It relies on a different method (linear system solving) and is not proved.

Another algorithm for computing the inverse of a polynomial modulo a regular chain can be found in [6, Algorithm Invert]. It is based on a Gröbner basis computation. It is based on Kalkbrener's definition of regular chains [12] and thus computes an inverse of a polynomial modulo the intersection of all the prime ideals which contain the ideal defined by the chain, which have dimension $|N|$ and do not meet the multiplicative family M generated by the nonzero elements of $K[N]$, i.e. modulo the radical of the ideal defined by the regular chain. However, [6] misses [5, Theorem 1.6] which implies that $\mathfrak{B}$ has the same set of associated prime ideals as its radical, hence that the computed inverse also is an inverse modulo $\mathfrak{B}$.

Theorem 1 enhances [8, Lemma 4] which is stated in the zerodimensional case only, and does not provide the inverse computation.

Theorem 1 enhances also [18, Proposition 5.3]. Indeed, this Proposition states that $\operatorname{res}(f, C) \neq 0$ if, and only if, the polynomial f does not annihilate on any "regular zero" of C, where "regular zeros" are defined as generic zeros of the associated prime ideals of $\mathfrak{B}$ which have dimension $|N|$ and do not meet the multiplicative family M generated by the nonzero elements of $K[N]$ (see [18, Definition 5.1]). However, [18] misses [5, Theorem 1.6] which states that this property is held by all the associated prime ideals of $\mathfrak{B}$. See the comments on [6, Algorithm Invert].

The fact that a polynomial f is regular modulo $\mathfrak{B}$ if, and only if, $\operatorname{res}(f, C) \neq 0$ is already stated in [14, Theorem 2.4]. However, the proof of that Theorem just refers to [8] and [18] and thus misses the use of [5, Theorem 1.6].

Relationship between Theorem 1 and [5, Theorem 1.6]. Theorem 1 implies "easily" [5, Theorem 1.6] in the particular case of regular chains, i.e. that the associated prime ideals of $\mathfrak{B}$ have dimension $|N|$ and do not meet the multiplicative family M generated by the nonzero elements of $K[N]$. This remark is interesting for [5, Theorem 1.6] is one of the most difficult results of the regular chains theory. See [5]. It stresses, moreover, the strong relationship between the two theorems.

Proof. The regular chain C is a triangular set. Thus, for any nonzero $f \in K[N]$ the iterated resultant $\operatorname{res}(f, C)$ also is a nonzero element of $K[N]$. Thus, by Theorem 1, for any associated prime ideal $\mathfrak{p}$ of $\mathfrak{B}$, we have $\mathfrak{p} \cap M = \varnothing$ whence $\dim \mathfrak{p} \geq |N|$. Since the initials of the element of C do not lie in $\mathfrak{p}$, the derivatives $x_1, \ldots, x_n$ are algebraically dependent over N modulo $\mathfrak{p}$ and $\dim \mathfrak{p} \leq |N|$. Therefore, $\dim \mathfrak{p} = |N|$.

Observe that Theorem 1 does not hold for general triangular sets, while [5, Theorem 1.6] does. This claim is easily proved by an example. Take $f = x - 1$ and $C = \{x^2 - 1, (x - 1)\, y^2 - 2\}$ with $x < y$. The set C is triangular but is not a regular chain, for the initial $x - 1$ of the second element of C is not regular modulo the ideal defined by the first element. The ideal $\mathfrak{B}$ is generated by $\{x + 1, y^2 + 1\}$. It is prime, hence equal to its unique associated prime, if we assume $K = \mathbb{Q}$. The polynomial f is regular modulo $\mathfrak{B}$. However, $\operatorname{res}(f, C) = 0$.

Computational comment. For computational purposes, it is desirable to avoid computing the resultant with respect to x_k of polynomials which do not both depend on x_k, as in [8, Definition 4] and [15, Definition 1]. The iterated resultant is then defined as follows:

$$\operatorname{res}(f, C) = \overline{\operatorname{res}}(\cdots \overline{\operatorname{res}}(f, c_n, x_n), \ldots, c_1, x_1)\,, \tag{6}$$

where $\overline{\operatorname{res}}(f, c_k, x_k)$ is equal to $\operatorname{res}(f, c_k, x_k)$ if $\deg(f, x_k) > 0$ else is equal to f. Lemma 1 still holds with this definition of iterated resultants. By Lemma 2 and Lemma 3, the vanishing conditions of the iterated resultant $\operatorname{res}(f, C)$ are the same with Formula (1) as with Formula (6). Therefore, Theorems 1 and 2 also hold with Formula (6).

Computation of algebraic inverses and normal forms. Consider the triangular set $C = \{(x - 1)\,(x - 2), y^2 - 1\}$ for the ordering $y > x$. Since the initials are equal to 1, it is a regular chain. Consider the polynomial $f = (x-1)\,y + (x-2)$. We have $p\,f = -1 = \mathrm{res}(f,\,C)$, where $p = (-y\,x + y + x - 2)\,(2\,x - 3)$. Thus f is regular modulo the ideal $\mathfrak{B} = (C) : I_C^\infty$. Its inverse is $-p$ modulo $\mathfrak{B}$. Observe that the function [2, Inverse] would have failed to compute the inverse of f, since it would have tried to invert the initial $x - 1$ of f modulo $\mathfrak{B}$, which is a zerodivisor modulo $\mathfrak{B}$, before computing the remainder of $y^2 - 1$ by f in the algorithm provided in [2, Figure 5]. Therefore, $\mathrm{NF}(1/f,\,C)$ succeeds with the new algorithm, given in Figure 1, while it fails with the old one, because of the inverse computation of f, w.r.t. C.

5 On the Regularity Property of Differential Polynomials

In this section, C denotes a regular differential chain of the differential polynomial ring R, defining a differential ideal $\mathfrak{A} = [C] : H_C^\infty$. Let $L = \mathrm{ld}\,C$ and $N = \Theta U \setminus \Theta L$.

The following Theorem provides an algorithm for deciding if a differential polynomial is regular modulo a differential ideal defined by a regular differential chain, and, if it is, for computing an inverse of it.

Theorem 3. *Let f be any differential polynomial, r be its partial remainder w.r.t. C and h a product of initials and separants of C such that $h\,f = r \mod \mathfrak{A}$. Together with the iterated resultant $q = \mathrm{res}(r,\,C)$, it is possible to compute a polynomial p such that*

$$p\,r = q \mod (C) : I_C^\infty$$

If f is a zerodivisor modulo $\mathfrak{A}$ then $q = 0$, else $h\,p/q$ is an inverse of f modulo $\mathfrak{A}$.

Proof. The key arguments are the following: on the one hand, by [4, Corollary 4 to Theorem 3], a differential polynomial is regular modulo $\mathfrak{A}$ if, and only if, its partial remainder with respect to C is regular modulo $\mathfrak{B} = (C) : H_C^\infty$; on the other hand, $\mathfrak{B} = (C) : I_C^\infty$ by [11, Lemma 6.1].

If f is a zerodivisor modulo $\mathfrak{A}$, then r is a zerodivisor modulo $\mathfrak{B}$ and $q = 0$ by Theorem 1. Assume f is regular modulo $\mathfrak{A}$. Then r is regular modulo $\mathfrak{B}$ and q is a nonzero element of $K[N]$ by Theorem 1. Since $\mathfrak{B} \subset \mathfrak{A}$, we have $h\,p\,f = q \mod \mathfrak{A}$. Thus $h\,p/q$ is an inverse of f modulo $\mathfrak{A}$.

A complete algorithm for computing the normal form of a rational differential fraction is presented in Figure 1. This algorithm is obtained from [2, The NF function, Figure 2] by udpating the method applied for computing inverses.

Theorem 4. *Let a/b be a rational differential fraction and C be a regular differential chain. If b is a zerodivisor modulo $\mathfrak{A}$, then $\mathrm{NF}(a/b,\,C)$ raises an error, else $\mathrm{NF}(a/b,\,C)$ returns the normal form of a/b modulo C.*

Proof. The Theorem is simply a restatement of [2, Proposition 5.3], taking into account the fact that inverses are computed using a method (Theorem 3) which succeeds if, and only if, the polynomial to be inverted is invertible.

Comment. The algorithm presented in Figure 1 has a drawback with respect to [2, The NF function, Figure 2]: if the denominator of the rational fraction is a zerodivisor, the algorithm does not exhibit a factorization of some element of C. This drawback may be easily overcome if one computes resultants by means of pseudoremainder sequences.

```
function NF($a/b$, $C$)
Parameters
  $a/b$ is a rational differential fraction such that $a, b \in R$.
  $C$ is a regular differential chain, defining a differential ideal $\mathfrak{A}$.
Result
  if $b$ is regular modulo $\mathfrak{A}$, then the normal form of $a/b$ modulo $\mathfrak{A}$, else an error
begin
Regularity test and inverse computation of the denominator
  Apply Theorem 3 over $b$:
    if $b$ is a zerodivisor modulo $\mathfrak{A}$ then
      error "the denominator is a zerodivisor modulo $\mathfrak{A}$"
    end if
    Denote $p_{\mathrm{b}}/q_{\mathrm{b}}$ an inverse of $b$ modulo $\mathfrak{A}$
Inverse computation of the separants (they are necessarily regular)
  Apply Theorem 1 over each separant $s_i$ of $C = \{c_1, \ldots, c_n\}$ and
                      denote $p_i/q_i$ an inverse of $s_i$ modulo $\mathfrak{A}$
  $(f_{n+2}, g_{n+2}) := (p_{\mathrm{b}}\, a, q_{\mathrm{b}})$
  Using Ritt's partial reduction algorithm, compute $d_1, \ldots, d_n \in \mathbb{N}$ and
                      $r_{n+1} \in K[N \cup L]$ such that $s_1^{d_1} \cdots s_n^{d_n} f_{n+2} \equiv r_{n+1} \mod \mathfrak{A}$
  $f_{n+1} := p_1^{d_1} \cdots p_n^{d_n} r_{n+1}$
  $g_{n+1} := q_1^{d_1} \cdots q_n^{d_n} g_{n+2}$
  Denote $x_i = \mathrm{ld}\, c_i \ (1 \le i \le n)$ and assume $x_n > \cdots > x_1$
  for $\ell$ from $n$ to 1 by $-1$ do
    $r_\ell := \mathrm{prem}(f_{\ell+1}, c_\ell, x_\ell)$
    Let $i_\ell$ denote the initial of $c_\ell$
    Let $d_\ell \in \mathbb{N}$ be such that $i_\ell^{d_\ell} f_{\ell+1} \equiv r_\ell \mod (c_\ell)$
Inverse computation of an initial (it is necessarily regular)
    Apply Theorem 1 over $i_\ell$ and denote $p_\ell/q_\ell$ an inverse of $i_\ell$ modulo $\mathfrak{A}$
    $f_\ell := p_\ell^{d_\ell} r_\ell$
    $g_\ell := q_\ell^{d_\ell} g_{\ell+1}$
  end do
  return $f_1/g_1$
the rational fraction may be reduced by means of a gcd computation
of multivariate polynomials over the field $K$
end
```

Fig. 1. The NF function

References

[1] Aubry, P., Lazard, D., Maza, M.M.: On the Theories of Triangular Sets. Journal of Symbolic Computation 28, 105–124 (1999)

[2] Boulier, F., Lemaire, F.: A Normal Form Algorithm for Regular Differential Chains. Mathematics in Computer Science 4(2), 185–201 (2010), doi:10.1007/s11786-010-0060-3

[3] Boulier, F.: Efficient computation of regular differential systems by change of rankings using Kähler differentials. Technical report, Université Lille I, 59655, Villeneuve d'Ascq, France, Ref. LIFL 1999–14, presented at the MEGA (2000), conference (November 1999), http://hal.archives-ouvertes.fr/hal-00139738

[4] Boulier, F., Lazard, D., Ollivier, F., Petitot, M.: Computing representations for radicals of finitely generated differential ideals. Applicable Algebra in Engineering, Communication and Computing 20(1), 73–121 (2009); (1997 Tech. rep. IT306 of the LIFL)

[5] Boulier, F., Lemaire, F., Maza, M.M.: Well known theorems on triangular systems and the D^5 principle. In: Proceedings of Transgressive Computing 2006, Granada, Spain, pp. 79–91 (2006), http://hal.archives-ouvertes.fr/hal-00137158

[6] Bouziane, D., Rody, A.K., Maârouf, H.: Unmixed–Dimensional Decomposition of a Finitely Generated Perfect Differential Ideal. Journal of Symbolic Computation 31, 631–649 (2001)

[7] Busé, L., Mourrain, B.: Explicit factors of some iterated resultants and discriminants. Mathematics of Computation 78, 345–386 (2009)

[8] Chen, C., Golubitsky, O., Lemaire, F., Maza, M.M., Pan, W.: Comprehensive Triangular Decomposition. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2007. LNCS, vol. 4770, pp. 73–101. Springer, Heidelberg (2007)

[9] Cox, D., Little, J., O'Shea, D.: Using Algebraic Geometry, 2nd edn. Graduate Texts in Mathematics, vol. 185. Springer, New York (2005)

[10] Faugère, J.-C., Gianni, P., Lazard, D., Mora, T.: Efficient computation of Gröbner bases by change of orderings. Journal of Symbolic Computation 16, 329–344 (1993)

[11] Hubert, É.: Factorization free decomposition algorithms in differential algebra. Journal of Symbolic Computation 29(4,5), 641–662 (2000)

[12] Kalkbrener, M.: A Generalized Euclidean Algorithm for Computing Triangular Representations of Algebraic Varieties. Journal of Symbolic Computation 15, 143–167 (1993)

[13] Kolchin, E.R.: Differential Algebra and Algebraic Groups. Academic Press, New York (1973)

[14] Lemaire, F., Maza, M.M., Pan, W., Xie, Y.: When does (T) equal sat(T)? In: Proceedings of thee International Symposium on Symbolic and Algebraic Computation, pp. 207–214. ACM Press, New York (2008)

[15] Li, B., Wang, D.: An Algorithm for Transforming Regular Chain into Normal Chain. In: Kapur, D. (ed.) ASCM 2007. LNCS (LNAI), vol. 5081, pp. 236–245. Springer, Heidelberg (2008)

[16] Ritt, J.F.: Differential Algebra. Dover Publications Inc., New York (1950)

[17] van der Waerden, B.L.: Algebra, 7th edn. Springer, Berlin (1966)

[18] Wang, D.: Computing Triangular Systems and Regular Systems. Journal of Symbolic Computation 30, 221–236 (2000)
[19] Yang, L., Hou, X., Xia, B.: A complete algorithm for automated discovering of a class of inequality-type theorems. Science in China Series F: Information Sciences 44(1), 33–49 (2001)
[20] Yang, L., Zhang, J., Hou, X.: An Efficient Decomposition Algorithm for Geometry Theorem Proving Without Factorization. In: Proceedings of ASCM, pp. 33–41 (1995)
[21] Zariski, O., Samuel, P.: Commutative Algebra. In: Commutative Algebra. Graduate Texts in Mathematics, Van Nostrand, New York, vol. 28,29. Springer, Heidelberg (1958)

Chemical Reaction Systems, Computer Algebra and Systems Biology*
(Invited Talk)

François Boulier, François Lemaire, Michel Petitot, and Alexandre Sedoglavic

Université Lille I, LIFL, 59655 Villeneuve d'Ascq, France

1 Introduction

In this invited paper, we survey some of the results obtained in the computer algebra team of Lille, in the domain of systems biology. So far, we have mostly focused on models (systems of equations) arising from generalized chemical reaction systems. Eight years ago, our team was involved in a joint project, with physicists and biologists, on the modeling problem of the circadian clock of the green algae *Ostreococcus tauri*. This cooperation led us to different algorithms dedicated to the reduction problem of the deterministic models of chemical reaction systems. More recently, we have been working more tightly with another team of our lab, the BioComputing group, interested by the stochastic dynamics of chemical reaction systems. This cooperation led us to efficient algorithms for building the ODE systems which define the statistical moments associated to these dynamics. Most of these algorithms were implemented in the MAPLE computer algebra software. We have chosen to present them through the corresponding MAPLE packages.

Chemical reaction systems provide a general setting for modeling in biology. More generally, chemical kinetics may be viewed as a prototype of nonlinear science, as pointed out in [33, chapter 1].

The reaction $A \longrightarrow B$ describes the transformation of a *species* A into a different species B. Species A is the *reactant* of the reaction. Species B is the *product*. One often endows a reaction with a symbol k, which parametrizes the speed of the transformation. The reaction is then denoted $A \xrightarrow{k} B$. The reaction $A \longrightarrow \varnothing$ describes the transformation of species A into a species which is not part of the model. This sort of reaction often occurs when one models biological phenomenons but one usually does not encounter it in chemistry, since it is not *equilibrated*. Symmetrically, the reaction $\varnothing \longrightarrow B$ describes the entry, in the model, of a species B, from outside the model. A more complicated reaction is $A+B \longrightarrow C$. It is interpreted as follows: when a molecule of A encounters a molecule of B, both may react and form a molecule of a third species C. Last, one sometimes encounters reactions denoted $A+B \rightleftarrows C$. Some authors consider it as a single revertible reaction. We view it as a pair of two reactions. The

* This work has benefited from the support of the French ANR (decision number ANR-2010-BLAN-0109-03).

right to the left reaction may be described as follows: every molecule of C may break itself and yield two molecules: one of species A and one of species B.

A chemical reaction system is a set of chemical reactions. Here are two examples. The first one is a classics of chemistry lectures: it is the simplest example of an enzymatic reaction. It describes the transformation of a substrate S into a product P, in the presence of some enzyme E. An intermediate complex ES is formed:

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftarrows}} ES \xrightarrow{k_2} E + P\,. \tag{1}$$

The second example, adapted from [27, Syst. (7.19)], models a two-species oscillator:

$$\varnothing \xrightarrow{a} A\,, \qquad A \xrightarrow{k_1} \varnothing\,, \qquad 2\,A + B \xrightarrow{k_2} 3\,A\,, \qquad \varnothing \xrightarrow{b} B\,. \tag{2}$$

In order to build systems of equations, i.e. mathematical models of chemical reaction systems, one needs more precise assumptions. There are at least two families of models: the deterministic and the stochastic ones.

2 Deterministic Modeling

There are different ways to associate a precise deterministic dynamics to a chemical reaction system. In this paper, we focus on the mass-action dynamics. The parameters of the reactions are called *kinetic constants*. To each species A, one associates a function $A(t)$ which represents the concentration of the species. The evolution of each concentration is given by an ordinary differential equation, in which kinetic constants appear as parameters. The system of ordinary differential equations built using the mass-action law, from a chemical reaction system, is called the *natural deterministic model* of the system. It is given by the following formula:

$$\frac{\mathrm{d}X}{\mathrm{d}t} = N \cdot V$$

where X is the column vector of the concentrations, N is the *stoichiometry matrix*, and V is the column vector of the *laws*. The law of a reaction is obtained by multiplying the kinetic constant by the product of the concentrations of the reactants. The stoichiometry matrix involves one row per species and one column per reaction. Its coefficient, row r and column c, is equal to the number of molecules of species r produced by the reaction c. This number is equal to the number of occurences of species r as a product, minus the number of occurences of species r as a reactant. On System (2), we have

$$X = \begin{pmatrix} A(t) \\ B(t) \end{pmatrix}, \qquad N = \begin{pmatrix} 1 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{pmatrix}, \qquad V = \begin{pmatrix} a \\ k_1\,A(t) \\ k_2\,A(t)^2\,B(t) \\ b \end{pmatrix}.$$

Natural deterministic models of chemical reaction systems were intensively studied. Many informations can be read in the stoichiometry matrix alone: the nullspace of its transpose provides linear conservation laws. System (2) does not have any, while computing a basis of this nullspace for System (1) yields the two laws:

$$-E(t) + S(t) + P(t) = \mathrm{cst}_1\,, \qquad E(t) + ES(t) = \mathrm{cst}_2\,. \tag{3}$$

A striking property of chemical reaction systems is a very simple necessary and sufficient condition for a system of polynomial ODE to be the natural deterministic model of a chemical reaction system: it is necessary and sufficient that, in the right hand side of the ODE which describes the evolution of any concentration $A(t)$, there does not exist any monomial which is both free of $A(t)$ and endowed with a minus sign [33, section 4.7.1]. Another striking, but much more difficult result is the *zero deficiency theorem* of Feinberg, Horn and Jackson [14], which gives a sufficient condition for a system to admit a unique steady state with strictly positive coordinates. This sufficient condition, which can be formulated in the setting of the graph theory, can be implemented very easily. This theorem was much studied, recently, from different points of views. See [8, 15] and the references therein.

The size of natural deterministic models becomes so large, on all but the simplest examples, that it forbids any further analysis. The computer algebra team of Lille has focused on the problem of reducing models. We make use of two types of reduction: a reduction method which permits to approximate the natural deterministic model, under some simplifying assumptions ; and a reduction method, which permits to reduce the number of parameters, and to separate parameters which have an effect on the coordinates and the stabilities of the steady points, from the ones which have an effect on their stabilities only.

2.1 Approximating Models

The approximation method implements the well-known quasi-steady state approximation technique. See [20] for a general presentation and [30, 39] for more chemically oriented texts. Our contribution [4] consisted in formulating it, in the context of chemical reaction systems, as the result of a differential elimination process [3]. In this section, we present it over System (1), by showing how to obtain the famous Henri, Michaelis and Menten formula [18, 25]

$$\frac{\mathrm{d}S}{\mathrm{d}t}(t) = -\frac{V_{\max}\, S(t)}{K + S(t)} \tag{4}$$

where $V_{\max}$ and K are two parameters which can be expressed from the kinetic constants of the chemical reaction system. See [29] for a closely related work. Our algorithm was successfully applied over a more complicated model [5, 6], featuring two genes, arising from the modeling problem of the circadian clock of the green algae *Ostreococcus tauri* [12].

Computation using DifferentialAlgebra. Our method relies on the assumption that the chemical reactions are split into two sets: the set of the fast reactions, and the set of the slow ones. In the case of the Henri, Michaelis and Menten reduction, the revertible reaction is supposed to be fast, compared to the third one: k_1, $k_{-1} \gg k_2$. One starts by building the natural deterministic model.

```
> with (LinearAlgebra):
> X := <E(t), S(t), ES(t), P(t)>:
> V := <k[1]*E(t)*S(t), k[-1]*ES(t), k[2]*ES(t)>:
> N := <<-1, -1, 1, 0> | <1, 1, -1, 0> | <1, 0, -1, 1>>:
> X, N, V;
                 [E(t) ]  [-1     1     1]
                 [     ]  [              ]  [k[1] E(t) S(t)]
                 [S(t) ]  [-1     1     0]  [              ]
                 [     ], [              ], [ k[-1] ES(t)  ]
                 [ES(t)]  [ 1    -1    -1]  [              ]
                 [     ]  [              ]  [  k[2] ES(t)  ]
                 [P(t) ]  [ 0     0     1]
```

Then, in the vector V, one replaces the laws corresponding to the fast reactions by two unknown laws $F_1(t)$ and $F_{-1}(t)$. Denote W this new vector. The approximated system is obtained by enlarging the ODE system $\mathrm{d}X/\mathrm{d}t = N \cdot W$, with the equation $k_1\, E(t)\, S(t) = k_{-1}\, E(t)\, S(t)$, which gives the algebraic variety where the fast reactions would be equilibrated, if the slow reactions did not exist.

```
> W := <F[1](t), F[-1](t), k[2]*ES(t)>:
> equilibre := 0 = k[1]*E(t)*S(t) - k[-1]*ES(t):
> redsys := map (diff, X, t) = N . W:
> redsys := [ seq (lhs (redsys) [i] = rhs (redsys) [i],
                           i = 1 .. Dimension (X)), equilibre ];
           d
redsys := [-- E(t) = -F[1](t) + F[-1](t) + k[2] ES(t),
           dt


           d
           -- S(t) = -F[1](t) + F[-1](t),
           dt


           d
           -- ES(t) = F[1](t) - F[-1](t) - k[2] ES(t),
           dt


           d
           -- P(t) = k[2] ES(t), 0 = k[1] E(t) S(t) - k[-1] ES(t)]
           dt
```

This differential-algebraic equation can now be simplified, by means of a differential elimination software, such as the DifferentialAlgebra package or the recent [1]. With the DifferentialAlgebra package, the output involves three cases. We only give the equation which expresses the evolution of $S(t)$, in the general case.

```
> with (DifferentialAlgebra):
> Field := field (generators = [k[1],k[-1],k[2]]);
                Field := field(generators = [k[1], k[-1], k[2]])


> R := DifferentialRing
    (blocks = [[F[1],F[-1]], [ES,E,P,S], [k[1](),k[-1](),k[2]()]],
     derivations = [t]);
                        R := differential_ring


> ideal := RosenfeldGroebner (redsys, basefield = Field, R):
ideal := [regular_differential_chain, regular_differential_chain,

    regular_differential_chain]


> Equations (ideal[1], solved, leader=diff(S(t),t));
                         2     2
    d           E(t) S(t)  k[1]  k[2] + E(t) S(t) k[1] k[-1] k[2]
   [-- S(t) = - ------------------------------------------------]
    dt                                                      2
                   E(t) k[1] k[-1] + S(t) k[1] k[-1] + k[-1]
```

This equation is not yet the Henri, Michaelis and Menten formula, since some minor hypotheses are missing: one still needs to take conservation laws (3) and some initial values into account. Last, one needs to rename $K = k_1/k_{-1}$ and $V_{\max} = k_2\, E(0)$ and to neglect some monomial $K\,E(0)$, assuming $S(0) \gg E(0)$. Here is a sequence of computations, which takes these hypotheses into account and leads to the sought formula.

```
> conservation_laws :=
       [E(t) + ES(t) = E0 + ES0,
        S(t) + ES(t) + P(t) = S0 + ES0 + P0]:
> hypotheses := [P0 = 0, ES0 = 0, op (conservation_laws)]:
> R := DifferentialRing (
       blocks = [[F[1],F[-1]], [ES,E,P,S],
                 [ES0(),E0(),P0(),S0()], [k[1](),k[-1](),k[2]()]],
       derivations = [t]):
> ideal := RosenfeldGroebner ([ op(redsys), op(hypotheses) ],
                                              R, basefield = Field):
> formula := Equations (ideal[1], solved, leader = diff (S(t),t)):
> formula := subs (k[-1]=K*k[1], formula):
> formula := normal (formula):
> formula := algsubs (k[2]*E0=Vmax, formula):
```

```
> formula := normal (subs (K*E0=0, formula));

                                   d            Vmax S(t)
                       formula := [-- S(t) = - ---------]
                                   dt           S(t) + K
```

Computation using MABSys. The same computation can also be performed by the dedicated MABSys package [23, 24], which relies on the MAPLE RegularChains [21] and ELPS [36] packages. The MABSys package gathers as input a chemical reaction system. Each reaction can be defined as fast or slow. The package permits to obtain directly the stoichiometry matrix, the vector of the laws, the system of the equilibria, the conservation laws that can be read from the stoichiometry matrix and the natural deterministic model.

```
> with(MABSys):
> R1 := NewReaction (E+S, C, MassActionLaw(k1), fast=true):
> R2 := NewReaction (C, E+S, MassActionLaw(km1), fast=true):
> R3 := NewReaction (C, E+P, MassActionLaw(k2)):
> RS := [R1,R2,R3]:

> ConservationLaws(RS);
                [C + P + S - C_0 - P_0 - S_0, C + E - C_0 - E_0]

> sys := ReactionSystem2ODEs(RS, [E,S,C,P]);

sys :=

   d
  [-- E(t) = -k1 E(t) S(t) + km1 C(t) + k2 C(t),
   t

  d
  -- S(t) = -k1 E(t) S(t) + km1 C(t),
  dt

  d                                             d
  -- C(t) = k1 E(t) S(t) - km1 C(t) - k2 C(t), -- P(t) = k2 C(t)]
  dt                                            dt
```

The package provides a function that performs the quasi-steady state approximation using the information provided by the `fast` boolean.

```
> output := ModelReduce(RS, [E,C,P,S], useConservationLaws=true):
> red_sys := output[1][1]:
> red_sys := subs (C_0=0, P_0=0, red_sys):
> red_sys[4];
```

```
d                   E_0 k2 k1 S(t) (k1 S(t) + km1)
-- S(t) = - ---------------------------------------------
dt            2     2                                   2
            k1  S(t)  + 2 S(t) km1 k1 + km1 k1 E_0 + km1
```

It also provides a function that permits to reduce the number of parameters of the system. Over this example, the parameter k_1 is removed, using the assumption that it is strictly positive. The change of coordinates is provided, following the syntax: "new parameter = rational function of the old parameters".

```
> output := InvariantizeByScalings(red_sys,[k1],[km1,k2,E,C,P,S]):
> red_sys2 := output[1]:
> red_sys2[4];
```

```
        d                S(t) k2 E_0 (S(t) + km1)
        -- S(t) = - ----------------------------------
        dt                2                           2
                    S(t)  + 2 km1 S(t) + E_0 km1 + km1
```

```
> output[2];
                                       km1
                                [km1 = ---]
                                       k1
```

Other functions, which are still prototypes, hence not yet integrated in MABSys, permit to perform the final approximation which yields the Henri, Michaelis and Menten formula.

2.2 Reducing and Reparametrizing Models

This section, in which we give some more details on the method which permitted us to remove the k_1 parameter, is much inspired from [38, chapter 2] and [22]. The goal consists in reducing and reparametrizing an initial model into a reduced one (the initial model possibly comes from the quasi-steady state approximation method). The reduced model is equivalent to the initial one, by an invertible change of coordinates, which preserves the most important properties: the positive steady points of the initial model are in bijection with the positive steady points of the reduced one, the initial model presents an oscillating behaviour if, and only if, the reduced model does, and so on.

The computed changes of coordinates are given by Lie symmetries of the initial system of the simplest type: they are restricted to scalings. Moreover, the method distinguishes the parameters which are supposed to be strictly positive (dividing by them is allowed) from the ones which are only supposed to be nonnegative (dividing by them is forbidden). The computations of the scalings is performed by the ELPS package. It is important to point out that this package computes, in general, a restricted set of the scalings of the input system, in order to preserve a worst case polynomial complexity [35].

The method is illustrated over System ([2]), whose natural deterministic model [27, Eqs. (7.20)] depends on two variables and four parameters.

```
> ODS := [
>     diff(A(t),t) = a - k1 * A(t) + k2 * A(t)^2 * B(t),
>     diff(B(t),t) = b - k2*A(t)^2*B(t)
> ];
           d                               2
  ODS := [-- A(t) = a - k1 A(t) + k2 A(t)  B(t),
           dt

           d                     2
           -- B(t) = b - k2 A(t)  B(t)]
           dt
```

The four parameters are supposed to be strictly positive. The steady points of the natural deterministic model are the zeros of non differential system obtained by equating to zero, the right hand sides of the model equations. The steady points obviously depend on the four parameters.

The reduction step. It makes use of the scalings of the natural deterministic model. These scalings can be described by the two following changes of coordinates. The first one depends on a parameter ν_1. The second one depends on a parameter ν_2.

$$(t,\, A,\, B,\, k_1,\, k_2,\, a,\, b) \longrightarrow \left(t,\, A\,\nu_1,\, B\,\nu_1,\, k_1,\, \frac{k_2}{\nu_1^2},\, a\,\nu_1,\, b\,\nu_1\right).$$

$$(t,\, A,\, B,\, k_1,\, k_2,\, a,\, b) \longrightarrow \left(\frac{t}{\nu_2^2},\, A\,\nu_2,\, B\,\nu_2,\, k_1\,\nu_2^2,\, k_2,\, a\,\nu_2^3,\, b\,\nu_2^3\right).$$

Assuming that the four parameters are strictly positive, these scalings permit to rewrite the natural deterministic model in a new coordinate set. The new system depends on two parameters instead of four. The change of coordinates is given using the syntax: “new variables = rational fractions of the old ones”.

```
> out := InvariantizeByScalings (ODS, [a,b,k1,k2]):
> ODS_reduced := out[1];
ODS_reduced :=

  d                             2       d                        2
 [-- A(t) = 1 - A(t) + k2 A(t)  B(t), -- B(t) = b - k2 A(t)  B(t)]
  dt                                    dt

> Change_of_coordinates := out[2];
```

```
  Change_of_coordinates :=
                                           2
                                      k2  a                    A k1      B k1
                     [b = b/a, k2 = -----, t = t k1, A = ----, B = ----]
                                          3                     a         a
                                        k1
```

The reparametrization step. It makes use of the scalings of the system which gives the steady points of `ODS_reduced`. These scalings can be described by the following change of coordinates, which depends on a parameter ν_3.

$$(A,\, B,\, k_2,\, b) \longrightarrow \left(A,\, B\,\nu_3,\, \frac{k_2}{\nu_3},\, b\right) .$$

The key idea is now very simple: apply these scalings, not on the system which gives the steady points of `ODS_reduced`, but on `ODS_reduced` itself. In the resulting system, the parameter k_2 is still present. However, it has no control anymore on the location of the steady points: it can only control their stabilities.

```
> out := SemiRectifySteadyPoints (ODS_reduced, [b,k2,A,B], []):
> Reparamatrized_system := out[1][1];
Reparamatrized_system :=
   d                             2       d                        2
  [-- A(t) = 1 - A(t) + A(t)  B(t), -- B(t) = (b - A(t)  B(t)) k2]
   dt                                    dt

> Change_of_coordinates := out[1][3];
                       Change_of_coordinates := [B = k2 B]
```

3 Stochastic Modeling

The dynamics of chemical reaction systems can also be studied with a stochastic point of view. The parameters of the reactions are *stochastic constants*, which are probabilities that reactions occur per unit of time [17, page 2342]. To each species A, one associates a random variable $A(t)$ which counts the number of molecules of A. In a celebrated paper [17], Gillespie provided a numerical simulation algorithm, based on a strong rigorous analysis, under some assumptions ; the main one being that the chemically reacting system keeps being well-stirred over the time.

In the context of biological modeling, Gillespie's main assumption is very unlikely. However, the analysis of the stochastic behaviour of chemical reaction systems is a very important counterpart of the deterministic analysis. In a living cell, some molecules may occur in a very little number: if one views a gene, in the "active" state, as a chemical species, the number of molecules is zero or one, and handling this number as a real valued concentration is contestable.

In such cases, it is known that there exist many different stochastic behaviours which are, at least, difficult to reproduce by a deterministic model. See [19, page 454]. Moreover, stochastic simulations may reproduce the surprising effects of the "noise" on the models, as pointed out in [41] and the references therein.

It is well-known that the evolution, over the time t, of the moments (mean, variance, covariance) of the random variable associated to each species may be described by a system of ordinary differential equations, at least for chemical systems the reactions of which do not involve more than one reactant [31, 32]. For more general systems, the differential system is, in general, infinite, and approximating it by a finite ODE system, by performing a so-called *moment closure*, is a difficult problem [16, 37]. In this area, our contribution [40] consisted in showing how to build these ODE systems by using Weyl algebra methods. The use of Weyl algebra led us to a few algorithmic improvements (reducing expressions swells and taking advantage of linear conservation laws at an early stage of the formula generation process).

4 Analysis of Statistical Moments

Before proceeding, one needs to associate a precise stochastic dynamics to a chemical reaction system. This can be achieved using stochastic Petri nets[1], endowed with their standard temporisation [40] ; or by relying on the Gillespie's stochastic simulation algorithm.

Define the *state* ν of a given chemical reaction system as a vector of non-negative integers (one coordinate per chemical species, each coordinate being a number of molecules) ; $\pi_\nu(t)$ as the probability that the chemical reaction system be in the state ν at time t ; and the probability generating function

$$\phi(t,\, z) = \sum_{\nu \geq 0} \pi_\nu(t)\, z^\nu\,, \tag{5}$$

where z^ν stands for $z_1^{\nu_1}\, z_2^{\nu_2} \cdots z_n^{\nu_n}$ (one "symbol" z_i per chemical species). Given any chemical reaction system, it is possible to compute a general equation for ϕ [33, Eq. (5.60)] as follows. To each reaction

$$\alpha_1\, A_1 + \alpha_2\, A_2 + \cdots + \alpha_n\, A_n \xrightarrow{c} \beta_1\, A_1 + \beta_2\, A_2 + \cdots + \beta_n\, A_n\,,$$

associate the differential operator

$$\frac{c}{\alpha!}\, \left(z^\beta - z^\alpha\right) \left(\frac{\partial}{\partial\, z}\right)^{\alpha}. \tag{6}$$

The very same operator can be denoted using *Euler operators* $\theta_i = z_i\, \partial / \partial\, z_i$ instead of partial derivatives

$$\frac{c}{\alpha!}\, \left(z^{\beta-\alpha} - 1\right)\, \theta^{\underline{\alpha}}, \tag{7}$$

[1] According to some sources, cited on the *Wikipedia*, Petri nets were invented by Carl Adam Petri to model chemical reaction systems at the age of 13.

where $\theta^{\underline{\alpha}}$ denotes the product of the $\theta_i^{\underline{\alpha_i}} = \theta_i\,(\theta_i - 1)\cdots(\theta_i - \alpha_i + 1)$. Define H as the sum of the differential operators (6), or (7), for all reactions of the considered chemical reaction system. The general equation for ϕ is:

$$\frac{\partial}{\partial t}\,\phi(t,\,z) = H\,\phi(t,\,z)\,. \tag{8}$$

With partial derivatives, the differential operator H, for System (2), is

$$a\,(z_A - 1) + k_1\,(1 - z_A)\,\frac{\partial}{\partial\,z_A} + \frac{k_2}{2}\,\left(z_A^3 - z_A^2\,z_B\right)\,\frac{\partial^3}{\partial\,z_A^2\,\partial\,z_B} + b\,(z_B - 1)\,.$$

With Euler operators, the differential operator H, for System (1), is

$$k_1\,\left(\frac{z_{ES}}{z_E\,z_S} - 1\right)\,\theta_E\,\theta_S + k_{-1}\,\left(\frac{z_E\,z_S}{z_{ES}} - 1\right)\,\theta_{ES} + k_2\,\left(\frac{z_E\,z_P}{z_{ES}} - 1\right)\,\theta_{ES}\,.$$

By differentiating the probability generating function (5) and evaluating it at $z_1 = z_2 = \cdots = z_n = 1$, one gets formulae which bind ϕ and the moments of the random variables which count the numbers of molecules. A mere evaluation yields: $\phi(t,z)\,|_{z=1} = 1$. Differentiating (5) with respect to any z_i and evaluating at $z = 1$ provides the expected value of the number of molecules of the chemical species associated to z_i.

These ideas are illustrated using a prototype software available at [34], on System (1). Variables are numbered, with the convention $(E,\,S,\,ES,P) = (1,4,2,3)$. The differential operator H is displayed using Euler operators.

```
> cr1 := ChemicalReaction (E+S = ES, k[1], ""):
> cr2 := ChemicalReaction (ES = E+S, k[-1], ""):
> cr3 := ChemicalReaction (ES = E+P, k[2], ""):
> CRS := ChemicalSystem (cr1,cr2,cr3):
> HSyst := HamiltonianSystem (CRS, z, theta, x(t));
                            HSyst := hamiltonian_system

> map (numero, [E,S,ES,P], HSyst);
                                    [1, 4, 2, 3]

> H := Hamiltonian (HSyst);
            /  z[2]       \
  H := k[1] |--------- - 1| theta[1] theta[4]
            \z[1] z[4]    /

               /z[1] z[4]    \
       + k[-1] |--------- - 1| theta[2]
               \  z[2]       /

               /z[1] z[3]    \
        + k[2] |--------- - 1| theta[2]
               \  z[2]       /
```

The next command permits to generate an ODE system for the expected values and the covariances of the random variables which count the molecules of E, S, ES and P (the variable $x_1(t)$ is the expected value of the random variable which counts the molecules of the enzyme E ; the variable $x_{1,4}(t)$ is the covariance of the random variables which count, respectively, the molecules of the enzyme E and the product P). Since some reactions of System (1) involve two reactants, the ODE system is infinite. The function truncates it. The parameter `iv` receives some information on initial values, for a further numerical integration.

```
> ODEs := mean_equations (HSyst, 'iv');


          d
ODEs := [-- x[1](t) = -%2 + (k[-1] + k[2]) x[2](t) - %1,
          dt

          d
          -- x[2](t) = %2 + (-k[-1] - k[2]) x[2](t) + %1,
          dt

          d
          -- x[3](t) = k[2] x[2](t),
          dt

          d                                       d
          -- x[4](t) = -%2 + k[-1] x[2](t) - %1, -- x[1,1](t) = ...
          dt                                      dt

%1 := k[1] x[1, 4](t)

%2 := k[1] x[4](t) x[1](t)
```

5 Conclusion

Beyond the improvements of the underlying theories, a lot of improvements can be brought to the software. Since this paper is software oriented, let us focus on that single issue. First, our packages could certainly be polished and merged. Second, they miss very important tools dedicated to the solving problem of polynomial systems in the field of the real numbers, such as [7, 10, 11, 26, 42]. See [28] for a recent study of the sizes of the problems that can be investigated using some of the best available such tools. However, even the nice integrated MAPLE package that we could foresee would miss two features: sort of a *model checking* functionality, which would permit to the practitioner to query models, as [2, 13] do ; and a user interface of the same quality as that of, say, *Cytoscape* [9]. These two missing features, which are both user interface related, may seem a bit irrelevant to the traditional audience of a computer algebra conference.

The need for them is however pretty obvious for any lecturer, teaching symbolic techniques close to ours, among other systems biology approaches.

References

[1] Bächler, T., Gerdt, V., Lange-Hegermann, M., Robertz, D.: Thomas Decomposition of Algebraic and Differential Systems. In: Gerdt, V.P., Koepf, W., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2010. LNCS, vol. 6244, pp. 31–54. Springer, Heidelberg (2010)

[2] Batt, G., Page, M., Cantone, I., Goessler, G., Monteiro, P., de Jong, H.: Efficient parameter search for qualitative models of regulatory networks using symbolic model checking. In: ECCB, vol. 26, pp. i603–i610 (2010)

[3] Boulier, F., Lazard, D., Ollivier, F., Petitot, M.: Computing representations for radicals of finitely generated differential ideals. Applicable Algebra in Engineering, Communication and Computing 20(1), 73–121 (2009); (1997 Tech. rep. IT306 of the LIFL)

[4] Boulier, F., Lefranc, M., Lemaire, F., Morant, P.-E.: Model Reduction of Chemical Reaction Systems using Elimination. Presented at the International Conference MACIS 2007 (2007), http://hal.archives-ouvertes.fr/hal-00184558

[5] Boulier, F., Lefranc, M., Lemaire, F., Morant, P.-E.: Applying a Rigorous Quasi-Steady State Approximation Method for Proving the Absence of Oscillations in Models of Genetic Circuits. In: Horimoto, K., Regensburger, G., Rosenkranz, M., Yoshida, H. (eds.) AB 2008. LNCS, vol. 5147, pp. 56–64. Springer, Heidelberg (2008)

[6] Boulier, F., Lefranc, M., Lemaire, F., Morant, P.-E., Ürgüplü, A.: On Proving the Absence of Oscillations in Models of Genetic Circuits. In: Anai, H., Horimoto, K., Kutsia, T. (eds.) AB 2007. LNCS, vol. 4545, pp. 66–80. Springer, Heidelberg (2007), http://hal.archives-ouvertes.fr/hal-00139667

[7] Brown, C.W.: QEPCAD B: a program for computing with semi-algebraic sets using CADs. SIGSAM Bulletin 37(4), 97–108 (2003), http://www.cs.usna.edu/~qepcad/B/QEPCAD.html

[8] Chaves, M., Sontag, E.D.: State-Estimators for Chemical Reaction Networks of Feinberg-Horn-Jackson Zero Deficiency Type. European Journal Control 8, 343–359 (2002)

[9] The Cytoscape Consortium. Cytoscape: An Open Source Platform for Complex Network Analysis and Visualization (2001-2010), http://www.cytoscape.org

[10] Dolzmann, A., Sturm, T.: Redlog: computer algebra meets computer logic. SIGSAM Bulletin 31(2), 2–9 (1997)

[11] El Din, M.S.: RAGLib: A library for real solving polynomial systems of equations and inequalities (2007), http://www-salsa.lip6.fr/~safey/RAGLib

[12] Derelle, É., et al.: Genome Analysis of the smallest free-living eukaryote Ostreococcus tauri unveils many unique features. Proceedings of the National Academy of Science of the USA 103(31) (August 2006)

[13] Fages, F., Soliman, S., Chabrier-Rivier, N.: Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM. Journal of Biological Physics and Chemistry 4, 64–73 (2004)

[14] Feinberg, M.: The Existence and Uniqueness of Steady States for a Classe of Chemical Reaction Networks. Arch. Rational Mech. Anal. 132, 311–370 (1995)

[15] Gatermann, K., Eiswirth, M., Sensse, A.: Toric ideals and graph theory to analyze Hopf bifurcations in mass action systems. Journal of Symbolic Computation 40(6), 1361–1382 (2005)
[16] Gillespie, C.S.: Moment-closure approximations for mass-action models. Systems Biology, IET 3(1) (2009)
[17] Gillespie, D.T.: Exact Stochastic Simulation of Coupled Chemical Reactions. Journal of Physical Chemistry 81(25), 2340–2361 (1977)
[18] Henri, V.: Lois générales de l'Action des Diastases. Hermann, Paris (1903)
[19] Kœrn, M., Elston, T.C., Blake, W.J., Collins, J.J.: Stochasticity in gene expression: from theories to phenotypes. Nature 6, 451–464 (2005)
[20] Kokotovic, P., Khalil, H.K., O'Reilly, J.: Singular Perturbation Methods in Control: Analysis and Design. Classics in Applied Mathematics, vol. 25. SIAM, Philadelphia (1999)
[21] Lemaire, F., Maza, M.M., Xie, Y.: The RegularChains library in MAPLE 10. In: Kotsireas, I.S. (ed.) The MAPLE Conference, pp. 355–368 (2005)
[22] Lemaire, F., Ürgüplü, A.: A method for semi-rectifying algebraic and differential systems using scaling type Lie point symmetries with linear algebra. In: Proceedings of ISSAC 2010, München, Germany, pp. 85–92 (August 2010)
[23] Lemaire, F., Ürgüplü, A.: Mabsys: Modeling and analysis of biological systems. In: Horimoto, K., Nakatsui, M., Popov, N. (eds.) Proceedings of Algebraic and Numeric Biology 2010, Castle of Hagenberg, Austria (August 2010)
[24] Lemaire, F., Ürgüplü, A.: The MABSys MAPLE package (2010), `http://www.lifl.fr/~lemaire/MABSys`
[25] Michaelis, L., Menten, M.: Die kinetik der invertinwirkung. Biochemische Zeitschrift 49, 333–369 (1973), Partial translation in english on `http://web.lemoyne.edu/~giunta/menten.html`
[26] Moroz, G., Rouillier, F.: DV: Un logiciel de classification des solutions réelles d'un système paramétré (2009), `http://www-spiral.lip6.fr/~moroz/fr/software.html`
[27] Murray, J.D.: Mathematical Biology I. An Introduction, 3rd edn. Interdisciplinary Applied Mathematics, vol. 17. Springer, Heidelberg (2002)
[28] Niu, W.: Qualitative Analysis of Biological Systems Using Algebraic Methods. PhD thesis, Université Paris VI, Paris (June 2011)
[29] Nöthen, A.L.: Quasistationarität und Fast-Invariante Mengen Gewönlicher Differentialgleichungen. PhD thesis, Rheinisch-Westfälischen Technischen Hochschule (2008)
[30] Othmer, H.G.: Analysis of Complex Reaction Networks in Signal Transduction, Gene Control and Metabolism (2006), `http://www.ricam.oeaw.ac.at/publications/download/summerschool/LectureNotes_Othmer.pdf`
[31] Paulsson, J.: Models of stochastic gene expression. Physics of Life Reviews 2, 157–175 (2005)
[32] Paulsson, J., Elf, J.: Stochastic Modeling of Intracellular Kinetics. In: Szallasi, Z., Stelling, J., Periwal, V. (eds.) System Modeling in Cellular Biology: From Concepts to Nuts and Bolts, pp. 149–175. The MIT Press, Cambridge (2006)
[33] Érdi, P., Tóth, J.: Mathematical models of chemical reactions: theory and applications of deterministic and stochastic models. Princeton University Press, Princeton (1989)
[34] Petitot, M.: The MAGNUS MAPLE software (2010), `http://www.lifl.fr/~petitot/recherche/exposes/ANB2010`

[35] Sedoglavic, A.: Reduction of Algebraic Parametric Systems by Rectification of Their Affine Expanded Lie Symmetries. In: Anai, H., Horimoto, K., Kutsia, T. (eds.) AB 2007. LNCS, vol. 4545, pp. 277–291. Springer, Heidelberg (2007)
[36] Sedoglavic, A., Ürgüplü, A.: Expanded Lie Point Symmetry, MAPLE package (2007), http://www.lifl.fr/~sedoglav/Software
[37] Singh, A., Hespanha, J.P.: Lognormal Moment Closures for Biochemical Reactions. In: Proceedings of the 45th IEEE Conference on Decision and Control, pp. 2063–2068 (2006)
[38] Ürgüplü, A.: Contribution to Symbolic Effective Qualitative Analysis of Dynamical Systems; Application to Biochemical Reaction Networks. PhD thesis, University Lille I, Lille, France (2010)
[39] Van Breusegem, V., Bastin, G.: Reduced order dynamical modelling of reaction systems: a singular perturbation approach. In: Proceedings of the 30th IEEE Conference on Decision and Control, Brighton, England, pp. 1049–1054 (December 1991)
[40] Vidal, S., Petitot, M., Boulier, F., Lemaire, F., Kuttler, C.: Models of Stochastic Gene Expression and Weyl Algebra. In: Horimoto, K., Nakatsui, M., Popov, N. (eds.) Proceedings of Algebraic and Numeric Biology 2010, Castle of Hagenberg, Austria, pp. 50–67 (August 2010)
[41] Vilar, J.M.G., Kueh, H.Y., Barkai, N., Leibler, S.: Mechanisms of noise-resistance in genetic oscillators. Proceedings of the National Academy of Science of the USA 99(9), 5988–5992 (2002)
[42] Xia, B.: DISCOVERER: A tool for solving semi-algebraic systems. ACM Commun. Comput. Algebra 41, 102–103 (2007)

On the Stability of Equilibrium Positions in the Circular Restricted Four-Body Problem

Dzmitry A. Budzko[1] and Alexander N. Prokopenya[2,3]

[1] Brest State University,
Kosmonavtov bul. 21, 224016, Brest, Belarus
master_booblik@tut.by
[2] Collegium Mazovia in Siedlce
ul. Sokolowska 161, 08-110, Siedlce, Poland
[3] Brest State Technical University,
Moskowskaya str. 267, 224017 Brest, Belarus
prokopenya@brest.by

Abstract. We consider the stability of equilibrium positions in the planar circular restricted four-body problem formulated on the basis of Lagrange's triangular solution of the three-body problem. The stability problem is solved in a strict nonlinear formulation on the basis of Arnold–Moser and Markeev theorems. Peculiar properties of the Hamiltonian normalization are discussed, and the influence of the third and fourth order resonances on stability of the equilibrium positions has been analyzed.

1 Introduction

The restricted many-body problem is a well-known model of celestial mechanics (see, for example, [1,2,3]), and equilibrium positions (or relative equilibrium positions) are its simple solutions which can be found in analytical form. As the stable equilibrium positions are highly interesting for applications, the stability problem has been attracting attention of most contributors to celestial mechanics during the past two hundred years. Finally, stability of equilibrium positions in the simplest case of the restricted three-body problem has been completely investigated (see [2]), and some general methods for studying the stability of the Hamiltonian systems have been developed [4,5,6]. However, application of these methods involves very bulky symbolic calculations which can be reasonably done only with computers and modern software such as the computer algebra system *Mathematica* [7], for example. Besides, stability analysis of more complicated dynamical systems requires improvement of available computing technique and designing new efficient algorithms of calculation, and this stimulates further investigations in this field.

The circular restricted four-body problem considered in the present paper is formulated similarly to the famous restricted three-body problem [1]. Three point particles $P_0,\ P_1,\ P_2$ having masses $m_0,\ m_1,\ m_2$, respectively, move uniformly on circular Keplerian orbits about their common center of mass. The particles are situated at the vertices of the equilateral triangle at any instant of time, and the corresponding exact solution of the general three-body problem is known as the Lagrange triangle. The fourth particle P_3 of negligible mass moves in the gravitational field of the primaries $P_0,\ P_1,\ P_2$, and the problem is to describe its motion. In our previous papers (see, for example, [8,9]) it was shown that the problem has eight equilibrium solutions but only three of them may be stable. We have determined the domains of their linear stability in the plane of the system parameters and proved that the three solutions are stable in the Liapunov sense for the majority of the parameters values from the domains of linear stability. But it turned out that for some values of the system parameters, the conditions of Arnold's theorem [4] are not fulfilled, and an analysis of the fifth and higher order terms in the Hamiltonian expansion is required for the entire solution of the stability problem. The main purpose of the present paper is to accomplish investigation of the stability of equilibrium solutions in the planar circular restricted four-body problem and to demonstrate the most important and useful algorithms for solving similar problems.

The paper is organized as follows. In Section 2, we describe equilibrium solutions and analyze their linear stability. Then we discuss the algorithm for normalization of the third order term in the Hamiltonian expansion (Section 3) and analyze stability of the equilibrium solutions under the third order resonance. In Section 4, we normalize the fourth order term of the Hamiltonian and apply theorems of Arnold and Markeev. And finally, in Section 5 we discuss stability of equilibrium solutions in the case when Arnold's theorem can not be applied.

2 Equilibrium Solutions and Their Linear Stability

In the rotating reference frame, where the particles $P_0,\ P_1,\ P_2$ rest in the xOy plane at the points $(0,0)$, $(1,0)$, $(1/2,\sqrt{3}/2)$, respectively, the Hamiltonian function of the system can be written in the form [9]

$$H = \frac{1}{2}\left(p_x^2+p_y^2\right) - xp_y + yp_x + \frac{1}{1+\mu_1+\mu_2}\left(\left(\mu_1+\frac{\mu_2}{2}\right)x + \frac{\mu_2\sqrt{3}}{2}y - \right.$$
$$\left. - \frac{1}{\sqrt{x^2+y^2}} - \frac{\mu_1}{\sqrt{(x-1)^2+y^2}} - \frac{2\mu_2}{\sqrt{(2x-1)^2+(2y-\sqrt{3})^2}}\right)\ , \quad (1)$$

where x, p_x and y, p_y are two pairs of canonically conjugate coordinate and momentum, and two mass parameters are given by

$$\mu_1 = m_1/m_0\ ,\quad \mu_2 = m_2/m_0\ .$$

Using the Hamiltonian (1), one can easily write the equations of motion of the particle P_3 and show that its equilibrium coordinates are determined by the system of two algebraic equations which can be represented in the form

$$(y-x\sqrt{3})\left(\frac{1}{(x^2+y^2)^{3/2}}-1\right) = \mu_1(y+\sqrt{3}(x-1))\left(\frac{1}{((x-1)^2+y^2)^{3/2}}-1\right),$$

$$\mu_2 = -\frac{2y\left(1-(x^2+y^2)^{-3/2}\right)}{(y+\sqrt{3}(x-1))\left(1-((x-1/2)^2+(y-\sqrt{3}/2)^2)^{-3/2}\right)} \ . \quad (2)$$

Each equation of system (2) determines a curve in the xOy plane, which can be easily visualized with the *Mathematica* built-in function *ContourPlot*, for example. So geometrically any equilibrium position of the particle P_3 corresponds to the point of intersection of two curves (Fig. 1).

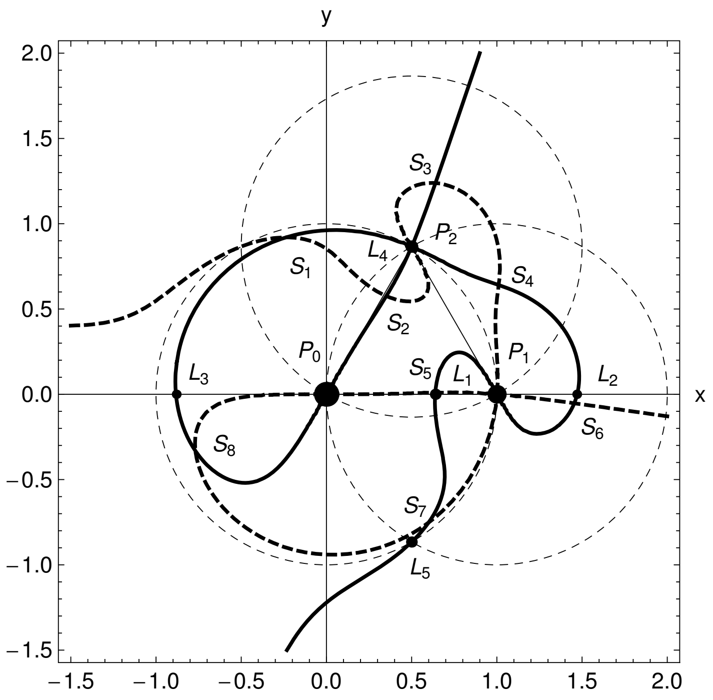


Fig. 1. Eight equilibrium positions $S_1, \ldots, S_8$ for $\mu_1 = 0.25$, $\mu_2 = 0.17$

Note that for any given $\mu_1 > 0$, the solid line in Fig. 1 determined by the first equation of system (2), is fixed and it always passes through the points P_0, P_1, P_2, and $(1/2, -\sqrt{3}/2)$. In case of $\mu_2 = 0$, the dashed line determined by the second equation of system (2) degenerates into the line $y = 0$ and the circle $x^2 + y^2 = 1$. Hence, system (2) has three roots at the Ox axis and two roots at the points P_2 and $(1/2, -\sqrt{3}/2)$, and these five roots correspond to the libration points L_1, L_2, L_3 and L_4, L_5 in the three-body problem (see [1], [2]). Increasing the value of μ_2, one can observe that the three equilibrium points located at the Ox axis when $\mu_2 = 0$, as well as the equilibrium point $(1/2, -\sqrt{3}/2)$, gradually move in the xOy plane along the solid line (the points $S_5 - S_8$ in Fig. 1). The point $x = 1/2$, $y = \sqrt{3}/2$ generates four new equilibrium positions (the points $S_1 - S_4$ in Fig. 1), one by each branch of the solid line outgoing the point P_2.

Thus, graphical analysis shows that there are eight roots of system (2) for small values of parameters μ_1, μ_2.

To localize each equilibrium position for any given values of parameters μ_1, μ_2, we have to find all solutions of equations (2). As these algebraic equations are nonlinear it is very difficult, if possible at all, to find their solutions in analytical form. In such a case it is natural to look for numerical solutions using the built-in *Mathematica* function *FindRoot*, for example. However, implementing any numerical method, we have to specify a starting point or zero approximation for every root, and the result depends substantially on the starting point because system (2) has several solutions. So it is not sufficient to choose the starting point somewhere in the domain, where the point S_1 is located, for example, to be sure that the function *FindRoot* will obtain the solution in the same domain as the result. One can assert only that if the starting point is chosen close enough to the corresponding equilibrium position then the function *FindRoot* will find it surely because solutions of the system (2) depend on parameters μ_1 and μ_2 continuously.

Note that for a given value of μ_1 and small values of μ_2, the roots of system (2) can be found in the form of a power series as

$$x = x_0 + x_1\mu_2^k + x_2\mu_2^{2k} + \ldots, \quad y = y_0 + y_1\mu_2^k + y_2\mu_2^{2k} + \ldots\ , \tag{3}$$

where the zero approximation (x_0, y_0) is determined by equations (2) under $\mu_2 = 0$. The corresponding algorithm of calculation and its implementation with *Mathematica* is described in [10]. This approach makes it possible to separate anyone of the eight equilibrium points $S_1, \ldots, S_8$ and to investigate its position and stability depending on the parameters μ_1, μ_2.

Denoting an equilibrium position of the particle P_3 in the xOy plane by (x_0, y_0), we can expand the Hamiltonian (1) in the Taylor series in the neighborhood of equilibrium point and represent it in the form

$$H = H_2 + H_3 + H_4 + \ldots\ , \tag{4}$$

where H_k is the kth order homogeneous polynomial with respect to canonical variables x, y, p_x, p_y. Note that zero-order term H_0 in (4) has been omitted as a constant, which doesn't influence the equations of motion, and the first-order term H_1 is equal to zero owing to equations determining equilibrium positions. Therefore, the first non-zero term in the expansion (4) is a quadratic one that is

$$H_2 = \frac{1}{2}\left(p_x^2 + p_y^2\right) - p_y x + p_x y + h_{20}x^2 + h_{11}xy + h_{02}y^2. \tag{5}$$

The corresponding expressions for coefficients h_{20}, h_{11}, and h_{02} are easily found (see [8,9]).

One can readily check that the linearized equations of motion determined by the quadratic part H_2 of the Hamiltonian (5) form the fourth-order linear system of differential equations with constant coefficients. Characteristic exponents $\lambda_1, \ldots, \lambda_4$ for such a system can be easily found (see [8]) and are represented in the form

$$\lambda_{1,2} = \pm i\sigma_1, \quad \lambda_{3,4} = \pm i\sigma_2, \tag{6}$$

where the frequencies σ_1, σ_2 are given by

$$\sigma_{1,2} = \left(1 + h_{20} + h_{02} \pm \sqrt{h_{20}^2 + h_{02}^2 + h_{11}^2 - 2h_{20}h_{02} + 4h_{20} + 4h_{02}}\right)^{1/2}. \quad (7)$$

Note that the Lagrange triangle may be stable only if parameters μ_1, μ_2 satisfy the condition (see [1])

$$27(\mu_1 + \mu_2 + \mu_1\mu_2) < (1 + \mu_1 + \mu_2)^2. \quad (8)$$

So it makes sense to analyze stability of equilibrium solutions in the restricted four-body problem only in the domain determined by inequality (8). Analysis of the frequencies (7) in this domain for all eight equilibrium positions (see Fig. 1) has shown [8] that for the points $S_2,\ S_3,\ S_5,\ S_6,\ S_8$ at least one frequency has an imaginary part for any values of parameters μ_1 and μ_2. Therefore, these five equilibrium positions are unstable. Equilibrium points S_1, S_4 and S_7 are stable in linear approximation if parameters μ_1, μ_2 are smaller than their values on the stability boundaries which are determined from the condition $\sigma_1 = \sigma_2$. The corresponding curve for the equilibrium point S_1 is shown as the dashed curve in the $\mu_1 O \mu_2$ plane in Fig. 2.

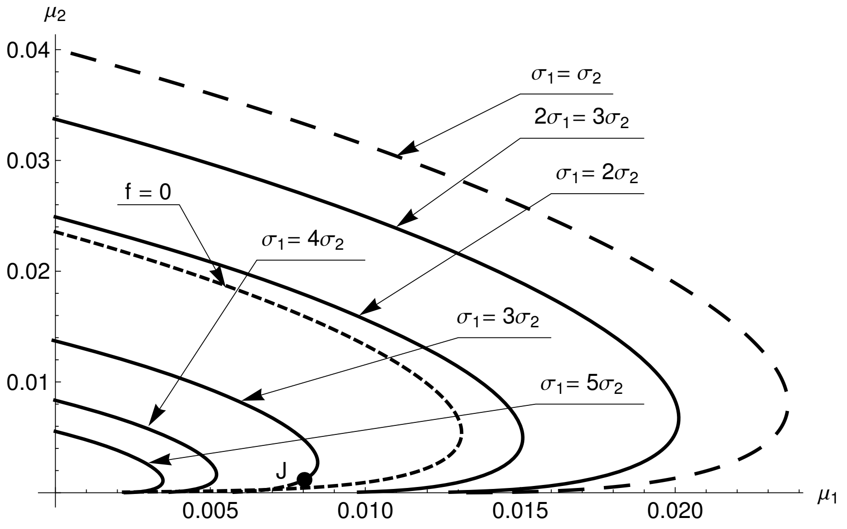


Fig. 2. Stability domain and resonance curves for equilibrium position S_1

3 Normalization of the Third-Order Term H_3

It is well known that the stability problem for the Hamiltonian system of differential equations belongs to the critical case, in Liapunov's sense [11], and it can be resolved only in a strict nonlinear formulation. The most general approach to studying such systems is the Poincaré method of normal forms that was used successfully in solving many problems of nonlinear mechanics (see [6]). According to this method we have to construct the canonical transformation that reduces the Hamiltonian function to the Birkhoff normal form [12] when the equations of motion can be solved. First step in constructing such transformation is the normalization of the quadratic part H_2 in the Hamiltonian expansion (4), and we do this applying an algorithm proposed in [2] and described in detail in [13]. Doing necessary symbolic calculations, we obtain the second order term H_2 in the form

$$H_2 = \frac{1}{2}\left(\sigma_1(p_1^2+q_1^2) - \sigma_2(p_2^2+q_2^2)\right) \ , \tag{9}$$

where p_1, q_1 and p_2, q_2 are two pairs of new canonically conjugated variables.

It should be emphasized that the quadratic form (9) is neither positive nor negative definite function and, hence, one cannot conclude on stability or instability of equilibrium solutions using the principle of linearized stability [11]. Therefore, the stability problem can be solved only in a strict nonlinear formulation. Note also that the stability analysis of the equilibrium positions S_1, S_4, S_7 is done in a similar way. So we'll analyze only the stability of the point S_1 in detail.

To normalize the third-order term H_3 in the Hamiltonian (4) we use the method of constructing the real-valued canonical transformation of Birkhoff's type described in [14]. It should be noted that, in contrast to [14], the system under consideration has two parameters μ_1 and μ_2, and due to this reason the calculations are much more bulky and difficult (see [9]). After realization of the first canonical transformation normalizing the quadratic part H_2, the third-order term H_3 takes the form

$$H_3 = \sum_{i+j+k+l=3} h^{(3)}_{ijkl} q_1^i q_2^j p_1^k p_2^l \ . \tag{10}$$

The corresponding expressions for $h^{(3)}_{ijkl}$ are quite bulky, so we do not write them here. And we would like to find such canonical transformation that the third order term H_3 in expansion (4) was eliminated. Generating function for such transformation can be sought in the form

$$S(\tilde{p}_1, \tilde{p}_2, q_1, q_2) = q_1\tilde{p}_1 + q_2\tilde{p}_2 + \sum_{i+j+k+l=3} s^{(3)}_{ijkl} q_1^i q_2^j \tilde{p}_1^k \tilde{p}_2^l \ , \tag{11}$$

where coefficients $s^{(3)}_{ijkl}$ are to be found. Then new momenta $\tilde{p}_1, \tilde{p}_2$ and coordinates $\tilde{q}_1, \tilde{q}_2$ are determined by the following relationships

$$\tilde{q}_1 = \frac{\partial S}{\partial \tilde{p}_1} \ , \ \tilde{q}_2 = \frac{\partial S}{\partial \tilde{p}_2} \ , \ p_1 = \frac{\partial S}{\partial q_1} \ , \ p_2 = \frac{\partial S}{\partial q_2} \ . \tag{12}$$

Note that these relationships can be considered as equations with respect to the old canonical variables q_1, q_2, p_1, p_2. On substituting (11) into (12) and solving these equations, we find q_1, q_2, p_1, p_2 in the form of second-degree polynomials in new canonical variables $\tilde{q}_1$, $\tilde{q}_2$, $\tilde{p}_1$, $\tilde{p}_2$. Then we substitute the corresponding expressions into (9), (10) and expand the Hamiltonian $H = H_2 + H_3$ into the Taylor series in powers of $\tilde{q}_1$, $\tilde{q}_2$, $\tilde{p}_1$, $\tilde{p}_2$. The expression obtained is again represented as a sum of homogeneous polynomials $\tilde{H}_k$ $(k = 2, 3, ...)$ with respect to canonical variables $\tilde{q}_1$, $\tilde{q}_2$, $\tilde{p}_1$, $\tilde{p}_2$. One can readily check that the second-order term $\tilde{H}_2$ preserves the form (9), while the third-order term $\tilde{H}_3$ is a sum of twenty terms of the form

$$\tilde{h}^{(3)}_{ijkl}\tilde{q}_1^i\tilde{q}_2^j\tilde{p}_1^k\tilde{p}_2^l \quad (i + j + k + l = 3) \tag{13}$$

with new coefficients $\tilde{h}^{(3)}_{ijkl}$ which are expressed as linear functions of old coefficients $h^{(3)}_{ijkl}$ and unknown coefficients $s^{(3)}_{ijkl}$ of the generating function (11). Obviously, the third-order term $\tilde{H}_3$ would be eliminated if all the coefficients $\tilde{h}^{(3)}_{ijkl}$ in (13) were equal to zero. Therefore, we can try to solve the system of twenty equations of the form $\tilde{h}^{(3)}_{ijkl} = 0$ and to find the coefficients $s^{(3)}_{ijkl}$ of the corresponding canonical transformation (12).

Analysis of the coefficients $\tilde{h}^{(3)}_{ijkl}$ shows that in fact we have five independent subsystems for determination of the coefficients $s^{(3)}_{ijkl}$. As all these subsystems are solved similarly, we consider only one of them that is formed by three coefficients of $\tilde{p}_1\tilde{p}_2^2$, $\tilde{p}_1\tilde{q}_2^2$, $\tilde{q}_1\tilde{q}_2\tilde{p}_2$ in the expression for $\tilde{H}_3$. It determines the coefficients $s^{(3)}_{0111}$, $s^{(3)}_{1002}$, $s^{(3)}_{1200}$ and is given by

$$\tilde{h}^{(3)}_{0012} = h^{(3)}_{0012} + s^{(3)}_{1002}\sigma_1 - s^{(3)}_{0111}\sigma_2 \ ,$$

$$\tilde{h}^{(3)}_{0210} = h^{(3)}_{0210} + s^{(3)}_{1200}\sigma_1 + s^{(3)}_{0111}\sigma_2 \ ,$$

$$\tilde{h}^{(3)}_{1101} = h^{(3)}_{1101} - s^{(3)}_{0111}\sigma_1 + 2s^{(3)}_{1002}\sigma_2 - 2s^{(3)}_{1200}\sigma_2 \ . \tag{14}$$

Note that coefficients $s^{(3)}_{0111}$, $s^{(3)}_{1002}$, and $s^{(3)}_{1200}$ appear only in the expressions for $\tilde{h}^{(3)}_{ijkl}$ given in (14) and so they are completely determined by this system. It has a unique solution for any values of $\tilde{h}^{(3)}_{ijkl}$ if its determinant being equal to $\sigma_1(4\sigma_2^2 - \sigma_1^2)$ is not zero. In such a case, we can set $\tilde{h}^{(3)}_{0012} = \tilde{h}^{(3)}_{0210} = \tilde{h}^{(3)}_{1101} = 0$ and find the corresponding coefficients $s^{(3)}_{0111}$, $s^{(3)}_{1002}$, and $s^{(3)}_{1200}$. Therefore, if $\sigma_1 \neq 0$ and the conditions

$$\sigma_1 \pm 2\sigma_2 \neq 0 \tag{15}$$

are fulfilled three terms (13) with coefficients $\tilde{h}^{(3)}_{0012}$, $\tilde{h}^{(3)}_{0210}$, and $\tilde{h}^{(3)}_{1101}$ are eliminated in $\tilde{H}_3$ by means of the canonical transformation (12).

An analysis of the rest of coefficients $\tilde{h}^{(3)}_{ijkl}$ shows that if $\sigma_2 \neq 0$ and the conditions

$$2\sigma_1 \pm \sigma_2 \neq 0 \tag{16}$$

are fulfilled, in addition to (15), all the coefficients $s^{(3)}_{ijkl}$ of the canonical transformation (12) are found in a unique way. In this case, we can set $\tilde{h}^{(3)}_{ijkl} = 0$

and find the canonical transformation such that the third-order term $\tilde{H}_3$ in the Hamiltonian (4) vanishes.

Note that conditions (15) and (16) mean an absence of the third-order resonances of frequencies in the system (see [6]). Analyzing frequencies (7), we obtain that for linearly stable equilibrium point S_1 there exist such values of parameters μ_1, μ_2, for which the condition of third-order resonance $\sigma_1 - 2\sigma_2 = 0$ is fulfilled (see Fig. 2). Thus, for the points (μ_1, μ_2) in the $\mu_1 O \mu_2$ plane located on the corresponding resonance curve the condition (15) is not fulfilled and, hence, system (14) does not have a solution in case of $\tilde{h}^{(3)}_{ijkl} = 0$. Due to the same reason the coefficients $\tilde{h}^{(3)}_{1002}$, $\tilde{h}^{(3)}_{1200}$, $\tilde{h}^{(3)}_{0111}$ can not be eliminated under the third-order resonance, as well. It means that the corresponding six resonance terms in $\tilde{H}_3$ cannot be eliminated.

Nevertheless, we can require the following conditions to be fulfilled

$$\tilde{h}^{(3)}_{0012} = \frac{B_1}{2\sqrt{2}}, \quad \tilde{h}^{(3)}_{0210} = -\frac{B_1}{2\sqrt{2}}, \quad \tilde{h}^{(3)}_{1101} = -\frac{B_1}{\sqrt{2}}, \tag{17}$$

$$\tilde{h}^{(3)}_{0111} = \frac{B_2}{\sqrt{2}}, \quad \tilde{h}^{(3)}_{1002} = \frac{B_2}{2\sqrt{2}}, \quad \tilde{h}^{(3)}_{1200} = -\frac{B_2}{2\sqrt{2}}, \tag{18}$$

where B_1, B_2 are some constants. Solving the systems of equations (17), (18), we obtain the corresponding coefficients $s^{(3)}_{ijkl}$ of the canonical transformation (12) and find the constants B_1 and B_2 as

$$B_1 = \frac{1}{\sqrt{2}}(h^{(3)}_{0012} - h^{(3)}_{0210} - h^{(3)}_{1101}), \quad B_2 = \frac{1}{\sqrt{2}}(h^{(3)}_{0111} + h^{(3)}_{1002} - h^{(3)}_{1200}). \tag{19}$$

Then the Hamiltonian (4) takes a form

$$\tilde{H} = \frac{1}{2}\sigma_1\left(\tilde{q}_1^2 + \tilde{p}_1^2\right) - \frac{1}{2}\sigma_2\left(\tilde{q}_2^2 + \tilde{p}_2^2\right) + \frac{B_1}{2\sqrt{2}}\left(\tilde{p}_1\tilde{p}_2^2 - \tilde{p}_1\tilde{q}_2^2 - 2\tilde{q}_1\tilde{q}_2\tilde{p}_2\right) + $$

$$+ \frac{B_2}{2\sqrt{2}}\left(\tilde{q}_1\tilde{p}_2^2 - \tilde{q}_1\tilde{q}_2^2 + 2\tilde{q}_2\tilde{p}_1\tilde{p}_2\right) + \tilde{H}_4 + \ldots. \tag{20}$$

Using the standard canonical transformation

$$\tilde{q}_1 = \sqrt{2\tau_1}\sin(\varphi_1 + \alpha)\ , \ \tilde{p}_1 = \sqrt{2\tau_1}\cos(\varphi_1 + \alpha)\ ,$$

$$\tilde{q}_2 = \sqrt{2\tau_2}\sin\varphi_2\ , \ \tilde{p}_2 = \sqrt{2\tau_2}\cos\varphi_2\ , \tag{21}$$

where parameter α is determined by the relationships

$$\cos\alpha = \frac{B_1}{B}, \quad \sin\alpha = \frac{B_2}{B}, \quad B = \sqrt{B_1^2 + B_2^2},$$

we rewrite the Hamiltonian (20) as

$$\tilde{H} = \sigma_1\tau_1 - \sigma_2\tau_2 + B\tau_2\sqrt{\tau_1}\cos(\varphi_1 + 2\varphi_2) + \tilde{H}_4(\varphi_1, \varphi_2, \tau_1, \tau_2) + \ldots\ . \tag{22}$$

We have done numerical analysis of parameter B for the equilibrium point S_1 under the third-order resonance and shown that it is not equal to zero for all points (μ_1, μ_2) belonging to the resonance curve. Therefore, applying Markeev's theorem [6], we can conclude that the equilibrium point S_1 in the circular restricted four-body problem, formulated on the basis of Lagrange's triangular solutions, is unstable under third-order resonance of the form $\sigma_1 = 2\sigma_2$.

4 Normalizing the Fourth-Order Term H_4

Let us assume that the condition $\sigma_1 \neq 2\sigma_2$ is fulfilled, and there is no resonance in the system up to the third order inclusively. Then after normalization of the second and third order terms we obtain the Hamiltonian (4) in the form

$$\tilde{H} = \tilde{H}_2 + \tilde{H}_4 + \dots \ , \tag{23}$$

where the second-order term

$$\tilde{H}_2 = \frac{1}{2}\left(\sigma_1(\tilde{p}_1^2 + \tilde{q}_1^2) - \sigma_2(\tilde{p}_2^2 + \tilde{q}_2^2)\right) \tag{24}$$

has the normal form, the third-order term $\tilde{H}_3$ is absent, and the fourth-order term $\tilde{H}_4$ may be written as

$$\tilde{H}_4 = \sum_{i+j+k+l=4} \tilde{h}^{(4)}_{ijkl} \tilde{q}_1^i \tilde{q}_2^j \tilde{p}_1^k \tilde{p}_2^l \ . \tag{25}$$

The sum (25) contains 35 terms but coefficients $\tilde{h}^{(4)}_{ijkl}$ are quite cumbersome, and we do not write them here. Again we look for the function

$$S(p_1^*, p_2^*, \tilde{q}_1, \tilde{q}_2) = \tilde{q}_1 p_1^* + \tilde{q}_2 p_2^* + \sum_{i+j+k+l=4} s^{(4)}_{ijkl} \tilde{q}_1^i \tilde{q}_2^j p_1^{*k} p_2^{*l} \ , \tag{26}$$

generating the canonical transformation reducing the fourth-order term $\tilde{H}_4$ to the simplest form. New momenta p_1^*, p_2^* and coordinates q_1^*, q_2^* are determined by the relationships

$$q_1^* = \frac{\partial S}{\partial p_1^*} \ , \ q_2^* = \frac{\partial S}{\partial p_2^*} \ , \ \tilde{p}_1 = \frac{\partial S}{\partial \tilde{q}_1} \ , \ \tilde{p}_2 = \frac{\partial S}{\partial \tilde{q}_2} \ . \tag{27}$$

Resolving (27) with respect to the old canonical variables $\tilde{q}_1$, $\tilde{q}_2$, $\tilde{p}_1$, $\tilde{p}_2$ in the neighborhood of the point $q_1^* = q_2^* = p_1^* = p_2^* = 0$ and substituting the solution into (23), we expand the Hamiltonian $\tilde{H}$ in the Taylor series in powers of q_1^*, q_2^*, p_1^*, p_2^*. Obviously, the second-order term H_2^* in this expansion again has the form (24), the third-order term H_3^* is absent, and the fourth-order term H_4^* is a sum of 35 terms of the form

$$h^{*(4)}_{ijkl} q_1^{*i} q_2^{*j} p_1^{*k} p_2^{*l} \ \ (i+j+k+l=4) \ ,$$

where new coefficients $h^{*(4)}_{ijkl}$ are linear functions of unknown coefficients $s^{(4)}_{ijkl}$ determining the generating function (26).

An analysis of the coefficients $h^{*(4)}_{ijkl}$ shows that they are again divided into several independent groups, and each group forms a system of equations determining some coefficients $s^{(4)}_{ijkl}$. If the following conditions

$$\sigma_1 \neq 0, \quad \sigma_2 \neq 0, \quad \sigma_1 \pm \sigma_2 \neq 0, \quad \sigma_1 \pm 3\sigma_2 \neq 0, \quad 3\sigma_1 \pm \sigma_2 \neq 0 \ , \tag{28}$$

are fulfilled we can solve the equations $h^{*(4)}_{ijkl} = 0$ and find coefficients $s^{(4)}_{ijkl}$ of the canonical transformation (27) eliminating the corresponding terms in (25). Nevertheless, there are ten terms in the expansion (25) which can not be eliminated. They can be only simplified in such a way that the fourth-order term $\tilde{H}_4$ takes the form

$$H^*_4 = \frac{1}{4}\left(c_{20}(p_1^{*2}+q_1^{*2})^2 + c_{11}(p_1^{*2}+q_1^{*2})(p_2^{*2}+q_2^{*2}) + c_{02}(p_2^{*2}+q_2^{*2})^2\right) \ .$$

Then, using the standard canonical transformation

$$\begin{aligned} q_1^* &= \sqrt{2\tau_1}\sin\varphi_1 \ , \ p_1^* = \sqrt{2\tau_1}\cos\varphi_1 \ , \\ q_2^* &= \sqrt{2\tau_2}\sin\varphi_2 \ , \ p_2^* = \sqrt{2\tau_2}\cos\varphi_2 \ , \end{aligned} \tag{29}$$

we rewrite the Hamiltonian (23) as

$$H = \sigma_1\tau_1 - \sigma_2\tau_2 + c_{20}\tau_1^2 + c_{11}\tau_1\tau_2 + c_{02}\tau_2^2 + H^*_5(\varphi_1,\varphi_2,\tau_1,\tau_2) + \dots \ . \tag{30}$$

Recall that Arnold's theorem [4] states that in the case of absence of resonances up to the fourth order inclusively (conditions (15),(16),(28)) equilibrium solutions are stable if

$$f = c_{20}\sigma_2^2 + c_{11}\sigma_1\sigma_2 + c_{02}\sigma_1^2 \neq 0 \ . \tag{31}$$

Numerical analysis of parameter f shows that for equilibrium point S_1, there exist such values of parameters μ_1, μ_2, for which $f = 0$ (see Fig. 2). For such μ_1, μ_2 the fifth- and higher-order terms in the expansion of the Hamiltonian (4) need to be analyzed to conclude on stability or instability of equilibrium solution.

Besides, there is a curve in the $\mu_1 O \mu_2$ plane, where the condition of fourth-order resonance of the form $\sigma_1 = 3\sigma_2$ is fulfilled. In this case, eight additional terms appear in the expression for H^*_4 because the following coefficients $h^{*(4)}_{ijkl}$ do not vanish and are expressed via two parameters A_1, A_2

$$\begin{aligned} h^{*(4)}_{0013} &= -\frac{1}{3}h^{*(4)}_{0211} = -\frac{1}{3}h^{*(4)}_{1102} = h^{*(4)}_{1300} = \frac{A_1}{4} \ , \\ h^{*(4)}_{1003} &= \frac{1}{3}h^{*(4)}_{0112} = -\frac{1}{3}h^{*(4)}_{1201} = -h^{*(4)}_{0310} = \frac{A_2}{4} \ . \end{aligned} \tag{32}$$

Parameters A_1, A_2 are obtained as the solutions of system (32) and are given by

$$A_1 = \frac{1}{2}(\tilde{h}^{(4)}_{0013} - \tilde{h}^{(4)}_{0211} - \tilde{h}^{(4)}_{1102} + \tilde{h}^{(4)}_{1300}),$$

$$A_2 = \frac{1}{2}(\tilde{h}^{(4)}_{0112} - \tilde{h}^{(4)}_{0310} + \tilde{h}^{(4)}_{1003} - \tilde{h}^{(4)}_{1201}).$$

As a result, the Hamiltonian (23) is reduced to the form

$$H^* = \frac{3\sigma_2}{2}\left(p_1^{*2} + q_1^{*2}\right) - \frac{\sigma_2}{2}\left(p_2^{*2} + q_2^{*2}\right) +$$

$$+\frac{1}{4}\left(c_{20}(p_1^{*2} + q_1^{*2})^2 + c_{11}(p_1^{*2} + q_1^{*2})(p_2^{*2} + q_2^{*2}) + c_{02}(p_2^{*2} + q_2^{*2})^2\right) +$$

$$+\frac{A_1}{4}\left(p_1^* p_2^{*3} - 3q_2^{*2}p_1^*p_2^* - 3q_1^*q_2^*p_2^{*2} + q_1^*q_2^{*3}\right) +$$

$$+\frac{A_2}{4}\left(q_1^* p_2^{*3} - 3q_1^*q_2^{*2}p_2^* + 3q_2^*p_1^*p_2^{*2} - p_1^*q_2^{*3}\right) \ . \tag{33}$$

Then doing the transformation (21), relating to the canonical variables q_1^*, q_2^*, p_1^*, p_2^*, we rewrite the Hamiltonian (33) as

$$H = 3\sigma_2\tau_1 - \sigma_2\tau_2 + c_{20}\tau_1^2 + c_{11}\tau_1\tau_2 + c_{02}\tau_2^2 +$$

$$+\tau_2\sqrt{\tau_1\tau_2(A_1^2 + A_2^2)}\cos(\varphi_1 + 3\varphi_2) + H_5^*(\varphi_1, \varphi_2, \tau_1, \tau_2) + \ldots \ .$$

According to the theorem of Markeev [2], stability of the equilibrium solutions under the fourth-order resonance depends on the values of $c_{20} + 3c_{11} + 9c_{02}$ and $3\sqrt{3(A_1^2 + A_2^2)}$. Our calculations show that in case of equilibrium point S_1, the inequality

$$c_{20} + 3c_{11} + 9c_{02} > 3\sqrt{3(A_1^2 + A_2^2)}$$

takes place if for $\mu_2 > 0.001178$ and the point (μ_1, μ_2) belongs to the curve $\sigma_1 = 3\sigma_2$ in Fig. 2. Therefore, on the basis of the Arnold and Markeev theorems we can conclude that the equilibrium position S_1 is stable in Liapunov's sense for any values of parameters μ_1, μ_2 from the domain of its linear stability in the $\mu_1 O \mu_2$ plane, except for the curves $\sigma_1 = 2\sigma_2$, $f = 0$ and part of the curve $\sigma_1 = 3\sigma_2$ below the point **J** in Fig. 2.

5 Stability Analysis in the Case of $f = 0$

Let us consider the points (μ_1, μ_2) belonging to the curve $f = c_{20}\sigma_2^2 + c_{11}\sigma_1\sigma_2 + c_{02}\sigma_1^2 = 0$ in the plane $\mu_1 O \mu_2$. For the corresponding values of parameters μ_1, μ_2, the Arnold's theorem can not be applied, and the analysis of the fifth- and higher-order terms in the Hamiltonian expansion (4) is required. Note that these terms are normalized similarly to the cases of normalization of H_3 and H_4. To normalize the fifth-order term, for example, we have to look for the generating function of the corresponding canonical transformation in the form of the fifth-order polynomial. Analyzing coefficients $h^{(5)}_{ijkl}$ in the expression for H_5, one can show that in the absence of resonances up to the fifth order inclusively there exist such coefficients of the generating function that the term H_5 in the Hamiltonian (4) vanishes.

Similar analysis of the sixth-order term H_6 in the Hamiltonian (4) shows that it can not be entirely eliminated but in the absence of resonances up to the sixth order inclusively there exists a canonical transformation reducing it to the form

$$H_6 = c_{30}\tau_1^3 + c_{21}\tau_1^2\tau_2 + c_{12}\tau_1\tau_2^2 + c_{03}\tau_2^3 \ . \tag{34}$$

Then we can verify the condition

$$d = c_{30}\sigma_2^3 + c_{21}\sigma_2^2\sigma_1 + c_{12}\sigma_2\sigma_1^2 + c_{03}\sigma_1^3 \neq 0 \tag{35}$$

at the points (μ_1, μ_2), where $f = c_{20}\sigma_2^2 + c_{11}\sigma_1\sigma_2 + c_{02}\sigma_1^2 = 0$. Our calculation has shown that condition (35) is fulfilled for all points at the curve $f = 0$, except for the point $(0.00679, 0.000271)$, where this curve crosses the resonance curve $\sigma_1 = 3\sigma_2$ (these curves are shown in larger scale in Fig. 3). According to the Markeev theorem [2], it means that the equilibrium point S_1 is stable for all points of the curve $f = 0$, except for the points, where this curve crosses the resonance curves shown in Fig. 3.

Remind that for $\mu_2 < 0.001178$, the equilibrium point S_1 is unstable owing to fourth-order resonance. Therefore, it is unstable at the point $(0.00679, 0.000271)$, as well. At the points $(0.00398, 0.000105)$, $(0.00264, 0.000065)$ in the plane $\mu_1 O \mu_2$, where the curve $f = 0$ crosses the resonance curves $\sigma_1 = 4\sigma_2$ and $\sigma_1 = 5\sigma_2$, conditions of the Markeev theorem are not fulfilled, and the stability problem for the corresponding values of parameters μ_1, μ_2 is still open.

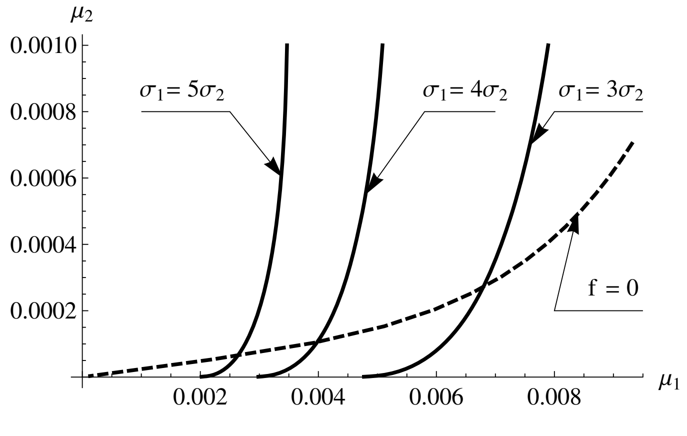


Fig. 3. Points of intersection of the curve $f = 0$ and the resonance curves

6 Conclusion

In the present paper, we have completely studied the stability of the equilibrium point S_1 in the planar circular restricted four-body problem formulated on the basis of the Lagrange triangular solution of the three-body problem. We proved that this equilibrium point is stable in Liapunov's sense for any values of parameters μ_1, μ_2 from the domain of its linear stability in the $\mu_1 O \mu_2$ plane, except for the resonance curve $\sigma_1 = 2\sigma_2$ and a part of the resonance curve $\sigma_1 = 3\sigma_2$ shown in Fig. 2. Besides, the stability problem is not solved for two points $(0.00398, 0.000105)$, $(0.00264, 0.000065)$ in the $\mu_1 O \mu_2$ plane, where the curve $f = 0$ crosses the resonance curves $\sigma_1 = 4\sigma_2$ and $\sigma_1 = 5\sigma_2$.

Note that all numerical and symbolic calculations and visualization of the obtained results have been done with the computer algebra system *Mathematica*. And all the calculations can be easily repeated for other equilibrium points.

References

1. Szebehely, V.: Theory of Orbits. The Restricted Problem of Three Bodies. Academic Press, New York (1967)
2. Markeev, A.P.: Libration Points in Celestial Mechanics and Cosmodynamics. Nauka, Moscow (1978) (in Russian)
3. Whipple, A.L., Szebehely, V.: The restricted problem of $n + \nu$ bodies. Celestial Mechanics 32, 137–144 (1984)
4. Arnold, V.I.: Small denominators and problems of stability of motion in classical and celestial mechanics. Uspekhi Math. Nauk. 18(6), 91–192 (1963) (in Russian)
5. Moser, J.: Lectures on the Hamiltonian Systems. Mir, Moscow (1973) (in Russian)
6. Markeev, A.P.: Stability of the Hamiltonian systems. In: Matrosov, V.M., Rumyantsev, V.V., Karapetyan, A.V. (eds.) Nonlinear Mechanics, pp. 114–130. Fizmatlit, Moscow (2001) (in Russian)
7. Wolfram, S.: The Mathematica Book, 4th edn. Wolfram Media/Cambridge University Press (1999)
8. Budzko, D.A.: Linear stability analysis of equilibrium solutions of restricted planar four-body problem. In: Gadomski, L., et al. (eds.) Computer Algebra Systems in Teaching and Research. Evolution, Control and Stability of Dynamical Systems, pp. 28–36. The College of Finance and Management, Siedlce (2009)
9. Budzko, D.A., Prokopenya, A.N.: Stability analysis of equilibrium solutions in the planar circular restricted four-body problem. In: Gadomski, L., et al. (eds.) Computer Algebra Systems in Teaching and Research. Differential Equations, Dynamical Systems and Celestial Mechanics, pp. 141–159. Wydawnictwo Collegium Mazovia, Siedlce (2011)
10. Budzko, D.A., Prokopenya, A.N.: Symbolic-numerical analysis of equilibrium solutions in a restricted four-body problem. Programming and Computer Software 36(2), 68–74 (2010)
11. Liapunov, A.M.: General Problem about the Stability of Motion. Gostekhizdat, Moscow (1950) (in Russian)
12. Birkhoff, G.D.: Dynamical Systems. GITTL, Moscow (1941) (in Russian)
13. Budzko, D.A., Prokopenya, A.N., Weil, J.A.: Quadratic normalization of the Hamiltonian in restricted four-body problem. Vestnik BrSTU. Physics, Mathematics, Informatics (5), 82–85 (2009) (in Russian)
14. Gadomski, L., Grebenikov, E.A., Prokopenya, A.N.: Studying the stability of equilibrium solutions in the planar circular restricted four-body problem. Nonlinear Oscillations 10(1), 66–82 (2007)

Semi-algebraic Description of the Equilibria of Dynamical Systems

Changbo Chen and Marc Moreno Maza

The University of Western Ontario, London N6A 1M8, Canada

Abstract. We study continuous dynamical systems defined by autonomous ordinary differential equations, themselves given by parametric rational functions. For such systems, we provide semi-algebraic descriptions of their hyperbolic and non-hyperbolic equilibria, their asymptotically stable hyperbolic equilibria, their Hopf bifurcations. To this end, we revisit various criteria on sign conditions for the roots of a real parametric univariate polynomial. In addition, we introduce the notion of *comprehensive triangular decomposition* of a semi-algebraic system and demonstrate that it is well adapted for our study.

1 Introduction

The study of polynomial dynamical systems by means of symbolic computation is one of the most popular application of computer algebra. Equilibria, limit cycles, center manifolds, normal forms and bifurcation analysis are the main notions used in the study of dynamical systems [30, 4, 21, 33]. These objects can be manipulated by a variety of symbolic methods [11,9,10,39,13,34,27,23, 24, 35, 19, 17, 16, 22, 36, 5, 31]. Among these notions, those which have received the greatest attention by the computer algebra community are equilibria and bifurcation analysis. Studying them for polynomial dynamical systems typically consists of: (1) setting up a (parametric) semi-algebraic system $\mathcal{S}$, (2) extracting from $\mathcal{S}$ a particular information.

The aim of this paper is twofold. Our first objective is to revisit the results that are practically useful for conducting equilibrium and bifurcation by means of symbolic computation. These results are gathered in Sections 2 and 3. They are generally stated in terms of the coefficients of a univariate polynomial and translate into semi-algebraic systems. A prototype of such results is the Routh-Hurwitz's criterion. While many of these criteria appear in the literature (for instance in [23, 24]) we also provide some new criteria, like Theorem 9, as well as new interpretation of classical results, like Theorem 13.

Our second objective is to exhibit tools that are well adapted for solving the semi-algebraic systems implementing the above mentioned results. Typical problems on parametric dynamical systems (see Problems 1, 2, 3) require to decompose the parameter space into connected semi-algebraic sets above which the qualitative behavior of the dynamical system is essentially constant. Taking also into consideration the fact that certain degenerated behaviors have no practical interest, we introduce, in Section 4, the notion of a *comprehensive triangular decomposition of a parametric semi-algebraic system*, together with an algorithm for computing it. This work extends some of our previous papers [6,8].

This paper attempts to be as self-contained and comprehensive as possible. While this is not a survey paper (as it contains new theorems and algorithms) a fair amount of classical results are recalled for the reader's convenience. In addition, we provide in Appendix A a complete process for analyzing the bistability of a biological model with the tools presented in this paper.

We dedicate the rest of this introduction to identify problems arising in the study of dynamical systems which are eligible to solutions based on semi-algebraic system solving. Some of these problems, namely Problems 1, 2, 3, are directly formulated in terms of dynamical systems. For a sake of clarity, the other problems, namely Problems 4 and 5, are stated in terms of conditions on the roots of a parametric univariate polynomial, which is meant to be the characteristic polynomial of the Jacobian matrix of the dynamical system under study.

We consider continuous dynamical systems defined by autonomous ordinary differential system of the following shape:

$$\begin{cases} \dot{y_1} = F_1(u_1, \ldots, u_d, y_1, \ldots, y_m), \\ \dot{y_2} = F_2(u_1, \ldots, u_d, y_1, \ldots, y_m), \\ \ \vdots \quad \vdots \\ \dot{y_m} = F_m(u_1, \ldots, u_d, y_1, \ldots, y_m). \end{cases} \tag{1}$$

where $F_1, \ldots, F_m$ are polynomials of $\mathbb{Q}[u_1, \ldots, u_d, y_1, \ldots, y_m]$. The variables $\mathbf{u} = (u_1, \ldots, u_d)$ are considered as parameters and the variables $\mathbf{y} = (y_1, \ldots, y_m)$ are seen as unknowns. In addition, we have $y_i = y_i(t)$ and $\dot{y_i} = \mathrm{d}y_i/\mathrm{d}t$ while the parameters $u_1, \ldots, u_d$ are independent of the derivation variable t. In the sequel, we simply write (1) as

$$\dot{\mathbf{y}} = F(\mathbf{u}, \mathbf{y}) \tag{2}$$

where $F(\mathbf{u}, \mathbf{y}) = (F_1(\mathbf{u}, \mathbf{y}), \ldots, F_m(\mathbf{u}, \mathbf{y}))$ is called the *vector field* of the system.

For any given parameter value $u \in \mathbb{R}^d$, one may notice that any $y \in \mathbb{R}^m$ such that $F_1(u, y) = \cdots = F_m(u, y) = 0$ holds, is a constant solution of System (1), which is called an *equilibrium* (or a *steady state*, or a *fixed point*). We are interested in the following problem regarding the equilibria of the given dynamical system.

Problem 1. *For a fixed parameter value u (or in absence of parameters) determine the number of equilibria of (1) and compute each of them (for instance, by means of isolation intervals). In presence of parameters, partition the parameter space into connected semi-algebraic sets, such that above each of them, the number of equilibria is constant and each equilibrium is a continuous function of the parameters.*

Problems 1 is a particular instance of the solving of semi-algebraic systems. Section 4 is dedicated to this more general question, with a view toward Problem 1.

We consider now a fixed parameter value u and a particular equilibrium y of System (1) at u. An important problem concerning the equilibrium y is to analyze its stability. We say y is *stable* if any solution of System (1) starting out close to y remains close to it. We say y is *asymptotically stable* if y is *stable* and if the solutions of System (1) starting out close to y become arbitrary close to it. If y is not stable, it is said to be *unstable*. The above discussion leads to enhance Problem 1 into the following ones, which deals with the number of asymptotically stable equilibria of System (1) depending or not on parameters.

Problem 2. *For a fixed parameter value u (or in absence of parameters) determine the number of asymptotically stable hyperbolic equilibria of (1) and compute each of them. In presence of parameters, partition the parameter space into connected semi-algebraic sets, such that above each of them, the number of asymptotically stable hyperbolic equilibria is constant and each of these equilibria is a continuous function of the parameters.*

The study of the system near the particular equilibrium y is usually done using the linear system

$$\dot{\mathbf{y}} = J(u, y)(\mathbf{y} - y), \tag{3}$$

where J is the *Jacobian matrix* of F:

$$J = \begin{pmatrix} \frac{\partial F_1}{\partial y_1} & \frac{\partial F_1}{\partial y_2} & \cdots & \frac{\partial F_1}{\partial y_m} \\ \frac{\partial F_2}{\partial y_1} & \frac{\partial F_2}{\partial y_2} & \cdots & \frac{\partial F_2}{\partial y_m} \\ \vdots & \vdots & & \vdots \\ \frac{\partial F_m}{\partial y_1} & \frac{\partial F_m}{\partial y_2} & \cdots & \frac{\partial F_m}{\partial y_m} \end{pmatrix}$$

We denote by

$$f(\lambda) = a_0\lambda^m + a_1\lambda^{m-1} + a_2\lambda^{m-2} + \cdots + a_{m-1}\lambda + a_m,$$

where $a_0 = 1$, the characteristic polynomial of J. If the matrix $J(u, y)$ has no eigenvalues with zero real parts, that is, if $f(u, y, \lambda)$ has no roots with zero real parts, then y is called a *hyperbolic equilibrium* at u; otherwise y is a *non-hyperbolic equilibrium* at u. In [32], Hartman and Grobman proved the following result: if y is a hyperbolic equilibrium, then near y, the phase portrait of the dynamical system (1) is topologically equivalent to that of the linearized dynamical system (3). The results imply that, for a hyperbolic equilibrium y, the phase flow of (1) is asymptotically stable near y if and only if the phase flow of (3) is asymptotically stable near y. Therefore, using standard results on linear differential systems [1], the phase flow of (1) is asymptotically stable near y if and only if all the complex roots of $f(u, y, \lambda)$ have negative real parts. This reduces Problem 2 to the following problem.

Problem 2'. *For a univariate polynomial $f(x) \in \mathbb{R}[x]$, determine whether all the complex roots of $f(x)$ have negative real parts or not.*

In the above analysis, we assume the equilibrium y is hyperbolic, so a natural question is how to determine whether y is hyperbolic or not. In other words, we want to solve the following problem:

Problem 3. *For a fixed parameter value u, determine whether each equilibrium of* (1) *is hyperbolic or not. In presence of parameters, partition the parameter space into connected semi-algebraic sets, such that above each of them, an equilibrium is always either hyperbolic or non-hyperbolic.*

This problem is equivalent to determine whether all the complex roots of the characteristic polynomial $f(u, y, \lambda)$ have nonzero real parts, which leads to the following general problem.

Problem 3'. *For a univariate polynomial $f(x) \in \mathbb{R}[x]$, determine whether $f(x)$ has complex roots with zero real parts or not.*

When y is a non-hyperbolic equilibrium of (1), if the characteristic polynomial $f(u, y, \lambda)$ has at least one complex root with positive real part, then y is an unstable equilibrium. Otherwise, the stability of y depends also on the higher order terms of the Taylor expansion of F near the point y. In this situation, one usually needs to apply the *Centre Manifold Theorem* [3] to reduce the original system to a low dimensional dynamical system defined on a centre manifold and further simplify it by computing its normal form. Finally, the normal form can be further reduced by removing terms that do not affect the stability of the equilibrium. Therefore, the first step towards stability analysis of non-hyperbolic equilibria of (1) is to determine when the characteristic polynomial has at least one complex root with positive real part or, equivalently, determine when $f(u, y, \lambda)$ has only complex roots with non-positive real parts, which leads to the following problem.

Problem 4. *For a univariate polynomial $f(x)$ with parametric coefficients, determine whether $f(x)$ has at least one complex root with positive real part. Equivalently, given two integers k_1 and k_2, determine whether $f(x)$ has zero as a root of multiplicity k_1 and k_2 pairs of purely imaginary roots while all the other complex roots have negative real parts.*

When non-hyperbolic equilibria are present, another more interesting phenomenon is the appearance of bifurcation. For the dynamical system (1), a *bifurcation* occurs at a parameter α_0 if there are parameter values α_1 arbitrarily close to α_0 with dynamics topologically non-equivalent to those at α_0. For example, the number or stability of equilibria or periodic orbits of (1) may change with perturbations of u from α_0. For a general dynamical system, such as (1), a systematic study is difficult. However, given an equilibrium y of (1) at u, necessary conditions for bifurcation can be obtained as follows. If a bifurcation of an equilibrium occurs near (u, y), then either or both conditions below are met:

– the characteristic polynomial f has zero as a root of multiplicity k, for some $k > 0$,
– the characteristic polynomial f has k pairs of purely imaginary roots, for some $k > 0$.

Therefore, the last problem we want to answer in this paper is as follows:

Problem 5. *Given non-negative integers k_1, k_2 and a polynomial $f(x)$ with parametric coefficients, determine whether $f(x)$ has zero as a root of multiplicity k_1 and k_2 pairs of purely imaginary roots while no other roots have zero real parts.*

A particular case of the above problem is $(k_1, k_2) = (0, 1)$. In this case, thus if the characteristic polynomial $f(u, y, \lambda)$ has a pair of purely imaginary roots and no other roots with zero real part, the limit cycle bifurcation that may occur at (u, y) is called a *Hopf bifurcation*. Such bifurcation has attracted the interest of many authors. In [20], the authors presented sufficient conditions for the appearance of Hopf bifurcations. In [23], the authors give sufficient and necessary conditions on Hopf bifurcations by further demanding that all the other eigenvalues have negative real roots, which is convenient for applying *Centre Manifold Theory* in order to reduce the dimension of dynamical systems. In [24], the authors present a framework for solving Problem 5.

2 On the Complex Roots of a Univariate Polynomial

As we have seen in the previous section, many problems related to dynamical systems reduce to studying the complex roots of a univariate polynomial with real coefficients. In particular, Problems 2', 3', 4 and 5 will be completely answered in the present section.

This section is firmly rooted in the papers [23, 24]. With respect to [23, 24] our main contribution in this section is Theorem 9, from which the main result of [23] (that is, Theorem 3.6 in [23] and Corollary 3 in this section), dedicated to Hopf bifurcation, can easily be derived. Theorem 9 provides two equivalent conditions for a polynomial with real coefficients to have only complex roots with non-positive real parts.

The proof of the first condition relies on several results of [23, 24], which are reviewed hereafter for the reader's convenience. To prove the second condition, we introduce Corollary 2 and Theorem 7. It should be pointed out that to deduce Corollary 3 from Theorem 9, this second condition is really needed. We also correct the error of sign difference in Theorem 3.1 of [23] (Theorem 1 in [24]) and revise it as Theorem 5 hereafter.

Let $f(x) \in \mathbb{R}[x]$ be a polynomial of degree m, and let us write

$$f(x) = a_0 x^m + a_1 x^{m-1} + \cdots + a_m.$$

After recalling the definition and standard properties (Lemma 1, Theorems 1, 3, 2, 4) of Hurwitz determinants, we discuss their relations with subresultant sequences in Section 2.2 and their use in the study of symmetric roots in Section 2.3.

Definition 1 (Hurwitz matrix). *We call* Hurwitz matrix *of f the $m \times m$ matrix $H = (H_{\mu\nu})$ defined by $H_{\mu\nu} = a_{2\nu-\mu}$ for $\nu = 1, \ldots, m$ and $\mu = 1, \ldots, m$,* *with the convention that $a_i = 0$ holds as soon as $i < 0$ or $i > m$ holds. For $i = 1, \ldots, m$, we denote by Δ_i the leading principal minors of H, which are called the* Hurwitz determinants *of H:*

$$\Delta_1 = a_1,\ \Delta_2 = \begin{vmatrix} a_1 & a_3 \\ a_0 & a_2 \end{vmatrix}, \ldots, \Delta_m = \begin{vmatrix} a_1 & a_3 & a_5 & \cdots & \cdots \\ a_0 & a_2 & a_4 & \cdots & \cdots \\ 0 & a_1 & a_3 & a_5 & \cdots \\ 0 & a_0 & a_2 & a_4 & \cdots \\ & & & & \ddots \end{vmatrix}.$$

It is easy to see that we have $\Delta_m = a_m \Delta_{m-1}$.

The following criterion provides a sufficient and necessary condition for a polynomial f to have only roots with negative real parts, which is therefore an answer to Problem 1.

Theorem 1 (Routh-Hurwitz's criterion [15]). *The real parts of all the zeros of $f(\lambda)$ are negative if and only if $\Delta_1 > 0$, $\Delta_2 > 0$, ..., $\Delta_{m-1} > 0$, $a_m > 0$.*

There is also another famous criterion equivalent to the above one, which is called Liénard-Chipart's Criterion.

Theorem 2 (Liénard-Chipart's criterion [15]). *The real parts of all the zeros of $f(\lambda)$ are negative if and only if we have:*

(1) *If m is odd, then all the below inequalities hold:*

$$a_m > 0,\ a_2 > 0,\ a_4 > 0,\ \ldots,\ a_{m-1} > 0,\ \Delta_2 > 0,\ \Delta_4 > 0,\ \ldots,\ \Delta_{m-1} > 0.$$

(2) *If m is even, then all the below inequalities hold:*

$$a_m > 0,\ a_1 > 0,\ a_3 > 0,\ \ldots,\ a_{m-1} > 0,\ \Delta_1 > 0,\ \Delta_3 > 0,\ \ldots,\ \Delta_{m-1} > 0.$$

2.1 Hurwitz Determinants and Stability of Hyperbolic Equilibria of Dynamical System

In this section, for a fixed parameter value $u \in \mathbb{R}^d$, let $y \in \mathbb{R}^m$ be an equilibrium of dynamical system (1).

Lemma 1 (Orlando's formula [14]). *Let λ_i, $i = 1, \ldots, m$, be the eigenvalues of $J(u, y)$ and Δ_{m-1} be the $(m-1)$-th Hurwitz determinant of its characteristic polynomial. Then we have:*

$$\Delta_{m-1} = (-1)^{\frac{1}{2}m(m-1)} \prod_{1 \le i < j \le m} (\lambda_i + \lambda_j).$$

Corollary 1 (Hyperbolic equilibrium criterion). *The following three properties hold.*

(1) *$J(u,y)$ have no zero eigenvalues if and only if $|J(u,y)| = (-1)^m a_m \neq 0$.*
(2) *If $\Delta_{m-1} \neq 0$, then $J(u,y)$ has no pure imaginary eigenvalues.*
(3) *If $\Delta_m = a_m \Delta_{m-1} \neq 0$, then y is a hyperbolic equilibrium.*

Proof. Property (1) is clear. Property (2) is an immediate consequence of Orlando's Formula. Property (3) follows from $|J(u,y)| = \lambda_1 \lambda_2 \cdots \lambda_m$.

Remark 1. *Necessary and sufficient conditions for $J(u,y)$ to have no pure imaginary eigenvalues (resp. y to be hyperbolic equilibrium) will be provided in Section 2.3.*

Theorem 3 (Lyapunov's first method on stability [28]). *The following properties hold.*

(i) If $J(u,y)$ has at least one eigenvalue with positive real parts, then y is unstable.
(ii) Assume that y is a hyperbolic equilibrium. If all the eigenvalues of $J(u,y)$ have negative real parts, then y is asymptotically stable.

Theorem 4 (Stability criterion for hyperbolic equilibria). *Let y be an equilibrium of System (1), we have:*

(1) *y is an asymptotically stable hyperbolic equilibrium if and only if*

$$\Delta_1 > 0,\, \Delta_2 > 0,\, \ldots,\, \Delta_{m-1} > 0,\, a_m > 0.$$

(2) *If y is hyperbolic, then y is unstable if and only if there exists some i, $1 \leq i \leq n$, such that $\Delta_i \leq 0$.*

Proof. Directly by Theorem 3 and Routh-Hurwitz Criterion.

2.2 Hurwitz Determinants and Subresultant Sequences

Let $\mathbb{A}$ be a commutative ring with identity and let $p \leq q$ be two positive integers. Let M be a $p \times q$ matrix with coefficients in $\mathbb{A}$. Let M_i be the square submatrix of M consisting of the first $p-1$ columns of M and the i_{th} column of M, for $i = p \cdots q$. Let $\det M_i$ be the determinant of M_i. We denote by $\mathrm{dpol}(M)$ the element of $\mathbb{A}[y]$, called the *determinant polynomial* of M, given by

$$\det M_p y^{q-p} + \det M_{p+1} y^{q-p-1} + \cdots + \det M_q.$$

Let $f_1(y), \ldots, f_p(y)$ be a set of polynomials of $\mathbb{A}[y]$. Let

$$q = 1 + \max(\deg f_1(y), \ldots, \deg f_p(y)).$$

The matrix M of $f_1, \ldots, f_p$ is defined by $M_{ij} = \mathrm{coeff}(f_i, y^{q-j})$.

Let $f = a_m y^m + \cdots + a_0$, $g = b_n y^n + \cdots + b_0$ be two polynomials of $\mathbb{A}[y]$ with positive degrees m and n. Let $\lambda = \min(m,n)$. Denote by $\mathrm{lc}(f)$ and $\mathrm{lc}(g)$ respectively the leading coefficient of f and g w.r.t. y. For any $0 \leq i < \lambda$, let

M be the matrix of the polynomials $y^{n-1-i}f, \ldots, yf, f, y^{m-1-i}g, \ldots, yg, g$. We define the i_{th} *subresultant* of f and g, denoted by $S_i(f, g, y)$ as

$$\begin{aligned} S_i(f, g, y) &= \text{dpol}(y^{n-1-i}f, \ldots, yf, f, y^{m-1-i}g, \ldots, yg, g) \\ &= \text{dpol}(M). \end{aligned}$$

Note that $S_i(f, g, y)$ is a polynomial in $\mathbb{A}[y]$ with degree at most i. Let $s_i(f, g, y) = \text{coeff}(S_i(f, g, y), y^i)$ and call it the *principle subresultant* coefficient of $S_i(f, g, y)$. If $m \geq n$, we define $S_\lambda(f, g, y) = g$, $S_{\lambda+1}(f, g, y) = f$, $s_\lambda = \text{lc}(g)$ and $s_{\lambda+1} = \text{lc}(f)$. If $m < n$, we define $S_\lambda = f$, $S_{\lambda+1} = g$, $s_\lambda = \text{lc}(f)$ and $s_{\lambda+1} = \text{lc}(g)$.

Let $\mathbb{A} = \mathbb{Q}[a_0, \ldots, a_m]$ and $f \in \mathbb{A}[x] = a_0x^m + a_1x^{m-1} + \cdots + a_{m-1}x + a_m$ be a polynomial of degree m. We write $f(x) = f_1(x^2) + xf_2(x^2)$. If $m = 2\ell+1$, we have $f_1(y) = a_1y^\ell + a_3y^{\ell-1} + \cdots + a_{2\ell+1}$ and $f_2(y) = a_0y^\ell + a_2y^{\ell-1} + \cdots + a_{2\ell}$. If $m = 2\ell$, we have $f_1(y) = a_0y^\ell + a_2y^{\ell-1} + \cdots + a_{2\ell}$ and $f_2(y) = a_1y^{\ell-1} + a_3y^{\ell-2} + \cdots + a_{2\ell-1}$.

Theorem 5. *Let $\Delta_1, \Delta_2, \ldots, \Delta_m$ be the Hurwitz determinants sequence of f. Then the following conclusion holds:*

(i) *If $m = 2\ell+1$, we have $\Delta_{m-1-2i} = \Delta_{2\ell-2i} = (-1)^{\frac{(\ell-i)(\ell-i-1)}{2}} s_i(f_1, \ell, f_2, \ell, y)$ hold, for $i = 0, 1, \ldots, \ell-1$.*

(ii) *If $m = 2\ell$, we have $\Delta_{m-1-2i} = \Delta_{2\ell-1-2i} = (-1)^{\frac{(\ell-i)(\ell-i-1)}{2}} s_i(f_1, \ell, f_2, \ell-1, y)$, for $i = 0, 1, \ldots, \ell-1$.*

(iii) *If $m = 2\ell+1$, for $i = 0, 1, \ldots, \ell$, we have*

$$\begin{aligned} \Delta_{m-2i} = \Delta_{2\ell+1-2i} &= (-1)^{\frac{(\ell-i)(\ell-i+1)}{2}} s_i(f_1, \ell, yf_2, \ell+1, y) \\ &= (-1)^{\frac{3(\ell-i)(\ell-i+1)}{2}} s_i(yf_2, \ell+1, f_1, \ell, y). \end{aligned}$$

(iv) *If $m = 2\ell$, we have $\Delta_{m-2i} = \Delta_{2\ell-2i} = (-1)^{\frac{(\ell-i)(\ell-i+1)}{2}} s_i(f_1, \ell, yf_2, \ell, y)$ hold, for $i = 0, 1, \ldots, \ell-1$.*

Proof. Here, we only prove (i) holds and leave the other cases for exercise.

When $m = 2\ell+1$, we have $f_1(y) = a_1y^\ell + a_3y^{\ell-1} \cdots + a_m$, $f_2(y) = a_0y^\ell + a_2y^{\ell-1} \cdots + a_{m-1}$. So the Sylvester matrix M formed by the coefficients of f_1 and f_2 is an $2\ell \times 2\ell$ matrix of the form:

$$M = \begin{bmatrix} a_1 & a_3 & a_5 & \cdots & a_m & & & & \\ & a_1 & a_3 & a_5 & \cdots & a_m & & & \\ & & \ddots & \ddots & & & \ddots & & \\ & & & a_1 & a_3 & a_5 & \cdots & a_m \\ a_0 & a_2 & a_4 & \cdots & a_{m-1} & & & & \\ & a_0 & a_2 & a_4 & \cdots & a_{m-1} & & & \\ & & \ddots & \ddots & & & \ddots & & \\ & & & a_0 & a_2 & a_4 & \cdots & a_{m-1} \end{bmatrix} \tag{4}$$

On the other hand, the Hurwitz matrix H of f is an $(2\ell+1)\times(2\ell+1)$ matrix whose elements are arranged like this:

$$H = \begin{bmatrix} a_1 & a_3 & a_5 & \cdots & a_m & & & & \\ a_0 & a_2 & a_4 & \cdots & a_{m-1} & & & & \\ & a_1 & a_3 & a_5 & \cdots & a_m & & & \\ & a_0 & a_2 & a_4 & \cdots & a_{m-1} & & & \\ & & \cdots & \cdots & & & & & \\ & & a_1 & a_3 & a_5 & \cdots & a_m & & \\ & & a_0 & a_2 & a_4 & \cdots & a_{m-1} & & \\ & & & a_1 & a_3 & \cdots & a_{m-2} & a_m & \end{bmatrix} \qquad (5)$$

Let H^* be the sub-matrix composed by the first 2ℓ rows and 2ℓ columns of H. We denote by H_{2i} the sub-matrix of H^*, formed by the first $2i$ rows and $2i$ columns, for $i = 1, 2, \ldots, \ell$. We denote by M_i the sub-matrix of M, formed by deleting the last i rows composed by the coefficients of $f_1(y)$ and the last i rows composed by the coefficients of $f_2(y)$ and then deleting the last $2i$ columns for $i = 0, 1, \ldots, \ell - 1$. Then it's easy to see that if we make the odd rows of $H_{2\ell-2i}$ "float up" one by one, we finally get the matrix M_i. So the number of row exchanges for $H_{2\ell-2i}$ is: $0+1+2+\cdots+(\ell-i-1) = \frac{(\ell-i)(\ell-i-1)}{2}$. Therefore, we have $\Delta_{2\ell-2i} = |H_{2\ell-2i}| = (-1)^{\frac{(\ell-i)(\ell-i-1)}{2}}|M_i| = (-1)^{\frac{(\ell-i)(\ell-i-1)}{2}} s_i(f_1, \ell, f_2, \ell, y)$, for $i = 0, 1, \ldots, \ell - 1$.

Remark 2. *This theorem is a corrected version of Theorem 1 in [24], where the sign differences between Δ_i and s_i are wrong.*

2.3 Hurwitz Determinants and Symmetric Roots

The following result is taken from [23]. Corollary 2 is a direct consequence.

Lemma 2 ([23]). *Given a univariate polynomial $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m$ of $\mathbb{R}[x]$, where $a_0 \neq 0$. We write $f(x)$ into the form: $f(x) = f_1(x^2) + xf_2(x^2)$. Then $f(x)$ has a pair of symmetric zeros z and $-z$ in $\mathbb{C}$ if and only if z^2 is a common zero of $f_1(y)$ and $f_2(y)$.*

Corollary 2. *Assume that $a_m \neq 0$, then $f(x)$ has a pair of symmetric zeros z and $-z$ in $\mathbb{C}$ if and only if z^2 is a common zero of $f_1(y)$ and $yf_2(y)$.*

Theorem 6 ([23]). *Let $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m \in \mathbb{R}[x]$ be a polynomial of degree m. Then $f(x)$ has exactly k pairs of symmetric roots z_i and $-z_i$ in $\mathbb{C}$ if and only if $\Delta_{m-1} = 0, \ldots, \Delta_{m-2k+1} = 0, \Delta_{m-2k-1} \neq 0$.*

Theorem 7. *Notation as above, if $a_m \neq 0$, then f has exactly k pairs of symmetric roots z_i and $-z_i$ if and only if $\Delta_m = 0, \ldots, \Delta_{m-2k+2} = 0, \Delta_{m-2k} \neq 0$.*

Proof. If $a_m \neq 0$, by Corollary 2, the number of symmetric roots, counted with multiplicities, of the polynomial f is equal to the number of common roots, counted with multiplicities, of the two polynomials $f_1(y)$ and $yf_2(y)$. According to the elementary properties of subresultant sequences the polynomials $f_1(y)$ and $f_2(y)$ have k common roots if and only if

$$s_0(f_1, yf_2, y) = 0, \ldots, s_{k-1}(f_1, yf_2, y) = 0, s_k(f_1, yf_2, y) \neq 0.$$

So by Theorem 5 and specialization property of subresultants [29, 7], f has exactly k pairs of symmetric roots if and only if $\Delta_m = 0, \ldots, \Delta_{m-2k+2} = 0, \Delta_{m-2k} \neq 0$.

Lemma 3 ([23]). *Let $f(x) \in \mathbb{R}[x]$ be a polynomial of degree m and $z_1, \ldots, z_k$ be arbitrary complex numbers. Let $f^*(x) = f(x)(x^2 - z_1^2) \cdots (x^2 - z_k^2)$. If Δ_i^* is the Hurwitz determinants of order i of the polynomial $f^*(x)$, then $\Delta_i = \Delta_i^*$, for $i = 1, \ldots, m$. Similarly, let $f^*(x) = f(x)x^k$, then we also have $\Delta_i = \Delta_i^*$ hold.*

Theorem 8. *The polynomial $f(x)$ has zero as root of multiplicity k and all the other roots in the left half-plane if and only if $a_{m-k+1} = \cdots = a_m = 0$ and $\Delta_1 > 0, \Delta_2 > 0, \ldots, \Delta_{m-k} > 0$.*

Proof. It follows directly from Routh-Hurwitz criterion and Lemma 3.

Theorem 9. *Let $f(x) \in \mathbb{R}[x]$ be a polynomial of degree m and $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m = f_1(x^2) + xf_2(x^2)$. Let $\Delta_1, \Delta_2, \ldots, \Delta_m$ be the Hurwitz determinants sequence of f. Then the following statements are equivalent:*

- *(i)* *$f(x)$ has k pairs of pure imaginary roots and all the other roots are in the left half-plane.*
- *(ii)* *$S_k(f_1, f_2, y)$ has k negative real roots and $\Delta_{m-1} = \Delta_{m-3} = \cdots = \Delta_{m-2k+1} = 0$, $\Delta_{m-2k} > 0, \Delta_{m-2k-1} > 0, \ldots, \Delta_1 > 0$.*
- *(iii)* *$S_k(f_1, yf_2, y)$ has k negative real roots and $a_m \neq 0, \Delta_m = \Delta_{m-2} = \cdots = \Delta_{m-2k+2} = 0$, $\Delta_{m-2k} > 0, \Delta_{m-2k-1} > 0, \ldots, \Delta_1 > 0$.*

Proof. "$(i) \Rightarrow (ii)$". Assume that $f(x)$ has k pairs of pure imaginary roots and all the other roots are in the left half-plane. Let $\pm i\omega_1, \ldots, \pm i\omega_k$ be the k pairs of pure imaginary roots, then we can write $f(x)$ as $f(x) = f^*(x)(x^2 + \omega_1^2) \cdots (x^2 + \omega_k^2)$, where $\omega_1^2 > 0, \ldots, \omega_k^2 > 0$ and $f^*(x)$ has only roots in the left half-plane. By Routh-Hurwitz criterion, we know that $\Delta_1^* > 0, \Delta_2^* > 0, \ldots, \Delta_{m-2k}^* > 0$. According to the Lemma 3, we know that $\Delta_i^* = \Delta_i$. Therefore, we have $\Delta_{m-2k} > 0, \Delta_{m-2k-1} > 0, \ldots, \Delta_1 > 0$ hold.

Moreover, by assumption we know the k pairs of pure imaginary roots are the only symmetric roots of $f(x)$, which implies $\Delta_{m-1} = \Delta_{m-3} = \cdots = \Delta_{m-2k+1} = 0, \Delta_{m-2k-1} \neq 0$. Therefore, by Theorem 5 we have $s_0(f_1, f_2, y) = 0, \ldots, s_{k-1}(f_1, f_2, y) = 0, s_k(f_1, f_2, y) \neq 0$, which implies that $S_k(f_1, f_2, y) = \gcd(f_1, f_2, y)$. On the other hand, since $\pm i\omega_1, \ldots, \pm i\omega_k$ are the symmetric roots of $f(x)$, by Lemma 2, $-\omega_1^2, \ldots, -\omega_k^2$ are the common roots of $f_1(y)$ and $f_2(y)$, that is, they are the real roots of $S_k(f_1, f_2, y)$. Therefore $S_k(f_1, f_2, y)$ has k negative real roots.

"$(ii) \Rightarrow (i)$" By the assumption, we have $\Delta_{m-1} = \Delta_{m-3} = \cdots = \Delta_{m-2k+1} = 0, \Delta_{m-2k-1} \neq 0$, which implies that

$$s_0(f_1, f_2, y) = s_1(f_1, f_2, y) = \cdots = s_{k-1}(f_1, f_2, y) = 0, s_k(f_1, f_2, y) \neq 0.$$

Therefore the degree of $S_k(f_1, f_2, y)$ is k and $S_k(f_1, f_2, y) = \gcd(f_1, f_2, y)$. Since $S_k(f_1, f_2, y)$ has k negative real roots, we know that $f_1(y)$ and $f_2(y)$ has k common negative real roots and no other common roots. So by Lemma 2, $f(x)$ has exactly k pairs of pure imaginary roots and no other symmetric roots. Let us write $f(x) = f^*(x)(x^2 + \omega_1^2) \cdots (x^2 + \omega_k^2)$, according to $\Delta_{m-2k} > 0, \Delta_{m-2k-1} > 0, \ldots, \Delta_1 > 0$ and Lemma 3, we know that all the roots of $f^*(x)$ are in the left half-plane. Therefore $f(x)$ has k pairs of pure imaginary eigenvalues and all the other roots are in the left half-plane.

The proof of equivalence of (i) and (iii) are similar. The only difference is that during the proof we need to use Theorem 7 instead of Theorem 6 and Corollary 2 instead of Lemma 2.

By the above theorem, we get the following corollary, which is the main theorem on Hopf bifurcation in [23,24].

Corollary 3 (Theorem 4 [24]). *Let $f(x) \in \mathbb{R}[x]$ be a degree m polynomial and write $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m = f_1(x^2) + xf_2(x^2)$ with $a_0 > 0$. Let $\Delta_1, \Delta_2, \ldots, \Delta_m$ be the Hurwitz determinants sequence of f. Then $f(x)$ has a pair of distinct roots, $i\omega$ and $-i\omega$, on the imaginary and all the other roots in the left half-plane if and only if $a_m > 0, \Delta_{m-1} = 0, \Delta_{m-2} > 0, \ldots, \Delta_1 > 0$.*

Proof. By the equivalence of (i) and (iii) in Theorem 9, we only need to prove that $a_m > 0, \Delta_{m-1} = 0, \Delta_{m-2} > 0, \ldots, \Delta_1 > 0$ if and only if $S_1(f_1, yf_2, y)$ has one negative real root and $a_m \neq 0, \Delta_m = 0, \Delta_{m-2} > 0, \ldots, \Delta_1 > 0$. By Theorem 5, we have $S_1(f_1, yf_2, y) = (-1)^{\frac{\ell(\ell-1)}{2}}(\Delta_{m-2}y + a_m\Delta_{m-3})$.

" $\Rightarrow$ " Since $a_m > 0, \Delta_{m-1} = 0$, we have $a_m \neq 0$ and $\Delta_m = a_m\Delta_{m-1} = 0$. Moreover, as $a_m > 0$ and $\Delta_{m-2} > 0, \Delta_{m-3} > 0$, we know that $S_1(f_1, yf_2, y)$ has one negative real root.

" $\Leftarrow$ " Since $S_1(f_1, yf_2, y)$ has one negative real root and $\Delta_{m-2} > 0, \Delta_{m-3} > 0$, we have $-\Delta_{m-2}a_m\Delta_{m-3} < 0$, which implies that $a_m > 0$. Moreover, by $\Delta_m = 0$, we have $\Delta_{m-1} = 0$.

Combining the result of Theorem 8 and Theorem 9, we get the answer to Problem 4. The answer to Problem 5 was first briefly mentioned in [24], which we summarize as the following Theorem.

Theorem 10. *Let $f(x) = a_0x^m + a_1x^{m-1} + \cdots + a_m$ be a univariate polynomial of $\mathbb{R}[x]$. Then $f(x)$ has a root 0 of multiplicity k_1 and has k_2 pairs of pure imaginary roots while no other roots have zero real parts if and only if the following holds:*

– *The coefficients of $f(x)$ satisfy $a_m = \cdots = a_{m-k_1+1} = 0, a_{m-k_1} \neq 0$.*

– *Denote* $a_0x^{m-k_1} + a_1x^{m-k_1-1} + \cdots + a_{m-k_1} = f_1(x^2) + xf_2(x^2)$. *Then there exists an integer* $k \geq k_2$ *such that* $S_k(f_1, f_2, y)$ *has* k_2 *negative real roots and*

$$\Delta_{m-k_1-1} = \Delta_{m-k_1-3} = \cdots = \Delta_{m-k_1-2k+1} = 0, \Delta_{m-k_1-2k-1} \neq 0.$$

Proof. It directly follows from Lemma 2, Lemma 3 and Theorem 6.

Remark 3. *In the above theorem, if both* $k_1 = 0$ *and* $k_2 = 0$*, then we get an answer to Problem* $3'$*. If* $k_1 = 0$ *and* $k_2 = 1$*, then we get the necessary and sufficient condition on Hopf bifurcation.*

The reader may notice that in [23, 24] there is also a theorem to provide sufficient and necessary conditions on Hopf bifurcation. More precisely, it is Theorem 3.5 in [23] and Theorem 3 in [24]. However, we find that (also noticed by the author) the condition provided there is only a sufficient condition.

In Theorem 10, we need to determine when a univariate polynomial S_k of degree k with parametric coefficients has k_2, $0 < k_2 \leq k$, negative real zeros. This problem can be reduced to an exhaustive case discussion on the signs of polynomials whose variables are the coefficients of S_k, by Sturm-Habicht sequence [18] or negative root discriminant sequence [37].

In Theorem 9, rather we want to determine when all the complex roots of a univariate polynomial with parametric real coefficients are real and negative. In the rest of this section, we provide a relatively simple answer by virtue of Descartes criterion and discriminant sequence [37,38].

Lemma 4 (Descartes criterion). *Let* $f(x) \in \mathbb{R}[x]$ *be a polynomial of degree* n*. Let* ν *be the number of sign variations of its coefficients sequence. Then there exists* $m \geq 0$ *such that the number of positive real roots of* $f(x)$ *equals* $\nu - 2m$*.*

Corollary 4. *Let* $f(x) = a_0x^n + \cdots + a_{n-1}x + a_n$ *be a polynomial of degree* n*. If* $f(x)$ *has* n *negative real roots, then we have* $a_ia_{i+1} > 0$ *for all* $0 \leq i \leq n-1$*.*

Proof. Since $f(x)$ has n negative real roots, $f(-x)$ has n positive real roots. By Descartes criterion, we have $a_i \neq 0$. On the other hand, since $f(x)$ has no positive real roots, we know that a_i have the same sign. Done.

Definition 2 (Discrimination matrix). *Given a polynomial with general symbolic coefficients,* $f(x) = a_0x^n + a_1x^{n-1} + \cdots + a_n$*, the following* $2n \times 2n$ *matrix in terms of the coefficients,*

$$\begin{bmatrix}
a_0 & a_1 & a_2 & \cdots & a_n & & & & \\
0 & na_0 & (n-1)a_1 & \cdots & a_{n-1} & & & & \\
0 & a_0 & a_1 & \cdots & a_{n-1} & a_n & & & \\
0 & 0 & na_0 & \cdots & 2a_{n-2} & a_{n-1} & & & \\
 & & & \cdots & \cdots & & & & \\
 & & & \cdots & \cdots & & & & \\
 & & & a_0 & a_1 & a_2 & \cdots & a_n \\
 & & & 0 & na_0 & (n-1)a_1 & \cdots & a_{n-1}
\end{bmatrix}$$

is called the discrimination matrix of $f(x)$, and denoted by $Discr(f)$. By d_k or $d_k(f)$ denote the determinant of the submatrix of $Discr(f)$, formed by the first k rows and the first k columns for $k = 1, 2, \ldots, 2n$.

Definition 3 (Discriminant sequence). *Let $D_k = d_{2k}, k = 1, \ldots, n$. We call the sequence $[D_1, D_2, \ldots, D_n]$ the discriminant sequence of $f(x)$, and denote it by $DiscrList(f)$. The last term D_n is just the discriminant of f.*

Definition 4 (Sign list). *We call the list $[sign(A_1), sign(A_2), \ldots, sign(A_n)]$ the sign list of a given sequence $A_1, A_2, \ldots, A_n$, where*

$$sign(A_i) = \begin{cases} 1, & A_i > 0 \\ 0, & A_i = 0 \\ -1, & A_i < 0 \end{cases}$$

Definition 5 (Revised sign list). *Given a sign list $[s_1, s_2, \ldots, s_n]$, we construct a new list $[t_1, t_2, \ldots, t_n]$ as follows: (which is called the revised sign list)*

- *If $[s_i, s_{i+1}, \ldots, s_{i+j}]$ is a section of the given list, where $s_i \neq 0, s_{i+1} = \cdots = s_{i+j-1} = 0, s_{i+j} \neq 0$, then, we replace the subsection $[s_{i+1}, \ldots, s_{i+j-1}]$ by the first $j-1$ terms of $[-s_i, -s_i, s_i, s_i, -s_i, -s_i, s_i, s_i, \ldots]$.*
- *Otherwise, let $t_k = s_k$, i.e. no changes for other terms.*

Theorem 11. *Given a polynomial $f(x) = a_0x^n + a_1x^{n-1} + \cdots + a_n$, where $a_0 \neq 0$ of $\mathbb{R}[x]$. If the number of sign changes of the revised sign list of $D_1, D_2, \ldots, D_n$ is ν, the number of non-vanishing members of the revised sign list is l, then we have: the number of distinct real roots of $f(x)$ equals $l - 2\nu$; the number of distinct pairs of conjugate imaginary roots of $f(x)$ is ν.*

Example 1. *Let $f = (x-1)(x^2+1)$, whose discriminant sequence is $[3, -4, -16]$. The sign list of it is: $[1, -1, 1]$. Its revised is the same to the sign list. So the number of distinct real roots of f is $3 - 2 = 1$.*

Theorem 12. *Let $f(x) \in \mathbb{R}[x]$ be a polynomial of degree n and $[D_1, D_2, \ldots, D_n]$ be its discriminant sequence. Then $f(x)$ has n negative real roots if and only if all its coefficients have the same nonzero sign and there exists k, $1 \leq k \leq n$, such that $\forall i \leq k$, $D_i > 0$ and for other i, we have $D_i = 0$.*

Proof. " $\Rightarrow$ " By Corollary 4, we know that all the coefficients of $f(x)$ have the same nonzero sign. On the other hand, since $f(x)$ has no imaginary real roots, the revised sign list of $[D_1, D_2, \ldots, D_n]$ has no sign changes according to Theorem 11. By the rule on constructing the revised sign list, we conclude that there exists k, $1 \leq k \leq n$, such that $\forall i \leq k$, $D_i > 0$ and for all $i > k$, $D_i = 0$.

" $\Leftarrow$ " If there exists k, $1 \leq k \leq n$, such that $\forall i \leq k$, $D_i > 0$ and for other i, we have $D_i = 0$. Then the revised sign list will look like this: $[1, \ldots, 1, 0, \ldots, 0]$ Therefore, the number of sign changes is 0. So $f(x)$ have no imaginary roots. Moreover, since the coefficients sequence of $f(x)$ has 0 sign variations, we know immediately that $f(x)$ has n negative real roots by Descartes Criterion.

3 Stability of Hyperbolic Equilibria in View of Bifurcation

In Section 2, we discussed the stability of a hyperbolic equilibria for a fixed parameter value. In this section, we study the stability of a hyperbolic equilibria under variation of parameters.

Definition 6 ([25]). *Let us consider a dynamical system that depends on parameters. The appearance of a topologically nonequivalent phase portrait under variation of parameters is called a bifurcation.*

Lemma 5 ([25]). *Given two hyperbolic equilibria of dynamical system (1), the phase portraits of system (1) near them are locally topologically equivalent if and only if at the two equilibria the Jacobian matrix J has the same number of eigenvalues with negative (positive) real parts.*

Theorem 13 (Boundary crossing theorem). *Given a parameter value α_0 of the dynamical system (1) and let β_0 be a hyperbolic equilibrium of system (1) at the parameter α_0. Then there exists a continuous function $y(u)$ defined in a small neighbourhood $O(\alpha_0)$ of α_0 satisfying $F(u, y(u)) = 0, y(\alpha_0) = \beta_0$. Moreover, the defining domain $O(\alpha_0)$ of $y(u)$ can be extended as long as $\Delta_m(u, y(u)) \neq 0$. In addition, inside the extended domain, there will be no bifurcation. In particular, the stability of $y(u)$ remains the same in the extended domain.*

Proof. Since β_0 is a hyperbolic equilibrium of system (1), we have $\Delta_m(\alpha_0, \beta_0) = (-1)^m \Delta_{m-1}(\alpha_0, \beta_0)\mathrm{Det}(J)(\alpha_0, \beta_0) \neq 0$. Since $\mathrm{Det}(J)(\alpha_0, \beta_0) \neq 0$, by the implicit function Theorem, we know that in a neighbourhood of α_0, there is one and only one continuous function $y(u)$ defined by $F(u, y(u)) = 0$ such that $y(\alpha_0) = \beta_0$. Moreover, we can extend the domain of the function $y(u)$ if only $\mathrm{Det}(J)(u, y(u)) \neq 0$. On the other hand, the real parts of the eigenvalues of $J(u, y(u))$ will not become zero, which implies that the number of the eigenvalues of $J(u, y(u))$ with negative real parts and positive real parts will remain the same, respectively. By Lemma 5, the phase portraits will remain locally topologically equivalent. Therefore, the stability will not change if only $\Delta_n(u, y(u)) \neq 0$.

Remark 4. *In 1929, Frazer and Duncan published a paper entitled "On the Criteria for the Stability of Small Motions" [12]. In that paper, the authors presented a theorem with the same name as above one, where they pointed out that when the system passes from a region of stability to the border of stability, Δ_n changes from positive to zero. Here by the language of bifurcation, we see that a dynamical system will keep structurally stable if only the parameter does not cross the boundary described by $\Delta_n = 0$.*

4 Comprehensive Triangular Decomposition of Parametric Semi-algebraic Systems

In this section, we introduce the notion of a *comprehensive triangular decomposition of a parametric semi-algebraic system*. Its purpose serves our needs in the study of parametric polynomial dynamical systems: solving the parametric semi-algebraic systems that arise from the results of Sections 2 and 3.

We start with some necessary notations. For the related concepts, the reader may refer to [6, 8, 2]. Let $\mathbf{k}$ is a field of characteristic zero and let $\mathbf{K}$ be its algebraic closure. Let d, m, n be positive integers such that we have $n = d + m$ and $d, m \geq 1$. Let $\mathbf{x} = x_1 < \cdots < x_n$ be ordered variables, which are divided into two groups $x_1 < \cdots < x_d$ and $x_{d+1} < \cdots < x_n$. We rename x_i as u_i for $1 \leq i \leq d$ and see $\mathbf{u} = u_1, \ldots, u_d$ as parameters. We rename x_i as y_{i-d} for $d + 1 \leq i \leq n$ and see $\mathbf{y} = y_1, \ldots, y_m$ as unknowns.

In this paper, we use "Z" to denote the zero set of a polynomial system, involving equations and inequations, in $\mathbf{K}^n$ and "$Z_{\mathbb{R}}$" to denote the zero set of a semi-algebraic system in $\mathbb{R}^n$. For a polynomial system $\mathcal{S}$ and point u, we denote by $\mathcal{S}(u)$ the specialized (or evaluated) system at u.

Let p be a non-constant polynomial of $\mathbf{k}[\mathbf{x}]$. Denote by $\operatorname{sep}(p)$ the separant (that is the derivative of p w.r.t. its main variable) of p. Let T be a regular chain of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Denote respectively by $\operatorname{mvar}(T)$, h_T, $\operatorname{sep}(T)$ and $W(T)$ the set of main variables of T, the product of initials of polynomials in T, the product of all $\operatorname{sep}(p)$ for $p \in T$ and the quasi-component of T. Let $p \in \mathbf{k}[\mathbf{u}, \mathbf{y}]$. Denote by $\operatorname{res}(p, T)$ the iterated resultant of p w.r.t. T. Denote by $\varnothing$ the empty regular chain.

In section 4.1, we introduce the concept of a *disjoint squarefree comprehensive triangular decomposition* (DSCTD) of a parametric constructible system cs of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$, which extends the notion of a CTD of an algebraic variety V of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$, introduced in [6]. We also present an algorithm for computing this new type of decomposition.

In section 4.2, we introduce the concept of a comprehensive triangular decomposition of a parametric semi-algebraic system (RCTD) $\mathcal{S}$ of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Moreover, we show that RCTD can be easily computed by combining DSCTD with our previous work on computing CAD via triangular decompositions [8].

4.1 Disjoint Squarefree Comprehensive Triangular Decomposition

Definition 7. *Let $R := [T, h]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Let $u \in \mathbf{K}^d$. We say that R* specializes well *at u if $R(u)$ is a squarefree regular system of $\mathbf{K}[\mathbf{y}]$ and $h_T(u) \neq 0$. Let $\mathcal{R} = \{R_1, \ldots, R_e\}$ be a finite set of regular systems of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. We say that $\mathcal{R}$* specializes disjointly well *at u, if: (i) each $R \in \mathcal{R}$ specializes well at u and (ii) the zero sets of $R_i(u)$ in $\mathbf{K}^n$ are pairwise disjoint.*

Denote by $\pi_{\mathbf{u}}$ the canonical projection onto the parameter space. Let $\uplus$ denote the disjoint union of two sets. Let cs be a constructible set of $\mathbf{K}^n$. Following the results of [6], we assume that cs is given as the union of the zero sets of finitely many regular systems in $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. In this section, we always assume that cs is represented by such a set of regular systems.

Definition 8. *Let cs be a constructible set of $\mathbf{K}^n$. A DSCTD of cs is a pair $(\mathcal{C}, (\mathcal{R}_C, C \in \mathcal{C}))$, where $\mathcal{C}$ is a finite partition of $\pi_{\mathbf{u}}(cs)$ into nonempty* *constructible sets, and, for each $C \in \mathcal{C}$, $\mathcal{R}_C$ is a finite set of regular systems of* $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ *such that for each point $u \in C$ the following conditions hold:*

(i) $\mathcal{R}_C$ specializes disjointly well at u;
(ii) we have $cs(u) = \uplus_{R \in \mathcal{R}_C} Z(R(u))$

Let $R := [T, h]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Let $T_\mathbf{u}$ (resp. $T_\mathbf{y}$) denote the set of polynomials in T whose main variables belong to $\mathbf{u}$ (resp. $\mathbf{y}$). Define $r_\mathbf{y} = \text{res}(h \cdot \text{sep}(T_\mathbf{y}), T_\mathbf{y})$. Let $W^\mathbf{u}(T_\mathbf{u})$ be the quasi-component of $T_\mathbf{u}$ in $\mathbf{K}^d$.

Let $p \in \mathbf{k}[\mathbf{u}, \mathbf{y}]$. Denote by $\text{coeffs}(p, \mathbf{y})$ the set of coefficients of p w.r.t. the variables $\mathbf{y}$. Let $V^\mathbf{u}(\text{coeffs}(p, \mathbf{y}))$ be the algebraic variety of $\text{coeffs}(p, \mathbf{y})$ in $\mathbf{K}^d$.

Definition 9. *We call* defining set *of the squarefree regular system $R := [T, h]$ the set denoted by $D^\mathbf{u}(R)$ and defined by $D^\mathbf{u}(R) := W^\mathbf{u}(T_\mathbf{u}) \setminus V^\mathbf{u}(\text{coeffs}(r_\mathbf{y}, \mathbf{y}))$.*

Lemma 6. *Let $R := [T, h]$ be a squarefree regular system of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$. Let $u \in \mathbf{K}^d$. Then R specializes well at u if and only if $u \in D^\mathbf{u}(R)$.*

Proof. Its proof is based on the specialization properties of subresultants and it is similar to the proof of Proposition 4 of [6].

Algorithm 1 computes a DSCTD of a constructible set. The proof of its termination and correctness is similar to that of the algorithm CTD in [6]. We also refer to [6] for the specifications of the subroutines MPD, SMPD and Intersect called in Algorithm 1.

The implementation of the DSCTD algorithm is available in the `RegularChains` library since `Maple13`. It sits inside the `ParametricSystemTool` module and is

Algorithm 1. DSCTD(cs)

```
Input: A constructible set $cs$ of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$.
Output: A DSCTD of $cs$.
1  let $\mathcal{R}$ be the set of regular systems representing $cs$
2  $\mathcal{R} := \mathsf{MPD}(\mathcal{R})$; $\mathcal{R}' := \{\ \}$
3  while $\mathcal{R} \neq \{\ \}$ do
4      let $R := [T, h] \in \mathcal{R}$; $\mathcal{R} := \mathcal{R} \setminus \{R\}$
5      $\mathcal{R}' := \mathcal{R}' \cup \{R\}$
6      $G := \text{coeffs}(\text{res}(\text{sep}(T_\mathbf{y})h, T_\mathbf{y}), \mathbf{y})$
7      $\mathcal{R} := \mathcal{R} \cup \mathsf{MPD}(\mathsf{Intersect}(G, R))$
8  $\mathcal{R} := \mathcal{R}'$; $\mathcal{C} := \{\ \}$
9  for $R \in \mathcal{R}$ do
10     $\mathcal{C} := \mathcal{C} \cup \{D^\mathbf{u}(R)\}$
11 $\mathcal{C} := \mathsf{SMPD}(\mathcal{C})$
12 for $C \in \mathcal{C}$ do
13     let $\mathcal{R}_C$ be the set of regular systems $R \in \mathcal{R}$ with $C \subseteq D^\mathbf{u}(R)$
14 return $(\mathcal{C}, (\mathcal{R}_C, C \in \mathcal{C}))$
```

implemented as the command `ComprehensiveTriangularize` with option the `‘disjoint’=‘yes’`.

Let cs be a constructible set of $\mathbf{K}^n$. Often, we only need to partition the parameter space into constructible sets such that above each of them:

1. either cs has no solutions;
2. or cs has infinitely many solutions;
3. or cs has a constant number of solutions and such that the solutions are continuous functions of the parameters.

A precise definition of this idea is stated in Definition 10.

Definition 10. *Let cs be a constructible set of $\mathbf{K}^n$. A weak DSCTD (WDSCTD) of cs is a pair $(\mathcal{C}, (\mathcal{T}_C, C \in \mathcal{C}))$, where*

- *$\mathcal{C}$ is a finite partition of $\mathbf{K}^d$ into nonempty constructible sets,*
- *for each $C \in \mathcal{C}$, $\mathcal{T}_C$ is a finite set of regular chains of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$ such that:*
  - *(i) either $\mathcal{T}_C$ is empty, which means that $cs(u)$ is empty for each $u \in C$*
  - *(ii) or $\mathcal{T}_C = \{\varnothing\}$, which means that $cs(u)$ is infinite for each $u \in C$;*
  - *(iii) or each $T \in \mathcal{T}_C$ satisfies $\mathrm{mvar}(T) = \mathbf{y}$ and for each $u \in C$, $\mathcal{T}_C$ specializes disjointly well at u and $cs(u) = \dot\cup_{T \in \mathcal{T}_C} Z(T(u))$.*

Algorithm 2 computes a WDSCTD of cs. It is not difficult to prove the termination and correctness of this algorithm.

Algorithm 2. WDSCTD(cs)

```
Input: A constructible set $cs$ of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$.
Output: A WDSCTD of $cs$.
1  let $\mathcal{R}$ be the set of regular systems representing $cs$
2  let $\mathcal{R}_0$ (resp. $\mathcal{R}_1$) be the set of regular systems $[T, h]$ in $\mathcal{R}$ such that
   $\mathbf{y} \subseteq \mathrm{mvar}(T)$ (resp. $\mathbf{y} \not\subseteq \mathrm{mvar}(T)$)
3  let $(\mathcal{C}, (\mathcal{R}_C, C \in \mathcal{C}))$ be a DSCTD of $\mathcal{R}_0$
4  let $\mathcal{E}_1$ be the projection of the constructible set $\mathcal{R}_1$ on $\mathbf{K}^d$
5  $\mathcal{D} := \{ \}$
6  if $\mathcal{E}_1$ is not empty then
7      $D := \mathcal{E}_1$; $\mathcal{T}_D := \{\varnothing\}$; $\mathcal{D} := \mathcal{D} \cup \{D\}$
8  for $C \in \mathcal{C}$ do
9      $C :=$ Difference$(C, \mathcal{E}_1)$
10     if $C$ is not empty then
11         $D := C$; $\mathcal{T}_D := \{T_{\mathbf{y}} \mid [T, h] \in \mathcal{R}_C\}$; $\mathcal{D} := \mathcal{D} \cup \{D\}$
12 $D :=$ Difference$(\mathbf{K}^n, \cup_{D \in \mathcal{D}} D)$
13 if $D$ is not empty then
14     $\mathcal{D} := \mathcal{D} \cup \{D\}$; $\mathcal{T}_D := \{ \}$
15 return $(\mathcal{D}, (\mathcal{T}_D, D \in \mathcal{D}))$
```

4.2 Comprehensive Triangular Decomposition of a Parametric Semi-1lgebraic System

Let $F = \{f_1, \ldots, f_s\}$, $P = \{p_1, \ldots, p_r\}$ be two finite sets of polynomials of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. We denote by $[F, P_>]$ the *basic semi-algebraic system* $\{f_1 = 0, \ldots, f_s = 0, p_1 > 0, \ldots, p_r > 0\}$. Its zero set in $\mathbb{R}^n$, denoted by $Z_{\mathbb{R}}(F, P_>)$, is called a basic semi-algebraic set. It is well known that any semi-algebraic set is a finite union of basic semi-algebraic sets of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. The set $cs := \{(u, y) \in \mathbb{C}^n \mid f_1(u, y) = 0, \ldots, f_s(u, y) = 0, p_1(u, y) \neq 0, \ldots, p_r(u, y) \neq 0\}$ is called the *associated constructible set* of $Z_{\mathbb{R}}(F, P_>)$.

In this section, we introduce the concept of the comprehensive triangular decomposition of a parametric basic semi-algebraic system and propose an algorithm to compute it.

Definition 11. *Let $R := [T, P]$ be a squarefree regular system of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. We call the pair $A := [T, P_>]$ a* squarefree semi-algebraic system *(SFSAS). The system R is called the* associated regular system *of A.*

Definition 12. *Let $\mathcal{S}$ be a basic semi-algebraic set of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$. Let cs be the associated constructible set of $\mathcal{S}$. A* comprehensive triangular decomposition *of $\mathcal{S}$ is a pair $(\mathcal{C}, (\mathcal{A}_C, C \in \mathcal{C}))$, where*
- *$\mathcal{C}$ is a finite partition of $\mathbb{R}^d$ into nonempty semi-algebraic sets,*
- *for each $C \in \mathcal{C}$, $\mathcal{A}_C$ is a finite set of SFSASes of $\mathbb{Q}[\mathbf{u}, \mathbf{y}]$ such that:*
  - (i) *either $\mathcal{A}_C$ is empty, which means that $\mathcal{S}(u)$ is empty for each $u \in C$;*
  - (ii) *or $\mathcal{A}_C = \{[\varnothing, \{\ \}]\}$, which implies that $cs(u)$ is infinite for each $u \in C$;*
  - (iii) *or C is a connected semi-algebraic set, each $A = [T, P_>] \in \mathcal{A}_C$ satisfies $\mathrm{mvar}(T) = \mathbf{y}$ and for each $u \in C$ we have:*
    - *the associated regular systems of $\mathcal{A}_C$ specializes disjointly well at u,*
    - *for each $A \in \mathcal{A}_C$, $Z_{\mathbb{R}}(A(u))$ is not empty,*
    - *$\mathcal{S}(u) = \uplus_{A \in \mathcal{A}_C} Z_{\mathbb{R}}(A(u))$.*

Next, we provide an algorithm for computing a CTD of a basic semi-algebraic set. It relies on a subroutine for decomposing real constructible sets into connected cylindrically arranged cells of $\mathbb{R}^d$. The subroutine can be easily described via the subroutines MPD, MakeCylindrical and MakeSemiAlgebraic in paper [8].

<u>**Calling sequence.**</u> $\mathsf{CAD}(\mathcal{C})$
Input. $\mathcal{C} := \{C_1, \ldots, C_e\}$ is a set of pairwise disjoint constructible sets of $\mathbb{C}^n$ given by polynomials in $\mathbb{Q}[\mathbf{x}]$ such that $\mathbb{C}^n = \cup_{i=1}^{e} C_i$.
Output. A CAD $\mathcal{E}$ of $\mathbb{R}^n$ such that for each element C of $\mathcal{C}$, the set $C \cap \mathbb{R}^n$ is a union of some cells in $\mathcal{E}$.

Step (1)**.** For $1 \leq i \leq e$, apply operation MPD to the family of regular systems representing C_i, so as to obtain another family $\mathcal{R}_i$ of regular systems representing C_i and whose zero sets are pairwise disjoint.

Step (2)**.** Let $\mathcal{R} := \cup_{i=1}^{e} \mathcal{R}_i$. Call algorithm MakeCylindrical$(\mathcal{R}, n)$, to compute a cylindrical decomposition $\mathcal{D}$ of $\mathbf{K}^n$ such that the zero set of each regular system in $\mathcal{R}$ is a union of some cells in $\mathcal{D}$.

Step (3)**.** Call algorithm MakeSemiAlgebraic to compute a CAD $\mathcal{E}$ of $\mathbb{R}^n$ such that, for each element D of $\mathcal{D}$, the set $D \cap \mathbb{R}^n$ is a union of some cells in $\mathcal{E}$.

Algorithm 3. CTD($\mathcal{S}$)

```
Input: A basic semi-algebraic set $\mathcal{S} := [F, P_>]$ of $\mathbf{k}[\mathbf{u}, \mathbf{y}]$.
Output: A CTD of $\mathcal{S}$.
1  let $cs$ be associated constructible set of $\mathcal{S}$
2  let $(\mathcal{C}, (\mathcal{T}_C, C \in \mathcal{C}))$ be a WDSCTD of $cs$
3  $\mathcal{D} := \mathsf{CAD}(\mathcal{C})$
4  for each $C \in \mathcal{C}$, for each $D \in \mathcal{D}$ such that $D \subseteq C$, let $\mathcal{T}_D = \mathcal{T}_C$
5  $\mathcal{E} := \{\ \}$
6  for $D \in \mathcal{D}$ do
7      if $\mathcal{T}_D = \{\ \}$ then
8          $E := D$; $\mathcal{A}_E := \{\ \}$; $\mathcal{E} := \mathcal{E} \cup \{E\}$
9      else if $\mathcal{T}_D = \{\varnothing\}$ then
10         $E := D$; $\mathcal{A}_E := \{[\varnothing, \{\ \}]\}$; $\mathcal{E} := \mathcal{E} \cup \{E\}$
11     else
12         let $s$ be a sample point of $D$
13         $E := D$; $\mathcal{E} := \mathcal{E} \cup \{E\}$
14         $\mathcal{A}_E := \{\ \}$
15         for $T \in \mathcal{T}_D$ do
16             $A := [T_{\mathbf{y}}, P_>]$
17             if $A(s)$ has real solutions then
18                 $\mathcal{A}_E := \mathcal{A}_E \cup \{A\}$
19 return $(\mathcal{E}, (\mathcal{A}_E, E \in \mathcal{E}))$
```

5 Conclusion

Based on the notion of a comprehensive triangular decomposition (CTD) presented in the last section, we have obtained a framework for analyzing the stability of the equilibria and compute the bifurcations of polynomial dynamical systems. Indeed, we can completely solve the problems introduced in Section 1.

Let us first have a look at Problem 1. Let $F(\mathbf{u}, \mathbf{x})$ be the right hand side polynomial equations of the dynamical system (1). It is usually required that $\mathbf{u}$ and $\mathbf{x}$ are both positive. Let $P(\mathbf{u}, \mathbf{x})$ be the corresponding set of positive inequality constraints. Let $(\mathcal{C}, (\mathcal{A}_C, C \in \mathcal{C}))$ be a CTD of $\mathcal{S} = [F, P_>]$. In the practice of dynamical systems, only the cells above which $\mathcal{S}$ has finitely many complex solutions are interesting. This fact has motivated our definition of the CTD of a semi-algebraic system. Let $C \in \mathcal{C}$ be a cell above which $\mathcal{S}$ has finitely many complex solutions, one of them at least being real, that is, a cell of type (iii) in Definition 12. The set C is a connected semi-algebraic subsets of $\mathbb{R}^d$, above which $\mathcal{A}_C$ is a finite set of SFSASes whose solutions are disjoint graphs of continuous functions above C; moreover the union of the graphs of these functions is exactly $C \cap Z_{\mathbb{R}}(\mathcal{S})$. Therefore, Problem 1 is solved.

Next, we look at Problem 2. A first and direct approach consists of computing a CTD of the system $\mathcal{S}$ augmented with the inequalities $\Delta_i > 0$, $1 \leq i \leq m$, where the Δ_i are the Hurwitz determinants, see Definition 1. A second approach consists of computing a CTD of the system $\mathcal{S}$ augmented with the inequality $\Delta_m > 0$ only and then apply the Boundary Crossing Theorem, that is Theorem 13.

Similarly, for each of the three other problems on bifurcation, we will first produce a semi-algebraic system by means of results in Section 2 and then apply CTD to solve it.

References

1. Arnold, V.I.: Ordinary Differential Equations. Springer, Heidelberg (1992)
2. Chen, C., Davenport, J.H., May, J., Moreno Maza, M., Xia, B., Xiao, R.: Triangular decomposition of semi-algebraic systems. In: Watt, S.M. (ed.) Proceedings ISSAC 2010, pp. 187–194 (2010)
3. Carr, J.: Applications of Centre Manifold Theory. Springer, Heidelberg (1981)
4. Carr, J.: Applications of Centre Manifold Theory. Springer, Heidelberg (1981)
5. Chen, C.: Algebraic analysis of stability for biological systems and the implemetation of a software pakage. Master's thesis, Peking University (2006) (in Chinese)
6. Chen, C., Golubitsky, O., Lemaire, F., Maza, M.M., Pan, W.: Comprehensive Triangular Decomposition. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2007. LNCS, vol. 4770, pp. 73–101. Springer, Heidelberg (2007)
7. Chen, C., Moreno Maza, M.: Algorithms for computing triangular decompositions of polynomial systems. In: CoRR, abs/1104.0689 (2011)
8. Chen, C., Moreno Maza, M., Xia, B., Yang, L.: Computing cylindrical algebraic decomposition via triangular decomposition. In: ISSAC 2009, pp. 95–102 (2009)
9. Chen, G., Dora, J.D.: Rational normal form for dynamical systems by Carleman linearization. In: Dooley, S. (ed.) Proc. 1999 International Symposium on Symbolic and Algebraic Computation (ISSAC), pp. 165–172. ACM Press, New York (1999)
10. Chen, G., Dora, J.D.: An algorithm for computing a new normal form for dynamical systems. Journal of Symbolic Computation 29(3), 393–418 (2000)
11. Chen, G., Dora, J.D., Stolovitch, L.: Nilpotent normal form via Carleman linearization (for systems of ordinary differential equations). In: Watt, S. (ed.) Proc. 1991 International Symposium on Symbolic and Algebraic Computation (ISSAC), pp. 281–288. ACM Press, New York (1991)
12. Frazer, R.A., Duncan, W.J.: On the criteria for the stability of small motions. Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character 124(795), 642–654 (1929)
13. Freire, E., Gamero, E., Ponce, E., García Franquelo, L.: An algorithm for symbolic computation of center manifolds. In: Proc. of ISAAC 1988, pp. 218–230. Springer, London (1989)
14. Fuller, A.T.: Conditions for a matrix to have only characteristic roots with negative real parts. Journal of Mathematical Analysis and Applications 23, 71–98 (1968)
15. Gantmacher, F.R.: The Theory of Matrices. Chelsea Publishing Company, New York (1959)
16. Gatermann, K., Eiswirtha, M., Sensse, A.: Toric ideals and graph theory to analyze Hopf bifurcations in mass action systems. Journal of Symbolic Computation 40(6), 1361–1382 (2005)

17. Gatermann, K., Hosten, S.: Computational algebra for bifurcation theory. Journal of Symbolic Computation 40(4-5), 1180–1207 (2005)
18. Gonzalez, L., Lombardi, H., Recio, T., Roy, M.-F.: Sturm-habicht sequence. In: ISSAC 1989: Proceedings of the ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation, pp. 136–146. ACM, New York (1989)
19. Guckenheimer, J., Myers, M., Sturmfels, B.: Computing Hopf bifurcations I. SIAM J. Num. Anal. 34(1), 1–21 (1997)
20. Guckenheimer, J., Myers, M., Sturmfels, B.: Computing hopf bifurcations i. SIAM J. Numer. Anal. 34(1), 1–21 (1997)
21. Hale, J., Koçak, H.: Dynamics and Bifurcations. Springer, Heidelberg (1991)
22. Hong, H., Liska, R., Steinberg, S.: Testing stability by quantifier elimination. Journal of Symbolic Computation 24(2), 161–187 (1997)
23. El Kahoui, M., Weber, A.: Deciding hopf bifurcations by quantifier elimination in a software-component architecture. J. Symb. Comput. 30(2), 161–179 (2000)
24. El Kahoui, M., Weber, A.: Symbolic equilibrium point analysis in parameterized polynomial vector fields. In: Ganzha, V., Mayr, E., Vorozhtsov, E. (eds.) Computer Algebra in Scientific Computing (CASC 2002), pp. 71–83 (2002)
25. Kuznetsov, Y.A.: Elements of Applied Bifurcation Theory. Springer, Heidelberg (1998)
26. Laurent, M.: Prion diseases and the "protein only" hypothesis: a theoretical dynamic study. Biochem. J. 318, 35–39 (1996)
27. Liu, X., Corless, R.M., Geddes, K.O.: Computation of center manifolds. Technical Report TR-00-15, Ontario Research Centre for Computer Algebra, 12 pages (2000), http://www.orcca.on.ca/TechReports
28. Miller, R.K., Michel, A.N.: Ordinary Differential Equations. Academic Press, London (1982)
29. Mishra, B.: Algorithmic Algebra. Springer, New York (1993)
30. Nayfeh, A.H.: Method of Normal Forms. Wiley Series in Nonlinear Sciences. John Wiley & Sons, New York (1993)
31. Niu, W., Wang, D.M.: Algebraic approaches to stability analysis of biological systems. Mathematics in Computer Science 1, 507–539 (2008)
32. Perko, L.: Differential Equations and Dynamical Systems. Springer-Verlag New York, Inc., New York (1991)
33. Schaeffer, D.G., Golubitsky, M.: Singularities and Groups in Bifurcation Theory, vol. 1. Springer, Heidelberg (1984)
34. Vallier, L.: An algorithm for the computation of normal forms and invariant manifolds. In: Proc. of ISSAC 1993, pp. 225–233. ACM Press, New York (1993)
35. Wang, D.M., Zheng, Z.M.: Differential Equations with Symbolic Computation. Birkhäuser Verlag, Basel (2005)
36. Wang, D., Xia, B.: Stability analysis of biological systems with real solution classfication. In: Kauers, M. (ed.) Proc. 2005 International Symposium on Symbolic and Algebraic Computation (ISSAC), pp. 354–361. ACM Press, New York (2005)
37. Yang, L.: Recent advances on determining the number of real roots of parametric polynomials. J. Symb. Comput. 28(1-2), 225–242 (1999)
38. Yang, L., Hou, X., Xia, B.: A complete algorithm for automated discovering of a class of inequality-type theorems. Science in China, Series F 44(6), 33–49 (2001)
39. Yu, P., Yuan, Y.: An efficient method for computing the simplest normal forms of vector fields. Int. J. Bifurcations & Chaos 13(1), 19–46 (2003)

A Example

In this section we present a complete process for analyzing the stability of a biochemistry network by means of the tools presented in this paper.

A.1 The Description of the Model

In [26], Laurent proposed a model for the dynamics of diseases of the central nervous system caused by prions, such as scrapie in sheep and goat, and "mad cow disease" or Creutzfeldt-Jacob disease in humans. The model is based on the protein-only hypothesis, which assumes that infection can be spread by particular proteins (prions) that can exist in two isomeric forms. The normal form PrP^C is harmless, while the infectious form PrP^{Sc} catalyzes a transformation from the normal form to itself. A natural question is whether *a small amount of* PrP^{Sc} *cause prion disease.*

The generic kinetic scheme of prion diseases is illustrated as follows:

$$\begin{tikzcd} \arrow[d, "1"] & & \\ PrP^C \arrow[r, "3"] \arrow[d, "2"] & PrP^{Sc} \arrow[r, "4"] & \text{Aggregates.} \\ {} & & \end{tikzcd}$$

Denote by $\left[PrP^C\right]$ and $\left[PrP^{Sc}\right]$ be respectively the concentrations of PrP^C and PrP^{Sc}. Let ν_i be the rate of Step i for $i = 1, \ldots, 4$. In the above diagram, Step 1 corresponds to the synthesis of native PrP^C, which is considered in the present analysis as a zero-order kinetic process, that is $\nu_1 = k_1$ for some constant k_1. Output reactions (Steps 2 and 4, which correspond to the degradation of native PrP^C and to the formation of aggregates respectively) are taken as first-order rate equations: $\nu_2 = k_2 \left[PrP^C\right]$, $\nu_4 = k_4 \left[PrP^{Sc}\right]$. Step 3 corresponds to the transformation from PrP^C to PrP^{Sc}, which is a nonlinear process

$$\nu_3 = \left[PrP^C\right] \frac{a\left(1 + b\left[PrP^{Sc}\right]^n\right)}{1 + c\left[PrP^{Sc}\right]^n}.$$

Hence we can describe the model by the following differential equations:

$$\frac{\mathrm{d}\left[PrP^C\right]}{\mathrm{d}t} = \nu_1 - \nu_2 - \nu_3$$
$$\frac{\mathrm{d}\left[PrP^{Sc}\right]}{\mathrm{d}t} = \nu_3 - \nu_4$$

To simplify notation, we set $x = \left[PrP^C\right]$, $y = \left[PrP^{Sc}\right]$. The model is therefore described by the dynamical system:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = k_1 - k_2 x - ax\frac{(1 + by^n)}{1 + cy^n}$$
$$\frac{\mathrm{d}y}{\mathrm{d}t} = ax\frac{(1 + by^n)}{1 + cy^n} - k_4 y$$

where experiments suggest to set $b = 2$, $c = 1/20$, $n = 4$, $a = 1/10$, $k_4 = 50$ and $k_1 = 800$. Now we have:

$$\begin{cases} \frac{\mathrm{d}x}{\mathrm{d}t} = f_1 \\ \frac{\mathrm{d}y}{\mathrm{d}t} = f_2 \end{cases} \text{with} \begin{cases} f_1 = \frac{16000+800y^4-20k_2x-k_2xy^4-2x-4xy^4}{20+y^4} \\ f_2 = \frac{2(x+2xy^4-500y-25y^5)}{20+y^4} \end{cases}. \tag{6}$$

Recall that a constant solution of the above differential equations is called an *equilibrium*, that is a point $(x, y) \in \mathbb{R}^2$ at which the right hand side equations vanish. By Routh-Hurwitz criterion (x, y) is asymptotically stable if

$$\Delta_1 = -(\frac{\partial f_1}{\partial x} + \frac{\partial f_2}{\partial y}) > 0 \text{ and } a_2 = \frac{\partial f_1}{\partial x} \cdot \frac{\partial f_2}{\partial y} - \frac{\partial f_1}{\partial y} \cdot \frac{\partial f_2}{\partial x} > 0.$$

In system (6), let p_1 and p_2 be respectively the numerators of f_1 and f_2. The parametric semi-algebraic sets $\mathcal{S}_1 : \{p_1 = p_2 = 0, k_2 > 0\}$ and $\mathcal{S}_2 : \{p_1 = p_2 = 0, k_2 > 0, \Delta_1 > 0, a_2 > 0\}$ encode respectively the equilibria and the asymptotically stable hyperbolic equilibria of System (6).

A.2 Studying the Equilibria with CTD

Firstly, we compute a CTD of $\mathcal{S}_1$. Let

$$R_1 = 100000k_2^8 + 1250000k_2^7 + 5410000k_2^6 + 8921000k_2^5 - 9161219950k_2^4 - 5038824999k_2^3 - 1665203348k_2^2 - 882897744k_2 + 1099528405056.$$

The polynomial R_1 has four real roots, two of them are positive. We denote them by $0 < \alpha_1 < \alpha_2$. Then the real line $\mathbb{R}$ is partitioned into 6 connected cells: $k_2 \leq 0$, $0 < k_2 < \alpha_1$, $k_2 = \alpha_1$, $\alpha_1 < k_2 < \alpha_2$, $k_2 = \alpha_2$ and $k_2 > \alpha_2$. For the first cell, namely $k_2 \leq 0$, there is no associated SFSAS, which implies that $\mathcal{S}$ has no real solutions. The second, fourth and sixth cells are associated with the same SFSAS, which is

$$A_1 := \begin{cases} (2y^4 + 1)x - 25y^5 - 500y &= 0 \\ (k_2 + 4)y^5 - 64y^4 + (2 + 20k_2)y - 32 &= 0 \\ k_2 &> 0. \end{cases}$$

The third and fifth cells are associated with the SFSAS A_2, which will not be displayed here due to its size. For each of the sixth cells, we can compute a sample point and substitute it into the corresponding SFSAS. Then we obtain the number of real solutions above the six cells, which are respectively $0, 1, 2, 3, 2, 1$. To summarize, we have the following conclusion.

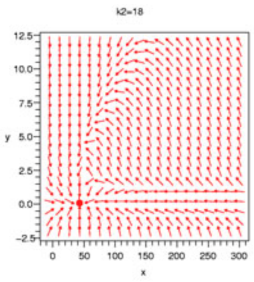
Conclusion 1. *Assume $k_2 > 0$. If $R_1 > 0$, then System (6) has 1 equilibrium; if $R_1 = 0$, then System (6) has 2 equilibria; if $R_2 < 0$, then System (6) has 3 equilibria.*

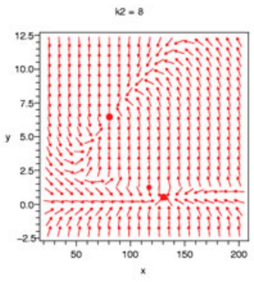
Similarly, we can also compute a CTD of $\mathcal{S}_2$ and then count the number of asymptotically stable hyperbolic equilibria above each cell. Let R_2 be the following polynomial.

$$\begin{aligned}R_2 &= 10004737927168k_2^9 + 624166300700672k_2^8 + 7000539052537600k_2^7\\ &+ 45135589467012800k_2^6 - 840351411856453750k_2^5 - 5009800435224844687 5k_2^4\\ &- 27388168989455000000k_2^3 - 86752092666960000000k_2^2\\ &+ 102960917356800000000k_2 + 59325460641024000000000.\end{aligned}$$

The following conclusion summarizes the conditions for the stability and bifurcation of System (6).

Conclusion 2. *Assume $k_2 > 0$. If $R_1 > 0$ (Figures 1 and 3), then the system has* one *hyperbolic equilibrium, which is asymptotically stable; if $R_1 < 0$ and $R_2 \neq 0$ (Figure 2), then the system has* three *hyperbolic equilibria, two of which are asymptotically stable, the other one being unstable; if $R_1 = 0$ or $R_2 = 0$ hold, the system experiences a* bifurcation.





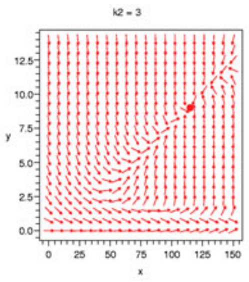


Fig. 1. **Fig. 2.** **Fig. 3.**

Remark 5. *This generalizes the illustrated results of Fig.1(c) in [26], where only concrete values of k_2 are given to make sure that system (6) is bistable. By symbolic methods presented here, we can give the precise condition.*

A.3 Explanation of the Experimental Results

From these figures, we also observe that, in Figure 1, the concentration of PrP^{Sc} (y-coordinate) finally becomes *low* and thus the system enters a *harmless* state. Conversely, in Figure 3 the concentration of PrP^{Sc} goes *high* and thus the systems enters a *pathogenic* state. In Figure 2, the system exhibits bistability, the *initial concentrations* of PrP^{Sc} determines whether the final state pathogenic or not. We thus deduce the following facts, as stated in paper [26]:

- The turnover rate k_2 determines whether it is possible for a pathogenic state to occur.
- As an *answer* to our question, a small amount of PrP^{Sc} does not lead to a pathogenic state when k_2 is large enough.
- Compounds that inhibit addition of PrP^{Sc} can be seen as a possible therapy against prion diseases. However, compounds that *increase the turnover rate* k_2 would be the best therapeutic strategy against prion diseases.

Normal Forms of Two $p : -q$ Resonant Polynomial Vector Fields

Victor Edneral[1] and Valery G. Romanovski[2,3]

[1] Skobeltsyn Institute of Nuclear Physics
of Lomonosov Moscow State University
Leninskie Gory 1, Moscow, 119991, Russia
edneral@theory.sinp.msu.ru

[2] CAMTP - Center for Applied Mathematics and Theoretical Physics
University of Maribor, Krekova 2, Maribor SI-2000, Slovenia

[3] Faculty of Natural Science and Mathematics, University of Maribor
Koroška cesta 160, SI-2000 Maribor, Slovenia
valery.romanovsky@uni-mb.si

Abstract. We investigate a property of normal forms of $p : -q$ resonant vector fields, which is related to isochronicity. The problem is reduced to studying polynomial ideals and their varieties which is performed using tools of computational algebra.

Keywords: planar differential equations, isochronicity, linearizability, normal forms, polynomial ideals, computational algebra.

1 Introduction

Consider a two-dimensional system of differential equations of the form

$$\begin{aligned} \dot{x} &= -\,y + a_0 x + P(x,y), \\ \dot{y} &= \;\; x + a_0 y + Q(x,y), \end{aligned} \tag{1}$$

where P and Q are analytic functions whose series expansions start with at least quadratic terms. The origin is a strong focus when $a_0 \neq 0$, a weak focus or a center when $a_0 = 0$. If the origin is a center, and the period of all solutions close to the origin is the same, then the singular point is called an isochronous center. The problem of isochronous center has been studied by many authors (see, e.g. [2,3,6,7,18] and references therein).

The notion of isochronicity can be generalized to the case when the singularity at the origin of (1) is of the focus type. A natural generalization proposed in [16] and widely used (see, e.g. [2,19] and references therein) is as follows.

Definition 1. *It is said that the singular point O at the origin of* (1) *(which is either a center or a focus) is isochronous if there is a polar ray $L = \varphi_0$ such that the minimal time required for each trajectory started at L sufficiently close to O to return to L is the same and equal to 2π.*

Some examples of isochronous foci in the sense of this definition for system (1) with P and Q being homogeneous polynomials are given by Rudenok [19]. Other definitions of an isochronous singular point which is not necessarily a center are given in [1,11].

A difficult problem arises how to determine isochronous systems inside a given family of polynomial systems. To our knowledge there are no regular methods to perform this task even for the case when P and Q are quadratic polynomials. We will limit our consideration to the case of polynomial systems (1) with $a_0 = 0$, that is

$$\begin{aligned} \dot{x} &= -\, y + P(x,y), \\ \dot{y} &= \, x + Q(x,y). \end{aligned} \tag{2}$$

The problem of isochronicity for system (2) has common features with the problem of distinguishing between a center and a focus. System (2) can be transformed to the normal form

$$\begin{aligned} \dot{u} &= -\, v + \sum_{j\geq 1}(a_j u - b_j v)(u^2+v^2)^j, \\ \dot{v} &= \, u + \sum_{j\geq 1}(b_j u + a_j v)(u^2+v^2)^j. \end{aligned} \tag{3}$$

It is well known (see, e.g. [2,4,18]) that if in the normal form $a_i = 0$ for all $i = 1, 2, \ldots$, then the system has a center at the origin, otherwise the origin is a focus. The normalizing transformation is not unique, however, the first non-zero coefficient a_i of the normal form does not depend on a choice of the normalizing transformation. When we are interested in determining isochronous systems within family (2), we use the fact that the condition $b_1 = b_2 = \cdots = 0$ determines isochronous systems (the coefficients b_k are called the *azimuthal coefficients*). However, by a surprising recent results of [1], the coefficients b_i of normal forms can depend on the choice of the normalizing transformation in such a way that if in the normal form there is a coefficient b_i different from zero it does not mean yet that the system is non-isochronous. This is a principal difference from the case of the center-focus problem. Thus, the problem of determining isochronous systems within a given family (2) appears to be more difficult than the problem of distinguishing between a center and a focus.

Since in the case of isochronicity we do not know a priori which normalizing transformation produces the normal form with all azimuthal coefficients b_i equal to zero, in this paper we will study normalization by means of the so-called distinguished transformations [4], that is the transformations where all resonant coefficients are chosen to be equal to zero. Computing a normal form is a highly laborious procedure. To simplify computations of a normal form of (2) it is convenient to complexify the real system by setting $x_1 = x + iy,\ x_2 = x - iy$. Then we obtain the system

$$\begin{aligned} \dot{x}_1 &= ix_1 + X_1(x_1,x_2) \\ \dot{x}_2 &= -ix_2 + X_2(x_1,x_2) \end{aligned}, \qquad X_2(x_1,\bar{x}_1) = \overline{X_1(x_1,\bar{x}_1)}\,. \tag{4}$$

A natural generalization of (4) is the system

$$\begin{aligned} \dot{x}_1 &= \quad px_1 + X_1(x_1, x_2) \\ \dot{x}_2 &= -qx_2 + X_2(x_1, x_2), \end{aligned} \tag{5}$$

where X_1 and X_2 are polynomials without constant and linear terms. The notion of the generalized isochronous center for system (5) was introduced in [20]. The generalized isochronicity of (5) is equivalent to the linearizability of the system.

In this paper, we study transformations of a 1:-2 resonant system (5) by means of distinguished normalizing substitutions in the case when X_1 and X_2 are homogeneous polynomials of degree 2 and of a 1:-3 resonant system with X_1 and X_2 being homogeneous cubic polynomials. The problem of our interest is an interrelation of the linearizability of system (5) and the vanishing of the "generalized" azimuthal coefficients of (5). It is shown that for two polynomial families mentioned above, the vanishing of "generalized" azimuthal coefficients yields linearizations of the systems.

2 Preliminaries

Applying the normalizing transformation

$$\begin{aligned} x_1 =& y_1 + \sum_{j+k\geq 2;\, j,k\geq 0} h_1^{(j,k)} y_1^j y_2^k\,, \\ x_2 =& y_2 + \sum_{j+k\geq 2;\, j,k\geq 0} h_2^{(j,k)} y_1^j y_2^k\,, \end{aligned} \tag{6}$$

we reduce (5) to the normal form

$$\begin{aligned} \dot{y}_1 =& \quad y_1(p + Y_1(y_1^q y_2^p)), \\ \dot{y}_2 =& -y_2(q - Y_2(y_1^q y_2^p)), \end{aligned} \tag{7}$$

where

$$Y_1(y_1, y_2) = \sum_{j=1}^{\infty} Y_1^{(qj,pj)} (y_1^q y_2^p)^j \quad \text{and} \quad Y_2(y_1, y_2) = \sum_{j=1}^{\infty} Y_2^{(qj,pj)} (y_1^q y_2^p)^j, \tag{8}$$

(see, e.g. [18, Chapter 3] for more details on the normalizing procedure and the correspondence between normal forms of real systems and their complexifications).

We define

$$G = qY_1 + pY_2, \qquad H = qY_1 - pY_2, \tag{9}$$

$w = y_1^q y_2^p$ and write

$$G(w) = \sum_{k=1}^{\infty} G_k w^k \quad \text{and} \quad H(w) = \sum_{k=1}^{\infty} H_k w^k.$$

Note that if $G(w) \equiv 0$ then the normalizing transformation is convergent, and the system admits an analytical first integral of the form $\Psi = x^q y^p + h.o.t.$ (see, e.g. [18] for details). In this case, the singular point at the origin of (5) is called a center. If $G \equiv 0$, and $H \equiv 0$ for some normalizing transformation, then the normal form is linear for any normalizing transformation (see, e.g. [4], [5], [18, Theorem 4.2.2]), and the system is called linearizable. The coefficients H_k of the function $H(w)$ can be regarded as the "generalized" azimuthal coefficients of system (5).

The resonant coefficients in normalizing transformations can be chosen arbitrarily yielding infinitely many normalizing transformations and normal forms. Among them of a particular importance is the distinguished transformation [4] (when the resonance coefficients are chosen to be equal to zero), since usually such transformations are used for performing a normalization. A normal form computed by means of a distinguished transformation is called the distinguished normal form. Generalizing the notion of d-isochronicity introduced in [13] for systems of the form (4) to systems of the form (5) we give the following definition.

Definition 2. *We say that the origin is a d-isochronous singular point for* (5) *if the distinguished normalizing substitution* (6) *transforms the system to the normal form* (7) *with* $H \equiv 0$. *The ideal* $\mathcal{H} = \langle H_1, H_2, \ldots \rangle$ *generated by the coefficients* H_k *of the function* H *is called the d-isochronicity ideal of the system* (5)*.*

The notion of d-isochronicity is a generalization of the notion of isochronicity since in the case when $p = q = i$ and (5) is a complexification of (2) the d-isochronous singular point of (5) corresponds to the isochronous singular point of (2) (see [13] for more details).

3 D-isochronicity of Two Systems

The study of d-isochronicity involves very laborious computations. The two systems which we treat below appear to be the simplest non-trivial systems with $p \neq q$ where it is possible to compute sufficiently many terms of normal forms to answer the question of d-isochronicity.

First we consider the complex quadratic system

$$\begin{aligned} \dot{x} &= x(1 - a_{10}x - a_{01}y - a_{-12}x^{-1}y^2), \\ \dot{y} &= -y(2 - b_{01}y - b_{10}x - b_{2,-1}y^{-1}x^2). \end{aligned} \tag{10}$$

The center problem for system (10) has been solved in [9], and the linearizability problem has been solved in [7].

Theorem 1. *System* (10) *has a d-isochronous singular point at the origin if and only if it is linearizable.*

Proof. We have computed the distinguished normal form of (10) up to order 16 using the package described in [8] and found

$$Y_1^{(2,1)} = -(10a_{01}a_{10}b_{10} + 10a_{01}b_{10}^2 + 10a_{01}^2b_{2,-1}+$$
$$+4a_{10}a_{-12}b_{2,-1} + 5a_{01}b_{01}b_{2,-1} + 12a_{-12}b_{10}b_{2,-1})/20,$$
$$Y_2^{(2,1)} = (-10a_{10}b_{01}b_{10} + 20a_{01}b_{10}^2 + 10b_{01}b_{10}^2 + 10a_{01}^2b_{2,-1}-$$
$$-4a_{10}a_{-12}b_{2,-1} + 15a_{01}b_{01}b_{2,-1} + 5b_{01}^2b_{2,-1} + 23a_{-12}b_{10}b_{2,-1})/20.$$

The other polynomials become too long, so we do not write them here, however, the interested reader can compute them using available computational facilities.

Then, we define

$$H_1 = 2Y_1^{(2,1)} - Y_2^{(2,1)}, \dots, H_5 = 2Y_1^{(10,5)} - Y_2^{(10,5)},$$

$$G_1 = 2Y_1^{(2,1)} + Y_2^{(2,1)},\ G_2 = 2Y_1^{(4,2)} + Y_2^{(4,2)},\ G_5 = 2Y_1^{(10,5)} + Y_2^{(10,5)}.$$

We tried to find the irreducible decomposition of the variety of the ideal

$$\mathcal{H}_5 = \langle H_1, \dots, H_5\rangle$$

using the routine *minAssGTZ* of Singular [17] which computes the minimal associate primes of a polynomial ideal using the algorithm of [10], but we were unable to complete the calculations working in the field of characteristic 0 and even in the field of the finite characteristic 32003. However, it turns out, polynomials $G_1, \dots, G_5$ vanish on the variety of the ideal $\mathcal{H}_5$. To see this we use the radical membership test. Indeed, computing in the field of characteristic 32003 the reduced Groebner bases of the ideals $\langle 1-wG_k, \mathcal{H}_5\rangle$ $(k = 1, 2, 3, 4, 5)$ we obtain in each case $\{1\}$. Thus, if $H_1 = \cdots = H_5 = 0$ then $G_1 = \cdots = G_5 = 0$. However, $\mathbf{V}(\langle G_1, \dots, G_5\rangle)$ is the center variety of (10) ([9]). Thus, if $H_1 = \cdots = H_5 = 0$ then $G \equiv 0$ yielding that the distinguished transformation to the normal form is convergent, and the system has a center at the origin. Although we cannot compute the minimal associate primes of the ideal $\mathcal{H}_5$, it turns out, we can compute the primes of the ideal $I = \langle G_1, \dots, G_5, \mathcal{H}_5\rangle$, which defines the same variety. Using the routine *minAssGTZ* [17] of SINGULAR [12], which finds the minimal associated primes of a polynomial ideal by means of the the method of [10] and performing the computations in the field of characteristic 32002 we find that the primes are:

$\tilde{Q}_1 = \langle a_{10}-2b_{10}, b_{10}^2+16001b_{01}b_{2,-1}- a_{01}b_{2,-1}, a_{01}b_{10}-16001a_{-12}b_{2,-1}, b_{01}a_{01} + 2a_{01}^2 + b_{10}a_{-12}\rangle$;
$\tilde{Q}_2 = \langle b_{01} + a_{01}, a_{10} - 7996b_{10}, b_{10}^2 + 2743a_{01}b_{2,-1}, a_{01}b_{10} + 1279a_{-12}b_{2,-1}, a_{01}^2 - 12811b_{10}a_{-12}\rangle$;
$\tilde{Q}_3 = \langle b_{01} - 3999a_{01}, a_{10} + b_{10}, b_{10}^2 - 5999a_{01}b_{2,-1}, a_{01}b_{10} - 4a_{-12}b_{2,-1}, a_{01}^2 + 15364b_{10}a_{-12}\rangle$;
$\tilde{Q}_4 = \langle b_{01} + 4a_{01}, a_{10} - 7999b_{10}, b_{10}^2 - 2a_{01}b_{2,-1}, a_{01}b_{10} - 6401a_{-12}b_{2,-1}, a_{01}^2 + 12801b_{10}a_{-12}\rangle$;
$\tilde{Q}_5 = \langle a_{-12}, a_{01}, b_{01}\rangle$;
$\tilde{Q}_6 = \langle a_{-12}, b_{10}, b_{01} + 2a_{01}\rangle$;
$\tilde{Q}_7 = \langle a_{-12}, b_{01} - 10667a_{01}, a_{10} + 3b_{10}, b_{10}^2 - 10668a_{01}b_{2,-1}\rangle$;

$\tilde{Q}_8 = \langle a_{-12}, a_{01}, a_{10}b_{10} - b_{10}^2 + 16001 b_{01}b_{2,-1}\rangle$;
$\tilde{Q}_9 = \langle b_{2,-1}, a_{01}, b_{01}, a_{10} + 2b_{10}\rangle$;
$\tilde{Q}_{10} = \langle b_{2,-1}, a_{01}, b_{01}, a_{10} - 2b_{10}\rangle$;
$\tilde{Q}_{11} = \langle b_{2,-1}, a_{-12}, b_{01} + a_{01}, a_{10} + b_{10}\rangle$;
$\tilde{Q}_{12} = \langle b_{2,-1}, a_{01}, a_{10} - b_{10}\rangle$;
$\tilde{Q}_{13} = \langle b_{2,-1}, b_{10}\rangle$.

Then, using the rational reconstruction algorithm of [21], we obtain the following ideals:

(1) $Q_1 = \langle a_{10} - 2b_{10}, b_{10}^2 - a_{01}b_{2,-1} - \frac{1}{2}b_{01}b_{2,-1}, a_{01}b_{10} + \frac{1}{2}a_{-12}b_{2,-1}, 2a_{01}^2 + a_{01}b_{01} + a_{-12}b_{10}\rangle$;
(2) $Q_2 = \langle a_{01} + b_{01}, a_{10} + \frac{19}{4}b_{10}, b_{10}^2 - \frac{4}{35}a_{01}b_{2,-1}, a_{01}b_{10} - \frac{28}{25}a_{-12}b_{2,-1}, a_{01}^2 - \frac{49}{5}a_{-12}b_{10}\rangle$;
(3) $Q_3 = \langle \frac{11}{8}a_{01} + b_{01}, a_{10} + b_{10}, b_{10}^2 + \frac{25}{16}a_{01}b_{2,-1}, a_{01}b_{10} - 4a_{-12}b_{2,-1}, a_{01}^2 + \frac{64}{25}a_{-12}b_{10}\rangle$;
(4) $Q_4 = \langle 4a_{01} + b_{01}, a_{10} + \frac{7}{4}b_{10}, b_{10}^2 - 2a_{01}b_{2,-1}, a_{01}b_{10} - \frac{2}{5}a_{-12}b_{2,-1}, a_{01}^2 - \frac{1}{5}a_{-12}b_{10}\rangle$;
(5) $Q_5 = \langle a_{-12}, a_{01}, b_{01}\rangle$;
(6) $Q_6 = \langle a_{-12}, b_{10}, 2a_{01} + b_{01}\rangle$;
(7) $Q_7 = \langle a_{-12}, \frac{2}{3}a_{01} + b_{01}, a_{10} + 3b_{10}, b_{10}^2 - \frac{1}{3}a_{01}b_{2,-1}\rangle$;
(8) $Q_8 = \langle a_{-12}, a_{01}, a_{10}b_{10} - b_{10}^2 - \frac{1}{2}b_{01}b_{2,-1}\rangle$;
(9) $Q_9 = \langle b_{2,-1}, a_{01}, b_{01}, a_{10} + 2b_{10}\rangle$;
(10) $Q_{10} = \langle b_{2,-1}, a_{01}, b_{01}, a_{10} - 2b_{10}\rangle$;
(11) $Q_{11} = \langle b_{2,-1}, a_{-12}, a_{01} + b_{01}, a_{10} + b_{10}\rangle$;
(12) $Q_{12} = \langle b_{2,-1}, a_{01}, a_{10} - b_{10}\rangle$;
(13) $Q_{13} = \langle b_{2,-1}, b_{10}\rangle$.

Since the modular computations have been applied we have to check the correctness of the obtained result. To this end, working now in the field of characteristic 0, with *intersect* of Singular we compute $Q = \cap_{k=1}^{13} Q_k$. Then, computing the reduced Groebner bases we check that $\langle 1 - wi_s, Q\rangle = \langle 1\rangle$ for each polynomial i_s from I and $\langle 1 - wq_m, I\rangle = \langle 1\rangle$ for all polynomials q_m in Q. This means that $\mathbf{V}(I) = \cup_{i=1}^{s}\mathbf{V}(Q_i) = \mathbf{V}(Q)$.

The ideals $Q_1, \ldots, Q_{13}$ give the linearizability conditions obtained in [7]. Therefore, the corresponding systems are linearizable, that is there are normalizing substitutions that linearize the systems. Then, by Theorem 4.2.2 of [18] the distinguished normalizing transformations linearize the corresponding systems as well. □

We consider also the system with homogeneous cubic nonlinearities

$$\begin{aligned} \dot{x} &= x(1 - a_{20}x^2 - a_{11}xy - a_{02}y^2 - a_{-13}x^{-1}y^3), \\ \dot{y} &= -y(3 - b_{3,-1}x^3y^{-1} - b_{20}x^2 - b_{11}xy - b_{02}y^2). \end{aligned} \tag{11}$$

The center problem for (11) was solved in [14] and the linearizability problem in [15].

For system (11) we were able to compute normal form only for the order 17 which give 4 pairs of the coefficients of the normal form. However, it is not sufficient to study the d-isochronicity of the system, so we consider a subfamily of (11) setting $a_{-13} = b_{3,-1} = 0$. Then we were able to compute normal form for the order 25 which give 6 pairs of the coefficients.

Theorem 2. *System (11) with $a_{-13} = b_{3,-1} = 0$ has a d-isochronous singular point at the origin if and only if it is linearizable.*

Proof. The first two pairs of the resonant coefficients is

$$\begin{aligned}
Y_1^{(3,1)} &= -(a_{11}a_{20} + a_{11}b_{20})/2,\\
Y_2^{(3,1)} &= (-a_{20}b_{11} + 2a_{11}b_{20} + b_{11}b_{20})/2,\\
Y_1^{(6,2)} &= (6a_{11}^2a_{20}^2 + 3a_{11}a_{20}^2b_{11} - 6a_{11}^2a_{20}b_{20} + 4a_{02}a_{20}^2b_{20}-\\
&\quad - 6a_{11}a_{20}b_{11}b_{20} - 6a_{11}^2b_{20}^2 - 3a_{11}b_{11}b20^2 - 4a_{02}b_{20}^3)/24,\\
Y_2^{(6,2)} &= (-3a_{11}a_{20}^2b_{11} + 6a_{11}^2a_{20}b_{20} + 8a_{20}^2b_{02}b_{20} + 6a_{11}a_{20}b_{11}b_{20}+\\
&\quad + 12a_{11}^2b_{20}^2 - 12a_{02}a_{20}b_{20}^2 - 12a_{20}b_{02}b_{20}^2 + 15a_{11}b_{11}b_{20}^2+\\
&\quad + 12a_{02}b_{20}^3 + 4b_{02}b_{20}^3)/24
\end{aligned}$$

and the other polynomials are too long to be presented here. Similarly as before, we define

$$H_1 = 3Y_1^{(3,1)} - Y_2^{(3,1)}, \dots, H_6 = 3Y_1^{(18,6)} - Y_2^{(18,6)}.$$

Let

$$\mathcal{H}_6 = \langle H_1, \dots, H_6 \rangle.$$

Again, we are unable to compute minimal associate primes working in the field of rational numbers, however, calculations in the field of the prime characteristic 32003 yield the following minimal associate primes:

(1) $Q_1 = \langle b_{20}, a_{20} \rangle$;
(2) $Q_2 = \langle a_{02} + b_{02}, a_{11} + b_{11}, a_{20} + b_{20} \rangle$;
(3) $Q_3 = \langle a_{11}, a_{20} - b_{20} \rangle$;
(4) $Q_4 = \langle = b_{02}, b_{11}, a_{02}, a_{11} \rangle$;
(5) $Q_5 = \langle b_{11}, b_{20}, a_{11} \rangle$;
(6) $Q_6 = \langle b_{11}, a_{02}, a_{11}, a_{20} + 16001b_{20} \rangle$.

Using the algorithm of [21] we find that $16001 \equiv -1/2 \bmod 32003$. Therefore, in the ring of polynomials over the field of rational numbers the ideal Q_6 corresponds to the ideal $\tilde{Q}_6 = \langle b_{11}, a_{02}, a_{11}, a_{20} - \frac{1}{2}b_{20} \rangle$. It is not difficult to check that

$$\sqrt{\mathcal{H}_6} = Q_1 \cap \cdots \cap Q_5 \cap \tilde{Q}_6. \tag{12}$$

Indeed, working in the field of characteristic zero with the routine *intersect* of Singular we first compute the ideal $Q = Q_1 \cap \cdots \cap Q_5 \cap \tilde{Q}_6$. Then, we check that for any polynomial $h \in \mathcal{H}_6$ the reduced Groebner basis of $\langle 1 - wh, Q \rangle$ is $\{1\}$

and for any polynomial $q \in Q$ the reduced Groebner basis of $\langle 1 - wq, \mathcal{H}_6 \rangle$ is $\{1\}$ as well. Therefore, according to the radical membership test, the equality (12) is correct, that is $Q_1, \dots, Q_5, \tilde{Q}_6$ are the minimal associate primes of $\mathcal{H}_6$ in the ring $\mathbb{Q}[a_{20}, a_{11}, \dots, b_{11}, b_{02}]$.

We now observe that zero sets of each of the ideals $Q_1, \dots, Q_5, \tilde{Q}_6$ give particular cases of conditions of Theorem 2 of [15]. Therefore, by the theorem, the corresponding systems are linearizable. $\square$

4 Concluding Remarks

We have studied two polynomial systems and found that for them the d-isochronicity is equivalent to the linearizability. Thus, a question which arises naturally is whether it is true that for polynomial systems (5) with $p \neq q$ d-isochronicity is equivalent to linearizability? It is shown in [13] that in the case $p = q$ the answer to this question is negative. Namely, it is shown in [13] that for the quadratic system (5) with $p = q = 1$ the d-isochronicity is equivalent to the linearizability, however, for the system with homogeneous cubic nonlinearities these properties are not equivalent. It appears the first step to answering the question posed above would be the study of d-isochronicity of system (11) with a_{-13} and $b_{3,-1}$ different from zero. However, it requires developing more efficient algorithms and program packages for computing normal forms.

Acknowledgments. The first author was supported by the Russian Foundation for Basic Research (project 11-01-00023-a) and the grants of the President of the Russian Federation for Support of the Leading Scientific Schools NSh.-3159.2010.2 and NSh.-41422010.2. The second author acknowledges the support by the Slovenian Research Agency and by the Transnational Access Programme at RISC-Linz of the European Commission Framework 6 Programme for Integrated Infrastructures Initiatives under the project SCIEnce (contract no. 026133).

References

1. Algaba, A., Reyes, M.: Characterizing isochronous points and computing isochronous sections. J. Math. Anal. Appl. 355, 564–576 (2009)
2. Amel'kin, V.V., Lukashevich, N.A., Sadovskii, A.P.: Nonlinear Oscillations in Second Order Systems. Belarusian State University, Minsk (1982) (in Russian)
3. Bardet, M., Boussaada, I., Chouikha, A.R., Strelcyn, J.-M.: Isochronicity conditions for some planar polynomial systems II. Bull. Sci. Math. 135, 230–249 (2011)
4. Bibikov, Y.N.: Local Theory of Nonlinear Analytic Ordinary Differential Equations. Lecture Notes in Mathematics, vol. 702. Springer, New York (1979)
5. Bruno, A.D.: Local Methods in Nonlinear Differential Equations. Nauka, Moscow (1979) (in Russian); Springer, Berlin (1989)
6. Chavarriga, J., Sabatini, M.: A survey of isochronous centers. Qual. Theory Dyn. Syst. 1, 1–70 (1999)

7. Christopher, C., Mardešić, P., Rousseau, C.: Normalizable, integrable, and linearizable saddle points for complex quadratic systems in $\mathbb{C}^2$. J. Dynam. Control Systems 9, 311–363 (2003)
8. Edneral, V.F.: On algorithm of the normal form building. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2007. LNCS, vol. 4770, pp. 134–142. Springer, Heidelberg (2007)
9. Fronville, A., Sadovski, A.P., Zoladek, H.: Solution of the $1:-2$ resonant center problem in the quadratic case. Fund. Math. 157, 191–207 (1998)
10. Gianni, P., Trager, B., Zacharias, G.: Gröbner bases and primary decomposition of polynomials. J. Symbolic Comput. 6, 146–167 (1988)
11. Giné, J., Grau, M.: Characterization of isochronous foci for planar analytic differential systems. Proc. Roy. Soc. Edinburgh Sect. A 135(5), 985–998 (2005)
12. Greuel, G.M., Pfister, G., Schönemann, H.: Singular 3.0 A computer algebra system for polynomial computations. Centre for Computer Algebra. University of Kaiserslautern (2005), http://www.singular.uni-kl.de
13. Han, M., Romanovski, V.G.: Isochronicity and normal forms of polynomial systems of ODEs. Submitted to Journal of Symbolic Computation
14. Hu, Z., Romanovski, V.G., Shafer, D.S.: 1:-3 resonant centers on image with homogeneous cubic nonlinearities. Computers & Mathematics with Applications 56(8), 1927–1940 (2008)
15. Kadyrsizova, Z., Romanovski, V.: Linearizability of 1:-3 resonant system with homogeneous cubic nonlinearities. In: International Symposium on Symbolic and Algebraic Computation (ISSAC 2008), Hagenberg, Austria, July 20-23, pp. 255–260. The Association for Computing Machinery, New York (2008)
16. Kukles, I.S., Piskunov, N.S.: On the isochronism of oscillations for conservative and nonconservative systems. Dokl. Akad. Nauk. SSSR 17(9), 467–470 (1937)
17. Pfister, G., Decker, W., Schönemann, H., Laplagne, S.: primdec.lib. A Singular 3.0 library for computing primary decomposition and radical of ideals (2005)
18. Romanovski, V.G., Shafer, D.S.: The Center and Cyclicity Problems: a Computational Algebra Approach. Birkhäuser Boston, Inc., Boston (2009)
19. Rudenok, A.E.: Strong isochronicity of a center and a focus of systems with homogeneous nonlinearities. Differ. Uravn. 45(2), 154–161 (2009) (in Russian); translation in Differ. Equ. 45(2), 159–167 (2009)
20. Wang, Q.L., Liu, Y.R.: Generalized isochronous centers for complex systems. Acta Math. Sin. (Engl. Ser.) 26(9), 1779–1792 (2010)
21. Wang, P.S., Guy, M.J.T., Davenport, J.H.: P-adic reconstruction of rational numbers. ACM SIGSAM Bull. 16(2), 2–3 (1982)

On Muldowney's Criteria for Polynomial Vector Fields with Constraints

Hassan Errami[1], Werner M. Seiler[2], Thomas Sturm[3], and Andreas Weber[1]

[1] Institut für Informatik II, Universität Bonn, Friedrich-Ebert-Allee 144, 53113 Bonn, Germany
{errami,weber}@cs.uni-bonn.de

[2] Institut für Mathematik, Universität Kassel, Heinrich-Plett-Straße 40 34132 Kassel, Germany
seiler@mathematik.uni-kassel.de

[3] Max-Planck-Institut für Informatik , RG 1: Automation of Logic, 66123 Saarbrücken, Germany
sturm@mpi-inf.mpg.de

Abstract. We study Muldowney's extension of the classical Bendixson-Dulac criterion for excluding periodic orbits to higher dimensions for polynomial vector fields. Using the formulation of Muldowney's sufficient criteria for excluding periodic orbits of the parameterized vector field on a convex set as a quantifier elimination problem over the ordered field of the reals we provide case studies of some systems arising in the life sciences. We discuss the use of simple conservation constraints and the use of parametric constraints for describing simple convex polytopes on which periodic orbits can be excluded by Muldowney's criteria.

1 Introduction and Preliminaries

In the study of ordinary differential equations the analysis of periodic trajectories is seen as an important goal in addition to describing the dynamics around fixed points. However, already for two-dimensional polynomial systems this question is related to Hilbert's 16th problem, which is still unsolved [1].

For the two-dimensional case the Bendixson-Dulac criterion gives a sufficient condition for the non-existence of periodic orbits. This criterion is parameterized by a Dulac function, and various techniques have been proposed to construct Dulac functions, which range form algebraic constructions for special systems to techniques involving the solution of certain partial differential equations [2,3,4,5,6].

For the higher-dimensional case there are extensions of the criterion of Bendixson-Dulac that also allow the use of Dulac functions [7]. However, little work seems to have been done to construct Dulac functions in the higher dimensional cases—except for addressing it as a problem [8,9].

Moreover, the common case of algebraic constraints in the simple form of conservation constraints have been used in ad hoc form by many authors—mainly to reduce 3D systems to 2D systems to be able to use the classical Bendixson-Dulac criterion—but have not been discussed in a more general setting.

In case studies of some systems arising in the life sciences we discuss the use of simple conservation constraints in a first line of investigation.

On the example of classical SIRS epidemiological model we show that even in this rather simple case different algorithmic strategies to use conservation constraints might lead to non-conclusive results for some, whereas others lead to conclusive results. Thus the fact that Muldowney's criteria are not coordinate independent pose an algorithmic problem.

In a second line of investigation we discuss the use of parametric constraints for describing simple convex polytopes on which periodic orbits can be excluded by Muldowney's criteria. We will show that for a 3-dimensional model of viral dynamics [10], for which Muldwowney's criteria cannot exclude the existence of periodic orbits on the entire positive real octant, there is a cuboid on which periodic orbits can be excluded.

1.1 The Bendixson-Dulac Criterion for 2-Dimensional Vector Fields

Consider an autonomous planar vector field

$$\frac{dx}{dt} = F(x,y), \qquad \frac{dy}{dt} = G(x,y), \qquad (x,y) \in \mathbb{R}^2.$$

Bendixson in 1901 [11] was the first to give a criterion yielding sufficient conditions for excluding oscillations. Dulac in 1937 [12] was able to generalize the result of Bendixson as follows:

Theorem 1 (Bendixson-Dulac criterion). *Let $B(x,y)$ be a scalar continuously differentiable function defined on a simply connected region $D \subset \mathbb{R}^2$ with no holes in it. If $\frac{\partial(BF)}{\partial x} + \frac{\partial(BG)}{\partial y}$ is not identically zero and does not change sign in D, then there are no periodic orbits lying entirely in D.*

For a modern proof we refer to [13, Theorem 1.8.2].

A common class of Dulac functions uses $B(x,y) = \mathrm{e}^{(U(x,y))}$, see e.g. [3]. By the chain rule the exponential function can be factored out yielding $\mathrm{e}^U \left(\frac{\partial U}{\partial x}F + \frac{\partial U}{\partial y}G + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y}\right)$. Hence, if F, G, $\frac{\partial U}{\partial x}$, and $\frac{\partial U}{\partial y}$ are rational functions, the Bendixson-Dulac criterion remains in the realm of the ordered field of the reals.

1.2 Muldowney's Extensions of the Bendixson-Dulac Criterion to Higher Dimensions

The algorithmic criteria discussed in the following can be seen as generalizations of the Bendixson-Dulac criterion for 2-dimensional vector fields to arbitrary dimensions.

The following theorem was proved by Muldowney [7, Theorem 4.1]: Suppose that one of the inequalities

$$\mu\left(\frac{\partial f^{[2]}}{\partial x}\right) < 0, \qquad \mu\left(-\frac{\partial f^{[2]}}{\partial x}\right) < 0 \tag{1}$$

holds for all $x \in \mathbb{R}^n$. Then the autonomous system with vector field $f : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ has no nonconstant periodic solutions. Here μ is some Lozinskiĭ norm and $f^{[2]}$ is one of the "compound matrices" of the Jacobian of the vector field f defined in [7]. As is also shown in [7] the criterion given in [7, Theorem 4.1] also holds when $x \in C$, where $C \subseteq \mathbb{R}^n$ is open and convex.

Remark. When $n = 2, \partial f^{[2]}/\partial x = \text{Trace}\, \partial f/\partial x = \text{div} f$, so that [7, Theorem 4.1] basically yield the results of Bendixson, i.e. the criterion of Muldowney can be seen as a generalization of the criterion of Bendixson from the planar case to arbitrary dimensions.

According to [7, (2.2)], the following expressions may be used as $\mu\left(\partial f^{[2]}/\partial x\right)$ in [7, Theorem 4.1], if the underlying norms for μ are the 1-norm, ∞-norm, and 2-norm respectively:

$$\max\left\{ \frac{\partial f_r}{\partial x_r} + \frac{\partial f_s}{\partial x_s} + \sum_{q \neq r,s} \left|\frac{\partial f_q}{\partial x_r}\right| + \left|\frac{\partial f_q}{\partial x_s}\right| \;:\; r, s = 1, \ldots, n,\; r \neq s \right\}, \tag{2}$$

$$\max\left\{ \frac{\partial f_r}{\partial x_r} + \frac{\partial f_s}{\partial x_s} + \sum_{q \neq r,s} \left|\frac{\partial f_r}{\partial x_q}\right| + \left|\frac{\partial f_s}{\partial x_q}\right| \;:\; r, s = 1, \ldots, n,\; r \neq s \right\}. \tag{3}$$

$$\lambda_1 + \lambda_2, \tag{4}$$

where λ_1, λ_2 are the two largest eigenvalues of $\left(\partial f^*/\partial x + \partial f/\partial x\right)/2$.

Thus for a formula Γ over the reals defining an open convex subset C of $\mathbb{R}^n$ and an autonomous polynomial vector field $f : \mathbb{R}^n \to \mathbb{R}^n$ a first-order formula φ over the ordered field of the reals defines a sufficient condition such that the vector field defined by f has no non-constant periodic solution on C. As usual with real quantifier elimination we use the language of ordered rings. In addition, we admit function symbols for the maximum and for the absolute values, which are both definable.

Specifically, for the criterion involving the 1-norm we obtain

$$\begin{aligned} \varphi_1 \equiv \forall x_1 \forall x_2 \cdots \forall x_n \Bigg(\Gamma \Longrightarrow \qquad & \qquad (5) \\ \max\left\{ \frac{\partial f_r}{\partial x_r} + \frac{\partial f_s}{\partial x_s} + \sum_{q \neq r,s} \left|\frac{\partial f_q}{\partial x_r}\right| + \left|\frac{\partial f_q}{\partial x_s}\right| \;:\; r, s = 1, \ldots, n,\; r \neq s \right\} < 0 \Bigg), \end{aligned}$$

and for the criterion involving the ∞-norm we obtain

$$\begin{aligned} \varphi_\infty \equiv \forall x_1 \forall x_2 \cdots \forall x_n \Bigg(\Gamma \Longrightarrow \qquad & \qquad (6) \\ \max\left\{ \frac{\partial f_r}{\partial x_r} + \frac{\partial f_s}{\partial x_s} + \sum_{q \neq r,s} \left|\frac{\partial f_r}{\partial x_q}\right| + \left|\frac{\partial f_s}{\partial x_q}\right| \;:\; r, s = 1, \ldots, n,\; r \neq s \right\} < 0 \Bigg). \end{aligned}$$

In [8] the problem of efficient automatic resolution of maxima and absolute values is addressed and computation examples are given. If all variables and parameters are known to be positive, the technique of *positive quantifier elimination* [14,15] can be used, which was first used to solve semi-algebraic criteria for the existence of Hopf bifurcation fixed points [16,17] arising in the context of chemistry and algebraic biology [18,19,20,21].

Extending Muldowney's criteria with Dulac functions. Although a simple generalization of the Dulac criterion to higher dimensions does not seem to hold in the general setting [7], for *positive* functions $0 < r \in C^1(\mathbb{R}^n \longrightarrow \mathbb{R})$ one can replace f by rf in [7, Theorem 4.1], cf. (1). The rather simple proof is given in [7, Remark (d)].

If $B = \mathrm{e}^U$ is used as a Dulac test function then by the chain rule the exponential function can be factored out also for Muldowney's criteria and the criterion remains in the realm of the ordered field of the reals, if all partial derivatives of U are rational functions.

Using conservation constraints. Any algebraic constraints on the vector field can be transferred into the first-order formula over the ordered field of the reals expressing Muldowney's criteria. Simple conservation constraints stating that the sum of certain state variables is constant—conditions that are commonly found in chemical reaction systems or in epidemiological models—will not induce a failure of the degree limited virtual substitution methods [22] for quantifier elimination, if these were successful on the unconstrained system.

Nevertheless, an elimination of a variable by the others in a conservation constrained will reduce the dimension of the system and thus change Muldowney's criteria instead of adding another equality to Muldowney's criteria on the original system. We will report on the results of some systematic tests on the simple SIRS system in Sect. 2.1.

Parametric specification of a convex subset. The first-order formula γ specifying the convex subset on which a proof for the non-existence of periodic orbits is sought by Muldowney's criteria can very well contain parameters, too. The quantifier elimination procedure automatically yields conditions on the parameters that are exact with respect to Muldowney's criteria—potentially not mentioning input parameters if no constraint on any of them is necessary.

In Sect. 2.3 we will use this technique using simple parametric cuboids in a case, for which the Muldowney criteria do not give a conclusive answer on the entire positive real octant, but the specification of a 3-dimensional parametric cuboid shows that only a parametric restriction on one variable is necessary.

2 Case Studies

2.1 The SIRS Epidemiological Model

We consider the widely used SIRS epidemiological model, a parameterized formally 3-dimensional system of ordinary differential equations, cf. (7–9). The systems is widely used and well studied [23,24,25,26,27]. So we will not provide new insights into the structure of the system, but it is well suited as a test object for our algorithmic methods.

To account for the lost of immunity, the classical susceptible (S), infected (I) and recovered (R) model is adjusted by allowing a fraction of the recovered individuals R to move back into the susceptible pool S at a rate γ. This susceptible, infected, recovered and susceptible (SIRS) model is expressed as

$$\frac{d}{dt}S(t) = \mu\,(S(t) + I(t) + R(t)) - \mu\,S(t) - \beta\,S(t)\,I(t) + \gamma\,R(t) \tag{7}$$

$$\frac{d}{dt}I(t) = \beta\,S(t)\,I(t) - (\mu + \nu)\,I(t) \tag{8}$$

$$\frac{d}{dt}R(t) = \nu\,I(t) - (\mu + \gamma)\,R(t) \tag{9}$$

where ν is the rate of loss of infectiousness and the total population size N remains constant (i.e. $S + I + R = N$ is constant). The parameter μ represents both, the birth and mortality rates. Assuming that birth and mortality rates are equal is justified on the grounds that the annual infection rate is considerably higher than the population growth. The parameter β is the transmission rate of the infection.

Using ad-hoc reductions to 2D-models. In the literature, reductions to 2D models using $S + I + R = N$ and replacing a suitable variable are commonly used. However, the question, which variable to choose is never addressed. In the following we give results for all possibilities showing that even for this simple example the results strongly differ. In all cases we use the scaling $N = 1$.

Eliminating R by $R = 1 - (I + S)$. In this case the criterion using the Dulac test function 1 returned the non-conclusive *true* as answer for $\neg\varphi$. However, using the Dulac function $\frac{1}{I(t)}$ the conclusive *false* as answer for $\neg\varphi$ was found within some milliseconds of computation time by REDLOG.

Eliminating I by $I = 1 - (S + R)$. Also in this case the criterion using the Dulac test function 1 returned the non-conclusive *true* as answer for $\neg\varphi$. We also obtained the the non-conclusive *true* as answer for $\neg\varphi$ when using the following Dulac test functions:

$$\frac{1}{R(t)S(t)}$$
$$\frac{1}{S(t)}$$
$$\frac{1}{R(t)}$$
$$R(t)$$
$$S(t)$$

Moreover, the computations using REDLOG did not come up with answers within 60 sec of computation time for several other Dulac test functions.

So using this elimination we did not come up with a conclusive answer by the Muldowney criteria.

Eliminating S by $S = 1 - (I + R)$. In this case the criterion using the Dulac function 1 returned $\beta - \gamma - 2\mu - \nu > 0$ as answer for $\neg\varphi$. Using the Dulac function $\frac{1}{I(t)}$ returned the conclusive *false*, as was the case for the Dulac function $\frac{1}{I(t)R(t)}$; for the Dulac function $\frac{1}{R(t)}$ the criterion returned $\beta - \mu - \nu > 0$. As the conclusive *false* was found for some Dulac function, we thus have proved that the SIRS system does not have periodic orbits on the positive real octant.

2.2 Computations on the 3D Model

Unconstrained model. For the 3D-SIRS model *not* using any conservation constraint the criterion using the Dulac test function 1 returned the non-conclusive *true* as answer for $\neg\varphi$. For all other Dulac tests functions we used we either obtained the non-conclusive *true* as answer for $\neg\varphi$, or REDLOG could not come up with a result within 60 sec of computation time.

Using the constraint $S + I + R = 1$. When adding the equation $S + I + R = 1$ to the input formula for the Muldowney criterion, we obtained for the Dulac test function the following formula as result for $\neg\varphi$:

2.3 A Model of Viral Dynamics

The following example is discussed in more depth in [8]. It consists of a simple mathematical model for the population dynamics of the human immunodefficiency type 1 virus (HIV-1) investigated in [10]. There a three-component model is described involving uninfected CD4 + T-cells, infected such cells and free viruses, whose densities at time t are denoted by $x(t), y(t), v(t)$, respectively.

$$\boxed{\begin{aligned} \frac{d}{dt}x(t) &= s - \mu\, x(t) - kx(t)\, y(t) \\ \frac{d}{dt}y(t) &= kx(t)\, y(t) - \alpha\, y(t) \end{aligned}} \qquad \boxed{\begin{aligned} \frac{d}{dt}x(t) &= s - \mu\, x(t) - \beta\, x(t)\, v(t) \\ \frac{d}{dt}y(t) &= \beta\, x(t)\, v(t) - \alpha\, y(t) \\ \frac{d}{dt}v(t) &= cy(t) - \gamma\, v(t) \end{aligned}}$$

Fig. 1. The 2D- and 3D-Tuckwell-Wan examples

In [10] a simplified two-component model employed by Bonhoeffer et al. [28] is investigated analytically. In [10] using the general Bendixson-Dulac criteria for 2D-vector fields with an ad hoc Dulac function $B(x, y) = 1/y$ it is shown that there are no periodic solutions for the system for positive parameter values and positive values of the state variables, i.e. the biologically relevant ones.

Remark. By "ad hoc" Dulac function we mean that the authors provide this function only and show that it is a Dulac function, but no other functions. No explanations or hints are given to the reader how this function was obtained.

In Table 1 the results for various low-degree rational and polynomial Dulac test functions are summarized. Notice that computation times for generating the formulas are negligible for these examples. Note that for $\neg\varphi$ the answer **false** gives the conclusive proof on the non-existence of periodic orbits on the positive cone.

As can be seen from the computation times given in Table 1 the quantifier elimination problems are not too hard. When performing tests with QEPCAD B [29] we could also solve all of these quantifier elimination problems in less than one second of computation time.

Table 1. Computation Results for the 2D-Tuckwall-Wan example (cf. Fig. 1) on the full positive octant

The computation times are the ones for the positive quantifier elimination in REDLOG.

Tuckwell-Wan 2D model	Used Dulac test function 1	$\frac{1}{x}$	$\frac{1}{y}$	$\frac{1}{xy}$	$\frac{1}{x+y}$	x	y	xy
Comp. Time [sec]	0.07	0.07	0.02	0.02	0.07	0.09	0.07	0.07
Result ($\neg\varphi$)	pc	pc	**false**	**false**	pc	pc	pc	pc

Here pc is the positivity condition on the parameters.

For the 3D-Tuckwell-Wan Model we tried several Dulac test functions but could not exclude the existence of a periodic orbit on $\mathbb{R}^{+3}$ for any of them. When specifying the parametric cube $(0, u_x) \times (0, u_y) \times (0, u_v)$ by adding the conditions $x(t) < u_x$, $y(t) < u_y$, and $v(t) < u_v$ for new parameters $u_x > 0$, $u_y > 0$, and $u_v > 0$—cf. Sect. 1.2—and using the trivial Dulac function 1—we obtain the following first-order formula for $\neg\varphi$ using Muldowney's criterion for the 1-norm (displayed in slightly hand edited form for better readability):

$$\begin{aligned}
&\exists v_1 \exists v_2 \exists v_3 : 0 < v_1 \wedge 0 < v_2 \wedge 0 < v_3 \wedge 0 < u_v \wedge 0 < u_x \wedge 0 < u_y \wedge \\
&\qquad 0 < c \wedge 0 < \mu \wedge 0 < s \wedge 0 < \alpha \wedge 0 < \beta \wedge 0 < \gamma \wedge \\
&\quad v_1 < u_v \wedge v_2 < u_x \wedge v_3 < u_y \wedge \\
&\quad 0 \leq \max(-\gamma - \alpha + |\beta v_2|, -\mu - \beta v_1 - \alpha + |c|, -\gamma - \mu - \beta v_1 + |\beta v_2| + |\beta v_1|)
\end{aligned}$$

This quantifier elimination problem can also be solved "by hand" rather easily, and accordingly in less than 0.1 sec of computation time we obtain by the positive quantifier elimination procedure of REDLOG the following quantifier-free equivalent for φ:

$$\min\left(\frac{\alpha+\gamma}{\beta}, \frac{\mu+\gamma}{\beta}\right) \geq u_x \wedge \alpha + \mu \geq c \tag{10}$$

For better readability we have provided in (10) a slightly hand-edited version of the result formula.

Notice that there is no dependency on u_y and u_v, i. e. we have given a proof that the parametric 3D-Tuckwall-Wan does not have periodic orbits on

$$(0, u_x) \times (0, \infty) \times (0, \infty)$$

provided u_x (and $\alpha, \mu, \gamma, \beta$) fulfills the condition given in (10).

Acknowledgement. This research was supported in part by *Deutsche Forschungsgemeinschaft* within SPP 1489.

References

1. Ilyashenko, Y.: Centennial history of Hilbert's 16th Problem. Bull. Am. Math. Soc., New Ser. 39(3), 301–354 (2002)
2. Osuna, O., Villaseñor, G.: On the Dulac functions. Qualitative Theory of Dynamical Systems, 1–7 (2011)
3. Cherkas, L., Grin, A.: On a Dulac function for the Kukles system. Differential Equations 46, 818–826 (2010)
4. Cherkas, L.: Quadratic systems with maximum number of limit cycles. Differential Equations 45, 1440–1450 (2009), doi: 10.1134/S0012266109100061.
5. Cherkas, L.A., Grin, A.: Algebraic aspects of finding a Dulac function for polynomial autonomous systems on the plane. Differential Equations 37, 411–417 (2001)
6. Cherkas, L.A.: Dulac function for polynomial autonomous systems on a plane. Differential Equations 33, 692–701 (1997)
7. Muldowney, J.S.: Compound matrices and ordinary differential equations. Rocky Mt. J. Math. 20(4), 857–872 (1990)
8. Weber, A., Sturm, T., Abdel-Rahman, E.O.: Algorithmic global criteria for excluding oscillations. Bulletin of Mathematical Biology 73(4), 899–917 (2011)
9. Weber, A., Sturm, T., Seiler, W.M., Abdel-Rahman, E.O.: Parametric qualitative analysis of ordinary differential equations: Computer algebra methods for excluding oscillations (Extended abstract) (Invited talk). In: Gerdt, V.P., Koepf, W., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2010. LNCS, vol. 6244, pp. 267–279. Springer, Heidelberg (2010)
10. Tuckwell, H.C., Wan, F.Y.M.: On the behavior of solutions in viral dynamical models. BioSystems 73(3), 157–161 (2004)
11. Bendixson, I.: Sur les curbes définiés par des équations différentielles. Acta Math. 24, 1–88 (1901)
12. Dulac, H.: Recherche des cycles limites. CR Acad. Sci. Paris 204, 1703–1706 (1937)
13. Guckenheimer, J., Holmes, P.: Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields. Applied Mathematical Sciences, vol. 42. Springer, Heidelberg (1983)
14. Sturm, T., Weber, A.: Investigating generic methods to solve hopf bifurcation problems in algebraic biology. In: Horimoto, K., Regensburger, G., Rosenkranz, M., Yoshida, H. (eds.) AB 2008. LNCS, vol. 5147, pp. 200–215. Springer, Heidelberg (2008)
15. Sturm, T., Weber, A., Abdel-Rahman, E.O., El Kahoui, M.: Investigating algebraic and logical algorithms to solve Hopf bifurcation problems in algebraic biology. Mathematics in Computer Science 2(3), 493–515 (2009); Special issue on 'Symbolic Computation in Biology'

16. El Kahoui, M., Weber, A.: Deciding Hopf bifurcations by quantifier elimination in a software-component architecture. Journal of Symbolic Computation 30(2), 161–179 (2000)
17. Weber, A.: Quantifier elimination on real closed fields and differential equations. In: Löwe, B. (ed.) Algebra, Logic, Set Theory – Festschrift für Ulrich Felgner zum 65, Geburtstag. Studies in Logic, vol. 4, pp. 291–315. College Publications (2007)
18. Boulier, F., Lefranc, M., Lemaire, F., Morant, P.E.: Applying a rigorous quasi-steady state approximation method for proving the absence of oscillations in models of genetic circuits. In: Horimoto, K., Regensburger, G., Rosenkranz, M., Yoshida, H. (eds.) AB 2008. LNCS, vol. 5147, pp. 56–64. Springer, Heidelberg (2008)
19. Boulier, F., Lefranc, M., Lemaire, F., Morant, P.E., Ürgüplü, A.: On proving the absence of oscillations in models of genetic circuits. In: Anai, H., Horimoto, K., Kutsia, T. (eds.) Ab 2007. LNCS, vol. 4545, pp. 66–80. Springer, Heidelberg (2007)
20. Gatermann, K., Eiswirth, M., Sensse, A.: Toric ideals and graph theory to analyze Hopf bifurcations in mass action systems. Journal of Symbolic Computation 40(6), 1361–1382 (2005)
21. Niu, W., Wang, D.: Algebraic approaches to stability analysis of biological systems. Mathematics in Computer Science 1(3), 507–539 (2008)
22. Weispfenning, V.: Quantifier elimination for real algebra—the quadratic case and beyond. Applicable Algebra in Engineering Communication and Computing 8(2), 85–101 (1997)
23. Lin, X.D., van den Driessche, P.: A threshold result for an epidemiological model. Journal of Mathematical Biology 30(6), 647–654 (1992)
24. Hadeler, K.P., van den Driessche, P.: Backward bifurcation in epidemic control. Mathematical Biosciences 146(1), 15–35 (1997)
25. Weber, A., Weber, M., Milligan, P.: Modeling epidemics caused by respiratory syncytial virus (RSV). Mathematical Biosciences 172(2), 95–113 (2001)
26. Brown, C.W., El Kahoui, M., Novotni, D., Weber, A.: Algorithmic methods for investigating equilibria in epidemic modeling. Journal of Symbolic Computation 41(11), 1157–1173 (2006); Special Issue on the Occasion of Volker Weispfenning's 60th Birthday
27. Ponciano, J.M., Capistrán, M.A.: First principles modeling of nonlinear incidence rates in seasonal epidemics. PLoS Computational Biology 7(2), e1001079 (2011)
28. Bonhoeffer, S., Coffin, J.M., Nowak, M.A.: Human immunodeficiency virus drug therapy and virus load. The Journal of Virology 71(4), 3275 (1997)
29. Brown, C.W.: QEPCAD B: A system for computing with semi-algebraic sets via cylindrical algebraic decomposition. ACM SIGSAM Bulletin 38(1), 23–24 (2004)

Knowledge-Based Automatic Generation of Partitioned Matrix Expressions

Diego Fabregat-Traver and Paolo Bientinesi

AICES, RWTH Aachen, Germany
{fabregat,pauldj}@aices.rwth-aachen.de

Abstract. In a series of papers it has been shown that for many linear algebra operations it is possible to generate families of algorithms by following a systematic procedure. Although powerful, such a methodology involves complex algebraic manipulation, symbolic computations and pattern matching, making the generation a process challenging to be performed by hand. We aim for a fully automated system that from the sole description of a target operation creates multiple algorithms without any human intervention. Our approach consists of three main stages. The first stage yields the core object for the entire process, the Partitioned Matrix Expression (PME), which establishes how the target problem may be decomposed in terms of simpler sub-problems. In the second stage the PME is inspected to identify predicates, the Loop-Invariants, to be used to set up the skeleton of a family of proofs of correctness. In the third and last stage the actual algorithms are constructed so that each of them satisfies its corresponding proof of correctness. In this paper we focus on the first stage of the process, the automatic generation of Partitioned Matrix Expressions. In particular, we discuss the steps leading to a PME and the knowledge necessary for a symbolic system to perform such steps. We also introduce Cl1ck, a prototype system written in Mathematica that generates PMEs automatically.

1 Introduction

In the context of the Formal Linear Algebra Methods Environment (FLAME) project [1], a methodology for the systematic derivation of algorithms for matrix operations has been developed and demonstrated. The approach has been successfully applied to all the operations included in the BLAS [2] and RECSY [3,4] libraries and to many included in the LAPACK [5] library. In general, the methodology applies to any operation that can be expressed in a "divide and conquer" fashion. As opposed to the concept of "Autotuning", which indicates the automatic tuning of a given algorithm [6,7,8], the word derivation refers to the actual generation of both algorithms and routines to solve a given target equation [9]. The remarkable results achieved using this methodology are the subject of a series of publications. a) For many standard operations, e.g. the Cholesky and the LU factorizations, all the previously known algorithms were systematically discovered, unifying them under a common root [10]. b) For more

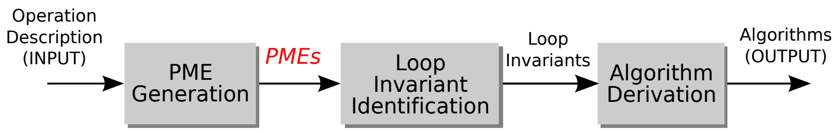


Fig. 1. The three main stages in the process of derivation of algorithms

involved operations like the Sylvester equation and the reduction of a generalized eigenproblem to standard form, the generated family of algorithms included new and better performing ones [11,12]. c) A related methodology for systematic analysis of round-off errors yielded bounds tighter than those previously known [13].

Although successful, the approach presents some limitations. The algorithms are generated through complex symbolic computations, steps often too complicated to be carried out by hand. Motivated by these difficulties, we aim for a symbolic system that, given as input the description of a matrix equation *Eq*, applies the steps dictated by the FLAME methodology to derive a family of algorithms to solve *Eq*. As shown in Fig. 1, the procedure consists of three successive stages—PME Generation, Loop-Invariant Identification, Algorithm Derivation—and is entirely determined by the mathematical description of the input operation.

In the first stage, the Partitioned Matrix Expression (PME) for the input operation is obtained. A PME is a decomposition of the original problem into simpler sub-problems in a "divide and conquer" fashion, exposing the computation to be performed in each part of the output matrices. An example is shown in Box 1. The second stage of the process deals with the identification of Boolean predicates, the Loop-Invariants, that describe the intermediate state of computation for the sought-after algorithms. Loop-invariants can be extracted from the PME, and are at the heart of the automation of the third stage. In the third and last stage of the methodology, each loop-invariant is used to set up a proof of correctness around which the algorithm is finally built. Notice that the objective is not proving the correctness of a given algorithm; vice-versa, the loop-invariant is chosen *before* the algorithm is built. Indeed, the algorithm is constructed to satisfy a given proof of correctness, i.e., to possess the chosen loop-invariant.

$$\boxed{\left(\begin{array}{c} X_T = \Omega(L_{TL}, U, C_T) \\ \hline X_B = \Omega(L_{BR}, U, C_B - L_{BL}X_T) \end{array}\right)}$$

Box 1. Partitioned Matrix Expression for the triangular Sylvester equation

This paper centers around the first stage of the derivation process, the generation of PMEs. To this end we introduce a formalism to input into the system the minimum amount of knowledge about the operation required by a system to

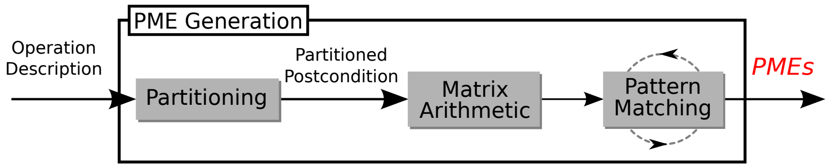


Fig. 2. Steps for the automatic generation of PMEs

perform all the subsequent stages automatically. We then describe the process for transforming an input equation into PMEs. As Fig. 2 shows, such process involves three steps: 1) the partitioning of the operands in the equation, 2) matrix arithmetic involving the partitioned operands, and 3) a sequence of iterations, each consisting of algebraic manipulation and pattern matching. We demonstrate that the process can indeed be automated through Cl1ck[1], a symbolic system written in Mathematica [14] that performs all the steps for the PME generation.

The paper is organized as follows. In Sect. 2 we categorize the input needed by a symbolic system. Partitionings of the operands and inheritance of properties are discussed in Sect. 3, while in Sect. 4 we describe how to use partitionings to obtain PMEs. We draw conclusion in Sect. 5.

2 Input to the System

Our first concern is to establish how a target operation should be formally described. Since we are aiming for a fully-automated system, i.e., without any human intervention, we need a formalism to unequivocally describe a target equation. We choose the language traditionally used to reason about program correctness: equations shall be specified by means of the predicates Precondition (P_{pre}) and Postcondition (P_{post}) [15]. The precondition enumerates the operands that appear in the equation and describes their properties, while the postcondition specifies the equation to be solved.

The Cholesky factorization will serve as an example: given a symmetric positive definite (SPD) matrix A, the goal is to find a lower triangular matrix L such that $LL^T = A$. Box 2 contains the predicates P_{pre} and P_{post} relative to the Cholesky factorization; the notation $L = \Gamma(A)$ indicates that L is the Cholesky factor of A.

Even though such a definition is unambiguous, it does not include all the information needed by a symbolic system to fully automate the derivation process. In Sect. 2.1 we discuss how a system expands its knowledge by "learning of" new equations, and in Sect. 3 we overview the ground knowledge that a system must possess relative to matrix partitioning and inheritance of properties.

[1] The name Cl1ck epitomizes the idea that the effort a user has to make to obtain algorithms consists in just *one click*.

$$L = \Gamma(A) \equiv \begin{cases} P_{\text{pre}} : \{\texttt{Unknown}(L) \ \wedge \ \texttt{LowerTriangular}(L) \ \wedge \\ \qquad\qquad \texttt{Known}(A) \ \wedge \ \texttt{SPD}(A)\} \\ P_{\text{post}} : \{LL^T = A\} \end{cases}$$

Box 2. Formal description for the Cholesky factorization

2.1 Pattern Learning

We refer to the pair of predicates (P_{pre} and P_{post}) in Box 2 as the *pattern* that identifies the Cholesky factorization. Such a pattern establishes that matrices L and A are one the Cholesky factor of the other provided that the constraints in the precondition are satisfied, and L and A are related as dictated in the postcondition ($LL^T = A$). For instance, in the expression

$$XX^T = A - BC,$$

in order to determine whether $X = \Gamma(A - BC)$, the following facts need to be asserted: i) X is an unknown lower triangular matrix; ii) the expression $A - BC$ is a known quantity (A, B and C are known); iii) the matrix $A - BC$ is symmetric positive definite.

The strategy for decomposing an equation in terms of simpler problems greatly relies on pattern matching. In the next section we describe how matrix equations can be rewritten in terms of sub-matrices, resulting in expressions seemingly more complicated than the initial formulation. Such expressions are thus inspected to find segments corresponding to known patterns.

Initially, Cl1ck only knows the patterns for a basic set of operations: addition, multiplication, inversion, and transposition of matrices, vectors and scalars. This information is hard-coded. More complex patterns are instead discovered during the process of PME generation. For instance, the first time the PME for the Cholesky factorization is generated, Cl1ck learns and stores the pattern specified by Box 2. Thanks to such patterns it will then be possible to identify that a Cholesky factorization may be decomposed into a combination of triangular systems and simpler Cholesky factorizations. As Cl1ck's pattern knowledge increases, also does its capability of tackling complex operations.

3 Partitioning and Inheritance

In this section we illustrate the first step towards the PME generation: the partitioning of the operands (Fig. 2). To this end we introduce a set of rules to partition and combine operands and to assert properties of expressions involving sub-operands. The application of these rules to the postcondition yields a predicate called *partitioned postcondition.* Due to constraints imposed by both the structure of the input operands and the postcondition, only few partitioning rules will be admissible.

3.1 Operands Partitioning and Direct Inheritance

As shown in Box 3, a generic matrix A can be partitioned in four different ways. The 1×1 rule (Box 3(d)) is special as it does not affect the operand; we refer to it as the *identity*. For a vector, only the 2×1 and 1×1 rules apply, while for scalars only the identity is admissible. When referring to any of the parts resulting from a non-identity rule, we use the terms sub-matrix or sub-operand, and for 2×2 partitionings we also use the term quadrant.

$$A_{m \times n} \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$$

where A_{TL} is $k_1 \times k_2$

(a) 2×2 rule

$$A_{m \times n} \rightarrow \left(\begin{array}{c} A_T \\ \hline A_B \end{array}\right)$$

where A_T is $k_1 \times n$

(b) 2×1 rule

$$A_{m \times n} \rightarrow \left(\begin{array}{c|c} A_L & A_R \end{array}\right)$$

where A_L is $m \times k_2$

(c) 1×2 rule

$$A_{m \times n} \rightarrow \left(A \right)$$

where A is $m \times n$

(d) 1×1 (identity) rule

Box 3. Rules for partitioning a generic matrix operand A. We use the subscript letters T, B, L, and R for Top, Bottom, Left, and Right, respectively.

The inheritance of properties plays an important role in subsequent stages of the algorithm generation process. Thus, when the operands have a special structure, it is beneficial to choose partitioning rules that respect the structure. For a symmetric matrix, for instance, it is convenient to create sub-matrices that exhibit the same property. The 1×2 and 2×1 rules break the structure of a symmetric matrix, as neither of the two sub-matrices inherit the symmetry. Therefore, we only allow 1×1 or 2×2 partitionings, with the extra constraint that the TL quadrant has to be square.

Box 4 illustrates the admissible partitionings for symmetric matrices. On the left, the identity rule is applied and the operand remains unchanged. On the right instead, a constrained 2×2 rule is applied, so that some of the resulting quadrants inherit properties. Both M_{TL} and M_{BR} are square and symmetric, and $M_{BL} = M_{TR}^T$ (or vice versa $M_{TR} = M_{BL}^T$). Each matrix type allows specific partitioning rules and inheritance of properties; for triangular, diagonal, symmetric, and SPD matrices a library of admissible partitioning rules is incorporated into Cl1ck.

3.2 Theorem-Aware Inheritance

Although frequent, direct inheritance of properties is only the simplest form of inheritance. Here we expose a more complex situation. Let A be an SPD matrix. Because of symmetry, the only admissible partitioning rules are the ones listed

$$M_{m\times m} \rightarrow \left(M\right) \quad \text{where } M \text{ is } m \times m \qquad \text{or} \qquad M_{m\times m} \rightarrow \left(\begin{array}{c|c} M_{TL} & M_{BL}^T \\ \hline M_{BL} & M_{BR} \end{array}\right) \quad \text{where } M_{TL} \text{ is } k \times k$$

Box 4. Partitioning rules for structured matrices

in Box 4; applying the 2×2 rule, we obtain

$$A_{m\times m} \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{BL}^T \\ \hline A_{BL} & A_{BR} \end{array}\right), \quad \text{where } A_{TL} \text{ is } k \times k \tag{1}$$

and both A_{TL} and A_{BR} are symmetric. More properties about the quadrants of A can be stated. For example, it is well known that *if A is SPD, then every principal sub-matrix of A is also SPD.* As a consequence, the quadrants A_{TL} and A_{BR} inherit the SPD property. Moreover, it can be proved that given a 2×2 partitioning of an SPD matrix as in (1), the following matrices (known as Schur complements) are also symmetric positive definite:

i) $A_{TL} - A_{BL}^T A_{BR}^{-1} A_{BL}$,
ii) $A_{BR} - A_{BL} A_{TL}^{-1} A_{BL}^T$.

The knowledge emerging from this theorem is hard-coded into Cl1ck. In Sect. 4 it will become apparent how this information is essential for the generation of PMEs.

3.3 Combining the Partitionings

The admissible rules are now applied to rewrite the postcondition. Since in general each operand can be decomposed in multiple ways, not one, but many partitioned postconditions are created. As an example, in the Cholesky factorization (Box 2) both the 1×1 and 2×2 rules are viable for both L and A, leading to four different rewrite sets (Tab. 1).

It is apparent that some of the expressions in the fourth column of Tab. 1 are not algebraically well defined. The rules in the second and third rows lead to ill-defined partitioned postconditions, thus they should be discarded. Despite leading to a well defined expression, the first row of the table should be discarded too, as the goal is a *Partitioned* Matrix Expression and it leads to an expression in which none of the operands has been partitioned. In light of these additional restrictions, the only viable set of rules for the Cholesky factorization is the one given in the last row of Tab. 1.

In summary, partitioning rules must satisfy both the constraints due to the nature of the individual operands, and those due to the operators appearing in the postcondition. In the next section we detail the algorithm used by Cl1ck to generate only the viable sets of partitioning rules.

Table 1. Application of the different combinations of partitioning rules to the postcondition

#	**L**	**A**	**Partitioned Postcondition**
1	$L \rightarrow (L)$	$A \rightarrow (A)$	$(L)\,(L)^T = (A)$
2	$L \rightarrow (L)$	$A \rightarrow \left(\begin{array}{c\|c} A_{TL} & A_{BL}^T \\ \hline A_{BL} & A_{BR} \end{array}\right)$	$(L)\,(L)^T = \left(\begin{array}{c\|c} A_{TL} & A_{BL}^T \\ \hline A_{BL} & A_{BR} \end{array}\right)$
3	$L \rightarrow \left(\begin{array}{c\|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$	$A \rightarrow (A)$	$\left(\begin{array}{c\|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \left(\begin{array}{c\|c} L_{TL}^T & L_{BL}^T \\ \hline 0 & L_{BR}^T \end{array}\right) = (A)$
4	$L \rightarrow \left(\begin{array}{c\|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$	$A \rightarrow \left(\begin{array}{c\|c} A_{TL} & A_{BL}^T \\ \hline A_{BL} & A_{BR} \end{array}\right)$	$\left(\begin{array}{c\|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right) \left(\begin{array}{c\|c} L_{TL}^T & L_{BL}^T \\ \hline 0 & L_{BR}^T \end{array}\right) = \left(\begin{array}{c\|c} A_{TL} & A_{BL}^T \\ \hline A_{BL} & A_{BR} \end{array}\right)$

3.4 Automation

We show how CL1CK performs the partitioning process automatically. The naive approach would be to exhaustively search among all the rules applied to all the operands, leading to a search space of exponential size in the number of operands. Instead, CL1CK utilizes an algorithm that traverses once the tree that represents the postcondition in prefix notation and yields only the viable sets of partitioning rules. The input to the algorithm is the predicates P_{pre} and P_{post} for a target operation. As an example we look at the triangular Sylvester equation $LX + XU = C$, defined using our formalism as in Box 5.

$$X = \Omega(L, U, C) \equiv \begin{cases} P_{\text{pre}} : \{\texttt{Known}(L) \wedge \texttt{LowerTriangular}(L) \wedge \\ \qquad\quad \texttt{Known}(U) \wedge \texttt{UpperTriangular}(U) \wedge \\ \qquad\quad \texttt{Known}(C) \wedge \texttt{Unknown}(X)\} \\ \\ P_{\text{post}} : \{LX + XU = C\}. \end{cases}$$

Box 5. Formal description for the triangular Sylvester equation

First, the algorithm transforms the postcondition to prefix notation (Fig. 3) and collects the name and the dimensionality of each operand. A list of disjoint sets, one per dimension of the operands is then created. This initial list for the Sylvester equation is $[\ \{L_r\}, \{L_c\}, \{U_r\}, \{U_c\}, \{C_r\}, \{C_c\}, \{X_r\}, \{X_c\}\ ]$, where r and c stand for *rows* and *columns* respectively. The algorithm traverses the tree, in a post-order fashion, to determine if and which dimensions are bound together. Two dimensions are bound to one another if the partitioning of one implies the partitioning of the other. If two dimensions are found to be bound, then their

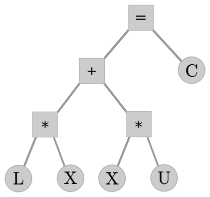


Fig. 3. Tree representation of the equation $LX + XU = C$

corresponding sets are merged together. As the algorithm moves from the leaves to the root of the tree, it keeps track of the dimensions of the operands' subtrees.

The algorithm starts by visiting the node corresponding to the operand L. There it establishes that, since L is lower triangular, the identity and the 2×2 partitioning rules are the only admissible ones. Thus, the rows and the columns of L are bound together, and the list becomes $[\ \{L_r, L_c\}, \{U_r\}, \{U_c\}, \{C_r\}, \{C_c\}, \{X_r\}, \{X_c\}\ ]$. The next node to be visited is that of the operand X. Since X has no specific structure, its analysis causes no bindings. Then, the node corresponding to the $*$ operator is analyzed. The dimensions of L and X have to agree according to the matrix product, therefore, a binding between L_c and X_r is imposed: $[\ \{L_r, L_c, X_r\}, \{U_r\}, \{U_c\}, \{C_r\}, \{C_c\}, \{X_c\}\ ]$. At this stage the dimensions of the product LX are also determined to be $L_r \times X_c$.

The procedure continues by analyzing the subtree corresponding to the product XU. Again, the lack of a specific structure in X does not cause any binding and the algorithm follows with the study of the node for the operand U. The triangularity of U imposes a binding between U_r and U_c leading to $[\ \{L_r, L_c, X_r\}, \{U_r, U_c\}, \{C_r\}, \{C_c\}, \{X_c\}\ ]$. Then, the node for the $*$ operator is analyzed, and a binding between X_c and U_r is found: $[\ \{L_r, L_c, X_r\}, \{U_r, U_c, X_c\}, \{C_r\}, \{C_c\}\ ]$. The dimensions of the product XU are determined to be $X_r \times U_c$.

The next node to be considered is the corresponding to the $+$ operator. It imposes a binding between the rows and the columns of the products LX and XU, i.e., between L_r and X_r, and between X_c and U_c. Since each of these pairs of dimensions already belong to the same set, no modifications are made to the list. The algorithm establishes that the dimensions of the $+$ node are $L_r \times U_c$. Next, the node associated to the operand C is analyzed. Since C has no particular structure, its analysis does not cause any modification. The last node to be processed is the equality operator $=$. This node binds the rows of C to those of L (C_r, L_r) and the columns of C to those of U (C_c, U_c). The final list consists of two separate groups of dimensions:

$$[\ \{L_r, L_c, X_r, C_r\}, \{U_r, U_c, X_c, C_c\}\ ].$$

Having created g groups of bound dimensions, the algorithm generates 2^g combinations of rules (the dimensions within each group being either partitioned or not), resulting in a family of partioned postconditions, one per combination. In practice, since the combination including solely identity rules does not lead to a PME, only $2^g - 1$ combinations are acceptable. In our example the algorithm

Table 2. Viable combinations of partitioning rules for the Sylvester equation

#	**L**	**U**	**C**	**X**
1	(L)	$\left(\begin{array}{c\|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right)$	$\left(\begin{array}{c\|c} C_L & C_R \end{array}\right)$	$\left(\begin{array}{c\|c} X_L & X_R \end{array}\right)$
2	$\left(\begin{array}{c\|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$	(U)	$\left(\begin{array}{c} C_T \\ \hline C_B \end{array}\right)$	$\left(\begin{array}{c} X_T \\ \hline X_B \end{array}\right)$
3	$\left(\begin{array}{c\|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)$	$\left(\begin{array}{c\|c} U_{TL} & U_{TR} \\ \hline 0 & U_{BR} \end{array}\right)$	$\left(\begin{array}{c\|c} C_{TL} & C_{TR} \\ \hline C_{BL} & C_{BR} \end{array}\right)$	$\left(\begin{array}{c\|c} X_{TL} & X_{TR} \\ \hline X_{BL} & X_{BR} \end{array}\right)$

found two groups of bound dimensions, therefore three possible combinations of rules are generated: 1) only the dimensions in the second group are partitioned, 2) only the dimensions in the first group are partitioned, or 3) all dimensions are partitioned. The resulting partitionings are listed in Tab. 2.

This very same process is used to find the bound dimensions of every target operation and, accordingly, only each and every viable combination of partitioning rules is generated.

4 Matrix Arithmetic and Pattern Matching

This section covers the second and third steps in the PME generation stage (Fig. 2). Within the *Matrix Arithmetic* step, symbolic arithmetic is performed and the = operator is distributed over the partitions, originating multiple equations. In (2) we display the result of these actions for the Cholesky factorization, where the symbol $\star$ means that the equation in the top-right quadrant is the transpose of the bottom-left one.

$$\left(\begin{array}{c|c} L_{TL} & 0 \\ \hline L_{BL} & L_{BR} \end{array}\right)\left(\begin{array}{c|c} L_{TL}^T & L_{BL}^T \\ \hline 0 & L_{BR}^T \end{array}\right) = \left(\begin{array}{c|c} A_{TL} & A_{BL}^T \\ \hline A_{BL} & A_{BR} \end{array}\right) \Rightarrow \left(\begin{array}{c|c} L_{TL}L_{TL}^T = A_{TL} & \star \\ \hline L_{BL}L_{TL}^T = A_{BL} & L_{BL}L_{BL}^T + L_{BR}L_{BR}^T = A_{BR} \end{array}\right)$$

The *Pattern Matching* step delivers the sought-after PME. Success of this process is dependent on the ability to identify expressions with known structure and properties. In order to facilitate pattern matching, we force equations to be in their *canonical form*. We state that an equation is in canonical form if a) its left-hand side only consists of those terms that contain at least one unknown object, and b) its right-hand side only consists of those terms that solely contain known objects.

This last step carries out an iterative process comprising three separate actions: 1) structural pattern matching: equations are matched against known patterns; 2) once a known pattern is matched, the unknown operands are flagged as known and the equation becomes a tautology; 3) algebraic manipulation: the remaining equations are rearranged in canonical form. We clarify the iterative process by illustrating, action by action, how Cl1ck works through the Cholesky factorization. The first iteration is depicted in Box 6, in which the top-left formula displays the initial state. In all the next expressions, green and red are used to highlight the known and unknown operands, respectively.

Structural pattern matching: All the equations in Box 6(a) are in canonical form. Through pattern matching, the top-left quadrant is the only one for which a match is found. Cl1ck identifies the equation as a Cholesky factorization (Box 6(b)), since the pattern in Box 2 is satisfied: the system recognizes that i) L_{TL} is lower triangular; ii) A_{TL} is SPD; and iii) the structure of the equation matches the one in the postcondition ($LL^T = A$).

Exposing new available operands: Having matched the top-left equation, Cl1ck turns the unknown operand L_{TL} into L_{TL}, and propagates the information to all the other quadrants (Box 6(c)). As a result, the top-left equation becomes a tautology.

Algebraic manipulation: All the remaining equations are still in canonical form, thus no operation takes place (Box 6(d)).

$$\left(\begin{array}{c|c} L_{TL}L_{TL}^T = A_{TL} & \star \\ \hline L_{BL}L_{TL}^T = A_{BL} & L_{BL}L_{BL}^T + L_{BR}L_{BR}^T = A_{BR} \end{array}\right)$$

(a) Initial state.

$$\left(\begin{array}{c|c} \boxed{L_{TL}} = \Gamma(A_{TL}) & \star \\ \hline L_{BL}L_{TL}^T = A_{BL} & L_{BL}L_{BL}^T + L_{BR}L_{BR}^T = A_{BR} \end{array}\right)$$

(b) Top-left equation is identified as a Cholesky sub-problem.

$$\left(\begin{array}{c|c} \boxed{L_{TL}} = \Gamma(A_{TL}) & \star \\ \hline L_{BL}\boxed{L_{TL}^T} = A_{BL} & L_{BL}L_{BL}^T + L_{BR}L_{BR}^T = A_{BR} \end{array}\right)$$

(c) L_{TL} becomes a known operand for the rest of equations.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline L_{BL}L_{TL}^T = A_{BL} & L_{BL}L_{BL}^T + L_{BR}L_{BR}^T = A_{BR} \end{array}\right)$$

(d) There is no need for algebraic manipulation.

Box 6. First iteration towards the PME generation

In this first iteration, one unknown operand, L_{TL}, has become known, and one equation has turned into a tautology. The knowledge encoded in such a tautology is of importance for a subsequent iteration. The **second iteration** is shown in Box 7.

Structural pattern matching: Box 7(a) reproduces the final state from the previous iteration. Among the two outstanding equations, the bottom-left one is identified (Box 7(b)), as it matches the pattern of a triangular system of equations with multiple right-hand sides (trsm). The pattern for a trsm is

$$\{XL^T = B \wedge \texttt{Output}(X) \wedge \texttt{Input}(L) \wedge \texttt{LowerTriangular}(L) \wedge \texttt{Input}(B)\}.$$

For the sake of brevity, we assume that Cl1ck had learned such pattern from a previous derivation; in practice, in case the system does not know the pattern, a nested task of PME generation would be initiated, yielding the required pattern.

Exposing new available operands: Once the TRSM is identified, the output operand L_{BL} becomes available and turns to green in the bottom-right quadrant (Box 7(c)).

Algebraic manipulation: The bottom-right equation is not in canonical form anymore: the product $L_{BL}L_{BL}^T$, now a known quantity, does not lay in the right-hand side. A simple manipulation brings the equation back to canonical form (Box 7(d)).

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline L_{BL}L_{TL}^T = A_{BL} & L_{BL}L_{BL}^T + L_{BR}L_{BR}^T = A_{BR} \end{array}\right)$$

(a) Initial state.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline \boxed{L_{BL}} = A_{BL}L_{TL}^{-T} & L_{BL}L_{BL}^T + L_{BR}L_{BR}^T = A_{BR} \end{array}\right)$$

(b) Bottom-left equation is identified as a triangular system of equations.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline \boxed{L_{BL}} = A_{BL}L_{TL}^{-T} & \boxed{L_{BL}}\boxed{L_{BL}^T} + L_{BR}L_{BR}^T = A_{BR} \end{array}\right)$$

(c) L_{BL} becomes a known operand.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline L_{BL} = A_{BL}L_{TL}^{-T} & L_{BR}L_{BR}^T = A_{BR} - L_{BL}L_{BL}^T \end{array}\right)$$

(d) State after the algebraic manipulation.

Box 7. Second iteration towards the PME generation

The process continues until all the equations are turned into tautologies. The third and **final iteration** for the Cholesky factorization is shown in Box 8, where the top formula replicates the final state from the previous iteration.

Structural pattern matching: Only one equation, the bottom-right one, remains unprocessed. At a first glance, one might recognize a Cholesky factorization, but the corresponding pattern in Box 2 requires A to be SPD. The question is whether the expression $A_{BR} - L_{BL}L_{BL}^T$ represents an SPD matrix. In order to answer the question, CL1CK applies rewrite rules and symbolic simplifications.

In Sect. 3.2 we explained that the following quantities are known to be SPD: A_{TL}, A_{BR}, $A_{TL} - A_{BL}^T A_{BR}^{-1} A_{BL}$, and $A_{BR} - A_{BL} A_{TL}^{-1} A_{BL}^T$. In order to determine whether $A_{BR} - L_{BL}L_{BL}^T$ is equivalent to any of these expressions, CL1CK makes use of the knowledge acquired throughout the previous iterations. Specifically, in the first two iterations it was discovered that $L_{TL}L_{TL}^T = A_{TL}$, and $L_{BL} = A_{BL}L_{TL}^{-T}$. Using these tautologies as rewrite rules, the expression $A_{BR} - L_{BL}L_{BL}^T$ is manipulated. First, the equality $L_{BL} = A_{BL}L_{TL}^{-T}$ is used to replace the instances of L_{BL}, yielding $A_{BR} - A_{BL}L_{TL}^{-T}L_{TL}^{-1}A_{BL}^T$, and equivalently, $A_{BR} - A_{BL}(L_{TL}L_{TL}^T)^{-1}A_{BL}^T$. Then, by virtue of the tautology $L_{TL}L_{TL}^T = A_{TL}$, $L_{TL}L_{TL}^T$ is replaced by A_{TL}, yielding $A_{BR} - A_{BL}A_{TL}^{-1}A_{BL}^T$. Now, this expression is known to be SPD. Thanks to these manipulations, CL1CK successfully associates the bottom right equation with the pattern for a Cholesky factorization.

Exposing new available operands: Once the expression in the bottom-right quadrant is identified, the system exposes the quantity L_{BR} as known. Since no equation is left, the process completes and the PME—formed by the three tautologies—is returned as output.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline L_{BL} = A_{BL}L_{TL}^{-T} & L_{BR}L_{BR}^T = A_{BR} - L_{BL}L_{BL}^T \end{array}\right)$$

(a) Initial state.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline L_{BL} = A_{BL}L_{TL}^{-T} & \boxed{L_{BR}} = \Gamma(A_{BR} - L_{BL}L_{BL}^T) \end{array}\right)$$

(b) Bottom-right equation is identified as a Cholesky factorization.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline L_{BL} = A_{BL}L_{TL}^{-T} & \boxed{L_{BR}} = \Gamma(A_{BR} - L_{BL}L_{BL}^T) \end{array}\right)$$

(c) L_{BR} becomes a known operand.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline L_{BL} = A_{BL}L_{TL}^{-T} & L_{BR} = \Gamma(A_{BR} - L_{BL}L_{BL}^T) \end{array}\right)$$

(d) Final PME.

Box 8. Final iteration towards the PME generation

By means of the described process, PMEs for a target equation are automatically generated. The PME for the Cholesky factorization is given in Box 9.

$$\left(\begin{array}{c|c} L_{TL} = \Gamma(A_{TL}) & \star \\ \hline L_{BL} = A_{BL}L_{TL}^{-T} & L_{BR} = \Gamma(A_{BR} - L_{BL}L_{BL}^T) \end{array}\right)$$

Box 9. Partitioned Matrix Expression for the Cholesky factorization

5 Conclusions

The work we presented sets the ground for the development of a symbolic system that, from the sole description of an operation, generates algorithms automatically. The core of our methodology stands in the PME. A PME encapsulates the information about the target operation in a way that facilitates the subsequent identification of loop-invariants. The loop-invariants then lead to the final algorithms through a technique based on program correctness. In this paper we introduce a symbolic system, Cl1ck, that automates the generation of PMEs.

In order to generate PMEs, Cl1ck first identifies how the operands in the operation may be partitioned. Instead of a brute force approach of exponential complexity, Cl1ck utilizes a tree-based algorithm that yields only the viable sets of partitioning rules. Through a process of pattern matching, each such set leads to a distinct PME. The key in the PME generation is Cl1ck's ability to identify known patterns. Initially, Cl1ck only recognizes elementary structures, but its knowledge expands by automatically learning the patterns associated with the operations it tackles. Thanks to this augmenting internal knowledge, the system may generate PMEs for increasingly complex operations.

To illustrate Cl1ck, we discussed the Cholesky factorization and, partially (due to space constraints), the Sylvester equation. Despite the fact that such operations differ in multiple ways—number and properties of the operands, number of valid sets of partitioning rules, number of PMEs—the steps performed by Cl1ck leading to the PMEs are exactly the same. As future work, we plan to add support for higher dimensional objects and the derivative operator.

Acknowledgements. The authors gratefully acknowledge the support received from the Deutsche Forschungsgemeinschaft (German Research Association) through grant GSC 111.

References

1. FLAME Project: FLAME Online Reference, http://z.cs.utexas.edu/wiki/flame.wiki/
2. Dongarra, J.J., Du Croz, J., Hammarling, S., Duff, I.S.: A set of level 3 basic linear algebra subprograms. ACM Trans. Math. Softw. 16, 1–17 (1990)
3. Jonsson, I., Kågström, B.: Recursive blocked algorithms for solving triangular systems—part i: one-sided and coupled sylvester-type matrix equations. ACM Transactions on Mathematical Software 28(4), 392–415 (2002)
4. Jonsson, I., Kågström, B.: Recursive blocked algorithms for solving triangular systems—part ii: Two-sided and generalized sylvester and lyapunov matrix equations. ACM Transactions on Mathematical Software 28(4), 416–435 (2002)
5. Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide, 3rd edn. Society for Industrial and Applied Mathematics, Philadelphia (1999)
6. Whaley, R.C., Dongarra, J.: Automatically tuned linear algebra software. In: SuperComputing 1998: High Performance Networking and Computing (1998)
7. Frigo, M., Johnson, S.G.: The design and implementation of FFTW3. Proceedings of the IEEE 93(2), 216–231 (2005); Special issue on "Program Generation, Optimization, and Platform Adaptation"
8. Püschel, M., Moura, J.M.F., Johnson, J., Padua, D., Veloso, M., Singer, B., Xiong, J., Franchetti, F., Gacic, A., Voronenko, Y., Chen, K., Johnson, R.W., Rizzolo, N.: SPIRAL: Code generation for DSP transforms. Proceedings of the IEEE 93(2), 232–275 (2005); special issue on "Program Generation, Optimization, and Adaptation"
9. Bientinesi, P., Gunnels, J.A., Myers, M.E., Quintana-Ortí, E.S., van de Geijn, R.A.: The science of deriving dense linear algebra algorithms. ACM Transactions on Mathematical Software 31(1), 1–26 (2005)
10. Bientinesi, P., Quintana-Ortí, E.S., van de Geijn, R.: FLAME lab: A farewell to indices. FLAME Working Note #11. Technical Report TR-2003-11, The University of Texas at Austin, Department of Computer Sciences (April 2003)
11. Quintana-Ortí, E.S., van de Geijn, R.A.: Formal derivation of algorithms: The triangular Sylvester equation. ACM Transactions on Mathematical Software 29(2), 218–243 (2003)

12. Poulson, J., van de Geijn, R., Bennighof, J.: Parallel algorithms for reducing the generalized hermitian-definite eigenvalue problem. FLAME Working Note #56. Technical Report TR-11-05, The University of Texas at Austin, Department of Computer Sciences (February 2011)
13. Bientinesi, P., van de Geijn, R.: A goal-oriented and modular approach to stability analysis. SIAM Journal on Matrix Analysis and Applications (to appear, 2011)
14. Wolfram Research: Mathematica Reference Guide, `http://reference.wolfram.com/mathematica/`
15. Gries, D., Schneider, F.B.: A Logical Approach to Discrete Math. Texts and Monographs in Computer Science. Springer, Heidelberg (1992)

Involutive Division Generated by an Antigraded Monomial Ordering

Vladimir P. Gerdt[1] and Yuri A. Blinkov[2]

[1] Laboratory of Information Technologies, Joint Institute for Nuclear Research, 141980 Dubna, Russia
gerdt@.jinr.ru

[2] Department of Mathematics and Mechanics, Saratov State University 410012 Saratov, Russia
BlinkovUA@info.sgu.ru

Abstract. In the present paper we consider a class of involutive monomial divisions pairwise constructed by the partition of variables into multiplicative and nonmultiplicative generated by a total monomial ordering. If this ordering is admissible or the inverse of an admissible ordering, then the involutive division generated possesses all algorithmically important properties such as continuity, constructivity, and noetherianity. Among all such divisions, we single out those generated by antigraded monomial orderings. We demonstrate, by example of the antigraded lexicographic ordering, that the divisions of this class are heuristically better than the classical Janet division. The last division is pairwise generated by the pure lexicographic ordering and up to now has been considered as computationally best.

1 Introduction

The notion of involutive monomial division introduced first in our paper [1] and then somewhat modified by Apel [2] is a cornerstone of the theory of involutive bases and their algorithmic construction. The basic idea behind this notion goes back to Janet [3] and consists in a proper partition of the variables for every element in a finite monomial set into the two subsets called multiplicative and nonmultiplicative. Given a polynomial set and an admissible monomial order, the partition of variables is defined in terms of the leading monomial set.

Each partition generates a restricted monomial division [4] called involutive if it is defined for an arbitrary monomial set and satisfies certain axioms [1], or if it is admissible for the given monomial set in accordance with [2].

In an involutive algorithm, the nonmultiplicative variables of a polynomial are used for its prolongation, that is, for the multiplication by these variables. The multiplicative variables of other polynomials in the set are used for reduction of the prolonged polynomial. An involutive basis is a polynomial set such that all its nonmultiplicative prolongations are multiplicatively reducible to zero.

In Apel's approach [2], an involutive division is algorithmically constructed for a polynomial set, and it is checked whether the set is involutive. If this is not the case the set is enlarged with an irreducible nonmultiplicative prolongation, and the process is iterated. Though such an approach allows to construct an optimal involutive division at every iteration, the completion algorithm is much less efficient than ours [4]. This is because constructing and optimizing an involutive division all over again at every iteration step is computationally costly, and one cannot properly use the "history" of completion. In our case, if the underlying division is good enough, e.g., Janet or Pommaret division [1], the partition of variables is changed rather smoothly from iteration to iteration, and one can effectively accelerate the computational process by using data of the iterations which have been performed.

If an involutive algorithm terminates it outputs an involutive basis which is a Gröbner basis of special structure determined by properties of underlying involutive division. In our approach, a reduced Gröbner basis is always a well defined subset of the involutive basis and can be extracted from the latter without any extra computation [4]. In so doing, just this Gröbner redundancy makes structural and combinatorial information on the ideals and modules more accessible via involutive bases than via reduced Gröbner ones. Thus, involutive bases based on Pommaret division turned out to be a useful tool in commutative algebra and formal theory of differential equations as was discovered by Seiler [5].

Our implementation of the Janet division algorithm [4] in *GINV* [6] [1], revealed its computational superiority over the fastest implementations of Buchberger's algorithm, e.g. in *Singular* [8], for most of benchmarks (see the Web page [9]) taken from [10] and other sources.

Apart from the three classical divisions generated by the partitions of variables have been used for completion of partial differential equations to involution and called in [1] after Thomas, Janet, and Pommaret, many other involutive divisions were found [11,12,13,14,15]. All of them are defined for arbitrary monomial sets [1] and, except a few special examples constructed by Semenov [13], possess such algorithmically important properties as continuity and constructivity [1].

However, in spite of intensive theoretical research and numerous computer experiments with different new divisions, none of them could compete algorithmically with Janet division. The last is specified by a permutation on the variables, and for n variables there are $n!$ different Janet divisions. A very few examples are known (cf. [15]) when a minimal Janet monomial basis is larger, for all possible permutations on the variables, than that for another division.

In the given paper we consider involutive divisions which are pairwise generated by total monomial orderings and single out a subclass of such divisions from the class of $\prec$ −divisions constructed by Semenov [13] and studied in [15]. In the cited papers symbol $\prec$ denotes an admissible monomial ordering. Our subclass contains divisions generated by antigraded monomial orderings. They are $\prec$ −divisions where $\prec$ is the inverse of an antigraded order.

We present results of computational experiments with one of such divisions generated by the antigraded lexicographic ordering $\succ_{\mathrm{alex}}$ (cf. [16]) and called

[1] GINV is now under development in collaboration with Technical University (RWTH) Aachen [7].

$\succ_{\rm alex}$-division. Our experimentation shows that in the vast majority of cases $\succ_{\rm alex}$-monomial bases are more compact than Janet bases. In addition, cardinality of the $\succ_{\rm alex}$-bases under permutations on the variables varies substantially less than that for Janet bases.

These properties of the $\succ_{\rm alex}$-division open new prospects in speeding-up computation of involutive bases and reduced Gröbner bases by the involutive algorithms.

2 Preliminaries

Hereafter we use the following notations combined, for compactness, in tabular form

X	set of variables $X = \{x_1, \ldots, x_n\}$
$\mathcal{K}$	coefficient field
$\mathcal{K}[X]$	commutative polynomial ring over $\mathcal{K}$ in X
$\mathcal{M}$	monoid of all power products in $\mathcal{K}[X]$
U, V, W	finite subsets of $\mathcal{M}$ or lists of elements in $\mathcal{M}$
u, v, w	elements in $\mathcal{M}$
$\lvert U\rvert$	cardinality of U
$\deg(u)$	total degree of u
$\deg_i(u)$	degree of u in the variable x_i
$\mathrm{lcm}(u, v)$	least common multiple of u and v
F, G, Q	finite subsets of $\mathcal{K}[X]$ or lists of elements in $\mathcal{K}[X]$
f, g, h	elements in $\mathcal{K}[X]$
$\langle F\rangle$	ideal in $\mathcal{K}[X]$ generated by F
$\sqsupset$	total monomial ordering
$\mathrm{lm}(f)$	leading monomial of f
$\mathrm{lm}(F)$	set or list of leading monomials in F
$\succ$	admissible monomial ordering
$\succ_{\rm lex}$	lexicographic monomial ordering
$\succ_{\rm alex}$	antigraded lexicographic monomial ordering
$\succ_{\rm grlex}$	graded lexicographic monomial ordering
$\mathcal{L}$	involutive monomial division
$\mathcal{J}$	Janet division
$\mathcal{P}$	Pommaret division
$\mathcal{L}(u, U)$	monoid of $\mathcal{L}$-multiplicative variables for $u \in U$
$NM_{\mathcal{L}}(u, U)$	set of $\mathcal{L}$-nonmultiplicative variables for $u \in U$
$NF_{\mathcal{L}}(f, F)$	$\mathcal{L}$-normal form of f modulo F
$\mathcal{C}_{\mathcal{L}}(U)$	$\mathcal{L}$-cone of U
$u \mid v$	divisibility relation on $\mathcal{M}$ ('u divides v')
$u \mid_{\mathcal{L}} v$	$\mathcal{L}$-divisibility relation on $U \times \mathcal{M}$ ('$u \in U$ $\mathcal{L}$-divides $v \in \mathcal{M}$')
$\bar{U}$	involutive completion of U

We shall consider a commutative multivariate polynomial ring $\mathcal{K}[X]$ over a field $\mathcal{K}$ in variables $X = \{x_1, \ldots, x_n\}$. Involutive divisions are defined on the monoid

$\mathcal{M}$ of monomials, i.e., power products, of $\mathcal{K}[X]$. For generality, in the input monomial or polynomial data for the algorithms of Section 4, we allow repetition of elements. In this case, we speak of lists, that is sequences of elements rather than of sets (cf. [17]).

Now we give the definitions of involutive division and involutive bases and refer a reader to [1] for more definitions and proofs (see also [4] and book [5]).

Definition 1. *An* involutive division $\mathcal{L}$ *is defined on* $\mathcal{M}$ *if for any nonempty set* $U \subset \mathcal{M}$ *and for any* $u \in U$ *a subset* $M_{\mathcal{L}}(u,U) \subseteq X$ *is defined that generates submonoid* $\mathcal{L}(u,U) \subset \mathcal{M}$ *of power products in* $M_{\mathcal{L}}(u,U)$ *and the following holds*

1. $v \in U \ \wedge \ u\mathcal{L}(u,U) \cap v\mathcal{L}(v,U) \neq \emptyset \Longrightarrow u \in v\mathcal{L}(v,U) \ \vee \ v \in u\mathcal{L}(u,U)\,,$
2. $v \in U \ \wedge \ v \in u\mathcal{L}(u,U) \Longrightarrow \mathcal{L}(v,U) \subseteq \mathcal{L}(u,U)$ *(transitivity),*
3. $u \in V \wedge \ V \subseteq U \Longrightarrow \mathcal{L}(u,U) \subseteq \mathcal{L}(u,V)$ *(filter axiom).*

Variables in $M_{\mathcal{L}}(u,U)$ *are* $\mathcal{L}$-multiplicative *for* u *and those in* $NM_{\mathcal{L}}(u,U) = X \setminus M_{\mathcal{L}}(u,U)$ *are* $\mathcal{L}$-nonmultiplicative. *If* $w \in u\mathcal{L}(u,U)$*, then* u *is* $\mathcal{L}$-(involutive) divisor *of* w *(denotation:* $u \mid_{\mathcal{L}} w$*).*

In the literature (see, for example, [13]–[15]) the first two axiomatic properties of involutive division in Definition 1 are often called *static* since they apply to a certain monomial set whereas the last property is called *filter axiom* and deals with behavior of involutive division under enlargement of the set. In Apel's approach [2], the filter axiom is not used. Instead, the underlying admissibility concept for a partition of the variables, that for the fixed monomial set is equivalent to the axioms 1-2, forces to perform an appropriate repartition of the variables to provide its admissibility for the enlarged set.

Definition 2. (Janet division) *Given a permutation* $\sigma \in S_n$*, for each* $1 \le i \le n$ *divide* $U \in \mathcal{M}$ *into groups labeled by non-negative integers* $d_1,\dots,d_i$

$$[d_1,\dots,d_i] = \{\ u \ \in U \mid d_j = \deg_{\sigma(j)}(u),\ 1 \le j \le i\ \}.$$

Variable $x_{\sigma(1)}$ *is multiplicative for* $u \in U$ *if* $\deg_{\sigma(1)}(u) = \max\{\deg_{\sigma(1)}(v) \mid v \in U\}$*. For* $i > 1$ $x_{\sigma(i)}$ *is multiplicative for* $u \in [d_1,\dots,d_{i-1}]$ *when*

$$\deg_{\sigma(i)}(u) = \max\{\deg_{\sigma(i)}(v) \mid v \in [d_1,\dots,d_{i-1}]\}.$$

Definition 3. (Pommaret division) *Given a permutation* $\sigma \in S_n$*, for the monomial* $v = x_{\sigma(1)}^{d_1}\cdots x_{\sigma(k-1)}^{d_{k-1}}x_{\sigma(k)}^{d_k}$ *with* $d_k > 0$ *variables* $x_{\sigma(j)}, j \ge k$ *are multiplicative. For* $v = 1$ *all the variables are multiplicative.*

Remark 1. Though there are $n!$ different Janet divisions and the same number of Pommaret divisions, in our previous papers we always considered Janet and Pommaret partitions of the variables for the identical permutation whereas Janet himself [3] (see also Seiler [5]) partitioned the variables in accordance with the reverse permutation

$$\sigma = \begin{pmatrix} 1 & 2 & \dots & n \\ n & n-1 & \dots & 1 \end{pmatrix}.$$

Definition 4. *The set* $\mathcal{C}(U) = \cup_{u\in U}\, u\,\mathcal{M} \subset \mathcal{M}$ *is the* cone *generated by a set* $U \subset \mathcal{M}$*, and the set* $\mathcal{C}_{\mathcal{L}}(U) = \cup_{u\in U}\, u\,\mathcal{L}(u,U) \subset \mathcal{M}$ *is the* $\mathcal{L}-$(involutive) cone *of* U*. A finite set* $\bar{U} \supseteq U \subset \mathcal{M}$ *is the* $\mathcal{L}-$completion *of* U *if* $\mathcal{C}(U) = \mathcal{C}_{\mathcal{L}}(\bar{U})$*. Under the equality*

$$\mathcal{C}_{\mathcal{L}}(U) = \mathcal{C}(U) \tag{1}$$

set U *is* $\mathcal{L}$-complete *or* $\mathcal{L}$-involutive*. If every finite set* U *admits* $\mathcal{L}-$*completion, then the involutive division* $\mathcal{L}$ *is* noetherian*.*

Remark 2. Given an *admissible* monomial ordering $\succ$, that is, the ordering such that

$$\begin{cases} (\ \forall u \in \mathcal{M}\setminus\{1\}\ )\ \ [\ u \succ 1\ ]\,, \\ (\ \forall u,v,w \in \mathcal{M}\ )\ \ [\ u \succ v \Longleftrightarrow u\cdot w \succ v\cdot w\ ]\,, \end{cases}$$

$\mathcal{L}$-*normal form* of a polynomial $g \in \mathcal{K}[X]$ modulo a finite polynomial set $F \subset \mathcal{K}[X]$ and $\mathcal{L}$-*autoreduction* of F are defined exactly as in Gröbner bases theory [18] with the only distinction: every $f \in F$ is allowed to be multiplied by the power products from $\mathcal{L}(\mathrm{lm}(f), \mathrm{lm}(F))$ where $\mathrm{lm}(f)$ and $\mathrm{lm}(F)$ denote, respectively, the leading monomial of f and the set of the leading monomials in F.

Definition 5. *Given an involutive division* $\mathcal{L}$ *and an admissible ordering* $\succ$ *a finite* $\mathcal{L}$*-autoreduced subset* $G \subset \mathcal{K}[X]$ *is an* $\mathcal{L}-$(involutive) basis *of ideal* $\mathcal{I} = \langle G\rangle$ *if*

$$(\forall f \in \mathcal{I})\ (\exists\, g \in G)\ [\ \mathrm{lm}(g)\ |_{\mathcal{L}}\ \mathrm{lm}(f)\ ]\,. \tag{2}$$

Proposition 1. *If a division* $\mathcal{L}$ *is continuous [1], then conditions (1) and (2) are respectively equivalent to*

$$(\forall u \in U)\ (\forall x \in NM_{\mathcal{L}}(u,U))\ \ [\ u\cdot x \in \mathcal{C}_{\mathcal{L}}(U)\ ]\,. \tag{3}$$

and

$$(\ \forall f \in G\ )\ (\ \forall x_i \in NM_{\mathcal{L}}(\mathrm{lm}(f), \mathrm{lm}(G))\ )\ \ [\ NF_{\mathcal{L}}(x_i\cdot f, G) = 0\ ]\,. \tag{4}$$

$NF_{\mathcal{L}}$ in (4) denotes here the $\mathcal{L}$-normal form.

Definition 6. *A monomial* $\mathcal{L}$*-basis* $\bar{U}$ *of* $\langle U\rangle$ *is* minimal *if for any other* $\mathcal{L}$*-basis* $\bar{U}_1$*, the inclusion* $\bar{U} \subseteq \bar{U}_1$ *holds. A polynomial* $\mathcal{L}$*-basis* G *is* minimal *if* $\mathrm{lm}(G)$ *is a minimal* $\mathcal{L}$*-basis of the initial ideal of* $\langle G\rangle$*.*

Proposition 2. *If a continuous division* $\mathcal{L}$ *is constructive [1], then an ideal in* $\mathcal{K}[X]$ *having a (finite)* $\mathcal{L}$*-basis, has also a minimal* $\mathcal{L}$*-basis. Given an ideal and an ordering, there is a unique monic minimal* $\mathcal{L}$ *basis.*

Remark 3. Since an involutive division is a restricted monomial division, condition (2) implies that an involutive basis is a Gröbner basis, which is generally redundant in the Gröbner sense.

Proposition 3. *The Janet and Pommaret divisions of Definitions 3 and 2 are continuous and constructive. The Janet division is noetherian whereas the Pommaret division is not.*

3 Pair Divisions Generated by Total Monomial Orderings

All involutive divisions known from the literature that satisfy Axioms 1-3 in Definition 1 possess the pair property in accordance with its definition given in [11]:

Definition 7. *An involutive division $\mathcal{L}$ is* pairwise *if for any finite set $U \subset \mathcal{M}$ with cardinality $|U| \geq 2$ the set of its $\mathcal{L}$-nonmultiplicative satisfies*

$$(\ \forall u \in U \ ) \ \ [\ NM_{\mathcal{L}}(u, U) = \bigcup_{v \in U \setminus \{u\}} NM_{\mathcal{L}}(u, \{u, v\}) \ ]. \tag{5}$$

The pair property was further studied by Semenov [13,14].

Formula (5) provides a regular procedure for construction of a pairwise division $\mathcal{L}$. In doing so, it makes sense, from the computational point of view, to take into account the following remark.

Remark 4. The computational cost of checking involutivity of a monomial or a polynomial basis with respect to a continuous division is determined by a number of nonmultiplicative prolongations to be examined in (3) or in (4), respectively. Hence, with respect to the involutivity checking, the quality of an involutive division specified for a monomial set is characterized by the total number of nonmultiplicative prolongations it generates for the set. Proposition 6 of Section 4 makes this fact more explicit. In the next section, we shall see that the above characterization is also relevant to computation of polynomial involutive bases.

Consider now a set $\{u, v\} \subset \mathcal{M}$ of two monomials and partition the variables for these monomials to satisfy the static Axiom 1 in Definition 1. In doing that it is useful to distinguish two alternatives:

1. Monomials u and v are comparable with respect to the partial (i.e., reflexive, antisymmetric and transitive) order induced by the conventional divisibility relation denoted by the vertical bar '$|$'. In other words, either $u \mid v$ or $v \mid u$. In this case Axiom 1 holds true for the set $\{u, v\}$ if all variables in X are specified as $\mathcal{L}$-multiplicative for both monomials.
2. Monomials u and v are incomparable with respect to the partial order induced, i.e., neither $u \mid v$ nor $v \mid u$. In this case, at least one of the variables in X and at least for one of the monomials in $\{u, v\}$ must be $\mathcal{L}-$nonmultiplicative. Otherwise, the least common multiple of monomial u, v denoted by $\mathrm{lcm}(u, v)$ would have both monomials as $\mathcal{L}$-divisors, which violates Axiom 1. It is sufficient and computationally optimal to assign just one variable as nonmultiplicative to one of the monomials in the pair.

A natural, uniform, and rather optimal recipe to make such assignment is to fix a permutation σ of the variables, a total monomial order $\sqsubset$ on $\mathcal{M}$ and to proceed as follows

$$NM_{\mathcal{L}}(u, \{u, v\}) := \begin{cases} \text{if } u \sqsupset v \text{ or } (u \sqsubset v \wedge v \mid u) \text{ then } \emptyset \\ \text{else } \{x_{\sigma(i)}\}, \ i = \min\{j \mid \deg_{\sigma(j)}(u) < \deg_{\sigma(j)}(v)\} \,. \end{cases} \tag{6}$$

Theorem 1. *The pair assignment specified in (6), together with formula (5)*[2] *yields an involutive division.*

Proof. Let a pair $\{u,v\}$ of monomials satisfy $u \sqsubset v$. Then if $u \mid v$, then there are none variables occurring in u have degree higher than those in v, and $NM_{\mathcal{L}}(u,\{u,v\}) = NM_{\mathcal{L}}(v,\{u,v\}) = \emptyset$. Otherwise rule (6) will specify a nonmultiplicative variable for u. If $u \sqsupset v$ and $v \mid u$ then the same reasoning is applicable to v. Therefore, partition (6) of variables satisfies static Axiom 1 in Definition 1 for any subset in an arbitrary monomial set $U \subset \mathcal{M}$ containing a pair of monomials. The union of nonmultiplicative variables in 6 preserves, obviously, the validity of Axiom 1 and satisfies also the filter axiom.

It remains to prove that the transitivity condition (Axiom 2) in Definition 1, which is trivially satisfied for a set of two elements, is also preserved for a bigger set U. Without loss of generality we can omit σ in (5), that is fix it as the identical permutation.

Let a triple $\{u_1,u_2,u_3\} \subset U$ be such that $u_2 \in u_1\mathcal{L}(u_1,U)$ and $u_3 \in u_2\mathcal{L}(u_2,U)$. It is clear that $u_1 \sqsupset u_2 \sqsupset u_3$. Indeed, the condition $u_1 \sqsubset u_2$ and $u_1 \mid u_2$ together with (5) would imply the existence of $x_i \in NM_{\mathcal{L}}(u_1,U)$ such that $\deg_i(u_2) > \deg_i(u_1)$ what contradicts to $\mathcal{L}$-divisibility of u_2 by u_1.

Assume that $u_3 \notin u_1\mathcal{L}(u_1,U)$. In this case $u_3 = u_1 \cdot v \cdot w$ where $v,w \in \mathcal{M}\setminus\{1\}$, $u_2 = u_1 \cdot v$, $u_3 = u_2 \cdot w$, and there is the variable x_i of lowest index i such that $x_i \mid w$ and $x_i \in NM_{\mathcal{L}}(u_1,U)$. From the pairwise construction (5)-(6) it follows

$$(\,\exists u_0 \in U\,)\ [\ u_0 \sqsupset u_1\ \wedge \exists i = \min\{j \mid \deg_j(u_1) < \deg_j(u_0)\}\ ]\,.$$

Since $u_0 \sqsupset u_2 = u_1 \cdot v$ and $x_i \in M_{\mathcal{L}}(u_2,U)$, the monomial v can contain only variables x_k with $k < i$ and such that $\deg_k(u_0) = \deg_k(u_1) + \deg_k(v)$. But the last equality implies that the lowest index variable in v is nonmultiplicative for u_1 what contradicts $u_2 \in u_1\mathcal{L}(u_1,U)$. $\square$

If the ordering $\sqsupset$ in (6) is admissible, then the transitivity property is immediately provided by (6) since in this case any monomial set is obviously $\mathcal{L}$-autoreduced. In the case when $\sqsupset$ is the inverse of an admissible ordering $\succ$ the correctness proof for procedure (5)–(6) was given in [15] where the generated division is called $\prec$-division. The above theorem shows the correctness of (5)-(6) in generating an involutive division for arbitrary total order $\sqsupset$.

The constructed pairwise division is admissible for $(U,\sqsupset)$ in the sense by Apel [2]. It is easy to see (cf. [13]) that the Janet division in Definition 2 is obtained this way when $\sqsupset$ is the pure lexicographical ordering, which we shall denote by $\succ_{\text{lex}}$, and such that

$$x_{\sigma(1)} \succ_{\text{lex}} x_{\sigma(2)} \succ_{\text{lex}} \cdots \succ_{\text{lex}} x_{\sigma(n)}\,.$$

If $\sqsupset$ is an admissible monomial ordering $\succ$, then (5)–(6) generates the $\succ$-division introduced by Semenov and investigated in [13]–[15], together with another division ($\prec$-division) generated by the inverse of $\succ$.

[2] In [13]-[15] the construction of involutive division by formula (5) is called *pairwise closure of involutive 2-partition.*

Now we specialize $\sqsupset$ in (5)–(6) to two different orderings which we shall compare with the Janet division.

Definition 8. [16] *The* graded lexicographic *monomial order* $\succ_{\text{grlex}}$ *is defined as follows:*

$$u \succ_{\text{grlex}} v \Longleftrightarrow \deg(u) > deg(v) \ \vee \ deg(u) = deg(v) \ \wedge \ u \succ_{\text{lex}} v\,. \tag{7}$$

Similarly, the antigraded *monomial order* $\succ_{\text{alex}}$ *is defined as follows:*

$$u \succ_{\text{alex}} v \Longleftrightarrow \deg(u) < deg(v) \ \vee \ deg(u) = deg(v) \ \wedge \ u \succ_{\text{lex}} v\,. \tag{8}$$

Remark 5. Unlike $\succ_{\text{grlex}}$ that is admissible, $\succ_{\text{alex}}$ is not. The last belongs to the family of *local* orderings (see [16,17]).

Definition 9. *The involutive division defined by (5)-(6) for the* $\succ_{\text{grlex}}$*-and* $\succ_{\text{alex}}$*-ordering will be called* $\succ_{\text{grlex}}$*-and* $\succ_{\text{alex}}$-division, *respectively.*

Proposition 4. *The division generated by a total ordering* $\sqsupset$ *is noetherian. If* $\sqsupset$ *is admissible ordering or the inverse of an admissible ordering, then the division it generates is continuous and constructive.*

Proof. To prove noetherianity we consider Thomas division [1] denoted by $\mathcal{T}$ with the partition of variables $x_i \in M_{\mathcal{T}}(u,U) \Longleftrightarrow \deg_i(u) = \max\{\,\deg_i(v) \mid v \in U\,\}$. Given a finite set $U \subset \mathcal{M}$, its $\mathcal{T}$-completion $\bar{U}_{\mathcal{T}}$ is given by

$$\bar{U}_{\mathcal{T}} = \{v \in \langle U\rangle \mid \deg_i(v) \leq \max\{\deg_i(u) \mid u \in U\}, 1 \leq i \leq n\}\,.$$

It is easy to see that the monomial set $\bar{U}_{\mathcal{T}}$ is also involutive for the division generated by an arbitrary total order $\sqsupset$ in accordance to (5)-(6).

The cases of admissible $\sqsupset$ and its inverse were analyzed by Semenov and Zyuzikov [15] who proved continuity and constructivity of the generated divisions. □

Corollary 1. *The* $\succ_{\text{grlex}}$*-and* $\succ_{\text{alex}}$*-divisions are continuous, constructive, and noetherian.*

Proof. The ordering $\succ_{\text{alex}}$ is the inverse of the following admissible monomial ordering which we call graded antilexicographic and denote by $\succ_{\text{galex}}$:

$$u \succ_{\text{galex}} v \Longleftrightarrow \deg(u) > deg(v) \ \vee \ deg(u) = deg(v) \ \wedge \ u \prec_{\text{lex}} v\,. \qquad \square$$

Example 1. The following table shows the partition of variables for the monomial set $U = \{x_1^2x_3, x_1x_2, x_2x_3, x_3^2\}$ into multiplicative (M) and nonmultiplicative (NM) for the identical permutation of indices and for divisions: Janet, Pommaret, $\succ_{\text{grlex}}$ and $\succ_{\text{alex}}$.

Table 1. Partition of variables for $U = \{x_1^2x_3, x_1x_2, x_1x_3, x_3^2\}$

Element in U	Involutive division							
	Janet		Pommaret		$\succ_{\text{grlex}}$		$\succ_{\text{alex}}$	
	M	NM	M	NM	M	NM	M	NM
$x_1^2x_3$	x_1, x_2, x_3	$-$	x_3	x_1, x_2	x_1, x_2, x_3	$-$	x_1	x_2, x_3
x_1x_2	x_2, x_3	x_1	x_2, x_3	x_1	x_2, x_3	x_1	x_1, x_2, x_3	$-$
x_1x_3	x_3	x_1, x_2	x_3	x_1, x_2	x_3	x_1, x_2	x_1, x_3	x_2
x_3^2	x_2, x_3	x_1	x_3	x_1, x_2	x_3	x_1, x_2	x_2, x_3	x_1

¿From Table 1 one can see that the given set U is not $\succ_{\text{alex}}$-autoreduced, since $x_1x_3 \mid_{\succ_{\text{alex}}} x_1^2x_3$. For the other three divisions U is involutively autoreduced. As we already noticed in our comments that follow the proof of Theorem 1, for a division generated pairwise by an admissible ordering, any set of (distinct) monomials is involutively autoreduced. In the example under consideration this concerns Janet and $\succ_{grlex}$-divisions. Though U is also $\mathcal{P}$-(Pommaret) autoreduced, generally it is not always the case $\succ_{grlex}$.

On the other hand, the involutive division that is pairwise generated by an antigraded total ordering $\sqsupset$, that is, such ordering which satisfies [16]

$$\deg(u) < \deg(v) \Longrightarrow u \sqsupset v\,, \tag{9}$$

with respect to autoreduction, is similar to Pommaret division. The following theorem clarifies this point.

Theorem 2. *Let $\sqsupset$ be a total monomial ordering satisfying (9), $\mathcal{L}$ be the involutive division generated by (5)–(6) for some permutation σ and $U \subset \mathcal{M}$ be a finite monomial set or list. Then for the set V generated by $\mathcal{L}-$autoreduction of U, that is, $V =$ **Autoreduce**$_{\mathcal{L}}(U)$, the inclusion holds*

$$V \subseteq \bar{U} \tag{10}$$

where $\bar{U}$ is the minimal $\mathcal{L}-$basis of the monomial ideal $\langle U\rangle$ generated by U.

Proof. We show first that

$$(\,\forall u \in U\,)\;(\,\forall x \in NM_{\mathcal{L}}(u,U)\,)\;\;[\;u \cdot x \in v\mathcal{L}(v,U) \setminus \{v\} \Longrightarrow v \sqsupset u\,] \tag{11}$$

Let $u \cdot x_{\sigma(i)} = v \cdot w$ where $w \in \mathcal{L}(v,U)$ and $\deg(w) \geq 1$. If $\deg(w) \geq 2$, then the implication (11) is a trivial consequence of (9). Consider now the case when $w = x_{\sigma(j)}$. Obviously, $i \neq j$ and, hence,

$$i = \min\{k \mid \deg_{\sigma(k)}(u) < \deg_{\sigma(k)}(v)\},\;\; j = \min\{k \mid \deg_{\sigma(k)}(v) < \deg_{\sigma(k)}(u)\}\,.$$

In accordance with (6), $u \sqsupset v$ would imply $x_{\sigma(j)} \in NM_{\mathcal{L}}(v,U)$. Therefore, $u \sqsubset v$.

Assume that (10) does not hold. Partition V into subsets V_1 and V_2 where $V_1 = V \setminus \bar{U} \neq \emptyset$ and $V_2 = V \cap \bar{U}$. Since $\langle V_2\rangle = \langle V\rangle$, we have

$$(\,\forall v \in V_1\,)\;(\,\exists u \in V_2\,)\;(\,\exists x \in NM_{\mathcal{L}}(u,V)\,)\,[\,u\cdot x \in V_1 \;\wedge\; u\cdot x \mid v\,]\,.$$

Consider now the set of all nonmultiplicative prolongations of the elements in V_2 such that these prolongations belong to V_1 and choose among them the prolongation $u_1 \cdot x_{i_1}$ with the maximal u_1 w.r.t. $\sqsupset$:

$$u_1 = \max_{\sqsupset}\{\, u \in V_2 \mid \exists x \in NM_{\mathcal{L}}(u, V),\ u \cdot x \in V_1 \,\}\,.$$

It is clear that $(\,\forall v \in V_1\,)\ [\,u_1 \sqsupset v\,]$. On the other hand,

$$u_1 \cdot x_{i_1} \notin V_2 \subset \bar{U} \Longrightarrow (\,\exists u_2 \in V_2\,)\ (\,w \in \mathcal{L}(u_2, V_2)\ \ [\,u_1 \cdot x_{i_1} = u_2 \cdot w\,]$$

and $u_2 \sqsupset u_1$. In accordance with (5)–(6) the elements in V_1 cannot generate nonmultiplicative variables for u_2. Therefore, $w \in \mathcal{L}(u_2, V)$ and V is not $\mathcal{L}$-autoreduced, a contradiction. □

As an immediate consequence of inclusion (10) we have the following result:

Corollary 2. *If U is an $\mathcal{L}$-basis of $\langle U \rangle$ where $\mathcal{L}$ is an involutive division pairwise generated by an antigraded monomial ordering, then the $\mathcal{L}$-autoreduction of U yields the minimal $\mathcal{L}$-basis of $\langle U \rangle$.*

Remark 6. With respect to this property of the $\sqsupset$-division generated by an antigraded ordering $\sqsupset$ it is similar to a globally defined involutive division [1]. For the last the partition of variables for a monomial is determined by the monomial itself irrespective of the other monomials in a set. Obviously, this implies inclusion (10). Pommaret division is an example of such globally defined one. An important consequence of this similarity is that for a $\sqsupset$-division $\mathcal{L}$, as well as for a globally defined division, an involutive algorithm augmented with $\mathcal{L}$-autoreduction outputs a minimal $\mathcal{L}$-basis. Such algorithms are considered, for example, in our first paper of [1] and also in [5].

4 Heuristical Superiority of $\succ_{\mathrm{alex}}$-division over Janet Division

In this section we consider algorithmic completion of monomial and polynomial sets to involution. We present here some results of our computational experiments on the monomial completion done with $\succ_{\mathrm{alex}}$-division and with Janet division. For these experiments we selected the $\succ_{\mathrm{alex}}$-division as a representative of the class of divisions pairwise generated by an antigraded total monomial ordering. In some of our computational experiments the $\succ_{\mathrm{grlex}}$-division was also used as a representative of the class generated by graded total monomial ordering. To perform computations we used Python, which serves as a scripting language for our *GINV* system written in C++ [6].

The monomial and polynomial completion algorithms presented below (for a more efficient version of the polynomial completion algorithm see [4]) output minimal involutive bases and are correct for an arbitrary constructive involutive division $\mathcal{L}$. Each of the algorithms terminates if and only if an $\mathcal{L}$-basis is finite for a given input. The last is a monomial set for the monomial completion or a polynomial set together with an admissible monomial ordering for the polynomial completion. Termination always holds when $\mathcal{L}$ is noetherian.

Algorithm. InvolutiveMonomialBasis$(U, \mathcal{L})$

```
Input: $U$, a finite set or list of monomials in $\mathcal{M}$; $\mathcal{L}$, an involutive division
Output: $\bar{U}$, a minimal $\mathcal{L}$-basis of $\langle U \rangle$
 1: choose $u \in U$ without proper divisors in $U \setminus \{u\}$
 2: $W := \{u\};\ V := \{ U \setminus \{u\} \cup \{ u \cdot x \mid x \in NM_{\mathcal{L}}(u, W) \}$
 3: while $V \neq \emptyset$ do
 4:    choose $v \in V$ without proper divisors in $V \setminus \{v\}$
 5:    $V := V \setminus \{v\}$
 6:    if $v \notin C_{\mathcal{L}}(W)$ then
 7:       $W := W \cup \{v\};\ V := V \cup \{ u \cdot x \mid u \in W, x \in NM_{\mathcal{L}}(u, W) \}$
 8:    fi
 9: od
10: return $\bar{U} := W$
```

The **while**-loop 3-9 in algorithm ***InvolutiveMonomialBasis*** shows that, given an input monomial set U, the computational efficiency of the completion is fully determined by the number of $\mathcal{L}$-nonmultiplicative prolongations analyzed in step 6. Thus, generally, that involutive division which leads to a smaller total number of nonmultiplicative prolongations to be processed in the course of monomial completion is more attractive from the computational point of view. This is also true for the polynomial case.

Algorithm. InvolutivePolynomialBasis $(F, \prec, \mathcal{L})$

```
Input: $F$, a finite set or list of polynomials in $\mathcal{K}[X] \setminus \{0\}$; $\prec$, an admissible
    ordering; $\mathcal{L}$, an involutive division
Output: $G$, a minimal involutive basis of $\langle F \rangle$
 1: choose $f \in F$ without proper divisors of $\mathrm{lm}(f)$ in $\mathrm{lm}(F) \setminus \{\mathrm{lm}(f)\}$
 2: $G := \{f\};\ Q := F \setminus G$
 3: do
 4:    $h := 0$
 5:    while $Q \neq \emptyset$ and $h = 0$ do
 6:       choose $p \in Q$ without proper divisors of $\mathrm{lm}(p)$ in $\mathrm{lm}(Q) \setminus \{\mathrm{lm}(p)\}$
 7:       $Q := Q \setminus \{p\};\ h := NF_{\mathcal{L}}(p, G)$
 8:    od
 9:    if $h \neq 0$ then
10:       for all $\{g \in G \mid \mathrm{lm}(h) \mid \mathrm{lm}(g)\}$ do
11:          $Q := Q \cup \{g\};\ G := G \setminus \{g\}$
12:       od
13:       $G := G \cup \{h\};\ Q := Q \cup \{ g \cdot x \mid g \in G,\ x \in NM_{\mathcal{L}}(\mathrm{lm}(g), \mathrm{lm}(G)) \}$
14:    fi
15: od while $Q \neq \emptyset$
16: return $G$
```

The number of $\mathcal{L}$-normal forms evaluated in algorithm ***InvolutivePolynomialBasis*** at step 7 is determined by the total number of $\mathcal{L}$-nonmultiplicative prolongations processed. In doing so, the normal form computation is the most costly step of the algorithm as well as in any algorithmic construction of Gröbner bases (cf. [18]).

Suppose we have two involutive divisions $\mathcal{L}_1$ and $\mathcal{L}_2$ and perform the monomial or polynomial completion for both divisions with the same input data, with the identical initialization steps 1-2 and with the same selection strategy of steps 4 and 6, respectively. Assume that at any run of the **while**-loop 3-9 in the monomial algorithm or at any run of the **do while**-loop 3-15 in the polynomial algorithm the relation

$$(\forall v \in V\,)\ [\,NM_{\mathcal{L}_1}(v,W) \subseteq NM_{\mathcal{L}_2}(v,W)\,] \tag{12}$$

or, respectively,

$$(\forall g \in G\,)\ [\,NM_{\mathcal{L}_1}(\mathrm{lm}(g),\mathrm{lm}(G)) \subseteq NM_{\mathcal{L}_2}(\mathrm{lm}(g),\mathrm{lm}(G))\,] \tag{13}$$

holds. If at least for one element v or q in the intermediate basis the inclusion (12) or (13) is proper, then for the given input data $\mathcal{L}_1$ is computationally more efficient than $\mathcal{L}_2$ (cf. [2]).

In addition, (12) and (13) imply the inequalities $|\bar{U}_1| \leq |\bar{U}_2|$ and $|\bar{G}_1| \leq |\bar{G}_2|$ for the output cardinalities where the subscripts 1 and 2 stay for $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively.

$\mathcal{J}$-(Janet) and $\mathcal{P}$-(Pommaret) divisions give an important example when inclusions (12) and (13) hold, and when the former division is computationally more efficient than the latter. Example 1 illustrates this fact.

Proposition 5. [1] *If a monomial set U is $\mathcal{P}$-autoreduced, then*

$$(\forall u \in U\,)\ [\,NM_{\mathcal{J}}(u,U) \subseteq NM_{\mathcal{P}}(u,U)\,]\,. \tag{14}$$

As it can be easily seen, in the algorithms ***InvolutiveMonomialBasis*** and ***InvolutivePolynomialBasis***, the intermediate sets U and G, respectively, are $\mathcal{L}$-autoreduced at every step of completion.

Remark 7. Whenever a $\mathcal{P}$-basis exists (finite) it is also a minimal $\mathcal{J}$-basis [19]. However, computation of the $\mathcal{P}$-basis is more efficient via $\mathcal{J}$-division due to the inclusion (14) that is proper for most of the intermediate data.

Definition 10. *Given an involutive division $\mathcal{L}$ and a monomial set U, the total number of $\mathcal{L}$-nonmultiplicative variables for the elements in U, that is, $\sum_{u\in U} |NM_{\mathcal{L}}(u,U)|$ will be called the* $\mathcal{L}$-size *of U and denoted by $\mathcal{L}(U)$.*

The $\mathcal{L}$-size $\mathcal{L}(\bar{U})$ of the output in algorithm ***InvolutiveMonomialBasis*** gives the number of nonmultiplicative prolongations processed in the course of completion to $\bar{U}$. To clarify this, denote by U_0 the monomial basis of $\langle U\rangle$ reduced in the Gröbner sense.

Proposition 6. *Let algorithm* ***InvolutiveMonomialBasis*** *take a monomial set U as an input and let $\bar{U}$ be the output for a noetherian and constructive involutive division $\mathcal{L}$. Then $\bar{U}$ is produced from U by running the* **while**-*loop in the algorithm exactly $\mathcal{L}(\bar{U})+|U\setminus\bar{U}|-1$ times provided repeated nonmultiplicative prolongations are avoided.*

Proof. Since $\mathcal{L}$ is constructive, the minimal involutive completion $\bar{U}$ of U is unique by Proposition 2. In addition to U_0 the input set U may contain some prolongations of U_0. Initially, the monomial set V contains $|U|-1$ elements of the input set and the nonmultiplicative prolongations of the element u selected at step 2. In the **while**-loop all nonmultiplicative prolongations of the intermediate set W are collected in V and examined at step 6. If the input set contains some nonmultiplicative prolongations of the elements in W they will be examined at a certain stage of the algorithm. All other elements in $U \setminus \{u\}$ whose number is given by $|U \setminus \bar{U}| - 1$ will also be processed in the loop. When the intermediate set W is enlarged at step 7 with element v, for every monomial that has been inserted in W the number of its nonmultiplicative prolongations may only be increased by the filter axiom in Definition 1. If this is the case, then the additional prolongations as well as those for v will be treated at a later run of the loop. □

Remark 8. Avoidance of repeated prolongations in the monomial or polynomial algorithms can be easily achieved by keeping for every element in the intermediate basis those of its nonmultiplicative variables for which the prolongations have been treated at step 6 or 7, respectively (see [4]).

We have performed computational experiments to analyze the behavior of the monomial completion for $\succ_{\text{alex}}$- and Janet divisions for different choices of the permutation σ on the variables which specifies these divisions in accordance to Definitions 2 and 9. In our experimentation, the input monomial sets had been randomly generated. The experimentation explicitly shows that the cardinality of the output monomial basis and its variation under a permutation on the variables are strongly correlated with the involutive size of the output basis.

Experimentally, we observed that nearly always $\succ_{\text{alex}}$-division yields a more compact basis than Janet division and, in addition, its variation under permutation on the variables is substantially less for the former division than for the latter one. The following example illustrates this experimental fact.

Example 2. $U = \{\, x_1x_2x_3^2x_5, x_2x_3x_4^2, x_4x_5 \,\}$

– Janet: (6, (3, 2)), (10, (4, 3)), (17, (5, 4)), (2, (5, 5)), (7, (6, 5)), (4, (6, 6)), (4, (7, 7)), (7, (7, 8)), (4, (8, 8)), (21, (8, 9)), (6, (8, 10)), (3, (9, 10)), (17, (9, 11)), (2, (10, 12)), (2, (10, 13)), (4, (11, 14)), (1, (11, 15)), (3, (12, 16)), $\sum_\sigma |\bar{U}| = 855,\ \sum_\sigma \mathcal{J}_\sigma(\bar{U}) = 930\,.$
– $\succ_{\text{alex}}$: (120, (3, 2)), $\sum_\sigma |\bar{U}| = 360,\ \sum_\sigma \succ_{\text{alex}_\sigma} (\bar{U}) = 240\,.$

Here U is the input monomial set, and for each of both divisions we show the sequence of outputs for $5! = 120$ permutations. Every element in the sequence has the structure $(i, (j, k))$ where i the number of cases obtained with the output basis $\bar{U}$ of cardinality j and of involutive size k. For each division, we also show the total sum of cardinalities of the output bases and of their involutive sizes. In this example for $\succ_{\mathrm{alex}}$-division both $|\bar{U}| = 3$ and $\mathcal{L}(\bar{U}) = 2$ are invariants under action of $\sigma \in S_5$, and the output basis is appreciably more compact than that for Janet division. For the last division, $|\bar{U}|$ varies from 3 to 16 and $\mathcal{J}(\bar{U})$ from 2 to 14.

We spent a lot of time to find monomial sets when $\succ_{\mathrm{alex}}$-division on average, taking into account all permutations on the variables, produces less compact basis than Janet division. One such set is as follows.

Example 3. $U = \{\, x_1^2 x_2 x_3^2, x_2^2 x_3 x_4^2, x_3^2 x_4 x_5^2, x_1 x_2 x_3 x_4 x_5, x_1^2 x_4^2 x_5 \,\}$

- Janet: (4, (16, 24)), (6, (17, 26)), (4, (18, 28)), (2, (18, 30)), (2, (19, 29)), (12, (19, 30)), (2, (19, 32)), (2, (20, 30)), (12, (20, 32)), (4, (20, 35)), (12, (21, 34)), (4, (21, 37)), (4, (22, 37)), (12, (23, 37)), (2, (23, 38)), (2, (23, 44)), (2, (24, 39)), (6, (24, 42)), (2, (24, 46)), (2, (25, 41)), (6, (25, 43)), (2, (28, 50)), (4, (28, 51)), (2, (29, 52)), (4, (30, 54)), (4, (33, 61)) ,
  $\sum_\sigma |\bar{U}| = 2648,\ \sum_\sigma \mathcal{J}_\sigma(\bar{U}) = 4482\,.$
- $\succ_{\mathrm{alex}}$: (6, (18, 31)), (18, (19, 33)), (34, (21, 38)), (12, (22, 41)), (6, (23, 42)), (18, (23, 45)), (2, (24, 46)), (6, (25, 49)), (8, (26, 50)), (2, (26, 53)), (2, (27, 53)), (2, (27, 54)), (4, (28, 57)) ,
  $\sum_\sigma |\bar{U}| = 2658,\ \sum_\sigma \succ_{\mathrm{alex}_\sigma} (\bar{U}) = 4432\,.$

The typical situation is shown in Figure 1. The integer nearby a sector shows $|\bar{U}|$ and its area is proportional to the total number of outputs of this cardinality. Sector 19-50 accumulates sets with cardinality varying from 19 to 50.

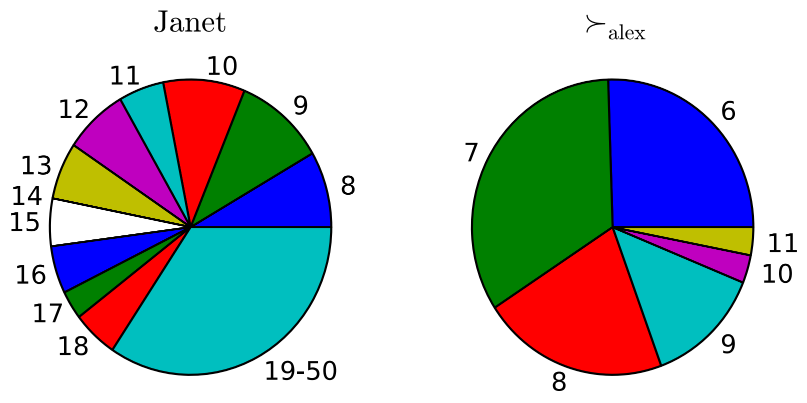


Fig. 1. $U = \{\, x_1^2 x_2^2 x_5, x_2^2 x_3 x_5, x_2 x_4, x_3^2, x_3 x_4 x_5^2 \,\}$

Table 2. Statistical data for monomial completion

$n = \lvert U\rvert$	3	4	5	6	7	8	9	10	11	12
Sample size	117	114	113	238	250	209	171	145	115	28
Janet										
Sample mean $\lvert\bar U\rvert$	3.8	8.3	16.5	32.0	73.9	173.9	376.6	817.4	1984.1	4653.9
Standard deviation	1.0	3.9	9.6	18.7	54.2	151.1	308.5	636.5	1629.4	3268.0
Standard error	0.1	0.4	0.9	1.2	3.4	10.5	23.6	52.9	151.9	617.6
$\succ_{\rm alex}$										
Sample mean $\lvert\bar U\rvert$	3.2	5.5	9.1	14.4	34.5	78.3	150.9	406.4	771.6	1452.8
Standard deviation	0.5	2.7	6.5	11.5	34.7	100.4	201.1	462.3	1065.3	1605.5
Standard error	0.0	0.3	0.6	0.7	2.2	6.9	15.4	38.4	99.3	303.4
$\succ_{\rm grlex}$										
Sample mean $\lvert\bar U\rvert$	4.6	12.4	29.2	67.8	171.4	404.6	896.5	2188.6	5412.1	11509.5
Standard deviation	1.7	8.7	18.9	42.4	104.2	303.2	714.4	1462.3	4241.0	7224.1
Standard error	0.2	0.8	1.8	2.7	6.6	21.0	54.6	121.4	395.5	1365.2

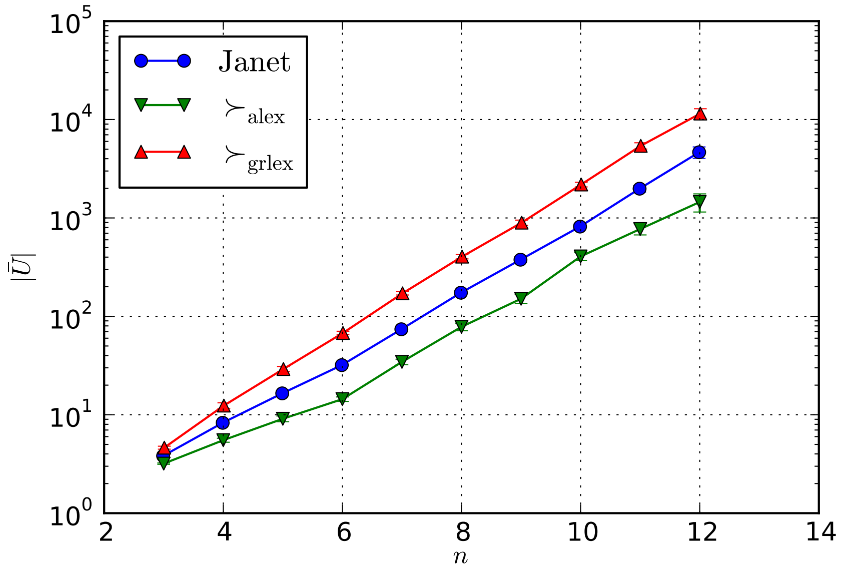


Fig. 2. Plotting the data of Table 2

An example of another type of computational experiments is presented in Table 2 and plotted in Figure 2. Here we randomly generated a reduced (in the Gröbner sense) monomial set U with n variables of cardinality $|U| = n$ and such that the exponents in every $u \in U$ range from 1 to n. For each such randomly generated set U we computed its minimal involutive completion $\bar U$. In Table 2 we show, for Janet division, for $\succ_{\rm alex}$-division and also for $\succ_{\rm grlex}$-division the number of samples generated for every $3 \leq n \leq 12$, the sample mean $|\bar{U}|$, the standard deviation, and the standard error.

5 Conclusion and Future Work

Our experimentation clearly shows that $\succ_{\mathrm{alex}}$-division is very attractive for involutive algorithms than the Janet division. Computational superiority of $\succ_{\mathrm{alex}}$-division over Janet division is expressed not only in a smaller number of nonmultplicative prolongations (number of involutive normal forms evaluated) to be examined but also in the higher stability under permutation of the variables. The last is also very important, since a priori it is not clear what particular division from the factorially many possible ones ($n!$ for n variables) is computationally better for a given problem. At the same time, Janet division is computationally better than any other division generated by an admissible monomial ordering. Our experiments with the $\succ_{\mathrm{grlex}}$-division confirm that.

It should be emphasized that differential completion is especially sensitive to the number of prolongations treated. In this case even a single extra prolongation may increase computational costs significantly. This was recently revealed with Thomas decomposition of nonlinear systems of partial differential equations into involutive subsystems [20] applied to practical problems. For this reason the $\succ_{\mathrm{alex}}$-division has particular attractivity for differential systems.

In practice, to make the involutive algorithms based on $\succ_{\mathrm{alex}}$-division computationally faster than the Janet division algorithms [4], one has to design appropriate data structures for the new division. Janet division admits special binary trees – Janet trees — as data structures providing fast search for the involutive divisor and fast partitioning updates for the variables in the intermediate basis after insertion of a new element into the basis (see [4] and the references therein). We shall look for good data structures for $\succ_{\mathrm{alex}}$-division. Another research direction is to take into account the "history" of completion in order to further decrease the number of nonmultiplicative prolongations processed in a completion algorithm.

Acknowledgements. The research presented in this paper was partially supported by grant 10-01-00200 from the Russian Foundation for Basic Research and by grant 3810.2010.2 from the Ministry of Education and Science of the Russian Federation. The authors are grateful to anonymous referees for their helpful remarks and suggestions.

References

1. Gerdt, V.P., Blinkov, Y.A.: Involutive bases of polynomial ideals. Mathematics and Computers in Simulation 45, 519–542 (1998); Minimal involutive bases, ibid, 543–560
2. Apel, J.: The theory of involutive divisions and an application to Hilbert function computations. J. Symbolic Computation 25, 683–704 (1998)

3. Janet, M.: Leçons sur les Systèmes d'Equations aux Dérivées Partielles. Cahiers Scientifiques, IV, Gauthier-Villars, Paris (1929)
4. Gerdt, V.P.: Involutive algorithms for computing Gröbner bases. In: Computational Commutative and Non-Commutative Algebraic Geometry, pp. 199–225. IOS Press, Amsterdam (2005)
5. Seiler, W.M.: Involution: The formal theory of differential equations and its applications in computer algebra. In: Algorithms and Computation in Mathematics, vol. 24. Springer, Heidelberg (2010)
6. Gerdt, V.P., Blinkov, Y. A.: Specialized computer algebra system GINV. Programming and Computer Software 34(2), 112–123 (2008)
7. http://wwwb.math.rwth-aachen.de/Janet/
8. Decker, W., Greuel, G.-M., Pfister, G., Schönemann, H.: Singular 3-1-2 - A computer algebra system for polynomial computations (2010), http://www.singular.uni-kl.de
9. http://cag.jinr.ru/wiki/
10. http://www.symbolicdata.org/
11. Gerdt, V.P.: Involutive division technique: some generalizations and optimizations. J. Math. Sciences 108(6), 1034–1051 (2002)
12. Chen, Y.-F., Gao, X.-S.: Involutive directions and new involutive divisions. Computers and Mathematics with Applications 41, 945–956 (2001)
13. Semenov, A.S.: On connection between constructive involutive divisions and monomial orderings. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2006. LNCS, vol. 4194, pp. 261–278. Springer, Heidelberg (2006)
14. Semenov, A.S.: Constructivity of involutive divisions. Programming and Computer Software 32(2), 96–102 (2007)
15. Semenov, A.S., Zyuzikov, P.A.: Involutive divisions and monomial orderings. Programming and Computer Software 33(3), 139–146 (2007); Involutive divisions and monomial orderings: Part II. Ibid 34(2), 107–111 (2008)
16. Cox, D., Little, J., O'Shea, D.: Using Algebraic Geometry, 2nd edn. Graduate Texts in Mathematics, vol. 185. Springer, New York (2005)
17. Greul, G.-M., Pfister, G.: A Singular Introduction to Commutative Algebra. Springer, Berlin (2007)
18. Becker, T., Weispfenning, V.: Gröbner Bases. A Computational Approach to Commutative Algebra. Graduate Texts in Mathematics, vol. 141. Springer, New York (1993)
19. Gerdt, V.P.: On the relation between Pommaret and Janet bases. In: Computer Algebra in Scientific Computing / CASC 2000, pp. 167–181. Springer, Berlin (2000)
20. Bächler, T., Gerdt, V.P., Lange-Hegermann, M., Robertz, D.: Thomas decomposition of algebraic and differential systems. In: Gerdt, V.P., Koepf, W., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2010. LNCS, vol. 6244, pp. 31–54. Springer, Heidelberg (2010)

Symbolic-Numerical Algorithms to Solve the Quantum Tunneling Problem for a Coupled Pair of Ions

A.A. Gusev, S.I. Vinitsky, O. Chuluunbaatar, V.P. Gerdt, and V.A. Rostovtsev

Joint Institute for Nuclear Research, Dubna, Russia
{gooseff,chuka,gerdt,rost}@jinr.ru, vinitsky@theor.jinr.ru

Abstract. Symbolic-numerical algorithms for solving a boundary value problem (BVP) for the 2D Schrödinger equation with homogeneous third type boundary conditions to study the quantum tunneling model of a coupled pair of nonidentical ions are described. The Kantorovich reduction of the above problem with non-symmetric long-range potentials to the BVPs for sets of the second order ordinary differential equations (ODEs) is given by expanding solution over the one-parametric set of basis functions. Symbolic algorithms for evaluation of asymptotics of the basis functions, effective potentials, and linear independent solutions of the ODEs in the form of inverse power series of independent variable at large values are given by using appropriate etalon equations. Benchmark calculation of quantum tunneling problem of coupled pair of identical ions through Coulomb-like barrier is presented.

1 Introduction

Quantum mechanical treatment on the basis of adiabatic description of penetration through a two-dimensional fission barrier has been studied for a long period of time [1,2]. Current interest is stimulated by the prominent papers in which the model of quantum tunneling problem of coupled pair of ions through truncated Coulomb barrier were investigated for identical mass and charges of ions [3,4]. Study of quantum tunneling problem for a coupled pair of ions with distinct mass and charges for their penetration through a nontruncated Coulomb barrier is an important problem.

The aim of this paper is to develop a symbolic-numerical algorithm (SNA) for solving the 2D boundary value problem (BVP) with homogeneous third type boundary conditions to analyze the above problem. In the framework of Kantorovich method (KM) [5], we search for a solution by means of expansion over the solution to the one-parametric eigenvalue problem calculated by program ODPEVP [6]. This way the BVP is reduced to a set of second order differential equations (ODEs) on the whole axis with homogeneous third type boundary conditions of general type. The main task here is to formulate these boundary conditions which are not invariant under reflection with respect to the x-axis.

This is because the conventionally used symmetric conditions are applicable only for identical particles. To apply the finite element method (FEM) to solving the BVP on a finite interval we need not only the adaptation of KANTBP 2.0 code [7], but also the elaboration of new symbolic algorithms to evaluate coefficients of the asymptotic expansion of both effective potential and solution to ODEs. These coefficients are needed to match evaluated asymptotic solutions with numerical ones at boundary points and extract the required matrix of transmission and the reflections amplitudes from numerical solutions.

In this paper, we present algorithms for calculation of the asymptotic expansions for solution to the eigenvalue problem with a long-range potential of general type. These asymptotic expansions are applied to evaluate the effective potentials of ODEs. The next step is to design a new algorithm for evaluation of the asymptotic expansion of linear independent solutions to ODEs. This distingue algorithm is substantially different from the previously elaborated one [8]. Instead of applying an expansion over the solution to an etalon equation, we propose to use an appropriate etalon equation with a long-range potential in the form of the inverse power series that provides a more economical and universal way to generate relevant recurrence relations and the corresponding FORTRAN subroutines.

The paper is organized as follows. In Section 2, the problem statement is done. In Section 3, the BVP is formulated for ODEs. Here the symbolic algorithms for the evaluation of asymptotic expansions of solutions to parametric BVP, for calculation of the corresponding integrals, and for the asymptotic expansion of linear independent solutions to ODEs together with their implementation in Maple are described. Section 4 is devoted to the benchmark calculation of the penetration coefficients for tunneling of the identical ions through long-range Coulomb like barriers. In Conclusion, we summarize the results and discuss the future applications of our SNAs.

2 Problem Statement

Wave function $\Psi(x,y)$ of model of heavy ion pair connected with oscillator potential scattering in the center mass coordinate system through Coulomb barriers satisfies the two-dimensional (2D) Schrödinger equation [3]:

$$\left\{-\frac{\partial^2}{\partial y^2}-\frac{\partial^2}{\partial x^2}+x^2+2(U_1(x_1)+U_2(x_2)-E)\right\}\Psi(x,y)=0, \tag{1}$$

where $x_1=s_2y+s_1x$, $x_2=s_2y-s_3x$ are variables in the laboratory coordinate system, parameters $s_2=\frac{\sqrt{m_1m_2}}{M}$, $s_1=\frac{m_2}{M}$, $s_3=\frac{m_1}{M}$ are defined via masses of ions m_1 and m_2 and total mass $M=m_1+m_2$ and reduced mass $\mu=\frac{m_1m_2}{M}$ in the oscillator units of length $x_{osc}=\sqrt{\frac{\hbar}{\mu\omega}}$ and energy $E_{osc}=\hbar\omega$ (ω is oscillator frequency). We choose barrier potential $U_i(x_i)$ of ions labelled by $i=1,2$ with charges $\hat{Z}_i>0$ in the form of the truncated Coulomb potential cut off on small $0<\bar{x}_{\min}<1$ and large $\bar{x}_{\max}>1$ distances from origin,

$$U_i\left(x_i\right)=\left\{\frac{\hat{Z}_i}{\bar{x}_{\min}}-\frac{\hat{Z}_i}{\bar{x}_{\max}},|x_i|\leq\bar{x}_{\min};\frac{\hat{Z}_i}{|x_i|}-\frac{\hat{Z}_i}{\bar{x}_{\max}},\begin{array}{l}|x_i|>\bar{x}_{\min}\\|x_i|\leq\bar{x}_{\max}\end{array};0,|x_i|>\bar{x}_{\max}\right\}, \quad (2)$$

or the Coulomb-like potentials that depend on the integer parameter $s \geq 2$ and truncation parameter $\bar{x}_{\min} > 0$ and defined as

$$U_i\left(x_i\right) = \hat{Z}_i(|x_i|^s + \bar{x}^s_{\min})^{-1/s}\,. \quad (3)$$

In both cases, the sum of barrier potential functions $U(x,y) = U_1\left(x_1\right) + U_2\left(x_2\right)$ has asymptotic form

$$U(x,y) \to \sigma_y \frac{Z_{12}}{y} + O(y^{-3}), \quad y \to \pm\infty, \quad (4)$$

where $\sigma_y = 1$ if $y > 0$ and $\sigma_y = -1$ if $y < 0$; $Z_{12} = 0$ for Eq. (2) and $Z_{12} = (\hat{Z}_1 + \hat{Z}_2)/s_2$ for Eq. (3).

The asymptotic boundary conditions for the solution $\Psi(y,x) = \{\Psi_{i_o}(y,x)\}_{i_o=1}^{N_o}$ with direction $v =\to$ can be written in the obvious form

$$\begin{aligned}\Psi_{i_o}(y \to -\infty, x) &\to B^{(0)}_{i_o}(x)\frac{\exp\left(\imath\left(p_{i_o}y - \sigma_y\frac{Z_{12}}{p_{i_o}}\ln(2p_{i_o}|y|)\right)\right)}{\sqrt{p_{i_o}}}\\ &\quad+\sum_{j=1}^{N_o}B^{(0)}_j(x)\frac{\exp\left(-\imath\left(p_jy - \sigma_y\frac{Z_{12}}{p_j}\ln(2p_j|y|)\right)\right)}{\sqrt{p_j}}R_{ji_o},\\ \Psi_{i_o}(y \to +\infty, x) &\to \sum_{j=1}^{N_o}B^{(0)}_j(x)\frac{\exp\left(\imath\left(p_jy - \sigma_y\frac{Z_{12}}{p_j}\ln(2p_j|y|)\right)\right)}{\sqrt{p_j}}T_{ji_o}, \quad (5)\\ \Psi_{i_o}(y, x \to \pm\infty) &\to 0.\end{aligned}$$

Here N_o is the number of open channels at fixed energy $2E = p^2 + \varepsilon^{(0)}_{i_o} > 0$, T_{ji_o} and R_{ji_o} are unknown transition and reflections amplitudes, $B^{(0)}_j(x)$ are the basis functions of oscillator with energy $\varepsilon^{(0)}_j = 2n+1$ at $n \geq 0$, $j = n+1$

$$\left\{-\frac{\partial^2}{\partial x^2} + x^2 - \varepsilon^{(0)}_j\right\}B^{(0)}_j(x) = 0, \quad \int_{-\infty}^{+\infty}B^{(0)}_j(x)B^{(0)}_{j'}(x)dx = \delta_{jj'}. \quad (6)$$

3 Formulation of BVP for a Set of the Kantorovich ODEs

We construct a desired solution of the BVP in the form of Kantorovich's expansion:

$$\Psi_{i'}(x,y) = \sum_{j=1}^{N}B_j(x;y)\chi_{ji'}(y). \quad (7)$$

The basis functions $B_j(x;y)$ of the fast variable x and the potential curves $E_i(y)$ that depend continuously on the slow variable y as a parameter are chosen as solutions of the BVPs for the equation on grid $\Omega_x\{x_{\min}(y), x_{\max}(y)\}$

$$\left\{-\frac{d^2}{dx^2}+x^2+2U(x,y)-\varepsilon_j(y)\right\}B_j(x;y)=0, \tag{8}$$

which are subject to the boundary, normalization, and orthogonality conditions

$$B_j(x_{\min}(y);y)=B_j(x_{\max}(y);y)=0,\ \langle B_i|B_j\rangle=\int\limits_{x_{\min}(y)}^{x_{\max}(y)} B_i(x;y)B_j(x;y)dx=\delta_{ij}. \tag{9}$$

By substituting (7) into (1)–(5) and by taking average over (9), we obtain the BVP for a set of N coupled ODEs that describes the slow subsystem for the partial solutions $\boldsymbol{\chi}^{(i')}(y)=(\chi_1^{(i')},...,\chi_N^{(i')})^T$:

$$\{\mathbf{H}-2E\,\mathbf{I}\}\ \boldsymbol{\chi}^{(i')}(y)=0,\quad \mathbf{H}=-\mathbf{I}\frac{d^2}{dy^2}+\mathbf{V}(y)+\mathbf{Q}(y)\frac{d}{dy}+\frac{d\mathbf{Q}(y)}{dy}. \tag{10}$$

Here $\mathbf{I}$ is the unit matrix, $\mathbf{V}(y)$ and $\mathbf{Q}(y)$ are the effective potential $N\times N$ matrices:

$$V_{ij}(y)=\varepsilon_j(y)\delta_{ij}+H_{ij}(y),\quad H_{ij}(y)=\int\limits_{x_{\min}(y)}^{x_{\max}(y)}\frac{\partial B_i(x;y)}{\partial y}\frac{\partial B_j(x;y)}{\partial y}dx, \tag{11}$$

$$Q_{ij}(y)=-\int\limits_{x_{\min}(y)}^{x_{\max}(y)} B_i(x;y)\frac{\partial B_j(x;y)}{\partial y}dx.$$

that is calculated numerically by means of program ODPEVP [6]. The boundary conditions at $y=y_{\min}\ll -1$ and $y=y_{\max}\gg 1$ are given by

$$\left.\frac{d\boldsymbol{\Phi}(y)}{dy}\right|_{y=y_{\min}}=\mathcal{R}(y_{\min})\boldsymbol{\Phi}(y_{\min}),\quad \left.\frac{d\boldsymbol{\Phi}(y)}{dy}\right|_{y=y_{\max}}=\mathcal{R}(y_{\max})\boldsymbol{\Phi}(y_{\max}), \tag{12}$$

where $\mathcal{R}(y)$ is an unknown $N\times N$ nonsymmetric matrix-function, $\boldsymbol{\Phi}(y)=\{\boldsymbol{\chi}^{(i_o)}(y)\}_{i_o=1}^{N_o}$ is the required $N\times N_o$ matrix solution, and N_o is the number of open channels, $N_o=\max_{2E\geq\varepsilon_j} j\leq N$ that is calculated numerically by means of the program KANTBP 3.0. It is a modified version of the program KANTBP 2.0 [7] including matching asymptotic solutions evaluated in the next sections with numerical ones at boundary points $y=y_{\min}\ll -1$ and $y=y_{\max}\gg 1$ in (12).

The matrix solution $\boldsymbol{\Phi}_v(y)=\boldsymbol{\Phi}(y)$ that describes the incidence of the particle and its scattering, having the asymptotic form “incident wave + outgoing waves”, is

$$\boldsymbol{\Phi}_v(y\to\pm\infty)=\begin{cases}\begin{cases}\mathbf{X}^{(+)}(y)\mathbf{T}_v, & y>0,\\ \mathbf{X}^{(+)}(y)+\mathbf{X}^{(-)}(y)\mathbf{R}_v, & y<0,\end{cases} & v=\rightarrow,\\ \begin{cases}\mathbf{X}^{(-)}(y)+\mathbf{X}^{(+)}(y)\mathbf{R}_v, & y>0,\\ \mathbf{X}^{(-)}(y)\mathbf{T}_v, & y<0,\end{cases} & v=\leftarrow,\end{cases} \tag{13}$$

with $\mathbf{R}_v$ and $\mathbf{T}_v$ being the reflection and transmission $N_o\times N_o$ matrices, v denotes the initial direction of the particle motion along the y-axis. Here the leading term of the asymptotic rectangle-matrix functions $\mathbf{X}^{(\pm)}(y)$ has the form

$$X^{(\pm)}_{ji_o}(y)\to p_j^{-1/2}\exp\left(\pm\imath\left(p_jy-\sigma_y\frac{Z_{12}}{p_j}\ln(2p_j|y|)\right)\right)\delta_{ji_o}, \tag{14}$$
$$p_{i_o}=\sqrt{2E-\varepsilon_{i_o}},\quad j=1,\ldots,N,\quad i_o=1,\ldots,N_o.$$

The matrix solution $\boldsymbol{\Phi}_v(y,E)$ is normalized so that

$$\int_{-\infty}^{\infty}\boldsymbol{\Phi}^\dagger_{v'}(y,E')\boldsymbol{\Phi}_v(y,E)dy=2\pi\delta(E'-E)\delta_{v'v}\mathbf{I}_{oo}, \tag{15}$$

where $\mathbf{I}_{oo}$ is the identity $N_o\times N_o$ matrix.

Suppose that a set of linear independent regular square-solutions $\boldsymbol{\Phi}_v^{\mathrm{reg}}(y)=\{\boldsymbol{\chi}^{(i')}_{\mathrm{reg}}(y)\}_{i'=1}^N$ for a problem under consideration with components $\boldsymbol{\chi}^{(i')}_{\mathrm{reg}}(y)=(\chi^{\mathrm{reg}}_{1i'}(y),\ldots,\chi^{\mathrm{reg}}_{Ni'}(y))^T$ is known at $y>0$, $v=\rightarrow$ or $y<0$, $v=\leftarrow$, i.e.,

$$\boldsymbol{\Phi}^{\mathrm{reg}}_{\rightarrow}(y)=\tilde{\mathbf{X}}^{(+)}(y),\quad y>0,\quad \boldsymbol{\Phi}^{\mathrm{reg}}_{\leftarrow}(y)=\tilde{\mathbf{X}}^{(-)}(y),\quad y<0.$$
$$\tilde{X}^{(\pm)}_{ji_o}(y)=X^{(\pm)}_{ji_o}(y),\quad j=1,\ldots,N,\quad i_o=1,\ldots,N_o. \tag{16}$$

In a case of some channels are closed, we must use additional leading terms of regular asymptotic functions correspondingly at $z>0$ and $z<0$

$$\tilde{X}^{(\pm)}_{ji_c}(y)\to q_j^{-1/2}\exp\left(\mp\left(q_jy+\sigma_y\frac{Z_{12}}{q_j}\ln(2q_j|y|)\right)\right)\delta_{ji_c}, \tag{17}$$
$$q_{i_c}=\sqrt{\epsilon_{i_c}-2E},\quad j=1,\ldots,N,\quad i_c=N_o+1,\ldots,N.$$

In this case, the required part of $\mathcal{R}_{\rightarrow}(y)$ at $y=y_{\max}>0$ and $\mathcal{R}_{\rightarrow}(y)$ matrix $y=y_{\min}<0$ can be found via the known set of linear independent regular solutions $\boldsymbol{\Phi}^{\mathrm{reg}}_v(y)$

$$\mathcal{R}_v(y)=\frac{d\boldsymbol{\Phi}^{\mathrm{reg}}_v(y)}{dy}\left(\boldsymbol{\Phi}^{\mathrm{reg}}_v(y)\right)^{-1}. \tag{18}$$

These matrix-functions $\mathcal{R}_v(y)$ by dimension of $N\times N$ are used for calculating numerical solutions $\boldsymbol{\Phi}^h_v(y)$ of BVP (10)–(12).

By using $\boldsymbol{\Phi}^h_{\rightarrow}(y_{\max})$ and $\mathcal{R}_{\rightarrow}(y)$ numerically calculated with KANTBP 3.0, we obtain the following matrix equations for the reflection $\mathbf{R}_{\rightarrow}$ and transmission $\mathbf{T}_{\rightarrow}$ matrices

$$\mathbf{Y}_{\rightarrow}^{(-)}(y_{\min})\mathbf{R}_{\rightarrow} = -\mathbf{Y}_{\rightarrow}^{(+)}(y_{\min}), \quad \mathbf{X}^{(+)}(y_{\max})\mathbf{T}_{\rightarrow} = \boldsymbol{\Phi}_{\rightarrow}^{h}(y_{\max}), \tag{19}$$
$$\mathbf{Y}_{\rightarrow}^{(\pm)}(y) = \frac{d\mathbf{X}^{(\pm)}(y)}{dy} - \mathcal{R}_{\rightarrow}(y)\mathbf{X}^{(\pm)}(y).$$

Note that, when some channels are closed, the $\mathbf{Y}_{\leftarrow}^{(\pm)}(y)$ and $\mathbf{X}^{(-)}(y)$ are rectangular $N \times N_o$ matrices. The reflection $\mathbf{R}_{\rightarrow}$ and transmission $\mathbf{T}_{\rightarrow}$ matrices are evaluated in terms of the pseudoinverse matrices of $\mathbf{Y}_{\rightarrow}^{(-)}(y_{\min})$ and $\mathbf{X}^{(+)}(y_{\max})$

$$\mathbf{R}_{\rightarrow} = -\left(\left(\mathbf{Y}_{\rightarrow}^{(-)}(y_{\min})\right)^{T}\mathbf{Y}_{\rightarrow}^{(-)}(y_{\min})\right)^{-1}\left(\mathbf{Y}_{\rightarrow}^{(-)}(y_{\min})\right)^{T}\mathbf{Y}_{\rightarrow}^{(+)}(y_{\min}), \tag{20}$$
$$\mathbf{T}_{\rightarrow} = \left(\left(\mathbf{X}^{(+)}(y_{\max})\right)^{T}\mathbf{X}^{(+)}(y_{\max})\right)^{-1}\left(\mathbf{X}^{(+)}(y_{\max})\right)^{T}\boldsymbol{\Phi}_{\rightarrow}^{h}(y_{\max}).$$

Having $\boldsymbol{\Phi}_{\leftarrow}^{h}(y_{\min})$ and $\mathcal{R}_{\leftarrow}(y)$ numerically calculated with KANTBP 3.0, we obtain the following matrix equations for the reflection $\mathbf{R}_{\leftarrow}$ and transmission $\mathbf{T}_{\leftarrow}$ matrices:

$$\mathbf{Y}_{\leftarrow}^{(+)}(y_{\max})\mathbf{R}_{\leftarrow} = -\mathbf{Y}_{\leftarrow}^{(-)}(y_{\max}), \quad \mathbf{X}^{(-)}(y_{\min})\mathbf{T}_{\leftarrow} = \boldsymbol{\Phi}_{\leftarrow}^{h}(y_{\min}), \tag{21}$$
$$\mathbf{Y}_{\leftarrow}^{(\pm)}(y) = \frac{d\mathbf{X}^{(\pm)}(y)}{dy} - \mathcal{R}_{\leftarrow}(y)\mathbf{X}^{(\pm)}(y).$$

Therefore, using the pseudoinverse matrices of $\mathbf{Y}_{\leftarrow}^{(+)}(y)$ and $\mathbf{X}^{(-)}(y)$, we obtain the following formulae:

$$\mathbf{R}_{\leftarrow} = -\left(\left(\mathbf{Y}_{\leftarrow}^{(+)}(y_{\max})\right)^{T}\mathbf{Y}_{\leftarrow}^{(+)}(y_{\max})\right)^{-1}\left(\mathbf{Y}_{\leftarrow}^{(+)}(y_{\max})\right)^{T}\mathbf{Y}_{\leftarrow}^{(-)}(y_{\max}), \tag{22}$$
$$\mathbf{T}_{\leftarrow} = \left(\left(\mathbf{X}^{(-)}(y_{\min})\right)^{T}\mathbf{X}^{(-)}(y_{\min})\right)^{-1}\left(\mathbf{X}^{(-)}(y_{\min})\right)^{T}\boldsymbol{\Phi}_{\leftarrow}^{h}(y_{\min}).$$

Let us now rewrite Eq. (13) in the matrix form at $y_{\pm} \to \pm\infty$

$$\begin{pmatrix} \boldsymbol{\Phi}_{\rightarrow}(y_+) & \boldsymbol{\Phi}_{\leftarrow}(y_+) \\ \boldsymbol{\Phi}_{\rightarrow}(y_-) & \boldsymbol{\Phi}_{\leftarrow}(y_-) \end{pmatrix} = \begin{pmatrix} \mathbf{0} & \mathbf{X}^{(-)}(y_+) \\ \mathbf{X}^{(+)}(y_-) & \mathbf{0} \end{pmatrix} + \begin{pmatrix} \mathbf{0} & \mathbf{X}^{(+)}(y_+) \\ \mathbf{X}^{(-)}(y_-) & \mathbf{0} \end{pmatrix}\mathbf{S}, \tag{23}$$

where the symmetric and unitary scattering matrix $\mathbf{S}$ is composed of the transmission and reflection matrices from (20) and (22)

$$\mathbf{S} = \begin{pmatrix} \mathbf{R}_{\rightarrow} & \mathbf{T}_{\leftarrow} \\ \mathbf{T}_{\rightarrow} & \mathbf{R}_{\leftarrow} \end{pmatrix}, \quad \mathbf{S}\mathbf{S}^{\dagger} = \mathbf{S}^{\dagger}\mathbf{S} = \mathbf{I}. \tag{24}$$

In addition, it should be noted that the functions $\mathbf{X}^{(\pm)}(y)$ satisfy relations

$$\mathbf{Wr}(\mathbf{Q}(y); \mathbf{X}^{(\mp)}(y), \mathbf{X}^{(\pm)}(y)) = \pm 2\imath\mathbf{I}_{oo}, \ \mathbf{Wr}(\mathbf{Q}(y); \mathbf{X}^{(\pm)}(y), \mathbf{X}^{(\pm)}(y)) = 0, \tag{25}$$

where $\mathbf{Wr}(\bullet;\mathbf{a}(y),\mathbf{b}(y))$ is a generalized Wronskian with a long derivative defined as

$$\mathbf{Wr}(\bullet;\mathbf{a}(y),\mathbf{b}(y)) = \mathbf{a}^T(y)\left(\frac{d\mathbf{b}(y)}{dy} - \bullet\mathbf{b}(y)\right) - \left(\frac{d\mathbf{a}(y)}{dy} - \bullet\mathbf{a}(y)\right)^T\mathbf{b}(y). \quad (26)$$

Remark 1. This Wronskian will be used below to estimate a desirable precision of the above expansion as well as the symmetry and unitarity properties of the scattering matrix **S** in (24).

Algorithm 1. Evaluating Effective Potential Asymptotics

Input. We evaluate the asymptotics of effective potentials (11) at large $|y|$ via the asymptotics of solutions to the eigenvalue problem (8), (9) at $|y/x| \gg 1$,

$$\left(\frac{d^2}{dx^2} + x^2 + 2U(x,y) - \varepsilon_j(y)\right)B_j(x;y) = 0, \int\limits_{x_{\min}(y)}^{x_{\max}(y)} B_i(x;y)B_j(x;y)dx = \delta_{ij}. (27)$$

with the Coulomb-like potential

$$2U(x,y) = 2\hat{Z}_1/\sqrt[s]{(s_2y+s_1x)^s + \bar{x}_{\min}^s} + 2\hat{Z}_2/\sqrt[s]{(s_2y-s_3x)^s + \bar{x}_{\min}^s}. \quad (28)$$

At **step 1** we find $B_j(x;y)$ and $\varepsilon_j(y)$ as a series expansion with $j = n+1$

$$B_j(x;y) = \sum_{k=0}^{k_{\max}} \frac{B_n^{(k)}(x)}{y^k},\ \varepsilon_j(y) = \sum_{k=0}^{k_{\max}} \frac{\varepsilon_n^{(k)}}{y^k}. \quad (29)$$

Substituting (29) to (27) and equating coefficients of the same powers of y, we arrive at a system of recurrence differential equations for evaluating coefficients $B_n^{(k)}(x)$ and $\varepsilon_n^{(k)}$, $k = 1,\ldots,k_{\max}$:

$$L(n)B_n^{(k)}(x) = f_n^{(k)}(x), \qquad L(n) = -\frac{d^2}{dx^2} - (2n+1) + x^2, \quad (30)$$

with the initial data $\varepsilon_n^{(0)} = 2n+1$ and with $B_n^{(0)}(x)$ as the known solution of the problem

$$\text{Ł}(n)B_n^{(0)}(x) = 0, \quad \int_{-\infty}^{+\infty} B_n^{(0)}(x)B_{n'}^{(0)}(x)dx = \delta_{nn'}. \quad (31)$$

In Eqs. (30), the right-hand sides $f_n^{(k)}(x)$ are defined by relations

$$f_n^{(k)}(x) = \sum_{p=1}^{k}(U^{(p)}(x) - \varepsilon_n^{(p)})B^{(k-p)}(x),$$

Table 1. Values of the partial sums (41) for $V_{jj} \equiv V_{jj}(y)$ from (11) depending on $k_{\max}$ for $s_1 = s_2 = s_3 = 1/2$, $\bar{x}_{\min} = 0.1$, $s = 8$, $\hat{Z}_1 = \hat{Z}_2 = 1$, $y = y_2^{match} = 12.5$. The last row contains the corresponding numerical values (n.v.).

$k_{\max}$	V_{11}	V_{22}	V_{33}	V_{44}	V_{55}	V_{66}
0	1.000000000	3.000000000	5.000000000	7.000000000	9.000000000	11.00000000
1	1.640000000	3.640000000	5.640000000	7.640000000	9.640000000	11.64000000
2	1.640000000	3.640000000	5.640000000	7.640000000	9.640000000	11.64000000
3	1.642048000	3.646144000	5.650240000	7.654336000	9.658432000	11.66252800
4	1.642048000	3.646144000	5.650240000	7.654336000	9.658432000	11.66252800
5	1.642067661	3.646242304	5.650495590	7.654827520	9.659238093	11.66372731
6	1.642065564	3.646236013	5.650485105	7.654812840	9.659219218	11.66370424
7	1.642065878	3.646238215	5.650492969	7.654832658	9.659259798	11.66377691
8	1.642065798	3.646237812	5.650491922	7.654830645	9.659256497	11.66377199
9	1.642065809	3.646237888	5.650492232	7.654831584	9.659258797	11.66377684
10	1.642065806	3.646237868	5.650492158	7.654831394	9.659258408	11.66377614
11	1.642065807	3.646237871	5.650492174	7.654831450	9.659258560	11.66377650
12	1.642065807	3.646237870	5.650492169	7.654831434	9.659258520	11.66377642
nv	1.642065807	3.646237871	5.650492170	7.654831437	9.659258529	11.66377644

where the coefficients $U^{(j)}(x)$ are determined by Taylor expansion of (28) at large y

$$2U(x,y) = \sum_{k=1}^{k_{\max}} \frac{U^{(k)}(x)}{y^k}, \tag{32}$$

$$\begin{aligned}
&U^{(1)}(x) = \sigma_y 2(\hat{Z}_1 + \hat{Z}_2)/s_2, && U^{(2)}(x) = \sigma_y 2x(\hat{Z}_1 s_1 - \hat{Z}_2 s_3)/s_2^2,\\
&U^{(3)}(x) = \sigma_y 2x^2(\hat{Z}_1 s_1^2 + \hat{Z}_2 s_3^2)/s_2^3, && U^{(4)}(x) = \sigma_y 2x^3(\hat{Z}_1 s_1^3 - \hat{Z}_2 s_3^3)/s_2^4,\\
&U^{(5)}(x) = \sigma_y 2x^4(\hat{Z}_1 s_1^4 + \hat{Z}_2 s_3^4)/s_2^5, && U^{(6)}(x) = \sigma_y 2x^5(\hat{Z}_1 s_1^5 - \hat{Z}_2 s_3^5)/s_2^6,\\
&U^{(7)}(x) = \sigma_y 2x^6(\hat{Z}_1 s_1^6 + \hat{Z}_2 s_3^6)/s_2^7, && U^{(8)}(x) = \sigma_y 2x^7(\hat{Z}_1 s_1^7 - \hat{Z}_2 s_3^7)/s_2^8,\\
&U^{(9)}(x) = \sigma_y 2x^8(\hat{Z}_1 s_1^8 + \hat{Z}_2 s_3^8)/s_2^9 - \sigma_y \bar{x}_{\min}^8(\hat{Z}_1 + \hat{Z}_2)/(4s_2^9),\\
&U^{(10)}(x) = \sigma_y 2x^9(\hat{Z}_1 s_1^9 - \hat{Z}_2 s_3^9)/s_2^{10} - \sigma_y 9x\bar{x}_{\min}^8(\hat{Z}_1 - \hat{Z}_2)/(8s_2^{10}).
\end{aligned}$$

The orthogonality and normalization conditions follow from (27) and (29)

$$I_{jj'}^{(k)} = \sum_{l=0}^{k} \int_{-\infty}^{\infty} B_{n_l}^{(l)}(x) B_{n_r}^{(k-l)}(x)dx = \delta_{k0}\delta_{n_l n_r} \tag{33}$$

where $n_l = j-1$, $n_r = j'-1$.

We find the asymptotics of matrix elements $H_{jj'}(y)$ and $Q_{jj'}(y)$ from (11) in the form of expansions

$$Q_{jj'}(y) = \sum_{k=1}^{k_{\max}} \frac{Q_{jj'}^{(k)}}{y^k}, \quad H_{jj'}(y) = \sum_{k=2}^{k_{\max}} \frac{H_{jj'}^{(k)}}{y^k}. \tag{34}$$

Table 2. The same as in Table 1, but for $Q_{jj'} \equiv Q_{jj'}(y)$ at $j \neq j'$

$k_{\max}$	$Q_{13}, 10^{-4}$	$Q_{15}, 10^{-6}$	$Q_{24}, 10^{-4}$	$Q_{26}, 10^{-6}$	$Q_{35}, 10^{-4}$	$Q_{46}, 10^{-4}$
3	0.00000000	0.000000	0.00000000	0.000000	0.00000000	0.00000000
4	1.73778562	0.000000	3.00993299	0.000000	4.25668806	5.49536066
5	1.73778562	0.000000	3.00993299	0.000000	4.25668806	5.49536066
6	1.79339476	1.605297	3.17046275	3.589554	4.57452077	6.02291528
7	1.78627679	1.605297	3.15813407	3.589554	4.55708537	6.00040628
8	1.78814526	1.713173	3.16568539	3.927259	4.57691814	6.04176657
9	1.78761568	1.705283	3.16415663	3.909616	4.57389135	6.03674256
10	1.78771659	1.711496	3.16457995	3.934625	4.57519301	6.03996585
11	1.78768515	1.710423	3.16444912	3.931266	4.57484597	6.03923893
12	1.78769198	1.710818	3.16447978	3.933127	4.57494688	6.03951537
nv	1.78769041	1.710734	3.16447143	3.932815	4.57491909	6.03944626

Here the coefficients $Q_{jj'}^{(k)}$ and $H_{jj'}^{(k)}$ are defined by the relations

$$Q_{jj'}^{(k)} = -\sum_{l=0}^{k-1} \int_{-\infty}^{+\infty} B_{n_l}^{(l)}(x) \hat{Q} B_{n_r}^{(k-1-l)}(x) dx, \quad \hat{Q} B_{n_l}^{(l)}(x) = l B_{n_l}^{(l)}(x),$$

$$H_{jj'}^{(k)} = \sum_{l=0}^{k-2} \int_{-\infty}^{+\infty} \hat{Q} B_{n_l}^{(l)}(x) \hat{Q} B_{n_r}^{(k-2-l)}(x) dx. \tag{35}$$

At **step 2**, we construct $B_n^{(k)}(x)$ as the expansion with unknown coefficients $b_{n;s}^{(k)}$

$$B_n^{(k)}(x) = \sum_{s=-M(k)}^{M(k)} b_{n;s}^{(k)} B_{n+s}^{(0)}(y). \tag{36}$$

Here $B_v^{(0)}(x)$ are solutions to (31) in terms of the Hermite polynomials [9]

$$B_v^{(0)}(x) = \frac{H_v(x) \exp(-x^2/2)}{\sqrt[4]{\pi}\sqrt{2^v}\sqrt{v!}}.$$

By means of the well-known recurrence relation for Hermite polynomials $H_v(x)$ we obtain the recurrence relations for basis functions $B_v^{(0)}(x)$:

$$xB_v^{(0)}(x) = +\frac{\sqrt{v+1}}{\sqrt{2}} B_{v+1}^{(0)}(x) + \frac{\sqrt{v}}{\sqrt{2}} B_{v-1}^{(0)}(x),$$

$$L(n) B_{n+s}^{(0)}(x) \equiv \left(-\frac{d^2}{dx^2} - (2n+1) + x^2 \right) B_{n+s}^{(0)}(x) = 2s B_{n+s}^{(0)}(x). \tag{37}$$

From (30), (32), and (37) we have the needed value of $M(k) = 2k+1$ in expansion (36) to provide calculation of nonzero terms only.

Substituting (36) to (30), taking into account (37), and equating coefficients of the identical powers of y, we arrive at a set of recurrence relations for evaluation of coefficients $E_n^{(k)}$ and $b_{n;s}^{(k)}$

$$2sb^{(k)}_{n;s} = f^{(k)}_{n;s}, \quad I^{(k)}_{jj'} = \sum_{l=0}^{k} \sum_{s=-2k-1}^{2k+1} b^{(l)}_{n_l;s} b^{(k-l)}_{n_r;s+n_l-n_r} = \delta_{k0}\delta_{n_l n_r}, \tag{38}$$

with initial data $\varepsilon^{(0)}_n = 2n+1$ and $b^{(0)}_{n;s} = \delta_{s0}$.

The corresponding coefficients $Q^{(k)}_{jj'}$ and $H^{(k)}_{jj'}$ in (35) have the following explicit form:

$$Q^{(k)}_{jj+t}(y) = -\sum_{k'=0}^{k-1} \sum_{s=\max(-k+1,k'-k+1-t)}^{\min(k-1,k-1-k'-t)} (k-1-k') b^{(k')}_{n;n+s} b^{(k-1-k')}_{n+t;n+s},$$

$$H^{(k)}_{jj+t}(y) = \sum_{k'=0}^{k-2} \sum_{s=\max(-k+2,k'-k+2-t)}^{\min(k-2,k-2-k'-t)} k'(k-2-k') b^{(k')}_{n;n+s} b^{(k-2-k')}_{n+t;n+s}. \tag{39}$$

At **step 3**, we evaluate sequentially the solutions $b^{(k)}_{n;s}$ and $\varepsilon^{(k)}_n$ to the set of recurrence relations (38) for each kth order ($k = 1, \ldots, k_{\max}$):

$$f^{(k)}_{n;0} = 0 \to \varepsilon^{(k)}_n; \quad b^{(k)}_{n;s\neq 0} = f^{(k)}_{n;s}/(s); I^{(k)}_{ii} = \delta_{k0} \to b^{(k)}_{n;0}. \tag{40}$$

At **step 4**, we substitute coefficients $b^{(k)}_{n;s}$ calculated in (40) into the expressions for the matrix elements (34), (39) evaluated at **step 2** and taking into account coefficients $\varepsilon^{(k)}_j$ calculated in (40). In doing so we produce the **output** containing the matrix elements as a series expansion of inverse powers of y for $k = 0, 1, \ldots, k_{\max}$ at $j, j' = 1, \ldots, N$ ($\varepsilon^{(k<0)}_j = H^{(k<2)}_{jj'} = Q^{(k<1)}_{jj'} = 0$):

$$\varepsilon_j(y) = \sum_{k=0}^{k_{\max}} \frac{\varepsilon^{(k)}_j}{y^k}, \quad H_{jj'}(y) = \sum_{k=2}^{k_{\max}} \frac{H^{(k)}_{jj'}}{y^k}, \quad Q_{jj'}(y) = \sum_{k=1}^{k_{\max}} \frac{Q^{(k)}_{jj'}}{y^k}. \tag{41}$$

The above described calculation was performed by the algorithm implemented in MAPLE up to $k_{\max} = 12$. For example, the explicit expression of the desirable nonzero coefficients $\varepsilon^{(k)}_j$, $H^{(k)}_{ij} = H^{(k)}_{ji}$ and $Q^{(k)}_{ij} = -Q^{(k)}_{ji}$ reads as ($j = n+1$):

$$\begin{aligned}
\varepsilon^{(0)}_j &= (2n+1), \quad \varepsilon^{(1)}_j = \sigma_y \frac{2(\hat{Z}_2 + \hat{Z}_1)}{s_2}, \quad \varepsilon^{(3)}_j = \sigma_y \frac{(2n+1)(\hat{Z}_2 s_3^2 + \hat{Z}_1 s_1^2)}{s_2^3}, \\
\varepsilon^{(4)}_j &= -\frac{(\hat{Z}_2 s_3 - \hat{Z}_1 s_1)^2}{s_2^4}, \quad \varepsilon^{(5)}_j = \sigma_y \frac{3(2n^2+2n+1)(\hat{Z}_2 s_3^4 + \hat{Z}_1 s_1^4)}{2s_2^5}, \\
Q^{(5)}_{jj-3} &= -\sigma_y \frac{\sqrt{n-2}\sqrt{n-1}\sqrt{2}\sqrt{n}(\hat{Z}_2 s_3^3 - \hat{Z}_1 s_1^3)}{3s_2^4}, \\
Q^{(4)}_{jj-2} &= -\sigma_y \frac{3\sqrt{n-1}\sqrt{n}(\hat{Z}_2 s_3^2 + \hat{Z}_1 s_1^2)}{4s_2^3},
\end{aligned} \tag{42}$$

Table 3. The same as in Table 1, but for $H_{jj'} \equiv H_{jj'}(y)$ at $j \neq j'$

$k_{\max}$	$H_{13}, 10^{-10}$	$H_{15}, 10^{-8}$	$H_{24}, 10^{-9}$	$H_{26}, 10^{-6}$	$H_{35}, 10^{-9}$	$H_{46}, 10^{-8}$
8	0.000	-7.3972	0.000	-1.65406	0.000	0.0000
9	0.000	-7.3972	0.000	-1.65406	0.000	0.0000
10	0.683	-8.1862	1.972	-1.90107	4.463	0.8643
11	0.683	-8.1256	1.972	-1.88752	4.463	0.8643
12	0.780	-8.1839	2.347	-1.91203	5.488	1.0969
nv	0.782	-8.1763	2.376	-1.91042	5.608	1.1334

$$Q^{(3)}_{jj-1} = -\sigma_y \frac{\sqrt{2}\sqrt{n}(-\hat{Z}_1 s_1 + \hat{Z}_2 s_3)}{s_2^2}, \quad Q^{(5)}_{jj-1} = -\sigma_y \frac{3\sqrt{2}n\sqrt{n}(\hat{Z}_2 s_3^3 - \hat{Z}_1 s_1^3)}{s_2^4},$$

$$H^{(7)}_{jj-3} = -\frac{3\sqrt{2}\sqrt{n}\sqrt{n-1}\sqrt{n-2}(\hat{Z}_2 s_3^2 + \hat{Z}_1 s_1^2)(-\hat{Z}_1 s_1 + \hat{Z}_2 s_3)}{2 s_2^5},$$

$$H^{(6)}_{jj-2} = -\frac{2\sqrt{n}\sqrt{n-1}(\hat{Z}_2 s_3 - \hat{Z}_1 s_1)^2}{s_2^4},$$

$$H^{(7)}_{jj-1} = \frac{3n\sqrt{2}\sqrt{n}(\hat{Z}_2 s_3^2 + \hat{Z}_1 s_1^2)(\hat{Z}_2 s_3 - \hat{Z}_1 s_1)}{2 s_2^5},$$

$$H^{(6)}_{jj} = \frac{2(2n+1)(\hat{Z}_2 s_3 - \hat{Z}_1 s_1)^2}{s_2^4}.$$

Remark 2. In the case of $\hat{Z}_1 = \hat{Z}_2$, $s_1 = s_3$ (i.e., for equal masses and charges), the set of equations (10) has even and odd parity solutions that are calculated separately: for the even solutions $n = 2j-2$ and for the odd solutions $n = 2j-1$. In this case, the above coefficients which contain terms like $(-\hat{Z}_1 s_1 + \hat{Z}_2 s_3)$ vanish when they have no terms $(\hat{Z}_1 s_1 + \hat{Z}_2 s_3)$.

Algorithm 2. Evaluation of the Asymptotic Solutions

Input. We calculate the asymptotic solution to the set of N ODEs at large values of the independent variable $|y| \gg 1$

$$\left[-\frac{1}{y^{d-1}}\frac{d}{dy}y^{d-1}\frac{d}{dy} + \varepsilon_i(y) + H_{ii}(y) - 2E\right]\chi_{ii'}(y) \tag{43}$$
$$= \sum_{j=1, j\neq i}^{N} \left[-Q_{ij}(y)\frac{d}{dy} - \frac{1}{y^{d-1}}\frac{d}{dy}y^{d-1}Q_{ij}(y) - H_{ij}(y)\right]\chi_{ji'}(y).$$

Here $d \geq 1$ is the dimension of configuration space of a general scattering problem [7] while in the considered case (10), we put $d = 1$ and calculate asymptotic solution on two intervals $-\infty < y \leq y_{\min}$ and $y_{\max} \leq y < \infty$. We suppose that coefficients of Eqs. (43) are present in the general form (41) and, in particular, in the form (42).

Step 1. We construct the solution to Eqs. (43) in the form:

$$\chi_{ji'}(y) = \left(\phi_{ji'}(y) + \psi_{ji'}(y)\frac{d}{dy}\right) R_{i'}(y), \tag{44}$$

where $\phi_{ji'}(y)$ and $\psi_{ji'}(y)$ are unknown functions, $R_{i'}(y)$ is known function. We choose $R_{i'}(y)$ as solutions of the auxiliary problem treated like etalon equation ($Z_{i'}^{(k<1)} = Z_{i'}^{(k>k'_{\max})} = 0$):

$$\left[-\frac{1}{y^{d-1}}\frac{d}{dy}y^{d-1}\frac{d}{dy} + \sum_{k=1}^{k'_{\max}}\frac{Z_{i'}^{(k)}}{y^k} - p_{i'}^2\right] R_{i'}(y) = 0. \tag{45}$$

Remark 3. If $Z_{i'}^{(k\geq3)} = 0$ then solutions to the last equation are presented via hypergeometric functions, exponential, trigonometric, Bessel, Coulomb functions, etc. For example, if the leading terms of the asymptotic solutions are given by formula

$$R_{i'}(y) = \frac{1}{\sqrt{p_{i'}y^{d-1}}}\exp\left(\pm\imath\left(p_{i'}y - \frac{Z_{i'}^{(1)}}{2p_{i'}}\ln(2p_{i'}|y|)\right)\right), \tag{46}$$

the coefficient $Z_{i'}^{(2)}$ of potential in the etalon equation (45) has the form:

$$Z_{i'}^{(2)} = -\frac{(d-3)(d-1)}{4} \pm \imath\frac{Z_{i'}^{(1)}}{p_{i'}} - \frac{(Z_{i'}^{(1)})^2}{p_{i'}^2}. \tag{47}$$

Step 2. At this step, we compute the coefficients $\phi_{i'}(y)$ and $\psi_{i'}(y)$ of the expansion (44) in the form of series by inverse powers of y ($\phi_{ji'}^{(k'<0)} = \psi_{ji'}^{(k'<0)} = 0$):

$$\phi_{ji'}(y) = \phi_{ji'}^{(0)} + \sum_{k'=1}^{k_{\max}}\frac{\phi_{ji'}^{(k')}}{y^{k'}}, \quad \psi_{ji'}(y) = \psi_{ji'}^{(0)} + \sum_{k'=1}^{k_{\max}}\frac{\psi_{ji'}^{(k')}}{y^{k'}}. \tag{48}$$

After substitution of (44),(48) into (43) with the use of Eq. (45), we arrive at the set of recurrence relations at $k' \leq k_{\max}$:

$$\left(\varepsilon_i^{(0)} - 2E + p_{i'}^2\right)\phi_{ii'}^{(k')} + \left(\varepsilon_i^{(1)} - Z_{i'}^{(1)}\right)\phi_{ii'}^{(k'-1)} - 2p_{i'}^2(k'-1)\psi_{ii'}^{(k'-1)} = -f_{ii'}^{(k')}, \tag{49}$$

$$\left(\varepsilon_i^{(0)} - 2E + p_{i'}^2\right)\psi_{ii'}^{(k')} + 2(k'-1)\phi_{ii'}^{(k'-1)} + \left(\varepsilon_i^{(1)} - Z_{i'}^{(1)}\right)\psi_{ii'}^{(k'-1)} = -g_{ii'}^{(k')},$$

where the right-hand sides $f_{ii'}^{(k)}$ and $g_{ii'}^{(k)}$ are defined by relations

$$f_{ii'}^{(k')} = -(k'-2)(k'-d)\phi_{ii'}^{(k'-2)} + \sum_{k=2}^{k'}\left(V_{ii}^{(k)} - Z_{i'}^{(k)}\right)\phi_{ii'}^{(k'-k)}$$

$$+\sum_{k=1}^{k'}\Big(Z_{i'}^{(k)}(2k'-2-k)\psi_{ii'}^{(k'-k-1)}+\sum_{j=1,j\neq i}^{N}\Big(\sum_{k''=1}^{k'}2Q_{ij}^{(k)}Z_{i'}^{(k'')}\psi_{ji'}^{(k'-k-k'')}$$
$$-2p_{i'}^2Q_{ij}^{(k)}\psi_{ji'}^{(k'-k)}+Q_{ij}^{(k)}(-2k'+k+d+1)\phi_{ji'}^{(k'-k-1)}+V_{ij}^{(k)}\phi_{ji'}^{(k'-k)}\Big)\Big); \qquad (50)$$
$$g_{ii'}^{(k)}=-(k'-1)(k'-3+d)\psi_{ii'}^{(k'-2)}+\sum_{k=2}^{k'}\left(V_{ii}^{(k)}-Z_{i'}^{(k)}\right)\psi_{ii'}^{(k'-k)}$$
$$+\sum_{j=1,j\neq i}^{N}\sum_{k=1}^{k'}\Big(2Q_{ij}^{(k)}\phi_{ji'}^{(k'-k)}-Q_{ij}^{(k)}(2k'+d-3-k)\psi_{ji'}^{(k'-k-1)}+V_{ij}^{(k)}\psi_{ji'}^{(k'-k)}\Big)$$

with initial conditions $p_{i'}^2=2E-\varepsilon_{i'}^{(0)}$, $\phi_{ii'}^{(0)}=\delta_{ii'}$, $\psi_{ii'}^{(0)}=0$, at $i'=i_o$ run the open channels $i_o=1,...,N_o$ and $p_{i'}=\imath q_{i'}$, $q_{i'}>0$, $q_{i'}^2=\varepsilon_{i'}^{(0)}-2E$ at $i'=i_c$ run the closed channels $i_c=N_o+1,...,N$ that follow from (14) and (17). Also from Eq. (49) at $k'=1$ and $i=i'$,

$$\left(\varepsilon_{i'}^{(1)}-Z_{i'}^{(1)}\right)\phi_{i'i'}^{(0)}=0,\quad \left(\varepsilon_{i'}^{(1)}-Z_{i'}^{(1)}\right)\psi_{i'i'}^{(0)}=0, \qquad (51)$$

we obtain condition $Z_{i'}^{(1)}=\varepsilon_{i'}^{(1)}$.

Step 3. Here we perform calculation of the coefficients $\phi_{ii'}^{(k')}$ and $\psi_{ii'}^{(k')}$ by a step–by–step procedure of solving Eqs. (49) for $2E\neq\varepsilon_{i'}^{(0)}$, $i\neq i'$ and $k'=2,\ldots,k_{\max}$:

$$\begin{aligned}
\phi_{ii'}^{(k')}&=\left[\varepsilon_i^{(0)}-\varepsilon_{i'}^{(0)}\right]^{-1}\left[-f_{ii'}^{(k')}-\left(\varepsilon_i^{(1)}-Z_{i'}^{(1)}\right)\phi_{ii'}^{(k'-1)}+2p_{i'}^2(k'-1)\psi_{ii'}^{(k'-1)}\right],\\
\psi_{ii'}^{(k')}&=\left[\varepsilon_i^{(0)}-\varepsilon_{i'}^{(0)}\right]^{-1}\left[-g_{ii'}^{(k')}-2(k'-1)\phi_{ii'}^{(k'-1)}-\left(\varepsilon_i^{(1)}-Z_{i'}^{(1)}\right)\psi_{ii'}^{(k'-1)}\right],\\
\phi_{i'i'}^{(k'-1)}&=-\left[2(k'-1)\right]^{-1}g_{i'i'}^{(k)}, \qquad (52)\\
\psi_{i'i'}^{(k'-1)}&=\left[2(k'-1)\left(2E-\varepsilon_{i'}^{(0)}\right)\right]^{-1}f_{i'i'}^{(k)}.
\end{aligned}$$

The above described algorithm has been implemented in MAPLE and FORTRAN to calculate the desirable $\phi_{ii'}^{(k')}$ and $\psi_{ii'}^{(k')}$ in the **output** up to $k_{\max}-1=11$ order.

Remark 4. The choice of appropriate values $y_{\min}$ and $y_{\max}$ for the constructed expansions of the linearly independent solutions for $p_{i_o}>0$ is controlled by the fulfillment of the Wronskian condition (26)

$$y^{d-1}Wr(\mathbf{Q}(y);\boldsymbol{\chi}^*(y),\boldsymbol{\chi}(y))=\pm 2\imath\mathbf{I}_{oo} \qquad (53)$$

up to the prescribed precision ε_{Wr}.

As a result, Algorithms 1 and 2 generate required asymptotic solution (5) up to the order $O(|y|^{-k_{\max}})$ at $|y|/|x|\gg 1$ that reduce the BVP (1) from plane $\mathbf{R^2}$

$$\psi_{i'}^{as}(x,y)=\sum_{j=1}^{N}\sum_{k=0}^{k_{\max}}y^{-k}\sum_{s=\min(1-j,-M(k))}^{M(k)}B_{j-1+s}^{(0)}(x)b_{j-1;s}^{(k)}\left(\phi_{ji'}^{(k-p)}+\psi_{ji'}^{(k-p)}\frac{d}{dy}\right)R_{i'}(y) \quad (54)$$

to a finite domain $\Omega_{xy}=[\Omega_x\{x_{\min},x_{\max}\}\times\Omega_y\{y_{\min},y_{\max}\}]$.

4 Benchmark Calculation of Penetration Coefficient

As a benchmark calculation we consider the BVPs (1)–(6) that model the quantum tunneling problem for a coupled pair of identical ions with the following values of parameters: $\bar{x}_{\max} = 5$ for Eq. (2) and $s = 8$ for Eq. (3), $s_1 = s_2 = s_3 = 1/2$, $\bar{x}_{\min} = 0.1$, $\hat{Z}_1 = \hat{Z}_2 = 0.5$ and $\hat{Z}_1 = \hat{Z}_2 = 1$ in oscillator units. For given number N of ODES (10), the values $x_{\min}$ and $x_{\max}$ of grid $\Omega_x\{x_{\min}, x_{\max}\}$ are chosen in the region $|x| > x_0 = \sqrt{2N+1}$ where the Hermite polynomial [9] (or of wave function in a general case) has none zeros. These values are computed with prescribed precision $eps > 0$ from the condition

$$\exp\left(\int_{x_0}^{x} dx\sqrt{x^2 - x_0^2}\right) \leq eps,$$

which in the given case leads to inequality

$$\exp\left(-x\sqrt{x^2 - x_0^2}/2\right)\left(x + \sqrt{x^2 - x_0^2}\right)^{x_0^2/2} x_0^{-x_0^2/2} \leq eps. \tag{55}$$

To find an approximate solution, at the first step we choose the initial approximation $x_{\max} = x_0$, after that it is increased with step equal 1 until (55) is satisfied. Values $y_{\min} < x_{\min}$ and $y_{\max} > x_{\max}$ were chosen from the condition that potential (2) or (3) is negligible on the interval $x_{\min} < x < x_{\max}$.

The matching points y_1^{match} and y_2^{match} of the numerical (11) and asymptotic (41) effective potential were calculated as follows:

$$y_1^{match} = \min\{y_1^E, y_1^Q, y_1^H\}, \quad y_2^{match} = \max\{y_2^E, y_2^Q, y_2^H\},$$
$$y_t^E = \sigma_y \sqrt[k_{\max}]{\frac{|E_N^{(k_{\max})}|}{eps}}, \quad y_t^Q = \sigma_y \sqrt[k_{\max}]{\frac{|Q_{NN-1}^{(k_{\max})}|}{eps}}, \quad y_t^H = \sigma_y \sqrt[k_{\max}]{\frac{|H_{NN}^{(k_{\max})}|}{eps}},$$

since $|E_j^{(k_{\max})}| < |E_N^{(k_{\max})}|$, $|Q_{jj'}^{(k_{\max})}| < |Q_{NN-1}^{(k_{\max})}|$, $|H_{jj'}^{(k_{\max})}| < |H_{NN}^{(k_{\max})}|$. So, the values $y_{\min}$ and $y_{\max}$ are chosen from the inequalities $y_{\min} < y_1^{match} < x_{\min}$ and $y_{\max} > y_2^{match} > x_{\max}$ taking into account. This gives

$$y_{\min} = \min\left[y_1^{match}, \min_j\left(-\sqrt[k_{\max}]{\frac{|\phi_{ji_o}^{(k_{\max})}|}{eps}}\right), \min_j\left(-\sqrt[k_{\max}]{\frac{|\psi_{ji_o}^{(k_{\max})}|}{eps}}\right)\right],$$
$$y_{\max} = \max\left[y_2^{match}, \max_j\left(\sqrt[k_{\max}]{\frac{|\phi_{ji_o}^{(k_{\max})}|}{eps}}\right), \max_j\left(\sqrt[k_{\max}]{\frac{|\psi_{ji_o}^{(k_{\max})}|}{eps}}\right)\right]. \tag{56}$$

In the considered examples, we used the grids $\Omega_x\{x_{\min}, x_{\max}\} = \{-10(768)10\}$ and $\Omega_y\{y_{\min}, y_{\max}\} = \{-125(200)-25(100)-6(200)6(100)25(200)125\}$ with the Lagrange elements of the order $p = 4$ between the nodes. In the above grids Ω_x and Ω_y, the number of grid elements is shown in the parentheses.

To illustrate Remark 2 by an example, we can point out the lines for $k_{\max} = 3$ (containing only zero values) in Table 2. Other zero values in Tables 2 and 3 point out different leading terms in the inverse power series expansion of matrix elements between various eigenfunctions. The numerical values of effective potentials calculated by ODPEVP [6] with a given precision *eps* of order of 10^{-10} in the last line of the Tables 1, 2 and 3 are in a good agreement with the asymptotic values from (41) in the matching points $y = y_t^{match}$.

In the calculation of solutions, we used the etalon equation (45) at $d = 1$ with the two sets of parameters taken in the first case as in Remark 3 and in second case as $k'_{\max} = 1$, $Z_{i'}^{(1)} = 2\sigma_y Z_{12}$, that corresponds to the known solutions on the open channels

$$R_{i_o}^{\pm}(p_{i_o}, y) = p_{i_o}^{-1/2} \begin{cases} (G_0(p_{i_o}, +y) \pm \imath F_0(p_{i_o}, +y)) \exp(\mp \imath \sigma_{i_o})/2, \, y > 0, \\ (G_0(p_{i_o}, -y) \mp \imath F_0(p_{i_o}, -y)) \exp(\pm \imath \sigma_{i_o})/2, \, y < 0, \end{cases} \qquad (57)$$

and on the closed channels

$$R_{i_c}(q_{i_c}, y) = q_{i_c}^{-1/2} t \exp(-t/2)\, U(1 + Z_{12}/q_{i_c}, 2, t), \quad t = 2q_{i_c}|y|. \qquad (58)$$

Here $F_0(p_{i_o}, y)$ and $G_0(p_{i_o}, y)$ are regular and irregular continuum zero order Coulomb functions; $\sigma_{i_o} = \arg \Gamma(1 + \imath Z_{12}/p_{i_o})$ is the Coulomb phase shift [9]; and $U(a, b, c)$ is the confluent hypergeometric function of second kind.

Remark 5. In the numerical calculation, the exponential small factor $\exp(-t/2)$ in $R_{i_c}(q_{i_c}, y)$ and its first derivative was neglected since this factor is canceled during evaluation of $\mathcal{R}(y)$ matrix in Eq. (18).

Required reflection $\mathbf{R}_{\rightarrow}$ and transmission $\mathbf{T}_{\rightarrow}$ matrixes calculated by formulas (20) via matrix of logarithmic derivatives $\mathcal{R}_{\rightarrow}(y)$ and solution $\boldsymbol{\Phi}^h_{\rightarrow}(y_{\max})$ calculated numerically on the above grid $\Omega_y\{y_{\min}, y_{\max}\}$ by means of the program KANTBP 3.0, including matching in the boundary points $y_{\min}$ and $y_{\max}$ of (12) with asymptotic solution evaluated in first case has the error of order 0.1% in comparison with a more accurate result obtained with asymptotic solution evaluated in the second case.

According to Remarks 1 and 4, the Wronskian condition depends on the number N of ODEs, on the value of threshold energy, on the type of etalon equation, etc. At the boundary points $y_{\min}$ and $y_{\max}$ of the above grid $\Omega_y\{y_{\min}, y_{\max}\}$, the absolute values ε_{Wr} of components of difference between the calculated Wronskian and its theoretical value (53) are less then 10^{-11}.

The total probabilities $T \equiv T_{11} = \sum_{j=1}^{N_o} |T_{1j}|^2$ of penetration through Truncated Coulomb (2) and Coulomb-like (3) potential barriers are shown in Fig. 1. The first of them is in a good agrement with results obtained by solving the BVP (1), (2), (5), and (6) in the 2D domain using Numerov method in papers [3,4]. These pictures illustrate the important peculiarity that a more realistic nontruncated Coulomb-like barrier having a more wide than truncated one, leads to a set of the probability maximums having a bigger half-width. It can be used for verification of the models and quantum transparency effect.

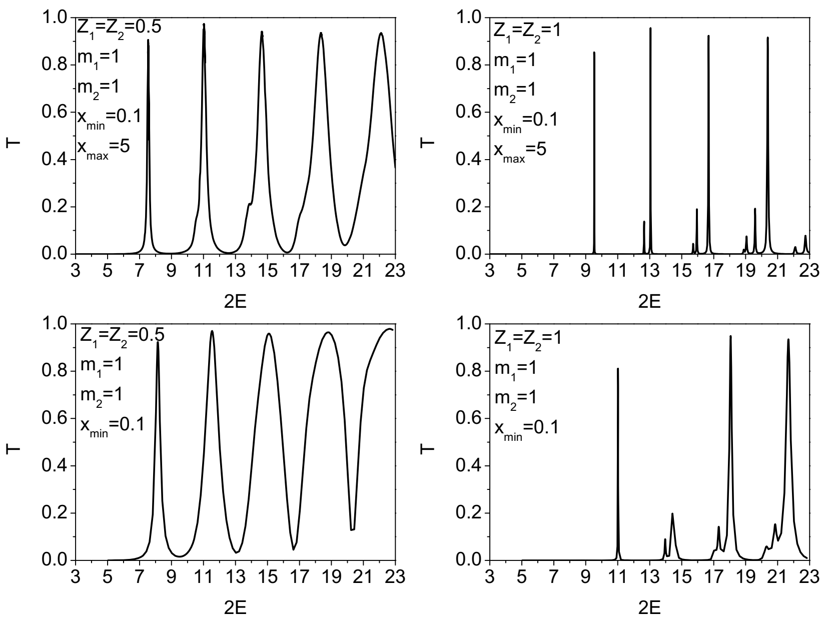


Fig. 1. The total probabilities $T \equiv T_{11} = \sum_{j=1}^{N_o} |T_{1j}|^2$ of penetration through Truncated Coulomb ([2]) at $x_{\max} = 5$ (upper panel), and Coulomb-like ([3]) (lower panel), potential barriers: $x_{\min} = 0.1$, $m_1 = m_2 = 1$, left panel: $\hat{Z}_1 = \hat{Z}_2 = 0.5$, right panel: $\hat{Z}_1 = \hat{Z}_2 = 1$.

5 Conclusion

The BVP for the 2D Schrödinger equation with long-range potentials from the 2D plane is reduced to sets of the BVPs for the ODEs in a finite 2D domain with help of the presented symbolic algorithms for evaluation of asymptotics of solutions and effective potentials of the ODEs. The BVPs for the resulting system of equations containing effective potentials, which are calculated by program ODPEVP [6], are solved by the new version of program KANTBP 2.0 using high-order precision approximations of the FEM [7]. The computational efficiency of the SNAs proposed is demonstrated by the benchmark calculation of quantum transmittance of long-range barriers for composite particles. The further development of the SNAs and software for solving the BVPs of the Schrödinger equation with long-range potentials can serve as a useful tool to study quantum transparency effects not only in heavy ion physics but also in quantum chemistry [11] and atomic physics [12].

Authors thank Profs. F.M. Pen'kov and P.M. Krassovitskiy for useful discussion. This work was done within the framework of the Protocols No. 4028-3-10/12 of collaboration between JINR and INP (Almaty) in dynamics of few-body systems and quantum transparency of barriers for structure particles and ions. The work was supported partially by RFBR (grants 10-01-00200 and 11-01-00523).

References

1. Hofmann, H.: Quantum mechanical treatment of the penetration through a two-dimensional fission barrier. Nucl. Phys. A 224, 116–139 (1974)
2. Hagino, K., Rowley, N., Kruppa, A.T.: A program for coupled-channel calculations with all order couplings for heavy-ion fusion reactions. Comput. Phys. Commun. 123, 143–152 (1999)
3. Pen'kov, F.M.: Metastable states of a coupled pair on a repulsive barrier. Phys. Rev. A 62, 044701-1-4 (2000)
4. Pen'kov, F.M.: Quantum Transmittance of Barriers for Composite Particles. JETP 91, 698–705 (2000)
5. Kantorovich, L.V., Krylov, V.I.: Approximate Methods of Higher Analysis. Wiley, New York (1964)
6. Chuluunbaatar, O., Gusev, A.A., Vinitsky, S.I., Abrashkevich, A.G.: ODPEVP: A program for computing eigenvalues and eigenfunctions and their first derivatives with respect to the parameter of the parametric self-adjoined Sturm–Liouville problem. Comput. Phys. Commun. 180, 1358–1375 (2009)
7. Chuluunbaatar, O., Gusev, A.A., Vinitsky, S.I., Abrashkevich, A.G.: KANTBP 2. 0: New version of a program for computing energy levels, reaction matrix and radial wave functions in the coupled-channel hyperspherical adiabatic approach. Comput. Phys. Commun. 179, 685–693 (2008)
8. Chuluunbaatar, O., Gusev, A., Gerdt, V., Kaschiev, M., Rostovtsev, V., Samoylov, V., Tupikova, T., Vinitsky, S.: A Symbolic-numerical algorithm for solving the eigenvalue problem for a hydrogen atom in the magnetic field: cylindrical coordinates. In: Ganzha, V.G., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2007. LNCS, vol. 4770, pp. 118–133. Springer, Heidelberg (2007)
9. Abramowitz, M., Stegun, I.A.: Handbook of Mathematical Functions. Dover, New York (1965)
10. Barnett, A.R., Feng, D.H., Steed, J.W., Goldfarb, L.J.B.: Coulomb wave functions for all real η and ρ. Comput. Phys. Comm. 8, 377–395 (1974)
11. Goodvin, G.L., Shegelski, M.R.A.: Three-dimensional tunneling of a diatomic molecule incident upon a potential barrier. Phys. Rev. A 72, 042713-1-7 (2005)
12. Giannakeas, P., Melezhik, V.S., Schmelcher, P.: D-wave confinement-induced resonances in harmonic waveguides. arXiv:1102.5686v1 (2011)

Symbolic-Numeric Investigation of the Aerodynamic Forces Influence on Satellite Dynamics

Sergey A. Gutnik

Moscow State Institute of International Relations (University) 76, Prospekt Vernadskogo, Moscow, 119454, Russia
s.gutnik@inno.mgimo.ru

Abstract. An approach for symbolic-numeric stability analysis of equilibrium orientations of a satellite in a circular orbit under the influence of gravitational and aerodynamic forces is considered. The stationary motions of a satellite are governed by a system of nonlinear algebraic equations. A computer algebra method based on an algorithm for the construction of a Groebner basis and the resultant concept is proposed for determining all equilibrium orientations of a satellite with a given aerodynamic torque and given principal central moments of inertia. It is shown that equilibrium orientations are determined by real solutions of algebraic equation of the twelfth degree. Evolution of domains with a fixed number of equilibria is investigated in detail. The stability analysis of equilibria is performed on the basis of Lyapunov theorem. The equilibrium orientations and their stability are analyzed numerically.

1 Introduction

Celestial mechanics is one of the most popular fields where symbolic computations are necessary to do very bulky calculations for solving many significant problems. In astrodynamics, successful application of computer algebra methods is a rare occasion in scientific papers. In this work, an example of symbolic–numeric investigation of satellite's dynamics under the influence of gravitational and aerodynamic torques is presented. It is a well known result that a satellite with different moments of inertia in the central Newtonian force field in a circular orbit has 24 equilibrium orientations [1]. However, at altitudes from 250 up to 500 km, rotational motion of a satellite is subjected to aerodynamic torque too. Therefore, it is necessary to study the joint action of gravitational and aerodynamic torques and, in particular, to analyze all possible satellite's equilibria in a circular orbit. Such solutions are used in practical space technology in the design of passive control systems of satellites.

This problem is considered in many papers. The basic problems of satellite's dynamics with an aerodynamic attitude control system have been presented in [1]. In [2], [3], and [4] all equilibrium orientations were found in some special cases, when the center of pressure is located on a satellite's principal central axis of inertia and on a satellite's principal central plane of inertia. The effect of the atmosphere on a satellite is reduced to the drag force applied to the center of pressure and directed against velocity of the satellite's center of mass relative to the air. The center of pressure is assumed to be at a fixed point in the satellite body.

In the present work, the problem of determining the classes of equilibrium orientations for general values of aerodynamic torque is considered. The equilibrium orientations are determined by real roots of the system of nonlinear algebraic equations. The investigation of equilibria was possible due to application of Computer Algebra Groebner basis and resultant methods. Evolution of domains with a fixed number of equilibria is investigated numerically in dependence of four dimensionless system parameters. Sufficient conditions for stability of all equilibrium orientations are obtained using generalized integral of energy.

2 Equations of Motion

Consider the motion of a satellite subjected to gravitational and aerodynamic torques in a circular orbit. We assume that 1) the gravity field of the Earth is central and Newtonian, 2) the satellite is a triaxial rigid body, 3) the effect of atmosphere on a satellite is reduced to the drag force applied at the center of pressure and directed against the velocity of the satellite's center of mass relative to the air, and the center of pressure is fixed in the satellite body. To write the equations of motion we introduce two right-handed Cartesian coordinate systems with origin in the satellite's center of mass O. $OXYZ$ is the orbital reference frame. The axis OZ is directed along the radius vector from the Earth center of mass to the satellite's center of mass, the axis OX is in the direction of a satellite's orbital motion. $Oxyz$ is the satellite's body reference frame; Ox, Oy, Oz are the principal central axes of inertia of a satellite. The orientation of the satellite's body reference frame $Oxyz$ with respect to the orbital reference frame is determined by means of the Euler angles ψ (precession), ϑ (nutation), and φ (spin). The direction cosines in transformation matrix between the frames $OXYZ$ and $Oxyz$ have the form:

$$\begin{aligned}
a_{11} &= \cos(x,X) = \cos\psi\cos\varphi - \sin\psi\cos\vartheta\sin\varphi,\\
a_{12} &= \cos(y,X) = -\cos\psi\sin\varphi - \sin\psi\cos\vartheta\cos\varphi,\\
a_{13} &= \cos(z,X) = \sin\psi\sin\vartheta,\\
a_{21} &= \cos(x,Y) = \sin\psi\cos\varphi + \cos\psi\cos\vartheta\sin\varphi,\\
a_{22} &= \cos(y,Y) = -\sin\psi\sin\varphi + \cos\psi\cos\vartheta\cos\varphi,\\
a_{23} &= \cos(z,Y) = -\cos\psi\sin\vartheta,\\
a_{31} &= \cos(x,Z) = \sin\vartheta\sin\varphi,\\
a_{32} &= \cos(y,Z) = \sin\vartheta\cos\varphi,\\
a_{33} &= \cos(z,Z) = \cos\vartheta.
\end{aligned} \tag{1}$$

Then equations of the satellite's attitude motion can be written in the Euler form [1], [2]:

$$\begin{aligned}
&A\dot{p} + (C-B)qr - 3\omega_0^2(C-B)a_{32}a_{33} = \tilde{h}_2 a_{13} - \tilde{h}_3 a_{12},\\
&B\dot{q} + (A-C)rp - 3\omega_0^2(A-C)a_{31}a_{33} = \tilde{h}_3 a_{11} - \tilde{h}_1 a_{13},\\
&C\dot{r} + (B-A)pq - 3\omega_0^2(B-A)a_{31}a_{32} = \tilde{h}_1 a_{13} - \tilde{h}_3 a_{11},
\end{aligned} \tag{2}$$

$$\begin{aligned}
p &= \dot{\psi} a_{31} + \dot{\vartheta}\cos\varphi + \omega_0 a_{21},\\
q &= \dot{\psi} a_{32} + \dot{\vartheta}\sin\varphi + \omega_0 a_{22},\\
r &= \dot{\psi} a_{33} + \dot{\vartheta} + \omega_0 a_{23}.
\end{aligned} \tag{3}$$

Here p, q, r are the projections of the satellite's angular velocity onto the axes Ox, Oy, Oz; A, B, C are the principal central moments of inertia of the satellite; ω_0 is the angular velocity of the orbital motion of the satellite's center of mass. $\tilde{h}_1 = -a_p Q$, $\tilde{h}_2 = -b_p Q$, $\tilde{h}_3 = -c_p Q$, Q is the atmospheric drug force acting on a satellite; a_p, b_p, c_p are the coordinates of the center of pressure of a satellite in the reference frame $Oxyz$. The dot designates differentiation with respect to time t.

Equations (2) along with (3) form a closed system of equations of motion of the satellite, for which the Jacobi Integral is valid

$$\begin{aligned}
H = {}&\frac{1}{2}(A\bar{p}^2 + B\bar{q}^2 + C\bar{r}^2) + \frac{3}{2}\omega_0^2[(A-C)a_{31}^2 + (B-C)a_{32}^2] +\\
&+ \frac{1}{2}\omega_0^2[(B-A)a_{21}^2 + (B-C)a_{23}^2] - (\tilde{h}_1 a_{11} + \tilde{h}_2 a_{12} + \tilde{h}_3 a_{13}),
\end{aligned} \tag{4}$$

where $\bar{p} = p - \omega_0 a_{21}$, $\bar{q} = q - \omega_0 a_{22}$, $\bar{r} = r - \omega_0 a_{23}$.

3 Equilibrium Orientations of a Satellite

Putting in (2) and (3) $\psi = \text{const}$, $\vartheta = \text{const}$, $\varphi = \text{const}$ and introducing the notation $\tilde{h}_i = \omega_0^2 \bar{h}_i (i = 1, 2, 3)$, we obtain the equations

$$\begin{aligned}
(C-B)(a_{22}a_{23} - 3a_{32}a_{33}) &= \bar{h}_2 a_{13} - \bar{h}_3 a_{12},\\
(A-C)(a_{21}a_{23} - 3a_{31}a_{33}) &= \bar{h}_3 a_{11} - \bar{h}_1 a_{13},\\
(B-A)(a_{21}a_{22} - 3a_{31}a_{32}) &= \bar{h}_1 a_{12} - \bar{h}_2 a_{11},
\end{aligned} \tag{5}$$

which allow us to determine the satellite's equilibria in the orbital reference frame.

Let $A \neq B \neq C$. Substituting the expressions for the direction cosines from (1) in terms of Euler angels into Eqs. (5), we obtain three equations with three unknowns ψ, ϑ, φ. The second procedure for closing Eqs. (5) is to add the following six orthogonality conditions for the direction cosines:

$$a_{i1}a_{j1} + a_{i2}a_{j2} + a_{i3}a_{j3} = \delta_{ij} \tag{6}$$

where δ_{ij} is the Kronecker delta and $(i, j = 1, 2, 3)$. Equations (5) and (6) form a closed system with respect to the direction cosines, which also specifies the equilibrium solutions of a satellite. We state the following problem for the system of equations (5), (6): determine all nine direction cosines, i.e., to find all the equilibrium orientations of the satellite when A, B, C, $\bar{h}_1$, $\bar{h}_2$, and $\bar{h}_3$ are given. The problem has been solved only for some specific cases when the center of pressure is located on a satellite's principal central axis of inertia Ox, when $\bar{h}_1 \neq 0$, $\bar{h}_2 = \bar{h}_3 = 0$ [2], [3] and when the pressure center locates in the satellite's principal central plane of inertia Oxz of the frame $Oxyz$ and $\bar{h}_1 \neq 0$, $\bar{h}_2 = 0$, $\bar{h}_3 \neq 0$ [4]. In the case $\bar{h}_1 = \bar{h}_2 = \bar{h}_3 = 0$, it has been proved that the system (5), (6) has 24 solutions describing the equilibrium orientations of a satellite-rigid body [1].

Here we consider the general case of the problem of defining the equilibria of a satellite when $\bar{h}_1 \neq 0$, $\bar{h}_2 \neq 0$, $\bar{h}_3 \neq 0$. A Computer Algebra approach to define all the equilibrium orientations of a satellite will be used. Projecting Eqs. (5) onto the axis of the orbiting frame $OXYZ$, we get the algebraic system, using the method given in [5]

$$\begin{aligned} Aa_{21}a_{31} + Ba_{22}a_{32} + Ca_{23}a_{33} &= 0, \\ Aa_{11}a_{21} + Ba_{12}a_{22} + Ca_{13}a_{23} - (\bar{h}_1 a_{21} + \bar{h}_2 a_{22} + \bar{h}_3 a_{23}) &= 0, \\ 3(Aa_{11}a_{31} + Ba_{12}a_{32} + Ca_{13}a_{33}) + \bar{h}_1 a_{31} + \bar{h}_2 a_{32} + \bar{h}_3 a_{33} &= 0. \end{aligned} \tag{7}$$

A solution of the system (6), (7) can be obtained using an algorithm for the construction of Groebner bases [6]. The method of Groebner bases is used to solve systems of nonlinear algebraic equations. It comprises an algorithmic procedure for reducing the problem involving polynomials of several variables to investigation of a polynomial of one variable. Using the computer algebra system Maple [7] Groebner[gbasis] package with *tdeg* option, we calculate the Groebner basis of the system (6), (7) of nine polynomials with nine variables a_{ij} $(i, j = 1, 2, 3)$ under the ordering on the total power of the variables. In the list of variables in the Maple Groebner package we use nine direction cosines, and in the list of polynomials, we include the polynomials from the left-hand sides f_i $(i = 1, 2, ...9)$ of the algebraic equations (6), (7):

map(factor,Groebner[gbasis]([f1,f2,f3, ... f9],tdeg(a11, a12, a13, ... a33))). Here we write down the polynomials in the Groebner basis that depend only on the variables a_{31}, a_{32}, a_{33}

$$\begin{aligned} &9[(B-C)^2 a_{32}^2 a_{33}^2 + (C-A)^2 a_{31}^2 a_{33}^2 + (A-B)^2 a_{31}^2 a_{32}^2] = \\ &\quad = (\bar{h}_1 a_{31} + \bar{h}_2 a_{32} + \bar{h}_3 a_{33})^2 (a_{31}^2 + a_{32}^2 + a_{33}^2), \\ &3(B-C)(C-A)(A-B)a_{31}a_{32}a_{33} - [\bar{h}_1 (B-C) a_{32} a_{33} + \\ &+ \bar{h}_2 (C-A) a_{31} a_{33} + \bar{h}_3 (A-B) a_{31} a_{32}](\bar{h}_1 a_{31} + \bar{h}_2 a_{32} + \bar{h}_3 a_{33}) = 0, \\ &a_{31}^2 + a_{32}^2 + a_{33}^2 = 1. \end{aligned} \tag{8}$$

Introducing the new variables $x = a_{31}/a_{32}$, $y = a_{33}/a_{32}$, $h_i = \bar{h}_i/(B-C)$, $\nu = (B-A)/(B-C)$, we deduce two equations for determining of x and y.

$$\begin{aligned} a_0y^2 + a_1y + a_2 = 0, \\ b_0y^4 + b_1y^3 + b_2y^2 + b_3y + b_4 = 0, \end{aligned} \tag{9}$$

where

$$\begin{aligned} a_0 &= h_3(h_2(1-\nu)x - h_1), \\ a_1 &= \nu(3(1-\nu) + h_3^2)x + (h_1x + h_2)(h_2(1-\nu)x - h_1), \\ a_2 &= \nu h_3(h_1x + h_2)x, \\ b_0 &= h_3^2, \\ b_1 &= 2h_3(h_1x + h_2), \\ b_2 &= (h_1x + h_2)^2 + h_3^2(1 + x^2) - 9 - 9(1-v)^2x^2, \\ b_3 &= 2h_3(h_1x + h_2)(1 + x^2), \\ b_4 &= (h_1x + h_2)^2(1 + x^2) - 9\nu^2x^2. \end{aligned}$$

Invoking the resultant concept we eliminate the variable y from the equations (9). Expanding the determinant of the resultant matrix of Eqs.(9), with the help of Maple symbolic matrix function, we obtain a twelfth degree algebraic equation in x:

$$\begin{aligned} p_0x^{12} + p_1x^{11} + p_2x^{10} + p_3x^9 + p_4x^8 + p_5x^7 + \\ + p_6x^6 + p_7x^5 + p_8x^4 + p_9x^3 + p_{10}x^2 + p_{11}x + p_{12} = 0, \end{aligned} \tag{10}$$

the coefficients of which depend in a rather complicated way on the parameters ν, h_1, h_2, h_3

$$\begin{aligned} p_0 = (1-\nu)^6p_{12}, \qquad p_1 = -(1-\nu)^5p_{11}, \qquad \dots \\ p_{11} = 2h_1^3h_2^3(2(1-\nu)h_2^2 - 2h_1^2 - \nu h_3^2 - 3\nu(1-\nu)), \quad p_{12} = -h_1^4h_2^4. \end{aligned} \tag{11}$$

By the definition of the resultant, to every root x of Eq.(10) there corresponds a common root y of the system (9). It can easily be shown that to every real root x of Eq.(10) there correspond 2 solutions for (5), (6). Since the number of real roots of Eq.(10) does not exceed 12, the satellite in a circular orbit can have at most 24 equilibria in the orbiting reference frame.

Using Eq.(10), (11) we can determine numerically all the relative equilibrium orientations of the satellite and analyze their stability. We have analyzed numerically dependence of the number of real solutions of Eq.(10) on the parameters, using factorization method. For a fixed values of ν and h_3, the number of real roots was determined at the nodes of a uniform grid in the plane (h_1, h_2). We have used the values of $\nu = 0.2$, $\nu = 0.4$, $\nu = 0.6$, $\nu = 0.8$ $(|\nu| < 1)$.

In the present work, we have implemented the bifurcation values of the parameters h_1 and h_3, corresponding to the qualitative change of domains with a fixed number of equilibria, which were defined in [4] for the special case when

$\bar{h}_1 \neq 0$, $\bar{h}_2 = 0$, $\bar{h}_3 \neq 0$. In [4] all the equilibrium solutions are determined by real roots of the algebraic equations of fourth degree and bifurcation values of parameters h_1 and h_3 when the number of real roots changes were found analytically: $|h_1| = 1$, $|h_3| = 1$, $|h_1| = 3$, $|h_3| = 3$, $|h_1| = 6$, $|h_3| = 6$. For this special case in the intervals $|h_1| < 1$, $|h_3| < 1$ – 24, 20, and 16 equilibria exist; in the next intervals $1 < |h_1| < 3$, $1 < |h_3| < 3$ – 16, 12, and 8 equilibria exist and in the intervals $|h_1| > 6$, $|h_3| > 6$ only 8 equilibria exist. We have used these bifurcation values of h_1 and h_3 when $h_2 = 0$ for our numerical calculations. For the first interval when $|h_3| < 1$ we define numerically the evolution of domains with 24, 20, and 16 equilibria. We have used a small step of the parameter h_3 ($h_3 = 0.1, 0.15, 0.25, 0.35, 0.5, 0.75, 0.8, 0.9$) because for $|h_3| < 1$ there are small domains with a fixed number of real roots of Eq.(10). For example, at $h_3 = 0.1$ ($\nu = 0.2$) when the parameter values $|h_1| < 0.2$ and $|h_2| < 0.2$ there is domain of existence of 24 equlibria (12 real roots). For the intervals $0.2 < |h_1| < 0.5$ and $0.2 < |h_2| < 0.6$ – 20 equlibria exist (10 real roots). For the next intervals $0.5 < |h_1| < 0.7$ and $0.6 < |h_2| < 0.7$, there is domain of existence of 16 equlibria (8 real roots). For $0.7 < |h_1| < 2$ and $0.7 < |h_2| < 2.5$ – 12 equlibria exist (6 real roots), and for $|h_1| > 2$ and $|h_2| > 2.5$ only 8 equlibria exist (4 real roots). At $h_3 = 0.25$ ($\nu = 0.2$) when the parameter values $|h_1| < 0.2$ and $|h_2| < 0.2$ there is domain of existence of 24 equlibria. For the intervals $0.2 < |h_1| < 0.3$ and $0.2 < |h_2| < 0.45$ – 20 equlibria exist. For the next intervals $0.3 < |h_1| < 0.8$ and $0.45 < |h_2| < 0.6$ there is domain of existence of 16 equlibria. For $0.8 < |h_1| < 1.6$ and $0.6 < |h_2| < 2.2$ – 12 equlibria exist, and for $|h_1| > 1.6$ and $|h_2| > 2.2$ only 8 equlibria exist. Analysis of the numerical results for $|h_3| < 1$ shows that five domains with the 24, 20, 16, 12, and 8 equilibria exist in the plane (h_1, h_2) for the intervals $0 < |h_3| \leq 0.8$. When we cross the bifurcation value $h_3 = 0.8$ domain with the 24 equilibria vanish and in the intervals $0.8 < |h_3| < 1$ only four domains with 20, 16, 12, and 8 equilibria exist. The value $|h_3| = 1$ is also bifurcation as in the special case. When we cross the bifurcation value $h_3 = 1$ the domain with 20 equilibria vanishes. In the interval $1 < |h_3| < 3$, only three domains with the 16, 12, and 8 equilibria exist. The value $|h_3| = 3$ is bifurcation, as in the special case. When we cross the bifurcation value $h_3 = 3$ the domain with 16 equilibria vanishes. In the interval $3 < |h_3| < 6$, only two domains with 12 and 8 equilibria exist. When the values of parameter $|h_3|$ of the aerodynamic torque are more than 6, the satellite has only 8 equilibrium orientations, which correspond to four real roots of Eq.(10).

4 Stability Analysis of Equilibria

To investigate the stability of equilibrium solutions $\psi = \psi_0 =$ const, $\vartheta = \vartheta_0 =$ const, $\varphi = \varphi_0 =$ const satisfying Equations (5), we can use the the Jacobi Integral of energy (4) as the Lyapunov function. After replacement $\psi \to \psi + \psi_0$, $\vartheta \to \vartheta + \vartheta_0$, $\varphi \to \varphi + \varphi_0$ where ψ, ϑ, φ are small deviations from the satellite's equilibrium ψ_0, ϑ_0, φ_0, the energy integral takes the form

$$H = \frac{1}{2}(A\bar{p}^2 + B\bar{q}^2 + C\bar{r}^2) + \frac{1}{2}(B - C)(A_{11}\psi^2 + A_{22}\vartheta^2 + A_{33}\varphi^2 + \\ + 2A_{12}\psi\vartheta + 2A_{13}\psi\varphi + 2A_{23}\vartheta\varphi) + O_3(\psi, \vartheta, \varphi) = \text{const}, \quad (12)$$

where coefficients A_{ij} depend on the parameters $\nu, h_1, h_2, h_3, \psi, \vartheta, \varphi$ in the form

$$\begin{aligned}
A_{11} &= \nu(a_{11}^2 - a_{21}^2) + (a_{13}^2 - a_{23}^2) + h_1 a_{11} + h_2 a_{12} + h_3 a_{13}, \\
A_{22} &= (3 + \cos^2\psi_0)(1 - \nu \sin^2\varphi_0)\cos 2\vartheta_0 - \frac{1}{4}\nu \sin 2\psi_0 \cos\vartheta_0 \sin 2\varphi_0 - \\
&\quad - (h_1 \cos\vartheta_0 \sin\varphi_0 + h_2 \cos\vartheta_0 \cos\varphi_0 - h_3 \sin\vartheta_0)\sin\psi_0, \\
A_{33} &= \nu((a_{22}^2 - a_{21}^2) - 3(a_{32}^2 - a_{31}^2)) + h_1 a_{11} + h_2 a_{12}, \qquad (13) \\
A_{12} &= -\frac{1}{2}\sin 2\psi_0 \sin 2\vartheta_0 + \nu(a_{11}a_{23} + a_{13}a_{21})\sin\varphi_0 - \\
&\quad - (h_1 a_{31} + h_2 a_{32} + h_3 a_{33})\cos\psi_0, \\
A_{13} &= \nu(a_{12}a_{22} + a_{12}a_{21}) + h_1 a_{22} - h_2 a_{21}, \\
A_{23} &= -\frac{3}{2}\nu \sin 2\vartheta_0 \sin 2\varphi_0 + \nu(a_{21}\cos\varphi_0 + a_{22}\sin\varphi_0)a_{23} - \\
&\quad - (h_1 \cos\varphi_0 - h_2 \sin\varphi_0)a_{13}.
\end{aligned}$$

It follows from Lyapunov theorem that the equilibrium solution is stable if the quadratic form (12),(13) is positive definite, i.e., the following inequalities take place:

$$\begin{aligned}
&A_{11} > 0, \qquad B > C, \\
&A_{11}\, A_{22} - A_{12}^2 > 0, \qquad (14) \\
&A_{11}\, A_{22}A_{33} + 2A_{12}A_{23}A_{13} - A_{11}A_{23}^2 - A_{22}A_{13}^2 - A_{33}A_{12}^2 > 0.
\end{aligned}$$

Substituting the expressions for A_{ij} from (13) for the corresponding equilibrium solution into (14), we obtain the conditions for stability of this solution. Using integral (12),(13), we have analyzed numerically stability conditions (14) for the equilibrium solutions. Analysis of the numerical results shows that stable equilibrium orientations of a satellite exist even for large aerodynamic torque when $|h_i| \geq 6.0$ $(i = 1, 2, 3)$. For such values of aerodynamic torque only eight equilibria exist, and two of them are stable. At $0 < |h_i| < 6.0$ $(i = 1, 2, 3)$ both four and two stable equilibria exist.

5 Conclusion

In this work, the attitude motion of a satellite under the action of gravitational and aerodynamic torques in a circular orbit has been investigated. The main attention was given to determination of a satellite equilibrium orientation in the orbital reference frame and to analysis of their stability. The symbolic-numeric method of determination of all the satellite equilibria is suggested in general case ($\bar{h}_1 \neq 0$, $\bar{h}_2 \neq 0$, $\bar{h}_3 \neq 0$). The symbolic computation system Maple is applied to reduce the satellite stationary motion system of nine algebraic equations with nine variables to a single algebraic equation of the twelfth degree with one variable, using the Groebner package for the construction of a Groebner basis and the resultant approach. It was shown that the equilibrium orientations are determined by real roots of single algebraic equation of the twelfth degree. Using this result of symbolic calculations we conclude that the satellite subjected to gravitational and aerodynamic torques can have no more than 24 equilibrium orientations in a circular orbit. The evolution of domains with a fixed number of equilibrium orientations was investigated numerically in the plane of two parameters (h_1, h_2) at a different values of parameters ν and h_3. Some general bifurcation values of h_3 corresponding to the qualitative change of domains with a fixed number of equilibria were determined. On the basis of numerical calculation, we can conclude that the number of satellite's isolated equilibria is no less than 8. Using the Lyapunov theorem, the sufficient conditions of stability of the equilibrium orientations are investigated numerically at different values of aerodynamic parameters. Analysis of the numerical simulation shows that the number of stable equilibria is no less than two. All the calculations considered here were implemented with the computer algebra system Maple. The results of the study can be used at the stage of preliminary design of the satellite with aerodynamic control system.

References

1. Sarychev, V.A.: Problems of orientation of satellites. Itogi Nauki i Tekhniki. Ser. "Space Research", vol. 11. VINITI, Moscow (1978)
2. Sarychev, V.A., Mirer, S.A.: Relative equilibria of a satellite subjected to gravitational and aerodynamic torques. Cele. Mech. Dyn. Astron. 76(1), 55–68 (2000)
3. Sarychev, V.A., Mirer, S.A., Degtyarev, A.A., Duarte, E.K.: Investigation of equilibria of a satellite subjected to gravitational and aerodynamic torques. Cele. Mech. Dyn. Astron. 97, 267–287 (2007)
4. Sarychev, V.A., Mirer, S.A., Degtyarev, A.A.: Equilibria of a satellite subjected to gravitational and aerodynamic torques with pressure center in a principal plane of inertia. Cele. Mech. Dyn. Astron. 100, 301–318 (2008)
5. Sarychev, V.A., Gutnik, S.A.: Relative equilibria of a gyrostat satellite. Cosmic Research 22, 323–326 (1984)
6. Buchberger, B.: A theoretical basis for the reduction of polynomials to canonical forms. SIGSAM Bulletin 10(3), 19–29 (1976)
7. Char, B.W., Geddes, K.O., Gonnet, G.H., Monagan, M.B., Watt, S.M.: Maple Reference Manual. Watcom Publications Limited, Waterloo (1992)

Practical Divide-and-Conquer Algorithms for Polynomial Arithmetic

William Hart[1,⋆] and Andrew Novocin[2,⋆⋆]

[1] University of Warwick, Mathematics Institute, Coventry CV4 7AL, UK
W.B.Hart@warwick.ac.uk
http://maths.warwick.ac.uk/~masfaw

[2] LIP/INRIA/ENS, 46 allée d'Italie, F-69364 Lyon Cedex 07, France
Andrew.Novocin@ens-lyon.fr
http://andy.novocin.com/pro

Abstract. We investigate two practical divide-and-conquer style algorithms for univariate polynomial arithmetic. First we revisit an algorithm originally described by Brent and Kung for composition of power series, showing that it can be applied practically to composition of polynomials in $\mathbb{Z}[x]$ given in the standard monomial basis. We offer a complexity analysis, showing that it is asymptotically fast, avoiding coefficient explosion in $\mathbb{Z}[x]$. Secondly we provide an improvement to Mulders' polynomial division algorithm. We show that it is particularly efficient compared with the multimodular algorithm. The algorithms are straightforward to implement and available in the open source FLINT C library. We offer a practical comparison of our implementations with various computer algebra systems.

Introduction

Univariate integer polynomials are important basic objects for computer algebra systems. In this paper we investigate two algorithms for univariate polynomial arithmetic over $\mathbb{Z}$. In particular, we study divide-and-conquer style algorithms for composition and division of polynomials.

Given two polynomials $f, g \in \mathbb{Z}[x]$ the polynomial *composition problem* is to compute $f(g(x)) \in \mathbb{Z}[x]$. Standard approaches include Horner's method [10], ranged Horner's method (which we describe in section 1.1), algorithms for composition of polynomials in a Bernstein basis (see [2]), and algorithms based on point evaluation followed by coefficient interpolation (see [13]).

Given $f, g \in \mathbb{Z}[x]$ the *division problem* is to find polynomials $q, r \in \mathbb{Z}[x]$ such that $f = gq + r$ where the $\deg(r) <= \deg(f)$, but the coefficients of terms of r whose degree is at least $\deg(g)$ are reduced modulo the leading coefficient of g.

Standard approaches to the division problem are the naive $\mathcal{O}(n^2)$ "schoolbook" method, a divide-and-conquer approach based on the middle product and

⋆ Author was supported by EPSRC Grant number EP/G004870/1.
⋆⋆ Author was partially supported by ANR project LaRedA.

a multimodular approach (see for example Victor Shoup's NTL [14] which employs the latter approach). The approach we present has the advantage of being simpler to implement than the middle product approach with comparable performance but much better performance than the "school-book" or multimodular approaches in real-world cases.

Some important applications include: i) exact division, i.e. where $r = 0$, ii) division by g with leading coefficient ± 1, iii) divisibility testing, i.e. to test if $f = gq$ for some $q \in \mathbb{Z}[x]$, with the algorithm returning false if (and as soon as) a non-zero coefficient is detected in the remainder r and iv) a basecase for power series division (with normalised divisor).

1 Polynomial Composition

We begin with a divide-and-conquer approach to polynomial composition.

Our Contribution. We present and analyze the divide-and-conquer technique of Brent and Kung [5], originally a component of a power series composition algorithm, applied instead to the composition of two polynomials $f, g \in \mathbb{Z}[x]$ given in the standard monomial basis. We give a theoretical complexity bound which is softly optimal in the size of the output and show that the algorithm is highly practical.

Problem Statement:

Given: $f = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0$ and $g = b_m x^m + \cdots + b_0$ in $\mathbb{Z}[x]$.
Find: a full expansion of $h = f(g(x))$

Assumptions: In our analysis we assume the use of fast arithmetic (see [1]), which is available in FLINT [9]. Also, only for the simplicity of bit-complexity analysis, we will assume throughout that coefficients of f and g are of $\mathcal{O}(m)$ bits, where m is the degree of g, the inner polynomial in the composition $f(g)$. We note that the algorithm still works when the coefficients are larger, but depending on the implementation of the fast polynomial arithmetic, the bit complexity will go up by some factor which is a quasilinear expression in the size of the coefficients.

The algorithm is simple to implement and works in the standard monomial basis. We will show that the algorithm performs well in practice by providing timings against the MAGMA computer algebra system [6]. We also provide a theoretical complexity analysis showing that, in the worst case, the algorithm uses $\mathcal{O}(nm \log(n) \log(nm))$ operations in $\mathbb{Z}$ and has a bit-complexity of $\mathcal{O}(n^2 m^2 \log(nm))$.

Assuming that $h = f(g)$ does not have special structure (i.e. h is dense with few cancellations) then this output has $\mathcal{O}(nm)$ coefficients each with bit-length $\mathcal{O}(nm)$. Simply writing down the output requires $\mathcal{O}(n^2 m^2)$ CPU-operations making our theoretical bound optimal, up to a factor $\mathcal{O}(\log(mn))$.

Related Works. The presented algorithm is an application of the divide-and-conquer technique of Brent and Kung [5], originally developed as a component of an algorithm for composition of power series. In the original application the bit complexity was not considered, however we show that the algorithm is asymptotically fast for polynomial composition in $\mathbb{Z}[x]$. The algorithm was rediscovered while implementing the number theory library FLINT [9], and we are grateful to Joris van der Hoeven for pointing out its first occurrence in the literature.

In [11] an algorithm is presented which is asymptotically fast for composition of polynomials in a Bernstein basis. However for polynomials presented in the usual monomial basis one must first perform a conversion to Bernstein basis to make use of this algorithm.

Conversion of orthogonal polynomials can be done in time $\mathcal{O}(n \log^2 n \log\log n)$, assuming the use of Fast Fourier Transform techniques (see [3]), however Bernstein bases are not orthogonal.

A standard method for converting from a Bernstein basis to a monomial basis involves computing a difference table, which costs $\mathcal{O}(n^2)$ operations for a polynomial of length n (degree $n-1$) in the Bernstein basis (see [4, Sect.2.8]). Thus to convert the eventual solution from Bernstein basis to monomial basis in our case will cost $\mathcal{O}((mn)^2)$ operations, each of which involves a subtraction of quantities of $\mathcal{O}(mn)$ bits. Thus the total bit complexity of the conversion alone is already significantly greater than that of our algorithm.

A different method is given in [13, Prob 3.4.2]. In this method, $K = 2^k$ is computed such that $mn + 1 \le K < 2mn + 2$. If possible compute ω, a primitive K^{th} root of unity, and the $K = 2^k$ points, ω^i for all $i = 0, \ldots, K-1$. Evaluate $h = f(g)$ at those K points (using fast arithmetic) and interpolate the coefficients of h. If a K^{th} root of unity is unavailable then use K other values for evaluation. Pan suggests that this method uses $\mathcal{O}(nm[\log(n) + \log(m) + \log^2(n)])$ operations in $\mathbb{Z}$ when roots of unity are available and $\mathcal{O}(nm[\log^2(nm)])$ operations in $\mathbb{Z}$ otherwise.

In order to apply Pan's method to polynomials in $\mathbb{Z}[x]$ one may work in a ring $\mathbb{Z}/p\mathbb{Z}$ where $p = 2^{2K} + 1$. There are then sufficiently many roots of unity, and moreover, the coefficients of $f(g(x))$ may be identified by their values $\pmod p$.

Interpolation of h is performed using the inverse FFT. To evaluate $f(g(x))$ at the roots of unity, Pan first evaluates $g(x)$ at the roots of unity using the FFT. This gives K values at which $f(x)$ must then be evaluated.

The Moenck-Borodin algorithm (see Algorithm 3.1.5 of [13]) evaluates $f(x)$ of degree n at n arbitrary points in $\mathcal{O}(n \log^2 n)$ operations. If the points are $w_1, w_2, \ldots, w_n$, one first reduces $f(x) \bmod (x - w_1)(x - w_2)\cdots(x - w_n)$. One then splits this product into two balanced halves and reduces mod each half separately. This process is repeated recursively until one has the reduction of $f(x)$ modulo each of the factors $(x - w_i)$.

Of the $\mathcal{O}(n \log^2 n)$ operations there are $\mathcal{O}(n \log n)$ multiplications. Each can be performed in our case using fast arithmetic in $\mathcal{O}(mn \log mn)$ bit operations (up to higher order log factors).

As we have $\mathcal{O}(mn)$ roots of unity to evaluate at, not n, we must perform this whole operation $\mathcal{O}(m)$ times. Thus the bit complexity of Pan's algorithm is $\mathcal{O}((mn)^2 \log n \log mn)$, which exceeds that of our algorithm by a factor of $\log n$.

Road Map. In section 1.1 we present Horner's method and Ranged Horner's method along with a complexity analysis. In section 1.2 we present the algorithm itself. In subsection 1.2 we provide a worst-case asymptotic complexity analysis. Finally, in section 1.3 we provide practical timings of our FLINT implementation and a comparison with MAGMA's polynomial composition algorithm.

Notations and Notes: Given two polynomials of length n, with coefficients of n bits, the Schönhage and Strassen Algorithm (SSA) for multiplying polynomials has a bit complexity of $\mathcal{O}(n^2 \log(n) \log\log n)$ (for more see [7, Sect.8.3]). We will ignore $\log\log n$ factors throughout the paper. Various standard tricks allow us to multiply polynomials of degree n with coefficients of m bits in time $\mathcal{O}(mn \log(mn))$ using SSA (again ignoring lower order log factors). For each algorithm we given both the bit-complexity model cost and the number of operations in $\mathbb{Z}$.

1.1 Horner's Method

In this section we apply Horner's algorithm for evaluating a polynomial f at a point p, to the problem of polynomial composition.

Horner's Evaluation Algorithm
Given: $f = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0$ in $\mathbb{Z}[x]$, p in $\mathbb{Z}$.
Find: $ans := f(p)$ in $\mathbb{Z}$

```
1. $ans := a_n$
2. For $i = n-1$ down to $i = 0$ do:
   (a) $ans := ans \cdot p + a_i$
3. Return $ans$
```

This algorithm computes $a_n p^n + a_{n-1}p^{n-1} + \cdots + a_0$ using n multiplications and n additions. When the point p is a polynomial g, n polynomial multiplications and n polynomial additions are performed.

Ranged Horner Composition. We will need a variant of this approach which we call Ranged Horner's algorithm for polynomial composition. We restrict the algorithm to use only l coefficients of f, from a_i to $a_{i+\ell-1}$, and replace p by a polynomial g. If one chooses $i = 0$ and $\ell = n+1$ then this algorithm returns a complete expansion of $h = f(g)$. The algorithm is always a direct application of Horner's method to the degree $\ell-1$ polynomial $F := a_{i+\ell-1}x^{\ell-1} + \cdots + a_{i+1}x + a_i$.

Algorithm 1. ***Ranged Horner Compose***

Input: *$f, g \in \mathbb{Z}[x]$, i, a starting index, and ℓ the length of the ranged composition.*

Output: *An expansion of $F(g) := a_{i+\ell-1}g^{\ell-1} + \cdots + a_{i+1}g + a_i$, where F is f divided by x^i without remainder then reduced modulo x^ℓ, a shifted truncation of f.*

```
1. $ans := a_{i+\ell-1}$
2. For $j = \ell - 2$ down to $j = 0$ do:
   (a) $ans := ans \cdot g$
   (b) $ans := ans + a_{i+j}$
3. Return $ans$
```

Bit-Complexity. We will now outline the bit-complexity analysis of Ranged Horner Composition.

Theorem 1. *Algorithm 1 terminates after $\mathcal{O}(\ell^2 m \log(\ell m))$ operations in $\mathbb{Z}$ with a bit-complexity bound of $\mathcal{O}(\ell^3 m^2 \log(\ell m))$ CPU operations.*

Proof. Let us analyze the cost of the k^{th} loop where $k = 1, \ldots, \ell - 1$. First we compute the degree and coefficient size of ans in the k^{th} loop.

Lemma 1. *At the beginning of the k^{th} loop of step 2 in Algorithm 1 we have the degree of $ans = (k-1)m$ and $\| ans \|_\infty \leq 2^{\mathcal{O}(km+(k-1)\log(m+1))}$.*

Proof. The degree of ans begins at 0 and increases by m in each loop giving degree $(k-1)m$ at the beginning of the k^{th} loop.

Now for an arbitrary loop let's suppose that $\| ans \|_\infty \leq 2^x$ and $ans = c_N x^N + \cdots + c_0$ where N is the current degree of ans. Recall that $g = b_m x^m + \cdots b_0$ and $\| g \|_\infty \leq 2^m$. The product $ans \cdot g$ can be written as

$$\sum_{s=0}^{s=N+m} x^s [\sum_{\{0 \leq i \leq N, 0 \leq j \leq m | s=i+j\}} (c_i \cdot b_j)].$$

In this form it can be seen that the largest coefficients of $ans \cdot g$ are the sum of $m+1$ numbers of norm $\leq 2^{m+x}$. Thus after this loop the coefficients are boundable by $2^{x+m+\log_2(m+1)}$. So the size of the coefficients of ans begin at m-bits and increase by $m + \log_2(m+1)$ finishing the proof of the lemma.

Now using fast polynomial multiplication the bit complexity of loop k is $\mathcal{O}(k^2 m(m + \log(m)\log(km))$ and uses $\mathcal{O}(km\log(km))$ operations in $\mathbb{Z}$. Summing this over $k = 1, \ldots, \ell - 1$ gives a bit-complexity of $\mathcal{O}(\ell^3 m^2 \log(\ell m))$ and $\mathcal{O}(\ell^2 m \log(\ell m))$ operations in $\mathbb{Z}$.

1.2 Divide and Conquer Algorithm

In this section we describe the main algorithm for polynomial composition. First we divide f of degree n into $k_1 := \lceil (n+1)/\ell \rceil$ sub-polynomials of length ℓ for some experimentally derived (and small) value of ℓ such that:

$$f := f_0 + f_1 \cdot x^{\ell} + f_2 \cdot x^{2\ell} + \cdots + f_{k_1-1} \cdot x^{(k_1-1)\ell}.$$

In the first iteration of the algorithm we compute the k_1 compositions, $h_{1,i} := f_i(g)$ for $0 \le i < k_i$ using (Ranged) Horner's method and we also compute g^{ℓ}. In the i^{th} iteration we start with $g^{2^{i-2}\ell}$ and compute the $k_i := \lceil (k_{i-1})/2 \rceil$ polynomials: $h_{i,j} := h_{i-1,2j} + g^{2^{i-2}\ell} \cdot h_{i-1,2j+1}$ then compute $g^{2^{i-1}\ell}$. Thus in iteration i our target polynomial $h = f(g)$ can be written:

$$h_{i,0} + h_{i,1} \cdot (g^{2^{i-1}\ell}) + h_{i,2} \cdot (g^{2^{i-1}\ell})^2 + \cdots + h_{i,k_i-1} \cdot (g^{2^{i-1}\ell})^{k_i-1}.$$

In each iteration the number of polynomials is halved while the length of the polynomials we work with is doubled. We experimentally determined that a value of $\ell = 4$ works well in practice.

Algorithm 2. ***Polynomial Composition Algorithm***
Input: $f, g \in \mathbb{Z}[x]$
Output: *An expansion of* $h := f(g)$

```
1. let $\ell := 4$, $i := 1$, and $k_i := \lceil \frac{n+1}{\ell} \rceil$
2. for $j = 0, \ldots, k_i - 1$
   (a) compute $h_{i,j} :=$ Algorithm 1$(f, g, j\ell, \ell)$
3. compute $G := g^{\ell}$.
4. while $(k_i > 1)$ do:
   (a) $k_{i+1} := \lceil k_i/2 \rceil$;
   (b) for $j = 0, \ldots, k_{i+1} - 1$ do:
        i. $h_{i+1,j} := h_{i,2j} + h_{i,2j+1} \cdot G$.
       ii. clear $h_{i,2j}$ and $h_{i,2j+1}$
   (c) if $k_{i+1} > 1$ then $G := G^2$
   (d) $i := i + 1$
5. return $h := h_{i,0}$
```

Complexity Analysis

Theorem 2. *Algorithm 2 terminates after* $\mathcal{O}(nm \log(n) \log(mn))$ *operations in* $\mathbb{Z}$ *with a bit-complexity bound of* $\mathcal{O}(n^2 m^2 \log(nm))$ *CPU operations.*

Proof. Although we chose $\ell = 4$ we will make this proof using any constant value of ℓ. The cost of step 2 is that of $\lceil (n+1)/\ell \rceil$ calls to Algorithm 1 using ℓ coefficients. Thus theorem 1 tells us that step 2 costs $\mathcal{O}(nm \log(m))$ operations in $\mathbb{Z}$ with bit complexity bound $\mathcal{O}(nm^2 \log(m))$.

Step 3 involves a constant number of multiplications (or repeated squarings) of g. By using the same logic as the proof of lemma 1 these multiplications are of polynomials with degree $\mathcal{O}(m)$ and coefficients of $\mathcal{O}(m + \log(m))$ bits, this gives $\mathcal{O}(m \log(m))$ operations in $\mathbb{Z}$ and bit complexity bound of $\mathcal{O}(m^2 \log(m))$ for step 3.

In the i^{th} loop of step 4 creating the $h_{i+1,j}$ involves k_{i+1} polynomial multiplications each of degree $\mathcal{O}(2^{i-2}m\ell)$ polynomials with coefficients bounded of $\mathcal{O}(2^{i-1}m\ell)$ bits (and k_{i+1} polynomial additions).
This will cost $\mathcal{O}(k_{i+1}2^i m \log(2^i m))$ operations in $\mathbb{Z}$ with bit-complexity bound $\mathcal{O}(k_{i+1}2^{2i}m^2 \log(2^i m))$. The cost of the i^{th} iteration of step 4c involves squaring a polynomial of degree $m\ell 2^{i-1}$ and whose coefficients are smaller than $m\ell 2^i$. The cost of this is $\mathcal{O}(m2^i \log(m2^i))$ operations in $\mathbb{Z}$ and $\mathcal{O}(m^2 2^{2i} \log(2^i m))$ bit operations. It can be shown without much difficulty that $k_i \le (n+1)/(\ell \cdot 2^{i-1}) + 1$. To sum these costs over the $\mathcal{O}(\log(n))$ iterations of step 4 gives $\mathcal{O}(\sum_{i=1}^{\log(n)} k_{i+1}2^i m[i + \log(m)])$ which is $\mathcal{O}(nm[\log(n)^2 + \log(n)\log(m)])$ operations in $\mathbb{Z}$ and a bit-complexity bound of $\mathcal{O}(\sum_{i=1}^{\log(n)} 2^{2i}m^2[i + \log(m)])$ which is $\mathcal{O}(m^2 n \cdot \sum_{i=1}^{\log(n)}[2^i i + 2^i \log(m)])$. It is trivial to show via induction that $\sum_{i=1}^{k} 2^i i = 2 + 2^{k+1}(k-1)$. This gives the bit-complexity bound as $\mathcal{O}(m^2 n[n\log(n) + n\log(m)])$ proving the theorem.

1.3 Practical Timings

In this section we present a timing comparison of the main algorithm as implemented in FLINT and MAGMA's polynomial composition algorithm. These tests are provided as evidence that our algorithm is indeed practical. These timings are measured in seconds and were made on a 2400MHz AMD Quad-core Opteron processor, using gcc version 4.4.1 with the -O2 optimization flag, although the processes only utilized one of the four cores. Each composition performed is of a polynomial, f, of length n with randomized coefficients of bit-length $\le m$, and a polynomial, g, of degree m with randomized coefficients of bit-length $\le m$ and returns an expansion of $h = f(g)$.

We also compared these timings with the function

$$(mn)^2 \ln(mn)/(.95 \cdot 10^9).$$

Table 1. Divide-and-conquer polynomial composition in FLINT

$n\backslash m$	20	40	80	160	320	640	1280
20	.0009	.0038	.016	.077	0.41	1.96	8.9
40	.0036	.015	.071	0.40	2.0	9.4	
80	0.02	.072	.412	2.09	9.63		
160	0.072	0.415	2.1	9.7			
320	0.44	2.1	9.7				
640	2.05	9.64					
1280	9.46						

Table 2. Polynomial composition in Magma

$n\backslash m$	20	40	80	160	320	640	1280
20	.006	.053	.160	.630	2.55	12.47	64.0
40	.04	.32	1.09	4.67	21.7	110	
80	.47	2.0	8.52	38.0	196.4		
160	3.6	15	70	360			
320	28	133	659				
640	238	1267					
1280	2380						

In this case the function accurately models the given timings, in all cases, up to a factor which varied between 0.71 and 1.29. This model matches our bit-complexity bound given in Theorem 2.

We compared the MAGMA timings with the function

$$n^3m^2 \ln(mn)/(2.94 \cdot 10^9).$$

This function accurately models the given timings, in all cases, up to a factor which varied between 0.54 and 1.46. This model matches our estimate for Horner's method given by Theorem 1 in the case when $\ell = n$.

2 Divide and Conquer Division

In his paper [12], Mulders describes recursive divide-and-conquer type algorithms for the *short product* of polynomials (returning only the low degree terms of the product) and the *opposite short product* (returning only the higher degree terms).

Suppose two polynomials of length at least N are multiplied, but one only wishes to compute the terms of the product of degree less than N. We will denote such a short product by $\mathrm{SM}(N)$.

A basic algorithm for computing $\mathrm{SM}(N)$ is to compute a full $N \times N$ product using a standard polynomial multiplication algorithm and discard the unwanted terms. But this is often wasteful. For example, the Karatsuba algorithm breaks the full product up into three half sized products, but in two of the half sized products, one again doesn't require all the terms.

This leads naturally to a recursive Karatsuba-type algorithm where a short product $SM(N)$ is replaced by one full product with polynomials of half the size, $FM(N/2)$, and two short products $SM(N/2)$. At the bottom of the recursion, below some cutoff, the short products are computed using classical multiplication, computing only the required terms.

Mulders' algorithm for the short product, denoted $\mathrm{SM}_\beta(N)$ for some parameter $\frac{1}{2} \leq \beta \leq 1$, is a generalisation of this technique, also breaking the short product up into three multiplications. But this time there is a full $\beta N \times \beta N$ product $FM(\beta N)$ and two short products $\mathrm{SM}_\beta((1-\beta)N)$. The Karatsuba-type algorithm above, is the special case $\beta = \frac{1}{2}$.

In general, the recursion for Mulders' algorithm can be expressed:

$$\mathrm{SM}_\beta(N) = \mathrm{FM}(\beta N) + 2\mathrm{SM}_\beta((1-\beta)N). \tag{1}$$

This is completely general in that the full multiplications $\mathrm{FM}(\beta N)$ can be performed using any algorithm for ordinary polynomial multiplication.

When the full products are computed using Karatsuba multiplication, Mulders derives the optimal value $\beta = 0.694$ for his algorithm.

In their paper [8], Hanrot and Zimmermann give a slight variant of Mulders algorithm in which the original product is split into a full $k \times k$ product and two short $(N-k) \times (N-k)$ products, where the cutoff k now depends on N. Their method gives a significant improvement over Mulders' original fixed cutoff $k = \beta N$.

Mulders, In section 7 of his paper, gives a brief description of a recursive divide-and-conquer technique for performing what he calls *short division*, namely division of a polynomial of length at most $2n-1$ by a polynomial of length n without computing a remainder. This algorithm is faster than a *long division*, in which one computes a quotient and remainder, and is based on the same principle as his short multiplication algorithm.

Mulders' short division algorithm reduces the problem recursively to one long division, one short multiplication and one short division. As with his short multiplication algorithm, Mulders uses a fixed cutoff, not depending on the length of the polynomials.

Mulders reported that the optimal cutoff for his algorithm was very nearly $\beta = 1/2$ and that there was little practical benefit in introducing a different cutoff.

In this paper we describe a variant of Mulders' algorithm which uses a variable cutoff in the manner of Hanrot and Zimmerman. In addition, instead of computing a remainder directly, we compute only the product of the quotient and divisor (from which the remainder is easily obtained by subtraction from the dividend). We show that this simple variant of Mulders' algorithm has very good performance in practice whilst remaining simple to implement. We call this algorithm Mulders' algorithm for simplicity.

We also give a slightly faster variant of this algorithm in which we replace the full division in Mulders recursion with a third recursive algorithm which returns only a short product of divisor and quotient. We optimistically call this *half full division.*

We provide details of timing experiments below, performed with our implementation of these algorithms. Our implementation is included in the FLINT (Fast Library for Number Theory) package.

We compare an implementation Mulders' algorithm, with parameter $\beta = \frac{1}{2}$, with our improved algorithm (with the same parameter). We then show that if the parameter is allowed to vary with the size of the polynomials, a further improvement is possible on some architectures. The optimal parameter can then often be quite far from $\beta = 1/2$.

We compare our short division implementation with the implementations of polynomial division in the packages NTL [14] and Magma [6].

2.1 Description of the Short Division Algorithm

We assume throughout the following that we have available an algorithm $\textsc{Mul}(f, g)$ for computing a full product of polynomials. We also require the classical algorithms for long and short division, $\textsc{Div}(f, g)$ and $\textsc{Div\_short}(f, g)$ respectively, returning a quotient q and remainder r (or in the case of short division, just the quotient) such that $f = gq + r$.

We also require that we have available algorithms for computing the following short products:

Algorithm 2.11. $\textsc{Mul\_short}(f, g, n)$

Input: *Polynomials* $f = \sum_{i=0}^{n_1} a_i x^i$ *and* $g = \sum_{j=0}^{n_2} b_j x^j$ *and a non-negative integer* n*.*

Output: *The low* n *terms of the product of* f *and* g*, i.e.* $\sum_{k=0}^{n-1} c_k x^k$ *where* $c_k = \sum_{i+j=k} a_i b_j$*.*

Algorithm 2.12. $\textsc{Mul\_short\_opp}(f, g, n)$

Input: *As for* $\textsc{Mul\_short}$*.*

Output: *All terms of the product of* f *and* g except *the first* n*, i.e.* $\sum_{k=n}^{n_1+n_2-1} c_k x^k$ *where* $c_k = \sum_{i+j=k} a_i b_j$*, if* $n_1 + n_2 - 1 \geq n$ *and* 0 *otherwise.*

Firstly we describe the basic divide-and-conquer type algorithm for doing a long division. However we do not return the remainder r, but the product of the divisor and quotient, qg, from which the remainder can be computed as $r = f - gq$.

In all of the algorithms below, we always reduce to the case where $\deg(f)$ is $m - 1$ and $\deg(g)$ is $n - 1$ with $m = 2n - 1$, so that the quotient also has degree $n - 1$. In the case where $m > 2n - 1$ we can truncate f to length $2n - 1$, do a division with remainder reducing the problem to one with a shorter dividend. When $m < 2n - 1$ the quotient will have length $l = m - n + 1$ and only depends on the leading $2l - 1$ terms of f and the leading l terms of g. We can compute this using a short division and multiply out and subtract to obtain the remainder $r = f - gq$.

Algorithm 2.13. $\textsc{Divide\_conquer\_div}(f, g)$

Input: *Polynomials* $f = \sum_{i=0}^{m-1} a_i x^i$ *and* $g = \sum_{j=0}^{n-1} b_j x^j$

Output: *The quotient* q *and the full product* gq*.*

```
1. If $n <$ Cutoff return $\textsc{Div}(f, g)$
2. If $m \neq 2n - 1$ reduce to the case $m = 2n - 1$
3. $n_1 := \lceil n/2 \rceil$, $n_2 := n - n_1$
4. Let $b_1, b_2$ be such that $g = b_1 x^{n_2} + b_2$ with deg $b_2 < n_2$
```

```
 5. Let $b_3, b_4$ be such that $g = b_3x^{n_1} + b_4$ with deg $b_4 < n_1$
 6. Let $a_1, a_2, a_3$ be such that $f = a_1x^{n_2+n-1} + a_2x^{n-1} + a_3$ with deg $a_2 < n_2$,
    deg $a_3 < n - 1$
 7. $(q_1, b_1q_1) :=$DIVIDE_CONQUER_DIV$(a_1x^{n_1-1}, b_1)$
 8. $bq_1 := b_1q_1x^{n_2}+$ MUL$(b_2, q_1)$ (††)
 9. $t := a_1x^{2n_2-1} + a_2x^{n_2-1} - bq_1$ div $x^{n_1-n_2}$
10. $t' := t \bmod x^{2n_2-1}$
11. $(q_2, b_3q_2) :=$ DIVIDE_CONQUER_DIV$(t', b_3)$ (*)
12. $bq_2 := b_3q_2x^{n_1}+$ MUL$(b_4, q_2)$ (**)
13. Return $q := q_1x^{n_2} + q_2$, $qg := bq_3x^{n_1} + bq_4$ (†)
```

It is easy to turn this algorithm into an algorithm for short division, returning the quotient q only.

Algorithm 2.14. DIVIDE_CONQUER_DIV_SHORT(f, g)

Input: *As for* DIVIDE_CONQUER_DIV.

Output: *The quotient q of f and g.*

In DIVIDE_CONQUER_DIV(f, g) replace the call to DIVIDE_CONQUER_DIV at (*) by a call to the function DIVIDE_CONQUER_DIV_SHORT, remove the line (**), replace DIV with DIV_SHORT, replace MUL(b_2, q_1) at (††) with MUL_SHORT_OPP$(b_2, q_1, n_1 - 1)$ and return only the quotient q at (†). □

This is our first variant of Mulders' short division algorithm (with $\beta = \frac{1}{2}$). We now describe an improved version of this algorithm.

From now on, we assume that we have available an algorithm for classical division which only returns the lowest $n-1$ terms of the product of the divisor and quotient. We call it DIV_CLASSICAL_HALF_FULL(f, g) since it returns about half of the product that our long division returns. The ordinary classical algorithm for long division can be modified in an obvious way to return this half product.

With this algorithm available we are now able to introduce an algorithm which we call *half full division*. As for the half full classical algorithm, this recursive algorithm performs the same operation as a full division, but only returns the lowest $n - 1$ terms of the product of the divisor and quotient.

At the bottom of the recursion, this algorithm does classical half full division which has fewer operations than a full long division.

We will use this algorithm instead of a full division in our improved version of Mulders' division algorithm, thus achieving a faster short division.

Algorithm 2.15. HALF_FULL_DIV(f, g)

Input: *As for* DIVIDE_CONQUER_DIV_SHORT.

Output: *The quotient q and the low $n - 1$ terms of the product gq.*

```
1. If $m \neq 2n - 1$ reduce to the case $m = 2n - 1$
2. If $n <$ CUTOFF return DIV_CLASSICAL_HALF_FULL$(f, g)$
3. Let $n_1 = \lceil n/2 \rceil$, $n_2 = n - n_1$
4. Let $b_1, b_2$ be such that $g = b_1x^{n_2} + b_2$ with deg $b_2 < n_2$
```

```
 5. Let $b_3, b_4$ be such that $g = b_3x^{n_1} + b_4$ with $\deg b_4 < n_1$
 6. Let $a_1, a_2, a_3$ be such that $f = a_1x^{n_2+n-1} + a_2x^{n-1} + a_3$ with $\deg a_2 < n_2$,
    $\deg a_3 < n - 1$
 7. $(q_1, b_1q_1) :=$ HALF_FULL_DIV$(a_1x^{n_1-1}, b_1)$
 8. $bq_1 := b_1q_1x^{n_2} +$ MUL$(b_2, q_1)$
 9. $t := a_1x^{2n_2-1} + a_2x^{n_2-1} - bq_1 \text{ div } x^{n_1-n_2}$
10. $t' := t \bmod x^{2n_2-1}$
11. $(q_2, b_3q_2) :=$ HALF_FULL_DIV$(t', b_3)$
12. $bq_2 := b_3q_2x^{n_1} +$ MUL$(b_4, q_2)$
13. Return $q := q_1x^{n_2} + q_2$, $qg := bq_3x^{n_1} + bq_4$
```

Finally we are able to describe our improved version of Mulders' short division algorithm. As in Mulders' paper we allow the algorithm to split the inputs into unequal parts, however as per Hanrot and Zimmerman, we will allow the cutoff to vary with the length of the input polynomial g. For this purpose we define a parameter $k = k(n)$ whose optimal value will be determined experimentally.

Algorithm 2.16. DIVIDE_CONQUER_DIV_SHORT_IMPROVED(f, g)

Input: *As for* DIVIDE_CONQUER_DIV_SHORT.

Output: *The quotient q of f and g.*

```
 1. If $n <$ CUTOFF return DIV_SHORT$(f, g)$
 2. If $m \neq 2n - 1$ reduce to the case $m = 2n - 1$
 3. Let $n_1 = \lceil n/2 \rceil + \mathrm{k(n)}$, $n_2 = n - n_1$
 4. Let $b_1, b_2$ be such that $g = b_1x^{n_2} + b_2$ with $\deg b_2 < n_2$
 5. Let $b_3, b_4$ be such that $g = b_3x^{n_1} + b_4$ with $\deg b_4 < n_1$
 6. Let $a_1, a_2, a_3$ be such that $f = a_1x^{n_2+n-1} + a_2x^{n-1} + a_3$ with $\deg a_2 < n_2$,
    $\deg a_3 < n - 1$
 7. $(q_1, b_1q_1) :=$ HALF_FULL_DIV$(a_1x^{n_1-1}, b_1)$
 8. $bq_1 := b_1q_1x^{n_2} +$ MUL_SHORT_OPP$(b_2, q_1, n_1 - 1)$
 9. $t := a_1x^{2n_2-1} + a_2x^{n_2-1} - bq_1 \text{ div } x^{n_1-n_2}$
10. $t' := t \bmod x^{2n_2-1}$
11. $q_2 :=$ DIVIDE_CONQUER_DIV_SHORT_IMPROVED$(t', b_3)$
12. Return $q := q_1x^{n_2} + q_2$
```

2.2 Mulders' vs Divide-conquer-div-short-improved

We began our timing experiments by comparing times for our first variant of Mulders' division algorithm and the improved short division algorithm with $k(n) = 0$.

Unless otherwise noted, all our division timings and comparisons in this paper were performed on a 2.4GHz AMD Opteron Server. Each computation was (automatically) repeated many times and the lowest timing was recorded in each case.

Given a length n (degree $n-1$) and a number of bits b we let f be a polynomial of length $2n - 1$ with uniformly random coefficients of $2b$ bits and g be a polynomial of length n with uniformly random coefficients of b bits. We computed the short division of f and g using both algorithms.

Table 3. Comparison of simple and improved Mulders' variants

$n\backslash b$	8	16	32	64
64	90.6μs	92.6μs	106μs	123μs
128	266μs	276μs	300μs	346μs
256	758μs	763μs	854μs	964μs
512	2.43ms	2.32ms	2.82ms	3.01ms

$n\backslash b$	8	16	32	64
64	87.4μs	90.0μs	100μs	117μs
128	243μs	256μs	278μs	328μs
256	714μs	752μs	788μs	975μs
512	1.98ms	1.96ms	2.50ms	2.81ms

Table 4. Short division in Magma and FLINT

$n\backslash b$	8	16	32	64	128	256	512	1024	2048	4096	8192
32	32.6μs	70.8μs	99.4μs	101μs	124μs	155μs	259μs	574μs	1.62ms	4.58ms	14.0ms
	29.0μs	30.0μs	33.4μs	39.4μs	49.1μs	70.8μs	113μs	237μs	597μs	1.60ms	4.63ms
64	162μs	260μs	353μs	394μs	460μs	573μs	960μs	2.20ms	6.59ms	20.0ms	60.0ms
	87.4μs	90.0μs	100μs	117μs	150μs	239μs	375μs	741μs	1.80ms	4.80ms	13.7ms
128	767μs	1.03ms	1.36ms	1.45ms	1.78ms	2.28ms	3.90ms	9.33ms	25.5ms	77.5ms	
	243μs	256μs	278μs	328μs	433μs	711μs	1.12ms	2.21ms	5.27ms	13.4ms	
256	3.67ms	4.33ms	5.58ms	6.04ms	7.50ms	10.3ms	16.7ms	37.5ms	107ms		
	714μs	752μs	788μs	975μs	1.31ms	1.98ms	3.07ms	6.18ms	13.7ms		
512	16.0ms	17.5ms	23.3ms	25.5ms	30.0ms	40.0ms	64.0ms	153ms			
	1.98ms	1.96ms	2.50ms	2.81ms	3.67ms	5.60ms	8.80ms	16.6ms			

Table 5. Exact division in NTL, Magma and FLINT

$n\backslash b$	8	16	32	64	128	256	512	1024	2048	4096	8192
32	43.0μs	43.8μs	49.8μs	76.6μs	145μs	294μs	684μs	1.93ms	5.87ms	20.1ms	73.3ms
	23.2μs	49.1μs	82.5μs	92.8μs	106μs	139μs	237μs	533μs	1.57ms	4.59ms	14.0ms
	29.1μs	29.0μs	32.0μs	34.8μs	46.3μs	68.0μs	106μs	227μs	558μs	1.57ms	4.57ms
64	104μs	105μs	116μs	176μs	323μs	646μs	1.48ms	4.00ms	12.1ms	40.9ms	148ms
	81.1μs	180μs	294μs	340μs	370μs	510μs	905μs	2.10ms	6.12ms	18.3ms	57.5ms
	83.3μs	84.4μs	88.1μs	98.5μs	136μs	231μs	360μs	721μs	1.73ms	4.56ms	13.3ms
128	284μs	285μs	306μs	461μs	785μs	1.53ms	3.37ms	8.76ms	28.1ms	84.7ms	
	310μs	668μs	1.17ms	1.30ms	1.46ms	2.00ms	3.54ms	8.24ms	25.0ms	70.0ms	
	211μs	212μs	228μs	262μs	376μs	688μs	1.06ms	2.12ms	4.98ms	12.5ms	
256	838μs	839μs	880μs	1.33ms	2.11ms	4.00ms	8.35ms	20.6ms	57.8ms		
	1.19ms	2.66ms	4.50ms	5.17ms	5.92ms	8.00ms	14.7ms	32.9ms	97.5ms		
	502μs	512μs	553μs	661μs	982μs	1.87ms	2.93ms	5.71ms	13.5ms		
512	1.71ms	1.71ms	1.80ms	2.68ms	4.26ms	8.33ms	17.0ms	43.4ms			
	4.67ms	10.7ms	18.3ms	20.0ms	23.9ms	32.2ms	57.5ms	133ms			
	1.18ms	1.19ms	1.31ms	1.63ms	2.50ms	5.00ms	7.77ms	15.9ms			

Our timings showed that the improved algorithm was marginally faster (up to 23% but on average much less). The biggest improvements are where n is large and b is small. In the interests of space we omit all but the timings in this range. Timings for the improved algorithm are shown on the right and those for the original on the left.

A second timing experiment we performed was to adjust the value of $k(n)$ in the improved short division algorithm. Whether or not this had an effect proved to be highly architecture sensitive. On a 1.8GHz AMD K8 machine we found that if n is the length of g then for $20 < n \leq 100$ the value of $k(n)$ should be about $n/5$ and for $n \leq 20$ the value of $k(n)$ should be $n/4$.

However, on a 2.4GHz AMD K10 machine, a value of $k(n) = 0$ was roughly optimal for all sizes.

2.3 Comparison with other Implementations

The most important comparison for the purposes of this paper is the comparison between our variant of Mulders' algorithm and the polynomial division available in other packages. For this purpose we compare with the best open source and the best proprietary packages we are aware of. In the former case we compare with NTL v5.5.2 and in the latter case with Magma v2.16-7.

Whilst we do not know the algorithm used by Magma, we know that NTL uses a multimodular approach, performing the division using multiple primes then recombining with the Chinese Remainder Algorithm. It leverages highly optimised functions for division over $\mathbb{Z}/p\mathbb{Z}$.

Our first comparison is made using random polynomials of lengths $2n-1$ and n respectively, as described in the previous section. This tests the most general case for our algorithm. NTL does not offer inexact division over $\mathbb{Z}$, thus the first comparison is with Magma only (the top row in each case).

Finally we compare NTL, Magma and our improved divide-and-conquer algorithm (timing rows in that order) on exact divisions. Here we construct polynomials f, g with the given lengths n and uniformly random coefficients with the given number of bits b and perform the division h/g where $h = f * g$.

3 Conclusions

We have provided efficient divide-and-conquer style algorithms for the composition and division of univariate polynomials over $\mathbb{Z}$.

In the former case, we show that the algorithm is asymptotically fast with respect to bit complexity, effectively handling coefficient explosion.

In the latter case we have provided two easy to implement variants of Mulders' algorithm and shown that, at least on modern computers, the divide-and-conquer approach deserves a closer look, often outperforming other commonly used methods.

References

1. Bernstein, D.: Multiprecision Multiplication for Mathematicians. In: Accepted by Advances in Applied Mathematics (2001), find at http://cr.yp.to/papers.html#m3
2. de Boor, C.: B-Form Basics. Geometric Modeling: Algorithms and New Trends, pp. 131–148. SIAM, Philadelphia (1987)

3. Bostan, A., Salvy, B.: Fast conversion algorithms for orthogonal polynomials (preprint)
4. Prautzsch, H., Boehm, W., Paluszny, M.: Bézier and B-Spline Techniques. Springer, Heidelberg (2002)
5. Brent, R., Kung, H.T.: $\mathcal{O}((n \log n)^3/2)$ Algorithms for composition and reversion of power series. In: Brent, R., Kung, H.T. (eds.) Analytic Computational Complexity, pp. 217–225. Academic Press, New York (1975)
6. Cannon, J.J., Bosma, W. (eds.): Handbook of Magma Functions, 2.17th edn. (2010), http://magma.maths.usyd.edu.au/magma
7. von zur Gathen, J., Gerhard, J.: Modern Computer Algebra. Cambridge University Press, Cambridge (1999)
8. Hanrot, G., Zimmermann, P.: A long note on Mulder's short product. Journal of Symbolic Computation 37(3), 391–401 (2004)
9. Hart, W.: Fast Library for Number Theory: an introduction. In: Fukuda, K., van der Hoeven, J., Joswig, M., Takayama, N. (eds.) ICMS 2010. LNCS, vol. 6327, pp. 88–91. Springer, Heidelberg (2010), http://www.flintlib.org
10. Knuth, D.: The Art of Computer Programming, volume 2: Seminumerical Algorithms, 3rd edn., pp. 486–488. Addison-Wesley, Reading (1997)
11. Liu, W., Mann, S.: An analysis of polynomial composition algorithms, University of Waterloo Research Report CS-95-24 (1995)
12. Mulders, T.: On Short Multiplications and Divisions. In: AAECC, vol. 11, pp. 69–88 (2000)
13. Pan, V.: Structured matrices and polynomials: unified superfast algorithms, p. 81. Springer, Heidelberg (2001)
14. Shoup, V.: NTL: A Library for doing Number Theory, open-source library, http://shoup.net/ntl/

Fast and Robust Symbolic Model Order Reduction with Analog Insydes

Matthias Hauser, Christian Salzig, and Alexander Dreyer

Fraunhofer Institute for Industrial Mathematics (ITWM),
Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany
www.itwm.fraunhofer.de/analog-insydes

Abstract. Nowadays analog circuits become more and more complex. The growing number of devices hinder to understand the full behavior of these and new methods are required to support the design. This paper presents two new methods for handling complex nonlinear analog circuits which are available in the new release Analog Insydes 2011, the Mathematica toolbox for symbolic modeling, analysis and reduction of analog circuits. The transient symbolic model order reduction allows the approximation of behavioral models keeping static and dynamic properties. The new solving algorithm for symbolic equation systems based on sequential equations accelerates the simulation of the reference system as well as the verification of the reduced models. Furthermore, it increases the robustness of the solver permitting analyzes of significantly larger symbolic systems. As example, a voltage controller circuit is reduced using the introduced methods.

1 Introduction

Modern analog circuits consist of a growing number of elements and their design processes require many time-consuming simulations to determine whether the circuit characteristics fulfill the specifications or not. Often, an additional analysis of the circuits is needed to identify reasons for unexpected effects. The analysis of such a large circuit is rather difficult since the corresponding describing system of equations has a large number of terms, so that the characteristic parts of the behavioral model cannot be seen easily.

To support the design process, we have realized methods for symbolic model order reduction of behavioral models in the software Analog Insydes [Web], which is a toolbox of the computer algebra platform Mathematica. The Fraunhofer ITWM developed this addon whose new release Analog Insydes 2011 includes the methods presented in the following paragraphs.

Reducing a system of equations to its most important terms leads to a system of decreased size that approximates the behavior of the original system. After this step, accelerated simulations can be executed and deeper insights into the system behavior can be generated.

Due to its interfaces to different simulators, Analog Insydes can import netlists and simulation data as well as to export behavioral models and simulation data. Thus, Analog Insydes generates reduced behavioral models that

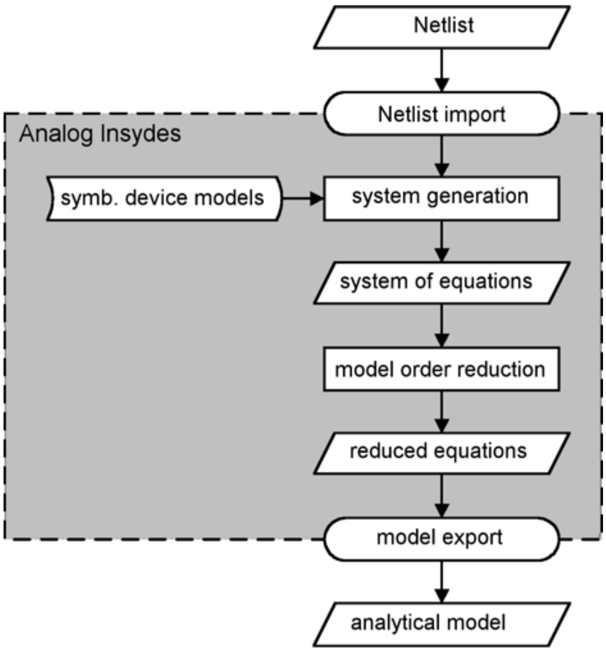


Fig. 1. Analog Insydes work flow

can be simulated by several simulators using common description languages like VHDL-AMS (see Fig. 1).

2 Symbolic Model Order Reduction

Starting with the netlist, the model parameters, and the operating point of an analog circuit Analog Insydes uses symbolic device models to generate the corresponding symbolic differential-algebraic system of equations (DAE):

$$f(\dot{x}, x, t) = 0 \tag{1}$$

Here, $f : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ denotes a model function, $x : \mathbb{R} \to \mathbb{R}^n$ the state vector, and t the time. Note that each equation is formulated as a sum of terms that are variables, parameters or linear and nonlinear expressions of them. To reduce this DAE the user has to specify additional conditions for fixing desired properties of the reduced model: the design point, the operating point, the mode (AC, DC or transient), and the input sources of the reduced model. Additionally a custom maximal error limit has to be set to guarantee a good approximation of the original system.

In the following lines we give a short description of the idea of the model order reduction (see Fig. 2):

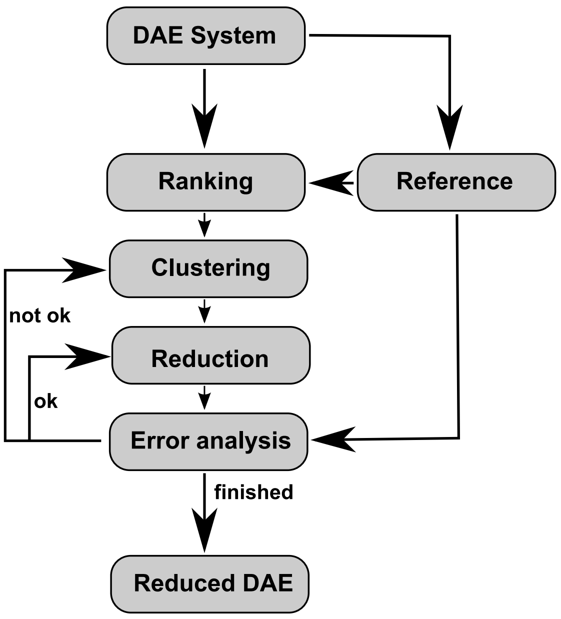


Fig. 2. Workflow of a symbolic model reduction in ANALOG INSYDES

First the algorithm calculates **reference** data of the original system (1). It will be used to verify the reduced system during the reduction process to check the occuring error. The reference consists of the simulation result of the system: in DC mode it is a list of DC operating points, in transient mode it is a list of transient curves depending on different input sources [Wichmann04].

In the next step a **ranking** of the terms of the circuit equations is performed. Here, the influence of the cancelation of single terms on the output behavior of the system is determined. This leads to an estimation which terms can be deleted without violating the predefined error limit. After removing one term from the original system the difference between the solution of the remaining ¨reduced¨ system and the reference is estimated. To determine the difference the user can choose between various error functions. This error is taken as ranking value.

Next, a **clustering** of the ranking result follows. All terms with similar ranking values are collected in clusters of terms to increase the efficiency of the following model reduction.

During the **reduction step** the procedure cancels the cluster of the smallest ranking values from the original system (1) and verifies the resulting system. If the resulting error crosses the pre-defined error limit the reduction is undone and sub-clusters are tested successively. Otherwise the reduction is kept and the next cluster with the smallest ranking values is taken. After testing all clusters, a reduced behavioral model is obtained which approximates the behavior of the original system within the given error-bounds. Note that analogous to the ranking step, many error functions can be used to determine the error made by canceling a cluster of terms in the reduction step.

2.1 Transient Symbolic Model Order Reduction

For approximating static systems we may only use the values of the system variables (like currents or voltages) in different operating points as circuit characteristics. To get insights in the transient behavior this is not sufficient. To keep much more of the dynamic circuit characteristics the introduced model order reduction method has to be extended for transient systems.

The transient model order reduction yields a reduced model that approximates certain static and dynamic behaviors of the original system. Here, the state variables, like the values of currents and voltages as well as the dynamic behavior of these (e. g. rise or reaction times) can be taken as circuit characteristics that are approximated.

Both, ranking and reduction step, have to verify the reduced system. They compare the resulting system behavior to the reference from the first step of the model reduction workflow. To rate the difference of characteristics between the reduced and the original system we had to design special error functions.

To compare just state variables it is sufficient to apply standard distance measures to the difference of solution curves. Denote the reference curve as ref, the solution curve of the reduced system as red, the distance can be defined in the p-norm as

$$||\mathsf{ref} - \mathsf{red}||_p = \left(\int_{t_0}^{t_{end}} |\mathsf{ref}(t) - \mathsf{red}(t)|^p \, \mathrm{d}t \right)^{1/p} \tag{2}$$

where $p \in \mathbb{N}$ and ref and red are defined in the interval $[t_0, t_{end}]$.

Using the maximum norm we get

$$||\mathsf{ref} - \mathsf{red}||_{\max} = \max_{t \in [t_0, t_{end}]} |\mathsf{ref}(t) - \mathsf{red}(t)| \, . \tag{3}$$

Comparing dynamic circuit characteristics requires special error functions.

For example, if the time a circuit needs to react on a unit step input is important, we use the function

$$error(\mathsf{ref}, \mathsf{red}) = |T(\mathsf{ref}) - T(\mathsf{red})| \, , \tag{4}$$

where the function T outputs the smallest time t^* for that holds that the output lies in the interval $[1-\epsilon, 1+\epsilon]$ for $\epsilon \in \mathbb{R}$ and all times $t \geq t^*$ (see Fig. 3).

In mathematical notation:

$$T(s) = \min\{ \hat{t} \,|\, s(t) \in [1-\epsilon, 1+\epsilon] \;\; \forall \, t \geq \hat{t}\} \tag{5}$$

for an output function $s : \mathbb{R} \to \mathbb{R}$.

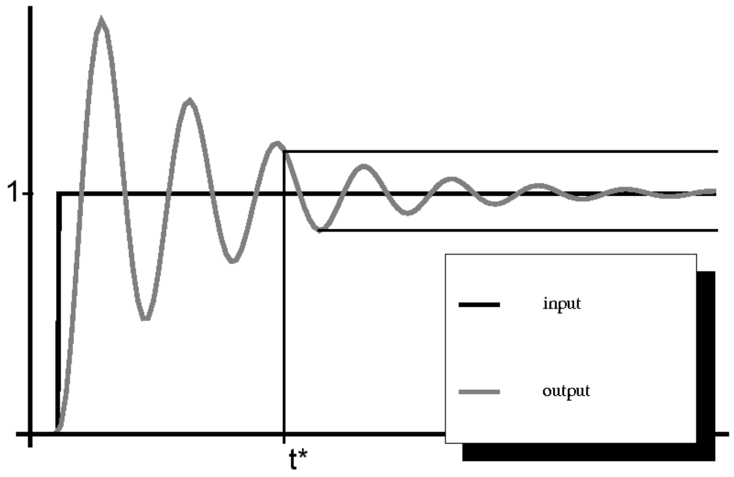


Fig. 3. Output of a circuit current from a step input

To analyze these effects, full transient and therefore really time consuming simulations are needed. Since the reference and reduction step contain a lot of these transient simulations, we introduce a new idea for saving simulation time.

Therefore, we can use a simplified simulation function to approximate the original solution in the ranking step. Due to the fact that the final error is checked in the reduction step at the end of the model order reduction method the pre-defined error limit is not exceeded anyway. However, the errors resulting from this simplified simulation function in the ranking step have an effect on the production of the clusters and thus on the choice which terms can be canceled without violating the error limit.

3 Numerical Advantages due to Sequential Equations

Considering the introduced model reduction workflow it is evident that simulation results of the system are needed frequently. Here in the reference and final reduction step many simulations are made to get this data. For simulating more quickly we present a new method of sequential equations. Let us assume that we can split the state variable $x : \mathbb{R} \to \mathbb{R}^n$ into

$$\begin{aligned} x_{seq} &= (x_{seq_1}, \ldots, x_{seq_{n_{seq}}}) \in \mathbb{R}^{n_{seq}} \quad , \\ x_{sim} &= (x_{sim_1}, \ldots, x_{sim_{n_{sim}}}) \in \mathbb{R}^{n_{sim}} \end{aligned} \tag{6}$$

with $n = n_{seq} + n_{sim}$ and define

$$x_{seq}^i = (x_{seq_1}, \ldots, x_{seq_i}) \in \mathbb{R}^i$$

for all $i \in \{1, \ldots, n_{seq}\}$. Furthermore, assume that we can divide the system of equations (1) into sequential equations f_{seq_i} and the remaining simultaneous equations f_{sim} such that the following holds

$$\begin{aligned} f_{seq_i}(\dot{x}_{seq}^{i-1}, \dot{x}_{sim}, x_{seq}^{i}, x_{sim}, t) = \\ x_{seq_i} - g_{seq_i}(\dot{x}_{seq}^{i-1}, \dot{x}_{sim}, x_{seq}^{i-1}, x_{sim}, t) = 0 \end{aligned} \tag{7}$$

where

$$g_{seq_i} : \mathbb{R}^{i-1} \times \mathbb{R}^{n_{sim}} \times \mathbb{R}^{i-1} \times \mathbb{R}^{n_{sim}} \times \mathbb{R} \to \mathbb{R}$$

for all $i \in \{1, \ldots, n_{seq}\}$. Then it holds for the remaining simultaneous equations that

$$f_{sim}(\dot{x}_{seq}, \dot{x}_{sim}, x_{seq}, x_{sim}, t) = 0 \tag{8}$$

with

$$f_{sim} : \mathbb{R}^{n_{seq}} \times \mathbb{R}^{n_{sim}} \times \mathbb{R}^{n_{seq}} \times \mathbb{R}^{n_{sim}} \times \mathbb{R} \to \mathbb{R}^{n_{sim}}.$$

Note that there can be more than one sequential block in the full system of equations [Platte06]. It is worth mentioning that the partition of equation (6) has to be chosen that equation (7) is fulfilled. In our application area of analog circuit analysis this is not the restriction at all, but this comes naturally from the structure of circuit equations.

In the case, that the equations contain derivatives of sequential variables, these are substituted by auxiliary variables. The latter are defined by a corresponding finite difference equation based on the integration method used in the numerical step. These additional equations are put to the simultaneous equation set.

Consider the following example equations:

$$x_1 - x_5^2 = 0 \tag{9}$$

$$x_2 - \left(\dot{x}_4 + \frac{x_1}{2}\right) = 0 \tag{10}$$

$$x_3 - \left(\frac{x_2}{x_1^2 + 1}\right) = 0 \tag{11}$$

$$\sqrt{x_4} + x_4 \cdot x_1 \cdot x_5 + \dot{x}_5 - \sin(t) = 0 \tag{12}$$

$$x_5 \cdot e^{x_1 + x_4} - 4(x_5^3 \cdot x_4 - x_2) = 0 \tag{13}$$

Here, the equations (9, 10, 11), and the corresponding variables $x_1, \ldots, x_3$ fulfill the sequential form (7). Thus they are called sequential. The remaining equations (12, 13) and the variables x_4, x_5 are then defined as simultaneous.

Using this, we can insert now the equations (9, 10, 11) repeatedly into the remaining equations (12, 13) until the two equations depend only on the simultaneous variables x_4, x_5:

$$\begin{aligned} \sqrt{x_4} + x_4 \cdot x_5^3 + \dot{x}_5 - \sin(t) = 0 \\ x_5 \cdot e^{x_5^2 + x_4} - 4(x_5^3 \cdot x_4 - \dot{x}_4 - \tfrac{x_5^2}{2}) = 0 \end{aligned} \tag{14}$$

This means that we have to solve this system for two variables and recover the variables $x_1, \ldots, x_3$ by using the equations (9, 10, 11) and the solution of the resulting system of equations (14). Summarizing, we have to do the following:

1. insert the sequential equations (7) into the simultaneous ones (8) until they depend only on the simultaneous variables x_{sim},
2. solve the resulting equations for the simultaneous variables x_{sim} (which also yields its derivations $\dot{x}_{sim}$),
3. compute the values for the sequential variables x_{seq} using the values of the simultaneous variables x_{sim} computed before and the sequential equations (7).

We only have to simulate the $n_{sim} \times n_{sim}$ system of equations to solve the full system (1). Note that the number of entries of the ¨reduced¨ jacobian (n_{sim}^2) is much smaller than the number of entries of the original one (n^2) which results in a much faster simulation of the system. Furthermore, we achieve a significantly better convergence of the simulation algorithms due to the decreased size of the dynamical part. Using the idea of sequential equations one can simulate industrial-sized systems.

4 Example

As industrial relevant example of an analog electronic circuit a voltage controller consisting of 14 MOSFETs, one resistor and one capacitance is taken. For simulating, the voltage controller circuit is connected to a testbench consisting of an input source, supply sources, and one RC-load.

After loading the netlist and model parameters to Analog Insydes a transient nonlinear symbolic system of equations of order 404 is generated (Fig. 5). Note that the corresponding circuit equations of size 404×404 have only 372 sequential equations. Thus the system which is solved internally consisting of the simultaneous equations is just of size 32×32 which leads to a stable and fast numerical simulation. Simulating the system takes 8.715s using a C-based simulator that uses the idea of sequential equations (see Sect. 3).

To accelerate the simulation and to delete all unimportant parts of the system of equations the symbolic model order reduction of Analog Insydes is run. The voltage controlling characteristic of the circuit shall be preserved. That means, that the output voltage V_{out} equals zero if the input voltage V_{in} exceeds a limit V_{lim}. Otherwise the output voltage is larger than zero.

In mathematical notation:

$$\begin{aligned} \text{if} \quad & V_{in} < V_{lim} \\ \text{then} \quad & V_{out} = 2.81V \\ \text{else} \quad & V_{out} = 0V \end{aligned}$$

To obtain this behavior, we take a ramp, that crosses the voltage limit V_{lim} where the voltage controller switches its state, as input voltage. As error functions the maximum norm (3), the $\mathcal{L}^2$-norm (2) and a self-defined error function

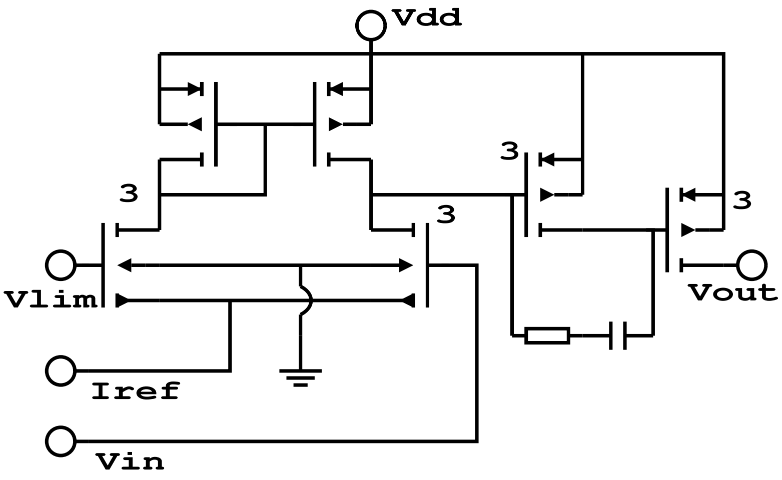


Fig. 4. Schematics of the voltage controller circuit

```
Equations        : 404 (100 %)
Variables        : 404 (100 %)
Derivative terms : 23 (100 %)
Sums in levels   : {2517 (100 %), 985 (100 %)}
Simulation time  : 8.715027 s (100 %)
```

Fig. 5. Data of the full system of the voltage controller

([4]) are taken. Note that here the function T returns the time t^* where the voltage controller switches the state, i. e. where the output function jumps from 2.81V to 0V.

Fig. 6, 7 and 8 show the data of the resulting reduced systems. The order of these systems is reduced to 48%–50% of the original value and the number of terms in level 0 to 27%–31%. The resulting simulation time is then 2.48s–2.65s, which is about 30% of the simulation time of the original system.

An easily interpretable result is that the procedure reduces the number of derivatives occuring in the system of equations from 23 to 2 or 3, respectively. Thereby, only those derivatives which correspond to the voltages of the capacitances of the voltage controller and the testbench are kept. Thus all derivatives used to model the MOSFETs (BSIM3v3, e. g. instrinsic charges) are deleted since the effect of these on the system behavior in this setting is too small. Here Analog Insydes enhances the insight into the system behavior thanks to the symbolic modeling of the behavioral model and the symbolic approximation methods.

Figure 9 shows the output behavior of the reference and the reduced systems based on the three error functions (red.max = maximum norm, red.L2 $=\mathcal{L}^2$-norm, red.t* = self-defined switching time error function).

```
Equations        : 202 (50 %)
Variables        : 202 (50 %)
Derivative terms : 3 (13 %)
Sums in levels   : {771 (30.6 %), 773 (78.5 %)}
Simulation time  : 2.656573 s (30.5 %)
```

Fig. 6. Data of the reduced system using the max norm

```
Equations        : 198 (49 %)
Variables        : 198 (49 %)
Derivative terms : 2 (8.7 %)
Sums in levels   : {710 (28.2 %), 730 (74.1 %)}
Simulation time  : 2.617357 s (30 %)
```

Fig. 7. Data of the reduced system using the $\mathcal{L}^2$-norm

```
Equations        : 195 (48.3 %)
Variables        : 195 (48.3 %)
Derivative terms : 2 (8.7 %)
Sums in levels   : {696 (27.6 %), 717 (72.8 %)}
Variables        : 195
Simulation time  : 2.480027 (28.45 %)
```

Fig. 8. Data of the reduced system using the self-defined switching time error function

The output of the reduced system of size 202 corresponding to the maximums norm is very similar to the simulation result of the original system. Note that if the output of the reduced system red would jump from "on" (2.81V) to "off" (0V) a little bit later than the reference ref, the resulting error $|\mathsf{ref}(\cdot) - \mathsf{red}(\cdot)|$ between the two output curves is large at this jump time. That means that the maximum error function (3) would lead to a large error that implies that this reduction is not allowed although there is only a small time delay in the jump time. To avoid these problems we used the second error function ($\mathcal{L}^2$-norm).

The system reduced with respect to the $\mathcal{L}^2$-norm is smaller than the previous ones (198 equations, especially only 2 derivative terms). Considering again Fig. 9 shows that the freedom, the $\mathcal{L}^2$-norm implies, was utilized. Although the descent of the output curve is more steep, the qualitative behavior of the system output is kept. To ensure that only the qualitative behavior of the original system is reproduced it is sufficient to use the third error function. It guarantees that the time point of state switching of the voltage controller is approximated and leads to a further reduction. The resulting model is of size 195×195.

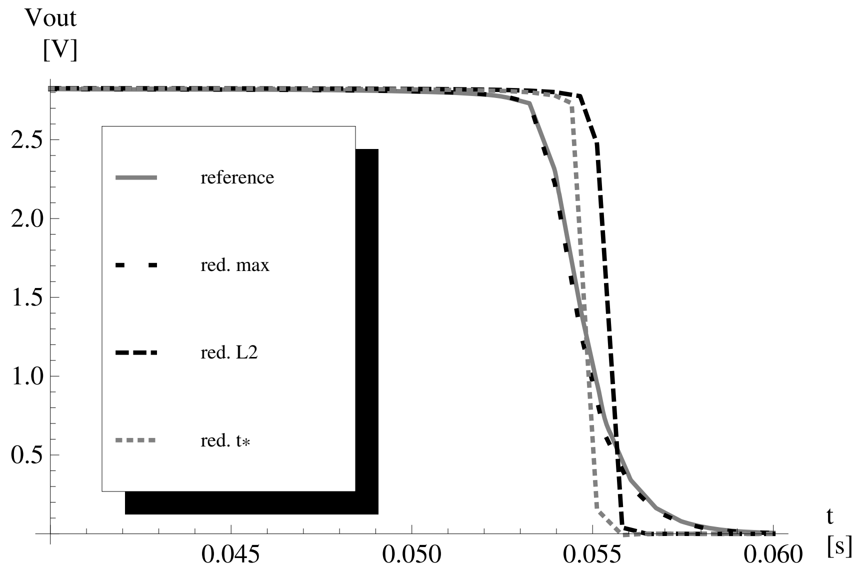


Fig. 9. Plot of the output of the original and the reduced systems

5 Conclusion

This contribution presented methods of the new release Analog Insydes 2011, the toolbox of Mathematica for modeling, analysis and reduction of analog circuits. The main focus laid on the extension of symbolic model order reduction methods to transient systems. The generated reduced transient systems keep static and dynamic circuit characteristics. This results in faster simulations and increased insights into the system behavior which supports the design of modern analog circuits.

Furthermore, accelerated and more robust simulations needed for complex error estimation tasks can be achieved by the idea of sequential equations. The introduced methods have been demonstrated on an example circuit to highlight the benefit of their usage and show the effect of different error functions.

References

[Ciccazzo08] Ciccazzo, A., Halfmann, T., Marotta, A., Rinaudo, S., Venturi, A.: Introduction of Symbolic Simplified Expressions in Circuit Optimization. In: Minisymposium: Optimization and Model Order Reduction in Circuit Design, The European Consortium For Mathematics In Industry (ECMI 2008), University College London, UK (2008)

[Halfmann08] Halfmann, T., Broz, J., Knoth, C., Platte, D., Rotter, P.: Generation of efficient behavioral models using model compilation and model reduction techniques. In: Proc. Xth International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SMACD 2008), Erfurt (2008)

[Platte06] Platte, D., Sommer, R., Broz, J., Dreyer, A., Halfmann, T., Barke, E.: Automatische nichtlineare Verhaltensmodellgenerierung mit sequentieller Gleichungsstruktur. In: 9. ITG/GMM-Fachtagung Analog 2006: Entwicklung von Analogschaltungen mit CAE-Methoden (ANALOG 2006), Dresden (2006)

[Platte08] Platte, D.: Simulation Efficiency of Analog Behavioral Models - Analyses and Improvements, Dissertation, Cuvillier Verlag Göttingen (July 2008)

[Salzig10] Salzig, C., Hauser, M.: Design of robust electronic circuits for yield optimization. In: XIth International Workshop on Symbolic and Numerical Methods, Modeling and Applications to Circuit Design (SM2ACD 2010), Tunis-Gammarth, Tunesia (October 2010)

[Web] Analog Insydes "the INtelligent SYmbolic DEsign System for analog circuits", the Mathematica toolbox for design, analysis and reduction of analog circuits, http://www.analog-insydes.com

[Wichmann04] Wichmann, T.: "Symbolische Reduktionsverfahren für nichtlineare DAE-Systeme". PHD Fraunhofer ITWM, Kaiserslautern (2004)

On Invariant Manifolds of Lagrange Systems

Valentin Irtegov and Tatyana Titorenko

Institute for System Dynamics and Control Theory SB RAS,
134, Lermontov str., Irkutsk, 664033, Russia
irteg@icc.ru

Abstract. This paper is a continuation of the previous work [1]. In the present paper we propose a new approach for obtaining and qualitative analysis of invariant manifolds of Lagrange systems, which possess cyclic first integrals. The main idea consists in the use of "extended" characteristic functions. The proposed approach is demonstrated by examples of concrete mechanical systems. The computer algebra system (CAS) "Mathematica" is applied for computations.

1 Introduction

A series of mechanical problems, for example, such as finding stationary sets and analysing their stability by the Routh–Lyapunov method, investigation of bifurcations and so on, is reduced to algebraic problems. The latter allows one to apply effectively the computer algebra methods and tools in solving these problems.

This paper considers the problem of obtaining invariant manifolds (IMs) for nonlinear conservative Lagrange systems with cyclic coordinates, which by means of the Legendre transformation are reduced to the Routh systems. The resulted Routh systems are smaller in dimension than the corresponding Lagrange systems. We propose a new approach for deriving and analysing IMs of such systems. This approach is based on the use of the "extended" Routh functions. The latter reduces the above problem to an algebraic problem of finding an unknown function depending on positional coordinates.

We demonstrate our technique by examples of analysis of concrete mechanical systems. One of the Lagrange systems under consideration has, in our opinion, interesting properties. It can be reduced to the linear Routh system by the Legendre transformation. The second example concerns the Clebsh–Tisserand–Brun problem. In this case, the Routh function is a nonlinear one. Such a choice of examples allows one to demonstrate some possibilities of the technique proposed and to reveal its specificities.

We used the computer algebra tools for performing direct computations and for conducting some computational experiments.

2 The Case of the Linear Routh System

In [1], a special class of the nonlinear conservative Lagrange systems with cyclic first integrals has been found. These systems are reduced to the linear Routh systems of the general type (see [1]) by means of the Legendre transformation. Let us consider such a system to demonstrate our approach.

We shall consider the Lagrange system with one cyclic q_1 and two positional coordinates q_2, q_3

$$2L = \frac{1}{(n_1 + (a_1 q_2 + b_1 q_3)^2)} \dot{q}_1^2 + a_0 \dot{q}_2^2 + c_0 \dot{q}_3^2 - (b q_2^2 + c q_3^2). \tag{1}$$

This system admits the first integral

$$\frac{\partial L}{\partial \dot{q}_1} = \frac{1}{(n_1 + (a_1 q_2 + b_1 q_3)^2)} \dot{q}_1 = p = const. \tag{2}$$

Here $a_0, a_1, b, b_1, c, c_0, c_1, n_1$ are some constants.

Let us construct the Routh function

$$R = L - p\dot{q}_1 = \frac{1}{2}[a_0 \dot{q}_2^2 + c_0 \dot{q}_3^2 - (b q_2^2 + c q_3^2) - p^2(n_1 + (a_1 q_2 + b_1 q_3)^2)]$$

corresponding to L. This function is quadratic in the velocities and the coordinates.

Next we construct the “extended” Routh function. To this end, we add the full derivative of a function $f(q_2, q_3)$ to R

$$\tilde{R} = R + p\,(\frac{\partial f}{\partial q_2}\dot{q}_2 + \frac{\partial f}{\partial q_3}\dot{q}_3),$$

where $f(q_2, q_3)$ is some unknown function, which should be determined.

The Lagrangian corresponding to $\tilde{R}$ can easily be reconstructed [1]. It writes

$$\begin{aligned} 2\tilde{L} = {} & a_0 \dot{q}_2^2 + c_0 \dot{q}_3^2 + \frac{1}{(n_1 + (a_1 q_2 + b_1 q_3)^2)}[\dot{q}_1 + \frac{\partial f}{\partial q_2}\dot{q}_2 + \frac{\partial f}{\partial q_3}\dot{q}_3]^2 \\ & -(b q_2^2 + c q_3^2). \end{aligned} \tag{3}$$

Comparing the above expressions for $\tilde{R}$ and for $\tilde{L}$ enables us to do the following conclusion.

Statement. Adding the full derivative of a function $f(q_2, q_3)$ multiplied by p (where p is the constant of the cyclic integral) to the function R is equivalent to adding the same derivative to the cyclic velocity of the function L, which corresponds to R.

The Routh equations, which are the same for R and $\tilde{R}$, have the form

$$\begin{aligned} \frac{d}{dt}\Big(\frac{\partial R}{\partial \dot{q}_2}\Big) - \frac{\partial R}{\partial q_2} &= a_0 \ddot{q}_2 + a_1 p^2 (a_1 q_2 + b_1 q_3) + b q_2 = 0, \\ \frac{d}{dt}\Big(\frac{\partial R}{\partial \dot{q}_3}\Big) - \frac{\partial R}{\partial q_3} &= c_0 \ddot{q}_3 + b_1 p^2 (a_1 q_2 + b_1 q_3) + c q_3 = 0. \end{aligned} \tag{4}$$

Let us state the problem to find the invariant manifolds of equations (4) with the aid of the “extended” function $\tilde{R}$.

For this purpose, according to the Routh–Lyapunov method we write down the stationary conditions $\tilde{R}$ with respect to the phase variables

$$\begin{aligned}
&\frac{\partial \tilde{R}}{\partial \dot{q}_2} = a_0 \dot{q}_2 + p\frac{\partial f}{\partial q_2} = 0, \quad \frac{\partial \tilde{R}}{\partial \dot{q}_3} = c_0 \dot{q}_3 + p\frac{\partial f}{\partial q_3} = 0,\\
&\frac{\partial \tilde{R}}{\partial q_2} = p\,(\frac{\partial^2 f}{\partial q_2^2}\dot{q}_2 + \frac{\partial^2 f}{\partial q_2 \partial q_3}\dot{q}_3) - bq_2 - p^2 a_1(a_1 q_2 + b_1 q_3) = 0,\\
&\frac{\partial \tilde{R}}{\partial q_3} = p\,(\frac{\partial^2 f}{\partial q_2 \partial q_3}\dot{q}_2 + \frac{\partial^2 f}{\partial q_3^2}\dot{q}_3) - cq_3 - p^2 b_1(a_1 q_2 + b_1 q_3) = 0. \qquad (5)
\end{aligned}$$

The solutions of equations (5) allow one to determine both the stationary solutions and the IMs of differential equations (4), which correspond to the “extended” Routh function $\tilde{R}$. We take interest in the IMs. To this end, we have to consider the cases when equations (5) are dependent.

Let us eliminate the velocities from last two equations (5) with the aid of the first two equations. As a result, we have the following conditions of degeneration for system (5)

$$\begin{aligned}
&\frac{\partial \hat{R}}{\partial q_2} = -p^2(\frac{1}{a_0}\frac{\partial^2 f}{\partial q_2^2}\frac{\partial f}{\partial q_2} + \frac{1}{c_0}\frac{\partial^2 f}{\partial q_2 \partial q_3}\frac{\partial f}{\partial q_3} + a_1(a_1 q_2 + b_1 q_3)) - bq_2 = 0,\\
&\frac{\partial \hat{R}}{\partial q_3} = -p^2(\frac{1}{a_0}\frac{\partial^2 f}{\partial q_2 \partial q_3}\frac{\partial f}{\partial q_2} + \frac{1}{c_0}\frac{\partial^2 f}{\partial q_3^2}\frac{\partial f}{\partial q_3} + b_1(a_1 q_2 + b_1 q_3)) - cq_3 = 0. \quad (6)
\end{aligned}$$

Here $\hat{R}$ is the function $\tilde{R}$, in which the generalized velocities are eliminated with the aid of equations (5).

The obtained conditions can be considered as partial differential equations for finding the function $f(q_2, q_3)$. It can be easily shown that these equations are also the partial derivatives of the Hamilton function (corresponding to the Routh function R), in which the generalized impulses are replaced by the derivatives of the function $f(q_2, q_3)$ with respect to the generalized coordinates. Having denoted this function by $\hat{H}$, we can write equations (6) as follows

$$\begin{aligned}
&\frac{\partial \hat{H}}{\partial q_2} = p^2\Big(\frac{1}{a_0}\frac{\partial^2 f}{\partial q_2^2}\frac{\partial f}{\partial q_2} + \frac{1}{c_0}\frac{\partial^2 f}{\partial q_2 \partial q_3}\frac{\partial f}{\partial q_3} + a_1(a_1 q_2 + b_1 q_3)\Big) + bq_2 = 0,\\
&\frac{\partial \hat{H}}{\partial q_3} = p^2\Big(\frac{1}{a_0}\frac{\partial^2 f}{\partial q_2 \partial q_3}\frac{\partial f}{\partial q_2} + \frac{1}{c_0}\frac{\partial^2 f}{\partial q_3^2}\frac{\partial f}{\partial q_3} + b_1(a_1 q_2 + b_1 q_3)\Big) + cq_3 = 0. \quad (7)
\end{aligned}$$

Such a PDE system, which can be used here to find the function $f(q_2, q_3)$, is usually called the Hamilton–Jacobi system [2]. System (7) is equivalent to the Hamilton–Jacobi equation, which, in the given case, writes

$$\begin{aligned}
&2\hat{H}\Big(q_2, q_3, \frac{\partial f}{\partial q_2}, \frac{\partial f}{\partial q_3}\Big) = p^2\Big[\frac{1}{a_0}\Big(\frac{\partial f}{\partial q_2}\Big)^2 + \frac{1}{c_0}\Big(\frac{\partial f}{\partial q_3}\Big)^2 + (a_1 q_2 + b_1 q_3)^2\Big]\\
&-(bq_2^2 + cq_3^2) = 0. \qquad (8)
\end{aligned}$$

By direct computation we can show that the quadratic form

$$2f(q_2, q_3) = c_1 q_2^2 + 2c_2 q_2 q_3 + c_3 q_3^2 \tag{9}$$

is one of solutions of equations (7) (and also (8)) for the defined values of constants c_1, c_2, c_3.

Substitute the derivatives of quadratic form (9) into equations (7) (or (8)) and equate the coefficients of the generalized coordinates q_2, q_3 in the obtained expressions to zero. As a result, we have the system of the quadratic equations

$$\left(\frac{c_1}{a_0}\right)^2 + \left(\frac{c_2}{\sqrt{a_0 c_0}}\right)^2 = -\frac{a_1^2 p^2 + b}{a_0 p^2}, \quad \left(\frac{c_2}{\sqrt{a_0 c_0}}\right)^2 + \left(\frac{c_3}{c_0}\right)^2 = -\frac{b_1^2 p^2 + c}{c_0 p^2},$$
$$\frac{c_2}{\sqrt{a_0 c_0}}\left(\frac{c_1}{a_0} + \frac{c_3}{c_0}\right) = -\frac{b_1 a_1}{\sqrt{a_0 c_0}}.$$

for finding c_1, c_2, and c_3.

Under the following denotations

$$\frac{c_1}{a_0} = x, \ \ \frac{c_3}{c_0} = z, \ \ \frac{c_2}{\sqrt{a_0 c_0}} = y, \ \ -\frac{a_1^2 p^2 + b}{a_0 p^2} = A, \ \ -\frac{b_1^2 p^2 + c}{c_0 p^2} = B, \ \ -\frac{b_1 a_1}{\sqrt{a_0 c_0}} = D,$$

the latter system writes

$$x^2 + y^2 = A, \quad y^2 + z^2 = B, \quad y(x + z) = D. \tag{10}$$

Equations (10) are reduced by means of the Gröbner bases method to the following system of equations

$$[(A-B)^2 + 4D^2]z^4 + [2(A-3B)D^2 - 2(A-B)^2 B]z^2 + [(B-A)B + D^2]^2 = 0,$$
$$2D[(B-A)B + D^2]y + [(A-B)^2 + 4D^2]z^3 + [(A-5B)D^2 - (A-B)^2 B]z = 0,$$
$$2D^2[(B-A)B + D^2]x - [4(B-A)D^2 - (A-B)^3]z^3 - [(A-B)^3 B - (A-3B)$$
$$\times (A-B)D^2 + 2D^4]z = 0,$$

which has four solutions

$$x = \pm\frac{(A+\bar{A})}{(A+B+2\bar{A})}, \ y = \pm\frac{D}{(A+B+2\bar{A})}, \ z = \pm\frac{(B+\bar{A})}{(A+B+2\bar{A})};$$
$$x = \pm\frac{(A-\bar{A})}{(A+B-2\bar{A})}, \ y = \pm\frac{D}{(A+B-2\bar{A})}, \ z = \pm\frac{(B-\bar{A})}{(A+B-2\bar{A})}.$$

Here $\bar{A} = \sqrt{AB - D^2}$.

By IMs definition, it can easily be verified that first two stationary equations (5) – for each found set of values c_1, c_2, c_3 – define the invariant manifolds of Routh equations (4). These IMs can be "lifted up" into the phase space of Lagrange system (1). For this purpose, it is necessary to add the corresponding cyclic integral to the IMs equations.

Analogously, the manifolds, which were obtained for the Routh equations, can be "lifted up" into the phase space of Lagrange system (3). In this case, the unknowns c_1, c_2, c_3 of the Lagrange function should be replaced by the corresponding found values. Here the cyclic integral corresponding to the function $\tilde{L}$ should be added to the IMs equations.

2.1 Investigation of Stability of the Invariant Manifolds

The "extended" Routh functions, which assume a stationary value on the IMs obtained with the aid of above technique, can be used as Lyapunov's functions for the investigation of stability of these IMs by the Routh–Lyapunov method.

To this end, let us write down the equations of one of the found IMs in terms of the initial parameters

$$\begin{aligned}\sqrt{a_0}\dot{q_2} + \Big[(\sqrt{c_0}(a_1^2p^2+b) - \sqrt{a_0}\rho)q_2 + \sqrt{c_0}a_1b_1p^2q_3\Big]\,\sigma = 0,\\ \sqrt{c_0}\dot{q_3} + \Big[\sqrt{a_0}a_1b_1p^2q_2 + (\sqrt{a_0}(b_1^2p^2+c) - \sqrt{c_0}\rho)q_3\Big]\,\sigma = 0,\end{aligned}$$

where

$$\rho = \sqrt{bc + p^2(ca_1^2 + bb_1^2)},\ \ \sigma = [2\sqrt{a_0c_0}\rho - c_0(a_1^2p^2+b) - a_0(b_1^2p^2+c)]^{-1/2}.$$

The equations of the perturbed motion, which have been derived in the neighborhood of the IM, are

$$\begin{aligned}\dot{y}_1 &= \Big[(c_0(a_1^2p^2+b) - \sqrt{a_0c_0}\rho)y_1 + a_0a_1b_1p^2y_2\Big](a_0c_0)^{-1/2}\,\sigma,\\ \dot{y}_2 &= \Big[c_0a_1b_1p^2y_1 + (a_0(b_1^2p^2+c) - \sqrt{a_0c_0}\rho)y_2\Big](a_0c_0)^{-1/2}\,\sigma.\end{aligned}$$

Here y_1, y_2 are deviations of the perturbed solution from the unperturbed one.

In the deviations y_1, y_2 the "extended" Routh function writes

$$2\Delta\tilde{R} = \frac{y_1^2}{a_0} + \frac{y_2^2}{c_0} - p^2n_1.$$

Having computed the derivative of the latter expression due to the equations of the perturbed motion, we have

$$\begin{aligned}\frac{d}{dt}\Delta\tilde{R} = \Omega\Big[&c_0^{3/2}\Big(\sqrt{c_0}(a_1^2p^2+b) - \sqrt{a_0}\rho\Big)y_1^2 + 2a_0c_0a_1b_1y_1y_2 + a_0^{3/2}\Big(\sqrt{a_0}(b_1^2p^2\\ &+c) - \sqrt{c_0}\rho)\Big)y_2^2\Big],\end{aligned}$$

where $\Omega = [a_0^3c_0^3(2\sqrt{a_0c_0}\rho - ca_0 - bc_0 - p^2(a_0b_1^2 + c_0a_1^2))]^{-1/2}$.

The following conditions $a_0 > 0,\ c_0 > 0$ hold by virtue of the positive kinetic energy of the system under consideration. Hence, the function $\Delta\tilde{R} + p^2n_1$ is the positive definite one. Its derivative $d\Delta\tilde{R}/dt$ is negative definite when the following conditions

$$a_1 \neq 0,\ \ b_1 \neq 0,\ \ p \neq 0,\ \ b + b_1^2p^2 < 0,\ c + \frac{a_1^2\,b\,p^2}{b + b_1^2\,p^2} < 0$$

hold. Consequently, the invariant manifold under investigation is asymptotically stable under above conditions and for the defined values of p.

2.2 On Lagrange Systems with $n > 2$ Positional Coordinates

Now, let us discuss an efficiency of our technique for the n-dimensional Lagrange systems of above type.

We have conducted a series of computational experiments for the Lagrange systems with one cyclic and $n = 3, \dots, 10$ positional coordinates. We can assert that the problem of finding the IMs for the Lagrange systems with $n > 2$ positional coordinates can be reduced (as well as the case when $n = 2$) to solving the system of algebraic quadratic equations of type (10). The complexity consists in solving these equations, the number of which depends on the number of the system positional coordinates. The number of such equations is $n + \begin{pmatrix} 2 \\ n \end{pmatrix}$, where $\begin{pmatrix} 2 \\ n \end{pmatrix}$ is a binomial coefficient.

Let us consider in support of the above reasoning the Lagrange system with one cyclic q_1 and three positional q_2, q_3, q_4 coordinates

$$2L = \frac{\dot q_1^2}{n_1 + (a_1 q_2 + b_1 q_3 + c_1 q_4)^2} + a_0 \dot q_2^2 + c_0 \dot q_3^2 + b_0 \dot q_4^2 - (b q_2^2 + c q_3^2 + a q_4^2). \quad (11)$$

In this case, the cyclic integral, the Routh function and the "extended" Routh function write

$$\frac{\partial L}{\partial \dot q_1} = \frac{\dot q_1}{n_1 + (a_1 q_2 + b_1 q_3 + c_1 q_4)^2} = p = const, \quad (12)$$

$$2R = L - p\dot q_1 = a_0 \dot q_2^2 + c_0 \dot q_3^2 + b_0 \dot q_4^2 - (b q_2^2 + c q_3^2 + a q_4^2) - p^2(n_1 + (a_1 q_2 \\ + b_1 q_3 + c_1 q_4)^2), \quad (13)$$

$$\tilde R = R + p\,(\frac{\partial f}{\partial q_2}\dot q_2 + \frac{\partial f}{\partial q_3}\dot q_3 + \frac{\partial f}{\partial q_4}\dot q_4), \quad (14)$$

respectively. Here $a_0, a_1, b, b_0, b_1, c, c_0, c_1, n_1$ are some constants.

The Lagrangian corresponding to $\tilde R$ has the form

$$2\tilde L = a_0 \dot q_2^2 + c_0 \dot q_3^2 + b_0 \dot q_4^2 + \frac{1}{(n_1 + (a_1 q_2 + b_1 q_3 + c_1 q_4)^2)}[\dot q_1 + \frac{\partial f}{\partial q_2}\dot q_2 + \frac{\partial f}{\partial q_3}\dot q_3 \\ + \frac{\partial f}{\partial q_4}\dot q_4]^2 - (b q_2^2 + c q_3^2 + a q_4^2). \quad (15)$$

Also as above, we shall look for the unknown function $f(q_2, q_3, q_4)$ in the form

$$2f(q_2, q_3, q_4) = d_1 q_2^2 + d_2 q_3^2 + d_3 q_4^2 + d_4 q_2 q_3 + d_5 q_2 q_4 + d_6 q_3 q_4, \quad (16)$$

where d_i are some constants, which should be determined.

Write down the stationary conditions $\tilde R$ with respect to the phase variables

$$\frac{\partial \tilde R}{\partial \dot q_2} = a_0 \dot q_2 + p\frac{\partial f}{\partial q_2} = 0, \quad \frac{\partial \tilde R}{\partial \dot q_3} = c_0 \dot q_3 + p\frac{\partial f}{\partial q_3} = 0, \quad \frac{\partial \tilde R}{\partial \dot q_4} = b_0 \dot q_4 + p\frac{\partial f}{\partial q_4} = 0,$$

$$\begin{aligned}
\frac{\partial \tilde{R}}{\partial q_2} &= p\,(\frac{\partial^2 f}{\partial q_2^2}\dot{q}_2 + \frac{\partial^2 f}{\partial q_2 \partial q_3}\dot{q}_3 + \frac{\partial^2 f}{\partial q_2 \partial q_4}\dot{q}_4) - bq_2 - p^2 a_1(a_1 q_2 + b_1 q_3 + c_1 q_4) = 0,\\
\frac{\partial \tilde{R}}{\partial q_3} &= p\,(\frac{\partial^2 f}{\partial q_2 \partial q_3}\dot{q}_2 + \frac{\partial^2 f}{\partial q_3^2}\dot{q}_3 + \frac{\partial^2 f}{\partial q_3 \partial q_4}\dot{q}_4) - cq_3 - p^2 b_1(a_1 q_2 + b_1 q_3 + c_1 q_4) = 0,\\
\frac{\partial \tilde{R}}{\partial q_4} &= p\,(\frac{\partial^2 f}{\partial q_2 \partial q_4}\dot{q}_2 + \frac{\partial^2 f}{\partial q_3 \partial q_4}\dot{q}_3 + \frac{\partial^2 f}{\partial q_4^2}\dot{q}_4) - aq_4 - p^2 c_1(a_1 q_2 + b_1 q_3\\
&\quad + c_1 q_4) = 0 \qquad (17)
\end{aligned}$$

and require that the obtained equations be dependent. To this end, we eliminate the velocities from the last three equations with the aid of the first three equations. As a result, we have the following conditions of degeneration for system (17)

$$\begin{aligned}
\frac{\partial \hat{R}}{\partial q_2} &= -p^2(\frac{1}{a_0}\frac{\partial^2 f}{\partial q_2^2}\frac{\partial f}{\partial q_2} + \frac{1}{c_0}\frac{\partial^2 f}{\partial q_2 \partial q_3}\frac{\partial f}{\partial q_3} + \frac{1}{b_0}\frac{\partial^2 f}{\partial q_2 \partial q_4}\frac{\partial f}{\partial q_4}\\
&\quad + a_1(a_1 q_2 + b_1 q_3 + c_1 q_4)) - bq_2 = 0,\\
\frac{\partial \hat{R}}{\partial q_3} &= -p^2(\frac{1}{a_0}\frac{\partial^2 f}{\partial q_2 \partial q_3}\frac{\partial f}{\partial q_2} + \frac{1}{c_0}\frac{\partial^2 f}{\partial q_3^2}\frac{\partial f}{\partial q_3} + \frac{1}{b_0}\frac{\partial^2 f}{\partial q_3 \partial q_4}\frac{\partial f}{\partial q_4}\\
&\quad + b_1(a_1 q_2 + b_1 q_3 + c_1 q_4)) - cq_3 = 0,\\
\frac{\partial \hat{R}}{\partial q_4} &= -p^2(\frac{1}{a_0}\frac{\partial^2 f}{\partial q_2 \partial q_4}\frac{\partial f}{\partial q_2} + \frac{1}{c_0}\frac{\partial^2 f}{\partial q_3 \partial q_4}\frac{\partial f}{\partial q_3} + \frac{1}{b_0}\frac{\partial^2 f}{\partial q_4^2}\frac{\partial f}{\partial q_4}\\
&\quad + c_1(a_1 q_2 + b_1 q_3 + c_1 q_4)) - aq_4 = 0. \qquad (18)
\end{aligned}$$

Also as above, we can consider the latter system as a PDE system for finding the function $f(q_2, q_3, q_4)$.

Having substituted the derivatives of quadratic form (16) into equations (18) and equated the coefficients of the generalized coordinates q_2, q_3, q_4 in the obtained expressions to zero, we have a system of quadratic equations for finding the unknowns d_i. This system under following denotations

$$\begin{aligned}
&\frac{d_1}{a_0} = x_1, \frac{d_2}{c_0} = y_1, \frac{d_3}{b_0} = z_1, \frac{d_4}{\sqrt{a_0 c_0}} = x_2, \frac{d_5}{\sqrt{a_0 b_0}} = z_2, \frac{d_6}{\sqrt{b_0 c_0}} = y_2, \frac{a_1 b_1}{\sqrt{a_0 c_0}} = A_1,\\
&\frac{b_1 c_1}{\sqrt{b_0 c_0}} = B_1, \frac{a_1 c_1}{\sqrt{a_0 b_0}} = C_1, \frac{b + a_1^2 p^2}{a_0 p^2} = A_2, \frac{c + b_1^2 p^2}{c_0 p^2} = B_2, \frac{a + c_1^2 p^2}{b_0 p^2} = C_2 \qquad (19)
\end{aligned}$$

writes

$$\begin{aligned}
&x_1 x_2 + x_2 y_1 + y_2 z_2 + A_1 = 0, \quad x_2^2 + y_1^2 + y_2^2 + B_2 = 0,\\
&y_1 y_2 + y_2 z_1 + x_2 z_2 + B_1 = 0, \quad x_1^2 + x_2^2 + z_2^2 + A_2 = 0,\\
&x_2 y_2 + x_1 z_2 + z_1 z_2 + C_1 = 0, \quad y_2^2 + z_1^2 + z_2^2 + C_2 = 0. \qquad (20)
\end{aligned}$$

Thus, in the case under consideration, the problem of finding the IMs, which correspond to the stationary values of Routh function $\tilde{R}$ (14), is reduced to solving the system of six algebraic quadratic equations with six variables $x_1, x_2, x_3, y_1, y_2, y_3$.

Using the Gröbner bases method, we have found the solutions of equations (20) under some conditions imposed on the problem parameters. These solutions in the initial parameters write

$$d_1 = -\frac{\sqrt{-a_0\varrho_1}}{\sqrt{3}p},\ d_2 = \pm\frac{c_0\sqrt{\varrho_1\varrho_2}}{3\sqrt{2a_0}\,a_1p^2},\ d_3 = \pm\frac{b_0\sqrt{\varrho_1\varrho_2}}{3\sqrt{2a_0}\,a_1p^2},\ d_4 = -\frac{\sqrt{-c_0\varrho_1}}{\sqrt{3a_0}p},$$
$$d_5 = -\frac{\sqrt{-b_0\varrho_1}}{\sqrt{3a_0}p}, d_6 = \mp\frac{\sqrt{b_0c_0\varrho_1\varrho_2}}{3\sqrt{2a_0}a_1p^2},\ b = \frac{2c_0\varrho_1\varrho_2}{9a_0a_1^2p^2},\ c = \frac{2b_0\varrho_1\varrho_2}{9a_0a_1^2p^2},$$
$$c_1 = \frac{\sqrt{b_0}\varrho_1}{3\sqrt{a_0}a_1p^2}, b_1 = \frac{\sqrt{c_0}\varrho_1}{3\sqrt{a_0}a_1p^2}.$$

Here $\varrho_1 = b + a_1^2p^2,\ \varrho_2 = b - 2a_1^2p^2$.

By IMs definition it can easily be verified that first three stationary equations (17) – for each found set of values d_i – define the invariant manifolds of the Routh equations corresponding to Routh functions R (13) and $\tilde{R}$ (14). These invariant manifolds can be "lifted up" into the phase space of Lagrange system (11). For this purpose, it is necessary to add relation (12) (the cyclic integral) to the IMs equations.

Likewise, the manifolds, which were obtained for the Routh equations, can be "lifted up" into the phase space of Lagrange system (15). In this case, the unknowns d_i of the Lagrange function should be replaced by the corresponding found values. Here the cyclic integral corresponding to the function $\tilde{L}$ should be added to the IMs equations.

The above example gives an insight on the computational complexity of the technique proposed. In the case of the Lagrange systems with $n > 2$ positional coordinates, we managed to find particular solutions for the equations of type (20) only.

3 The Case of the Nonlinear Routh System

Let us consider the Clebsh–Tisserand–Brun problem [3], [4]. In this case, the differential equations – under corresponding interpretation of the problem variables – describe both the motion of a rigid body in an ideal fluid (Clebsh's problem) and the motion of a rigid body with a fixed point in a potential force field of a special type (the Tisserand–Brun problem). The Lagrangian of this system in the Euler angles θ, φ, ψ writes

$$\begin{aligned}2L = {} & C(\dot\varphi + \cos\theta\dot\psi)^2 + A(\sin\varphi\sin\theta\dot\psi + \cos\varphi\,\dot\theta)^2 + B(\cos\varphi\sin\theta\dot\psi \\ & - \sin\varphi\,\dot\theta)^2 - \mu\sigma,\end{aligned} \tag{21}$$

where A, B, C are the inertial moments, μ is a constant of the force field, $\varrho = A\sin^2\varphi + B\cos^2\varphi,\ \sigma = C\cos^2\theta + \sin^2\theta\varrho$.

The system assumes the cyclic integral

$$\frac{1}{2}(A-B)\sin 2\varphi \sin\theta\,\dot\theta + C\cos\theta\dot\varphi + \sigma\dot\psi = p = \text{const}. \tag{22}$$

Let us state the problem to find the invariant manifolds of Lagrange system ([21]). For this purpose, we shall use the above approach.

According to the approach, we construct the Routh function

$$R = L - p\dot\psi = \frac{1}{2}C(1-\frac{C}{C+\varrho\tan^2\theta})\dot\varphi^2 + \frac{Cp\dot\varphi}{C\cos\theta+\varrho\sin\theta\tan\theta} + \frac{1}{4\sigma}[2(A-B)$$
$$p\sin 2\varphi\sin\theta\,\dot\theta - (A-B)C\sin 2\varphi\sin 2\theta\,\dot\varphi\,\dot\theta + 2(C\cos^2\theta(A\cos^2\varphi + B\sin^2\varphi)$$
$$+AB\sin^2\theta)\,\dot\theta^2 - 2p^2] - \frac{\mu\sigma}{2}$$

corresponding to L, and the "extended" Routh function

$$\tilde R = R + mf(\theta)\,\dot\theta, \qquad (m = \text{const})$$

where $f(\Theta) = dS(\Theta)/d\Theta$, $S(\Theta)$ is some smooth function.

In the given case, we add to R the full derivative of a function $S(\theta)$ depending on one variable only. Here $f(\theta) = dS(\Theta)/d\Theta$ is an unknown function, which should be determined.

Next write down the stationary conditions $\tilde R$ with respect to the phase variables

$$\frac{\partial\tilde R}{\partial\dot\varphi} = (1-\frac{C}{C+\varrho\tan^2\theta})\dot\varphi - \frac{(A-B)\sin 2\varphi\sin 2\theta}{4\sigma}\,\dot\theta + \frac{p}{C\cos\theta+\varrho\sin\theta\tan\theta}$$
$$= 0,$$

$$\frac{\partial\tilde R}{\partial\dot\theta} = \frac{1}{4\sigma}\Big[4(C\cos^2\theta(A\cos^2\varphi + B\sin^2\varphi) + AB\sin^2\theta)\dot\theta - (A-B)C\sin 2\varphi$$
$$\times\sin 2\theta\dot\varphi + 2(A-B)p\,\sin 2\varphi\sin\theta\Big] + mf(\theta) = 0,$$

$$\frac{\partial\tilde R}{\partial\varphi} = \frac{(A-B)C^2\sin 2\varphi\tan^2\theta}{2(C+\varrho\tan^2\theta)^2}\,\dot\varphi^2 - \frac{(A-B)Cp\sin 2\varphi\sin\theta\tan\theta}{(C\cos\theta+\varrho\sin\theta\tan\theta)^2}\dot\varphi$$
$$+\frac{(A-B)}{4\sigma}\Big[C((A-B)\sin^2 2\varphi\sin^2\theta - 2\sigma\cos 2\varphi)\sin 2\theta\,\dot\varphi\dot\theta - 2\sin 2\varphi$$
$$\times(C\cos^2\theta + A\sin^2\theta)(C\cos^2\theta + B\sin^2\theta)\,\dot\theta^2 + 2p\,\sin\theta(2\sigma\cos 2\varphi$$
$$-(A-B)\sin^2 2\varphi\sin^2\theta)\,\dot\theta\Big] + \frac{1}{2}(A-B)(\frac{p^2}{\sigma^2} - \mu)\sin 2\varphi\sin^2\theta = 0,$$

$$\frac{\partial\tilde R}{\partial\theta} = \frac{C^2\varrho\sec^2\theta\tan\theta}{(C+\varrho\tan^2\theta)^2}\dot\varphi^2 + \frac{Cp\,((C-\varrho)\sin\theta - \varrho\sec\theta\tan\theta)\dot\varphi}{(C\cos\theta+\varrho\sin\theta\tan\theta)^2} + \frac{(A-B)}{8\sigma^2}$$
$$\times\Big[4C\sin 2\varphi(\varrho\sin^2\theta - C\cos^2\theta)\dot\varphi\dot\theta - (A-B)C\sin^2 2\varphi\sin 2\theta\dot\theta^2$$
$$-4p\cos\theta\sin 2\varphi((\varrho-2C)\sin^2\theta - C\cos^2\theta)\dot\theta\Big] + \frac{(\mu\sigma^2-p^2)(C-\varrho)\sin 2\theta}{2\sigma^2}$$
$$+m\dot\theta f'(\theta) = 0. \tag{23}$$

and require the dependence of these equations.

To this end, we find the expressions

$$\dot{\varphi} = -\frac{\cot\theta[p\,((A-B)\cos 2\varphi + A + B)\csc\theta + (A-B)mf(\theta)\sin 2\varphi]}{2AB},$$
$$\dot{\theta} = \frac{m((A-B)\cos 2\varphi - (A+B))f(\theta) - (A-B)p\,\csc\theta\sin 2\varphi}{2AB} \tag{24}$$

for $\dot{\varphi}$, $\dot{\theta}$ from the first two equations (23) and substitute these expressions into the last two equations. As a result, we have

$$\begin{aligned}
&\Big[(AB\mu(4\cos 2\theta - \cos 4\theta) - 3AB\mu + 8p^2)\csc^2\theta - 8m^2f^2(\theta)\Big]\tan 2\varphi\\
&-16mp\,\csc\theta\, f(\theta) = 0,\\
&2m^2[(A-B)\cos 2\varphi - (A+B)]f(\theta)f'(\theta) - 2(A-B)mp\,\csc\theta\sin 2\varphi f'(\theta)\\
&+2(A-B)mp\cot\theta\csc\theta\sin 2\varphi f(\theta) + 2p^2(A+B+(A-B)\cos 2\varphi)\cot\theta\csc^2\theta\\
&+AB\mu(2C + (A-B)\cos 2\varphi - (A+B))\sin 2\theta = 0.
\end{aligned} \tag{25}$$

First equation (25) enables us to find the relation

$$\tan 2\varphi = -\frac{2mp\csc\theta f(\theta)}{m^2f^2(\theta) - p^2\csc^2\theta + AB\mu\sin^2\theta} \tag{26}$$

between the variables φ and θ.

Eliminate the variable φ from 2nd equation (25) with the aid of the above relation. The resulted equation can be considered as a differential equation with respect to $f(\theta)$. After its integration, we have

$$\begin{aligned}
&2(A-B)\sqrt{4m^2p^2\csc^2\theta f^2(\theta) + (m^2f^2(\theta) - p^2\csc^2\theta + AB\mu\sin^2\theta)^2} - 2(A+B)\\
&\times m^2f^2(\theta) + AB(A+B-2C)\mu\cos 2\theta - 2(A+B)p^2\cot^2\theta + 8C_1 = 0,
\end{aligned} \tag{27}$$

where C_1 is a constant of integration. We can use equation (27) for finding the function $f(\Theta)$.

Now we investigate the compatibility of equations (24), (26). To this end, eliminate the variable φ from equations (24) by relation (26). As a result, we have two equations with respect to $\dot{\theta}$, θ

$$\dot{\theta} = \frac{mf(\theta)[(A-B)(p^2\csc^2\theta + m^2f^2(\theta) + AB\mu\sin^2\theta) - (A+B)\rho_1]}{2AB\rho_1},$$

$$\begin{aligned}
&\frac{mp\cos\theta}{\rho_2}\Big[f(\theta)(p^2\csc^2\theta + m^2f^2(\theta) + 3AB\mu\sin^2\theta) + (p^2\csc^2\theta + m^2f^2(\theta)\\
&-AB\mu\sin^2\theta)f'(\theta)\Big]\,\dot{\theta} = -\frac{p\cot\theta\csc\theta}{2AB\rho}\Big[(A+B)\rho\\
&-(A-B)(p^2\csc^2\theta + m^2f^2(\theta) - AB\mu\sin^2\theta)\Big].
\end{aligned}$$

Here $\rho_1 = \sqrt{4m^2p^2\csc^2\theta f^2(\theta) + (m^2f^2(\theta) - p^2\csc^2\theta + AB\mu\sin^2\theta)^2}$,
$\rho_2 = p^2(p^2\csc^2\theta + 2m^2f^2(\theta) - 2AB\mu\sin^2\theta) + (m^2f^2(\theta) + AB\mu\sin^2\theta)^2$.

Having eliminated the variable $\dot\theta$ from the 2nd equation by the first one (here $f'(\theta)$ can be found from (27)), we obtain the following compatibility condition of the equations under investigation

$$\mu p\,\cos\theta\Big[C(p^2\csc^2\theta + m^2f^2(\theta) - ABC\mu\sin^2\theta$$
$$-(A-B)\sqrt{4m^2p^2\csc^2\theta f(\theta)^2 + (m^2f^2(\theta) - p^2\csc^2\theta + AB\mu\sin^2\theta)^2}\Big] = 0. \quad (28)$$

The latter equation can also be used for finding the function $f(\Theta)$.

Thus, we have: when conditions (27), (28) hold the expressions (24), (26) are a solution of the stationary equations.

Equations (27), (28) are compatible when the following conditions

$$\text{(i) } C = A + B,\ C_1 = \frac{(A+B)(AB\mu - 2p^2)}{8AB};\quad \text{(ii) } p = 0;\quad \text{(iii) } \mu = 0$$

hold. Let us consider these conditions in detail.

(i) In this case, equations (27), (28) are reduced to the equation

$$(A-B)\sqrt{4m^2p^2\csc^2\theta f^2(\theta) + (m^2f^2(\theta) - p^2\csc^2\theta + AB\mu\sin^2\theta)^2} + (A+B)$$
$$\times(p^2\csc^2\theta + m^2f^2(\theta) - AB\mu\sin^2\theta) = 0,$$

which has four solutions

$$f(\theta) = \pm\frac{\csc\theta[(A-B)\sqrt{\mu}\sin^2\theta + \eta]}{2m},\ f(\theta) = \pm\frac{\csc\theta[(A-B)\sqrt{\mu}\sin^2\theta - \eta]}{2m}.$$

Here $\eta = \sqrt{(A+B)^2\mu\sin^4\theta - 4p^2}$.

Having substituted the above values for $f(\theta)$ into equations (24), (26), we have obtained the following solutions

$$\dot\theta = \pm\frac{\csc\theta\sqrt{(A+B)^2\mu\sin^4\theta - 4p^2}}{A+B},$$
$$\varphi = \pm\frac{1}{2}\arctan\frac{2p}{\sqrt{(A+B)^2\mu\sin^4\theta - 4p^2}} \quad (29)$$

of stationary equations (23).

By IMs definition it can easily be verified that expressions (29) define the invariant manifolds of the Routh equations, which are the same for R and $\tilde R$.

In the initial phase space, the following invariant manifolds

$$\dot\theta = \pm\frac{\csc\theta\sqrt{(A+B)^2\mu\sin^4\theta - 4p^2}}{A+B},\ \dot\psi = \frac{2p\csc^2\theta}{A+B},$$
$$\varphi = \pm\frac{1}{2}\arctan\frac{2p}{\sqrt{(A+B)^2\mu\sin^4\theta - 4p^2}}$$

correspond to the found IMs. These IMs were obtained by addition of cyclic integral (22) (after the substitution of corresponding values $\dot\theta, \dot\varphi, \varphi$ (29) into this integral) to equations (29).

(ii) In this case, equation (28) turns into an identity, and equation (27) writes

$$4m^2 f^2(\theta) - A[\mu((A-B) + 2(B-C)\cos 2\theta) + 8C_1] = 0.$$

The latter equation has two solutions

$$f(\theta) = \pm\frac{\sqrt{A[\mu((A-B) + 2(B-C)\cos 2\theta) + 8C_1]}}{2m}.$$

Having substituted the values for $f(\theta)$ into (24), (26), we have the following solutions

$$\dot\theta = \pm\frac{\sqrt{\mu((A-B) + 2(B-C)\cos 2\theta) + 8C_1}}{2\sqrt{A}},\ \varphi = 0 \tag{30}$$

of stationary equations (23).

By IMs definition it can easily be verified that expressions (30) define the invariant manifolds of the Routh equations.

In the initial phase space, the following invariant manifolds

$$\dot\theta = \frac{\sqrt{\mu((A-B) + 2(B-C)\cos 2\theta) + 8C_1}}{2\sqrt{A}},\ \dot\psi = 0,\ \varphi = 0;$$
$$\dot\theta = -\frac{\sqrt{\mu((A-B) + 2(B-C)\cos 2\theta) + 8C_1}}{2\sqrt{A}},\ \dot\psi = 0,\ \varphi = 0 \tag{31}$$

correspond to the found IMs. These IMs were obtained by addition of cyclic integral (22) (after the substitution of corresponding values $\dot\theta, \dot\varphi, \varphi$ (30) into this integral) to equations (30).

When $C_1 = (2B - A - 2C)\mu/8$ equations (31) with respect to $\dot\theta$ are reduced to the equations

$$\dot\theta = \frac{\sqrt{(B-C)\mu}\cos\theta}{\sqrt{A}} \text{ and } \dot\theta = -\frac{\sqrt{(B-C)\mu}\cos\theta}{\sqrt{A}}, \quad \text{respectively.}$$

These equations can be integrated in the elementary functions

$$\theta = \arctan\Big[\tanh\Big(\frac{1}{2}\Big(\frac{\sqrt{(B-C)\mu}\,t}{\sqrt{A}} + C_2\Big)\Big)\Big],$$

where C_2 is a constant of integration.

Thus, we have obtained the family of solutions

$$\theta = \arctan\Big[\tanh\Big(\frac{1}{2}\Big(\frac{\sqrt{(B-C)\mu}\,t}{\sqrt{A}} + C_2\Big)\Big)\Big],\ \psi = \text{const},\ \varphi = 0,$$

which belongs to IMs (31) and satisfies the differential equations of Lagrange system (21).

(iii) When $\mu = 0$ the initial problem is reduced to the problem of the motion of a rigid body with a fixed point in Euler's case. Analysis of invariant manifolds, in this case, is similar to the above analysis of the Clebsh–Tisserand–Brun problem. The present paper does not discuss this case.

4 Conclusion

We have discussed a new approach for finding and qualitative analysis of invariant manifolds of nonlinear conservative Lagrange systems with cyclic coordinates, which by means of the Legendre transformation are reduced to the Routh systems. The proposed approach is based on the use of the "extended" Routh functions. The cases of both linear and nonlinear Routh systems have been considered for demonstrating the possibilities of the approach. A link of the considered technique with the Hamilton–Jacobi method has been revealed. It was also shown that the IMs obtained for the Routh equations with the help of our technique can be "lifted up" into the phase spaces of the corresponding Lagrange systems.

References

1. Irtegov, V.D., Titorenko, T.N.: On Reduction of Lagrange Systems. In: Gerdt, V.P., Koepf, W., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2010. LNCS, vol. 6244, pp. 123–133. Springer, Heidelberg (2010)
2. Gelfand, I.M., Fomin, S.V.: Variational Calculus. GIF-ML, Moscow (1961)
3. Stekloff, V.A.: Remarque sur un problème de Clebsch sur le mouvement d'un corps solide dans un liquide indefini et sur le problème de M. Brun. Comptes Rend. VCXXXV, 526–528 (1902)
4. Appel, P.: Traité de Mécanique Rationnelle, vol. 2. Gauthier–Villars, Paris (1953)

Construction of Explicit Optimal Value Functions by a Symbolic-Numeric Cylindrical Algebraic Decomposition

Hidenao Iwane[1], Akifumi Kira[2], and Hirokazu Anai[1,2]

[1] IT System Laboratories, Fujitsu Laboratories Ltd.
Kamikodanaka 4-1-1, Nakahara-ku, Kawasaki 211-8588, Japan
iwane@jp.fujitsu.com
[2] Graduate School of Mathematics, Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan
a-kira@math.kyushu-u.ac.jp, h.anai@kyudai.jp

Abstract. Recently parametric treatment of constraint solving and optimization problems has received considerable attention in science and engineering. In this paper we show an efficient and systematic algorithm for parametric programming, *i.e.* computing exact optimal value functions, based on a specialized symbolic-numeric cylindrical algebraic decomposition. We also present some practical application examples from system and control theory.

1 Introduction

Parametric approach to constraint solving and optimization problems has significant impact on many engineering applications. In recent years multi-parametric programming has received considerable attention in engineering and industry (see [19,20]). In particular, several special classes of multi-parametric optimization problems have been intensively studied together with the associated control problems such as model predictive control and process system engineering, see [13,4,12] for details.

Parametric optimization (parametric programming) is one of the key methodologies to analyze the effect of variations and uncertainty in system and control problems. The standard approach to parametric optimization problems is based on the sensitivity analysis theory, which provides solutions in the neighborhood of the nominal value of the varying parameters, see [6,14] for sensitivity analysis theory for optimization problems. This has been a main tool for solving multi-parametric programming problems, that is computing a complete map of the optimal solution in the space of varying parameters [10,20,19].

The algorithms based on the sensitivity analysis theory have been proposed for several important classes of multi-parametric optimization problems [19,10]. The algorithm for a *linear* case first finds an initial point to perform the linearization and achieves linearization. It is known that some classes of multi-parametric *quadratic* programming can be solved by reducing to the linear cases. The algorithm for multi-parametric *convex* nonlinear programming provides approximated optimal value functions and requires to execute iterative procedure at each step until a prescribed tolerance is satisfied. Through a systematic partition procedure of the parameter space, the algorithms enable us to get a complete map of the optimal solution as a conditional piecewise linear function of the parameters. Each piecewise linear function is derived from first-order estimation of the analytical nonlinear optimal function. For multi-parametric *nonconvex* nonlinear programming problems a branch and bound approach is usually employed. Its main idea is to construct convex parametric underestimators and overestimators of the objective function and then continue branch and bound until the difference between the underestimators and the overestimators, called *global parametric gap*, is within a certain given tolerance.

It has been observed that there exist significant gaps between obtained approximated optimal value functions derived by using the above mentioned (numerical) approaches and exact ones for multi-parametric nonlinear programming (even for convex cases). Moreover, for nonconvex cases construction of overestimators is a critical part in branch and bound approach and it is not done in a unified way. In fact there are several ways of constructing overestimators and they have different features in terms of three attributes: ease of obtaining, tightness, and function description (linear/nonlinear). These facts motivate us to develop an effective algorithm to construct *exact* solutions in a *systematic* way. The issues does need to be addressed.

Whilst a parameter space approach to robust control design using parametric optimization accomplished by a symbolic method "*quantifier elimination (QE)*" has been proposed (see [7] for QE). It has been successfully applied to practical control design applications, see for examples [11,15,21]. Moreover, some other approaches to utilize parametric optimization effectively in the context of bilevel/hieratical optimization are presented for control design problems [2,18]. QE-based methods have some remarkable efficacy in solving parametric programming problems since they can obtain exact feasible objective/parameter regions even for nonconvex and nonlinear cases. An algorithm for multi-parametric programming based on quantifier elimination is proposed in [1] and parametric optimization by cylindrical algebraic decomposition (CAD) is presented in [13]. These methods are indeed effective, however they have a bad computational complexity since they just utilizes QE/CAD as it is.

In this paper, because of this situation, we present an efficient algorithm for solving parametric optimization problem based on a specialized cylindrical algebraic decomposition using symbolic-numeric computation. The proposed method provides us exact optimal value functions even for nonlinear and also nonconvex problems. We have implemented all the algorithms proposed in this paper on Maple as a part of SyNRAC [21]. We show some computational examples including simple illustrative examples and a practical example from dynamic programming in order to demonstrate the effectiveness of our approach.

The rest of the paper is organized as follows: Section 2 is devoted to explain parametric optimization problems. In Section 3 we show our solution approach for solving a parametric optimization problem based on a specialized cylindrical algebraic decomposition which uses symbolic-numeric computation. Some computational examples are presented in Section 4. Concluding remarks and future direction are discussed in Section 5.

2 Parametric Optimization

A typical parametric optimization problem is given as follows:

$$\begin{array}{ll} \text{Minimize/Maximize} & f(\boldsymbol{x}, \boldsymbol{\theta}) \\ \text{subject to} & \boldsymbol{\theta} \in \mathcal{T}, \boldsymbol{x} \in \mathcal{P}_\theta, \end{array} \tag{1}$$

where $\mathcal{T} \subseteq \mathbb{R}^m$ and $\mathcal{P}_\theta \subseteq \mathbb{R}^n$ represent the feasible parameter space and the decision space, respectively. $f(\boldsymbol{x}, \boldsymbol{\theta})$ is a parametric nonlinear $\mathbb{R}$-valued function. The objective is to minimize (or maximize) f with respect to $\boldsymbol{x}$ for any value of given parameter $\boldsymbol{\theta}$ in $\mathcal{T}$, in other words, the solution of the problem (1) is derived as a function of the parameter $\boldsymbol{\theta}$.

In this paper, we restrict our attention to the case where f is given as a polynomial function, and $\mathcal{T}$ and $\mathcal{P}_\theta$ are given as semi-algebraic (possibly empty) sets. Since maximization of f is equivalent to minimization of the negative of f that is also polynomial, we focus on minimization without loss of generality. Let $\mathcal{T}'$ $(\subseteq \mathcal{T})$ be the set of all parameters such that $\mathcal{P}_\theta \neq \emptyset$, then, for any $\boldsymbol{\theta} \in \mathcal{T}'$, we define

$$f_{opt}(\boldsymbol{\theta}) := \inf_{\boldsymbol{x} \in \mathcal{P}_\theta} f(\boldsymbol{x}, \boldsymbol{\theta}), \quad \boldsymbol{\theta} \in \mathcal{T}'.$$

The function f_{opt} is called the *optimal value function*. Our goal is to find an explicit (exact) expression of the optimal value function.

2.1 Symbolic Approach to Parametric Optimization

We briefly explain how we can solve the parametric optimization problem (1) by using quantifier elimination.

If we assume that the infimum $f_{opt}(\boldsymbol{\theta})$ is attained as a minimum for every $\boldsymbol{\theta} \in \mathcal{T}'$, then we can construct a first-order formula ψ_{opt} with a free variable y that is true at $y = y_0$ if and only if $y_0 = f_{opt}(\boldsymbol{\theta})$:

$$\psi_{opt} = \exists \boldsymbol{x}(y = f(\boldsymbol{x}, \boldsymbol{\theta}) \wedge \boldsymbol{\theta} \in \mathcal{T} \wedge \boldsymbol{x} \in \mathcal{P}_\theta \wedge (\forall \boldsymbol{x}'(\boldsymbol{x}' \in \mathcal{P}_\theta \rightarrow y \leq f(\boldsymbol{x}', \boldsymbol{\theta})))).$$

By eliminating $\boldsymbol{x}$ from ψ_{opt} we obtain a quantifier-free formula $\xi_{opt}(\boldsymbol{\theta}, y)$, which expresses the optimal value function.

Next we consider the feasible objective region which includes the information of the optimal value function. We can construct a first-order formula with a free variable y that is true at $y = y_0$ if and only if y_0 is in the feasible objective region:

$$\psi_{feasible}(y, \boldsymbol{\theta}) = \exists \boldsymbol{x}(y = f(\boldsymbol{x}, \boldsymbol{\theta}) \wedge \boldsymbol{\theta} \in \mathcal{T} \wedge \boldsymbol{x} \in \mathcal{P}_\theta). \tag{2}$$

By eliminating $\boldsymbol{x}$ from $\psi_{feasible}$ we obtain a quantifier-free formula $\xi_{feasible}(\boldsymbol{\theta}, y)$. Then $y_{opt} = f_{opt}(\boldsymbol{\theta})$ is an infimum value for $\boldsymbol{\theta}$ in the feasible objective region. Obviously $\psi_{feasible}$ is easier to solve than ψ_{opt} by using QE.

3 Solution Approach Based on Cylindrical Algebraic Decomposition

In this section we show our special cylindrical algebraic decomposition algorithm for parametric optimization problems.

3.1 Cylindrical Algebraic Decomposition and Quantifier Elimination

Quantifier elimination (QE) is a powerful tool to resolve non-convex and parametric optimization problems exactly. Cylindrical algebraic decomposition (CAD), introduced by George E. Collins [8], is a general-purpose symbolic method aiming for QE. We briefly sketch the basic idea of CAD. Assume that we are given a prenex formula ψ with q free variables $x_1, \ldots, x_q$ and $(r-q)$ quantified variables $x_{q+1}, \ldots, x_r$

$$\psi(x_1, \ldots, x_q) \equiv \mathsf{Q}_{q+1}x_{q+1} \ldots \mathsf{Q}_r x_r \; \varphi(x_1, \ldots, x_r),$$

where $\mathsf{Q}_j \in \{\exists, \forall\}$ and φ is a quantifier-free formula. We can assume, by transposing terms if necessary, each atomic formula in φ is represented in the form $f \, \rho \, 0$, where f is a polynomial with rational coefficients on $x_1, \ldots, x_r$ and $\rho \in \{\leq, <, =, \neq\}$. Let $F \subseteq \mathbb{Q}[x_1, \ldots, x_r]$ be the set of polynomials appearing in φ as the left hand sides of atomic formulas. A subset $C \subseteq \mathbb{R}^r$ is said to be *sign-invariant* for F if every polynomial in F has a constant sign on all points in C. Then $\psi(S)$ is either "true" or "false" for all $S \in C$.

Suppose we have a finite sequence $\mathcal{D}_1, \ldots, \mathcal{D}_r$ for F which has the following properties:

1. Each $\mathcal{D}_i$ is a partition of $\mathbb{R}^i$ into finitely many connected semi-algebraic sets called *cells*.
2. $\mathcal{D}_{i-1}$, $1 < i \leq r$, consists exactly of the projections of all cells in $\mathcal{D}_i$ along the coordinate of the i-th variable in $(x_1, \ldots, x_r)$. For each cell $C \in \mathcal{D}_{i-1}$ we can determine its preimage $P(C) \subseteq \mathcal{D}_i$ under the projection.
3. Each cell $C \in \mathcal{D}_r$ is sign-invariant for F. Moreover for each cell $C \in \mathcal{D}_r$ we are given a *sample point* $S \in C$ in such a form that we can determine the sign of $f(S)$ for each $f \in F$ and thus evaluate $\varphi(S)$.

Then the partition $\mathcal{D}_r$ of $\mathbb{R}^r$ for F is called an *F-invariant cylindrical algebraic decomposition* of $\mathbb{R}^r$. A CAD algorithm computes such a sequence $\mathcal{D}_1, \ldots, \mathcal{D}_r$ and it consists of three phases; the *projection phase*, the *base phase*, and the *lifting phase*.

Projection phase: We first construct from $F \subseteq \mathbb{Q}[x_1, \ldots, x_r]$ a new finite set $F' \subseteq \mathbb{Q}[x_1, \ldots, x_{r-1}]$ that satisfies a special condition called "delineability", where the order of the real roots of all polynomials in F as univariate polynomials in x_r does not change above each connected cell in $\mathcal{D}_{r-1}$.

The step constructing F' from F is called a *projection* and denoted by $F' := \textsc{Proj}(F, x_r)$. We call polynomials in F' *projection polynomials* and their irreducible factors *projection factors*. Iterative application of the Proj operator leads to a finite sequence

$$F_r, \ldots, F_1, \quad \text{where } \; F_r := F, \; F_i := \textsc{Proj}(F_{i+1}, x_{i+1})$$

for $1 \leq i < r$. The Proj operator, in general, computes certain coefficients, discriminants, and resultants derived from polynomials in F_{i+1} and their higher derivatives by regarding those as univariate polynomials in x_{i+1}. The final set F_1 consists of univariate polynomials in x_1.

Base phase: Then we construct a partition $\mathcal{D}_1$ of the real line $\mathbb{R}^1$ into finitely many intervals that are sign-invariant for F_1. This step is called the *base phase*, achieved by isolating the real zeros of the univariate polynomials in F_1.

Lifting phase: The partitions $\mathcal{D}_i$ of $\mathbb{R}^i$ for $2 \leq i \leq r$ are computed recursively: The roots of all polynomials in F_i as univariate polynomials in x_i are delineated above each connected cell $C \in \mathcal{D}_{i-1}$. Thus we can cut the *cylinder* above C into finitely many connected semi-algebraic sets (cells) called *stack*. This is done by real root isolation of the univariate polynomials derived through specializing the polynomials in F_i by a sample point of C. Then $\mathcal{D}_i$ is a collection of all such cells obtained from all cylinders above the cells of $\mathcal{D}_{i-1}$.

A finite sequence $\mathcal{D}_1, \ldots, \mathcal{D}_r$ for F has a tree structure: The first level of nodes under the root of the tree corresponds to the cells in $\mathcal{D}_1$. The second level of nodes stands for the cells in $\mathcal{D}_2$, i.e., the cylinders over the cells of $\mathbb{R}^1$. The leaves represent the cells of $\mathcal{D}_r$, i.e., a CAD of $\mathbb{R}^r$. A sample point of each cell is stored in its corresponding node or leaf. At each level of the tree there are a number of projection polynomials F_i whose signs define a cell when evaluated over a sample point.

3.2 Parametric Optimization by a Specialized CAD

A feasible objective region of a given parametric optimization problem can be expressed by a first-order formula (2). Since $y_{opt} = f_{opt}(\boldsymbol{\theta})$ is the infimum value, we can obtain $f_{opt}(\boldsymbol{\theta})$ from $\psi_{feasible}$. A strategy is to first construct a CAD of the feasible objective region, then for each stack over a cell of the parameter space, pick the highest sections which is lower than or equal to the true cells. Our spacial CAD algorithm is more efficient than to the above strategy because we can avoid lifting more cells.

Then we explain our special CAD algorithm which computes directly an optimal value function from (2). In order to compute the optimal value function for parameters $\boldsymbol{\theta}$, the variable order of CAD is set to be $(\boldsymbol{\theta}, y, \boldsymbol{x})$. Our algorithm is based on a partial CAD algorithm [9] and the only difference is evaluation of truth value of cells. Let $C \subseteq \mathcal{T} \subseteq \mathbb{R}^m$ be a cell and $S \in \mathbb{R}^m$ be a sample point for C. The children of C are denoted by $c_1, \ldots, c_k$ and the $(m+1)$-st coordinates of their sample points are denoted by $y_1, \ldots, y_k \in \mathbb{R}$. When y_p is the infimum value of the feasible objective region for $\boldsymbol{\theta}$, p is an even integer and p is less than or equal to j such that $\xi_{feasible}(S, y_j)$ is true. We can avoid lifting cells from these properties.

- If the truth value of c_i for $\psi_{feasible}$ is true then we can set the truth value of c_j to false for all $j = i+1, \ldots, k$.
- If c_i is a sector cell, i.e., i is an odd integer, and its truth value for $\psi_{feasible}$ is true, then we can set the truth value of c_i and c_{i-1} to false and true, respectively. Note that from the first property the truth value of c_{i-1} is reset to false when c_{i-1} is not the infimum value.

The following algorithm utilizes the ideas discussed above:

Algorithm: a special CAD for solving a parametric optimization problem.

Input : parametric optimization problem:

$$\begin{array}{ll} \text{Minimize} & f(\boldsymbol{x}, \boldsymbol{\theta}) \\ \text{subject to} & \varphi(\boldsymbol{x}, \boldsymbol{\theta}) \end{array}$$

Output: optimal value function

```
⟨1⟩ [Formulation] $\psi(\boldsymbol{x}, y, \boldsymbol{\theta}) = \exists \boldsymbol{x}(y = f(\boldsymbol{x}, \boldsymbol{\theta}) \wedge \varphi(\boldsymbol{x}, \boldsymbol{\theta}))$.
⟨2⟩ [Projection] Compute the projection factors.
⟨3⟩ [Initialization] Initialize a list $L$ as the cells of $\mathcal{D}_1$.
⟨4⟩ [Choice] If $L$ is empty, go to step ⟨9⟩, otherwise, choose a cell $c_i$ from $L$.
⟨5⟩ [Optimal Value] If the truth value of $c_i$ is undetermined, go to step ⟨6⟩. If
    the truth value is true and its dimension is $m+1$ then for all cell $c_j$ within
    same stack such that $j > i$ the truth value of $c_j$ is set to be false, and if $c_i$ is
    a sector cell then the truth value of $c_i$ is set to be false and the truth value
    of $c_{i-1}$ is set to be true. Go to step ⟨4⟩.
⟨6⟩ [Stack Construction] Construct a stack over $c_i$ and from a sample point for
    each child cell. Insert all of the children in $L$.
⟨7⟩ [Trial Evaluation] Try to determine the truth value of $\psi$ for the children of
    $c_i$ by using a partial CAD approach.
⟨8⟩ [Propagation] Determine the truth value of as ancestors of $c_i$ as possible by
    using a partial CAD approach and insert all of the cell determined truth
    value in this step in $L$.
⟨9⟩ [Solution Formula Construction] Construct a quantifier-free formula from
    the CAD.
```

We have implemented our proposed method combined with a symbolic-numeric CAD approach [16,17] as a part of SyNRAC [21].

4 Computational Examples

Here we show computational results for some example problems. We solved all the QE problems in the examples by using SyNRAC on a PC with Intel(R) Core(TM) i3 CPU U330 1.20 GHz and 2.92 GByte of memory and timing data are all given in second (CPU-time).

4.1 Illustrative Examples

Example 1: First consider the following nonlinear parametric optimization problem [17]:

$$\begin{array}{ll}\text{Minimize} & g(x_1, x_2, \theta) \equiv 45\theta^2 + 80\theta x_1 + 120\theta + x_2 - 43x_1^2 - 70x_1x_2 - 78x_2^2 \\ \text{subject to} & \theta \geq x_1 + x_2, x_1 \geq 0, x_2 \geq 0, \\ & 15\theta \geq 10x_1 + 19x_2 + 1000000.\end{array}$$

This can be recast as the following first-order formula:

$$\begin{array}{l}\exists x_1 \exists x_2\ (y = g(x_1, x_2, \theta) \wedge \theta \geq x_1 + x_2 \wedge x_1 \geq 0 \wedge x_2 \geq 0 \wedge \\ \qquad 15\theta \geq 10x_1 + 19x_2 + 1000000).\end{array} \tag{3}$$

By performing QE for (3) we obtain the quantifier-free formula $\xi_{feasible}(\theta_1, \theta_2, y)$. Actually by using SyNRAC for the QE computation we obtain

$$\begin{array}{rl}\xi_{feasible}(\theta, y) \equiv & (3\theta \geq 20000 \wedge 4y \leq 273\theta^2 + 1960480\theta - 17200000000 \wedge \\ & y \geq 45\theta^2 + 120\theta) \vee (312y \leq 14040\theta^2 + 374440\theta + 1 \wedge \\ & 361y \geq -1305\theta^2 + 2340436605\theta - 780001900000 \wedge \\ & 2340\theta \geq 15600019) \vee (43y \leq 3535\theta^2 + 5160\theta \leq 0 \wedge \\ & 312y \geq 14040\theta^2 + 374440\theta + 1 \wedge 49\theta \geq 860000)\end{array}$$

in 7.54 seconds and 50,393 cells appears during the QE computation. The feasible region $\xi_{feasible}(\theta, y)$ corresponds to the gray part in Figure 1.

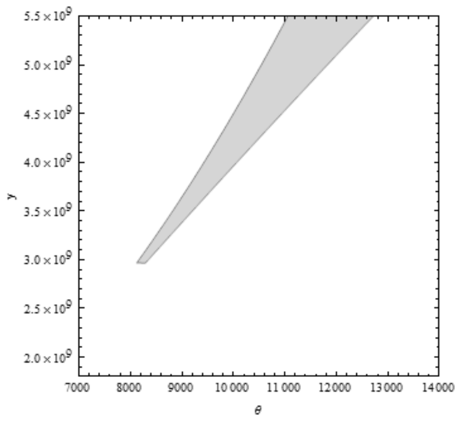


Fig. 1. Feasible region $\xi_{feasible}(\theta, y)$

By using our proposed method we can obtain the following optimal value function:

$$\begin{array}{rl}\xi_{opt}(\theta, y) \equiv & (\frac{20000}{3} \leq \theta \leq \frac{7800019}{1170} \wedge y = 45\theta^2 + 120\theta) \vee \\ & (\theta \geq \frac{7800019}{1170} \wedge 361y = -1305\theta^2 + 2340436605\theta - 780001900000),\end{array}$$

in 3.51 seconds. Our method is much faster than the conventional QE approach and we observe that the number of occurring cells is greatly reduced to 17,777.

Example 2: Next we consider the following nonlinear parametric optimization problem [19, p. 31]:

$$\begin{array}{ll} \text{Minimize} & f(x_1, x_2) \equiv x_1^3 + 2x_1^2 - 5x_1 + 2x_2^2 - 3x_2 - 6 \\ \text{subject to} & 2x_1 + x_2 \leq 2.5 + \theta_1, \\ & 0.5x_1 + x_2 \leq 1.5 + \theta_2, \\ & x_1 \geq 0, x_2 \geq 0, 0 \leq \theta_1 \leq 0.25, 0 \leq \theta_2 \leq 0.25. \end{array}$$

This problem can be formulated as the following first-order formula:

$$\begin{array}{ll} \exists x_1 \exists x_2 & (y = f(x_1, x_2) \wedge 2x_1 + x_2 \leq 2.5 + \theta_1 \wedge \\ & 0.5x_1 + x_2 \leq 1.5 + \theta_2 \wedge \\ & x_1 \geq 0 \wedge x_2 \geq 0 \wedge 0 \leq \theta_1 \leq 0.25 \wedge 0 \leq \theta_2 \leq 0.25). \end{array} \tag{4}$$

By performing QE for (4) we obtain the quantifier-free formula $\xi_{feasible}(\theta_1, \theta_2, y)$. By using SyNRAC for the QE computation we obtain

$$\begin{array}{rl} \xi_{feasible}(\theta_1, \theta_2, y) \equiv & 1728y^2 + 11056y - 47349 \leq 0 \wedge y \leq 2\theta_2^2 + 3\theta_2 - 6 \wedge \\ & 0 \leq \theta_1 \leq \frac{1}{4} \wedge 0 \leq \theta_2 \leq \frac{1}{4}, \end{array}$$

in 86.03 seconds and the number of cells occurring in the QE computation is 325,613. By using our proposed method we can obtain the following optimal value function:

$$\xi_{opt}(\theta_1, \theta_2, y) \equiv 1728y^2 + 11056y - 47349 = 0 \wedge y \leq -6 \wedge 0 \leq \theta_1 \leq \frac{1}{4} \wedge 0 \leq \theta_2 \leq \frac{1}{4},$$

in 7.69 seconds. The number of cells occurring in the QE computation decreases to 35,825.

4.2 Dynamic Programming

Dynamic programming (DP), proposed by Bellman [3], is widely used in many optimization problems as a technique for breaking down a large problem into a sequence of much smaller problems. Especially, it works well for multi-stage decision problems that are generally considered to be a representation of real-world situations where a sequence of decisions are made to minimize the total sum of stage-wise costs. A typical N-stage decision problem of a deterministic case has the following structure: If the process is in state $x_n \in \mathcal{X}_n$ at stage n, and decision $u_n \in \mathcal{U}_n(x_n)$ is chosen, where $\mathcal{U}_n(x)$ represents the set of all feasible decisions in state x at stage n, then we incur a cost $c_n(x_n, u_n)$, and the process goes deterministically to the next state $x_{n+1} = f_n(x_n, u_n) \in \mathcal{X}_{n+1}$. Hence, the problem is formulated as follows:

$$\begin{array}{lll} & \text{Minimize} & \sum_{n=0}^{N-1} c_n(x_n, u_n) + c_N(x_N) \\ \mathrm{P}(x_0) & \text{subject to} & x_{n+1} = f_n(x_n, u_n), \quad n = 0, 1, \ldots, N-1 \\ & & u_n \in U_n(x_n), \quad n = 0, 1, \ldots, N-1\,. \end{array}$$

Now, we assume that $\mathrm{P}(x_0)$ has at least one optimal solution for each x_0 (e.g., cases obeying Weierstrass' Theorem), and let $v_0(x)$ be the optimal value of $\mathrm{P}(x)$ for any initial state x, then we can get the optimal value function $v_0(x)$ by recursively computing the following DP equation backward through time:

$$v_N(x) = c_N(x), \quad x \in \mathcal{X}_N \tag{5a}$$

$$v_n(x) = \min_{u \in \mathcal{U}_n(x)} \left\{ c_n(x,u) + v_{n+1}(f_n(x,u)) \right\},\ x \in \mathcal{X}_n,\ n = 0, \ldots, N-1\,. \tag{5b}$$

In each step of the iteration, we thus have to solve the parametric optimization problem, specified by the right-hand side of (5b), consisting of a single decision variable. However, though the dynamic programming proves the validity of this functional equation, an answer to the question whether the equation is tractable at all, and/or what techniques will be required to solve it, will depend entirely on the properties of the functions $\{c_n\}$ and $\{f_n\}$. For this reason, DP computer codes have restricted their attention to problems with finite solution sets, and it is usually the case that problems with continuous decision variables have to be solved numerically, based on discrete approximation. At this point, we attempt to support the dynamic programming procedure, for continuous problems, with our special CAD algorithm. This collaboration scheme leads a successful solution for a wide class of problems described as polynomials.

To illustrate how the special CAD algorithm, in conjunction with dynamic programming, solves the multi-stage decision problems, we take the optimal temperature control problem, adopted from Bertsekas [5] and Pistikopoulos et al. [19], for example. Pistikopoulos et al. [19] pointed out that constraints (6e) and (6f), additionally inserted, impede a successful solution of the DP equation.

$$\begin{aligned}
\text{Minimize} \quad & J = \textstyle\sum_{n=0}^{2} u_n^2 + 100(x_3 - 1500)^2 && \text{(6a)}\\
\text{subject to} \quad & x_1 = 0.55x_0 + 0.45u_0, && \text{(6b)}\\
& x_2 = 0.60x_1 + 0.40u_1, && \text{(6c)}\\
& x_3 = 0.65x_2 + 0.35u_2, && \text{(6d)}\\
& 200 \le x_1 \le 400, && \text{(6e)}\\
& 500 \le x_2 \le 1000, && \text{(6f)}\\
& 0 \le u_0, u_1, u_2 \le 3000\,. && \text{(6g)}
\end{aligned}$$

A certain material is passed through a sequence of three ovens in a kiln (see Figure 2). We denote

x_0: initial temperature of the material,
x_n, $n = 1, 2, 3$: temperature of the material at the exit of oven n,
u_{n-1}, $n = 1, 2, 3$: prevailing temperature in oven n,

and their heat transfer phenomena are assumed as in (6b)-(6d). Additionally, to make a good product, we respect the path constraints (6e) and (6f). Our goal is to get the final temperature close to a given target $T = 1500$, while expending relatively little energy. Hence the objective function is formulated as in (6a).

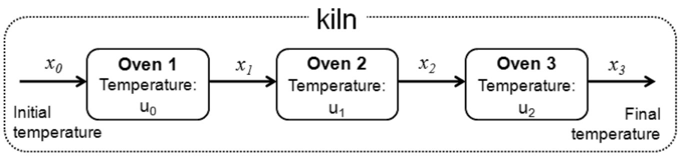


Fig. 2. Schematic diagram of the ceramic kiln

Of course, we may apply the special CAD algorithm directly to this problem. However, the computation of QE on this problem did not terminate in an hour. By using the collaboration scheme, we can effectively reduce the volume of computation required to solve the problem. At first, the initial condition for the dynamic programming equation is given by

$$v_3(x_3) = 100(x_3 - 1500)^2 \qquad (-\infty < x_3 < \infty)\,. \tag{7}$$

For the next-to-last stage, we have

$$\mathrm{P}_2(x_2) \quad \begin{array}{l} \text{Minimize } \; u_2^2 + v_3(x_3) \\ \text{subject to } x_3 = 0.65x_2 + 0.35u_2, \; 0 \le u_2 \le 3000\,, \end{array}$$

where $v_3(x_3)$ is given by (7). We minimize the objective function with respect to u_2 for each x_2 in $[500, 1000]$. Applying the special CAD to this problem yields

$$v_2(x_2) = \begin{cases} \frac{169}{4}x_2^2 - 58500x_2 + 29250000 & (500 \le x_2 \le \frac{51000}{91}) \\ \frac{169}{53}x_2^2 - \frac{780000}{53}x_2 + \frac{900000000}{53} & (\frac{51000}{91} < x_2 \le 1000)\,, \end{cases}$$

and hence we obtain the optimal control law for the last oven as follows:

$$u_2^* = \pi_2^*(x_2) = \begin{cases} 3000 & (500 \le x_2 \le \frac{51000}{91}) \\ -\frac{91}{53}x_2 + \frac{210000}{53} & (\frac{51000}{91} < x_2 \le 1000)\,. \end{cases}$$

Next, we consider the parametric problem at stage $n = 1$ below, to evaluate $v_1(x_1)$ for each x_1 in $[200, 400]$, using v_2 obtained in the previous step.

$$\mathrm{P}_1(x_1) \quad \begin{array}{l} \text{Minimize } \; u_1^2 + v_2(x_2) \\ \text{subject to } x_2 = 0.60x_1 + 0.40u_1, \; 0 \le u_1 \le 3000, \\ \qquad\qquad\quad 500 \le x_2 \le 1000\,. \end{array}$$

Similarly, applying the special CAD algorithm to this subproblem, we obtain

$$v_1(x_1) = \frac{507}{667}x_1^2 - \frac{3900000}{667}x_1 + \frac{7500000000}{667}\,,$$
$$u_1^* = -\frac{338}{667}x_1 + \frac{1300000}{667}\,.$$

Finally, we determine the value of $v_0(x_0)$ for each x_0 in $\mathbb{R}$.

$$\mathrm{P}_0(x_0)\quad \begin{array}{l} \text{Minimize } \ u_0^2 + v_1(x_1) \\ \text{subject to } x_1 = 0.55x_0 + 0.45u_0,\ 0 \le u_0 \le 3000, \\ \qquad\qquad 200 \le x_1 \le 400\,. \end{array}$$

Applying the special CAD algorithm again, we arrive at

$$v_0(x_0) = \begin{cases} \frac{121}{81}{x_0}^2 - \frac{88000}{81}x_0 + \frac{556634680000}{54027} & (-\frac{23000}{11} \le x_0 \le -\frac{4818830}{7337}) \\ \frac{61347}{307867}{x_0}^2 - \frac{858000000}{307867}x_0 + \frac{3000000000000}{307867} & (-\frac{4818830}{7337} < x_0 \le -\frac{1740160}{7337}) \\ \frac{121}{81}{x_0}^2 - \frac{176000}{81}x_0 + \frac{530398720000}{54027} & (-\frac{1740160}{7337} < x_0 \le \frac{8000}{11}), \end{cases}$$

$$u_0^* = \pi_0^*(x_0) = \begin{cases} -\frac{11}{9}x_0 + \frac{4000}{9} & (-\frac{23000}{11} \le x_0 \le -\frac{4818830}{7337}) \\ -\frac{50193}{307867}x_0 + \frac{351000000}{307867} & (-\frac{4818830}{7337} < x_0 \le -\frac{1740160}{7337}) \\ -\frac{11}{9}x_0 + \frac{8000}{9} & (-\frac{1740160}{7337} < x_0 \le \frac{8000}{11})\,. \end{cases}$$

This completes the solution of the problem. We obtained the desired exact optimal value function and the exact optimal control policy for the three ovens. The computation time for each iteration is less than one second.

5 Conclusion

We have presented an efficient and systematic algorithm for constructing exact optimal value functions of parametric optimizations based on a specialized symbolic-numeric cylindrical algebraic decomposition. The effectiveness of our approach is demonstrated by computational examples from some applications.

As the example from dynamic programming in this paper indicates, the combined use of our symbolic-numeric method with exploiting the structure of problems is a promising direction for reducing the total amount of computation.

Acknowledgements. This work was partially supported by Grant-in-Aid for Scientific Research (B) No. 21340025, Grant-in-Aid for JSPS Fellows No. 09J05487 and Global COE Program "Math-for-Industry" (Kyushu University).

References

1. Anai, H.: A symbolic-numeric approach to multi-parametric programming for control design. In: Proc. ICROS-SICE International Conference 2009, pp. 3525–3530 (2009)
2. Anai, H., Hara, S., Kanno, M., Yokoyama, K.: Parametric polynomial spectral factorization using the sum of roots and its application to a control design problem. J. Symb. Comput. 44(7), 703–725 (2009)
3. Bellman, R.: Dynamic Programming. Princeton Univ. Press, Princeton (1957)
4. Bemporad, A., Morari, M., Dua, V., Pistikopoulos, E.N.: The explicit linear quadratic regulator for constrained systems. Automatica 38(1), 3–20 (2002)

5. Bertsekas, D.: Dynamic Programming and Optimal Control, 3rd edn. Athena Scientific, Belmont (2005)
6. Bonnans, J.F., Shapiro, A.: Perturbation Analysis of Optimization Problems. Springer, Heidelberg (2000)
7. Caviness, B., Johnson, J. (eds.): Quantifier Elimination and Cylindrical Algebraic Decomposition. Texts and monographs in symbolic computation. Springer, Heidelberg (1998)
8. Collins, G.E.: Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In: Becvar, J. (ed.) MFCS 1975. LNCS, vol. 32. Springer, Heidelberg (1975)
9. Collins, G.E., Hong, H.: Partial cylindrical algebraic decomposition for quantifier elimination. Journal of Symbolic Computation 12(3), 299–328 (1991)
10. Domingueza, L.F., Narcisoa, D.A., Pistikopoulos, E.N.: Recent advances in multiparametric nonlinear programming. Computers & Chemical Engineering 34(5), 707–716 (2010); (Selected Paper of Symposium ESCAPE 19, June 14-17, 2009, Krakow, Poland)
11. Dorato, P., Yang, W., Abdallah, C.: Robust multi-objective feedback design by quantifier elimination. J. Symb. Comput. 24(2), 153–159 (1997)
12. Dua, P., Kouramas, K., Dua, V., Pistikopoulos, E.: MPC on a chip – recent advances on the application of multi-parametric model-based control. Computers & Chemical Engineering 32(4-5), 754–765 (2008)
13. Fotiou, I.A., Rostalski, P., Parrilo, P.A., Morari, M.: Parametric optimization and optimal control using algebraic geometry methods. International Journal of Control 79(11), 1340–1358 (2006)
14. Fiacco, A.V.: Introduction to Sensitivity and Stability Analysis in Nonlinear Programming. Academic Press, London (1983)
15. Hyodo, N., Hong, M., Yanami, H., Hara, S., Anai, H.: Solving and visualizing nonlinear parametric constraints in control based on quantifier elimination. Appl. Algebra Eng. Commun. Comput. 18(6), 497–512 (2007)
16. Iwane, H., Yanami, H., Anai, H., Yokoyama, K.: An effective implementation of a symbolic-numeric cylindrical algebraic decomposition for quantifier elimination. In: Proceedings of the 2009 International Workshop on Symbolic-Numeric Computation, vol. 1, pp. 55–64 (2009)
17. Iwane, H., Yanami, H., Anai, H.: An effective implementation of a symbolic-numeric cylindrical algebraic decomposition for optimization problems. In: Proceedings of the 2011 International Workshop on Symbolic-Numeric Computation (2011)
18. Kanno, M., Yokoyama, K., Anai, H., Hara, S.: Parametric optimization in control using the sum of roots for parametric polynomial spectral factorization. In: Wang, D. (ed.) ISSAC, pp. 211–218. ACM, New York (2007)
19. Pistikopoulos, E., Georgiadis, M., Dua, V. (eds.): Multi-parametric programming: theory, algorithms, and applications, vol. 1. Wiley-VCH, Chichester (2007)
20. Pistikopoulos, E., Georgiadis, M., Dua, V. (eds.): Multi-parametric model-based control: theory and applications, vol. 2. Wiley-VCH, Chichester (2007)
21. Yanami, H., Anai, H.: The Maple package SyNRAC and its application to robust control design. Future Generation Comp. Syst. 23(5), 721–726 (2007)

Convection in a Porous Medium and Mimetic Scheme in Polar Coordinates

Bülent Karasözen[1], Anastasia Trofimova[2], and Vyacheslav Tsybulin[2]

[1] Department of Mathematics & Institute of Applied Mathematics, Middle East Technical University, Ankara, Turkey
bulent@metu.edu.tr
[2] Southern Federal University, Rostov-on-Don, Russia
trofimova.anastasia@gmail.com, tsybulin@math.rsu.ru

Abstract. Analytical investigation of natural convection of the incompressible fluid in the porous media based on the Darcy hypothesis (Lapwood convection) gives intriguing branching off of one-parameter family of convective patterns. This scenario may be suppressed in computations when governing equations are approximated by schemes which do not preserve the cosymmetry property. We consider the problem in polar coordinates and construct a mimetic finite-difference scheme using computer algebra tools. The family of steady states is computed and it is demonstrated that this family disappears under non-mimetic approximation.

Introduction

Interest in fluid convection in a porous medium is driven by different applications in geophysics and energetics [1]. Because many studies are based on direct simulation the numerical schemes must be robust and it should inherit the key properties of underlying equations (in particular, conservation laws). The mimetic finite-difference methods that provide the fundamental identities of vector calculus have been successfully employed [2]. Inheritance of continuous and discrete symmetries of the underlying system is also important requirement for the numerical scheme. Among these properties, cosymmetry [3] plays an important role in Darcy or Lapwood convection. D. Lyubimov [4] found the branching off a family of steady states from the state of rest and V. Yudovich [3] explained this phenomenon using cosymmetry theory. It should be noted that the members of cosymmetric family of steady convective patterns have different spectrum of stability whenever in the symmetry situation all members of the family have the identical spectra. Numerical studies for families of steady states in Darcy convection were carried out only for the case of rectangular enclosures and cartesian coordinates, see [5] and references here.

We derive a mimetic scheme to compute the family of steady convective fluid patterns in annular enclosure filled with a porous medium. Equations for stream function and temperature in the polar coordinates is discretized by the finite-difference method. Special attention is given to the approximation of the Jacobian as well as the buoyancy terms. This point was supported by computer algebra system Maple both for manipulation with nonlinear terms and realization of free parameters method. Derived scheme preserves the cosymmetry and discrete symmetry of the problem.

The paper is organized as follows. The governing equations in polar coordinates are presented in Section 1 and emphasizing the importance of the cosymmetry property of the problem. In Section 2, the mimetic finite difference scheme is developed. The numerical results given in Section 3 demonstrate the existence of the continuous family of steady states in an annular enclosures. Destruction of the family of steady states was observed in case of non-conservative schemes.

1 Governing Equations and the Cosymmetry

We consider an incompressible fluid which saturates the porous medium in a annular enclosure $D = [R_1, R_2] \times [\Phi_1, \Phi_2]$ heating from below. The temperature on the boundary is given by a linear function of the height.

The planar problem of Darcy convection in polar coordinates (r, φ) is governed by dimensionless equations:

$$\partial_t \theta = \Delta\theta + G(\psi) - J(\theta, \psi) \equiv F_1(\theta, \psi), \tag{1}$$

$$0 = \Delta\psi - \lambda G(\theta) \equiv F_2(\theta, \psi), \quad \Delta = \partial_r^2 + \frac{1}{r}\,\partial_r + \frac{1}{r^2}\,\partial_\varphi^2, \tag{2}$$

where the Jacobian operator $J(\theta, \psi)$ and the term $G(\theta)$ are given:

$$J(\theta, \psi) = \frac{1}{r}\left[\partial_r(\theta\,\partial_\varphi\,\psi) - \partial_\varphi(\theta\,\partial_r\,\psi)\right], \tag{3}$$

$$G(\theta) = \frac{1}{r}\left[\partial_\varphi(\cos\varphi\,\theta) + \partial_r(r\sin\varphi\,\theta)\right]. \tag{4}$$

The dependent variables $\psi(r, \varphi, t)$ and $\theta(r, \varphi, t)$ denote, respectively, the stream function and the perturbations of temperature from a linear conductive profile. The Rayleigh number is given by $\lambda = g\beta l^2 K\delta\theta/\nu\chi$, here g is acceleration due to gravity, β is the thermal expansion coefficient, ν is the kinematic viscosity, K is the permeability coefficient, χ is the thermal diffusivity of the fluid, l is the length parameter.

The boundary conditions are:

$$\theta = 0, \quad \psi = 0, \qquad on \quad \partial D. \tag{5}$$

The initial condition is only defined for the temperature, $\theta(r, \varphi, 0) = \theta_0(r, \varphi)$, because the stream function can be expressed from (2) and (5).

Equations (1)–(5) impose the equilibrium $\theta = \psi = 0$ (state of rest), which remains stable while $\lambda < \lambda_{cr}$. It was shown in [3] that the first critical value λ_{cr} has multiplicity of two for any domain D and onset of the family of steady states occurs when λ passes λ_{cr}. This is a consequence of the linear cosymmetry that exists for the system (1)–(5). It is easy to check that the pair $(\psi, -\theta)$ means the cosymmetry and its orthogonality to the right-hand sides of (1)–(2). We multiply (1) by ψ and (2) by $-\theta$, sum them and integrate over the domain D. Then using Greens formula, integration by parts and taking into account the boundary conditions (5) we obtain

$$\int\limits_D \Big[F_1(\theta,\psi)\psi - F_2(\theta,\psi)\theta \Big] r dr d\varphi = 0. \tag{6}$$

We stress that the identity (6) holds because the Jacobian and the buoyancy term obey the following equalities

$$I_{J\psi} = \int\limits_D J(\theta,\psi)\psi \, r dr d\varphi = 0, \tag{7}$$

$$I_{G\theta} = \int\limits_D G(\theta)\theta \, r dr d\varphi = 0. \tag{8}$$

It should be noted that the finite-difference approximation of the Jacobian must preserve the finite-dimensional analog of (7) and this is directly connected with cosymmetry conservation. It is desirable also to require that the Jacobian approximation nullifies the discrete analog of the integral

$$I_{J\theta} = \int\limits_D J(\theta,\psi)\theta \, r dr d\varphi = 0, \tag{9}$$

and preserve the skew-symmetry of the Jacobian $J(\theta,-\psi) = -J(\psi,\theta)$.

The problem has a discrete symmetry if $\Phi_2 = 2\pi - \Phi_1$, then the equations (1)–(2) are invariant with respect to the transformation

$$R^{\varphi} : \{\varphi,\theta,\psi\} \mapsto \{2\pi - \varphi, \theta, -\psi\}. \tag{10}$$

2 Finite–Difference Scheme

Mimetic non-staggered and staggered finite-difference schemes on uniform and nonuniform rectangular grids were derived in [6–8] for preserving the cosymmetry in the case of rectangular domain. We derive here finite-difference scheme in polar coordinates that preserves the cosymmetry of the problem. The equations (1)–(5) are discretized using uniform grids: $r_i = R_1 + ih_r$, $i = 1\ldots n$, $\varphi_j = \Phi_1 + jh_\varphi$, $j = 1\ldots m$, $h_r = (R_2 - R_1)/(n+1)$, $h_\varphi = (\Phi_2 - \Phi_1)/(m+1)$ The temperature θ and the stream function ψ are denoted at the nodes (r_i, φ_j) as $\theta_{i,j}$ and $\psi_{i,j}$. We introduce then the staggered grids: $r_{i-1/2} = R_1 + (i-1/2)h_r$, $i = 1\ldots n+1$, $\varphi_{j-1/2} = \Phi_1 + (j-1/2)h_\varphi$, $j = 1\ldots m+1$.

The discretized boundary conditions are formulated as follows ($i = 0, n+1$, $j = 0,\ldots,m+1$ or $i = 1,\ldots,n$, $j = 0, m+1$):

$$\theta_{i,j} = 0, \quad \psi_{i,j} = 0.$$

We construct the approximation of the problem applying the two-node difference and averaging operators for integer and half-integer values of i and j

$$(\delta_1\theta)_{i-1/2,j} = \frac{\theta_{i,j} - \theta_{i-1,j}}{h_r}, \qquad (\delta_2\theta)_{i,j-1/2} = \frac{\theta_{i,j} - \theta_{i,j-1}}{h_\varphi} \tag{11}$$

$$(\delta_0^1\theta)_{i-1/2,j} = \frac{\theta_{i,j} + \theta_{i-1,j}}{2}, \qquad (\delta_0^2\theta)_{i,j-1/2} = \frac{\theta_{i,j} + \theta_{i,j-1}}{2}. \tag{12}$$

Then we derive the operators on three-node stencils $D_1 = \delta_0^1\delta_1$, $D_2 = \delta_0^2\delta_2$ and the discrete analog of Laplacian:

$$\varDelta_h\theta_{i,j} = \left[\frac{1}{r}\,\delta_1(r\delta_1\theta) + \frac{1}{r^2}\,\delta_2\delta_2\theta\right]_{i,j}. \tag{13}$$

To approximate the Jacobian J we need in the averaging and difference operators on a four-point stencil: $d_0 = \delta_0^1\delta_0^2$, $d_1 = \delta_0^2\delta_1$, $d_2 = \delta_0^1\delta_2$.

2.1 Approximation of the Buoyancy Term

The approximation of buoyancy term $G_{i,j}$ is specific in the case of polar coordinates. It must satisfy the discrete analog of the condition (8):

$$\widetilde{I}_{G\theta} \equiv \sum_{i=1}^{n}\sum_{j=1}^{m} G_{i,j}\,\theta_{i,j}\,r_i h_r h_\varphi = 0. \tag{14}$$

When the difference operators D_1, D_2 are directly applied to approximation of the buoyancy term (4), the resulting scheme does not conserve cosymmetry. Therefore we constructed the conservative scheme, which satisfies in addition to the preservation of 'buoyancy' condition (8).

The buoyancy term (4) can be rewritten as:

$$G(\theta) = \frac{1}{r}\big(\cos\varphi\,\partial_\varphi\theta + r\sin\varphi\,\partial_r\theta\big). \tag{15}$$

The finite-difference analogue of $G(\theta)$ is constructed as a linear combination of approximations (15) and (4). Thus, the resulting approximation of the buoyancy term can be written as:

$$G(\theta)\,|_{(r_i,\varphi_j)} \approx G_{i,j}(\theta) = \nu G_{i,j}^{(1)} + (1-\nu)G_{i,j}^{(2)}, \tag{16}$$

$$G_{i,j}^{(1)} = \frac{[\cos\varphi\,D_2\theta + r\,\sin\varphi\,D_1\theta]_{i,j}}{r_i}, \quad G_{i,j}^{(2)} = \frac{[D_2(\cos\varphi\,\theta) + D_1(r\,\sin\varphi\,\theta)]_{i,j}}{r_i}.$$

By substituting (16) to $\widetilde{I}_{G\theta}$ (14) and using computer algebra system Maple, the discrete form of (14) is obtained:

$$\widetilde{I}_{G\theta} = (\frac{1}{2} - \nu)\sum_{i=1}^{n}\sum_{j=1}^{m} \theta_{i,j}\,\left[\theta_{i+1,j}(r_{i+1} - r_i)h_\varphi\sin\varphi_j + \right. \tag{17}$$

$$\left. +\;\; \theta_{i,j+1}h_r(\cos\varphi_{j+1} - \cos\varphi_j)\right].$$

Thus, only $\nu = 1/2$ provides the mimetic approximation of buoyancy term (16).

Here Maple was employed schematically in the following manner. We introduce operators

```
> D1:=f -> ( (i,j)->(f[i+1,j]-f[i-1,j])/2/h[r]);
> D2:=f -> ( (i,j)->(f[i,j+1]-f[i,j-1])/2/h[phi]);
```

take a coarse grid and form the boundary conditions

```
> n[r]:=2; n[phi]:=1;
> bc_psi:=seq(seq(psi[i,j]=0,i=0..n[r]+1),j=[0,n[phi]+1]),
>     seq(seq(psi[i,j]=0,j=1..n[phi]),i=[0,n[r]+1]);
> bc_theta:=subs(psi=theta,[bc_psi])[];
> bc:=bc_psi,bc_theta:
```

and evaluate the sum

```
> S:=0:  for i to n[r] do  for j to n[phi] do
> G1:=(cos(phi[j])*D2(psi)(i,j)+r[i]*sin(phi[j])*D1(psi)(i,j))/r[i];
> G2:=((cos(phi[j+1])*psi[i,j+1]-cos(phi[j-1])*psi[i,j-1])/2/h[phi]
>   +sin(phi[j])*(r[i+1]*psi[i+1,j]-r[i-1]*psi[i-1,j])/2/h[r])/r[i];
> S:=S+(nu*G1+(1-nu)*G2) *psi[i,j] *r[i]*h[r]*h[phi];
> od; od;
> factor( subs(bc,S) );
```

$$\frac{1}{2}\sin(\phi_1)\psi_{2,1}\psi_{1,1}h_\phi(r_1 - r_2)(2\nu - 1)$$

After the prompt ν is entered we check it on a finer grid.

2.2 Approximation of the Jacobian

The key point in preservation of cosymmetry in the discretized problem is approximation of the Jacobian [6]. The application of system Maple to derive this was given in [7]. We follow here this method. The approximation of the Jacobian must supply the fulfillment of discrete analogues of (7) and (9):

$$\widetilde{I}_{J\psi} = \sum_{i=1}^{n}\sum_{j=1}^{m} J_{i,j}\,\psi_{i,j}\,r_i h_r h_\varphi = 0, \quad \widetilde{I}_{J\theta} = \sum_{i=1}^{n}\sum_{j=1}^{m} J_{i,j}\,\theta_{i,j}\,r_i h_r h_\varphi = 0. \quad (18)$$

Let write the analogue of Arakawa formula [9] in polar coordinates with the free parameter α:

$$J(\theta,\psi)\mid_{(r_i,\varphi_j)} \approx J_{i,j}(\theta,\psi) = \alpha J^{(1)}_{i,j} + (1-\alpha)J^{(2)}_{i,j}, \quad (19)$$
$$J^{(1)}_{i,j} = \frac{1}{r_i}\left[D_1(\theta D_2\psi) - D_2(\theta D_1\psi)\right]_{i,j}, \quad J^{(2)}_{i,j} = \frac{1}{r_i}\left[d_1(d_0\theta d_2\psi) - d_2(d_0\theta d_1\psi)\right]_{i,j}.$$

The first condition in (18) is satisfied for any values of α. It was checked by direct substitution of (19) to (18) and reorganization of terms using computer algebra system Maple analogically [7].

$$\widetilde{I}_{J\theta} = \frac{3\alpha - 1}{4} \sum_{i=1}^{n} \sum_{j=1}^{m} \psi_{i,j} \Gamma_{i,j},$$

$$\begin{aligned} \Gamma_{i,j} = [&\theta_{i,j-1}(\theta_{i-1,j-1} - \theta_{i+1,j-1}) + \theta_{i+1,j}(\theta_{i+1,j-1} - \theta_{i+1,j+1}) + \\ &+ \theta_{i,j+1}(\theta_{i+1,j+1} - \theta_{i-1,j+1}) + \theta_{i-1,j}(\theta_{i-1,j+1} - \theta_{i-1,j-1})] \,. \end{aligned}$$

Thus we find that only $\alpha = 1/3$ supplies the inheritance the Jacobian properties (7) and (9) through the finite-difference approximation (19). It is simply to check that a formulae (19) preserves the skew-symmetry of the Jacobian and correct treating of discrete symmetries.

2.3 Computation of the Family of Steady States

Using discrete operators (11)–(13), (16), (19) we derive the discretized form of governing equations (1) and (2):

$$\partial_t \theta_{i,j} = \Delta_h \theta_{i,j} + G_{i,j}(\psi) - J_{i,j}(\theta, \psi), \quad 0 = \Delta_h \psi_{i,j} - \lambda\, G_{i,j}(\theta). \tag{20}$$

Let introduce two vectors of nodal values (the temperature and the stream function):

$$\Theta = (\theta_{11}, \theta_{12}, \dots, \theta_{ij}, \theta_{i,j+1}, \dots, \theta_{nm}), \quad \Psi = (\psi_{11}, \psi_{12}, \dots, \psi_{ij}, \psi_{i,j+1}, \dots, \psi_{nm}).$$

Then the resulting system of ordinary differential equations may be writhen as:

$$\dot{\Theta} = A\Theta + B\Psi - F(\Theta, \Psi), \quad 0 = A\Psi - \lambda\, B\Theta, \tag{21}$$

where A is the diagonal matrix corresponding to the approximation of the Laplacian Δ_h and matrix B corresponds to the operator G. The nonlinear vector-function $F(\Theta, \Psi)$ corresponds to some finite-difference approximation of the Jacobian. One can express Ψ from second equation (21) and substitute it to the first one. To study nonsteady convective regimes or to check convergence to a steady pattern we apply the direct approach and integrate the system of ordinary differential equations using the fourth order Runge–Kutta method.

To compute a family of steady states we apply the technique based on the cosymmetric version of the implicit function theorem [3]. First realization of this method was given in [10] for the system of ordinary differential equations derived via Galerkin approach. We applied here the technique which was derived for finite-difference method [6]. When λ is slightly larger than λ_{cr} the state of rest $\theta = \psi = 0$ lost stability and the family of stable steady states is branched off [3]. Starting from the vicinity of unstable zero equilibrium we integrate the ordinary differential equations (21) up to a point θ_0 close to some stable equilibrium on the family. Then we correct the point θ_0 by the Newton method. To predict the next point on the family we determine the kernel of the linearization matrix (Jacobi matrix) at the point θ_0 and use the Adams-Bashforth method. This procedure is repeated to obtain the entire family.

3 Numerical Results

We present here numerical results for convective regimes in the trapezoidal enclosure and semi–ring. In both cases the equilibrium $\theta = \psi = 0$ is stable when $\lambda < \lambda_{cr}$. At the $\lambda > \lambda_{cr}$ the continuous family of steady convective regimes emerges. The stability spectrum of each state contains the zero eigenvalue, which corresponds to neutral direction along family.

Computer experiment in MATLAB consists of computation of the family of the steady states and its continuation under increasing Rayleigh number. Our goal was to find instability on the family and study the number of arcs of instability. To present the results we use Nusselt numbers defined as:

$$Nu_h = \int\limits_{\Phi_1}^{\Phi_2} \frac{\partial \theta}{\partial r} r\Big|_{r=R_1} d\varphi, \qquad Nu_v = \int\limits_{R_1}^{R_2} \frac{1}{r}\frac{\partial \theta}{\partial \varphi}\Big|_{\varphi=\varphi^*} dr, \quad \varphi^* = \frac{\Phi_1 + \Phi_2}{2},$$

where Nu_h corresponds to the integral value of the heat flux defined for the centered vertical section. The value Nu_v may be considered as a cumulative heat flux through the outer cylinder of the annular enclosure.

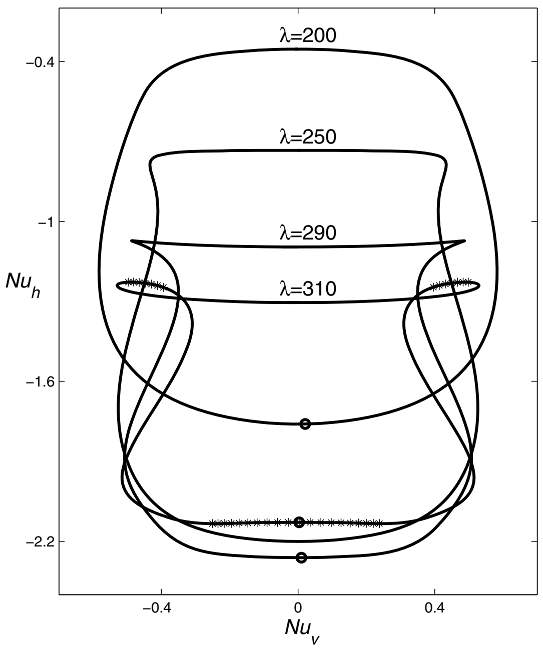


Fig. 1. The transformation of the families of the stationary regimes when Rayleigh number increases λ; trapezoidal enclosure, unsteady regimes are marked by stars

3.1 Cosymmetric Family of Steady States for Trapezoidal Enclosure

Firstly we consider the trapezoidal enclosure $D_1 = [R_1, R_2] \times [\Phi_1, \Phi_2]$, $R_1 = 1$, $R_2 = 2$, $\Phi_1 = 11\pi/12$, $\Phi_2 = 2\pi - \Phi_1$. This domain is close to the rectangle and validation was provided by comparison with results [6]. It should be noted given problem has no boundary layers and this allows to use rather crude grids. Computation on different grids displays that the grid with 16×24 internal nodes provides rather good accuracy for λ_{cr} and we employed mainly this grid for further computation. Such a choice is important because we made a lot of computations to find the family of steady states for each Rayleigh value and continue the family up to instability on it. Each computation includes Newton iterations with calculation of right-hand side and the Jacobi matrix.

So, in the case of the trapezoidal enclosure D_1, the family of steady states is branched off at $\lambda_{cr} \approx 105$. With the growth of the Rayleigh number λ, the radial heat flux increases and the form of the family is changed. Fig. 1 demonstrates the evolution of the family of steady states. At $\lambda > 300$ the instability on the family is detected in three places simultaneously. Zones of instability are marked by stars in Fig. 1.

Generally, the family consists of stationary patterns with two convective cells or one main and two auxiliary convective cells. It demonstrates Fig. 2 where we present streamlines, isotherms and spectra for some convective patterns. Rayleigh number $\lambda = 200$ is rather far from λ_{cr} but all states are stable. It was established by computation of stability spectra and checked by direct

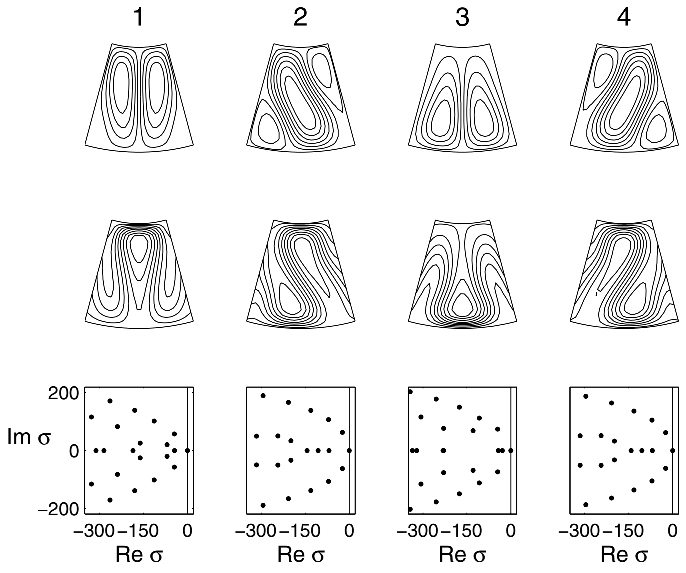


Fig. 2. Streamlines, isotherms and spectrum of steady states from the family; $\lambda = 200$, trapezoidal enclosure

computation of transient behavior under different initial perturbations. The presence of zero eigenvalue (10^{-7}) in the spectra indicates that this convective structure belong to the one–parameter family of steady states. Moreover, the spectra distribution is different along the family. It should be noted that discrete symmetry R^{φ} becomes apparent in that the family include regimes transient each other at the action discrete symmetry (10).

3.2 Cosymmetric Family of Steady States for Semi–ring

We found more variety in convective flows in the case of the domain $D_2 = [1,2] \times [\pi/2, 3\pi/2]$. Patterns with several convective cells through the azimuthal coordinate require more nodes in this direction. The results show that even the coarse grid with 16×48 internal nodes provides sufficient accuracy for computations of convective patterns.

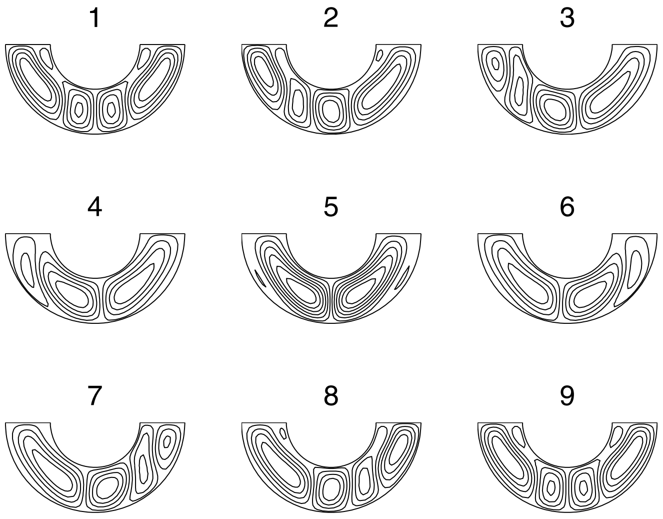


Fig. 3. Streamlines of the stationary regimes from the family; $\lambda = 80$, semi–ring

The family of steady states is branched off at $\lambda_{cr} \approx 42$ and the instability on the family is detected at the Rayleigh number $\lambda > 94$. The character of instability is monotonic, and with increasing of λ the arcs of instability grow. Figure 3 presents streamlines for steady states belonging to the family computed at the Rayleigh number $\lambda = 80$. Depending on the initial data, patterns consisting from 2 to 6 convective cells. Due to discrete symmetry of the problem R^{φ} the family contains the pair of states which may be obtained applying of transformation of discrete symmetry (10). For example, one can compare regimes 1 and 9, 2 and 8, 3 and 7, 4 and 6 in Fig. 3.

Stability analysis of the spectrum shows that the convective states 1 and 3 are unstable. The zero eigenvalue in the spectrum of each state indicates that the convective structures are belonging to the one–parameter family, whereas the occurrence of positive eigenvalues indicates instability of these regimes.

3.3 Preservation of Cosymmetry

Now we present the numerical results about destruction of the family of steady states under non-mimetic approximation. To preserve cosymmetry in discrete analogue of the underlying system it is very important to use correct approximation of the Jacobian and the buoyancy term. We consider impact of parameter α (see formulae (19) for the approximation of the Jacobian) on the size of the family of steady states as well as parameter ν (see formulae (16)) on the destruction of the family.

Firstly, we consider approximations preserving cosymmetry property. Fig. 4 gives the families computed for different values of α with the Rayleigh number $\lambda = 200$ for the case of the trapezoidal enclosure D_1. We note that $\alpha = 0$ and $\alpha = 1$ corresponds to approximation of the Jacobian with $J^{(2)}$ and $J^{(1)}$ respectively and Arakawa approximation corresponds to the value $\alpha = 1/3$.

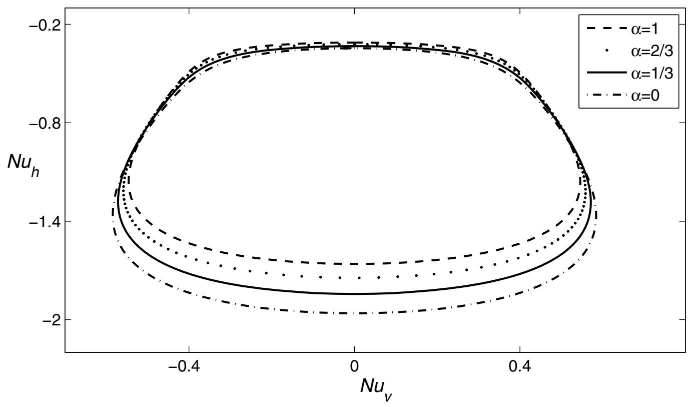


Fig. 4. Family of steady states for different values of α; $\lambda = 200$, trapezoidal enclosure

It was shown in Section 3 that the cosymmetry property (7) is preserved for any value of α, but only the Arakawa approximation provides conservation of important characteristics of the Lapwood convection problem like skew–symmetry and nullification of the gyroscopic force (9). Fig. 4 illustrates that for $\alpha < 1/3$ the size of the family is overestimated and for $\alpha > 1/3$ the size of the family is underestimated. When the Rayleigh number λ decreases, the family collapses to a point with some delay for $\alpha < 1/3$ (and earlier for $\alpha > 1/3$) than for the correct approximation with $\alpha = 1/3$.

Now we present results with non-mimetic approximation of buoyancy term (16). In the case of trapezoidal enclosure and $\lambda = 200$ we found that the family

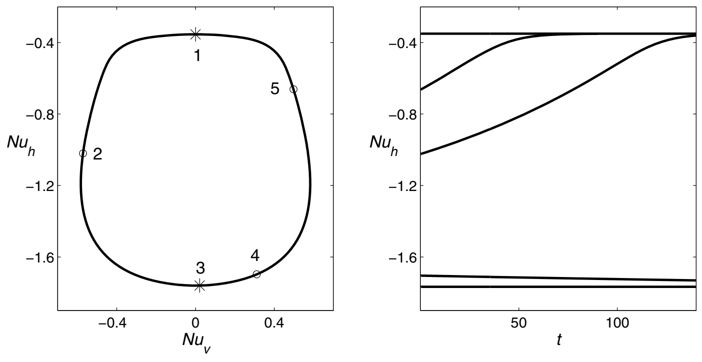


Fig. 5. Family of steady states and convergence from different initial points (circles) to the isolated stationary states (stars); $\lambda = 200$, trapezoidal enclosure

of steady states is destroyed and only two isolated stationary states exist. The evolution from different initial points is presented in Fig. 5 in the case of the approximation of buoyancy term with $\nu = 1$. In Section 2, the expression (17) displays that the discrete analog of the identity (8) doesn't conserve when $\nu \neq 1/2$. But it implies cosymmetry destruction and disappearing of the family of steady states. On the left part of Fig. 5 one can see the results of computer experiment at $\nu = 1$. We took a number of starting points (marked by circles) on the family (closed curve) and carried out the computations up to convergence. It can be seen that only two regimes (points marked by stars) are realized. On the right part of Fig. 5 the transient behavior of the Nusselt number Nu_h is given. It should be noted that convergence to the isolated steady state may take a long time.

4 Summary

The computer algebra systems provide powerful tools for the derivation of numerical schemes with desirable properties [11, 12]. In the case of cosymmetric problem the strong nonuniqueness of solutions is a new challenge [3]. The continuous family of steady state patterns was established for the Lapwood convection. To reproduce this effect we derived mimetic numerical scheme with special approximations of the Jacobian and the buoyancy term. Computer algebra system Maple was used to combine nonlinear terms and the method of free parameters. It was shown that a non-mimetic scheme can destroy the family of steady states.

Results of computation of the families of steady states for two annular domains are given. Non-uniform stability spectra was found for convective patterns belonging to the family of steady states. The continuation of the family up to the appearance of unstable states is done and the scenario with simultaneous instability at three points was detected. So, these families do not result from the symmetry, this is due to the cosymmetrical effect. In this work, we have considered the two-dimensional case by neglecting the variations of the flows on the axial coordinate. The consideration of three dimensional convection in cylindrical domains would be the next problem. It is also of interest to study the coexistence of planar flows and three-dimensional patterns.

Acknowledgements. A.T. and V.T. were supported by Russian Ministry for Education and Research, (2.1.1/6095) and Russian Foundation for Basic Research (11-01-00708).

References

1. Nield, D.A., Bejan, A.: Convection in porous media. Springer, New York (2006)
2. Margolin, L., Shashkov, M.: Finite volume methods and the equations of finite scale: A mimetic approach. Int. J. Numer. Meth. Fluids 56, 991–1002 (2008)
3. Yudovich, V.I.: Cosymmetry, degeneracy of the solutions of operator equations, and the onset of filtrational convection. Math. Notes 49, 540–545 (1991)
4. Lyubimov, D.V.: On the convective flows in the porous medium heated from below. J. Appl. Mech. Techn. Phys. 16, 257–261 (1975)
5. Tsybulin, V.G., Nemtsev, A.D., Karasözen, B.: A mimetic finite-difference scheme for convection of multicomponent fluid in a porous medium. In: Gerdt, V.P., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2009. LNCS, vol. 5743, pp. 322–333. Springer, Heidelberg (2009)
6. Karasözen, B., Tsybulin, V.G.: Finite-difference approximation and cosymmetry conservation in filtration convection problem. Phys. Letters A 262, 321–329 (1999)
7. Karasözen, B., Tsybulin, V.G.: Conservative finite difference schemes for cosymmetric systems. In: Proc. 4th Conf. on Computer Algebra in Scientific Computing, pp. 363–375. Springer, Heidelberg (2001)
8. Karasözen, B., Tsybulin, V.G.: Cosymmetry preserving finite-difference methods for equations of convection in a porous medium. Appl. Num. Math. 55, 69–82 (2005)
9. Arakawa, A.: Computational design for long-term numerical integration of the equations of fluid motion: two-dimensional incompressible flow. J. Comp. Phys. 1, 119–143 (1966)
10. Govorukhin, V.N.: Numerical simulation of the loss of stability for secondary steady regimes in the Darcy plane-convection problem. Doklady Akademii Nauk. 363, 806–808 (1998)
11. Ganzha, V.G., Vorozhtsov, E.V.: Numerical Solutions for Partial Differential Equations. Problem Solving Using Mathematica. CRC Press, Boca Raton (1996)
12. Gerdt, V.P., Blinkov, Y.A.: Involution and Difference Schemes for the Navier–Stokes Equations. In: Gerdt, V.P., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2009. LNCS, vol. 5743, pp. 94–105. Springer, Heidelberg (2009)

Computations in Finite Groups and Quantum Physics

Vladimir V. Kornyak

Laboratory of Information Technologies
Joint Institute for Nuclear Research
141980 Dubna, Russia
kornyak@jinr.ru

Abstract. Mathematical core of quantum mechanics is the theory of unitary representations of symmetries of physical systems. We argue that quantum behavior is a natural result of extraction of "observable" information about systems containing "unobservable" elements in their descriptions. Since our aim is physics where the choice between finite and infinite descriptions can not have any empirical consequences, we consider the problem in the finite background. Besides, there are many indications from observations — from the lepton mixing data, for example — that finite groups underly phenomena in particle physics at the deep level. The "finite" approach allows to reduce any quantum dynamics to the simple permutation dynamics and, thus, to express quantum observables in terms of permutation invariants of symmetry groups and their integer characteristics such as sizes of conjugate classes, sizes of group orbits, class coefficients, and dimensions of representations. Our study has been accompanied by computations with finite groups, their representations and invariants. We have used both our *C* implementation of algorithms for working with groups and computer algebra system *GAP*.

1 Introduction

Symmetry is the leading mathematical principle in quantum mechanics: only systems containing indistinguishable particles demonstrate quantum behavior — any violation of identity of particles destroys quantum interferences.

Mathematical description of any system uses arbitrarily chosen marks for registration and identification elements of the system. Elements of systems with symmetries are decomposed into "homogeneous" sets — group orbits. Only such relations and statements (they are called *invariants*) have objective meaning as are not dependent on relabeling elements lying on the same group orbit. An example of such invariant is the number of elements of a group orbit. To fix an element of a group orbit is possible only with respect to some additional system which appears as "*coordinate system*", or "*observer*", or "*measuring device*". For example, no objective meaning can be attached to electric potentials φ and ψ or to points of space, denoted (marked) as vectors $\mathbf{a}$ and $\mathbf{b}$. But the combinations denoted as $\psi - \varphi$ or $\mathbf{b} - \mathbf{a}$ (in more general group notation $\varphi^{-1}\psi$ and $\mathbf{a}^{-1}\mathbf{b}$) are meaningful. These are examples of typical situations where observable objects or relations are group invariants depending on pairs of elements related to observed system and to observer.

The question of "whether the real world is discrete or continuous" or even "finite or infinite" is entirely *metaphysical*, since neither empirical observations nor logical arguments can validate one of the two adoptions — this is a matter of belief or taste. Since the choice between finite (discrete) and infinite (continuous) descriptions can not have any empirical consequences — "physics is independent of metaphysics" — we can boldly take advantage of "finite" consideration without any risk to destroy the physical content of a problem.

In this paper, we consider *finite quantum mechanics* from constructive, algorithmic point of view. Using the fact that *any representation* of finite group can be embedded into a permutation representation, we show that any quantum dynamics can be reduced to *permutations*, and quantum observables can be expressed in terms of *permutation invariants*. Note that the interpretational issues like "*wavefunction collapse*", "*many-worlds*", "*many-minds*" etc. disappear in the finite background. We discuss also experimental evidences of fundamental role of finite symmetry groups in particle physics.

2 Dynamical Systems and Quantum Evolution

Let us consider **dynamical system** with the finite set of (classical) **states** $\Omega = \{\omega_1, \ldots, \omega_{\mathsf{N}}\}$ in the **discrete time** $t \in \mathcal{T}$, where $\mathcal{T} = \mathbb{Z}$ or $\mathcal{T} = [0, 1, \ldots, T]$. We assume that a finite **symmetry group** $\mathsf{G} = \{\mathsf{g}_1, \ldots, \mathsf{g}_{\mathsf{M}}\} \leq \mathrm{Sym}(\Omega)$ acts on the set of states.

Classical evolution (trajectory) of the dynamical system is a sequence of states evolving in time $\ldots, \omega_{t-1}, \omega_t, \omega_{t+1}, \ldots \in \Omega^{\mathcal{T}}$.

For reasons that will be clear later, we define **quantum evolution** as a sequence of permutations $\ldots p_{t-1}, p_t, p_{t+1} \ldots \in \mathsf{G}^{\mathcal{T}}$, $p_t \in \mathsf{G}$.

In most physical problems, the whole set of states Ω has a special structure of a set of functions $\Omega = \Sigma^{\mathsf{X}}$ on some **space** X with values in some set of **local states** Σ. In dynamical systems with such structure of the set of states nontrivial *gauge structures* — used in physical theories for description of forces — arise naturally. We assume that the space is a finite set $\mathsf{X} = \left\{\mathsf{x}_1, \ldots, \mathsf{x}_{|\mathsf{X}|}\right\}$ possessing nontrivial group of **space symmetries** $\mathsf{F} = \left\{\mathsf{f}_1, \ldots, \mathsf{f}_{|\mathsf{F}|}\right\} \leq \mathrm{Sym}(\mathsf{X})$. The local states form a finite set $\Sigma = \left\{\sigma_1, \ldots, \sigma_{|\Sigma|}\right\}$ provided with the group of **internal symmetries** $\Gamma = \left\{\gamma_1, \ldots, \gamma_{|\Gamma|}\right\} \leq \mathrm{Sym}(\Sigma)$. To combine the space F and internal Γ groups into the symmetry group G of the whole set of states $\Omega = \Sigma^{\mathsf{X}}$ we use the following equivalence class of *split extensions*

$$\mathbf{1} \rightarrow \Gamma^{\mathsf{X}} \rightarrow \mathsf{G} \rightarrow \mathsf{F} \rightarrow \mathbf{1}, \tag{1}$$

where Γ^{X} is the group of Γ-valued functions on the space X. This is a natural generalization of constructions used in physical theories. Explicit formulas for group operations in G expressed in terms of operations in F and $\varGamma$ are given in [1,2] — we do not need them here.

The most popular and intuitive approach to quantization — particularly well suited for dynamical systems with space — is Feynman's path integral: the amplitude of quantum transition from initial to final state is computed by summing up the amplitudes along all possible classical trajectories connecting these states. As is well known, Feynman's approach is equivalent to the traditional matrix formulation of quantum mechanics where the evolution of a system from an initial to a final state is described by an **evolution matrix** U: $|\psi_0\rangle \rightarrow |\psi_T\rangle = U\,|\psi_0\rangle$. The evolution matrix of a quantum dynamical system can be represented as the product of matrices corresponding to elementary time steps: $U = U_{T\leftarrow T-1}\cdots U_{t\leftarrow t-1}\cdots U_{1\leftarrow 0}$. In fact, it can be shown by straightforward examination that Feynman's quantization rules — "multiply subsequent events" and "sum up alternative histories" — is simply a rephrasing of the matrix multiplication rule. For the sake of uniformity of consideration we adopt the evolution matrix approach throughout this paper.

Quantum mechanical evolution matrices are unitary operators acting in Hilbert spaces of (quantum) *state vectors* (called also "*wave functions*", "*amplitudes*" etc.). *Quantum mechanical particles* are associated with unitary representations of certain groups. These representations are called "*singlets*", "*doublets*", and so on, in accordance with their dimensions. Multidimensional representations describe the *spin*. A *quantum mechanical experiment* is reduced to comparison of the system state vector ψ with some sample state vector ϕ provided by a "*measuring apparatus*". According to the Born rule, the probability to observe the coincidence of the states is equal to $\left|\langle\phi\mid\psi\rangle\right|^2/\left(\langle\phi\mid\phi\rangle\,\langle\psi\mid\psi\rangle\right)$.

3 Groups, Numbers and Representations

All transitive actions of a finite group $\mathsf{G} = \{\mathsf{g}_1,\ldots,\mathsf{g}_\mathsf{M}\}$ on finite sets $\varOmega = \{\omega_1,\ldots,\omega_\mathsf{N}\}$ can easily be described [3]. Any such set is in one-to-one correspondence with *right* $H\backslash\mathsf{G}$ (or *left* G/H) *cosets* of some subgroup $H \leq \mathsf{G}$. The set $\varOmega$ is called a *homogeneous space* of the group G (G-*space*). Action of G on $\varOmega$ is *faithful*, if the subgroup H does not contain normal subgroups of G. We can write action in the form of permutations

$$\pi(g) = \binom{\omega_i}{\omega_i g} \sim \binom{Ha}{Hag}, \qquad g, a \in \mathsf{G}, \quad i = 1, \ldots, \mathsf{N}.$$

Maximal transitive set $\varOmega$ is the set of all elements of the group G itself, i.e., the set of cosets of the trivial subgroup $H = \{\mathbf{1}\}$. The corresponding action is called *regular* and can be represented by the permutations

$$\varPi(g) = \binom{\mathsf{g}_i}{\mathsf{g}_i g}, \qquad i = 1, \ldots, \mathsf{M}. \tag{2}$$

To introduce a "quantitative" ("statistical") description, let us assign to the elements of the set $\varOmega$ numerical "weights" from some suitable *number system*

$\mathcal{N}$ containing at least *zero* and *unity*. This allows to rewrite permutations by matrices — this is called *permutation representation*:

$$\pi(g) \rightarrow \rho(g) = \left(\rho(g)_{ij}\right), \quad \text{where } \rho(g)_{ij} = \delta_{\omega_i g, \omega_j}; \quad i, j = 1, \ldots, \mathsf{N}. \tag{3}$$

Here $\delta_{\alpha,\beta}$ is the Kronecker delta on Ω.

The *cycle type* of a permutation is array of multiplicities of lengths of cycles in decomposition of the permutation into disjoint cycles. The cycle type is usually denoted by $1^{k_1}2^{k_2}\cdots n^{k_n}$, where k_i is the number of cycles of the length i in the permutation. The *characteristic polynomial* of permutation matrix (3) can be written immediately from the cycle type of the corresponding permutation $\pi(g)$:

$$\chi_{\rho(g)}(\lambda) = \det\left(\rho(g) - \lambda \mathrm{I}\right) = (\lambda - 1)^{k_1}\left(\lambda^2 - 1\right)^{k_2} \cdots (\lambda^n - 1)^{k_n}. \tag{4}$$

The matrix form of permutations (2) representing the *regular* action

$$\Pi(g) \rightarrow \mathrm{P}(g) = \left(\mathrm{P}(g)_{ij}\right), \quad \mathrm{P}(g)_{ij} = \delta_{e_i g, e_j}, \quad i, j = 1, \ldots, \mathsf{M} \tag{5}$$

is called the *regular representation* — this is a special case of (3).

For the sake of freedom of algebraic manipulations, one assumes usually that $\mathcal{N}$ is an algebraically closed field — a standard choice is the field of complex numbers $\mathbb{C}$. If $\mathcal{N}$ is a field, then the set Ω can be treated as a basis of linear vector space $\mathcal{H} = \mathrm{Span}\left(\omega_1, \cdots, \omega_{\mathsf{N}}\right)$.

The field $\mathbb{C}$ is excessively large — most of its elements are non-constructive. What is really needed can be constructed as follows. As is clear from (4), all *eigenvalues* of permutation matrices are Eth roots of unity, where E is the *exponent* of the group G — the least common multiple of orders of the group elements. The Eth roots of unity can be expressed in terms of $\mathcal{P}$th roots, where $\mathcal{P}$ is some divisor of E called *conductor*. As a first step, we combine the roots of unity with *natural numbers* $\mathbb{N} = \{0, 1, \ldots\}$ to construct the set $\mathcal{N}_{\mathcal{P}} = \mathbb{N}\left[\mathsf{r}\right]$ of polynomials of the form $n_1 + n_2\mathsf{r} + \cdots + n_{\mathcal{P}}\mathsf{r}^{\mathcal{P}-1}$, where $n_k \in \mathbb{N}$; r is *primitive* $\mathcal{P}$th root of unity, i.e. period of r is equal exactly to $\mathcal{P}$. For intuitive perception one could bear in mind the symbolics $\mathsf{r} = \mathrm{e}^{2\pi i/\mathcal{P}}$ for the primitive root, but we will never use this representation. The following *algebraic* definitions are sufficient for all computations

1. *Multiplication*: $\mathsf{r}^k \times \mathsf{r}^m = \mathsf{r}^{k+m \mod \mathcal{P}}$,
2. *Complex conjugation*: $\overline{\mathsf{r}^k} = \mathsf{r}^{\mathcal{P}-k}$.

If $\mathcal{P} = 1$, then $\mathcal{N}_1$ is the *semi-ring of natural numbers* $\mathbb{N}$.

If $\mathcal{P} \geq 2$, then *negative integer numbers* can be introduced via the definition $(-1) = \sum\limits_{k=1}^{p-1} \mathsf{r}^{\frac{\mathcal{P}}{p}k}$, where p is any factor of $\mathcal{P}$. So we obtain the *ring of integers* $\mathbb{Z}$.

If $\mathcal{P} \geq 3$, then the set $\mathcal{N}_{\mathcal{P}}$ is a *commutative ring* embeddable into the field of complex numbers $\mathbb{C}$. This is the ring of *cyclotomic integers*: $\mathcal{N}_{\mathcal{P}} = \mathbb{Z}\left[\mathsf{r}\right] / \left\langle \Phi_{\mathcal{P}}\left(\mathsf{r}\right)\right\rangle$. Here $\Phi_{\mathcal{P}}\left(\mathsf{r}\right)$ is the $\mathcal{P}$th *cyclotomic polynomial* — the product of the binomials $\mathsf{r} - \zeta$, where ζ runs over *all primitive* $\mathcal{P}$th roots of unity.

The ring $\mathcal{N}_\mathcal{P}$ is sufficient for almost all computations with finite quantum models. For simplicity of linear algebra we extend the ring $\mathcal{N}_\mathcal{P}$ to the $\mathcal{P}$th *cyclotomic field* $\mathbb{Q}_\mathcal{P} = \mathbb{Q}\left[\mathsf{r}\right] / \left\langle \Phi_\mathcal{P}\left(\mathsf{r}\right)\right\rangle$. When computing matrices of *unitary* representations *square roots of dimensions* of representations arise as normalization factors. Since square roots of integers are always cyclotomic integers we can treat all irrationalities arising in computations — roots of unity and square roots of dimensions — as belonging to a ring of cyclotomic integers $\mathcal{N}_n$ with some n (usually $n > \mathcal{P}$). We can also construct a minimal *abelian number field* $\mathcal{F}$ containing a given set of irrationalities. It is a subfield of the cyclotomic field $\mathbb{Q}_n$. The term *abelian* means here that $\mathcal{F}$ is an extension with abelian Galois group. The command `Field(gens)` in the computer algebra system *GAP* [4] returns the *smallest* field that contains all elements from the list *gens*. As to the finite quantum systems discussed in this paper, the roots of unity and other irrationalities are only intermediate entities in description of quantum behavior — they disappear in the final "observables".

Any linear representation of a finite group is equivalent to unitary, since one can always construct invariant inner product from an arbitrary one by "averaging over the group". Starting from, e.g., the *standard inner product* in K-dimensional Hilbert space $\mathcal{H}$

$$(\phi \mid \psi) \equiv \sum_{i=1}^{\mathsf{K}} \overline{\phi^i}\psi^i \tag{6}$$

we can come via the averaging to the *invariant inner product*:

$$\langle \phi \mid \psi \rangle \equiv \frac{1}{|G|}\sum_{g\in G}(U(g)\,\phi \mid U\,(g)\,\psi)\,. \tag{7}$$

Here U is a representation of a group G in the space $\mathcal{H}$.

An important transformation of group elements — an analog of change of coordinates in physics — is the conjugation: $a^{-1}ga \rightarrow g'$, $g, g' \in \mathsf{G}$, $a \in \mathrm{Aut}\left(\mathsf{G}\right)$. Conjugation by an element of the group itself, i.e., if $a \in \mathsf{G}$, is called an *inner automorphism.* The equivalence classes with respect to the inner automorphisms are called *conjugacy classes.* The starting point in study of representations of a group is its decomposition into conjugacy classes

$$\mathsf{G} = K_1 \sqcup K_2 \sqcup \cdots \sqcup K_\mathsf{m}.$$

The group multiplication induces *multiplication* on the classes. The product of K_i and K_j is the *multiset* of all possible products ab, $a \in K_i$, $b \in K_j$, decomposed into classes. This multiplication is obviously commutative, since ab and ba belong to the same class: $ab \sim a^{-1}\left(ab\right)a = ba$. Thus, the multiplication table for classes is given by

$$K_iK_j = K_jK_i = \sum_{k=1}^{\mathsf{m}} c_{ijk}K_k. \tag{8}$$

The *natural integers* c_{ijk} — multiplicities of classes in the multisets — are called *class coefficients.*

This is a short list of main properties of linear representations of finite groups:

1. Any irreducible representation is contained in the regular representation. More specifically, there exists matrix T transforming simultaneously all matrices (5) to the form
$$\mathrm{T}^{-1}\mathrm{P}(g)\mathrm{T} = \begin{pmatrix} D_1(g) & & & \\ & d_2\left\{\begin{matrix} D_2(g) & & \\ & \ddots & \\ & & D_2(g)\end{matrix}\right. & & \\ & & \ddots & \\ & & & d_{\mathsf{m}}\left\{\begin{matrix} D_{\mathsf{m}}(g) & & \\ & \ddots & \\ & & D_{\mathsf{m}}(g)\end{matrix}\right. \end{pmatrix}, \tag{9}$$
and any irreducible representation is one of D_j's. The numbers of non-equivalent irreducible representation and conjugacy classes coincide. The number d_j is the dimension of the irreducible component D_j and simultaneously the multiplicity of its occurrence in the regular representation. It is clear from (9) that for the dimensions of irreducible representations the following relation holds: $d_1^2 + d_2^2 + \cdots + d_{\mathsf{m}}^2 = |\mathsf{G}| = \mathsf{M}$. The dimensions of irreducible representations divide the group order: $d_j \mid \mathsf{M}$.
2. Any irreducible representation D_j is determined uniquely by its *character* χ_j defined as the trace of the representation matrix: $\chi_j(g) = \mathrm{Tr} D_j(g)$. This is a function on the conjugacy classes since $\chi_j(g) = \chi_j\left(a^{-1}ga\right)$. Obviously, $\chi_j(\mathbf{1}) = d_j$.
3. A compact form of recording all irreducible representations is the *character table*. The columns of this table are numbered by the conjugacy classes, while its rows contain values of characters of non-equivalent representation:

	K_1	K_2	$\cdots$	K_{m}
χ_1	1	1	$\cdots$	1
χ_2	$\chi_2(K_1) = d_2$	$\chi_2(K_2)$	$\cdots$	$\chi_2(K_{\mathsf{m}})$
$\vdots$	$\vdots$	$\vdots$		$\vdots$
χ_{m}	$\chi_{\mathsf{m}}(K_1) = d_{\mathsf{m}}$	$\chi_{\mathsf{m}}(K_2)$	$\cdots$	$\chi_{\mathsf{m}}(K_{\mathsf{m}})$

.

By convention, the 1st column corresponds to the identity class, and the 1st row contains the *trivial* representation.

4 Finite Quantum Systems

In quantum mechanics all possible states of every physical system are represented by vectors ψ in a Hilbert space $\mathcal{H}$. It is assumed that vectors ψ and ψ' describe identical states if they are proportional through a complex factor:

$\psi' = \lambda\psi,\ \lambda \in \mathbb{C}$. Evolution of the system from any initial state ψ_0 into the corresponding final state ψ_T is described by an *unitary* operator U: $|\psi_T\rangle = U\,|\psi_0\rangle$. The unitarity means that U belongs to the automorphism group of the Hilbert space: $U \in \mathrm{Aut}\,(\mathcal{H})$. One may regard $\mathrm{Aut}\,(\mathcal{H})$ as a faithful representation of respective abstract group G. In the continuous time the dynamics can be expressed by the Schrödinger equation

$$i\frac{\mathrm{d}}{\mathrm{d}t}\,|\psi\rangle = H\,|\psi\rangle$$

in terms of the local *Hermitian* operator H called the *Hamiltonian* or *energy operator*. If H is independent of time, then the relation $U = \mathrm{e}^{-iHT}$ holds.

A finite quantum system is formulated in exactly the same way. The only difference is that now the group G is a finite group of order M having unitary representation U in K-dimensional Hilbert space $\mathcal{H}_{\mathsf{K}}$ over some abelian number field $\mathcal{F}$ instead of $\mathbb{C}$. All possible evolution operators form the finite set $\{U_1, \ldots, U_{\mathsf{M}}\}$ of unitary matrices from U.

Since the matrices U_j are non-singular, one can always introduce Hamiltonians by the formula $H_j = i\ln U_j \equiv \sum\limits_{k=0}^{p-1}\lambda_k U_j^k$, where p is period of U_j, λ_k's are some coefficients[1]; but there is no need to do so.

More generally, *hermitian operators* A describing *observables* in quantum formalism can be written as elements of the group algebra representation:

$$A = \sum_{k=1}^{\mathsf{M}} \alpha_k U_k.$$

Finite groups — unless they are many-component direct products — can be often generated by a small number of elements. For example, all simple and all symmetric groups are generated by two elements. The algorithm restoring the whole group from n_g generators is very simple. It is reduced to $n_g\,(\mathsf{M} - n_g - 1)$ group multiplications. So the finite quantum models are well suited for computer algebra methods.

4.1 Reducing Quantum Dynamics to Permutations

It follows from decomposition [9] that any K-dimensional representation U can be extended to an N-dimensional representation $\widetilde{\mathrm{U}}$ in a Hilbert space $\mathcal{H}_{\mathsf{N}}$, in such a way that the representation $\widetilde{\mathrm{U}}$ corresponds to the *permutation action* of the group G on some N-element set of entities $\Omega = \{\omega_1, \ldots, \omega_{\mathsf{N}}\}$. It is clear that $\mathsf{N} \geq \mathsf{K}$.

[1] Note that the logarithmic function being essentially a construction from continuous mathematics introduces into the λ_k's a *non-algebraic* element — namely, π — expressed by *infinite* sum of elements from $\mathcal{F}$. In other words, the λ_k's are elements of a *transcendental extension* of $\mathcal{F}$.

The case when N is strictly greater than K is most interesting. Clearly, the additional "hidden parameters" — appearing in this case due to increase of the number of states (dimension of space) — in no way can affect the data relating to the space $\mathcal{H}_\mathsf{K}$ since both $\mathcal{H}_\mathsf{K}$ and its complement in $\mathcal{H}_\mathsf{N}$ are invariant subspaces of the extended space $\mathcal{H}_\mathsf{N}$. Thus, *any quantum problem* in K-dimensional Hilbert space can be reformulated in terms of permutations of N things.

From the algorithmic point of view, manipulations with permutations are much more efficient than the linear algebra operations with matrices. Of course, degrees of permutations N might be much larger than dimensions of matrices K. However, the very possibility to *reduce quantum dynamics to permutations* is much more important conceptually than the algorithmic issues.

4.2 Connection with Observation: The Born Rule

In quantum mechanics, the link between mathematical description and experiment is provided by the *Born rule*, stating that the *probability* to observe a quantum system being in the state ψ by apparatus tuned to the state ϕ is expressed by the number

$$\mathbf{P}(\phi,\psi)=\frac{\left|\langle\phi\mid\psi\rangle\right|^2}{\langle\phi\mid\phi\rangle\,\langle\psi\mid\psi\rangle}. \tag{10}$$

This expression can be rewritten in a form including the pair "system–apparatus" in more symmetric way

$$\mathbf{P}(\phi,\psi)=\frac{\left|\langle\phi\mid\psi\rangle\right|^2}{\left|\langle\phi\mid\psi\rangle\right|^2+\left\|\phi\wedge\psi\right\|^2}.$$

Here $\phi\wedge\psi$ is exterior (Grassmann) product of the vectors ϕ and ψ, which is the $\mathsf{K}(\mathsf{K}-1)/2$-dimensional vector with the components in the unitary basis $(\phi\wedge\psi)^{ij}=\phi^i\psi^j-\phi^j\psi^i$ and with the square of norm

$$\left\|\phi\wedge\psi\right\|^2=\sum_{i=1}^{\mathsf{K}-1}\sum_{j=i}^{\mathsf{K}}\left|\phi^i\psi^j-\phi^j\psi^i\right|^2.$$

There are many philosophical speculations concerning the concept of probability and its interpretation. However, what is really used in practice is the *frequency interpretation*: the probability is the ratio of the number of favorable cases to the total number of cases. In the case of finite sets there are no complications at all: the probability is the rational number — the ratio of the number of singled out elements of a set to the total number of elements of the set.

It can be shown that if data about states of a system and apparatus are represented in the permutation basis by *natural numbers*, then formula (10) gives *rational numbers* in the invariant subspaces of the permutation representation also, in spite of possible presence of cyclotomics and square roots in the intermediate computations.

Let us consider permutation action of the group $\mathsf{G} = \{\mathsf{g}_1, \dots, \mathsf{g}_\mathsf{M}\}$ on the set of entities $\Omega = \{\omega_1, \dots, \omega_\mathsf{N}\}$. We will describe the (quantum) states of the system and apparatus in the permutation representation by the vectors

$$|n\rangle = \begin{pmatrix} n_1 \\ \vdots \\ n_\mathsf{N} \end{pmatrix} \text{ and } |m\rangle = \begin{pmatrix} m_1 \\ \vdots \\ m_\mathsf{N} \end{pmatrix}, \tag{11}$$

respectively. It is natural to assume that n_i and m_i are *natural numbers*, interpreting them as the "multiplicities of occurrences" of the element ω_i in the system and apparatus states, respectively. In other words, the vectors $|n\rangle$ and $|m\rangle$ are elements of N-dimensional module H_N over the semi-ring $\mathbb{N}$. Permutation action of G on Ω is equivalent to matrix representation of G in the module H_N. We can turn the module H_N into the Hilbert space $\mathcal{H}_\mathsf{N}$ by extending the semi-ring $\mathbb{N}$ to an abelian number field $\mathcal{F}$ compatible with the structure of G.

Of course, due to the symmetry the numbers n_i and m_i are not observable. Only their *invariant combinations* are observable. Since the standard inner product defined in (6) is invariant for the permutation representation, in accordance with the Born rule we have

$$\mathbf{P}(m, n) = \frac{\left(\sum_i m_i n_i\right)^2}{\sum_i {m_i}^2 \sum_i {n_i}^2}. \tag{12}$$

It is clear that for non-vanishing natural vectors $|n\rangle$ and $|m\rangle$ expression (12) is a rational number strictly greater than zero. This means, in particular, that it is impossible to observe destructive quantum interference here. However, the *destructive interference* of the vectors with natural components can be observed in the proper invariant subspaces of the permutation representation.

5 Example: Group of Permutations of Three Things

S_3 is the smallest non-commutative group providing a non-trivial quantum behavior. Nevertheless, S_3 has important applications in the lepton sector of flavor physics. The group consists of six elements having the following representation by permutations

$$\mathsf{g}_1 = (),\ \mathsf{g}_2 = (2,3),\ \mathsf{g}_3 = (1,3),\ \mathsf{g}_4 = (1,2),\ \mathsf{g}_5 = (1,2,3),\ \mathsf{g}_6 = (1,3,2). \tag{13}$$

The group can be generated by many pairs of its elements. Let us choose, for instance, g_2 and g_6 as generators. S_3 decomposes into three conjugacy classes

$$K_1 = \{\mathsf{g}_1\},\quad K_2 = \{\mathsf{g}_2,\ \mathsf{g}_3,\ \mathsf{g}_4\},\quad K_3 = \{\mathsf{g}_5,\ \mathsf{g}_6\} \tag{14}$$

with the following multiplication table

$$K_1K_j = K_j,\quad K_2^2 = 3K_1 + 3K_3,\quad K_2K_3 = 2K_2,\quad K_3^2 = 2K_1 + K_3.$$

The group S_3 has the following character table

$$\begin{array}{c|ccc} & K_1 & K_2 & K_3 \\ \hline \chi_1 & 1 & 1 & 1 \\ \chi_2 & 1 & -1 & 1 \\ \chi_3 & 2 & 0 & -1 \end{array}\ . \tag{15}$$

Matrices of permutation representation of generators are

$$P_2 = \begin{pmatrix} 1 & \cdot & \cdot \\ \cdot & \cdot & 1 \\ \cdot & 1 & \cdot \end{pmatrix} \text{ and } P_6 = \begin{pmatrix} \cdot & \cdot & 1 \\ 1 & \cdot & \cdot \\ \cdot & 1 & \cdot \end{pmatrix}. \tag{16}$$

The eigenvalues of P_2 and P_6 are $(1, 1, -1)$ and $\left(1, \mathsf{r}, \mathsf{r}^2\right)$, respectively; r is a primitive third root of unity with cyclotomic polynomial $\Phi_3(\mathsf{r}) = 1 + \mathsf{r} + \mathsf{r}^2$.

Since any permutation representation contains one-dimensional invariant subspace with the basis vector $(1, \ldots, 1)^{\mathrm{T}}$, the only possible structure of decomposition of permutation representation into irreducible parts is the following

$$\widetilde{U}_j = \begin{pmatrix} 1 & 0 \\ 0 & U_j \end{pmatrix}, \quad j = 1, \ldots, 6, \tag{17}$$

where the matrices 1 and U_j correspond to one-dimensional trivial (character χ_1) and two-dimensional faithful (character χ_3) representations, respectively.

To construct decomposition (17) we should determine matrices U_j and T such that $\widetilde{U}_j = \mathrm{T}^{-1} P_j \mathrm{T}$. In addition we impose unitarity on all the matrices. Clearly, it suffices to perform the procedure only for matrices of generators. There are different ways to construct decomposition (17).

If we start with the diagonalization of P_6, we come to the following[2]

$$\begin{gathered} U_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},\ U_2 = \begin{pmatrix} 0 & \mathsf{r}^2 \\ \mathsf{r} & 0 \end{pmatrix},\ U_3 = \begin{pmatrix} 0 & \mathsf{r} \\ \mathsf{r}^2 & 0 \end{pmatrix}, \\ U_4 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},\ U_5 = \begin{pmatrix} \mathsf{r}^2 & 0 \\ 0 & \mathsf{r} \end{pmatrix},\ U_6 = \begin{pmatrix} \mathsf{r} & 0 \\ 0 & \mathsf{r}^2 \end{pmatrix}. \end{gathered} \tag{18}$$

[2] Note the peculiarity of representation (18) — its matrices are very similar to matrices of permutations: there is exactly one non-zero entry in each column and in each row. But in contrast to permutation matrices in which any non-zero entry is *unity*, non-zeros in (18) are *roots of unity.* This is because S_3 is one of the so-called *monomial groups* [5] for which all irreducible representations can be constructed as induced from one-dimensional representations of their subgroups — choosing diagonal form for U_6 is just equivalent to inducing (18) from representation of cyclic subgroup $\mathbb{Z}_3 \leq \mathsf{S}_3$. Most groups, at least of small orders, are just monomial. For example, it can be checked with the help of *GAP* that the total number of all non-isomorphic groups of order < 384 is equal to 67424, but only 249 of them are *non-monomial.* The minimal non-monomial group is the 24-element group $\mathsf{SL}(2, 3)$ of 2×2 matrices in the characteristic 3 with unit determinants.

The transformation matrix (up to inessential degrees of freedom for its entries) takes the following form

$$\mathrm{T} = \frac{1}{\sqrt{3}}\begin{pmatrix}1 & 1 & \mathsf{r}^2\\ 1 & \mathsf{r}^2 & 1\\ 1 & \mathsf{r} & \mathsf{r}\end{pmatrix}, \quad \mathrm{T}^{-1} = \frac{1}{\sqrt{3}}\begin{pmatrix}1 & 1 & 1\\ 1 & \mathsf{r} & \mathsf{r}^2\\ \mathsf{r} & 1 & \mathsf{r}^2\end{pmatrix}. \tag{19}$$

Otherwise, the diagonalization of P_2 leads to another second component of decomposition (17) (we present here only the generator matrices)

$$U_2' = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}, \quad U_6' = \begin{pmatrix}-\frac{1}{2} & \frac{\sqrt{3}}{2}\\ -\frac{\sqrt{3}}{2} & -\frac{1}{2}\end{pmatrix}.$$

The transformation matrix in this case takes the form

$$\mathrm{T}' = \begin{pmatrix}\frac{1}{\sqrt{3}} & \sqrt{\frac{2}{3}} & 0\\ \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{2}}\end{pmatrix}, \quad \mathrm{T}'^{-1} = \begin{pmatrix}\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}}\\ \sqrt{\frac{2}{3}} & -\frac{1}{\sqrt{6}} & -\frac{1}{\sqrt{6}}\\ 0 & -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\end{pmatrix}. \tag{20}$$

The matrix T' is known in particle physics under the names *Harrison-Perkins-Scott* or *tribimaximal* mixing matrix. It is used to description of neutrino oscillation data.

The information about "quantum behavior" is encoded, in fact, in transformation matrices like (19) or (20).

Let $|n\rangle = \begin{pmatrix}n_1\\ n_2\\ n_3\end{pmatrix}$ and $|m\rangle = \begin{pmatrix}m_1\\ m_2\\ m_3\end{pmatrix}$ be system and apparatus state vectors in the "permutation" basis. Transformation of these vectors from the permutation to "quantum" basis with the help of, say, (19) leades to

$$\left|\widetilde{\psi}\right\rangle = \mathrm{T}^{-1}\left|n\right\rangle = \frac{1}{\sqrt{3}}\begin{pmatrix}n_1 + n_2 + n_3\\ n_1 + n_2\mathsf{r} + n_3\mathsf{r}^2\\ n_1\mathsf{r} + n_2 + n_3\mathsf{r}^2\end{pmatrix},$$

$$\left|\widetilde{\phi}\right\rangle = \mathrm{T}^{-1}\left|m\right\rangle = \frac{1}{\sqrt{3}}\begin{pmatrix}m_1 + m_2 + m_3\\ m_1 + m_2\mathsf{r} + m_3\mathsf{r}^2\\ m_1\mathsf{r} + m_2 + m_3\mathsf{r}^2\end{pmatrix}.$$

Projections of the vectors onto two-dimensional invariant subspace are:

$$\left|\psi\right\rangle = \frac{1}{\sqrt{3}}\begin{pmatrix}n_1 + n_2\mathsf{r} + n_3\mathsf{r}^2\\ n_1\mathsf{r} + n_2 + n_3\mathsf{r}^2\end{pmatrix}, \quad \left|\phi\right\rangle = \frac{1}{\sqrt{3}}\begin{pmatrix}m_1 + m_2\mathsf{r} + m_3\mathsf{r}^2\\ m_1\mathsf{r} + m_2 + m_3\mathsf{r}^2\end{pmatrix}. \tag{21}$$

The same manipulation with matrix (20) leads to

$$\left|\psi'\right\rangle = \begin{pmatrix}n_1\sqrt{\frac{2}{3}} - n_2\frac{1}{\sqrt{6}} - n_3\frac{1}{\sqrt{6}}\\ -n_2\frac{1}{\sqrt{2}} + n_3\frac{1}{\sqrt{2}}\end{pmatrix}, \quad \left|\phi'\right\rangle = \begin{pmatrix}m_1\sqrt{\frac{2}{3}} - m_2\frac{1}{\sqrt{6}} - m_3\frac{1}{\sqrt{6}}\\ -m_2\frac{1}{\sqrt{2}} + m_3\frac{1}{\sqrt{2}}\end{pmatrix}. \tag{22}$$

Constituents of Born's probability (10) for the two-dimensional subsystem — clearly, the same in both cases (21) and (22) — are

$$\langle\psi\,|\,\psi\rangle = \mathrm{Q}_3\left(n,n\right) - \frac{1}{3}\mathrm{L}_3\left(n\right)^2, \tag{23}$$

$$\langle\phi\,|\,\phi\rangle = \mathrm{Q}_3\left(m,m\right) - \frac{1}{3}\mathrm{L}_3\left(m\right)^2, \tag{24}$$

$$\left|\langle\phi\,|\,\psi\rangle\right|^2 = \left(\mathrm{Q}_3\left(m,n\right) - \frac{1}{3}\mathrm{L}_3\left(m\right)\mathrm{L}_3\left(n\right)\right)^2, \tag{25}$$

where $\mathrm{L}_\mathsf{N}\left(n\right) = \sum_{i=1}^{\mathsf{N}} n_i$ and $\mathrm{Q}_\mathsf{N}\left(m,n\right) = \sum_{i=1}^{\mathsf{N}} m_i n_i$ are linear and quadratic permutation invariants, respectively.

Note that:

1. Expressions (23)–(25) consist of the *invariants of permutation representation*. This is a manifestation of fundamental role of permutations in quantum description.
2. Expressions (23) and (24) are always positive rational numbers for $|n\rangle$ and $|m\rangle$ with different components.
3. Conditions for *destructive quantum interference* — vanishing Born's probability — are determined by the equation
$$3\left(m_1n_1 + m_2n_2 + m_3n_3\right) - \left(m_1 + m_2 + m_3\right)\left(n_1 + n_2 + n_3\right) = 0.$$
This equation has infinitely many solutions in natural numbers. An example of such a solution is: $|n\rangle = \begin{pmatrix}1\\1\\2\end{pmatrix}, \quad |m\rangle = \begin{pmatrix}1\\3\\2\end{pmatrix}$.

Thus, we have obtained essential features of quantum behavior from "permutation dynamics" and "natural" interpretation (11) of quantum amplitude by a simple transition to invariant subspaces.

Recall once more that any permutation representation contains the trivial one-dimensional subrepresentation and, hence, has $(\mathsf{N}-1)$-dimensional invariant subspace. The inner product in this subspace can be expressed in terms of the permutation invariants by the formula

$$\langle\phi\,|\,\psi\rangle = \mathrm{Q}_\mathsf{N}\left(m,n\right) - \frac{1}{\mathsf{N}}\mathrm{L}_\mathsf{N}\left(m\right)\mathrm{L}_\mathsf{N}\left(n\right).$$

The identity $\mathrm{Q}_\mathsf{N}\left(n,n\right) - \frac{1}{\mathsf{N}}\mathrm{L}_\mathsf{N}\left(n\right)^2 \equiv \frac{1}{\mathsf{N}^2}\sum_{i=1}^{\mathsf{N}}\left(\mathrm{L}_\mathsf{N}\left(n\right) - \mathsf{N}n_i\right)^2$ shows explicitly that $\langle\psi\,|\,\psi\rangle > 0$ for $|n\rangle$ with different components n_i. This inner product does not contain irrationalities for natural $|n\rangle$ and $|m\rangle$. This is not the case for other invariant subspaces. Nevertheless irrationalities disappear in the squared modulus of the inner product $|\langle\phi\,|\,\psi\rangle|^2$. To give a simple illustration let us consider the cyclic group $\mathbb{Z}_3$. Its three-dimensional permutation representation decomposes into three one-dimensional irreducible components. E.g., for the generator $g = (1,2,3)$ of $\mathbb{Z}_3$ we have

$$P = \begin{pmatrix} \cdot & 1 & \cdot \\ \cdot & \cdot & 1 \\ 1 & \cdot & \cdot \end{pmatrix} \longrightarrow \widetilde{U} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \mathsf{r} & 0 \\ 0 & 0 & \mathsf{r}^2 \end{pmatrix}, \quad \mathsf{r} \text{ is a primitive third root of unity.}$$

The inner product in one-dimensional subspace corresponding to the eigenvalue, say r, contains irrationalities: $\langle\phi\,|\,\psi\rangle = \frac{1}{3}\left(\mathrm{Q}_3\,(m,n) + \mathsf{r}C(m,n) + \mathsf{r}^2C'(m,n)\right)$, but $|\langle\phi\,|\,\psi\rangle|^2 = \frac{1}{9}\left(\mathrm{Q}_3\,(m,m) - C(m,m)\right)\left(\mathrm{Q}_3\,(n,n) - C(n,n)\right)$ is free of them. The invariants $C(m,n) = m_1n_3 + m_2n_1 + m_3n_2$ and $C'(m,n) = m_1n_2 + m_2n_3 + m_3n_1$ are specific for the group $\mathbb{Z}_3$ in contrast to $\mathrm{L}_{\mathsf{N}}\,(n)$ and $\mathrm{Q}_{\mathsf{N}}\,(m,n)$ that are common to all permutation groups.

6 Finite Symmetry Groups in Particle Physics

At present, all observations concerning fundamental particles [6] are compatible with the Standard Model (SM). The SM is a gauge theory with the group of internal (gauge) symmetries $\Gamma = \mathsf{SU}\,(3) \times \mathsf{SU}\,(2) \times \mathsf{U}\,(1)$. In the context of Grand Unified Theory (GUT) Γ is assumed to be a subgroup of some larger (simple) group. With respect to space-time symmetries, the elementary particles are divided into two classes: *bosons*, responsible for physical forces (roughly speaking, they are elements of the gauge group) and *fermions*, usually treated as particles of matter. The fermions of the SM are divided into three *generations* of *quarks* and *leptons* as follows (antiparticles are omitted for brevity):

	Generations					
	1		2		3	
Up-quarks	Up	u	Charm	c	Top	t
Down-quarks	Down	d	Strange	s	Bottom	b
Charged leptons	Electron	e^-	Muon	μ^-	Tau	τ^-
Neutrinos	Electron neutrino	ν_e	Muon neutrino	ν_μ	Tau neutrino	ν_τ

Between generations particles differ only by their mass and quantum property called *flavor*. The flavor changing transitions — taking place in such phenomena as weak decays of quarks and neutrino oscillations — are described by 3×3 unitary *mixing matrices*. The outputs of experiments allow to calculate magnitudes of elements of these matrices.

In the case of quarks ("*in the quark sector*"), the mixing matrix describing transitions between up- and down-type quarks is the *Cabibbo–Kobayashi–Maskawa* (CKM) matrix

$$V_{\mathrm{CKM}} = \begin{pmatrix} V_{ud} & V_{us} & V_{ub} \\ V_{cd} & V_{cs} & V_{cb} \\ V_{td} & V_{ts} & V_{tb} \end{pmatrix},$$

where $\left|V_{\alpha\beta}\right|^2$ represents the probability that the quark (of flavor) β decays into a quark α. The current experimental data rounded to three significant digits are:

$$\begin{pmatrix} |V_{ud}| & |V_{us}| & |V_{ub}| \\ |V_{cd}| & |V_{cs}| & |V_{cb}| \\ |V_{td}| & |V_{ts}| & |V_{tb}| \end{pmatrix} = \begin{pmatrix} 0.974 & 0.225 & 0.004 \\ 0.225 & 0.974 & 0.041 \\ 0.009 & 0.040 & 0.999 \end{pmatrix}.$$

More precise values can be found in [6].

In the lepton sector weak interaction processes are described by the *Pontecorvo–Maki–Nakagawa–Sakata* (PMNS) mixing matrix

$$U_{\mathrm{PMNS}} = \begin{pmatrix} U_{e1} & U_{e2} & U_{e3} \\ U_{\mu 1} & U_{\mu 2} & U_{\mu 3} \\ U_{\tau 1} & U_{\tau 2} & U_{\tau 3} \end{pmatrix}.$$

Here indices e, μ, τ correspond to neutrino flavors — this means that the neutrinos ν_e, ν_μ, ν_τ are produced with e^+, μ^+, τ^+ (or produce e^-, μ^-, τ^-), respectively, in weak processes. The indices $1, 2, 3$ correspond to the *mass eigenstates*, i.e., neutrinos ν_1, ν_2, ν_3 with definite masses m_1, m_2, m_3. Numerous experiments with solar, atmospheric, reactor, and accelerator neutrinos indicate the existence of discrete symmetries that can not be deduced from the SM. The phenomenological pattern is the following [7]:

1. ν_μ and ν_τ flavors are presented with equal weights in all three mass eigenstates ν_1, ν_2, ν_3 (this is called "*bi-maximal mixing*"): $\left|U_{\mu i}\right|^2 = \left|U_{\tau i}\right|^2, \quad i = 1, 2, 3;$
2. all three flavors are presented equally in ν_2 ("*trimaximal mixing*"): $\left|U_{e2}\right|^2 = \left|U_{\mu 2}\right|^2 = \left|U_{\tau 2}\right|^2;$
3. ν_e is absent in ν_3: $\left|U_{\mu 3}\right|^2 = 0.$

These relations together with the normalization condition for probabilities allow to determine moduli-squared of all matrix elements:

$$\left(|U_{li}|^2\right) = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} & 0 \\ \frac{1}{6} & \frac{1}{3} & \frac{1}{2} \\ \frac{1}{6} & \frac{1}{3} & \frac{1}{2} \end{pmatrix}. \tag{26}$$

A particular form of unitary matrix satisfying data (26) was suggested by Harrison, Perkins, and Scott in [8]:

$$U_{\mathrm{TB}} = \begin{pmatrix} \sqrt{\frac{2}{3}} & \frac{1}{\sqrt{3}} & 0 \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{6}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} \end{pmatrix}. \tag{27}$$

This so-called *tribimaximal* (TB) mixing matrix coincides — up to the trivial permutation of two columns corresponding to the renaming $\nu_1 \rightleftarrows \nu_2$ of states — with transformation matrix ([20]) decomposing the natural permutation representation of the group S_3 into irreducible components. This means that we can identify the flavor basis with the representation basis of permutations of three things, and the mass basis is a basis of irreducible decomposition of this representation. In [9] Harrison and Scott study in detail connections of the neutrino mass matrix with the character table and class algebra of the group S_3. At present, much effort is devoted to the construction and study of models based on finite flavor symmetries (for recent reviews, see, for example, [10,11]). The most popular groups for constructing such models are:

– $\mathsf{T} = \mathsf{A}_4$ — the tetrahedral group;
– T' — the double covering of A_4;
– $\mathsf{O} = \mathsf{S}_4$ — the octahedral group;
– $\mathsf{I} = \mathsf{A}_5$ — the icosahedral group;
– D_N — the dihedral groups (N even);
– Q_N — the quaternionic groups (4 divides N);
– $\Sigma\left(2N^2\right)$ — the groups in this series have the structure $(\mathbb{Z}_N \times \mathbb{Z}_N) \rtimes \mathbb{Z}_2$;
– $\Delta\left(3N^2\right)$ — the structure $(\mathbb{Z}_N \times \mathbb{Z}_N) \rtimes \mathbb{Z}_3$;
– $\Sigma\left(3N^3\right)$ — the structure $(\mathbb{Z}_N \times \mathbb{Z}_N \times \mathbb{Z}_N) \rtimes \mathbb{Z}_3$;
– $\Delta\left(6N^2\right)$ — the structure $(\mathbb{Z}_N \times \mathbb{Z}_N) \rtimes \mathsf{S}_3$.

As to the quark sector, observations do not give such sharp picture as in the lepton case. In [12] the D_{14} symmetry was suggested for explanation of the value of the Cabibbo angle (one of the parameters of the CKM matrix), but without any connection with the leptonic symmetries. The natural attempts to find discrete symmetries unifying leptons and quarks still remain not very successful, though there are some encouraging observations, for example, the *quark-lepton complementarity* (QLC) — observation that the sum of quark and lepton mixing angles is equal approximately to $\pi/4$.

The origin of finite symmetries among fundamental particles is unclear. There are different attempts to explain — sometimes looking a bit complicated and artificial, for example, these symmetries are treated as symmetries of manifolds arising at compactification of a higher dimensional theory to four spacetime dimensions [13]. The idea that symmeties at the most fundamental level are *per se* finite looks more attractive in our opinion. In this approach, unitary groups used in physical theories can be treated simply as repositories of all finite groups having faithful representation of corresponding dimensions: $\mathsf{U}\left(n\right)$ contains all finite groups with faithful n-dimensional representations. Of course, due to redundancy of the field $\mathbb{C}$, $\mathsf{U}\left(n\right)$ is not a minimal group with this property.

Such small groups as S_3, A_4, etc. are most likely only remnants of large combinations of more fundamental finite symmetries that are expected to exist at the GUT scale. Unfortunately the GUT scale (10^{16} GeV) being close to the Planck scale (10^{19} GeV) is out of reach of experiments (the most powerful colliders to date can provide only about 10^4 GeV). Thus, the only practical way is to construct models, study them by the computational group theory methods, and compare consequences of these models with available experimental data.

7 Conclusion

"Finite" analysis shows that quantum behavior is a manifestation of indistinguishability of objects, i.e., fundamental impossibility to trace the identity of homogeneous objects in the process of their evolution.

Only "statistical" statements about numbers of certain invariant combinations of elements may have objective significance. These statements can be expressed in terms of group invariants and natural numbers characterizing symmetry groups, such as dimensions of its representations, class coefficients etc.

Any quantum mechanical problem can be reduced to permutations since permutation representations contain all other representations. This — together with natural interpretation of quantum amplitudes as vectors of "multiplicities of occurences" of underlying permuted entities — makes quantum mechanical problems constructive and particularly suitable for their study by computer algebra and computational group theory methods.

The models based on finite groups are now extensively studied in particle physics, since there are strong observational evidences of finite symmetries in fundamental physical processes.

Acknowledgment. The work was supported by the grants 01-01-00200 from the Russian Foundation for Basic Research and 3810.2010.2 from the Ministry of Education and Science of the Russian Federation.

References

1. Kornyak, V.V.: Quantization in discrete dynamical systems. J. Math. Sci. 168(3), 390–397 (2010)
2. Kornyak, V.V.: Structural and symmetry analysis of discrete dynamical systems. In: Cellular Automata, pp. 1–45. Nova Science Publishers Inc., New York (2010), http://arxiv.org/abs/1006.1754
3. Hall Jr., M.: The Theory of Groups. Macmillan, New York (1959)
4. http://www.gap-system.org/
5. Kirillov, A.A.: Elements of the Theory of Representations. Springer, Heidelberg (1976)
6. Nakamura, K., et al.: (Particle Data Group): The review of particle physics. J. Phys. G 37, 075021, 1–1422 (2010)
7. Smirnov, A.Y.: Discrete Symmetries and Models of Flavor Mixing, p. 14 (2011); arXiv:1103.3461
8. Harrison, P.F., Perkins, D.H., Scott, W.G.: Tri-bimaximal mixing and the neutrino oscillation data. Phys. Lett. B 530, 167 (2002); arXiv: hep-ph/0202074

9. Harrison, P.F., Scott, W.G.: Permutation symmetry, Tri-bimaximal neutrino mixing and the S3 group characters. Phys. Lett. B 557, 76 (2003); arXiv: hep-ph/0302025
10. Ishimori, H., Kobayashi, T., Ohki, H., Okada, H., Shimizu, Y., Tanimoto, M.: Non-abelian discrete symmetries in particle physics. Prog. Theor. Phys. Suppl. 183, 1–173 (2010); arXiv:1003.3552
11. Ludl, P.O.: Systematic Analysis of Finite Family Symmetry Groups and Their Application to the Lepton Sector, arXiv:0907.5587
12. Blum, A., Hagedorn, C.: The Cabibbo Angle in a Supersymmetric D14 Model. Nucl. Phys. B 821, 327–353 (2009)
13. Altarelli, G., Feruglio, F.: Discrete flavor symmetries and models of neutrino mixing. Rev. Mod. Phys. 82(3), 2701–2729 (2010)

Regular and Singular Boundary Problems in Maple

Anja Korporal[1,⋆], Georg Regensburger[2,⋆⋆], and Markus Rosenkranz[3]

[1] Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, Altenberger Str. 69, 4040 Linz, Austria

[2] INRIA Saclay – Île de France, Project DISCO, L2S, Supélec, 91192 Gif-sur-Yvette Cedex, France

[3] University of Kent, Cornwallis Building, Canterbury, Kent CT2 7NF, United Kingdom

Abstract. We describe a new Maple package for treating boundary problems for linear ordinary differential equations, allowing two-/multi-point as well as Stieltjes boundary conditions. For expressing differential operators, boundary conditions, and Green's operators, we employ the algebra of integro-differential operators. The operations implemented for regular boundary problems include computing Green's operators as well as composing and factoring boundary problems. Our symbolic approach to singular boundary problems is new; it provides algorithms for computing compatibility conditions and generalized Green's operators.

Keywords: Linear boundary problem, Singular Boundary Problem, Generalized Green's operator, Green's function, Integro-Differential Operator, Ordinary Differential Equation.

1 Introduction

Although boundary problems clearly play an important role in applications and in Scientific Computing, there is no systematic support for solving them symbolically in current computer algebra systems. In this paper, we describe a Maple package with algorithms for regular as well as singular boundary problems for linear ordinary differential equations (LODEs). While a first version of the package with functions for regular boundary problems was presented in [1], the methods and the implementation for singular problems are new. A prototype implementation for regular boundary problems in the TH∃OREM∀ system was described in [2] as part of a general symbolic framework for boundary problems, including also some first steps towards linear partial differential equations (LPDEs).

⋆ Partially supported by the RISC PhD scholarship program of the government of Upper Austria

⋆⋆ Supported by the Austrian Science Fund (FWF): J 3030-N18.

In Section 2, we recall the algebra of integro-differential operators providing the algebraic structure for computing with boundary problems. We describe its implementation in Maple, where we use a normal form approach in contrast to [2]. In Section 3, we outline our symbolic approach for solving boundary problems. For an analytic treatment of boundary problems for LODEs, see for example [3,4] or [5] for further applications. The functions we present include the computation of Green's operators and Green's functions as well as the factorization of boundary problems.

We introduce generalized boundary problems in Section 4 and develop an algorithm for computing generalized Green's operators. The main step of the algorithm is to determine compatibility conditions for arbitrary boundary problems in an algebraic setting; the special case of two-point boundary problems of second order is discussed in [6, Lecture 34]. For singular boundary problems and generalized or modified Green's functions in Analysis, we refer for example to [4] and [7], and in the context of generalized inverses to [8, Sect. 9.4], [9], and [10, Sect. H].

The Maple package *IntDiffOp* is available with an example worksheet at `http://www.risc.jku.at/people/akorpora/index.html`.

2 Integro-Differential Operators

We first recall the definition of integro-differential algebras and operators, see [11] and [12] for further details. For the similar notion of differential Rota-Baxter algebras, we refer to [13]. As a motivating example, consider the algebra $\mathcal{F} = C^\infty(\mathbb{R})$ with the usual derivation and the integral operator $\int\colon f \mapsto \int_a^x f(\xi)\,d\xi$ for a fixed $a \in \mathbb{R}$. The essential algebraic identities satisfied by the derivation and the integral operator are the Leibniz rule, the Fundamental Theorem of Calculus, and Integration by Parts. Note also that $f(a) = f - \int f'$, so the evaluation $\mathtt{E}_a\colon f \mapsto f(a)$ at the initialization point a of the integral can also be expressed in terms of the derivation and integral.

We call $(\mathcal{F}, \partial, \int)$ an *integro-differential algebra* if $(\mathcal{F}, \partial)$ is a commutative differential algebra over a commutative ring K and $\int$ is a K-linear right inverse (section) of $\partial = {}'$, meaning $(\int f)' = f$, such that the *differential Baxter axiom*

$$(\textstyle\int f')(\int g') + \int (fg)' = (\int f')g + f(\int g')$$

holds. We call $\mathtt{E} = 1 - \int \circ \partial$ the *evaluation* of $\mathcal{F}$. We say that an integro-differential algebra over a field K is ordinary if $\operatorname{Ker}(\partial) = K$. For an ordinary integro-differential algebra, the evaluation can be interpreted as a multiplicative linear functional (character) $\mathtt{E}\colon \mathcal{F} \to K$. This allows treating initial value problems, but for doing boundary problems we need additional characters $\varphi\colon \mathcal{F} \to K$ (in the above example, evaluations $\mathtt{E}_c\colon f \mapsto f(c)$ at various points $c \in \mathbb{R}$).

Let $(\mathcal{F}, \partial, \int)$ be an ordinary integro-differential algebra over a field K and let $\Phi \subseteq \mathcal{F}^*$ be a set of multiplicative linear functionals $\varphi\colon \mathcal{F} \to K$ including $\mathtt{E}$. The *integro-differential operators* $\mathcal{F}_\Phi[\partial, \int]$ are defined in [11] as the K-algebra

Table 1. Rewrite Rules for Integro-Differential Operators

$fg \to f \cdot g$	$\partial f \to f\partial + f'$	$\int f \int \to (\int f) \int - \int (\int f)$
$\varphi\psi \to \psi$	$\partial\varphi \to 0$	$\int f \partial \to f - \int f' - \mathrm{E}(f)\,\mathrm{E}$
$\varphi f \to \varphi(f)\,\varphi$	$\partial \int \to 1$	$\int f \varphi \to (\int f)\,\varphi$

generated by the symbols ∂ and $\int$, the "functions" $f \in \mathcal{F}$ and the "functionals" $\varphi \in \Phi$, modulo the Noetherian and confluent rewrite system of Table 1.

The representation of integro-differential operators in our Maple implementation is based on the fact that every integro-differential operator has a unique normal form as a sum of a differential, integral, and boundary operator. The normal forms of differential operators are as usual $\sum f_i \partial^i$, integral operators can be written uniquely (up to bilinearity) as sums of terms of the form $f \int g$, and the normal forms of *boundary operators* are given by

$$\sum_{\varphi \in \Phi} \Big(\sum_{i \in \mathbb{N}} f_{i,\varphi} \varphi \partial^i + \sum_{j \in \mathbb{N}} g_{j,\varphi} \varphi \textstyle\int h_{j,\varphi} \Big), \tag{1}$$

with only finitely nonzero summands. Stieltjes *boundary conditions* are boundary operators where $f_{i,\varphi} = a_{\varphi,i} \in K$ and $g_{j,\varphi} = 1$. They act on $\mathcal{F}$ as linear functionals in the dual space $\mathcal{F}^*$. See [14] for Stieltjes boundary conditions in Analysis.

From Table 1 formulas can be derived for expressing the product of integro-differential operators directly in terms of normal forms; see [15] for the case $\Phi = \{\mathrm{E}\}$. Implementing these formulas leads to faster computations since we need not reduce in each step. In our package, we use for the underlying "integro-differential algebra" all the smooth functions in one variable representable in Maple, together with the usual derivation and the integral operator $\int = \int_0^x$, both computed by Maple . We take as characters $\Phi = \{\mathrm{E}_c \mid c \in \mathbb{R}\}$.

We created data types for the different kinds of operator, representing integro-differential operators as triples $\texttt{INTDIFFOP}(a, b, c)$, where a is a differential operator, b an integral operator and c a boundary operator. Differential operators are represented as lists $\texttt{DIFFOP}(f_0, f_1, \ldots)$ and integral operators as lists of pairs of the form $\texttt{INTOP}(\texttt{INTTERM}(f_1, g_1), \texttt{INTTERM}(f_2, g_2), \ldots)$. In order to have a unique representation for integral operators, one would need a basis of the underlying integro-differential algebra and use only basis elements for the g_i. In our implementation, we use the following heuristic approach: We split sums in the g_i and move scalar factors to the coefficients f_i.

Due to (1), a boundary operator $\texttt{BOUNDOP}$ contains a list of evaluations at different points. Each evaluation $\texttt{EVOP}$ is a triple containing the evaluation point, the local part $\sum f_{i,\varphi}\varphi\partial^i$ and the global part $\sum g_{j,\varphi}\varphi \int h_{j,\varphi}$. Hence we use the expression $\texttt{BOUNDOP}(\texttt{EVOP}(c, \texttt{EVDIFFOP}(f_0, \ldots), \texttt{EVINTOP}(\texttt{EVINTTERM}(g_1, h_1), \ldots), \ldots)$ for the representation of boundary operators.

In the following example, we first enter some operators of different types. For displaying the operators, we use D for ∂, A for $\int$ and E[c] for the evaluation E_c.

```
> T := DIFFOP(0,0,1);
```

$$T \ := \ \mathrm{D}^2$$

```
> G := INTOP(INTTERM(1,1));
```

$$G \ := \ \mathrm{A}$$

```
> B := BOUNDOP(EVOP(1, EVDIFFOP(1), EVINTOP(EVINTTERM(1,1))));
```

$$B \ := \ \mathrm{E}[1] \ + \ ((\mathrm{E}[1]) \ . \ A)$$

Now we show how to add and multiply integro-differential operators and how to apply them to a function $f \in \mathcal{F}$.

```
> ApplyOperator(G, f(x));
```

$$\int_0^x f(x)dx$$

```
> MultiplyOperator(G,G);
```

$$(x \ . \ A) \ - \ (A \ . \ x)$$

```
> MultiplyOperator(T,G,G);
```

$$1$$

```
> S := AddOperator(T, G, B);
```

$$S \ := \ \mathrm{D}^2 \ + \ A \ + \ \mathrm{E}[1] + \ ((\mathrm{E}[1]) \ . \ A)$$

```
> ApplyOperator(S, f(x));
```

$$\frac{\mathrm{d}^2}{\mathrm{dx}^2} f(x) + \int_0^x f(x)dx + f(1) + \int_0^1 f(x)dx$$

3 Regular Boundary Problems in Maple

In this section, we demonstrate how to compute with regular boundary problems in our Maple package. For an integro-differential algebra $\mathcal{F}$, a boundary problem is given by a monic differential operator $T = \partial^n + c_{n-1}\partial^{n-1} + \cdots + c_1\partial + c_0$ and boundary conditions $\beta_1, \ldots, \beta_m$. Given a forcing function $f \in \mathcal{F}$, we want to find $u \in \mathcal{F}$ such that

$$\boxed{\begin{array}{l} Tu = f, \\ \beta_1 u = \cdots = \beta_n u = 0. \end{array}} \tag{2}$$

A boundary problem is called regular if for each $f \in \mathcal{F}$ there is exactly one $u \in \mathcal{F}$ satisfying (2). We want to solve a boundary problem not only for a fixed f but to compute the Green's operator mapping each forcing function f to its unique solution u. In other words, we solve a whole family of inhomogeneous differential equations, parameterized by a "symbolic" right-hand side f. We restrict ourselves to homogeneous conditions because the general solution is then obtained by adding a particular solution satisfying the inhomogeneous conditions.

For convenience, we shortly recall the abstract linear algebra setting for boundary problems over a vector space $\mathcal{F}$ as described in [16]. For $U \leq \mathcal{F}$ we define the *orthogonal* as $U^\perp = \{\beta \in \mathcal{F}^* : \beta(u) = 0 \text{ for all } u \in U\} \leq \mathcal{F}^*$. Similarly, for $\mathcal{B} \leq \mathcal{F}^*$, we define $\mathcal{B}^\perp = \{v \in \mathcal{F} : \beta(v) = 0 \text{ for all } \beta \in \mathcal{B}\} \leq \mathcal{F}$. A subspace U (resp. $\mathcal{B}$) is *orthogonally closed* if $U = U^{\perp\perp}$ (resp. $\mathcal{B} = \mathcal{B}^{\perp\perp}$). Every subspace $U \leq \mathcal{F}$ is orthogonally closed and every finite dimensional subspace $\mathcal{B} \leq \mathcal{F}^*$ is orthogonally closed. For a linear map $T \colon \mathcal{F} \to \mathcal{G}$ between vector spaces, the transpose map $T^* \colon \mathcal{G}^* \to \mathcal{F}^*$ is defined by $\gamma \mapsto \gamma \circ T$. The image of an orthogonally closed space under the transpose map is orthogonally closed.

A *boundary problem* is given by a pair $(T, \mathcal{B})$, where T is a surjective linear map and $\mathcal{B} \leq \mathcal{F}^*$ is an orthogonally closed subspace of the dual space. We call $u \in \mathcal{F}$ a solution of $(T, \mathcal{B})$ for a given $f \in \mathcal{F}$ if $Tu = f$ and $u \in \mathcal{B}^\perp$. A boundary problem is *regular* if for each f there exists a unique solution u. The *Green's operator* of a regular problem maps each f to its unique solution u. We also write $(T, \mathcal{B})^{-1}$ for the Green's operator. A boundary problem is *regular* iff $\mathcal{B}^\perp$ is a complement of $\operatorname{Ker} T$ so that $\mathcal{F} = \operatorname{Ker} T \dotplus \mathcal{B}^\perp$ as a direct sum.

For $\mathcal{F} = C^\infty[a, b]$, a monic differential operator T is always surjective and $\dim \operatorname{Ker} T = n < \infty$. Moreover, variation of constants can be used to compute a distinguished right inverse: If T has order n and $u_1, \ldots, u_n$ is a fundamental system for it, the *fundamental right inverse* is given by

$$T^\blacklozenge = \sum_{i=1}^{n} u_i {\textstyle\int} d^{-1} d_i, \tag{3}$$

where d is the determinant of the Wronskian matrix W for $(u_1, \ldots, u_n)$ and d_i the determinant of the matrix W_i obtained from W by replacing the i-th column by the n-th unit vector. Equation (3) is valid in arbitrary integro-differential algebras provided the n-th order operator T has a fundamental system $(u_1, \ldots, u_n)$ with invertible Wronskian matrix; see [11] or [12]. This will be assumed from now on, together with the condition $\dim \mathcal{B} < \infty$ appropriate for LODEs.

Regularity of a boundary problem $(T, \mathcal{B})$ can be tested algorithmically as follows. If $(u_1, \ldots, u_n)$ is a basis for $\operatorname{Ker} T$ and $(\beta_1, \ldots, \beta_m)$ for $\mathcal{B}$, we have a regular problem iff the *evaluation matrix*

$$\beta(u) = \begin{pmatrix} \beta_1(u_1) & \ldots & \beta_1(u_n) \\ \vdots & \ddots & \vdots \\ \beta_m(u_1) & \ldots & \beta_m(u_n) \end{pmatrix} \tag{4}$$

is regular; see [16, Cor. A.17] or [17, p. 184] for the special case of two-point boundary conditions. Of course this implies $m = n$, but we will consider more general types of boundary problems in Section 4 where this is no longer the case. It will also be convenient to use the notation (4) for arbitrary $u_1, \ldots, u_n \in \mathcal{F}$ and boundary conditions $\beta_1, \ldots, \beta_m$.

The algorithm for computing the Green's operator is described in detail in [11]; see also [2]. The main steps consist in computing the fundamental right inverse $T^\blacklozenge \in \mathcal{F}[\partial, {\textstyle\int}]$ from a given fundamental system as in (3) and the projector

$P \in \mathcal{F}[\partial, \textstyle\int]$ onto $\operatorname{Ker} T$ along $\mathcal{B}^{\perp}$. Then the Green's operator is then computed as $G = (1 - P)T^{\blacklozenge}$.

For a boundary problem we need to enter a monic differential operator T and a list of boundary conditions $(b1, \ldots, bm)$ as described in Section 2 in the form $\mathrm{BP}(T, \mathrm{BC}(b1, \ldots, bm))$. We use the Maple function *dsolve* for computing a fundamental system of T. As an example, we compute the Green's operator for the simplest two-point boundary problem $u'' = f$, $u(0) = u(1) = 0$. From the Green's operator for two-point boundary problems, we can extract the Green's function [18], which is usually used in Analysis to represent the Green's operator.

```
> T := DIFFOP(0,0,1):
> b1 := BOUNDOP(EVOP(0, EVDIFFOP(1), EVINTOP())):
> b2 := BOUNDOP(EVOP(1, EVDIFFOP(1), EVINTOP())):
> Bp := BP(T, BC(b1, b2));
```

$$Bp \;:=\; \mathrm{BP}(\mathrm{D}^2,\ \mathrm{BC}(\mathrm{E}[0],\ \mathrm{E}[1]))$$

```
> IsRegular(Bp);
```

$$true$$

```
> GreensOperator(Bp);
```

$$(x\ .\ \mathrm{A}) - (\mathrm{A}\ .\ x) - ((x\ \ \mathrm{E}[1])\ .\ \mathrm{A}) + ((x\ \ \mathrm{E}[1])\ .\ \mathrm{A}\ .\ x)$$

```
> GreensFunction(%);
```

$$\begin{cases} -\xi + x\xi & 0 <= \xi \text{ and } \xi <= x \text{ and } x <= 1 \\ -x + x\xi & 0 <= x \text{ and } x <= \xi \text{ and } \xi <= 1 \end{cases}$$

For simplifying boundary problems, we can apply factorizations into lower order problems along given factorizations of the differential operators. Further details and proofs of the following results can be found in [16] and [11]. The composition of two boundary problems $(T_1, \mathcal{B}_1)$ and $(T_2, \mathcal{B}_2)$ is defined as

$$(T_1, \mathcal{B}_1) \circ (T_2, \mathcal{B}_2) = (T_1 T_2, T_2^*(\mathcal{B}_1) + \mathcal{B}_2). \tag{5}$$

The composition $(T_1, \mathcal{B}_1) \circ (T_2, \mathcal{B}_2)$ of two regular boundary problems is regular with Green's operator

$$((T_1, \mathcal{B}_1) \circ (T_2, \mathcal{B}_2))^{-1} = (T_2, \mathcal{B}_2)^{-1}(T_1, \mathcal{B}_1)^{-1}. \tag{6}$$

Given a regular boundary problem $(T, \mathcal{B})$, every factorization $T = T_1 T_2$ can be lifted to a factorization $(T, \mathcal{B}) = (T_1, \mathcal{B}_1) \circ (T_2, \mathcal{B}_2)$, where $(T_1, \mathcal{B}_1)$ and $(T_2, \mathcal{B}_2)$ are regular and $\mathcal{B}_2 \leq \mathcal{B}$. For factorizing a differential operator, we use the function *DFactor* in the Maple package *DEtools*. As an easy example, we show how to factor the boundary problem from above; more examples for solving and factoring boundary problems can be found in our example worksheet.

```
> Bp := BP(T, BC(b1, b2));

                    Bp := BP(D^2, BC(E[0], E[1]))

> f1, f2 := FactorBoundaryProblem(Bp);

         f1, f2 := BP(D, BC(E[1] . A)), BP(D, BC(E[0]))
```

4 Singular Boundary Problems

For illustrating the main issues with singular boundary problems, we consider the boundary problem

$$\boxed{\begin{array}{l} u'' = f, \\ u'(0) = u'(1) = 0; \end{array}} \tag{7}$$

see for example [4, Page 215] or [18, Section 3.5] from a Symbolic Computation perspective. This problem is singular since it is not solvable for all $f \in \mathcal{F}$. It can easily be seen that if $u'' = f$, then f has to fulfill the *compatibility condition* $u'(1) = \int_0^1 f(\xi)\, d\xi = 0$. Moreover, uniqueness fails as well: If a solution $u \in \mathcal{F}$ exists, then also $u + c$ solves the problem for all $c \in \mathbb{R}$.

Our goal here is to generalize the symbolic approach of the previous section to problems of the kind (7). Since we want to compute generalized Green's operators, we cannot give up uniqueness of solutions—but we no longer require existence. Of course, uniqueness of solutions can always be achieved by imposing additional boundary conditions. On the other hand, adding too many conditions introduces new compatibility conditions, which we want to avoid (see after Lemma 1 for the precise statement). For the boundary problem (7), we can add for example the condition $u(1) = 0$ and consider the problem

$$\boxed{\begin{array}{l} u'' = f, \\ u'(0) = u'(1) = u(1) = 0. \end{array}} \tag{8}$$

This does not introduce any new compatibility conditions as we will see later (see before Lemma 2).

A boundary problem has at most one solution for each forcing function f iff $\mathcal{B}^{\perp} \cap \operatorname{Ker} T = \{0\}$. We see that for (7) we have $\mathcal{B}^{\perp} \cap \operatorname{Ker} T = \mathbb{R}$ while in (8) the intersection is $\{0\}$. The regularity test for boundary problems in terms of the evaluation matrix (4) can be generalized from the setting in Section 3.

Lemma 1. *Let $U = [u_1, \ldots, u_n] \leq \mathcal{F}$ and $\mathcal{B} = [\beta_1, \ldots, \beta_m] \leq \mathcal{F}^*$ with β_i and u_j linearly independent. Then $U \cap \mathcal{B}^{\perp} = \{0\}$ iff the evaluation matrix $\beta(u)$ has full column rank.*

Proof. Let b_j denote the columns of $\beta(u)$. The evaluation matrix has deficient column rank iff there exists a linear combination $\sum_{j=1}^n \lambda_j b_j = 0$ with at least one $\lambda_j \neq 0$. This is the case iff there exist a nonzero $u = \sum_{j=1}^n \lambda_j u_j \in U \cap \mathcal{B}_1^{\perp}$. $\square$

As mentioned for the example (8), singular boundary problems typically impose *compatibility conditions* on the admissible forcing functions. We can now make this precise: Clearly, a function f is admissible iff it is of the form Tu for a function u that satisfies the boundary conditions from $\mathcal{B}$, so the space of admissible functions is $T(\mathcal{B}^\perp)$. The compatibility conditions provide an implicit description of this space, comprising all those linear functionals that annihilate $T(\mathcal{B}^\perp)$. In other words, the compatibility conditions are the subspace $T(\mathcal{B}^\perp)^\perp$ of $\mathcal{F}^*$. This also makes precise what we mean by adding boundary conditions without imposing additional compatibility conditions: We enlarge $\mathcal{B}$ to $\tilde{\mathcal{B}}$ so as to ensure $\tilde{\mathcal{B}}^\perp \cap \operatorname{Ker} T = \{0\}$ despite retaining $T(\mathcal{B}^\perp) = T(\tilde{\mathcal{B}}^\perp)$.

For tackling the problem of existence, we modify the forcing function. In the example (8), this looks as follows: Since a solution exists only for forcing functions that fulfill $\int_0^1 f(\xi)\,d\xi = 0$, we consider the problem

$$\boxed{\begin{array}{l} u'' = f - \int_0^1 f(\xi)\,d\xi, \\ u'(0) = u'(1) = u(1) = 0, \end{array}} \tag{9}$$

which now always has a unique solution. For those f that fulfill the compatibility condition, problem (8) remains unchanged.

The general idea is that we project an arbitrary forcing function into the space of admissible functions. But this involves choosing those "exceptional functions" that we want to filter out. Even in the simple example (8), we might as well project f to $f - \frac{1}{2}x\int_0^1 f(\xi)\,d\xi$ instead of $f - \int_0^1 f(\xi)\,d\xi$. In the second case, we have filtered out the constant functions, in the first case the linear-homogeneous ones. The space $\mathcal{E}$ of exceptional functions can be any complement of the space $T(\mathcal{B}^\perp)$ of admissible functions, like $\mathcal{E} = [1]$ or $\mathcal{E} = [x]$ in this example.

Definition 1. *A* generalized boundary problem *is given by a triple $(T, \mathcal{B}, \mathcal{E})$, where $(T, \mathcal{B})$ is a boundary problem and $\mathcal{E} \le \mathcal{F}$. A generalized boundary problem is called* regular *if*

$$\mathcal{B}^\perp \cap \operatorname{Ker} T = \{0\} \quad \textit{and} \quad \mathcal{F} = T(\mathcal{B}^\perp) \dotplus \mathcal{E}.$$

The generalized Green's operator *maps each forcing function f to the unique solution of the boundary problem*

$$\boxed{\begin{array}{l} Tu = Qf, \\ \beta_1 u = \ldots = \beta_m u = 0, \end{array}}$$

where $\mathcal{B} = [\beta_1, \ldots, \beta_m]$ and Q is the projector onto $T(\mathcal{B}^\perp)$ along $\mathcal{E}$. We also write $(T, \mathcal{B}, \mathcal{E})^{-1}$ for the Green's operator.

If $(T, \mathcal{B}, \mathcal{E})$ is regular, the restriction $T|_{\mathcal{B}^\perp} \colon \mathcal{B}^\perp \to T(\mathcal{B}^\perp)$ is bijective. So the generalized Green's operator is given by

$$G = T|_{\mathcal{B}^\perp}^{-1} Q. \tag{10}$$

We begin with computing the projector Q. For this we derive first an explicit description of the space of compatibility conditions.

Proposition 1. *Let $(T, \mathcal{B}, \mathcal{E})$ be a generalized boundary problem and let G be any right inverse of T. Then we have*

$$T(\mathcal{B}^\perp)^\perp = G^*(\mathcal{B} \cap (\operatorname{Ker} T)^\perp). \tag{11}$$

Moreover, $\dim T(\mathcal{B}^\perp)^\perp = \dim \mathcal{E}$ *for any complement* $\mathcal{E}$ *with* $\mathcal{F} = T(\mathcal{B}^\perp) \dot{+} \mathcal{E}$.

Proof. With [16, Prop. A.6], we see that $T(\mathcal{B}^\perp)^\perp = (T^*)^{-1}(\mathcal{B})$. Since T is surjective, T^* is injective, and for any right inverse G of T, G^* is a left inverse of T^*. Hence $(T^*)^{-1}(\mathcal{B}) = G^*(\mathcal{B} \cap \operatorname{Im} T^*)$ by [16, Prop. A.13]. Again by [16, Prop. A.6], we have $\operatorname{Im} T^* = (\operatorname{Ker} T)^\perp$, and hence $T(\mathcal{B}^\perp)^\perp = G^*(\mathcal{B} \cap (\operatorname{Ker} T)^\perp)$.

Since $\dim \mathcal{B} < \infty$, by the first statement also $\dim T(\mathcal{B}^\perp)^\perp < \infty$. But $T(\mathcal{B}^\perp)$ is orthogonally closed; see for example [16, Section A.1]. Therefore we obtain

$$\dim T(\mathcal{B}^\perp)^\perp = \operatorname{codim} T(\mathcal{B}^\perp)^{\perp\perp} = \operatorname{codim} T(\mathcal{B}^\perp),$$

and the statement follows immediately from [16, Prop. A.14]. □

Note that $s = \dim \mathcal{E} = \operatorname{codim} T(\mathcal{B}^\perp)$ counts the number of (linearly independent) compatibility conditions. Equation (11) is the key for an algorithmic description of the projector Q onto $T(\mathcal{B}^\perp)$ along $\mathcal{E}$. The space $\mathcal{E}$ is given as part of the problem description, and it can be specified by a basis $(w_1, \dots, w_s)$. Since the other space $T(\mathcal{B}^\perp)$ has finite codimension s, it can be specified in terms of s linearly independent compatibility conditions, and Equation (11) can be used to compute these in terms of T and $\mathcal{B}$. For that we just have to determine a basis of $\mathcal{B} \cap (\operatorname{Ker} T)^\perp$ and then apply any right inverse G of T, for example the fundamental right inverse $T^\blacklozenge$ defined in Section 3.

For determining a basis of $\mathcal{B} \cap (\operatorname{Ker} T)^\perp$ we first compute the kernel of the transpose of the evaluation matrix $\beta(u)$, where $(u_1, \dots, u_n)$ is any basis of $\operatorname{Ker} T$ and $(\beta_1, \dots, \beta_m)$ any basis of $\mathcal{B}$. If $w = (w_1, \dots, w_m)^t \in \operatorname{Ker} \beta(u)^t$, then

$$w^t(\beta_1, \dots, \beta_m)^t = \sum_{i=1}^m w_i \beta_i \in \mathcal{B} \cap (\operatorname{Ker} T)^\perp,$$

hence a basis of $\mathcal{B} \cap (\operatorname{Ker} T)^\perp$ can be obtained by computing the products $(v_1^t(\beta_1, \dots, \beta_m)^t, \dots, v_k^t(\beta_1, \dots, \beta_m)^t)$, where $(v_1, \dots, v_k)$ is a basis of $\operatorname{Ker} \beta(u)^t$.

Using Proposition 1, we can now verify that the compatibility conditions of the boundary problems (7) and (8) are the same. In both cases we have $T = \partial^2$, so we can choose the fundamental right inverse $\textstyle\int\!\int = x\int - \int x$ and $(1, x)$ as a basis of $\operatorname{Ker} T$. The evaluation matrices are given by

$$\beta(u) = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad \beta(u) = \begin{pmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}.$$

In the first case, a basis of $\beta(u)^t$ is given by $((-1, 1)^t)$, hence $(\mathrm{E}_1\partial - \mathrm{E}_0\partial)$ is a basis of $\mathcal{B} \cap (\operatorname{Ker} T)^\perp$. In the second case, a basis of $\beta(u)^t$ is given by $((-1, 1, 0)^t)$

and the basis of $\mathcal{B} \cap (\operatorname{Ker} T)^\perp$ is again $(\mathrm{E}_1\partial - \mathrm{E}_0\partial)$. Multiplying this basis by the right inverse of T, we get as a basis for the compatibility conditions

$$\begin{aligned}(\mathrm{E}_1\partial - \mathrm{E}_0\partial)\cdot(x{\textstyle\int} - {\textstyle\int} x) &= \mathrm{E}_1(x\partial+1){\textstyle\int} - \mathrm{E}_0(x\partial+1){\textstyle\int} - \mathrm{E}_1\partial{\textstyle\int} x + \mathrm{E}_0\partial{\textstyle\int} x\\ &= \mathrm{E}_1 x + \mathrm{E}_1{\textstyle\int} - \mathrm{E}_0 x - \mathrm{E}_0{\textstyle\int} - \mathrm{E}_1 x + \mathrm{E}_0 x = \mathrm{E}_1{\textstyle\int} = {\textstyle\int_0^1},\end{aligned}$$

which agrees with our heuristic considerations after (7).

We can now compute the projector Q just as the kernel projector P for standard boundary problems (mentioned in Section 3). If $(\kappa_1, \ldots, \kappa_s)$ is a basis for the compatibility conditions $T(\mathcal{B}^\perp)^\perp$ and $(w_1, \ldots, w_s)$ a basis for $\mathcal{E}$, then the corresponding evaluation matrix $\kappa(w)$ is regular by Lemma 1, which can be applied to $\mathcal{F} = T(\mathcal{B}^\perp) \dotplus \mathcal{E} = T(\mathcal{B}^\perp)^{\perp\perp} \dotplus \mathcal{E}$ since $T(\mathcal{B}^\perp)$ is orthogonally closed. Hence we can compute the projector Q onto $T(\mathcal{B}^\perp)$ along $\mathcal{E}$ as

$$Q = 1 - \sum_{i=1}^{s} w_i \tilde{\kappa}_i,$$

where $(\tilde{\kappa}_1, \ldots, \tilde{\kappa}_s)^t = \kappa(w)^{-1} \cdot (\kappa_1, \ldots, \kappa_s)^t$; see for example [16, Lemma A.1].

The final step for computing the generalized Green's operator (10) is to find the inverse function $T|_{\mathcal{B}^\perp}^{-1}$. In the regular case, we started with an arbitrary right inverse of T and multiplied with a projection onto $\mathcal{B}^\perp$ along $\operatorname{Ker} T$. But this step cannot be generalized to our setting. Our approach is to embed the generalized problem into a standard one in the following sense.

First note that the evaluation matrix of a regular generalized boundary problem has full column rank by Lemma 1, so it has a left inverse.

Lemma 2. *Let $(T, \mathcal{B}, \mathcal{E})$ be a regular generalized boundary problem. Let $\beta(u)^-$ be a left inverse of $\beta(u)$ and $(\tilde{\beta}_1, \ldots, \tilde{\beta}_n)^t = \beta(u)^-(\beta_1, \ldots, \beta_m)^t$. Then the boundary problem $(T, \tilde{\mathcal{B}})$ is regular, where $\tilde{\mathcal{B}} \leq \mathcal{B}$ is spanned by $\tilde{\beta}_1, \ldots, \tilde{\beta}_n$.*

The proof of the statement is obvious, since the evaluation matrix $\tilde{\beta}(u)$ is given by $\beta(u)^-\beta(u) = 1_n$. Hence the problem $(T, \tilde{\mathcal{B}})$ is regular. In our package, we always choose the Moore-Penrose pseudoinverse as a left inverse $\beta(u)^-$ of the evaluation matrix $\beta(u)$. The generalized boundary problem (8) for example embeds into the standard boundary problem

$$\boxed{\begin{array}{l} u'' = f,\\ u'(0) + u'(1) - 2\,u(1) = u'(0) + u'(1) = 0. \end{array}} \tag{12}$$

The Green's operator for this regular problem according to Section 3 is given by $x{\textstyle\int} - {\textstyle\int} x - \frac{1}{2}(x+1) + {\textstyle\int_0^1} x$. The next proposition tells us how to compute the generalized Green' s operator from it.

Proposition 2. *Let $(T, \mathcal{B}, \mathcal{E})$ be a regular generalized boundary problem and let $(T, \tilde{\mathcal{B}})$ be a regular boundary problem with $\tilde{\mathcal{B}} \leq \mathcal{B}$. Then*

$$(T, \mathcal{B}, \mathcal{E})^{-1} = (T, \tilde{\mathcal{B}})^{-1}\, Q,$$

where Q is the projector onto $T(\mathcal{B}^\perp)$ along $\mathcal{E}$.

Proof. Since $\tilde{\mathcal{B}} \leq \mathcal{B}$, we have $\mathcal{B}^\perp \leq \tilde{\mathcal{B}}^\perp$. Hence the maps $T|_{\mathcal{B}^\perp}^{-1}$ and $\tilde{G} = (T, \tilde{\mathcal{B}})^{-1}$ coincide on $\mathcal{B}^\perp$. Since $T|_{\mathcal{B}^\perp} \colon \mathcal{B}^\perp \to T(\mathcal{B}^\perp)$ is a bijection, we can compute the restriction $T|_{\mathcal{B}^\perp}^{-1}$ by first applying a projector onto $T(\mathcal{B}^\perp)$ and then $\tilde{G}$. Hence $T|_{\mathcal{B}^\perp}^{-1} = \tilde{G}Q$, where Q is again the projection onto $T(\mathcal{B}^\perp)$ along $\mathcal{E}$. Hence the generalized Green's operator is given by $G = T|_{\mathcal{B}^\perp}^{-1} Q = \tilde{G}Q^2 = \tilde{G}Q$. $\square$

Applying the previous proposition to Example (8) leads to the generalized Green's operator $x{\textstyle\int} - {\textstyle\int} x - \frac{1}{2}(x^2+1){\textstyle\int}_0^1 + {\textstyle\int}_0^1 x$. For a more involved example illustrating the MAPLE functions in our package, we refer to the Appendix.

5 Outlook

We are currently investigating in how far the composition of boundary problems (5) can be extended to generalized boundary problems such that an analog of the "reverse order law" (6) holds. We can see in the example below (13) that for such a generalization, we also have to modify the second component with the boundary conditions. The question under which conditions a reverse order law holds for different classes of generalized inverses—not necessarily related to integro-differential operators—is extensively studied in the literature, see for example [19] and the references therein.

The search for generalized composition laws is intimately connected with the question of "embedding" a singular boundary problem into a regular problem of higher order. For example in [18], the Green's operator G of the generalized boundary problem $(\partial^2, [\mathrm{E}_0\partial, \mathrm{E}_1\partial, {\textstyle\int}_0^1], [1])$ can be factored as $G = \tilde{G} \circ \partial$ where $\tilde{G}$ is the standard Green's operator of the boundary problem $(\partial^3, [\mathrm{E}_0\partial, \mathrm{E}_1\partial, {\textstyle\int}_0^1])$. Hence $\tilde{G} = G \circ {\textstyle\int}_0^x$ and, assuming (6) for the composition, also

$$(\partial^3, [\mathrm{E}_0\partial, \mathrm{E}_1\partial, {\textstyle\int}_0^1], [0]) = (\partial, [\mathrm{E}_0], [0]) \circ (\partial^2, [\mathrm{E}_0\partial, \mathrm{E}_1\partial, {\textstyle\int}_0^1], [1]), \qquad (13)$$

since ${\textstyle\int}_0^x$ is the Green's operator of the boundary problem $(\partial, [\mathrm{E}_0])$. The singular second-order problem is thus embedded into a regular third-order one.

Multi-point boundary problems can also be treated by our method, yielding a suitable Green's operator just as in the classical two-point setting. Generalizing the extraction procedure for Green's functions is future work, see [20] for an analytic description of Green's functions for multi-point boundary problems.

Going from LODEs to LPDEs, more drastic changes are necessary since geometry enters the picture. For example, the Green's operator of the inhomogeneous wave equation $u_{xx} - u_{tt} = f$ with homogeneous Dirichlet data on the x-axis integrates f over a certain triangle whose tip is at (x, t). In terms of the operator algebra, this means one must incorporate the chain and substitution rule along with explicit operators encoding change of variables. A first approach along these lines, for the very simple case of linear coordinate changes, was presented in [2] and is currently being refined. Studying singular boundary problems for LPDEs from a symbolic point of view is also very interesting; see for example [21] for a Gröbner bases approach to compute the (hierarchy of) compatibility conditions for elliptic boundary problems. It would be tempting to combine the tools of involutive systems used there with the setting of operator rings used here.

References

1. Korporal, A., Regensburger, G., Rosenkranz, M.: A Maple package for integro-differential operators and boundary problems. ACM Commun. Comput. Algebra 44(3), 120–122 (2010); Also presented as a poster at ISSAC 2010
2. Rosenkranz, M., Regensburger, G., Tec, L., Buchberger, B.: A symbolic framework for operations on linear boundary problems. In: Gerdt, V.P., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2009. LNCS, vol. 5743, pp. 269–283. Springer, Heidelberg (2009)
3. Coddington, E.A., Levinson, N.: Theory of ordinary differential equations. McGraw-Hill Book Company, Inc., New York (1955)
4. Stakgold, I.: Green's functions and boundary value problems. John Wiley & Sons, New York (1979)
5. Duffy, D.G.: Green's functions with applications. Studies in Advanced Mathematics. Chapman & Hall/CRC, Boca Raton, FL (2001)
6. Agarwal, R.P., O'Regan, D.: An introduction to ordinary differential equations. Universitext. Springer, New York (2008)
7. Loud, W.S.: Some examples of generalized Green's functions and generalized Green's matrices. SIAM Rev. 12, 194–210 (1970)
8. Ben-Israel, A., Greville, T.N.E.: Generalized inverses, 2nd edn. Springer, New York (2003)
9. Boichuk, A.A., Samoilenko, A.M.: Generalized inverse operators and Fredholm boundary-value problems. VSP, Utrecht (2004)
10. Nashed, M.Z., Rall, L.B.: Annotated bibliography on generalized inverses and applications. In: Generalized Inverses and Applications, pp. 771–1041. Academic Press, New York (1976)
11. Rosenkranz, M., Regensburger, G.: Solving and factoring boundary problems for linear ordinary differential equations in differential algebras. J. Symbolic Comput. 43(8), 515–544 (2008)
12. Rosenkranz, M., Regensburger, G., Tec, L., Buchberger, B.: Symbolic analysis for boundary problems: From rewriting to parametrized Gröbner bases. In: Langer, U., Paule, P. (eds.) Numerical and Symbolic Scientific Computing: Progress and Prospects. Springer, Wien (to appear, 2011)
13. Guo, L., Keigher, W.: On differential Rota-Baxter algebras. J. Pure Appl. Algebra 212(3), 522–540 (2008)
14. Brown, R.C., Krall, A.M.: Ordinary differential operators under Stieltjes boundary conditions. Trans. Amer. Math. Soc. 198, 73–92 (1974)
15. Regensburger, G., Rosenkranz, M., Middeke, J.: A skew polynomial approach to integro-differential operators. In: May, J.P. (ed.) Proceedings of ISSAC 2009, pp. 287–294. ACM, New York (2009)
16. Regensburger, G., Rosenkranz, M.: An algebraic foundation for factoring linear boundary problems. Ann. Mat. Pura Appl. (4) 188(1), 123–151 (2009)
17. Kamke, E.: Differentialgleichungen. Lösungsmethoden und Lösungen. Teil I: Gewöhnliche Differentialgleichungen. Akademische Verlagsgesellschaft, Leipzig (1967)
18. Rosenkranz, M.: A new symbolic method for solving linear two-point boundary value problems on the level of operators. J. Symbolic Comput. 39(2), 171–199 (2005)
19. Djordjević, D.S.: Further results on the reverse order law for generalized inverses. SIAM J. Matrix Anal. Appl. 29(4), 1242–1246 (2007)

20. Agarwal, R.P.: Boundary value problems for higher order differential equations. World Scientific Publishing Co. Inc., Teaneck (1986)
21. Krupchyk, K., Tuomela, J.: The Shapiro-Lopatinskij condition for elliptic boundary value problems. LMS Journal of Computation and Mathematics 9, 287–329 (2006)

A Example

Now we will give a detailed example for computations with generalized boundary problems. We introduced a new datatype GBP$(T, \text{BC}(b1, \ldots, bm), \text{ES}(f1, \ldots, fk))$, where T and $(b1, \ldots, bm)$ again are a differential operator and boundary conditions and $(f1, \ldots, fm)$ is a basis of the exceptional space. We added the new procedures *CompatibilityConditions, IsComplement* and *Projector*, which will be explained later and extended the procedures *GreensOperator* and *IsRegular*. The first one now also computes the Green's Operator for a generalized boundary problem and the second one tests the condition $\operatorname{Ker} T \cap \mathcal{B}^{\perp} = \{0\}$ also for generalized boundary problems.

We consider the more complicated example

$$\boxed{\begin{array}{l} u'''' + u'' = f \\ u'(0) = u''(0) = u''(\pi) = u'''(0) = u'''(\pi) = 0. \end{array}} \tag{14}$$

We enter the boundary problem stated above and compute a fundamental system for the differential operator $T = \mathrm{D}^4 + \mathrm{D}^2$.

```
> T := DIFFOP(0, 0, 1, 0, 1):
> b[1] := BOUNDOP(EVOP(0, EVDIFFOP(0, 1), EVINTOP())):
> b[2] := BOUNDOP(EVOP(0, EVDIFFOP(0, 0, 1), EVINTOP())):
> b[3] := BOUNDOP(EVOP(0, EVDIFFOP(0, 0, 0, 1), EVINTOP())):
> b[4] := BOUNDOP(EVOP(Pi, EVDIFFOP(0, 0, 1), EVINTOP())):
> b[5] := BOUNDOP(EVOP(Pi, EVDIFFOP(0, 0, 0, 1), EVINTOP())):
> Bp := BP(T, BC(b[1],b[2],b[3],b[4],b[5])):
> fs := FundamentalSystem(T);
```

$$[x,\ \sin(x),\ \cos(x),\ 1]$$

Now we add another boundary condition b[6] in order to achieve uniqueness of solutions. This can be checked by considering the column rank of the evaluation matrix. We further verify that the compatibility conditions of both problems are the same.

```
> b[6] := BOUNDOP(EVOP(Pi, ZEROEDOP, EVINTOP(EVINTTERM(1,1)))):
> BpA := BP(T, BC(b[1],b[2],b[3],b[4],b[5],b[6])):
> IsRegular(BpA);
```

$$true$$

```
> CompatibilityConditions(Bp);
```

$$\mathrm{BC}((\mathrm{E}[\mathrm{Pi}]\,.\,\mathrm{A}\,.\,(\sin(x)),\ (\mathrm{E}[\mathrm{Pi}])\,.\,\mathrm{A}\,.\,(\cos(x)))$$

```
> CompatibilityConditions(BpA);
```

$$\mathrm{BC}((\mathrm{E}[\mathrm{Pi}]\,.\,\mathrm{A}\,.\,(\sin(x)),\ (\mathrm{E}[\mathrm{Pi}])\,.\,\mathrm{A}\,.\,(\cos(x)))$$

Now we enter a generalized boundary problem and check that our choice $[1,\ x]$ as exceptional space is a complement of $T(\mathcal{B}^{\perp})$. Then we compute the projector Q onto $T(\mathcal{B}^{\perp})$ and the Green's operator for the generalized boundary problem $(T, [\mathrm{b}[1], \mathrm{b}[2], \mathrm{b}[3], \mathrm{b}[4], \mathrm{b}[5], \mathrm{b}[6]], [1, x])$,

```
> gBp := GBP(T, BC(b[1],b[2],b[3],b[4],b[5],b[6]), ES(1,x)):
> IsComplement(gBp);
```

$$true$$

```
> Q := Projector(gBp):
```

$$Q := 1 - \frac{1}{2}((\mathrm{E}[\mathrm{Pi}])\,.\,A\,.\,(\sin(x))) + \left(\left(-\frac{\mathrm{Pi}}{4} + \frac{x}{2}\right).\,(\mathrm{E}[\mathrm{Pi}])\,.\,A\,.\,(\cos(x)\right)$$

```
> G := GreensOperator(gBp):
```

Finally we verify that the Green's operator G fulfills the equation $TG = Q$ and the six boundary conditions.

```
> simplify(SubtractOperator(MultiplyOperator(T, G), Q))
```

$$0$$

```
> seq(simplify(MultiplyOperator(b[i], G)), i=1..6);
```

$$0,\ 0,\ 0,\ 0,\ 0,\ 0$$

Algebraic Structures as Typed Objects

Heinz Kredel[1] and Raphael Jolly[2]

[1] IT-Center, University of Mannheim, Germany
[2] Databeans, Paris, France
kredel@rz.uni-mannheim.de, raphael.jolly@free.fr

Abstract. Following the research direction of strongly typed, generic, object oriented computer algebra software, we examine the modeling of algebraic structures as typed objects in this paper. We discuss the design and implementation of algebraic and transcendental extension fields together with the modeling of real algebraic and complex algebraic extension fields. We will show that the modeling of the relation between algebraic and real algebraic extension fields using the delegation design concept has advantages over the modeling as sub-types using sub-class implementation. We further present a summary of design problems, which we have encountered so far with our implementation in Java and present possible solutions in Scala.

1 Introduction

We proposed a software architecture for computer algebra systems which builds on other software projects as much as possible in [1] . Only the parts specific to computer algebra are then to be implemented. We identified three major parts for computer algebra software.

– run-time infrastructure with memory management and parallelism,
– statically typed object oriented algorithm libraries,
– dynamic interactive scripting interpreters.

In this paper we elaborate on the second point: the research area of strongly typed, generic, object oriented computer algebra systems, namely the modeling of algebraic structures as typed objects. A prominent part of algebra deals with the study of field extensions. To make use of extension fields in computers we need to restrict ourselves to effective constructions which can be performed on a computer. Such fields are named *computable fields*.

Given some base fields, especially the prime fields, like rational numbers or modular integers (computing modulo a prime number), we will examine the design and implementation of algebraic and transcendental extension fields. Algebraic extension fields can be constructed as modular univariate polynomials (computing modulo an irreducible polynomial) and are as such computable fields. Transcendental extension fields can be constructed as (multivariate) polynomial fractions (also called rational functions) and as we can efficiently compute multivariate polynomial greatest common divisors to remove common factors, these fields are also computable. Sub-fields of algebraic extension fields, like real algebraic and complex algebraic extension fields are also of great importance and are also computable fields. The design and the implementation of these field extensions together with the modeling of the relation between algebraic and real algebraic extension fields is the topic of this paper.

When trying to implement our research agenda in a given programming language (Java) – in contrast of inventing a suitable language [2] – some design problems will eventually remain. We give a summary of such remaining design problems or implementation trade-offs and show how to solve them in Scala [3].

1.1 Related Work

The related work published on type systems for computer algebra or abstract data type (ADT) approaches to computer algebra has been summarized in [1]. Type-safe design considerations in computer algebra are mostly centered around the *axiom* computer algebra system and are described, e.g. starting with [4,2]. In application areas like constructive algebraic topology there exists strong demand for type-safe algorithms. For example the Kenzo system takes an object oriented approach with strong run-time type system [5]. The system handles 'chain complexes', which consist of algebraic structures together with mappings between them. Also for constructions in the formal theory of differential equations there is demand for object oriented software, see e.g. [6]. MuPAD has a simple object oriented layer for algebraic structures and so called categories, see [7]. DoCon has support for field extension towers in a Haskell package [8]. Using a suitable existing programming language has restrictions compared to the Axiom approach [2]. However, there are also benefits: we do not have to invent and improve memory management, parallel computing and networking support [9] and can ride on the advantages of computer science in this area [10]. Further related work is mentioned in the paper as required.

1.2 Outline

In Section 2 we discuss the design and implementation of algebraic and transcendental extension fields together with the modeling of real algebraic and complex algebraic extension fields. Section 3 presents a summary of design problems of our typed object-oriented approach when using Java as implementation language and presents possible solutions in Scala. Finally Section 4 draws some conclusions.

2 Algebraic Structures as Typed Objects

In this section we first give an introduction into the object oriented type systems. For more details see our earlier articles [1,11,12,13]. Then we discuss the design and implementation of algebraic and transcendental extension fields together with the modeling of real algebraic and complex algebraic extension fields. The constructed extension fields can be used in other algorithms on polynomials with coefficients from such fields, for example in (parallel) Gröbner base or greatest common divisor computations. Other algorithms like polynomial factorization will however require an implemented case for such fields. Due to space limitations we will not discuss performance, but will give some hints on computing times and possible improvements. We will also not be able to discuss mappings of elements between the various extension rings and fields, as for example evaluation homomorphisms between polynomial rings. Currently all such mappings and their application have to be coded explicitly, but some automatic mapping construction and convenient coercion would be desirable, at least in scripting interpreters (for a special case see Subsection 3.2).

2.1 Ring Elements and Ring Factories

The basic building blocks of the type system consists of the interfaces `RingElem` and `RingFactory` and the classes which implement them, see figure 1. `RingElem` defines the methods which we expect to be available on all ring elements, for example `multiply()`, `isZERO()` or `isUnit()` with the obvious meanings. The construction of ring elements is done by factories, modeled after the *abstract factory* creational design pattern [14]. The factory `RingFactory` defines the construction methods for elements, for example `getONE()` to create the one element from the ring, `parse()` to create an element from a string representation or query methods such as `isAssociative()`.

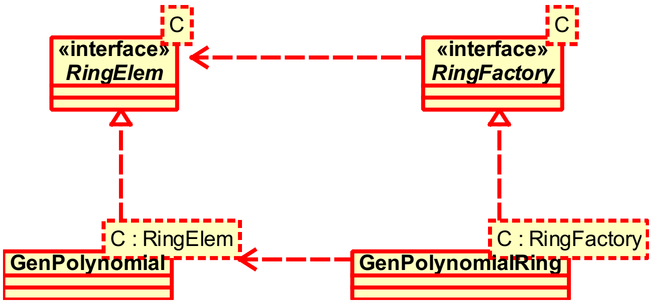


Fig. 1. Basic types

The polynomial class `GenPolynomial` with type parameter `C` for the coefficient type implements the `RingElem` interface and specifies that coefficients must be of type `RingElem`. In addition to the methods mandated by the interface, the `GenPolynomial` implements the methods like `leadingMonomial()` or `degree()`. Polynomials are to be created with a polynomial factory `GenPolynomialRing`. In addition to the ring factory methods it defines for example methods to create random polynomials. The constructor for `GenPolynomialRing` takes parameters for a factory for the coefficients, the number of variables, the names for the variables and a term order object `TermOrder`. The relation between the factory of the coefficients and the polynomial ring factory is modeled after the (constructor) dependency injection pattern and implements the inversion of control principle.

Figure 1 shows dependency arrows from the factories to the element interfaces as factories create the respective elements. The modeling of the constructors is not shown as it is not denotable in Java. The constructors of ring elements implement an opposite dependency : each constructor takes a corresponding ring factory as parameter. It is only indirectly enforced since the `RingElem` interface specifies a method `factory()` to obtain the corresponding factory.

The factory methods are not static (which is apparent from the modeling as an interface) since a ring factory might depend on other rings or specific parameters. In case of a polynomial factory it depends on a factory for the coefficients and at least the number of variables of the polynomial ring. By this modeling the types of elements of algebraic structures are not simply denoted in the program text but have to be created as programming objects (by instantiating the respective classes). Type denotations show up explicitly in Java program code and are mostly inferred in Scala via type resolution.

For example a polynomial $w^2 - 2 \in \mathbb{Q}[w]$ can be constructed by first constructing an object for the ring $\mathbb{Q}[w]$ and then reading and constructing the polynomial $w^2 - 2$ with the factory method `parse()`.

```
BigRational rf = new BigRational(1); // here element = factory
GenPolynomialRing<BigRational> pf
 = new GenPolynomialRing<BigRational>(rf,new String[]{ "w" });
GenPolynomial<BigRational> a = pf.parse("w^2 - 2");
```

This example is continued in Subsection 2.3.

2.2 Algorithms and Factories

Our implemented algorithms are in fact meta-algorithms or functors. They do not only compute elements of algebraic structures but simultaneously construct the required algebraic structures during the computation. So several implemented methods map pairs of algebraic structures together with some elements to other algebraic structures and elements. For example Hensel lifting is an algorithm which maps

$$((\mathbb{Z}[x], a), (\mathbb{Z}_p[x], (a_1, ..., a_r)), (\mathbb{N}, k)) \mapsto (\mathbb{Z}_{p^k}[x], (b_1, ..., b_r)).$$

With the meaning $(a \in \mathbb{Z}[x], (a_1, ..., a_r) \in \mathbb{Z}_p[x]^r, k \in \mathbb{N}) \mapsto (b_1, ..., b_r) \in \mathbb{Z}_{p^k}[x]^r$. Using type annotations it would read

$$(a : \mathbb{Z}[x], (a_1, ..., a_r) : \mathbb{Z}_p[x]^r, k : \mathbb{N}) \mapsto (b_1, ..., b_r) : \mathbb{Z}_{p^k}[x]^r.$$

It is important to understand that $\mathbb{Z}_{p^k}[x]$ is an object constructed during the computation. The type annotation hides this fact. Note, the rings $\mathbb{Z}_p$ and $\mathbb{Z}_{p^k}$ are not distinguished by a Java type. To correctly model this as distinct types needs the concept of *dependent types* which is not available in Java 6 but is available in Scala, see Subsection 3.2.

2.3 Algebraic and Transcendental Field Extensions

In this subsection we discuss the typed object oriented modeling of algebraic extension rings / fields, see figure 2. In slight abuse of number theoretic terminology we call elements of algebraic extensions *algebraic numbers*. The modeling of real and complex algebraic numbers is discussed in the next subsection (2.4). The construction of quotient fields of polynomial rings is not discussed further as it does not present new modeling challenges. For an introduction to algebraic fields and more references see Subsection 2.1 in [15].

Algebraic numbers are elements of some algebraic extension field L of a field K. If L is generated by a single element α we write $L = K(\alpha)$. L being an algebraic extension then means that there exists a polynomial $f \in K[x]$ such that $f(\alpha) = 0$ in L. If we can compute in K, the field L can be represented as $L = K[x]_{/(f)}$ and we can compute also in L. This representation is slightly ambiguous as it is not specified which conjugate of α is meant. With this representation L can be implemented as the residue class ring of $K[x]$ modulo the defining monic minimal polynomial $f \in K[x]$. It is implemented by class `AlgebraicNumberRing` which implements the interface `RingFactory` as mentioned before. Elements of this ring are implemented by class `AlgebraicNumber` implementing the interface `RingElem`. The constructor for `AlgebraicNumber-Ring` requires the defining polynomial to be provided. The name 'ring' is chosen, because the defining polynomial might not be irreducible. The constructor for `AlgebraicNumber` elements requires the corresponding algebraic number factory and an element $a \in K[x]$, which represents $a(\alpha) \in K(\alpha)$.

The construction of algebraic extension fields works uniformly for all fields K. For example it could be $\mathbb{Z}_p$, $\mathbb{Q}$, another algebraic extension $K(\beta)$ or a transcendental extension $K(y)$. Transcendental extensions are implemented by classes `QuotientRing` with elements `Quotient` from package `edu.jas.ufd`. By this design we can implement arbitrary towers of field extensions of some base field $\mathbb{Z}_p$ or the rational numbers $\mathbb{Q}$. For example, we can construct

$$\mathbb{Q}(\sqrt{2})(x)(\sqrt{x}) \text{ or } \mathbb{Z}_p(x)[y].$$

The latter denotes a polynomial ring over an infinite field of finite characteristic. For example the construction of the first field can be done step by step as in the following sequence.

$$\begin{aligned}\mathbb{Q} \mapsto_1 \mathbb{Q}[w] \mapsto_2 \mathbb{Q}[w]_{/(w^2-2)} &\mapsto_3 (\mathbb{Q}[w]_{/(w^2-2)})(x) \\ \mapsto_4 (\mathbb{Q}[w]_{/(w^2-2)})(x)[wx] &\mapsto_5 (\mathbb{Q}[w]_{/(w^2-2)})(x)[wx]_{/(wx^2-x)}\end{aligned}$$

Step 1 constructs a polynomial ring over the rational numbers, step 2 constructs an algebraic number ring with polynomial $w^2 - 2$ to represent $\mathbb{Q}(\sqrt{2})$. Step 3 constructs a polynomial ring in x and the quotient field of it, Step 4 constructs a polynomial ring for the ring from step 3 and finally step 5 constructs an algebraic number ring with polynomial $wx^2 - x$ to represent $\mathbb{Q}(\sqrt{2})(x)(\sqrt{x})$. These steps can be followed exactly in the programming language with the construction of objects from the classes `GenPolynomialRing`, `AlgebraicNumberRing` and

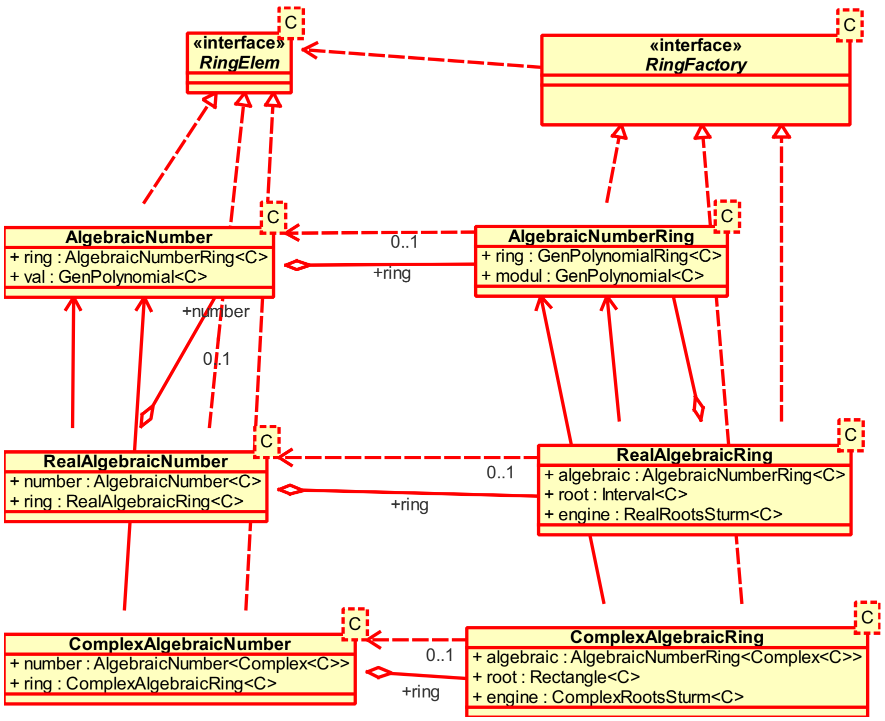

«interface»
RingElem
C
«interface»
RingFactory
C
AlgebraicNumber
+ ring : AlgebraicNumberRing<C>
+ val : GenPolynomial<C>
C
AlgebraicNumberRing
+ ring : GenPolynomialRing<C>
+ modul : GenPolynomial<C>
C
0..1
+ring
+number
0..1
RealAlgebraicNumber
+ number : AlgebraicNumber<C>
+ ring : RealAlgebraicRing<C>
C
RealAlgebraicRing
+ algebraic : AlgebraicNumberRing<C>
+ root : Interval<C>
+ engine : RealRootsSturm<C>
C
0..1
+ring
ComplexAlgebraicNumber
+ number : AlgebraicNumber<Complex<C>>
+ ring : ComplexAlgebraicRing<C>
C
ComplexAlgebraicRing
+ algebraic : AlgebraicNumberRing<Complex<C>>
+ root : Rectangle<C>
+ engine : ComplexRootsSturm<C>
C
0..1
+ring

Fig. 2. Algebraic Number classes

`QuotientRing`. Again, the construction is explicit and we can therefore build any such field extension towers as desired.

In the example, an element of $\mathbb{Q}(\sqrt{2})(x)(\sqrt{x})$ is denoted as

```
AlgebraicNumber<Quotient<AlgebraicNumber<BigRational>>> elem;
```

When creating elements of these rings it is first necessary to construct the corresponding algebraic structures as programming objects in the same sequence as above. A sequence to create a factory for `elem` could be as follows.

```
... // see above
GenPolynomial<BigRational> a = pf.parse("w^2 - 2");
AlgebraicNumberRing<BigRational> af
 = new AlgebraicNumberRing<BigRational>(a);
String[] vx = new String[]{ "x" };
GenPolynomialRing<AlgebraicNumber<BigRational>> tf
 = new GenPolynomialRing<AlgebraicNumber<BigRational>>(af,vx);
QuotientRing<AlgebraicNumber<BigRational>> qf
 = new QuotientRing<AlgebraicNumber<BigRational>>(tf);
String[] vw = new String[]{ "wx" };
GenPolynomialRing<Quotient<AlgebraicNumber<BigRational>>> qaf
 = new GenPolynomialRing<Quotient<AlgebraicNumber<BigRational>>>(qf,vw);
```

```
GenPolynomial<Quotient<AlgebraicNumber<BigRational>>> b
 = qaf.parse("wx^2 - x");
AlgebraicNumberRing<Quotient<AlgebraicNumber<BigRational>>> fac
 = new AlgebraicNumberRing<Quotient<AlgebraicNumber<BigRational>>>(b);
```

The first field extension $\mathbb{Q}(\sqrt{2})$ is constructed as object in `af` as algebraic number ring. Then the transcendental extension $\mathbb{Q}(\sqrt{2})(x)$ is constructed as quotient field in object `qf`. And the last extension $\mathbb{Q}(\sqrt{2})(x)(\sqrt{x})$ is constructed again as algebraic number field in object `fac`. Then it is possible to construct elements from this field, e.g. `elem = fac.parse("wx + x^5");`

The construction process looks tedious and contains a lot of 'boiler plate' type denotations. However, the construction is very precise, it is type safe and explicit! So every type and semantics of algebraic structures can be precisely constructed, an important advantage for the engineering of mathematical software libraries. Note, the type denotations are minimized in Scala through its type resolution capabilities, see Subsection 3.2.

Recall, that in Java the type information is 'erased' in the byte code and so it is not accessible at run-time. So using Java and (Java based) Scala classes in Jython [16] or JRuby [17] scripting interpreters access the raw objects and we have only run-time type safety, see Subsection 2.5. In any case the construction of algebraic structures is the same and so the construction of elements of such structures is run-time type safe, as the corresponding factories must be constructed explicitly.

The construction process can be shortened by a class implementing the *builder pattern* [14] to make it easier as follows.

```
RingFactory fac = ExtensionFieldBuilder
                  .baseField(new BigRational(1))
                  .algebraicExtension("w", "w^2 - 2")
                  .transcendentExtension("x")
                  .algebraicExtension("wx", "wx^2 - x")
                  .build();
```

There are several algorithms which can work with such field towers. For example in [1] we have shown that one can factor polynomials with coefficients from the ring $\mathbb{Q}(\sqrt{2})(x)(\sqrt{x})$. Primitive elements for multiple algebraic extensions can be computed by methods `primitiveElement()` from class `PolyUtilApp`. Moreover, residue class rings modulo multivariate (prime) ideals can be used as extension rings. Note, the `build()` method is a perfect place to implement structural optimizations and simplifications of the field tower. For example moving algebraic extensions to the "bottom" of the tower and moving transcendental extensions to the "top" of the field tower, or replacing some algebraic extensions by primitive elements, or replacing the tower by a single multivariate residue class ring represented by a Gröbner base. Also type specialization techniques could be implemented in this method, see [18].

2.4 Real Algebraic Numbers and Complex Algebraic Numbers

Real algebraic numbers $g(\alpha)$ are elements of a real algebraic field extension $L = K(\alpha)$ over a field K, for $\alpha \in \mathbb{R}$. They are represented as polynomials

$g \in K[x]$ modulo a defining polynomial $f \in K[x]$ for the algebraic field extension together with an isolating interval $I = [l, r] \subseteq \mathbb{R}$ for a specific real root $\alpha \in I$ of f. The rationale for real algebraic numbers as programming objects is the ability to represent results of polynomial and ideal root finding algorithms precisely and not as unstructured isolating intervals, as it is done by contemporary commercial computer algebra systems. Note, $K(\alpha)$ is a sub-field of $E = K[x]_{/(f)}$. So one would like to implement $K(\alpha)$ as a sub-type of E via the sub-class implementation scheme.

However, to avoid the problem of type erasure for sub-classes with interfaces we model the relation according to the *delegation concept*, a compositional design concept [14]. That is, we do not model real and complex algebraic numbers as sub-classes of algebraic numbers but use delegation to algebraic numbers and algebraic number factories, see figure 2. The main advantages and disadvantages of delegation versus the inheritance approach are discussed in Subsection 3.1.

The implementation is contained in class `RealAlgebraicNumber` with factory class `RealAlgebraicRing`. The factory class contains an instance of the real root computation `engine`, which is used to refine intervals as required. The real root computation is contained in class `RealRootsSturm` for computation via Sturm sequences. The class is a sub-class of `RealRootAbstract`. There exist faster algorithms to compute isolating intervals than Sturm sequences, they will be implemented in future releases (see Subsection 2.1 in [15]). This design allows then the definition of polynomials with real algebraic coefficients

```
GenPolynomial<RealAlgebraicNumber<BigRational>>.
```

Moreover, for such polynomials we can also use real root isolation algorithms and instantiate and use for example

```
RealRootsSturm<RealAlgebraicNumber<BigRational>>.
```

This is possible since we implemented method `realSign()` which is used in the method `signum()` of a real algebraic number. `BigRational` and `RealAlgebraicNumber` implement the interface `Rational` which defines a method `getRational()` to compute a rational approximation of the middle of the isolating interval to a prescribed accuracy.

Multiple real algebraic field extensions, for example by the third root of 3 and its square root and the fifth root of 2 $\mathbb{Q}(+\sqrt[3]{3})(+\sqrt{+\sqrt[3]{3}})(+\sqrt[5]{2})$ using $[1, 2]$ as isolating intervals, can be constructed as follows.

```
fac = ExtensionFieldBuilder
      .baseField(new BigRational())
      .realAlgebraicExtension("q", "q^3 - 3","[1,2]")
      .realAlgebraicExtension("w", "w^2 - q","[1,2]")
      .realAlgebraicExtension("s", "s^5 - 2","[1,2]")
      .build();
```

One possible implementation of complex algebraic numbers is similar to real algebraic numbers but using a bounding box in the complex plane to uniquely identify a specific complex algebraic number. Such a implementation uses the complex root computation from classes `ComplexRootsSturm` which is a subclass of `ComplexRootsAbstract`. Unfortunately this representation of complex algebraic numbers can not be used in a recursive setting since it is not possible to obtain a real algebraic representation for the real or imaginary parts from it. As a consequence `ComplexRootsSturm<ComplexAlgebraicNumber<.>>` can not be implemented. An alternate representation is as real roots of the ideal generated by the real and imaginary part of the given polynomial. That is, after substitution of $z \mapsto a + bi$ in the polynomial $f(z)$, we have $f(a,b) = f_r(a,b) + f_i(a,b)i$. Then we consider the real roots of the ideal generated by $f_r(a,b)$ and $f_i(a,b)$. So a specific $\gamma_k \in L = L'(i)$ with $f(\gamma_k) = 0$ is represented as $\alpha_k + \beta_k i \in L'(i)$ with $f_r(\alpha_k, \beta_k) = 0$ and $f_i(\alpha_k, \beta_k) = 0$ where $L' = K(\alpha, \beta)$ with real algebraic numbers α and β. All required algorithms are already implemented in classes `Ideal` and `PolyUtilApp` in package `edu.jas.application`. So we arrive at a representation as `Complex<RealAlgebraicNumber<RealAlgebraic-Number<.>>>` which is suitable in the recursion, i.e. for complex root computation of polynomials with coefficients from such rings[1].

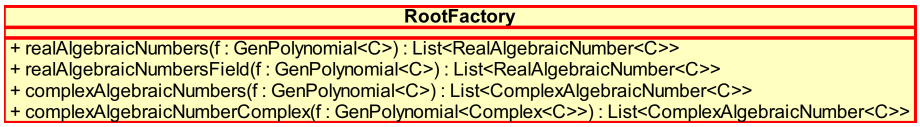

RootFactory
+ realAlgebraicNumbers(f : GenPolynomial<C>) : List<RealAlgebraicNumber<C>>
+ realAlgebraicNumbersField(f : GenPolynomial<C>) : List<RealAlgebraicNumber<C>>
+ complexAlgebraicNumbers(f : GenPolynomial<C>) : List<ComplexAlgebraicNumber<C>>
+ complexAlgebraicNumberComplex(f : GenPolynomial<Complex<C>>) : List<ComplexAlgebraicNumber<C>>

Fig. 3. Factory for algebraic numbers and fields

We conclude with the class `RootFactory`, see figure 3. It implements a functor for creating real algebraic numbers for polynomial real roots and a functor for creating complex algebraic numbers for polynomial complex roots. All functors internally construct first the real algebraic number rings respectively the complex algebraic number rings. The rings are accessible by the `factory()` method of the `RingElem` interface and an approximation of the magnitude can be obtained via the `getRational()` method. The respective methods `realAlgebraicRoots()` and `complexAlgebraicRoots()` for zero dimensional ideals are at the moment contained in class `PolyUtilApp` in package `edu.jas.application`.

For example we can compute real roots over the field `fac` using this root factory. Therefore we build a polynomial ring over `fac` in the variable, say `y`, parse for example the polynomial $y^2 - \sqrt{\sqrt[3]{3}\sqrt[5]{2}}$ and compute its two real roots.

```
GenPolynomial elem = pfac.parse("y^2 - w s");
List<RealAlgebraicNumber> roots = RootFactory.realAlgebraicNumbers(elem);
```

The real root isolation needs 1.2 seconds and the approximation to 50 digits needs a total of 5.2 seconds on an AMD running at 3 GHz and IcedTea6 JVM. The decimal approximation (via `getRational()` from the roots) shows the two real roots requested with 50 decimal digits as

[1] According to the fundamental theorem of algebra, for a constructive version see the Weierstraß-Durand-Kerner fixpoint method.

```
-1.1745272686769866126436905900619307101229226521299
 1.1745272686769866126436905900619307101229226521299.
```

2.5 Algebraic Structures in Interactive Scripting Interpreters

The example $\mathbb{Q}(\sqrt{2})(x)(\sqrt{x})$ from Subsection 2.3 can be constructed in a scripting interpreter in the same way as in the Java example (when not using `ExtensionFieldBuilder`). The (Jython) methods `AN()` and `RF()` construct algebraic respectively transcendental extensions and the `PolyRing` class represents a `GenPolynomialRing`. The `gens()` method for a ring R returns a list of generators as an R-algebra (including generators for coefficient rings represented in R). The generators are constructed in the sequence defined by the extension tower composition. `QQ` denotes the rational numbers.

```
Q = PolyRing(QQ(),"w2",PolyRing.lex);
[e,w2] = Q.gens();
root = w2**2 - 2;
Q2 = AN(root,field=True);
Qp = PolyRing(Q2,"x",PolyRing.lex);
Qr = RF(Qp);
Qwx = PolyRing(Qr,"wx",PolyRing.lex);
[ewx,wwx,ax,wx] = Qwx.gens();
rootx = wx**2 - ax;
Q2x = AN(rootx,field=True);
```

Finally a polynomial ring over this field extension can be constructed.

```
Yr = PolyRing(Q2x,"y",PolyRing.lex)
[e,w2,x,wx,y] = Yr.gens();
f = ( y**2 - x ) * ( y**2 - 2 ); // = y**4 - ( x + 2 ) * y**2 + 2 * x
```

Note, the variable `x`, set by the `gens()` method, correctly represents the generator `x` from ring `Qp` but as element of the polynomial ring `Yr`. There is a source of confusion with the method `gens()` as all returned generators must be listed in the assignment (and in the correct semantic sequence). So we also see the unused variables `e` (1 in `Yr`), `ewx` (1 in `Qwx`), `wwx` ($\sqrt{2}$ in `Qwx`), `ax` (x in `Qwx`) and `w2` ($\sqrt{2}$ in `Yr`). These usability problems can be solved in Scala as described in Subsection 3.2 or by an extension field builder for the scripting interface. For example with `EF(QQ()).extend("w2","w2^2 - 2").extend("x") .extend("wx","wx^2 - x").build()`.

In the example we can then compute the factorization over the extension fields (in 9.5 seconds on an AMD at 3 GHz, 5.7 seconds after JIT warm-up, also IcedTea6 JVM) and this looks as expected, `f = ( y - wx ) * ( y - w2 ) * ( y + wx ) * ( y + w2 )`.

Note, in scripting interpreters we only have run-time type safety, as the scripts are not compiled and statically type checked.

3 Problems

In this section, we summarize some problems we have studied in the library design and implementation. As we are using general purpose languages and not developing our own like in Axiom [2], there will inevitably exist some design problems which will have no satisfactory solution until said languages are made to evolve, if ever.

3.1 Generic Types and Sub-classes

The first one can be demonstrated with the following JAS Java code defining the classes for polynomials

```
class GenPolynomial<C extends RingElem<C>>
      implements RingElem<GenPolynomial<C>> { ... }
```

and solvable polynomials

```
class GenSolvablePolynomial<C extends RingElem<C>>
      extends GenPolynomial<C> { ... }
```

The intention of this sub-class definition[2] is to be able to use algorithms written for polynomials also for solvable polynomials. One would like to add "`implements RingElem<GenSolvablePolynomial<C>>`" to the declaration of `GenSolvablePolynomial`. However this will lead to two usages of the `RingElem` interface with different type parameters. Due to the design of Java generics to be compatible with existing non-generic code, the nested type parameters are *erased* and the double interface usage would turn into a compile error.

The same problem occurs with classes `AlgebraicNumber` and `RealAlgebraic-Number` that we have studied in Subsection 2.4. If the latter could be made a subclass of the former, we could reuse algorithms, e.g. for primitive element computation. However, then we could not have `RealAlgebraicNumber<C extends GcdRingElem<C>>` to implement `GcdRingElem<RealAlgebraicNumber<C>>` but only `GcdRingElem<AlgebraicNumber<C>>`. This means that we could build polynomials over such a ring, but no further real algebraic number fields based on it, because for example no `getRational()` method is provided by the interface `AlgebraicNumber<C>`.

Thus we are conducted to resort to the delegation concept, which avoids the above type-erasure problem, but forbids to use algorithms written specifically for `AlgebraicNumbers` with `RealAlgebraicNumbers`.

A third solution, that we have investigated in ScAS, is to use neither delegation nor subclassing, but instead to have both classes to inherit from a common, abstract superclass[3]. That way, code reuse is made possible, and the problem of knowing what class should be a subtype/subset of the other, is avoided.

3.2 Dependent Types

A second problem is that in order to parametrize the list of variables of a polynomial or the module of integer residue classes, some dependent types are required

[2] Which should be reversed to be mathematically right, the problem here being that subclassing does not provide for subtyping in a rigorous sense, namely that of a subset of possible values.

[3] As explained in [13], Subsection 7.1 "Interfaces as types", such abstract class is roughly equivalent to a category in Axiom.

(see [13], Subsection 7.3 "Dependent types"). We have investigated if we could use Scala's dependent types [19] and found that it is possible. In the rest of the subsection some familiarity with Scala is required.

The basic principle is illustrated below. Let us take modular integers for instance. We need to define a type like `Mod(7)` which depends on the value `7`. The goal is to forbid arithmetic operations between integers with different moduli. In the current state of the library this is not done:

```
object Ring {
  trait Factory[T <: Ring[T]]
}
trait Ring[T <: Ring[T]] {
  val factory: Ring.Factory[T]
  def +(that: T): T
}
object Mod {
  def apply(mod: Int) = new Factory(mod)
  class Factory(val mod: Int) extends Ring.Factory[Mod] {
    def apply(value: Int) = new Mod(this)(value%mod)
  }
}
class Mod(val factory: Mod.Factory)(val value: Int)
         extends Ring[Mod] {
  def +(that: Mod) = factory(this.value+that.value)
  override def toString = value.toString
}
```

A use case is given below.

```
val r = Mod(7)
r(4)+r(4) // 1
val s = Mod(2)
r(4)+s(1) // problem : this works
```

So we have found a new design, where we have replaced the type parameter `T` by an abstract type member `E`, and now we can define the ring element as an inner class of the ring factory, or, in other words, as a type which depends on the factory.

```
trait Ring {
  type E <: Element
  trait Element {
    def +(that: E): E
  }
}
class Mod(val mod: Int) extends Ring {
  type E = Element
  class Element(val value: Int) extends super.Element {
    def +(that: E) = apply(this.value+that.value)
    override def toString = value.toString
  }
```

```
    def apply(value: Int) = new Element(value%mod)
  }
```

A use case is given again below.

```
object r extends Mod(7)
r(4)+r(4) // 1
object s extends Mod(2)
r(4)+s(1) // type mismatch, as expected
```

In the case of polynomials we could have a dependent type which fulfills all our requirements as follows:

```
class Polynomial[C <: Ring, P](val ring: C,
                               val variables: Array[String],
                               val ordering: Comparator[P]) {
  type E // the type of the elements of the ring
}
```

When using Scala as a script interpreter [20], the design will moreover provide a solution to the problem mentioned in Subsection 2.5 that each definition of ring/extension field factory must redefine all generators in the factory tower. This will be achieved through implicit conversion, with such factory declarations as below, which will be able to "lift" values to the correct level in the ring/field tower.

```
implicit r extends Mod(7)
implicit p extends Polynomial(r, Array("x"))
implicit q extends Polynomial(p, Array("y"))
// and so on
```

We intend to rewrite the whole ScAS library based on this new principle. It will be the subject of a future publication.

3.3 Package Structure

Lastly, there is a question about whether several algorithm flavors (for e.g. gcd, factorization and so on) could be implemented as polynomial (factory) subclasses as is currently investigated in ScAS, or should remain in distinct class hierarchies as in JAS (see [13], Subsection 7.5 "Recursive types").

4 Conclusions

We have discussed previously [1], how our typed object oriented approach with the Java and Scala programming languages makes it possible to implement non-trivial, type-safe algebraic structures which can be stacked and plugged together in unprecedented ways. In this paper we examined the modeling of field extensions using ring / field factories which represent the corresponding algebraic structures. All examples and the underlying mathematical algorithms from Section 2 (together with almost all algorithms from [21]) have been implemented and are available under a GPL license from a Git-repository at [11].

The construction process is very precise and explicit, so that no misinterpretation of the algebraic structure is possible. However, it is tedious and contains a lot of 'boiler plate' type denotations in case of Java as implementation language. The type denotations can be minimized in the Scala implementation through its type resolution capabilities. In scripting interpreters the type safety is only enforced at run-time due to the interpretative execution. But for the engineering of reliable and comprehensive mathematical software libraries the precise construction and type-safe modeling of algebraic structures is an important advantage. Compared to other computer algebra systems we can represent real roots not only by simple isolating intervals but as elements of algebraic structures which can be reused for further computations.

The design problems we encountered in Java can be resolved by modeling using alternative ways and in more advanced object oriented programming languages like Scala. Dependent types and the coercion facility in Scala will need further studies. Future work will include the implementation of faster algorithms for root isolation and to improve the (recursive) complex root isolation and representation.

Acknowledgments. We thank our colleagues Thomas Becker, Wolfgang K. Seiler, Thomas Sturm, Axel Kramer, Jaime Gutierrez, Sherm Ostrowsky, Ted Kosan and others for various discussions on the design of and the requirements for JAS and ScAS. Thanks also to the referees for the insightful suggestions to improve the paper.

References

1. Kredel, H., Jolly, R.: Generic, type-safe and object oriented computer algebra software. In: Gerdt, V.P., Koepf, W., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2010. LNCS, vol. 6244, pp. 162–177. Springer, Heidelberg (2010)
2. Watt, S.: Aldor. In: Computer Algebra Handbook, pp. 265–270. Springer, Heidelberg (2003)
3. Odersky, M.: The Scala programming language. Technical report (2003-2011), http://www.scala-lang.org/ (accessed June 2011)
4. Jenks, R., Sutor, R. (eds.): Axiom The Scientific Computation System. Springer, Heidelberg (1992)
5. Rubio, J., Sergeraert, F.: Constructive algebraic topology. Bulletin des Sciences Mathematiques 126(5), 389–412 (2002)
6. Calmet, J., Seiler, W.M.: Computer algebra and field theories. Mathematics and Computers in Simulation 45, 33–37 (1998)
7. Drescher, K.: MuPAD multi processing algebra data tool - Axioms, Categories and Domains. Technical report, Manuscript available via Citeseer (1995), http://www2.math.uni-paderborn.de/
8. Mechveliani, S.: DoCon - The Algebraic Domain Constructor. Technical report (2007), http://botik.ru/pub/local/Mechveliani/docon/
9. Maza, M.M., Stephenson, B., Watt, S.M., Xie, Y.: Multiprocessed parallelism support in ALDOR on SMPs and multicores. In: PASCO, pp. 60–68 (2007)

10. Taboada, G., Tourino, J., Doallo, R.: Java for high performance computing: Assessment of current research and practice. In: Proc. PPPJ 2009, pp. 30–39. ACM, New York (2009)
11. Kredel, H.: The Java algebra system (JAS). Technical report (since 2000), `http://krum.rz.uni-mannheim.de/jas/`
12. Kredel, H.: Evaluation of a java computer algebra system. In: Kapur, D. (ed.) ASCM 2007. LNCS (LNAI), vol. 5081, pp. 121–138. Springer, Heidelberg (2008)
13. Kredel, H.: On a Java Computer Algebra System, its performance and applications. Science of Computer Programming 70(2-3), 185–207 (2008)
14. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns. Addison-Wesley, Reading (1995); Entwurfsmuster, Addison-Wesley, Deutsch (1996)
15. Grabmaier, J., Kaltofen, E., Weispfenning, V. (eds.): Computer Algebra Handbook. Springer, Heidelberg (2003)
16. Jython Developers: Jython implementation of the high-level, dynamic, object-oriented language Python written in 100% pure Java. Technical report (1997-2011), `http://www.jython.org/` (accessed June 2011)
17. JRuby Developers: JRuby a Java powered Ruby implementation. Technical report (2003-2011), `http://jruby.org/` (accessed June 2011)
18. Dragan, L., Watt, S.: Type specialization in aldor. In: Gerdt, V.P., Koepf, W., Mayr, E.W., Vorozhtsov, E.V. (eds.) CASC 2010. LNCS, vol. 6244, pp. 73–84. Springer, Heidelberg (2010)
19. Odersky, M., Cremet, V., Röckl, C., Zenger, M.: A nominal theory of objects with dependent types. In: Cardelli, L. (ed.) ECOOP 2003. LNCS, vol. 2743, pp. 303–329. Springer, Heidelberg (2003)
20. Jolly, R.: Object Scala found - a JSR223-compliant version of the Scala interpreter. In: Scala Days 2011 (to appear, 2011)
21. Becker, T., Weispfenning, V.: Gröbner Bases - A Computational Approach to Commutative Algebra. Graduate Texts in Mathematics. Springer, Heidelberg (1993)

On Two-Generated Non-commutative Algebras Subject to the Affine Relation

Viktor Levandovskyy[1], Christoph Koutschan[2], and Oleksandr Motsak[3]

[1] Lehrstuhl D für Mathematik, RWTH Aachen, Germany
viktor.levandovskyy@math.rwth-aachen.de
[2] RISC, Johannes Kepler University, Linz, Austria
Koutschan@risc.uni-linz.ac.at
[3] TU Kaiserslautern, Germany
motsak@mathematik.uni-kl.de

Abstract. We consider algebras over a field $\mathbb{K}$, generated by two variables x and y subject to the single relation $yx = qxy + \alpha x + \beta y + \gamma$ for $q \in \mathbb{K}^*$ and $\alpha, \beta, \gamma \in \mathbb{K}$. We prove, that among such algebras there are precisely five isomorphism classes. The representatives of these classes, which are ubiquitous operator algebras, are called model algebras. We derive explicit multiplication formulas for $y^m \cdot x^n$ in terms of standard monomials $x^i y^j$ for many algebras of the considered type. Such formulas are used in e. g. establishing formulas of binomial type and in an implementation of non-commutative multiplication in a computer algebra system. By using the formulas we also study centers and ring-theoretic properties of the non-commutative model algebras.

In this paper we study non-commutative algebras in two generators obeying single affine relation. Many operator algebras, coming from different areas of natural sciences, are built from algebras in two generators, see Sect. 2.1 for examples. One of generators, say x, often corresponds to the operator of the multiplication with the function x. Another operator, say y, corresponds to a linear operator, acting on functions in the variable x.

In the main Theorem we identify precisely five types of non-isomorphic algebras, which we call model algebras, among them. Despite the fact that many such algebras have been studied in the literature (see e. g. [3,6,2,1], many aspects and properties are too scattered in the existing literature. Another point of this note is to search systematically for closed form of multiplication formulas on monomials. Such closed forms are needed, among other, in computer algebra, where many sophisticated algorithms heavily rely on basic multiplication among monomials. It is not enough to have such formulas just for model algebras, since isomorphisms do not preserve monomials but turn them into polynomials. It turned out, that there are still several cases, where we were not able to derive closed formulas in terms of standard monomials. With our approach, however, one is still able to derive formulas of certain type for them.

1 Preliminaries

Let $\mathbb{K}$ be a field. Moreover, let A be an associative $\mathbb{K}$-algebra and $q \in \mathbb{K}^*$. We use the following notations: $[a,b]_q := ab - q \cdot ba$ is a *q-commutator* of $a, b \in A$. The *commutator* or the *Lie bracket* is $[a,b] := [a,b]_1 = ab - ba$. We also write $[n] = [n]_q = \frac{q^n-1}{q-1}$ for the q-number, $(a;q)_n := \prod_{k=0}^{n-1}(1-aq^k)$ for the q-Pochhammer symbol, $[n]^{\underline{k}} = \frac{(q^{n-k+1};q)_k}{(1-q)^k}$ for the q-falling factorial and $\begin{bmatrix} n \\ k \end{bmatrix} = \begin{bmatrix} n \\ k \end{bmatrix}_q = \frac{[n]!}{[n-k]![k]!}$ for the q-binomial coefficient. Note, that $\begin{bmatrix} n \\ k \end{bmatrix} = 0$ for $k > n$.

Lemma 1. *$\forall a, b, c \in A$ and $\lambda, \mu \in \mathbb{K}$ the following identities hold.*

- $[a,b]_q = -q(ba - \frac{1}{q}ab) = -q[b,a]_{q^{-1}}, \quad [a,a]_q = (1-q)a^2$
- $[a+\lambda, b]_q = [a,b]_q + \lambda(1-q)b, \quad [a, b+\mu]_q = [a,b]_q + \mu(1-q)a$
- $[ab, c]_q = a[b,c]_q + q \cdot [a,c]b = a[b,c] + [a,c]_q b$

In particular, $[a,b] = -[b,a]$ and $[ab,c] = a[b,c] + [a,c]b$.

We study two-generated non-commutative $\mathbb{K}$-algebras with affine relations

$$A(q,\alpha,\beta,\gamma) := \mathbb{K}\langle x, y \mid yx = q \cdot xy + \alpha x + \beta y + \gamma \rangle$$

for $q \in \mathbb{K}^*$ and $\alpha, \beta, \gamma \in \mathbb{K}$. The scalar q plays an important role and we distinguish two major cases. If $q = 1$, an algebra is of *Lie type*, that is it is isomorphic to a factor-algebra of the universal enveloping algebra of a finite-dimensional Lie algebra. If $q \neq 1$, in the research of *quantum algebras* one distinguishes two situations (which lead to different behaviour of algebras): either q is transcendental over some subfield $k \subset \mathbb{K}$ or q is a root of unity in $\mathbb{K}$. Without assumptions on q we will write $\mathbb{K}(q)$ in general (thus encompassing the case $q \in \mathbb{K}^*$ as well), while in the case $q = 1$ just $\mathbb{K}$ will be used. The following Lemma is well-known.

Lemma 2. *$A(q,\alpha,\beta,\gamma)$ has $\{x^a y^b \mid (a,b) \in \mathbb{N}_0^2\}$ as a $\mathbb{K}(q)$-basis.*

2 Main Theorem and Applications

Theorem 1. *$A(q,\alpha,\beta,\gamma)$ is isomorphic to one of the five* ***model algebras****:*

1. *the commutative algebra $\mathbb{K}[x,y]$,*
2. *the first Weyl algebra $A_1 = \mathbb{K}\langle x, d \mid dx = xd + 1\rangle$ (the algebra of linear differential operators with coefficients from $\mathbb{K}[x]$),*
3. *the shift algebra $S_1 = \mathbb{K}\langle x, s \mid sx = xs + s\rangle$ (the universal enveloping algebra of the non-abelian solvable two-dimensional Lie algebra; the algebra of linear shift operators with coefficients from $\mathbb{K}[x]$),*
4. *the q-commutative algebra $\mathbb{K}_q[x,y] := \mathbb{K}(q)\langle x, y \mid yx = q \cdot xy\rangle$ (Manin's quantum plane; the algebra of linear q-shift operators with coeff's from $\mathbb{K}(q)[x]$)*
5. *the first q-Weyl algebra $A_1^{(q)} = \mathbb{K}(q)\langle x, \partial \mid \partial x = q \cdot x\partial + 1\rangle$ (the algebra of linear q-differential operators with coefficients from $\mathbb{K}(q)[x]$).*

Moreover, the model algebras are pairwise non-isomorphic (see Prop. 3).

In Tables 1 and 2 we write isomorphisms to model algebras and write formulas for the multiplication in every concrete class of algebras. In some cases we also write down the recurrence formulas for the coefficients in the expansion of $y^m x^n$ in terms of standard monomials $x^a y^b$. For some algebras we put simpler formulas for $y^m x$ and yx^n as well as a part of our proof.

By writing **not known yet** in the table we mean, that up to now, no explicit formula in terms of of standard monomials is known to us. However, by applying an isomorphism (for instance, the one we give explicitly in the table) to the explicit multiplication formula of the corresponding model algebra (Algebra Class in the table), we obtain a non-expanded formula for any algebra in the table.

Proof. While for some of the above cases the explicit formulas for $y^m x^n$ are rather simple (and therefore easily found), others are quite complicated and required some work. A good strategy for finding a general formula for $y^m x^n$ is to study the special cases yx^n and $y^m x$ first. Once this is done, further multiplications by y (and x, respectively) lead to the general formula. However, for the most general commutation rules (e.g., $yx = xy + \alpha x + \beta y + \gamma$), this strategy fails.

All the formulas for $y^m x$, yx^n, and $y^m x^n$ have in common that they are easily proved by induction. As an example, consider the algebra $(1, 0, \beta, \gamma)$. We have stated above that $y^m x = xy^m + my^{m-1}(\beta y + \gamma)$. For $m = 1$ this reduces just to the given commutation relation $yx = xy + \beta y + \gamma$. Now consider $y^{m+1}x = y \cdot (y^m x)$ which by induction hypothesis is $yxy^m + my^m(\beta y + \gamma) = (xy + \beta y + \gamma)y^m + my^m(\beta y + \gamma)$ which after collecting powers gives the desired formula $xy^{m+1} + (m + 1)y^m(\beta y + \gamma)$. Similarly, the general formula

$$y^m x^n = \frac{1}{\beta^m} \sum_{i=0}^{m} \binom{m}{i} (-\gamma)^{m-i} (x + i\beta)^n (\beta y + \gamma)^i$$

can be shown (now we use induction on n). A straightforward calculation shows that this formula for $n = 1$ reduces to the one given above for $y^m x$. Thus it has to be investigated what happens after multiplying another x from the right:

$$\begin{aligned}(\beta y + \gamma)^i x &= \sum_{j=0}^{i} \binom{i}{j} (\beta y)^j \gamma^{i-j} x = \sum_{j=0}^{i} \binom{i}{j} \beta^j \left(xy^j + jy^{j-1}(\beta y + \gamma) \right) \gamma^{i-j} \\ &= x \sum_{j=0}^{i} \binom{i}{j} \beta^j y^j \gamma^{i-j} + \beta \left(\sum_{j=0}^{i} \binom{i-1}{j-1} i(\beta y)^{j-1} \gamma^{i-j} \right) (\beta y + \gamma) \\ &= x(\beta y + \gamma)^i + \beta i(\beta y + \gamma)^{i-1}(\beta y + \gamma) \\ &= (x + \beta i)(\beta y + \gamma)^i\end{aligned}$$

We have additionally checked the validity of the formulas above with our respective implementations in computer algebra systems Singular:Plural [5] and Mathematica [4].

Table 1. Multiplication Formulas for Algebras of Lie Type

Algebra Type	Relation	Algebra Class
$(1,0,0,0)$	$yx = xy$	$YX = XY$
Isomorphism: $X \to x\ ,\quad Y \to y$ $y^m x^n = x^n y^m$		
$(1,\alpha,0,0)$	$yx = xy + \alpha x$	$YX = XY + Y$
Isomorphism: $X \to -\alpha^{-1} y\ ,\quad Y \to x$ $y^m x^n = x^n (y + n\alpha)^m = \sum_{k=0}^{m} \binom{m}{k} (n\alpha)^{m-k} x^n y^k,$ Coeff. recurrence: $C_k = \frac{(k+1)n\alpha}{m-k} C_{k+1}$		
$(1,0,\beta,0)$	$yx = xy + \beta y$	$YX = XY + Y$
Isomorphism: $X \to \beta^{-1} x\ ,\quad Y \to y$ $y^m x^n = (x + m\beta)^n y^m = \sum_{k=0}^{n} \binom{n}{k} (m\beta)^{n-k} x^k y^m,$ Coeff. recurrence: $C_k = \frac{(k+1)m\beta}{n-k} C_{k+1}$		
$(1,\alpha,\beta,0)$	$yx = xy + \alpha x + \beta y$	$YX = XY + Y$
Isomorphism: $X \to -\alpha^{-1} y\ ,\quad Y \to \alpha x + \beta y$ $yx^n = \frac{1}{\beta}\big((x+\beta)^n(\alpha x + \beta y) - \alpha x^{n+1}\big),\ \ y^m x = \frac{1}{\alpha}\big((\alpha x + \beta y)(y+\alpha)^m - \beta y^{m+1}\big),$ $y^m x^n =$ not known yet		
$(1,0,0,\gamma)$	$yx = xy + \gamma$	$YX = XY + 1$
Isomorphism: $X \to x\ ,\quad Y \to \gamma^{-1} y$ $yx^n = x^{n-1}(xy + n\gamma),\ \ y^m x = (xy + m\gamma) y^{m-1},$ $y^m x^n = \sum_{k=0}^{n} \binom{m}{k} n^{\underline{k}} \gamma^k x^{n-k} y^{m-k} = \sum_{k=0}^{\min\{m,n\}} \frac{m!n!\gamma^k x^{n-k} y^{m-k}}{k!(m-k)!(n-k)!},$ Coeff. recurrence: $C_k = \frac{(m-k+1)(n-k+1)\gamma}{k} C_{k-1}$		
$(1,\alpha,0,\gamma)$	$yx = xy + \alpha x + \gamma$	$YX = XY + Y$
Isomorphism: $X \to -\alpha^{-1} y\ ,\quad Y \to \alpha x + \gamma$ $yx^n = x^n y + n x^{n-1}(\alpha x + \gamma),\ \ y^m x = \frac{1}{\alpha}\big((\alpha x + \gamma)(y+\alpha)^m - \gamma y^m\big),$ $y^m x^n = \frac{1}{\alpha^n} \sum_{i=0}^{n} \binom{n}{i} (-\gamma)^{n-i} (\alpha x + \gamma)^i (y + i\alpha)^m$		
$(1,0,\beta,\gamma)$	$yx = xy + \beta y + \gamma$	$YX = XY + Y$
Isomorphism: $X \to \beta^{-1} x\ ,\quad Y \to \beta y + \gamma$ $y^m x = x y^m + m y^{m-1}(\beta y + \gamma),\ \ yx^n = \frac{1}{\beta}\big((x+\beta)^n(\beta y + \gamma) - \gamma x^n\big),$ $y^m x^n = \frac{1}{\beta^m} \sum_{i=0}^{m} \binom{m}{i} (-\gamma)^{m-i} (x + i\beta)^n (\beta y + \gamma)^i$		
$(1,\alpha,\beta,\gamma)$	$yx = xy + \alpha x + \beta y + \gamma$	$YX = XY + Y$
Isomorphism: $X \to -\alpha^{-1} y\ ,\quad Y \to \alpha x + \beta y + \gamma$ $y^m x^n =$ not known yet		

Table 2. Multiplication Formulas for Quantum Algebras

Algebra Type	Commutation	Algebra Class
$(q,0,0,0)$	$yx = qxy$	$YX = qXY$
Isomorphism: $X \to x\ ,\quad Y \to y$ $y^m x^n = q^{mn} x^n y^m$		
$(q,\alpha,0,0)$	$yx = qxy + \alpha x$	$YX = qXY$
Isomorphism: $X \to x\ ,\quad Y \to y - \alpha(1-q)^{-1}$ $y^m x^n = x^n (q^n y + [n]\alpha)^m$		
$(q,0,\beta,0)$	$yx = qxy + \beta y$	$YX = qXY$
Isomorphism: $X \to x - \beta(1-q)^{-1}\ ,\quad Y \to y$ $y^m x^n = (q^m x + [m]\beta)^n y^m$		
$(q,\alpha,\beta,0)$	$yx = qxy + \alpha x + \beta y$	$YX = qXY$
Isomorphism: $X \to x - \beta(1-q)^{-1}\ ,\quad Y \to y - \alpha(1-q)^{-1}$ $y^m x = x(qy+\alpha)^m + \beta \sum_{k=1}^{m} y^k \alpha^{m-k} \sum_{i=0}^{k-1} \binom{m-k+i}{i} q^i,$ $y^m x^n =$ not known yet		
$(q,0,0,\gamma)$	$yx = qxy + \gamma$	$YX = qXY + 1$
Isomorphism: $X \to x\ ,\quad Y \to \gamma^{-1} y$ $y^m x^n = \sum_{k=0}^{n} \begin{bmatrix} m \\ k \end{bmatrix} [n]^{\underline{k}} q^{(n-k)(m-k)} \gamma^k x^{n-k} y^{m-k}.$		
$(q,\alpha,0,\gamma)$	$yx = qxy + \alpha x + \gamma$	$YX = qXY + 1$
Isomorphism: $X \to \gamma^{-1} x\ ,\quad Y \to y - \alpha(1-q)^{-1}$ $y^m x^n = \sum_{k=0}^{n} \sum_{j=0}^{m-k} \begin{bmatrix} n \\ k \end{bmatrix} \gamma^k \left(\frac{\alpha}{1-q}\right)^{m-j-k} c_{j,k,m,n} x^{n-k} y^j,$ where $c_{j,k,m,n} = \sum_{i=0}^{m-j-k} (-1)^i \binom{m}{i+j+k} \binom{i+j}{j} [i+j+k]^{\underline{k}} q^{(i+j)(n-k)}$		
$(q,0,\beta,\gamma)$	$yx = qxy + \beta y + \gamma$	$YX = qXY + 1$
Isomorphism: $X \to x - \beta(1-q)^{-1}\ ,\quad Y \to \gamma^{-1} y$ $y^m x^n = \sum_{k=0}^{n} \sum_{j=0}^{n-k} \begin{bmatrix} m \\ k \end{bmatrix} \gamma^k \left(\frac{\beta}{1-q}\right)^{n-j-k} c_{j,k,m,n} x^j y^{m-k},$ where $c_{j,k,m,n} = \sum_{i=0}^{n-j-k} (-1)^i \binom{n}{i+j+k} \binom{i+j}{j} [i+j+k]^{\underline{k}} q^{(i+j)(m-k)}$		
(q,α,β,γ)	$yx = qxy + \alpha x + \beta y + \gamma$	$YX = qXY + 1$
Isomorphism: $X \to x - \beta(1-q)^{-1}\ ,\quad Y \to ((1-q)y - \alpha)(\gamma(1-q) + \alpha\beta)^{-1}$ $y^m x^n =$ not known yet		

2.1 Operator Algebras and Model Algebras

Fix a constant $c \in \mathbb{K}^*$. Then the *c-shift operator* acts as $s_c(f(x)) = f(x-c)$. The corresponding *c-shift algebra* is $\mathbb{K}\langle x, s_c \mid s_c \cdot x = x \cdot s_c - cs_c\rangle$. For $c = 1$ one recovers discrete shift operator. If $c < 0$ (resp. $c > 0$), s_c is called an *advance operator* (resp. a *time-delay operator*) in both discrete and continuous settings. The corresponding algebras are of the type $(1, 0, \beta = -c, 0)$ and thus they are isomorphic to $\mathbb{K}\langle X, Y \mid YX = XY + Y\rangle$, the model shift algebra.

Let $c = (c_1, c_2)$ for $c_i \in \mathbb{K}^*$. The *c-difference operator* acts as $\Delta_c(f(x)) = \frac{f(x+c_1)-f(x)}{c_2}$. The corresponding *c-difference algebra* is

$$\mathbb{K}\langle x, \Delta_c \mid \Delta_c \cdot x = x \cdot \Delta_c + c_1\Delta + \tfrac{c_1}{c_2}\rangle.$$

For $c = (1, 1)$ one recovers discrete difference operator; for $c = (\triangle x, \triangle x)$ the first-order divided difference operator. The corresponding algebras are of the type $(1, 0, \beta = c_1, \gamma = c_1 c_2^{-1})$ and hence they are isomorphic to $\mathbb{K}\langle X, Y \mid YX = XY + Y\rangle$, the model shift algebra.

Following Chyzak and Salvy [2], the *q-dilation* and *q-shift* operators give rise to the same operator algebra, the q-commutative model algebra $\mathbb{K}_q[x, y]$. Both continuous and discrete *q-difference* operators [2] give rise to the algebra $\mathbb{K}(q)\langle x, y \mid yx = qxy + (q-1)x\rangle$ of the type $(q, \alpha = q-1, 0, 0)$. Hence it is isomorphic to the q-commutative model algebra $\mathbb{K}_q[x, y]$.

Let $c = (c_1, c_2)$ for $c_i \in \mathbb{K}(q)^*$ with $q^{c_i} \neq 0$. The *c-q-differential operator* acts as $\Delta_c^{(q)}(f(x)) = \frac{f(q^{c_1}x)-f(x)}{(q^{c_2}-1)x}$. The corresponding *c-q-differential algebra* is

$$\mathbb{K}(q)\langle x, \Delta_c^{(q)} \mid \Delta_c \cdot x = q^{c_1}x \cdot \Delta_c + (q^{c_1}-1)\cdot(q^{c_2}-1)^{-1}\rangle.$$

For $c = (1, 1)$ one recovers the *q-differential operator* $D_q(f(x)) = \frac{f(qx)-f(x)}{qx-x}$. Otherwise, we use Table 2 and by sending $x \to X, \Delta_c^{(q)} \to Y := (q^{c_2}-1)(q^{c_1}-1)^{-1}\Delta_c^{(q)}$ we obtain the isomorphic algebra $\mathbb{K}(q)\langle X, Y \mid YX = q^{c_1}XY + 1\rangle$. Let $\tilde{q} = q^{c_1}$, then the subalgebra $\mathbb{K}(\tilde{q})\langle X, Y \mid YX = \tilde{q}XY + 1\rangle$ of the previous algebra is the first $\tilde{q}$-Weyl model algebra.

Consider the differentiation $y = \frac{d}{dt}$ and the operator $x(f(t)) := e^{\lambda t} \cdot f(t)$ for $\lambda \in \mathbb{K}^*$. Then the algebra, generated by x, y has the relation $yx = xy + \lambda x$ and it is isomorphic to the model shift algebra.

Of course, there are operators obeying relations, which are not affine. Consider the *integration* operator $I(f(x)) := \int_0^x f(t)dt$. Its relation with x reads as $Ix = xI - I^2$. Similarly, let $x = t^{-1}$ and $y = \frac{d}{dt}$. Then the relation is $yx = xy - x^2$. Both algebras can be realized as G-algebras. It is interesting to study model algebras for non-affine relations.

Remark 1. Note, that isomorphy of q-shift and q-commutative algebras does not have an analogue in the classical situation, since for $q = 1$ the model shift algebra is not isomorphic to the model commutative algebra. Thus the following question arises: is there a quantum algebra (clearly, with non-affine relation), which becomes shift model algebra in the limit $q \to 1$?

2.2 Binomial Theorems

Notation: in a noncommutative algebra A, for two elements $a, b \in A \setminus \mathbb{K}$, we define $[a+b]^n := \sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i}$. Respectively, we define $[a+b]_q^n := \sum_{i=0}^{n} \begin{bmatrix} n \\ i \end{bmatrix}_q a^i b^{n-i}$. Then, if x, y commute, one expresses the binomial theorem as $(x+y)^n = [x+y]^n$. Respectively, if $yx = qxy$, we obtain $(x+y)^n = [x+y]_q^n$.

Using the formulas obtained above, we can provide formulas of binomial type, which are important in applications. Among the variety of possible presentations in such formulas we aim at those, which express $(x+y)^n$ in terms of standard monomials $x^i y^j$.

In the free associative algebra $\mathbb{K}\langle a, b\rangle$, we can write $(a+b)^n = \sum_{w \in \langle a,b\rangle_n} w$, that is w run through all words of length n in the free monoid $\langle a, b\rangle$. One defines a *misordering index* [1] of w to be the number of operations, each of them exchanges two neighbour non-equal letters, needed to move all a's to the left (thus finishing when a standard monomial has been achieved), starting from the last letter in w. For example, the misordering index of a standard monomial is 0, while the misordering index of $bbbab$ is 3, since the sequence of exchange operations is $bbbab, bbabb, babbb, abbbb$. We say also that $bbbab$ *converges* to $abbbb$ here. It is known, that in any algebra $A(q, \alpha, \beta, \gamma)$ the leading monomial of a polynomial $y^m \cdot x^n$ is $x^n y^m$. Hence, the coefficients of a standard monomial $x^a y^b$ of degree $a+b$ will appear from the multiplication, applied on every word, which converges to $x^a y^b$. And closed formulas for multiplication allow to perform this task symbolically.

Lemma 3. *Let $A = A_1$ be the first Weyl algebra, where $\partial x = x\partial + 1$ holds. Then the following binomial theorem takes place:*

$$(x+d)^n - [x+d]^n = \sum_{k=0}^{n-2} \sum_{j=0}^{n-k-2} \binom{n}{j} \binom{n-j}{k} g(n-j-k) x^k d^j$$

where $g(n) := (n-1)!!$, if n is even and 0 otherwise. Alternatively we can write

$$(x+d)^n - [x+d]^n = \sum_{0 \le k \le n-2} \sum_{\substack{0 \le j \le n-k-2 \\ n-j-k \text{ even}}} \binom{n}{j} \binom{n-j}{k} (n-j-k-1)!! x^k d^j$$

$$= \sum_{0 \le k \le n-2} \sum_{\substack{0 \le j \le n-k-2 \\ n-j-k \text{ even}}} \frac{n!}{j! k! (\frac{n-j-k}{2})!} \left(\frac{1}{2}\right)^{\frac{n-j-k}{2}} x^k d^j.$$

Lemma 4. *For the shift algebra S_1, where $xs = sx + s$ holds, we obtain the following binomial theorem:*

$$(x+s)^n = [x+s]^n + \sum_{k=0}^{n-1} \sum_{j=0}^{n-k-1} \binom{n}{k} S(n-k, j) x^k s^j$$

where $S(n, k)$ denote the Stirling numbers of the second kind.

We omit the technical proofs for both Lemmas. They can be done by induction, using the multiplication formulas.

3 Application in Computer Algebra Implementation

As described in [5], a general multiplication in a non-commutative G-algebra boils down to the multiplication of $y^m \cdot x^n$ for a couple of variables x, y such that $x^a y^b$ is a standard word. In general, the polynomial $y^m \cdot x^n$ involves other variables as well, but the case, when x, y generate a subalgebra of the type $A(q, \alpha, \beta, \gamma)$, appears often enough. Suppose from now on we are in such situation.

In [5] it has been proposed to address each pair of non-commuting and non-q-commuting variables separately. To each such pair a matrix M is assigned, such that $M_{ij} = y^i \cdot x^j$ is a polynomial, written in terms of standard monomials. There is a general multiplication algorithm, which uses matrix entries of lower degree in order to compute the higher degrees on demand.

There are several different strategies on the usage of the formulas for enhancing the polynomial multiplication. Of course, this problem barely has an analogue in the commutative case. Initialization of non-commutative relation between y and x saves the relation $yx = q \cdot xy + \alpha x + \beta y + \gamma$ as a part of data structure on the algebra, where the computations take place.

1. Faster computation, considerable memory usage: As proposed in [5], the results of all required multiplications $y^i \cdot x^j$ and the intermediate multiplications in lower degree will be saved. Due to the same principles applied for the search of previously computed elements of lower degree, the multiplication matrix will be filled with many elements. On the other hand, the intermediate elements will be reused intensively and this leads to fast arithmetics in the algebra.

2. Saving memory, slower computation: All required multiplications $y^i \cdot x^j$ will be done according to the formulas, the results will not be saved for the future use. Thus this way uses the least amount of memory, but can take much longer, especially if many multiplications are requested repeatedly.

3. Mixing 1 and 2 and using formulas: Computing, by utilizing formulas, the requested elements and storing them into the multiplication matrix eliminates the need to compute and store intermediate elements from the approach 1. Storing the demanded elements increases the chances for the future reuse of matrix entries. Still, there are more possibilities to develop strategies by mixing both approaches and working with multiplication matrices dynamically, like keeping (e. g. by periodic cleaning) higher degree part of the matrix sparse while being as dense as possible in the lower degree part. But the question, how to determine the value, which distinguishes high degree from low degree, is open. At last, but not at least, we have experimented with counting the requests to each matrix entry, thus having a metric for the usability of every single entry. This is useful while following the strategy, which uses periodic cleaning of multiplication matrices.

Experiments. Let us do experiments with the most general case: $A = A(q, \alpha, \beta, \gamma)$, where q, α, β, γ are transcendental over the base field. As described above, in general the product $y^a \cdot x^b$ is computed (inductively) either as $y \cdot (y^{a-1} \cdot x^b)$ or as $(y^a \cdot x^{b-1}) \cdot x$. Let us consider the products $y^i \cdot x$ and $y \cdot x^i$. The determination of the computational method for these products can be made during run-time by analyzing given q, α, β, γ. Both $y^i \cdot x$ and $y \cdot x^i$ are of the same length (with $2(i+1)$ terms), with the same leading term and of the same internal byte-size. Counting the byte-size of both expressions for $i = 1..10, 15, 20$ we obtain $8, 21, 40, 65, 96, 133, 176, 225, 280, 341, 736, 1281$. Indeed this sequence coincides with octagonal numbers[1] shifted by 1. Hence the byte-size $s(i)$ of $y^i \cdot x$ is $3i^2 + 5i + 2$.

Further on we look for some computation-specific patterns, for instance, during the computation of a left Gröbner basis. We use the implementation of *slim Gröbner basis* algorithm in Singular for highly resource-demanding and tasks like the computation of Bernstein-Sato polynomials with two different algorithms. With the latter algorithms one computes in the tensor product of model algebras (Weyl and shift algebras), what suggests using formulas. By cashing below we mean the use of the multiplication table for saving once computed elements.

We experiment with the following strategies: 1. Using formulas and caching 2. Using formulas without caching the results 3. Caching the results, obtained without formulas. The timings and maximal memory usage are collected in the Table 3. Since we are interested in the efficiency of the caching, we count the requests to compute every needed elementary product $y^a \cdot x^b$ as above in a separate computation. In the process of computation of Bernstein-Sato polynomial of various Reiffen curves $f(x,y) \in \mathbb{K}[x,y]$, there is a complicated computation in the tensor product of two Weyl algebras and $\mathbb{K}[s]$. We count requests to compute $y^m x^n = x^n y^m$ in Table 4 and the number of requests to compute $\partial_x^m x^n$ in the first Weyl algebra, where $[\partial_x, x] = 1$ holds, in Table 5. All the data is available online from `http://www.mathematik.uni-kl.de/~motsak/ncSAtests`. Timings and memory are given in seconds resp. in Kb. The tests were run on a PC running 64 bit Arch Linux 2.6.38, having 16 GB RAM and Intel Core i7 CPU 860 at 2.80GHz (4 Cores/8 Threads).

We gain some speedup by using both caching and formulas. It can be enhanced by optimizing the way of caching, especially for algebras with many variables.

4 Centers and Ring-Theoretic Properties of Model Algebras

By using the formulas, we compute explicitly the centers of non-commutative model algebras, depending on the ground field $\mathbb{K}$. Recall, that for some $f \in A$ one defines the centralizer subalgebra $C(f) = \{a \in A \mid fa = af\} \supseteq \mathbb{K}[f]$.

Proposition 1. *For the algebras of Lie type one has*

- *If* $\operatorname{char} \mathbb{K} = 0$, $Z(A_1) = \mathbb{K}$ *and* $Z(S_1) = \mathbb{K}$.

[1] `http://oeis.org/A000567`

Table 3. Time and Memory Comparison for Different Multiplication Strategies

Name	Cache + Formulas		Formulas only		Cache only	
	time	memory	time	memory	time	memory
reiffen11-mod	8.04	9.912	8.14	11.999	7.97	9.912
reiffen45-3-ann	8525.40	518.262	8517.73	523.502	8547.37	518.261
reiffen45-6-ann	158.25	46.509	160.20	46.506	160.54	46.509
reiffen57-mod	36.16	11.422	39.28	11.415	36.41	11.422
reiffen59-mod	80.03	15.759	87.90	15.746	81.03	15.759
reiffen67-mod	216.81	45.369	261.30	45.361	222.15	45.369
reiffen68-mod	117.58	142.254	141.97	138.494	118.31	142.254
reiffen76-mod	389.54	99.026	630.48	99.026	389.42	99.026
reiffen86-mod	298.83	69.610	453.93	69.610	297.81	69.610

Table 4. Number of Requests for $y^m x^n = x^n y^m$

	n	1	2	3	4	5	6	7	8	9	10
m											
1	23711	18629	17628	14796	8368	2899	2444	1315	296	186	32
2	8264	4952	4806	4947	2952	728	715	549	47	26	0
3	4900	3002	3233	3202	1577	286	277	237	18	9	0
4	2084	1230	1268	1189	585	104	82	60	6	3	0
5	215	155	118	131	127	59	48	37	2	1	0
6	62	45	30	30	26	6	3	0	0	0	0
7	19	14	9	9	8	2	1	0	0	0	0
8	3	2	1	1	1	0	0	0	0	0	0

Table 5. Number of Requests for $\partial_x^m x^n = x^n \partial_x^m + \ldots$

	n	1	2	3	4	5	6	7	8	9	10	11
m												
1	27345	22324	21914	20484	14636	5702	4076	3104	1515	1005	563	164
2	12627	10267	9799	9219	6910	2592	1888	1523	718	455	246	72
3	4271	3319	2904	2895	2544	942	763	691	300	181	90	26
4	1149	872	604	659	780	277	273	275	105	58	24	6
5	247	203	50	79	224	54	65	83	26	11	0	0

- *If* $\operatorname{char}\mathbb{K} = p$*,* $Z(A_1) = \mathbb{K}[x^p, \partial^p]$ *and* $Z(S_1) = \mathbb{K}[x^p - x, s^p]$.

For the quantum algebras one has

- *If q is not a root of unity,* $Z(\mathbb{K}_q[x, y]) = \mathbb{K}(q)$ *and* $Z(A_1^{(q)}) = \mathbb{K}(q)$.
- *If q is a primitive root of unity of order p over $\mathbb{K}$,* $Z(\mathbb{K}_q[x, y]) = \mathbb{K}(q)[x^p, y^p]$ *and* $Z(A_1^{(q)}) = \mathbb{K}(q)[x^p, \partial^p]$.

Proof. Since all the proofs are similar, let us consider the q-Weyl algebra $A_1^{(q)}$. We compute the center as $Z(A) = C(x) \cap C(\partial)$. Since A is a $\mathbb{Z}$-graded algebra (e. g. with $\deg x = -1$, $\deg d = 1$), $C(x), C(\partial)$ and $Z(A)$ are $\mathbb{Z}$-graded subalgebras. The 0-th graded part of A is $\mathbb{K}(q)[x\partial]$. For $k \in \mathbb{Z}_+$, the k-th graded part of A is $A_k = \mathbb{K}(q)[x\partial]\partial^k$ and $A_{-k} = \mathbb{K}(q)[x\partial]x^k$. By Thm. 1, we see that $\partial^m x = q^m x\partial^m + [m]_q\partial^{m-1} \in A_{m-1}$. Thus for $f = \sum_\alpha c_\alpha(x)\partial^\alpha$ one has $0 = fx - xf = \sum_\alpha c_\alpha(x)(\partial^\alpha x - x\partial^\alpha)$. Note, that $\partial^\alpha x - x\partial^\alpha = (q^\alpha - 1)x\partial^\alpha + [\alpha]_q\partial^{\alpha-1}$ is graded. So, for all α $(q^\alpha - 1)x\partial^\alpha + [\alpha]_q\partial^{\alpha-1} = 0$, that is $q^\alpha = 1$. Hence $q^p = 1$ implies $C(x) = \mathbb{K}(q)[x, \partial^p]$, $C(\partial) = \mathbb{K}(q)[x^p, \partial]$ and thus $Z(A) = \mathbb{K}(q)[x^p, \partial^p]$.

It is known, that over any field $A(q, \alpha, \beta, \gamma)$ is a G-algebra (or a PBW algebra) [5,1]. Thus it is a Noetherian domain of Gel'fand-Kirillov dimension 2, which is Cohen-Macaulay and Auslander-regular [1]. However, the global homological dimension is between 1 and 2.

Proposition 2. *$\operatorname{gl.dim} A(q, \alpha, \beta, \gamma) = 1$ if and only if $\operatorname{char} \mathbb{K} = 0$ and $A(q, \alpha, \beta, \gamma)$ is isomorphic to the Weyl algebra.*

Proof. Let $A = A(q, \alpha, \beta, \gamma)$. Because of Cohen-Macaulay property, $\operatorname{gl.dim} A = 2$ if and only if there exist a module M of finite dimension over $\mathbb{K}(q)$. We look for $M = A/L$ for an ideal $L \subset A$. Over $\mathbb{K}[x, y]$ and $\mathbb{K}_q[x, y]$ all 1-dimensional modules are described by ideals $\langle x - a, y - b\rangle$ for $a, b \in \mathbb{K}(q)$. In the shift algebra there are ideals $\langle x - a, s\rangle$ for $a \in \mathbb{K}$ while in the q-Weyl algebra these ideals are $\langle x - a, y - ((1 - q)a)^{-1}\rangle$ for $a \in \mathbb{K}(q)^*$.
Consider the case when A is the Weyl algebra. If $\operatorname{char} \mathbb{K} = p > 0$, from Prop. 1 follows, that $I_p = \langle x^p, \partial^p\rangle$ is a proper two-sided ideal and A/I_p is finite dimensional, thus $\operatorname{gl.dim} A = 2$. If $\operatorname{char} \mathbb{K} = 0$, it is known that A has no finite-dimensional representations, hence $\operatorname{gl.dim} A = 1$.

Lemma 5. *For any field $\mathbb{K}$, there are no nonzero $\mathbb{K}$-algebra homomorphisms from $A_1(\mathbb{K})$ to $\mathbb{K}_q[x, y]$ or to $\mathbb{K}[x, y]$.*

Proof. Assume there is a homomorphism of $\mathbb{K}$-algebras $\phi : A_1(\mathbb{K}) \to \mathbb{K}_q[x, y]$. Thus there exists $X = \phi(x), D = \phi(\partial) \in \mathbb{K}_q[x, y]$, such that $DX - XD = 1$. Write $D = \sum_\alpha c_\alpha x^{\alpha_1} y^{\alpha_2}$ for $c_\alpha \in \mathbb{K}$ and $\mathbb{N}_0^2 \ni \alpha = (\alpha_1, \alpha_2)$. Analogously $X = \sum_\beta d_\beta x^{\beta_1} y^{\beta_2}$. Then in $\mathbb{K}_q[x, y]$ one has $DX - XD = \sum_{\alpha,\beta} c_\alpha d_\beta (q^{\beta_1\alpha_2} - q^{\beta_2\alpha_1}) x^{\alpha_1+\beta_1} y^{\alpha_2+\beta_2}$ and the coefficient by $1 = x^0y^0$ vanishes. In the limit $q \to 1$, that is in $\mathbb{K}[x, y]$ we obtain $DX - XD = 0$. Hence the only homomorphism from $A_1(\mathbb{K})$ to $\mathbb{K}[x, y]$ or to $\mathbb{K}_q[x, y]$ is 0.

Proposition 3. *Five model algebras are pairwise non-isomorphic over any field.*

Proof. Let $\operatorname{char} \mathbb{K} = 0$. From Prop. 1 we see that $A_1(\mathbb{K}), S_1(\mathbb{K}), A_1^{(q)}(\mathbb{K}) \not\cong \mathbb{K}[x, y]$. By Prop. 2 and Lemma 5 we conclude $S_1(\mathbb{K}), \mathbb{K}_q[x, y], A_1^{(q)}(\mathbb{K}) \not\cong A_1(\mathbb{K})$. For any field $\mathbb{K}$, $A_1^{(q)}(\mathbb{K}) \not\cong \mathbb{K}_q[x, y]$: let U, V be affine subspaces of $\mathbb{K}^2$ of all 1-dimensional representations of both algebras in $\mathbb{K}$. Then U, V are zero sets of corresponding ideals $I = \langle(1 - q)ab + 1\rangle$ and $J = \langle(1 - q)cd\rangle = \langle c\rangle \cap \langle d\rangle$, what implies $\mathbb{K}[U] \ncong \mathbb{K}[V]$. Since the variety $W \subset \mathbb{K}^2$ of 1-dim. representations of S_1 is $W = \{(a,b) \mid ba = ab + b\} = \{(a,0) \mid a \in \mathbb{K}\}$ cannot be in bijection with either U or V, S_1 is not isomorphic to $A_1^{(q)}(\mathbb{K})$ or $\mathbb{K}_q[x,y]$. Also $\mathbb{K}[x,y]$ with $\mathbb{K}^2$ as the variety of 1-dim. representations is not isomorphic to other model algebras for any $\mathbb{K}$. Now, let $\operatorname{char} \mathbb{K} = p$. Then $A_1(\mathbb{K})$ has finite dimensional representations since $m = \operatorname{tr}(1_{m\times m}) = \operatorname{tr}(DX - XD) = 0$ for a $m \times m$ representation X, D of $A_1(\mathbb{K})$. Hence $p \mid m$ and the smallest irreducible representation is in dimension p. Thus $A_1(\mathbb{K})$ cannot be isomorphic to other model algebras. The remaining cases can be proved analogously.

Future work. includes the study of Ore localizations of model algebras, for which no analog of the "five models" theorem is not known yet. Linear (anti-) endomorphisms of model algebras are of interest as well.

Acknowledgments. The authors are grateful to H. Schönemann and O. Yena for discussions on the subject. We would like to thank anonymous referees for valuable suggestions. The second author was supported by the Austrian Science Fund (FWF): P20162-N18. The first and third authors are grateful to the SCIEnce project (Transnational access) at RISC for supporting their visits to RISC and the usage of computational infrastructure at RISC.

References

1. Bueso, J., Gómez-Torrecillas, J., Verschoren, A.: Algorithmic methods in non-commutative algebra. Applications to quantum groups. Kluwer Acad. Publ., Dordrecht (2003)
2. Chyzak, F., Salvy, B.: Non–commutative elimination in Ore algebras proves multivariate identities. J. Symbolic Computation 26(2), 187–227 (1998)
3. Dixmier, J.: Enveloping Algebras. AMS, Providence (1996)
4. Koutschan, C.: HolonomicFunctions (User's Guide). Technical Report 10-01, RISC Report Series, University of Linz, Austria (2010)
5. Levandovskyy, V., Schönemann, H.: Plural — a computer algebra system for non-commutative polynomial algebras. In: Proc. ISSAC, pp. 176–183. ACM Press, New York (2003)
6. McConnell, J., Robson, J.: Noncommutative Noetherian rings. AMS, Providence (2001)

Acceleration of the Inversion of Triangular Toeplitz Matrices and Polynomial Division*

Brian J. Murphy

Department of Mathematics and Computer Science
Lehman College of the City University of New York
Bronx, NY 10468 USA
brian.murphy@lehman.cuny.edu

Abstract. Computing the reciprocal of a polynomial in z modulo a power z^n is well known to be closely linked to polynomial division and equivalent to the inversion of an $n \times n$ triangular Toeplitz matrix. The degree k of the polynomial is precisely the bandwidth of the matrix, and so the matrix is banded iff $k \ll n$. We employ the above equivalence and some elementary but novel and nontrivial techniques to obtain minor yet noticeable acceleration of the solution of the cited fundamental computational problems.

Keywords: Reciprocal of a polynomial modulo a power, Polynomial division, Triangular Toeplitz matrix inversion, Banded triangular Toeplitz matrices.

1 Introduction

1.1 Our Subjects

Our goal is to accelerate direct computation of the inversion of a triangular Toeplitz matrix as well as a banded triangular Toeplitz matrix. Triangular Toeplitz matrices play a central role in the displacement representation of general Toeplitz matrices, which is fundamental in the study of structured matrices (see [1], [2], [3]). Banded triangular Toeplitz matrices have been employed for preconditioning of computations with general Toeplitz matrices (see [4], [5]).

Perhaps even more important are the applications of our work to polynomial division, which is fundamental for polynomial computations and is closely related to the computation of the reciprocal of a polynomial $u(z)$ modulo z^n. The latter task is equivalent to the inversion of an $n \times n$ triangular Toeplitz matrix U where the degree k of the polynomial is precisely the bandwidth of the matrix (see [6], [2], [3]). Namely, every solution algorithm for one of these two computational tasks automatically computes the solution to the other one as well. Furthermore the paper [6] selects three algorithms for reciprocals and three for inversion, including the four most popular algorithms. Each of the six algorithms is well known in its own right and has a natural derivation, distinct from the five others.

* Supported by PSC CUNY Awards 609 62400–0040 and 393 6327000–41.

The paper reveals, however, that each of the three polynomial versions computes exactly the same auxiliary and output values as its matrix counterpart.

In spite of such an isomorphism, it can be useful to devise solution algorithms by employing both polynomial and matrix representations for this problem. In particular this turned out to be useful for us. Extensive coverage of algorithm design employing links between various computations with polynomial and structured matrices can be found in [7], [2], and [3].

1.2 Previous Work

The inverse of a nonsingular triangular Toeplitz matrix maintains this structure and can be computed by Gaussian elimination in $O(n^2)$ arithmetic operations (ops). For the reciprocal modulo z^n this algorithm essentially amounts to the classical long polynomial division [6, Section 4]. The record fell to $O(M(n))$ ops in [8] based on Newton's iteration, where $M(n)$ denotes the cost, in ops, of polynomial multiplication modulo z^n. Over the fields supporting Fast Fourier Transform (FFT), this bound turns into $O(n \lg n)$. The transition to a matrix version simplifies the derivation a bit (see [6, Section 4]), and a number of nontrivial works yielded parallel acceleration (see [9], [10], [11], [12], and the references therein), but the record sequential bound of $O(M(n))$ remains unbeaten since 1972. The Newton–Sieveking's algorithm appeared virtually with no change in various books and surveys (see, e.g., [13], [14], [2], [15]). Some minor improvement in the implementation was proposed in [16], for which the cost of computation is $(1.\bar{6} + o(1))M(n)$ ops. Related work was done in [17], [18], [19], and [20]. Additional progress was made in [21] taking advantage of wrapping properties of convolution to devise an algorithm requiring $(1.5 + o(1))M(n)$ ops and recently in [22] a further improvement appears requiring $(1.\bar{4} + o(1))M(n)$ ops. Each improvement served to lower the constant of proportionality, but the $O(M(n))$ ops bound stands.

1.3 Our Progress

The complexity of the Newton–Sieveking's algorithm in its implementation in [16] is dominated by the cost of performing five FFTs at $2n^{th}$ roots of unity at each recursive stage. (We assume here and hereafter that $n = 2^i$ is an integer power of two.) As with the others, our algorithm does not surpass the asymptotic record bound from 1972, but building on the work in [16] accelerates the process by roughly a factor of 10/9 by decreasing the number of FFTs at each recursive stage to three at $4n^{th}$ roots of unity where each FFT is short-circuited [23] to compute only $3n$ of $4n$ entries. This yields an algorithm requiring $(1.5 + o(1))M(n)$ ops which is in the same class as the algorithm from [21], but seemingly bested by the algorithm from [22]. We present this algorithm as an alternative to these others for several reasons. With a bit of speculation we note that implementation details may favor one over another under different circumstances. On massively parallel hardware the algorithm herein provides a more step efficient approach while work efficiency is not significantly if at all impaired.

Note that polynomial multiplication is not consistent in terms of the number of ops required for its performance from one algorithm to another, so that the $M(n)$ component hides some small variations. The new algorithm is not difficult to implement and finally, because it serves to seed another algorithm presented herein that addresses the banded case.

In the case of banded input we yield additional acceleration. However, this acceleration occurs incrementally as the stages unfold via four successive methods that each serve to initialize the one that follows it. The algorithm begins in the same manner as in the non-banded case so that the factor remains 10/9 until the bandwidth is exhausted. For one stage thereafter it is 10/4.5 and then settles at 10/3. That is, until a cross-over point is reached and the factor can again be lowered. The final factor is $10/(2+2^{-i})$ where i is the number of iterations applied with the previous factor.

The value of this limited progress is due foremost to the fundamental importance of the problems, however our technique of algorithm design can be of independent interest, as it exploits the duality between structured matrices and polynomial computations. The method is mostly elementary but nontrivial, particularly in the case of our algorithm for banded input.

1.4 Organization of Our Paper

We provide some definitions and basic facts and also link computation of polynomial reciprocal to triangular Toeplitz inversion in the next section. We recall the basic steps of the divide and conquer matrix version of the Newton–Sieveking's algorithm from [16] in section 3. We present our algorithm for triangular Toeplitz inversion and then yield additional acceleration in the case of banded triangular Toeplitz input in section 4. We speculate on some implementation issues in Section 5.

2 Preliminaries

2.1 Definitions and Basic Facts

– I is the $n \times n$ identity matrix and $\mathbf{e}_1$ is its 1^{st} column, where n is understood by context.
– $\mathbf{0} = (0)_{i=1}^{n}$ is the zero vector where n is understood by context.
– $\mathrm{wrap}_n(\mathbf{v}) = (w_i)_{i=1}^{n}$ where m-vector $\mathbf{v} = (v_i)_{i=1}^{m}$, $v_i = 0$ for $i > m$ and $w_i = \sum_{j=0}^{\lfloor m/n \rfloor} v_{i+jn}$.
– $F_n(\mathbf{v})$ denotes the n-vector obtained by applying the discrete Fourier transform (DFT) to n-vector $\mathrm{wrap}_n(\mathbf{v})$ where $\mathbf{v}$ is a vector of arbitrary size. $W_n(\mathbf{u})$ denotes the n-vector obtained by applying the inverse DFT to n-vector $\mathbf{u}$. $F_n(\mathbf{v})$ and $W_n(\mathbf{u})$ can be computed in $O(n \lg n)$ ops via FFT and inverse FFT (IFFT) respectively. Let $m(n)$ be the number of ops required to compute a DFT of order n via FFT. $F_n(\mathbf{e}_1) = (1)_{i=1}^{n}$ is available without computation.

- $\mathbf{u} \star \mathbf{v} = (u_i v_i)_{i=0}^{n-1}$ is the element-wise product of two n-vectors $\mathbf{u} = (u_i)_{i=0}^{n-1}$ and $\mathbf{v} = (v_i)_{i=0}^{n-1}$. Let $\mathbf{v}^2 = \mathbf{v} \star \mathbf{v}$.
- $\mathbf{u} \otimes \mathbf{v} = (c_i)_{i=0}^{n_1+n_2-2}$ is the convolution of n_1-vector $\mathbf{u} = (u_i)_{i=0}^{n_1-1}$ and n_2-vector $\mathbf{v} = (v_i)_{i=0}^{n_2-1}$, where $c_i = \sum_{j=0}^{i} u_j v_{i-j}$, for $0 \leq i < n_1+n_2-1$, $u_i = 0$, for $i \geq n_1$ and $v_i = 0$, for $i \geq n_2$. It is well known [13] that $\mathbf{u} \otimes \mathbf{v} = \mathbf{v} \otimes \mathbf{u}$ and that $W_n(F_n(\mathbf{u}) * F_n(\mathbf{v})) = \mathrm{wrap}_n(\mathbf{u} \otimes \mathbf{v}) = \mathrm{wrap}_n(\mathbf{u} \otimes \mathrm{wrap}_n(\mathbf{v}))$.
- (L, M) forms the 1×2 block matrix with the blocks L and M.
- $T = (t_{ij})_{i,j=1}^{m,n}$ is a Toeplitz matrix if whenever defined entries $t_{i+1,j+1} = t_{ij}$.
- $Z = (z_{ij})_{i,j=1}^{n}$ denotes the $n \times n$ downshift matrix if $z_{ij} = 1$ where $i = j+1$ and $z_{ij} = 0$ otherwise . $Z^0 = I$. $Z(\mathbf{v}) = \sum_{i=0}^{n-1} v_i Z^i$ and $Z^{\mathrm{T}}(\mathbf{v}) = \sum_{i=0}^{n-1} v_i (Z^{\mathrm{T}})^i$ are $n \times n$ lower and upper triangular Toeplitz matrices respectively defined by the n-vector $\mathbf{v} = (v_i)_{i=0}^{n-1}$, which is the first column of $Z(\mathbf{v})$ and the first row of $Z^{\mathrm{T}}(\mathbf{v})$. $Z^{-1}(\mathbf{v})$ is a lower triangular Toeplitz matrix defined by its first column $\bar{\mathbf{v}} = Z^{-1}(\mathbf{v})\mathbf{e}_0$. Note that $(Z^{-1}(\mathbf{v}))^{\mathrm{T}} = (Z^{\mathrm{T}}(\mathbf{v}))^{-1}$.
- $A = Z(\mathbf{a})$, $A^{-1} = Z(\bar{\mathbf{a}})$, $B = Z(\mathbf{b})$ and $B^{-1} = Z(\bar{\mathbf{b}})$ where $\mathbf{a} = (a_i)_{i=1}^{n}$, $\bar{\mathbf{a}} = (\bar{a}_i)_{i=1}^{n}$, $\mathbf{b} = (b_i)_{i=1}^{m}$, and $\bar{\mathbf{b}} = (\bar{b}_i)_{i=1}^{m}$, if $a_i = b_i$ for all $i \leq \min(n, m)$ then $\bar{a}_i = \bar{b}_i$ for all $i \leq \min(n, m)$. Therefore, the choice to limit $n = 2^i$ to a positive integer power of two leads to no loss of generality.

The algorithms presented herein explicitly invert lower triangular Toeplitz matrices, but they apply equally to the upper triangular case since $(Z^{-1}(\mathbf{v}))^{\mathrm{T}} = (Z^{\mathrm{T}}(\mathbf{v}))^{-1}$.

2.2 Polynomial Reciprocal and Triangular Toeplitz Inversion

Recall that $Z^n = 0$ and the matrix polynomials $Z(\mathbf{u}) = \sum_{i=0}^{n-1} u_i Z^i$ form the algebra $\mathcal{A}_n$ of $n \times n$ lower triangular Toeplitz matrices isomorphic to the algebra of polynomials $u(z) = \sum_{i=0}^{n-1} u_i z^i$ modulo z^n with coefficient vector $\mathbf{u}$. Such a vector $\mathbf{u}$ is the first column of the matrix $Z(\mathbf{u})$, which defines both this matrix and the polynomial $u(z)$. The trailing entries of the vectors are the leading coefficients of the polynomial. If the polynomial has degree k, then all but the first $k+1$ of them vanish and the associated lower triangular Toeplitz matrix is banded with the bandwidth k.

Clearly, a matrix $Z(\mathbf{u})$ is nonsingular if and only if $u_0 \neq 0$. If $Z(\mathbf{u})Z(\mathbf{v}) = I$ (that is if $Z(\mathbf{v}) = Z(\mathbf{u})^{-1}$), then $v(z)u(z) = 1 \mod z^n$ and so $Z(\mathbf{v}) \in \mathcal{A}_n$. This reveals the isomorphism of an $n \times n$ lower triangular Toeplitz inversion and the computation of a reciprocal of a polynomial in z modulo z^n, so that any algorithm that solves one of the two problems also solves the other one.

2.3 Linking Toeplitz Matrix and Polynomial Computation

Another derivation of the isomorphic pair of equations

$$Z(\mathbf{u})Z(\mathbf{v}) = I \quad \text{and} \quad v(z)u(z) = 1 \mod z^n$$

reduces one to the other, multiplication of pairs of polynomials and multiplication of a general Toeplitz matrix by a vector.

Let us recall a simple but fundamental link between polynomials and Toeplitz matrices [6], [3, Section 2.4].

Theorem 1. *(Cf. [3, (2.4.3)].) The matrix equation*

$$\begin{pmatrix} u_0 & & O \\ \vdots & \ddots & \\ \vdots & \ddots & u_0 \\ u_m & \ddots & \vdots \\ & \ddots & \vdots \\ O & & u_m \end{pmatrix} \begin{pmatrix} v_0 \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} p_0 \\ \vdots \\ \vdots \\ p_m \\ \vdots \\ p_{m+n} \end{pmatrix} \tag{1}$$

is equivalent to the polynomial equation

$$\left(\sum_{i=0}^{m} u_i x^i\right)\left(\sum_{i=0}^{n} v_i x^i\right) = \sum_{i=0}^{m+n} p_i x^i.$$

The theorem immediately reduces polynomial multiplication to Toeplitz matrix-by-vector multiplication and vice versa [3, Section 2.4]. Furthermore set

$$\mathbf{p} = (p_0, \ldots, p_{m+n})^{\mathrm{T}} = (1, 0, \ldots, 0)^{\mathrm{T}},$$

keep only the first n equations of the matrix and polynomial equations from the theorem, and obtain that $Z(\mathbf{u})Z(\mathbf{v}) = I$ and $u(z)v(z) = 1 \mod z^n$. Again this shows the equivalence of $n \times n$ lower triangular Toeplitz inversion and computation of the reciprocal of a polynomial in z modulo z^n.

Theorem 1 demonstrates that

$$\begin{pmatrix} \mathbf{x}_1 \\ U\mathbf{v} \\ \mathbf{x}_2 \end{pmatrix} = \mathbf{u} \otimes \mathbf{v}$$

for $(2n\text{-}1)$-vector $\mathbf{u}$ and n-vector $\mathbf{v}$ where $\mathbf{x}_1$ and $\mathbf{x}_2$ are $(n\text{-}1)$-vectors and $U\mathbf{v}$ is the n-vector product of $n \times n$ Toeplitz matrix U and n-vector $\mathbf{v}$ depicted above as the central $n \times n$ block of the matrix in (1) and whose first row and first column is composed of $(2n\text{-}1)$-vector $\mathbf{u}$. This is often used to compute Toeplitz-matrix-by-vector products via FFT [3].

One can then see that

$$\begin{pmatrix} \mathbf{x}_0 \\ U\mathbf{v} \\ \mathbf{x}_2 \end{pmatrix} = \mathbf{w} \otimes \mathbf{v} \tag{2}$$

where $\mathbf{w} = \begin{pmatrix} s \\ \mathbf{u} \end{pmatrix}$ is a $(2n)$-vector defined by prefixing $\mathbf{u}$ with scalar s, $\mathbf{x}_0$ is an n-vector, and $\mathbf{u}$, U, $\mathbf{v}$, and $\mathbf{x}_2$ are as defined for the previous equation.

3 Divide-and-Conquer for Triangular Toeplitz Inversion

We begin with the basic reduction of our inversion problem to polynomial multiplication and the fast FFT-based convolution of the respective coefficient vectors. The recursive block substitution algorithm in [16] relies on the matrix equation

$$T^{-1} = \begin{pmatrix} A & 0 \\ C & A \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} & 0 \\ -A^{-1}CA^{-1} & A^{-1} \end{pmatrix}. \tag{3}$$

For notational convenience from here on, let T be a $(2n) \times (2n)$ triangular Toeplitz matrix and recall that $n = 2^i$ for a positive integer i. C is an $n \times n$ Toeplitz matrix, defined by the $2n-1$ entries in its first row and first column or equivalently by the $(2n-1)$st dimensional subvector $\mathbf{c}$ made up of the last $2n-1$ entries of the vector $T\mathbf{e}_1$. A is an $n \times n$ triangular Toeplitz matrix.

Because T^{-1} is lower triangular Toeplitz, this matrix equation reduces the inversion problem of computing the vector $\bar{\mathbf{t}} = T^{-1}\mathbf{e}_1$ for input T to the same task for half-size input A and two multiplications of half-size Toeplitz matrices by vectors, that is of C by $\bar{\mathbf{a}} = A^{-1}\mathbf{e}_1$ and of A^{-1} by the product $\mathbf{s} = C\bar{\mathbf{a}}$. Recursively one arrives at the solution in $O(M(n))$ ops. In spite of its distinct derivation, the algorithm is isomorphic to the Newton–Sieveking's algorithm for computing polynomial reciprocal modulo a power, that is it shares all the input, output and auxiliary computed values [6, Section 4].

From this point, the process in [16] essentially amounts to computation of the first column $\mathbf{y} = Y\mathbf{e}_1$ of the matrix $Y = A^{-1}CA^{-1}$ provided one has already computed the vector $\bar{\mathbf{a}}$ and is carried out as two truncated convolutions via FFT.

The first one computes $\mathbf{c} \otimes \bar{\mathbf{a}} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{s} \\ \mathbf{x_2} \end{pmatrix}$ via FFT as

$$\begin{pmatrix} \mathbf{x_1} + \mathbf{x_2} \\ \mathbf{s} \end{pmatrix} = W_{2n}(F_{2n}(\mathbf{c}) \star F_{2n}(\bar{\mathbf{a}})).$$

Then, after extracting $\mathbf{s}$, the second computes

$$\bar{\mathbf{a}} \otimes \mathbf{s} = \begin{pmatrix} \mathbf{y} \\ \mathbf{x} \end{pmatrix} \text{ via FFT as } \begin{pmatrix} \mathbf{y} \\ \mathbf{x} \end{pmatrix} = W_{2n}(F_{2n}(\bar{\mathbf{a}}) \star F_{2n}(\mathbf{s})).$$

The task is completed by extracting $\mathbf{y}$. $F_{2n}(\bar{\mathbf{a}})$ is computed only once so that this requires $10m(n) + O(n)$ ops.

At this stage the algorithm in [16] has yielded its minor acceleration versus the Newton–Sieveking's algorithm, and here we obtain further speedup.

4 Our Speedup

4.1 Triangular Toeplitz Speedup

We write $S = CA^{-1}$ and observe that

$$T \begin{pmatrix} A^{-1} \\ 0 \end{pmatrix} = \begin{pmatrix} I \\ S \end{pmatrix}.$$

We then write $Y = A^{-1}S$ and observe that

$$\begin{pmatrix} A^{-1} & 0 \\ 0 & A^{-1} \end{pmatrix} \begin{pmatrix} I \\ S \end{pmatrix} = \begin{pmatrix} A^{-1} \\ Y \end{pmatrix},$$

so that

$$Y = (0, I) \begin{pmatrix} A^{-1} & 0 \\ 0 & A^{-1} \end{pmatrix} T \begin{pmatrix} A^{-1} \\ 0 \end{pmatrix}.$$

To determine $\mathbf{y} = Y\mathbf{e}_1$, the paper [16] computes the product $A^{-1}CA^{-1}\mathbf{e}_1$, but we compute the product

$$(0, I) \begin{pmatrix} A^{-1} & 0 \\ 0 & A^{-1} \end{pmatrix} T \begin{pmatrix} A^{-1} \\ 0 \end{pmatrix} \mathbf{e}_1$$

instead. As in [16], to realize our speedup, we reduce the computation to multiplication of the associated polynomials or equivalently to computing convolutions with the respective coefficient vectors, in this case $\mathbf{t} = T\mathbf{e}_1$ and $\bar{\mathbf{a}} = A^{-1}\mathbf{e}_1$. Whereas the paper [16] computes the convolution of the coefficient vectors $\mathbf{c}$ and $\bar{\mathbf{a}}$ and requires an immediate truncation of the resulting vector before convolving it with $\bar{\mathbf{a}}$, our method does not require truncation between convolution operations. We compute the two successive convolutions

$$\mathbf{t} \otimes \bar{\mathbf{a}} = \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{s} \\ \mathbf{x}_0 \end{pmatrix} = \begin{pmatrix} I \\ S \\ X_0 \end{pmatrix} \mathbf{e}_1$$

and then

$$\bar{\mathbf{a}} \otimes \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{s} \\ \mathbf{x}_0 \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{a}} \\ \mathbf{y} \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{pmatrix} A^{-1} \\ Y \\ X_1 \\ X_2 \end{pmatrix} \mathbf{e}_1.$$

Taking advantage of the commutativity of convolution and computing the above via FFT we have

$$\begin{pmatrix} \bar{\mathbf{a}} \\ \mathbf{y} \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = W_{4n}(F_{4n}(\mathbf{t}) \star F_{4n}(\bar{\mathbf{a}})^2) \tag{4}$$

from which we extract $\mathbf{y}$. Note that because we know $\bar{\mathbf{a}}$ a priori we can short-circuit the FFTs [23] and process only the trailing three quarters of each vector as mentioned in section 1.3.

Let us elaborate just a bit. Short-circuiting the FFTs for the above convolutions is a fairly simple procedure that can be simplified further by slightly modifying (4) as follows:

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{y} \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = W_{4n}(F_{4n}(\mathbf{t}) \star F_{4n}(\bar{\mathbf{a}})^2 - F_{4n}(\bar{\mathbf{a}})) \tag{5}$$

Assume we rely on a radix-2 decimation in frequency FFT and radix-2 decimation in time IFFT to compute (5). In stages one and two of the FFT the leading n entries of the coefficient vector interact only with their corresponding entries in the third and then second quarter of the coefficient vector respectively. Thereafter, through to completion of computation of the FFT the leading n entries do not interact with other entries of the coefficient vector. Clearly then we have no need to compute the leading n entries of $F_{4n}(\mathbf{t})$ and $F_{4n}(\bar{\mathbf{a}})$ in order to compute the trailing $3n$ entries of each. Furthermore, due to our zero padding, we need not compute their values even in the first two stages, since the first stage will not alter their values and the result from the second stage will not be needed. Now let $\mathbf{v} = F_{4n}(\mathbf{t}) \star F_{4n}(\bar{\mathbf{a}})^2 - F_{4n}(\bar{\mathbf{a}})$ and let $4nW_{4n}(\mathbf{v}) = \left(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\right)^T$ where each of the $\mathbf{x}_i$ is an n-vector. Let $\mathbf{x}_i^{(j)}$ be the state of the entries corresponding to $\mathbf{x}_i$, j stages prior to completion of the IFFT. This means, for example that if all but the final two stages of an IFFT were applied to $\mathbf{v}$ the leading n entries of the resulting vector would be $\mathbf{x}_1^{(2)}$ and that $\mathbf{x}_i^{(0)} = \mathbf{x}_i$. While $j > 2$, $\mathbf{x}_1^{(j)}$ has no interaction with $\mathbf{x}_i^{(j)}$ for i = 2, 3, 4. Clearly then all but the final two stages of the IFFT can be carried out on the trailing $3n$ entries of $\mathbf{v}$ in the usual fashion, while avoiding any work on its leading n entries. In fact, we can compute $\mathbf{x}_2^{(2)}$, $\mathbf{x}_3^{(1)}$, and $\mathbf{x}_4^{(1)}$ independent of $\mathbf{x}_1^{(j)}$ for all j. At this point we can work backwards through the IFFT. For notational convenience we will assume that appropriate action is taken to factor out twiddle factors as needed in each of the $\mathbf{x}_i^{(j)}$ presented in the discussion that follows. Having computed $\mathbf{x}_3^{(1)}$ and knowing that $\mathbf{x}_1 = \mathbf{0}$, we can compute $\mathbf{x}_1^{(1)} = \mathbf{x}_1^{(0)} - \mathbf{x}_3^{(1)} = -\mathbf{x}_3^{(1)}$. Once we have determined $\mathbf{x}_1^{(1)}$, we can compute $\mathbf{x}_1^{(2)} = \mathbf{x}_1^{(1)} - \mathbf{x}_2^{(2)} = -\mathbf{x}_3^{(1)} - \mathbf{x}_2^{(2)}$. Having obtained $\mathbf{x}_1^{(2)}$, the IFFT can proceed from where it left off and complete its final two stages. However, because our goal is to determine $\mathbf{x}_2/4n$ only, our task can be simplified by computing $\mathbf{x}_2^{(1)} = -\mathbf{x}_3^{(1)} - 2\mathbf{x}_2^{(2)}$ followed by $\mathbf{x}_2 = -\mathbf{x}_4^{(1)} + \mathbf{x}_2^{(1)}$ instead.

Clearly then computation of (5) requires $9m(n) + O(n)$ ops.

4.2 Further Speedup for Banded Triangular Toeplitz Matrices

Where T is banded with bandwidth $k \leq n$, C in (3) is an upper triangular Toeplitz matrix. In this case

$$\mathbf{t} \otimes \bar{\mathbf{a}} = \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{s} \\ \mathbf{0} \end{pmatrix} \quad \text{and} \quad \bar{\mathbf{a}} \otimes \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{s} \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{a}} \\ \mathbf{y} \\ \mathbf{x} \\ \mathbf{0} \end{pmatrix}, \quad \text{so that}$$

$$\begin{pmatrix} \bar{\mathbf{a}} + \mathbf{x} \\ \mathbf{y} \end{pmatrix} = W_{2n}(F_{2n}(\mathbf{t}) \star F_{2n}(\bar{\mathbf{a}})^2). \tag{6}$$

Since $\bar{\mathbf{a}} + \mathbf{x}$ is unknown prior to computation there will be no short-circuiting of these FFTs, however, the order of these FFTs is half that encountered in (4) for the non-banded case. This is possible because wrapping these convolutions in (6) preserves $\mathbf{y}$ so that we can still extract it. To obtain $F_{2n}(\mathbf{t})$ requires computation of only one fourth of its components. The rest are already known courtesy of (4) in the previous stage. Clearly then this computation requires $4.5m(n) + O(n)$ ops.

Once the bandwidth of T is exhausted, one may apply (6) rather than (4) at each stage thereafter. However, after a single application of (6) it becomes possible to achieve further speedup by changing techniques again.

We already know that

$$T \begin{pmatrix} A^{-1} \\ 0 \end{pmatrix} \mathbf{e}_1 = \begin{pmatrix} I \\ S \end{pmatrix} \mathbf{e}_1 = \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{s} \end{pmatrix} = \mathrm{wrap}_{2n}(\mathbf{t} \otimes \bar{\mathbf{a}}),$$

where I and S are $n \times n$ matrices and so the resulting $\mathbf{e}_1$ and $\mathbf{s}$ are clearly n-vectors. Because T is k-banded, only the first k elements of $\mathbf{t}$ can be non-zero. It follows that only the first $n + k - 1$ elements of $\mathbf{t} \otimes \bar{\mathbf{a}}$ can be non-zero. Since $\mathbf{e}_1$ accounts for the first n elements of $\mathbf{t} \otimes \bar{\mathbf{a}}$, only the first $k - 1$ elements of $\mathbf{s}$ can be non-zero. Of course, we only need the leading $k - 1$ potentially non-zero components of $\mathbf{s}$ to compute the second convolution $\bar{\mathbf{a}} \otimes \mathbf{s}$. Let $h = 2^{\lceil \lg k \rceil}$ and we can obtain those Fourier coefficients for $\mathbf{s}$ that we will need later while employing a minimal amount of zero padding as follows:

$$F_{2h}(\mathbf{s}) = F_{2h}(\mathbf{t} \otimes \bar{\mathbf{a}} - \mathbf{e}_1) = F_{2h}(\mathbf{t}) \star F_{2h}(\bar{\mathbf{a}}) - F_{2h}(\mathbf{e}_1). \tag{7}$$

Recall that $F_{2h}(\mathbf{e}_1)$ is composed exclusively of ones so that its use in the element-wise subtraction above requires relatively little computational effort and note that $F_{2h}(\mathbf{t})$ is known from the stage where (6) was applied. The only computations required then are to determine $F_{2h}(\bar{\mathbf{a}})$ and to apply the two element-wise operations. The latter are dispensed with in $O(h)$ ops and we can compute $F_{2h}(\bar{\mathbf{a}})$ from $\bar{\mathbf{a}}$ in $2m(h) + O(n)$ ops. The $O(n)$ term is the result of wrapping n-vector $\bar{\mathbf{a}}$ into $(2h)$-vector $\mathrm{wrap}_{2h}(\bar{\mathbf{a}})$ before applying the FFT. However, an alternative method will provide us with $F_{2h}(\bar{\mathbf{a}})$ in only $O(n)$ ops.

Relying for a moment on matrix representation, recall that once we have $\mathbf{s}$, we want to compute the product $y = A^{-1}\mathbf{s}$. Since only the first $k-1$ elements of $\mathbf{s}$ can be non-zero, only the first $k-1$ columns of A^{-1} contribute to computation of $y = A^{-1}\mathbf{s}$. With this in mind we partition A^{-1} into $h \times h$ blocks and multiply each block in the first column of block matrix A^{-1} by the h dimensional sub-vector of $\mathbf{s}$ made up of its first h entries, which contains all $k-1$ of its potentially non-zero elements.

Again, we resort to the use of the coefficient vectors of the associated polynomials. Let Toeplitz matrix $A^{-1}_{i,j}$ be the $(i,j)^{th}$ $h \times h$ block of block matrix A^{-1}, let h-vector $\bar{\mathbf{a}}_0 = \mathbf{0}$, let h-vectors $\bar{\mathbf{a}}_i = A^{-1}_{i,1}\mathbf{e}_1$, for $i = 1, \ldots, n/h$, let $(2h)$-vectors

$$\hat{\mathbf{a}}_i = \begin{pmatrix} \bar{\mathbf{a}}_{i-1} \\ \bar{\mathbf{a}}_i \end{pmatrix},$$

for $i = 1, \ldots, n/h$, then using (2),

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y}_i \end{pmatrix} = \mathrm{wrap}_{2h}(\hat{\mathbf{a}}_i \otimes \mathbf{s}) = W_{2h}(F_{2h}(\hat{\mathbf{a}}_i) \star F_{2h}(\mathbf{s})), \tag{8}$$

$$\text{and } \mathbf{y} = \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_{n/h} \end{pmatrix}.$$

Obviously, computing $\mathbf{y}$ via (7) and (8) is dependent on $F_{2h}(\bar{\mathbf{a}}_i)$, for $i = 1, \ldots, n/h$. We can compute each of the $F_{2h}(\bar{\mathbf{a}}_i)$ given $\bar{\mathbf{a}}_i$ in $2m(h)$ ops. Note that $F_{2h}(\bar{\mathbf{a}}_i)$, for $i = 1, \ldots, n/2h$ will have already been computed in previous stages and can be reused. Computing the $n/2h$ remaining $F_{2h}(\bar{\mathbf{a}}_i)$, for $i = n/(2h+1), \ldots, n/h$ will require $m(n) - O(n)$ ops. Each of the IFFTs is computed in $2m(h) + O(h)$ ops, so that all n/h IFFTs combined require $2m(n) + O(n)$ ops. Since

$$\sum_{i=1}^{n/2h} F_{2h}(\hat{\mathbf{a}}_{2i}) = F_{2h}(\bar{\mathbf{a}}),$$

given the $F_{2h}(\bar{\mathbf{a}}_i)$, we compute $F_{2h}(\bar{\mathbf{a}})$ in $O(n)$ additional ops as promised above and therefore obtain $F_{2h}(\mathbf{s})$ in $O(n)$ ops. In this way we compute $\mathbf{y}$ via (7) and (8) in $3m(n) + O(n)$ ops.

Once employed we can apply (7) and (8), as described in the previous paragraph, to each stage thereafter. However, (7) and (8) can be exploited in another manner to further lower the operation count once they have already been utilized at least once as already described above. Rather than computing additional $F_{2h}(\bar{\mathbf{a}}_i)$ at each stage, we use only those $F_{2h}(\bar{\mathbf{a}}_i)$ already computed during previous stages to determine the new $\mathbf{y}_i$.

Clearly, we will not have enough $F_{2h}(\bar{\mathbf{a}}_i)$ to compute all of $\mathbf{y}$ in the manner we did above. However each of the $\mathbf{y}_i$ we can produce corresponds to one of the $\bar{\mathbf{a}}_i$ of what would otherwise be the next stage in the recursion. This allows us to extend $\bar{\mathbf{a}}$ with the new $\mathbf{y}_i$. Therefore, compute

$$F_{2h}(\bar{\mathbf{a}}') = F_{2h}(\bar{\mathbf{a}}) + F_{2h}\left(\sum \begin{pmatrix} \mathbf{y}_i \\ \mathbf{y}_{i+1} \end{pmatrix}\right)$$

summing over each consecutive pair of the newly generated $\mathbf{y}_i$. This entails $m(h) + O(h)$ ops. Thus we compute a new $F_{2h}(\mathbf{s})$ via (7) in $m(h) + O(h)$ ops. We utilize this $F_{2h}(\mathbf{s})$ in multiple computations, via (8), to produce the next sequence of $\mathbf{y}_i$ in the current stage without need to compute any new $F_{2h}(\bar{\mathbf{a}}_i)$. Instead we simply compute the results of the pairwise multiplication and IFFT of (8) in $2m(h) + O(h)$ ops for each of the $\mathbf{y}_i$ in the sequence. We repeat until $\mathbf{y}$ has been determined. This requires $(2 + 2^{-s})m(n) + O(n)$ ops, where s is the number of stages in which we previously applied (7) and (8) via our first method.

5 Some Potential Implementation Advantages

Our inversion algorithm based on (4) enjoys practical advantages over the algorithm in [16], which stem from the commutativity of vector convolution. To begin with, computing the square of a complex number requires less floating point operations than general complex multiplication. From a hardware perspective note that limiting the need to access memory is essential where the goal is high performance of numerical software [24]. These complex squaring operations require less working storage and can be applied to the elements of $F_{2N}(\bar{\mathbf{a}})$ as soon as they are generated from their respective FFT operations, that is, while these values still reside in the registers where they were initially produced, thereby reducing the need for cache/memory access between operations.

The memory requirement of (7) with (8) nearly doubles with each iteration when applied as first suggested, whereas the second method by which we propose using (7) and (8) maintains a constant footprint. Therefore depending on the size of the matrix and the memory hierarchy of the computational device in use, it may be advantageous to transition from one method to the other before reaching the optimal crossover point in terms of the number of ops required to complete the inversion. For example, on a general purpose central processing unit this can prevent cache misses by avoiding the need to store additional vector segments that exceed the cache capacity. Another example would involve computation on a graphics processing unit, a field which has evoked a great deal of interest from researches lately. Here the memory is often partitioned so that it is expensive in terms of latency to cross a partition. The above mentioned early transition can help avoid the need to cross such boundaries.

Where the dimension of T is not a power of two and T is banded the last stage in which we apply (8) can simply avoid computing those instances of (8) that would otherwise result in extraneous elements and where T is not banded additional short-circuiting may be possible.

The aforementioned implementation issues call for examination.

References

1. Kailath, T., Kung, S.Y., Morf, M.: Displacement Ranks of Matrices and Linear Equations. Journal Math. Analysis and Appls. 68(2), 395–407 (1979)
2. Bini, D., Pan, V.Y.: Polynomial and Matrix Computations, Volume 1. Fundamental Algorithms. Birkhäuser, Boston (1994)
3. Pan, V.Y.: Structured Matrices and Polynomials: Unified Superfast Algorithms. Birkhäuser/Springer, Boston/New York (2001)
4. Chan, R.: Toeplitz preconditioners for Toeplitz systems with nonnegative generating functions. IMA J. Numer. Anal. 11, 333–345 (1991)
5. Lin, F.R., Ching, W.K.: Inverse Toeplitz preconditioners for Hermitian Toeplitz systems. Numer. Linear Algebra Appl. 12, 221–229 (2005)
6. Bini, D., Pan, V.Y.: Polynomial Division and Its Computational Complexity. Journal of Complexity 2, 179–203 (1986)
7. Pan, V.Y.: Complexity of Computations with Matrices and Polynomials. SIAM Review 34(2), 225–262 (1992)

8. Sieveking, M.: An Algorithm for Division of Power Series. Computing 10, 153–156 (1972)
9. Bini, D.: Parallel Solution of Certain Toeplitz Linear Systemns. SIAM J. on Computing 13(2), 268–276 (1984)
10. Bini, D., Pan, V.Y.: Improved Parallel Polynomial Division. SIAM J. on Computing 22(3), 617–627 (1993); Proc. version in FOCS 1992, pp. 131–136. IEEE Computer Society Press, Los Alamitos (1992)
11. Reif, J.H., Tate, S.R.: Optimum Size Division Circuits. SIAM J. on Computing 19(5), 912–925 (1990)
12. Pan, V.Y., Landowne, E., Sadikou, A.: Polynomial Division with a Remainder by Means of Evaluation and Interpolation. Information Processing Letters 44, 149–153 (1992)
13. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: The Design and Analysis of Algorithms. Addison-Wesley, Reading (1974)
14. Borodin, A.B., Munro, I.: Computational Complexity of Algebraic and Numeric Problems. American Elsevier, New York (1975)
15. von zur Gathen, J., Gerhard, J.: Modern Computer Algebra, 2nd edn. Cambridge University Press, Cambridge (2003) (first edition, 1999)
16. Commenges, D., Monsion, M.: Fast inversion of triangular Toeplitz matrices. IEEE Trans. Automat. Control AC-29, 250–251 (1984)
17. Bernstein, D.J.: Removing redundancy in high-precision Newton iteration (2004), http://cr.yp.to/fastnewton.html#fastnewton-paper
18. van der Hoeven, J.: Newton's method and FFT trading. Journal of Symbolic Computing 45(8), 857–878 (2010)
19. Hanrot, G., Zimmermann, P.: Newton iteration revisited, http://www.loria.fr/~zimmerma/papers/fastnewton.ps.gz
20. Schoenhage, A., Vetter, E.: A New Approach to Resultant Computations and Other Algorithms with Exact Division. In: European Symposium on Algorithms, pp. 448–459 (1994)
21. Schonhage, A.: Variations on computing reciprocals of power series. Inf. Process. Lett. 74(1-2), 41–46 (2000)
22. Harvey, D.: aster algorithms for the square root and reciprocal of power series. Math. Comp. 80, 387–394 (2011)
23. Murphy, B.: Short-circuited FFTs for computing paritally known convolutions (2010), http://comet.lehman.cuny.edu/bmurphy/research/ScFFT.pdf
24. Dongarra, J., Hammarling, S., Sorensen, D.: Block reduction of matrices to condensed form for eigenvalue computations. J. Comp. Appl. Math. 27, 215 (1989)

Computing a Basin of Attraction to a Target Region by Solving Bilinear Semi-Definite Problems[⋆]

Zhikun She and Bai Xue

SKLSDE, LMIB and School of Mathematics and Systems Science, Beihang University, China
zhikun.she@buaa.edu.cn

Abstract. In this paper, we present a sum of squares programming based method for computing a basin of attraction to a target region as large as possible by iteratively searching for Lyapunov-like functions. We start with the basic mathematical notions and show how attraction to a target region can be ensured by Lyapunov-like functions. Then, we present an initial framework for getting an increasing sequence of basins of attraction by iteratively computing Lyapunov-like functions. This framework can be realized by solving bilinear semi-definite problems based on sums of squares decomposition. We implement our algorithm and test it on some interesting examples. The computation results show the usefulness of our method.

1 Introduction

Stability of a nonlinear continuous system of ordinary differential equations (ODE system) is a very important subject in control design and pure theoretical analysis. Especially, for the classical theory of stability where Lyapunov stability is the most common definition [7], finding a region of attraction to an equilibrium of an ODE system is of significant importance in engineering and science [13,12]. Usually, there are two kinds of methods aiming to estimate regions of attraction to an equilibrium.

One kind of methods is the Lyapunov based method [1,2,5,10,16,26,25] according to the local stability theorem. Their key problem is to search for Lyapunov functions which quantitatively prove local stability. In cases where the system is polynomial, due to decidability of the theory of real-closed fields [27], for a given polynomial with parametric coefficients, one can decide whether there are instantiations of these parameters resulting in a Lyapunov function [22]. In addition, a method that uses Gröbner bases has also been used to choose the parameters in Lyapunov functions in an optimal way [3]. Moreover, a method based on sum of squares (SOS) decomposition [14,15] has appeared that can compute Lyapunov functions for some realistic examples.

The other kind of methods is the trajectory reversing based method [4,6,11,29] by considering the topological property. For example, a backward integration technique is used in [4] for estimating regions of asymptotic stability; a polynomial level-set method, associated with an implicit time-stepping algorithm, is used in [29] for backwardly advection of a small initial neighborhood of the equilibrium; a numerical level set

⋆ This work was partly supported by NSFC-61003021, Beijing Nova Program and SKLSDE-2011ZX-16.

method is used in [6,11] to estimate a region of attraction by solving a time-dependent Hamilton-Jacobi-Isaacs partial differential equation.

However, being based on the classical stability, all the above methods do not directly allow reasoning about a basin of attraction to a target region [20], in which there may be no equilibrium or there may be a limit cycle and an unstable equilibrium. Moreover, for many realistic applications, trajectories are required to stay in the target region forever, that is, practical stability [9] instead of classical stability is considered.

Theoretically, practical stability can be analyzed based on comparison principles [9,30,31]. However, these theories are in general unpractical, which has been pointed out in [21]. Thus, a feasible approach [21] is presented to analyze practical stability of uncertain nonlinear systems described by a parameterized family of ordinary differential equations with uncertain initial values. Recently, an interval based branch and relax algorithm [20] is used to provide a basin of attraction to a target region of a polynomial system by computing a Lyapunov-like function.

In this paper, we present a sum of squares programming based method for computing a basin of attraction to a target region as large as possible by iteratively searching for Lyapunov-like functions with sum of squares formula. We start with the basic mathematical notions used in this paper and show how attraction to a target region can be ensured by Lyapunov-like functions. Then, we present an initial framework for getting an increasing sequence of basins of attraction by iteratively computing Lyapunov-like functions. This framework can be under-approximated based on sums of squares and then realized by solving bilinear semi-definite problems.

We use the toolbox **PENBMI** [8] to implement our algorithm and test it on several interesting examples. The computation results and computation times show the efficiency and usefulness of our method.

Note that, if every trajectory starting from the target region always stays in it, then our method can be used for practical stability analysis. Moreover, assuming that the target region is a region of attraction to an equilibrium, our method can be used to produce a larger region of attraction.

2 Problem Formulation

In this section we will introduce the basic mathematical notions used in this paper and show how attraction to a target region can be ensured by Lyapunov-like functions.

For an ordinary differential equation $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$, where $\boldsymbol{x} \in \mathbb{R}^n$, we denote by $\boldsymbol{x}(\cdot, \boldsymbol{x}_0) : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ its trajectory starting from $\boldsymbol{x}_0$, where $\mathbb{R}_{\geq 0} = \{t : t \geq 0\}$.

The following definition of stability that we will use allows us to explicitly specify a target region and a basin of attraction:

Definition 1. *[20] Given an ordinary differential equation $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ with sets U and TR such that $TR \subset U \subseteq \mathbb{R}^n$, where U is an explicit parameter, the differential equation is stable with respect to U and the target region TR if for every point $\boldsymbol{x}_0 \in U$, the trajectory $\boldsymbol{x}(\cdot, \boldsymbol{x}_0)$ will*

1. *always stay in U (for all $t \in \mathbb{R}_{\geq 0}, \boldsymbol{x}(t, \boldsymbol{x}_0) \in U$),*
2. *and eventually reach TR (there is a $t_1 \in \mathbb{R}_{\geq 0}$ such that $\boldsymbol{x}(t_1, \boldsymbol{x}_0) \in TR$).*

Moreover, U is called as a basin of attraction to the target region TR.

In Definition 1, trajectories may enter and leave TR infinitely. Moreover, by allowing an explicit parameter U, one can specify a desired basin of attraction. This helps us to avoid situations, where a differential equation is stable, but the found Lyaunov-like function only proves attraction with a tiny region. Further, by allowing a target region instead of a single equilibrium point, the method can also be applied in cases where no equilibrium exists (e.g., when we want to study attraction to a limit cycle).

Without loss of generality, we can assume that $\mathbf{0} \in TR$, where $\mathbf{0}$ may not be the equilibrium. In order to ensure the above stability notion, we use the following adapted notion of a Lyapunov-like function:

Definition 2. *For a given differential equation $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ with sets TR and B such that $\mathbf{0} \in TR$ and $TR \subset B$, where B is an explicit parameter, a continuously differential function $V(\boldsymbol{x})$ is called a Lyapunov-like function with respect to TR if and only if the following constraints*

1. $\forall \boldsymbol{x} \in \mathbb{R}^n\, [V(\mathbf{0}) = 0 \wedge [V(\boldsymbol{x}) > 0 \Leftrightarrow \boldsymbol{x} \neq \mathbf{0}]]$ [1] *and*
2. $\forall \boldsymbol{x} \in B \left[\boldsymbol{x} \notin TR \Rightarrow \frac{d}{dt}V(\boldsymbol{x}) < 0\right]$

hold. Here, $\Rightarrow$ denotes an implication symbol, $\Leftrightarrow$ denotes a biimplication symbol, and $\frac{d}{dt}V(\boldsymbol{x})$ denotes the time-derivative of V along f, that is, $\frac{\partial V}{\partial \boldsymbol{x}}^T f(\boldsymbol{x})$.

Then a Lyapunov-like function $V(\boldsymbol{x})$ ensures that sublevel sets of V in B are never left and the target region TR is eventually reached. Here, given a closed set B and a Lyapunov-like function V, we define a s-sublevel set of V in B to be $\{x \in B : V(x) \leq s < t, t = \min_{\boldsymbol{x}\in\partial B} V(\boldsymbol{x})\}$, where $s > 0$ and ∂B denotes the boundary of B, and denote this s-sublevel set by $V^B_{\leq s}$. Similarly, we can denote the set $\{x \in B : V(x) < s\}$ by $V^B_{<s}$.

Theorem 1. *Given an ordinary differential equation $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ with an open target region TR and a closed set B such that $TR \subset B$, assume that $V(\boldsymbol{x})$ is a Lyapunov-like function with respect to TR. Then, for all the s-sublevel set $V^B_{\leq s}$ in B such that $TR \subset V^B_{\leq s}$, the differential equation is stable with respect to $V^B_{\leq s}$ and TR. Moreover, $V^B_{\leq s}$ is a basin of attraction to the target region TR.*

Proof. Since $V(\boldsymbol{x})$ is a Lyapunov-like function with respect to TR, due to Definition 2, $\forall \boldsymbol{x} \in B[\boldsymbol{x} \notin TR \Rightarrow \frac{d}{dt}V(\boldsymbol{x}) < 0]$ holds.

Let $B' = \{x \in B : V(x) \leq t, t = \min_{\boldsymbol{x}\in\partial B} V(\boldsymbol{x})\}$. Clearly, $V^B_{\leq s} \subset B' \subset B$. Since $TR \subset V^B_{\leq s}$, then every trajectory starting in $V^B_{\leq s}$ will not leave $V^B_{\leq s}$ at all, which can be easily obtained by constructing a contradiction.

Moreover, since TR is an open set and $B' \setminus TR$ is closed, due to the continuity of $\frac{d}{dt}V(\boldsymbol{x})$, we know that $\frac{d}{dt}V$ has a maximum ϵ in $B' \setminus TR$. Obviously, the maximum ϵ is a negative number.

Let $\boldsymbol{x}_0$ be an arbitrary, but fixed point in $V^B_{\leq s}$. For proving the trajectory starting from $\boldsymbol{x}_0$ will eventually enter TR, it is sufficient to only consider the case $\boldsymbol{x}_0 \in V^B_{\leq s} \setminus TR$.

[1] We can prove Theorem 1 without this condition. However, with this condition, we can in Section 3 compute a basin of attraction more conveniently. Otherwise, we need to compute a basin of attraction in a optimal way [18], increasing the computational complexity [23,24].

We assume that for all $t \in \mathbb{R}_{\geq 0}$, $\boldsymbol{x}(t, \boldsymbol{x}_0) \notin TR$, and we derive a contradiction. Since $\boldsymbol{x}(t, \boldsymbol{x}_0) \notin TR$ and $V^B_{\leq s}$ is never left, for all $t \in \mathbb{R}_{\geq 0}$, $\boldsymbol{x}(t, \boldsymbol{x}_0) \in V^B_{\leq s} \setminus TR$. Thus, for all $t \in \mathbb{R}_{\geq 0}$, $\frac{d}{dt}V(\boldsymbol{x}(t, \boldsymbol{x}_0)) \leq \epsilon$, implying that $V(\boldsymbol{x}(t, \boldsymbol{x}_0)) \leq \epsilon t + V(\boldsymbol{x}_0)$. Since ϵ is negative, this implies that as t goes to infinity, $V(\boldsymbol{x}(t, \boldsymbol{x}_0))$ goes to minus infinity, contradicting to the fact that $V(\boldsymbol{x}) > 0$ in $V^B_{\leq s} \setminus TR$. Thus, there exists a $t \in \mathbb{R}_{\geq 0}$ such that $\boldsymbol{x}(t, \boldsymbol{x}_0) \in TR$.

Therefore, due to Definition 1, the differential equation is stable with respect to $V^B_{\leq s}$ and TR. Moreover, $V^B_{\leq s}$ is a basin of attraction to the target region TR. □

Note that without the requirement that the target region TR is open in Theorem 1, trajectories may go arbitrarily close to TR but not enter TR for case that the boundary of TR forms a limit cycle.

Does a Lyapunov-like function $V(x)$ guarantee that every trajectory starting in $V^B_{\leq s}$ will eventually stay in TR? No! The reason is that it will not prohibit a trajectory from infinitely often entering TR, staying within TR for a period of time and then leaving TR again. This intuition can be illustrated by a two-dimensional case in Fig. 1.

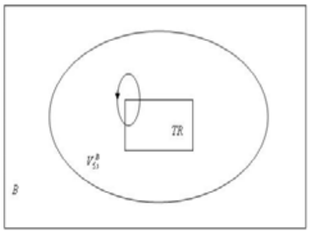


Fig. 1. An example with a cycle

However, with the assumption that every trajectory starting from TR always stays in TR, then every trajectory starting in a basin of attraction $V^B_{\leq s}$ will eventually stay in TR, which implies that the system is practically stable [9]. Moreover, assuming that TR is a region of attraction to an equilibrium, we can obtain a larger region of attraction to the equilibrium by iteratively computing Lyapunov-like functions.

3 Computing a Basin of Attraction

In our previous work [20], we have introduced an interval based branch-and-relax algorithm to provide a basin of attraction to target region by computation of a Lyapunov-like function. However, we cannot iteratively provide basin of attractions. In this section, we will use a sum of squares programming based method to iteratively compute an increasing sequence of basins of attraction by repeatedly computing Lyapunov-like functions. In this way, we can obtain a basin of attraction as large as possible.

3.1 Framework of our Algorithm

Without loss of generality, we can assume that TR is defined by the set $\{\boldsymbol{x} : p(\boldsymbol{x}) < 0\}$ and $\mathbf{0} \in TR$, where $p(\boldsymbol{x})$ is a polynomial.

In addition, we suppose that $V(\boldsymbol{x})$ is a Lyapunov-like function such that $\Omega = \{\boldsymbol{x} : V(\boldsymbol{x}) \le 1\}$ is an initial basin of attraction to the target region *TR*.[2]

Starting from the initial basin of attraction Ω, we try to enlarge Ω by computing a function $V'(\boldsymbol{x})$ such that the following conditions holds

$$\Omega \subset \Omega', \tag{1}$$

$$forall \boldsymbol{x} \in \mathbb{R}^n \left[V'(\boldsymbol{0}) = 0 \wedge [V'(\boldsymbol{x}) > 0 \Leftrightarrow \boldsymbol{x} \neq \boldsymbol{0}]\right], \text{ and} \tag{2}$$

$$\forall \boldsymbol{x} \in \Omega' \left[V(\boldsymbol{x}) \ge 1 \Rightarrow \frac{d}{dt}V'(\boldsymbol{x}) < 0\right], \tag{3}$$

where $\Omega' = \{\boldsymbol{x} : V'(\boldsymbol{x}) \le 1\}$. Such an enlargement can be ensured by Proposition 1.

Proposition 1. *Assume that $V'(\boldsymbol{x})$ is a function such that Conditions (1), (2) and (3) hold. Then, $\Omega' = \{\boldsymbol{x} : V'(\boldsymbol{x}) \le 1\}$ is also a basin of attraction to TR.*

Proof. Since Conditions (1), (2) and (3) hold, due to Definition 2, $V'(\boldsymbol{x})$ is a Lyapunov-like function with respect to the set $\{\boldsymbol{x} : V(\boldsymbol{x}) < 1\}$. From Condition (2) and Theorem 1, Ω' is a basin of attraction to the set $\{\boldsymbol{x} : V(\boldsymbol{x}) < 1\}$. That is, every trajectory starting in Ω' will always stay in Ω' and eventually enter the set $\{\boldsymbol{x} : V(\boldsymbol{x}) < 1\}$.

Since Ω is a basin of attraction to *TR*, every trajectory starting in Ω will enter *TR* eventually. Thus, every trajectory starting in Ω' will always stay in Ω' and enter *TR* eventually. According to Definition 1, Ω' is a basin of attraction to *TR*. □

Hence, by iteratively computing Lyapunov-like functions satisfying Conditions (1), (2) and (3), we can arrive at Algorithm 1 to compute a basin of attraction to *TR* as large as possible as follows.

Algorithm 1. Computing a basin of attraction to *TR*

```
Input: a polynomial differential system $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$ and a target region TR.
Output: a basin of attraction to TR.
1: compute an initial Lyapunov-like function $V(\boldsymbol{x})$ such that $\Omega = \{\boldsymbol{x} : V(\boldsymbol{x}) \le 1\}$ is an estimate
   of basin of attraction to TR;
2: while a function $V'(\boldsymbol{x})$ such that constraints (1), (2) and (3) hold is found do
3:     $V(\boldsymbol{x}) := V'(\boldsymbol{x})$ and $\Omega := \{\boldsymbol{x} : V(\boldsymbol{x}) \le 1\}$;
4: end while
5: return $\Omega$ as a basin of attraction to TR.
```

3.2 Implementation Using Bilinear Semi-definite Programming

In this subsection, for a polynomial vector field, we will explain how to use the sum of squares programming existing in the literature to implement Algorithm 1, such that we can obtain a basin of attraction to the given target region *TR* as large as possible.

[2] Suppose that $V(\boldsymbol{x})$ is a Lyapunov-like function such that $V^B_{\le s}$ is a basin of attraction to target region. Letting $V'(\boldsymbol{x}) = V(\boldsymbol{x})/s$, according to Definition 2 and Theorem 1, then $V'(\boldsymbol{x})$ is also a Lyapunov-like function and $\Omega = \{\boldsymbol{x} : V'(\boldsymbol{x}) \le 1\}$ is a basin of attraction to target region. So, this supposition is feasible.

For this, we want to explain two things:

1. one is how to compute an initial Lyapunov-like function $V(\boldsymbol{x})$ such that $\Omega = \{\boldsymbol{x} : V(\boldsymbol{x}) \leq 1\}$ is an estimate of basin of attraction to TR;
2. the other is how to find a function $V'(\boldsymbol{x})$ such that Conditions (1), (2) and (3) hold.

Let $\mathbb{R}[\boldsymbol{x}]$ be a polynomial ring over $\mathbb{R}$ and $\sum$ be the set of sum of squares polynomials, that is, $\sum = \{q \in \mathbb{R}[\boldsymbol{x}] | q = \sum_{i=1}^{t} f_i^2, f_i \in \mathbb{R}[\boldsymbol{x}]\}$.

In addition, we assume that V and V' in Algorithm 1 are all sum of squares polynomials of degree d, where d is an even parameter.

We will start with under-approximations of the conditions (1), (2) and (3) using sum of squares formula, in the sense that the solution set of these under-approximations is a subset of the original conditions. And then we explain how to solve these under-approximations using bilinear semi-definite programming.

First, by introducing $l(\boldsymbol{x}) = h \sum_{i=1}^{n} x_i^d$, we can under-approximate the condition (2) using the condition that there is a $h > 0$ such that $V'(\boldsymbol{x}) - l(\boldsymbol{x}) \in \sum$ due to the following proposition.

Proposition 2. *For all $V_0'(\boldsymbol{x})$ such that there is a $h > 0$ such that $V_0'(\boldsymbol{x}) - l(\boldsymbol{x}) \in \sum$, $V_0'(\boldsymbol{x})$ is positive definite.*

Second, we can under-approximate the condition (3) using the condition that there exist $m > 0$ and $s_1(\boldsymbol{x}), s_2(\boldsymbol{x}), s_3(\boldsymbol{x}) \in \sum$ such that

$$-s_1(\boldsymbol{x})(1 - V'(\boldsymbol{x})) - s_2(\boldsymbol{x})(V(\boldsymbol{x}) - 1) - s_3(\boldsymbol{x})\frac{d}{dt}V'(\boldsymbol{x}) - m \in \sum,$$

which is ensured by the following proposition.

Proposition 3. *For all $V_0'(\boldsymbol{x})$ such that there exist $m > 0$ and $s_1(\boldsymbol{x}), s_2(\boldsymbol{x}), s_3(\boldsymbol{x}) \in \sum$ such that $-s_1(\boldsymbol{x})(1 - V_0'(\boldsymbol{x})) - s_2(\boldsymbol{x})(V(\boldsymbol{x}) - 1) - s_3(\boldsymbol{x})\frac{d}{dt}V_0'(\boldsymbol{x}) - m \in \sum$, $V_0'(\boldsymbol{x})$ satisfies $\{\boldsymbol{x} \in \mathbb{R}^n : 1 - V_0'(\boldsymbol{x}) \geq 0, V(\boldsymbol{x}) \geq 1\} \subseteq \{\boldsymbol{x} \in \mathbb{R}^n : -\frac{d}{dt}V_0'(\boldsymbol{x}) > 0\}$.*

Proof. Since $\{\boldsymbol{x} \in \mathbb{R}^n : 1 - V_0'(\boldsymbol{x}) \geq 0, V(\boldsymbol{x}) \geq 1\} \subseteq \{\boldsymbol{x} \in \mathbb{R}^n : \frac{d}{dt}V_0'(\boldsymbol{x}) < 0\}$ is equivalent to

$$\{\boldsymbol{x} \in \mathbb{R}^n : 1 - V_0'(\boldsymbol{x}) \geq 0, V(\boldsymbol{x}) - 1 \geq 0, \frac{d}{dt}V_0'(\boldsymbol{x}) \geq 0\} = \emptyset, \tag{4}$$

it is enough to prove that $V_0'(\boldsymbol{x})$ makes Condition (4) hold.

Let $q(\boldsymbol{x}) = -s_1(\boldsymbol{x})(1 - V_0'(\boldsymbol{x})) - s_2(\boldsymbol{x})(V(\boldsymbol{x}) - 1) - s_3(\boldsymbol{x})\frac{d}{dt}V_0'(\boldsymbol{x}) - m$. Then $q(\boldsymbol{x}), q(\boldsymbol{x}) + m \in \sum$. In addition, let $f = (q(\boldsymbol{x}) + m) + s_1(\boldsymbol{x})(1 - V_0'(\boldsymbol{x})) + s_2(\boldsymbol{x})(V(\boldsymbol{x}) - 1) + s_3(\boldsymbol{x})\frac{d}{dt}V_0'(\boldsymbol{x})$. Then, $f \equiv 0$.

If there is an $\boldsymbol{x}_0 \in \{\boldsymbol{x} \in \mathbb{R}^n : 1 - V_0'(\boldsymbol{x}) \geq 0, V(\boldsymbol{x}) - 1 \geq 0, \frac{d}{dt}V_0'(\boldsymbol{x}) \geq 0\}$, then $f(\boldsymbol{x}_0) \geq m$, contradicting $f \equiv 0$. Thus, Condition (4) holds and we finish proving this proposition. □

Third, for under-approximating the condition (1), by introducing a small parameter $\epsilon > 0$,[3] we first construct a new region $\Omega_\epsilon = \{\boldsymbol{x} : V(\boldsymbol{x}) \leq 1 + \epsilon\}$. Further, we under-approximately replace $\Omega_\epsilon \subseteq \Omega'$ by the condition that there exists $s_0(\boldsymbol{x}) \in \sum$ such that

$$-s_0(\boldsymbol{x})(1 + \epsilon - V(\boldsymbol{x})) + (1 - V'(\boldsymbol{x})) \in \sum,$$

[3] The small positive parameter ϵ can also be viewed as a stopping criterion for our iterative algorithm.

which is ensured by the following proposition.

Proposition 4. *For all $V_0'(\boldsymbol{x})$ such that there exists $s_0(\boldsymbol{x}) \in \sum$ such that $-s_0(\boldsymbol{x})(1+\epsilon - V(\boldsymbol{x})) + (1 - V'(\boldsymbol{x})) \in \sum$, $V_0'(\boldsymbol{x})$ satisfies $\{\boldsymbol{x} : V(\boldsymbol{x}) \le 1+\epsilon\} \subseteq \{\boldsymbol{x} : V_0'(\boldsymbol{x}) \le 1\}$, implying that $\Omega \subset \Omega_\epsilon \subseteq \Omega'$.*

Proof. Since $\{\boldsymbol{x} : V(\boldsymbol{x}) \le 1+\epsilon\} \subseteq \{\boldsymbol{x} : V_0'(\boldsymbol{x}) \le 1\}$ is equivalent to

$$\{\boldsymbol{x} \in \mathbb{R}^n : 1+\epsilon - V(\boldsymbol{x}) \ge 0, V_0'(\boldsymbol{x}) - 1 > 0\} = \emptyset, \tag{5}$$

it is enough to prove that $V_0'(\boldsymbol{x})$ makes Condition (5) hold.

Let $q(\boldsymbol{x}) = -s_0(\boldsymbol{x})(1+\epsilon - V(\boldsymbol{x})) + (1 - V'(\boldsymbol{x}))$. Then $q(\boldsymbol{x}) \in \sum$. In addition, let $g = V_0'(\boldsymbol{x}) - 1$ and $f = q(\boldsymbol{x})(V_0'(\boldsymbol{x}) - 1) + s_0(\boldsymbol{x})(1+\epsilon - V(\boldsymbol{x}))(V_0'(\boldsymbol{x}) - 1)$. Then $f + g^2 \equiv 0$.

If there is an $\boldsymbol{x}_0 \in \{\boldsymbol{x} \in \mathbb{R}^n : 1+\epsilon - V(\boldsymbol{x}) \ge 0, V_0'(\boldsymbol{x}) - 1 > 0\}$, then $f(\boldsymbol{x}_0) + g^2(\boldsymbol{x}_0) > 0$, contradicting $f + g^2 \equiv 0$. So, Condition (5) holds and we finish proving this proposition. □

Combining the above under-approximations, it is straightforward to obtain that: if there exist $V'(\boldsymbol{x}) \in \sum$, $m > 0$, $h > 0$ and $s_0(\boldsymbol{x}), s_1(\boldsymbol{x}), s_2(\boldsymbol{x}), s_3(\boldsymbol{x}) \in \sum$ such that

$$V'(\boldsymbol{x}) - h\sum_{i=1}^{n} x_i^d \in \sum, \tag{6}$$

$$-s_0(\boldsymbol{x})(1+\epsilon - V(\boldsymbol{x})) + (1 - V'(\boldsymbol{x})) \in \sum, \text{ and} \tag{7}$$

$$-s_1(\boldsymbol{x})(1 - V'(\boldsymbol{x})) - s_2(\boldsymbol{x})(V(\boldsymbol{x}) - 1) - s_3(\boldsymbol{x})\frac{d}{dt}V'(\boldsymbol{x}) - m \in \sum \tag{8}$$

hold, then the same $V'(\boldsymbol{x})$ make the conditions (1), (2) and (3) hold.

It can be seen from the structure of Conditions (6)~(8) that finding $V'(\boldsymbol{x}) \in \sum$, $m > 0$, $h > 0$ and $s_0(\boldsymbol{x}), s_1(\boldsymbol{x}), s_2(\boldsymbol{x}), s_3(\boldsymbol{x}) \in \sum$ such that the conditions (6)~(8) hold is a bilinear semi-definite programming problem. For simplicity, we denote this problem as *BSDP1*. So, we can use the bilinear semi-definite programming tools, e.g., **PENBMI** [8], to get a feasible solution on m, h, $s_0(\boldsymbol{x})$, $s_1(\boldsymbol{x})$, $s_2(\boldsymbol{x})$ and $V'(\boldsymbol{x})$. Due to under-approximations, the computed $V'(\boldsymbol{x})$ make the constraints (1), (2) and (3) hold.

Now, we try to compute an initial Lyapunov-like function $V(\boldsymbol{x})$, which is also a sum of squares polynomial of degree d, such that $\Omega = \{\boldsymbol{x} : V(\boldsymbol{x}) \le 1\}$ is a basin of attraction to TR, defined by $\{\boldsymbol{x} : p(\boldsymbol{x}) < 0\}$.

After similar under-approximations, we try to compute $V(\boldsymbol{x})$, m, h, $s_0'(\boldsymbol{x})$, $s_1'(\boldsymbol{x})$, $s_2'(\boldsymbol{x})$ and $s_3'(\boldsymbol{x})$ such that

$$m > 0, h > 0, \tag{9}$$

$$V(\boldsymbol{x}), s_0'(\boldsymbol{x}), s_1'(\boldsymbol{x}), s_2'(\boldsymbol{x}), s_3'(\boldsymbol{x}) \in \sum, \tag{10}$$

$$V(\boldsymbol{x}) - h\sum_{i=1}^{n} x_i^d \in \sum, \tag{11}$$

$$s_0'(\boldsymbol{x})p(\boldsymbol{x}) + (1 - V(\boldsymbol{x})) \in \sum, \text{ and} \tag{12}$$

$$-s_1'(\boldsymbol{x})(1 - V(\boldsymbol{x})) - s_2'(\boldsymbol{x})p(\boldsymbol{x}) - s_3'(\boldsymbol{x})\frac{d}{dt}V(\boldsymbol{x}) - m \in \sum. \tag{13}$$

Clearly, this is also a problem of bilinear semi-definite programming, denoted as *BSDP2*, and can be solved by **PENBMI** [8], which will return a feasible solution on $V(\boldsymbol{x})$, m, h, $s_0'(\boldsymbol{x})$, $s_1'(\boldsymbol{x})$, $s_2'(\boldsymbol{x})$ and $s_3'(\boldsymbol{x})$. Due to under-approximations, the computed $V(\boldsymbol{x})$ is a Lyapunov-like function such that $\Omega = \{\boldsymbol{x} : V(\boldsymbol{x}) \leq 1\}$ is an estimate of basin of attraction to *TR*.

4 Examples

In this section, five corresponding examples are presented and their corresponding basins of attraction to target regions are obtained by computing Lyapunov-like functions. Note that our implementation is based on the bilinear semi-definite problem solver **PENBMI** [8].

Example 1. Consider the following well-known Van der Pol equation:

$$\begin{cases} \dot{x}_1 = -x_2 \\ \dot{x}_2 = x_1 - (1 - x_1^2)x_2 \end{cases}$$

Clearly, it has an unstable limit cycle and the origin is a stable equilibrium. Moreover, the biggest region of attraction to the origin is the region enclosed by the limit cycle. However, the limit cycle cannot be explicitly represented and is usually visualized from the numerical solution.

Let $TR = \{\boldsymbol{x} : x_1^2 + x_2^2 < 0.01\}$ and $\epsilon = 0.001$.

1. Letting $d = 2$, we get $\Omega_2 = \{\boldsymbol{x} : 0.79405861286333x_1^2 - 0.62862184963676x_1x_2 + 0.45911896150590x_2^2 \leq 1\}$ as a basin of attraction to *TR*, whose boundary is depicted by green color in Fig. 2.
2. Letting $d = 6$, we get $\Omega_6 = \{\boldsymbol{x} : 0.000092646265x_1x_2^4 + 0.044231935525449x_1x_2 + 0.74507743769888 \times 10^{-3}x_1^2x_2 - 0.36262391827106x_2^4 + 0.15639138437193x_2^6 + 0.19919008040546x_1^2 + 0.16582352025455 \times 10^{-3}x_1x_2^2 + 0.35041765992891x_2^2 - 0.29996383177857 \times 10^{-3}x_1^3 - 0.00006355778238x_2^3 + 0.14528153665491x_1^3x_2 - 0.76326071243990x_1^2x_2^2 + 0.47126582058381x_1x_2^3 - 0.41790303496728 \times 10^{-3}x_1^4x_2 - 0.35991027002498x_1^4 + 0.00038881657835x_1^3x_2^2 - 0.44709824391353 \times 10^{-3}x_1^2x_2^3 - 0.23594343877042x_1^5x_2 + 0.44651553003767 \times 10^{-4}x_2^5 + 0.24310074159962x_1^6 + 0.94103618864951x_1^4x_2^2 - 1.0352745194725x_1^3x_2^3 + 0.28135622596667 \times 10^{-3}x_1^5 + 1.0174067778764x_1^2x_2^4 - 0.45450941605253x_1x_2^5 \leq 1\}$ as a basin of attraction to *TR*, whose boundary is depicted by red color in Fig. 2.

Example 2. The following system comes from [1] and is also studied in [26]:

$$\begin{cases} \dot{x}_1 = -0.42x_1 - 1.05x_2 - 2.3x_1^2 - 0.5x_1x_2 - x_1^3 \\ \dot{x}_2 = 1.98x_1 + x_1x_2 \end{cases}$$

Let $\epsilon = 0.001$ and $TR = \{\boldsymbol{x} : x_1^2 + x_2^2 \leq 0.01\}$. The basin of attraction to *TR* is unbounded, but not $\mathbb{R}^2$.

Letting $d = 6$, we get $\Omega_6 = \{\boldsymbol{x} : -0.045451724128868x_1x_2 + 2.17062562455x_1^2x_2 + 0.096773071723153x_1x_2^2 + 0.33969734015x_2^2 + 0.885166369842x_1^2 - 1.03307547539x_1^3 +$

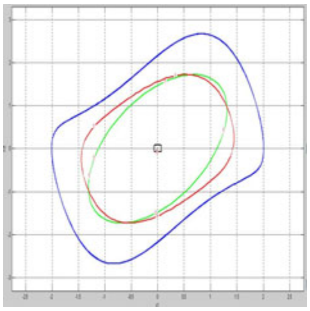


Fig. 2. Basins of Attraction for Example 1
(black curve – ∂TR; green curve – $\partial\Omega_2$; red curve – $\partial\Omega_6$; blue curve – the limit cycle)

$0.654218150175x_2^3 - 1.71896808483x_1^3x_2 + 2.56750235247x_1^2x_2^2 + 0.04702636437x_1x_2^3 + 1.090468414511x_1^4x_2 - 1.036336482374x_1^3x_2^2 + 1.3772219959x_1^2x_2^3 - 0.0634812976x_1x_2^4 - 0.274100822457x_1^5x_2 + 0.24652334591x_1^4x_2^2 - 0.2300072821x_1^3x_2^3 + 0.27041132416x_1^2x_2^4 + 1.3409958413730x_1^4 + 0.40816098435769x_2^4 - 0.9675630392x_1^5 + 0.1450156781311 4x_2^5 + 0.25864379007912x_1^6 + 0.056657925718930x_2^6 - 0.022396371772690x_1x_2^5 \leq 1\}$ as a basin of attraction to *TR*, whose boundary is depicted by red color in Fig. 3.

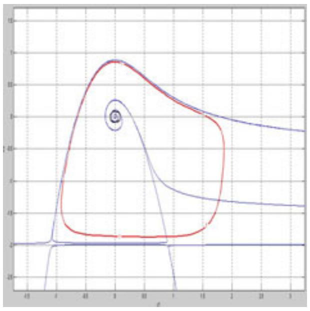


Fig. 3. A Basin of Attraction for Example 2
(black curve – ∂TR; red curve – $\partial\Omega_6$; blue curves – trajectories)

Example 3. We consider the following system:

$$\begin{cases} \dot{x}_1 = x_2 - x_3 + (1 - x_1^2 - x_2^2 - x_3^2)(x_1 - 2x_1x_3) \\ \dot{x}_2 = -x_1 + x_3 + (1 - x_1^2 - x_2^2 - x_3^2)x_2 \\ \dot{x}_3 = x_1 - x_2 + (1 - x_1^2 - x_2^2 - x_3^2)x_3 \end{cases}$$

Obviously, $(0, 0, 0)$ and $(1, 0, 1)$ are equilibria and for any arbitrary but fixed constant $c \in (\sqrt{3}, \sqrt{3})$, the intersection of $x_1 + x_2 + x_3 = c$ and $x_1^2 + x_2^2 + x_3^2 = 1$ is a cycle.

Let $TR = \{\boldsymbol{x} : x_1^2 + x_2^2 + x_3^2 \leq 1.01\}$. Clearly, *TR* contain cycles. In addition, the basin of attraction to *TR* is unbounded, but not all of $\mathbb{R}^2$.

Let $\epsilon = 0.001$, $d = 2$, $d_{s_0} = 0, d_{s_1} = 4, d_{s_2} = 2, d_{s_3} = 0$. We get $\Omega_2 = \{\boldsymbol{x} : 0.55660892919976x_1^2 + 0.20677987929356x_1x_2 + 0.0470439488x_1x_3 + 0.25267039x_2^2 - 0.13411242278944x_2x_3 + 0.44554905459094x_3^2 \leq 1\}$ as a basin of attraction to TR, whose boundary is depicted by green color in Fig. 4. Moreover, the intersection of Ω_2 and the x_1-x_2 space is depicted in Fig. 5.

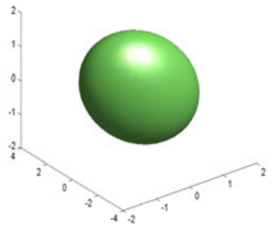


Fig. 4. A Basin of Attraction for Example 3 (green curve – $\partial\Omega_2$)

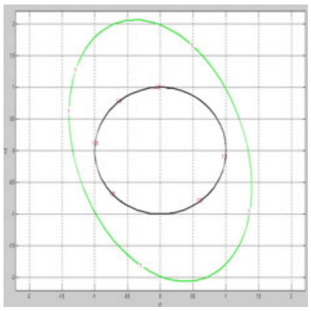


Fig. 5. Projection of a Basin of Attraction for Example 3 (black curve – boundary of the intersection of TR and the $x_1 - x_2$ space; green curve – boundary of the intersection of Ω_2 and the $x_1 - x_2$ space)

Note that all the above computations were performed on a Hasee notebook of Core II Duo, 2.00 GHz with 2 GB RAM. The computation times, the degrees d, the individual degrees for multipliers s_0, s_1, s_2 and s_3 (i.e., $d_{s_0}, d_{s_1}, d_{s_2}, d_{s_3}$), and the required iterative steps for Algorithm 1 are listed as a summary in Table 1.

Note that our previous interval based branch-and-relax algorithm [20] can be used to provide an initial basin of attraction to target region for our iterative computation in Algorithm 1. However, all the methods for reasoning about classical stability do not directly allow reasoning about a basin of attraction to a target region.

Table 1. Computation Times and so on for bilinear semi-definite programming

Example	d	d_{s_0}	d_{s_1}	d_{s_2}	d_{s_3}	CPU time(s)	iterative steps
1	2	2	2	0	0	7.094	5
1	6	2	4	2	0	1413.985	7
2	6	2	6	2	0	3061.438	6
3	2	0	4	2	0	517.64	14

5 Conclusion

In this paper, we present a sum of squares programming based approach for computing a basin of attraction to a target region as large as possible by iteratively searching for Lyapunov-like functions with sums of squares formula. Such a search can be realized by solving bilinear semi-definite problems. We implement our algorithm and test it on some interesting examples. The computation results show the usefulness of our method.

We will further optimize our algorithms and then generalize them for analyzing stability of hybrid systems [17,19,31].

Acknowledgement. The authors thank the developers of the toolbox **PENBMI**.

References

1. Chesi, G., Garulli, A., Tesi, A., Vicino, A.: LMI-based Computation of Optimal Quadaraic Lyapunov Function for Odd Polynomial Systems. Int. J. Robust and Nonlinear Control 15, 35–49 (2005)
2. Chiang, H.D., Thorp, J.S.: Stability regions of nonlinear dynamical systems: A constructive methodology. IEEE Transactions on Automatic Control 34(12), 1229–1241 (1989)
3. Forsman, K.: Construction of Lyapunov functions using Gröbner bases. In: Proceedings of the 30th IEEE Conference on Decision and Control, pp. 798–799 (1991)
4. Genesio, R., Tartaglia, M., Vicino, A.: On the estimation of asymptotic stability regions: State of the art and new proposals. IEEE Trans. on Automatic Control 30(8), 747–755 (1985)
5. Jarvis-Wloszek, Z.: Lyapunov based analysis and controller synthesis for polynomial systems using sum-of-squares optimization, Ph.D. Dissertation, University of California (2003)
6. John Koo, T., Su, H.: A Computational Approach for Estimating Stability Regions. In: Proc. of the IEEE Conference on Computer Aided Control Systems Design, pp. 62–68 (2006)
7. Khalil, H.C.: Nonlinear Systems. Prentice Hall, Englewood Cliffs (2002)
8. Kočvara, M., Stingl, M.: PENBMI Users Guide (Version 2.1) (2005), `http://www.penopt.com`
9. Lakshmikantham, V., Leela, S., Martynyuk, A.A.: Practical Stability of Nonlinear System. World Scientific, Singapore (1990)
10. Levin, A.: An analytical method of estimating the domain of attraction for polynomial differential equations. IEEE Transactions on Automatic Control 39(12), 2471–2475 (1994)
11. Mitchell, I.M., Bayen, A.M., Tomlin, C.J.: A Time-Dependent HamiltonCJacobi Formulation of Reachable Sets for Continuous Dynamic Games. IEEE Transactions on Automatic Control 50(7), 947–957 (2005)
12. Noldus, E., Spriet, J., Verriest, E., Van Cauwenberghe, A.: A New Lyapunov Technique for Stability Analysis of Chemical Reactors. Automatica 10, 675–680 (1974)

13. Pai, M.A.: Power System Stability. North-Holland, Amsterdam (1981)
14. Papachristodoulou, A., Prajna, S.: On the construction of Lyapunov functions using the sum of squares decomposition. In: Proceedings of the 41st IEEE Conference on Decision and Control, pp. 3482–3487 (2002)
15. Parrilo, P., Lall, S.: Semidefinite programming relaxations and algebraic optimization in control. Eur. J. Control 9, 307–321 (2003)
16. Peterfreund, N., Baram, Y.: Convergence analysis of nonlinear dynamical systems by nested Lyapunov functions. IEEE Trans. on Automatic Control 43(8), 1179–1184 (1998)
17. Podelski, A., Wagner, S.: Region stability proofs for hybrid systems. In: Raskin, J.-F., Thiagarajan, P.S. (eds.) FORMATS 2007. LNCS, vol. 4763, pp. 320–335. Springer, Heidelberg (2007)
18. Shields, D.N., Storey, C.: The behaviour of optimal Lyapunov functions. Int. J. Control 21, 561–573 (1975)
19. Ratschan, S., She, Z.: Safety Verification of Hybrid Systems by Constraint Propagation-based Abstraction Refinement. ACM Transactions on Embedded Computing Systems 6(1), Article No. 8, 1–23 (2007)
20. Ratschan, S., She, Z.: Provding a Basin of Attraction to A Target Region of Polynomial Systems by Computation of Lyapunov-like Functions. SIAM Journal on Control and Optimization 48(7), 4377–4394 (2010)
21. Ryali, V., Moudgalya, K.M.: Practical Stability Analysis of Uncertain Nonlinear Systems. In: National Conference on Control and Dynamic Systems, IIT Bombay, pp. 27–29 (2005)
22. She, Z., Xia, B., Xiao, R., Zheng, Z.: A semi-algebraic approach for asymptotic stability analysis. Nonlinear Analysis: Hybrid System 3(4), 588–596 (2009)
23. She, Z., Zheng, Z.: Condition number based complexity estimate for computing local extrema. J. of Computational and Applied Mathematics 230(1), 233–242 (2009)
24. She, Z., Xia, B., Zheng, Z.: Condition number based complexity estimate for solving polynomial systems. J. of Computational and Applied Mathematics 235(8), 2670–2678 (2011)
25. She, Z., Xue, B., Zheng, Z.: Algebraic Analysis on Asymptotic Stability of Continuous Dynamical Systems. In: Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation (2011)
26. Tan, W., Packard, A.: Stability region analysis using polynomial and composite polynomial Lyapunov functions and sum-of-squares programming. IEEE Transactions on Automatic Control 53(2), 565–571 (2008)
27. Tarski, A.: A Decision Method for Elementary Algebra and Geometry. University of California Press, Berkeley (1951)
28. Vincent, T., Grantham, W.: Nonlinear and Optimal Control Systems. Wiley-Interscience, New York (1997)
29. Wang, T., Lall, S., West, M.: Polynomial level-set methods for nonlinear dynamical systems analysis. In: Proc. of Allerton Conf. on Communication, Control and Computing (2005)
30. Weiss, L.: Converse theorems for finite time stability. SAIM Journal on Applied Mathematics, 1319–1324 (1968)
31. Xu, X., Zhai, G.: Practical Stability and Stabilization of Hybrid and Switched System. IEEE Trans. Automat. Control 50(11), 1897–1903 (2005)

Symbolic-Numeric Solution of Ill-Conditioned Polynomial Systems (Survey Talk Overview) (Invited Talk)

Agnes Szanto[⋆]

North Carolina State University
aszanto@ncsu.edu

Abstract. This is a survey talk about some recent symbolic-numeric techniques to solve ill-conditioned multivariate polynomial systems. In particular, we will concentrate on systems that are over-constrained or have roots with multiplicities, and are given with inexact coefficients. First I give some theoretical background on polynomial systems with inexact coefficients, ill-posed and ill-conditioned problems, and on the objectives when trying to solve these systems. Next, I will describe a family of iterative techniques which, for a given inexact system of polynomials and given root structure, computes the nearest system which has roots with the given structure. Finally, I present a global method to solve multivariate polynomial systems which are near root multiplicities and thus have clusters of roots. The method computes a new system which is "square-free", i.e. it has exactly one root in each cluster near the arithmetic mean of the cluster. This method is global in the sense that it works simultaneously for all clusters.

The results presented in the talk are joint work with Itnuit Janovitz-Freireich, Bernard Mourrain, Scott Pope, Lajos Rónyai, Olivier Ruatta, and Mark Sciabica.

Overview

In recent years there has been intensive research on extending the applicability of symbolic and numerical methods to handle problems which are given with limited accuracy and which were traditionally called "ill-conditioned". The integration of numerical and symbolic techniques resulted in a remarkable progress in the applicability, versatility, robustness and efficiency of the algorithms for the solution of problems such as approximate GCD, approximate polynomial factorization, solution of under- and over-constrained approximate polynomial systems, approximation of the solution of differential equations with Lie-symmetries. This talk will concentrate on describing recent techniques for:

– The solution of near-consistent over-constrained polynomial systems, i.e. systems with more equations than unknowns such that the coefficients are small

[⋆] The author was partially supported by NSF grant CCR-0347506.

perturbations of systems which have common roots over the complex numbers. Note that over-constrained multivariate systems generically do not have common roots;
– The solution of multivariate polynomial systems near root multiplicities. The set of complex roots of these systems form finitely many clusters with small radius.

The main motivation for studying these two types of systems come from the observation that they arise frequently in many important applications areas such as geometric modeling, computer vision, robotics, etc.

Before presenting the methods of solution, we give some theoretical framework, including precise definitions of polynomials with inexact coefficients, forward and backward error, ill-posed and ill-conditioned problems, and certification of the solution.

All symbolic-numeric algorithms that solve ill-conditioned polynomial systems contain the following two components to some degree:

1. they adapt symbolic techniques – such as Gröbner bases, border basis, or resultant computation – to the inexact case,
2. they reduce the problem to classical numerical methods such as numerical eigenvalue computation or Newton or Gauss-Newton method to locally iterate roots of non-linear functions.

The first family of methods we present in the talk falls in the second category: these are Gauss-Newton type iterative methods that compute the nearest system to the input which have some prescribed root structure. We discuss in detail the case when the input is near an over-constrained system with k common roots [10], but the same ideas also work for systems near root multiplicities [8], and have many other extensions [1]. Our main tool is the multivariate interpolation method developed in [7] and [9]. Using interpolation we were able to generalize to the multivariate case univariate results of [12] and [11] which use polynomial division. By close inspection of the interpolation method, we can express the distance of the input system from the set of systems with a given root multiplicity structure as the optimal value of a rational function of the roots. In the univariate case we show the connection of our method to the optimization problem formulated by Karmarkar and Lakshman in [6] for the nearest GCD. In the multivariate case we generalize the expressions of Karmarkar and Lakshman, and give component-wise iteration functions to compute the optimum.

In the second part of the talk I describe results that reduce the problem of solving polynomial systems near root multiplicities to well-conditioned numerical eigenvalue computation [2,3,4,5]. We present a method, based on the adaptation of a suitable symbolic algorithm for the computation of the radical of an ideal, to compute the "approximate radical" of a zero dimensional ideal I in $\mathbb{C}[x_1, \ldots, x_m]$ which has zero clusters. The approximate radical ideal has exactly one root in each cluster for sufficiently small clusters. The method is "global" in the sense that it does not require any local approximation of the zero clusters: it reduces the problem to the computation of the numerical nullspace of the so called "matrix of traces", a matrix computable from the coefficients of the generating polynomials of I. To compute the numerical nullspace of the matrix of traces we propose to use Gauss elimination with pivoting, and we prove that if I has k distinct zero clusters each of radius at most ε in the ∞-norm, then k steps of Gauss elimination on the matrix of traces yields a submatrix with all entries asymptotically equal to ε^2. We also prove that the computed approximate radical has one root in each cluster with coordinates which are the arithmetic mean of the cluster, up to an error term asymptotically equal to ε^2. We obtain the coordinates of the roots of the approximate radical as eigenvalues of the so called "multiplication matrices", computed from maximal non-singular submatrices of the matrix of traces. Finally, we present some simple symbolic techniques to compute the matrix of traces from the coefficients of the input polynomials.

References

1. S. E. Hutton, Exact Sums-of-Squares Certificates in Numeric Algebraic Geometry, PhD thesis, North Carolina State University (2011)
2. Janovitz-Freireich, I., Rónyai, L., Szántó, Á.: Approximate radical of ideals with clusters of roots. In: ISSAC 2006, pp. 146–153. ACM, New York (2006)
3. Janovitz-Freireich, I., Rónyai, L., Szántó, Á.: Approximate radical for clusters: a global approach using Gaussian elimination or SVD. Math. Comput. Sci. 1, 393–425 (2007)
4. Janovitz-Freireich, I., Szántó, Á., Mourrain, B., Rónyai, L.: Moment matrices, trace matrices and the radical of ideals. In: ISSAC 2008, pp. 125–132. ACM, New York (2008)
5. Janovitz-Freireich, I., Szántó, Á., Mourrain, B., Rónyai, L.: On the Computation of Matrices of Traces and Radicals of Ideals. Submitted to Journal of Symbolic Computation (2009); arXiv:0901.2778
6. Karmarkar, N.K., Lakshman, Y.N.: On approximate GCDs of univariate polynomials. Journal of Symbolic Computation 26, 653–666 (1998)
7. Mourrain, B.: Isolated points, duality and residues, J. Pure Appl. Algebra 117/118, 469–493 (1997); Algorithms for algebra, Eindhoven (1996)
8. Pope, S., Szanto, A.: Nearest multivariate system with given root multiplicities. Journal of Symbolic Computation, 606–625 (2009)
9. Ruatta, O.: Dualité algébrique, structures et applications, PhD thesis, Université de la Méditérranée (2002)
10. Ruatta, O., Sciabica, M., Szanto, A.: Over-constrained Weierstrass iteration and the nearest consistent system (2009) (accepted for publication)
11. Zeng, Z.: Computing multiple roots of inexact polynomials. Mathematics of Computation 74, 869–903 (2005)
12. Zhi, L., Wu, W.: Nearest singular polynomials, J. Symbolic Comput. 26, 667–675 (1998); Symbolic numeric algebra for polynomials

Symbolic-Manipulation Constructions of Hilbert-Space Metrics in Quantum Mechanics

Miloslav Znojil

Nuclear Physics Institute ASCR, 250 68 Řež, Czech Republic
znojil@ujf.cas.cz
http://gemma.ujf.cas.cz

Abstract. The recently formulated quantum-mechanics problem of the determination of the Hilbert-space metric Θ which renders a given Hamiltonian H self-adjoint is addressed. Via an exactly solvable example of the so called Gegenbauerian quantum-lattice oscillator it is demonstrated that the construction (basically, the solution of the so called Dieudonné's operator equation) and analysis of suitable $\Theta = \Theta(H)$ (i.e., the determination of their domain's "exceptional-point" boundary) may enormously be facilitated via symbolic algebraic manipulations and via the MAPLE-supported numerics and graphics.

1 Introduction

In the series of papers [1] - [8], we interpreted the Dieudonné's quasi-Hermiticity relation [9]

$$H^\dagger \Theta = \Theta H \tag{1}$$

as an operator equation which connects a given, "input" quantum Hamiltonian H with an unknown, "output" operator Θ called the Hilbert-space metric of the quantum system in question [10]. In all of these papers we felt addressed by the underlying physics (i.e., by quantum mechanics in its form described, say, in Refs. [11,12]). We did not pay too much attention to the description of the underlying constructive mathematics. In our present paper, we intend to fill this gap by redirecting our attention to the computer-assisted symbolic-manipulation background of our results. We will point out that although none of our constructions required a particularly sophisticated code, all of them still offered an efficient substitute for the laborious hand-made calculations and/or for the difficult numerical analyses.

With this purpose in mind we shall employ here just very elementary illustrative models. In particular, we shall assume that both of the operators H and Θ in Eq. (1) are defined in a vector space $\mathcal{V}$ with the Dirac-ket elements $|\psi\rangle \in \mathcal{V}$. For the sake of simplicity we shall further assume that $\dim \mathcal{V} = N < \infty$. In such a setting one may select various toy-model matrices $H^{(N)}$ and construct the eligible metrics Θ. We shall choose the Gegenbauerian quantum $N-$site lattices and study $H = H^{(N)}(a)$ and $\Theta = \Theta^{(N)}(a)$ of Ref. [2] (cf. section 2). The reasons of a facilitated algebraic tractability of Eq. (1) will be clarified in section 3, with some complementary numerical aspects mentioned in section 4.

2 Gegenbauerian Quantum $N-$site Lattices

The general Hermitian conjugation operation as prescribed, say, by Eq. (16) of Ref. [12] must be compatible with the principles of Quantum Mechanics. This means that our choice of the metric Θ must guarantee the Hermiticity of the observables (i.e., in our present paper, just of the Hamiltonian H) with respect to this conjugation [10]. Such a requirement implies the necessity of the validity of the above-mentioned relation (1).

The latter relation will be called here, for the sake of brevity, Dieudonné equation. As long as this is the matrix equation, it seems to be an overdetermined constraint. Its N^2 items have to be satisfied by the mere $N(N+1)/2$ independent matrix elements of the general N by N real and symmetric matrix Θ with positive eigenvalues. Via a deeper study of an illustrative example taken from Ref. [2] we intend to demonstrate that the situation is much more user friendly.

First of all, the Hermitian conjugation may be perceived as a symmetry of Eq. (1) so that just its upper-triangular nontrivial subset remains relevant. This implies that the whole set of equations is in fact inderdeterminate. *A priori*, the independent solutions Θ will form an $N-$parametric family. This observation is compatible with the explicit constructive results published in Ref. [2].

Secondly, the message delivered by Ref. [2] was aimed at the physics audience. Our present study will complement these results by their more systematic derivation and by the more explicit explanation of their formal structure. Keeping this aim in mind we shall consider the $N-$dimensional matrix Schrödinger equation

$$H^{(N)}\,|\psi_n^{(N)}\rangle = E_n^{(N)}\,|\psi_n^{(N)}\rangle \tag{2}$$

with the prescribed bound-state eigenvectors

$$|\psi_n^{(N)}\rangle = \begin{pmatrix} \langle 0|\psi_n^{(N)}\rangle = G(0,a,E_n) \\ \langle 1|\psi_n^{(N)}\rangle = G(1,a,E_n) \\ \vdots \\ \langle N-1|\psi_n^{(N)}\rangle = G(N-1,a,E_n) \end{pmatrix} \tag{3}$$

where, in the notation of MAPLE [13], the symbol $G(n,a,x)$ denotes the nth Gegenbauer polynomial $G(n,a,x)$ equal to polynomial $C_n^a(x)$ in the notation of Ref. [14] or to $C_n^{(a)}(x)$ according to Ref. [15]. Under such an assumption, naturally, the explicit form of the related Hamiltonian is the tridiagonal array

$$\left[\begin{array}{cccccc} 0 & 1/2\,a^{-1} & 0 & 0 & \dots & 0 \\ 2\,\frac{a}{2\,a+2} & 0 & (2\,a+2)^{-1} & 0 & \dots & \vdots \\ 0 & \frac{2\,a+1}{2\,a+4} & 0 & (2\,a+4)^{-1} & \ddots & 0 \\ 0 & 0 & \frac{2\,a+2}{2\,a+6} & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & 0 & (2\,a+2N-4)^{-1} \\ 0 & \dots & 0 & 0 & \frac{2\,a+N-1}{2\,a+2N-2} & 0 \end{array}\right] \qquad (4)$$

which defines the manifestly asymmetric N by N matrix $H^{(N)}$.

The validity of such an assignment is equivalent to the standard three-term recurrences for the Gegenbauer polynomials while the N by N matrix truncation is equivalent to the implicit-equation identification of the real and non-degenerate spectrum $\sigma(H^{(N)}) \equiv \{E_n\}$ of the bound-state energies with the roots of the Nth Gegenbauer polynomial,

$$G(N, a, E_n) = 0\,. \qquad (5)$$

Naturally, such a secular equation may be considered solvable with an arbitrary numerical precision. The only nontrivial task represented by the complete description of the model will lie in the choice of a metric Θ compatible with Dieudonné Eq. (1). The method has not been described in Ref. [2] because it just consisted in the brute-force insertion of a general real and symmetric ansatz for $\Theta^{(N)}$ and in the subsequent trial and error analysis of Eq. (1) after its insertion.

In the metric-construction problem, the key difficulties are twofold. Firstly, the ansatz for $\Theta^{(N)}$ contains too many (i.e., $N(N+1)/2$) unknowns, and there are no criteria for the clarification which ones of them should be selected as the "optimal" independent set. Even at the very small integers N, preliminary MAPLE-based brute-force algebraic symbolic-manipulation-solution experiments starting from Eq. (1) and from a few randomly selected $N-$plets of the tentative independent matrix elements of Θ generated just the obscure many-page results for all of the $N-$ and $a-$dependent matrix elements of $\Theta(H)$. The second difficulty emerged with the necessity of the parameter-range-specifying guarantee of the obligatorily positive-definite nature of any resulting N by N matrix candidate $\Theta_\alpha(H)$ for the metric (characterized or distinguished, in general, by a suitable multiindex α).

The core of the success (i.e., of the resolution to both of these parallel algebraic-numerical difficulties) has been revealed to lie in an interactive and iterative approach to both of the problems. In more detail this approach is to be described in what follows.

3 The Dieudonné's Equation

In the light of the old Dieudonné's idea [9] it seems interesting to replace the current textbook Hermiticity property $H = H^\dagger$ of the current selfadjoint Hamiltonians in quantum mechanics by the weaker assumption containing a nontrivial "metric" $\Theta \neq I$ [10]. In this context relation (1) guarantees the reality of the energies provided only that we require that the operator $\Theta = \Theta^\dagger$ is, roughly speaking [16], positive and invertible, i.e., tractable as a metric in the Hilbert space of states $\mathcal{H}^{(S)}$ where the superscript means "standard".

The recent growth of popularity and applicability of the quantum models requiring the unusual metrics $\Theta = \Theta^{(S)} > I$ may be found reviewed, e.g., in Refs. [11,17,18,19]. In our present paper we circumvent a number of technicalities by studying just the quantum models defined in finite-dimensional Hilbert spaces. Thus, we may identify $\mathcal{H}^{(F)} \equiv \mathbb{C}^N$. Moreover, for the sake of definiteness, we shall only pay attention to the models where the Hamiltonian matrices possess the tridiagonal real-matrix form.

3.1 An Interactive Algebraic-solution Technique

As an illustrative example we shall use the Gegenbauerian quantum lattice model of Ref. [2] described in the preceding section. In fact, in Ref. [2] we described the results showing the feasibility of the brute-force construction of the complete family of the metrics $\Theta(H)$ admitted by the Dieudonné's linear algebraic constraints (1). In our present continuation of this effort we intend to provide a deeper insight in the problem explaining the reasons why our construction of the metrics appeared to be so successful.

We have to admit that with our very specific choice of the Gegenbauerian model in Ref. [2] we were unexpectedly fortunate. This fact may be demonstrated, say, by the recollection of the similar constructive attempts based on a different choice of the N by N "input" Hamiltonian as reported in Ref. [5]. After the construction of $\Theta(H)$ at the dimension as low as $N = 4$ it has been argued there that the construction at the very next $N = 5$ appeared almost prohibitively complicated. This is really in contrast with the results of Ref. [2] which proved *valid at any integer* N.

The core of the dimension-independent universality of the above-mentioned Gegenbauerian result may be seen in the combination of the extremely simple bidiagonal form of the Hamiltonians $H^{(N)}(a)$ with the comparably simple $a-$dependence of its matrix elements. The relevance of both of these ingredients becomes obvious when we recall the explicit $N = 4$ sample of the Hamiltonian

$$H^{(4)}(a) = \begin{bmatrix} 0 & (2\,a)^{-1} & 0 & 0 \\ \frac{2\,a}{2\,a+2} & 0 & (2\,a+2)^{-1} & 0 \\ 0 & \frac{2\,a+1}{2\,a+4} & 0 & (2\,a+4)^{-1} \\ 0 & 0 & \frac{2\,a+2}{2\,a+6} & 0 \end{bmatrix}$$

together with the general ansatz for the metric

$$\Theta^{(4)}(a) = \begin{bmatrix} k & b & c & d \\ b & f & g & h \\ c & g & m & n \\ d & h & n & j \end{bmatrix} .$$

In such a setting we may study the 16-plet of the resulting relations, out of which Nr. 1, Nr. 6, Nr. 11 and Nr. 16 (i.e., diagonal items) remain trivial while the off-diagonal items form an antisymmetric matrix. Out of the remaining six independent items (say, Nr. 2, 3, 4, 7, 8, and 12) there is just one (viz., Nr. 4) which involves just two unknown quantities (viz., h and c). This leads to the decision of taking d, c, b, and k as independent parameters and of eliminating, in the first step, h via item Nr. 4,

$$h = \frac{c\,(a+1)}{2\,(a+2)\,a} .$$

The inspection of the new set of items reveals that the simplest one is now just Nr. 3 which defines g as a function of b and d, with the next-step Nr. 8 defining n as a function of d and (newly known) g. We are left with the three items Nr. 2, 7 and 12 which couple (k, f), (f, m) and (m, j), respectively. As long as we decided to use k as the fourth unconstrained parameter this means that in the same order we now define, step by step, the missing items f, m and, ultimately, j. The result is complete yielding

$$h = 1/2\,\frac{c\,(a+1)}{(a+2)\,a}$$

$$g = 1/2\,\frac{ba + 3\,b + 2\,da^2 + 4\,da + 2\,d}{(a+3)\,a}$$

$$n = 1/2\,\frac{-6\,da - 10\,d + ba + 3\,b}{(a+3)\,a\,(2\,a+1)}$$

$$f = 1/2\,\frac{\left(2\,ca^2 + k\,a + ca + 2\,k\right)(a+1)}{(a+2)\,a^2}$$

$$m = 1/2\,\frac{2\,ca^3 + ca^2 - 7\,ca + k\,a^2 + 5\,k\,a + 6\,k}{(a+3)\,a^2\,(2\,a+1)}$$

$$j = -1/4\,\frac{6\,ca^2 + 10\,ca - k\,a^2 - 5\,k\,a - 6\,k}{a^2\,(2\,a+1)\,(a+2)\,(a+1)} .$$

We may conclude that using MAPLE one obtains the necessary answers quickly. The same applies to the models at higher N.

3.2 The Case of General *N*

After the above-explained heuristic exercise we are prepared to consider the real and symmetric general ansatz for the metric

$$\Theta^{(N)}(a)=\left[\begin{array}{cccc}\theta_{11} & \theta_{12} & \dots & \theta_{1,N}\\ \theta_{12} & \theta_{22} & \ddots & \vdots\\ \vdots & \ddots & \ddots & \theta_{N-1,N}\\ \theta_{1,N} & \dots & \theta_{N-1,N} & \theta_{NN}\end{array}\right]. \tag{6}$$

and prove the general result.

Definition 1. *At any $N \geq 2$ the insertion of the N by N Hamiltonian $H = H^{(N)}(a)$ given by Eq. (4) and of the general real and symmetric matrix ansatz (6) for the metric $\Theta = \Theta^{(N)}(a)$ defines the N by N matrix array (1) of the linear Dieudonné equations $\mathcal{M}_{i,j} = 0$. Its* **ordered version** *has the form $r_\alpha = 0$ with*

$$\begin{aligned} r_1 &= \mathcal{M}_{1,N}\,,\\ r_2 = \mathcal{M}_{1,N-1}\,, r_3 &= \mathcal{M}_{2,N}\,,\\ r_4 = \mathcal{M}_{1,N-2}\,, r_5 = \mathcal{M}_{2,N-1}\,, r_6 &= \mathcal{M}_{3,N}\,,\\ &\dots\,,\\ r_{(N-1)(N-2))/2+1} = \mathcal{M}_{1,2}\,,\dots\,, r_{N(N-1)/2} &= \mathcal{M}_{N-1,N}\,. \end{aligned} \tag{7}$$

Theorem 1. *In terms of the freely variable $N-$plet of the real initial parameters Θ_{1j}, $j = 1,2,\dots,N$ the Dieudonné equation in its ordered version (7) defines, step by step, the respective "missing" matrix elements*

$$\begin{aligned} &\Theta_{2,N}\,,\\ \Theta_{2,N-1}\,,&\Theta_{3,N}\,,\\ &\dots\,,\\ \Theta_{2,2}\,,\Theta_{3,3}\,,\Theta_{4,4}\,,\dots\,,&\Theta_{N,N} \end{aligned} \tag{8}$$

in recurrent manner.

Proof. Once we revealed the diagonal-wise-arranged recurrent pattern it is easy and entirely straightforward to verify its validity by the corresponding trivial rearrangement of the two matrix multiplications in Eq. (1).

Remark 1. The diagonal-wise recurrent nature of Eq. (1) given by Theorem 1 has only been revealed by the *post factum* inspection of the results of Ref. [2].

4 The Formulation of Quantum Theory Using an *Ad Hoc* Triplet of Hilbert Spaces

4.1 The Positive Definiteness of the Metric

During the recent years we are witnessing the remarkable growth of popularity of the building of quantum models which combine the "false" non-Hermiticity

$H \neq H^\dagger$ of the comparatively elementary Hamiltonian acting in a "friendly" Hilbert space $\mathcal{H}^{(F)}$ with the simultaneous "sophisticated" Hermiticity $H = H^\ddagger$ of the same Hamiltonian in another, less usual, amended, "standard" Hilbert space $\mathcal{H}^{(S)}$ endowed with a nontrivial metric $\Theta = \Theta^{(S)} \neq I$.

The key to the consistency of such a formulation of quantum theory lies in the correct choice of the latter operator [10]. In Ref. [12] we summarized some of the mathematical features of such an approach to the phenomenological quantum model-building. We pointed out there that one of the main difficulties often lies in the guarantee of the positive definiteness of the metric.

The essence of the problem has been made entirely transparent and obvious when we eliminate the unfortunate confusion caused by the traditional terminology. Our key point was that in fact, the innovated formalism never leaves the abstract theoretical framework of quantum theory. Just a few new mathematical tricks (like, typically, an unusual, non-unitary generalization of the most common Fourier transformation) are being added to the traditional textbook recipes.

In particular, the Dieudonné-equation constraint imposed on a Hamiltonian H is in fact equivalent to the manifested Hermiticity of its isospectral image

$$\mathfrak{h} = \Omega \, H \, \Omega^{-1} = \mathfrak{h}^\dagger \tag{9}$$

In principle (though not always in practice), the latter operator is defined as acting in another physical Hilbert space denoted by the third symbol $\mathcal{H}^{(P)}$. In this space the traditional, trivial metric $\Theta^{(P)} = I$ is used.

In such a notation [12], both the Hilbert spaces $\mathcal{H}^{(P)}$ and $\mathcal{H}^{(S)}$ may be perceived as unitarily equivalent. We may deduce

$$\mathfrak{h}^\dagger = \left(\Omega^{-1}\right)^\dagger \, H^\dagger \, \Omega^\dagger \,. \tag{10}$$

After we abbreviate $\Omega^\dagger \Omega := \Theta$ we end up with the Dieudonné's relation (1) as well as with the necessity of the positivity of the metric.

4.2 The Gegenbauerian Illustrative Example

For our real and finite-dimensional Gegenbauerian Hamiltonians $H = H^{(N)}(a)$ which are given in advance, the Dieudonné's relation (1) forms the set of N^2 constraints imposed upon the $[N(N+1)/2]-$plet of the unknown real matrix elements of the metric matrix $\Theta = \Theta^\dagger > 0$. In papers [1] - [8], we proposed the non-numerical, symbolic-manipulation approach to constructions of a complete solution of this linear algebraic system. What remains for us to construct is the appropriate domain $\mathcal{D}$ of free parameters for which these candidates for the metric remain positive definite, i.e., truly eligible in the appropriate definitions of the generalized Hermitian conjugation and/or of the appropriate Hilbert-space inner product.

In Ref. [2] we discussed a few specific examples of candidates $\Theta^{(N)}(a)$ for the Gegenbauerian metrics. We revealed that such a study leads to a purely numerical description supporting the hypothesis that the domains $\mathcal{D}^{(N)}(a)$ change

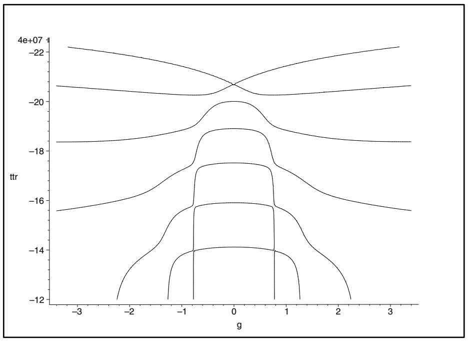


Fig. 1. Seven eigenvalues $p = p(g)$ of metric $\Theta_g^{(7)}(1)$

"smoothly" with N and lead to the non-empty and sufficiently large limiting domains $\lim_{N\to\infty} \mathcal{D}^{(N)}(a) = \mathcal{D}^{(\infty)}(a) \neq \emptyset$.

A comparatively weak N-dependence characterizes even the domains $\mathcal{D}^{(N)}(a)$ at small $N \gtrapprox 5$. The direct evaluation of the eigenvalues of $\Theta^{(N)}(a)$ (i.e., the more precise determination of the boundaries $\partial\mathcal{D}^{(N)}(a)$) only suffers of the errors caused by the multiple-scale nature of these eigenvalues.

In Ref. [2] we were only able to provide a transparent graphical illustration of the free-parameter-dependence of the spectrum of selected $\Theta^{(N)}(a)$s at the very first nontrivial dimension $N = 3$. In the context of programming in MAPLE (offering an adaptable floating-point precision arithmetics) the remedy is easy. One may take, say, the $N = 3$ toy metric of Ref. [2],

$$\Theta_g^{(3)}(a) = \begin{bmatrix} 2\,a^2 & 2\,ga & 0 \\ 2\,ga & a+1 & g \\ 0 & g & \frac{a+2}{2\,a+1} \end{bmatrix}$$

and represent the triplet of eigenvalues $p_j(g)$, $j = 1, 2, 3$ (as sampled in Figure Nr. 3 of *loc. cit.* at $a = 1$) in logarithmic scale yielding, say, the adapted $a = 1$ secular equation

$$\det \begin{bmatrix} 2 - e^{-ttr-20} & 2\,g & 0 \\ 2\,g & 2 - e^{-ttr-20} & g \\ 0 & g & 1 - e^{-ttr-20} \end{bmatrix} = 0$$

i.e., the non-polynomial version of our eigenvalue problem,

$$4 - 8\,e^{-ttr-20} + 5\,\left(e^{-ttr-20}\right)^2 - 6\,g^2 - \left(e^{-ttr-20}\right)^3 + 5\,g^2 e^{-ttr-20} = 0\,.$$

In a test run the numerical analysis of this equation reproduced the results given in Table Nr. 1 of *loc. cit.*.

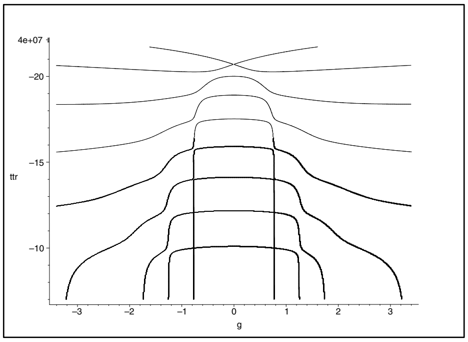


Fig. 2. Nine eigenvalues $p = p(g)$ of metric $\Theta_g^{(9)}(1)$

On this basis one may expect that the key problem brought by the rescaling appears tractable by the MAPLE-based numerical software. The main gain came with the substantial extension of the feasibility of the graphical determinations of the parameter-dependence of the physical domains $\mathcal{D}_g^{(N)}(a)$ with the growth of N. The characteristic illustration is offered by Figures 1 and 2 which clearly demonstrate the emergence of an obvious pattern which was not accessible without rescaling.

5 Summary

¿From the point of view of Quantum Mechanics it is rather unfortunate that for a given Hamiltonian $H \neq H^\dagger$ the specification of the metric $\Theta(H)$ prescribed by Dieudonné Eq. (1) is ambiguous [10]. In this context, our series of papers [1] - [8] has been devoted to the constructive study of the one-to-many mappings $H \to \Theta(H)$. In essence, we offered there a new methodical recipe of a systematic suppression of the ambiguity of the menu of eligible $\Theta(H)$s.

In our present continuation and extension of these papers, we decided to explain the symbolic-manipulation aspects of such a recipe in more detail. Emphasizing that such a problem would be hardly tractable and/or solvable without an essential interaction between the abstract quantum theory and the symbolic-manipulation techniques and algebraic constructions assisted by contemporary computers.

One of byproducts of such an interaction between methods has been shown to lie in the amendment of the numerical aspects of the necessary simultaneous analysis of physical domains and of the other properties of *both* of the physics-representing operators H and Θ. On the basis of these results one can conclude that the confirmation of feasibility of a methodical symbiosis between algebra and analysis contributes to the contemporary quick growth of popularity of phenomenological applications of models with nonhermitian matrices in optics [20].

Acknowledgments. Work was supported by the GAČR grant Nr. P203/11/1433, by the MŠMT "Doppler Institute" project Nr. LC06002 and by the Inst. Res. Plan AV0Z10480505.

References

1. Znojil, M.: Fundamental length in quantum theories with PT-symmetric Hamiltonians. Phys. Rev. D. 80, 045022, 13 pages (2009)
2. Znojil, M.: Gegenbauer-solvable quantum chain model. Phys. Rev. A 82, 052113,10 pages (2010)
3. Znojil, M.: Scattering theory using smeared non-Hermitian potentials. Phys. Rev. D. 80, 045009, 12 pages (2009)
4. Znojil, M.: Cryptohermitian picture of scattering using quasilocal metric operators. Symmetry, Integrability and Geometry: Methods and Applications 5, 085, 21 pages (2009)
5. Znojil, M.: Determination of the domain of the admissible matrix elements in the four-dimensional PT-symmetric anharmonic model. Phys. Lett. A 367, 300–306 (2007)
6. Znojil, M.: Fundamental length in quantum theories with PT-symmetric Hamiltonians II: The case of quantum graphs. Phys. Rev. D. 80, 105004, 20 pages (2009)
7. Znojil, M.: Anomalous real spectra of non-Hermitian quantum graphs in a strong-coupling regime. J. Phys. A: Math. Theor. 43, 335303 (2010)
8. Znojil, M.: Complete set of inner products for a discrete PT-symmetric square-well Hamiltonian. J. Math. Phys. 50, 122105 (2009)
9. Dieudonne, J.: Quasi-Hermitian operators. In: Proc. Int. Symp. Lin. Spaces, pp. 115–122. Pergamon, Oxford (1961)
10. Scholtz, F.G., Geyer, H.B., Hahne, F.J.H.: Quasi-Hermitian Operators in Quantum Mechanics and the Variational Principle. Ann. Phys. (NY) 213, 74 (1992)
11. Bender, C.M.: Making sense of non-hermitian Hamiltonians. Rep. Prog. Phys. 70, 947–1018
12. Znojil, M.: Three-Hilbert-space formulation of Quantum Mechanics. Symmetry, Integrability and Geometry: Methods and Applications 5, 001, 19 pages (2009)
13. Char, B.W., et al.: Maple V Language Reference Manual. Springer, New York (1993)
14. Gradshteyn, I.S., Ryzhik, I.M.: Tablicy integralov, summ, ryadov i proizvedenii, Nauka, Moscow (1971)
15. Abramowitz, M., Stegun, I.A.: Handbook of Mathematical Functions. Dover, New York (1970)
16. Siegl, P.: Supersymmetric quasi-Hermitian Hamiltonians with point interactions on a loop. J. Phys. A: Math. Theor. 41, 244025 (2008)
17. Dorey, P., Dunning, C., Tateo, R.: The ODE/IM correspondence. J. Phys. A: Math. Theor. 40, R205–R283 (2007)
18. Davies, E.B.: Linear Operators and Their Spectra. Cambridge University Press, Cambridge (2007)
19. Mostafazadeh, A.: Pseudo-Hermitian Representation of Quantum Mechanics. Int. J. Geom. Meth. Mod. Phys. 7, 1191–1306 (2010)
20. Makris, K.G., El-Ganainy, R., Christodoulides, D.N., Musslimani, Z.H.: Beam Dynamics in PT Symmetric Optical Lattices. Phys. Rev. Lett. 100, 103904 (2008)

Author Index