ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ
«ФЕДЕРАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЦЕНТР
«ИНФОРМАТИКА И УПРАВЛЕНИЕ»
РОССИЙСКОЙ АКАДЕМИИ НАУК»
ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР им. А. А. ДОРОДНИЦЫНА
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ»

КОМПЬЮТЕРНАЯ АЛГЕБРА

Материалы 4-й международной конференции

Москва, 28–29 июня 2021 г.

COMPUTER ALGEBRA

4th International Conference Materials

Moscow, June 28–29, 2021

МАКС ПРЕСС

МОСКВА – 2021

УДК 519.6(063)
ББК 22.19;31
К63

Ответственные редакторы:
С.А. Абрамов – д-р физ.-мат. наук
Л.А. Севастьянов – д-р физ.-мат. наук

Рецензенты:
Ю.О. Трусова – канд. техн. наук
К.П. Ловецкий – канд. физ.-мат. наук

Компьютерная алгебра. Материалы 4-й международной конференции. Москва, 28–29 июня 2021 г./ под ред. С.А. Абрамова, Л.А. Севастьянова. – Москва : МАКС Пресс, 2021. – 124 с.
ISBN 978-5-317-06623-9
https://doi/org/10.29003/m2019.978-5-317-06623-9

Международная конференция проводится совместно Вычислительным центром им. А.А.Дородницына ФИЦ "Информатика и управление" РАН и Российским университетом дружбы народов. В представленных на конференции докладах обсуждаются актуальные вопросы компьютерной алгебры — научной дисциплины, алгоритмы которой ориентированы на точное решение математических и прикладных задач с помощью компьютера.

УДК 519.6(063)
ББК 22.19;31

Computer algebra: 4th International Conference Materials. Moscow, June 28–29, 2021/ Ed. S.A. Abramov, L.A. Sevastyanov. Moscow : MAKS Press, 2021. – 124 p.
ISBN 978-5-317-06623-9
https://doi/org/10.29003/m2019.978-5-317-06623-9

The international conference is organized jointly by Dorodnicyn Computing Center of Federal Research Center "Computer Science and Control" of Russian Academy of Science and Peoples' Friendship University of Russia. The talks presented at the conference discuss actual problems of computer algebra — the discipline whose algorithms are focused on the exact solution of mathematical and applied problems using a computer.

ISBN 978-5-317-06623-9

Conference Chair

I.A. Sokolov Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia

Program Committee General Co-Chairs

Yu.G. Evtushenko Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
K.E. Samuylov Applied Mathematics and Communications Technology Institute, Peoples' Friendship University of Russia, Russia

Program Committee Vice Chairs

L.A. Sevastianov Peoples' Friendship University of Russia, and Joint Institute for Nuclear Research, Russia
S.A. Abramov Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia

Program Committee

V.A. Artamonov Moscow State University, Russia
M. Barkatou Universite de Limoges, France
A.V. Bernstein Skolkovo Institute of Science and Technology, Russia
A.A. Bogolubskaya Joint Institute for Nuclear Research, Russia
Yu.A. Blinkov Saratov State University, Russia
O.V. Druzhinina Institute of Informatics Problems, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
Yu.A. Flerov Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences
R.R. Gontsov Institute for Information Transmission Problems of Russian Academy of Sciences, Russia
V.V.Kornyak Joint Institute for Nuclear Research, Russia
A.V.Korolkova Peoples' Friendship University of Russia, Russia
D.S.Kulyabov Peoples' Friendship University of Russia, and Joint Institute for Nuclear Research, Russia
W.Lee University of Stirling, Scotland, UK
G.I. Malaschonok National University of Kyiv-Mohyla Academy, Ukrain
M.D. Malykh Peoples' Friendship University of Russia, Russia
A.A. Mikhalev Moscow State University, Russia
A.V. Mikhalev Moscow State University, Russia
M. Petkovšek University of Ljubljana, Slovenia
A.N. Prokopenya Warsaw University of Life Sciences, Poland
T.M. Sadykov Plekhanov Russian University of Economics, Russia
V.A. Serebryakov Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
D. Ştefănescu University of Bucharest, Romania
M. Wu East China Normal University, Shanghai, P.R.China

Organising Committee Chair

G.M. Mikhailov Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
D.V. Divakov Peoples' Friendship University of Russia, Russia

Organising Committee Vice Chairs

A.A. Ryabenko Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
A.A. Tiutiunnik Peoples' Friendship University of Russia, Russia

Organising Committee

A.V. Demidova Peoples' Friendship University of Russia, Russia
V.V. Dorodnicyna Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
D.E. Khmelnov Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
K.B. Teimurazov Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
T.R. Velieva Peoples' Friendship University of Russia, Russia
S.V. Vladimirova Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia
Y.A. Zonn Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, Russia

Foreword

The fourth International Conference "Computer algebra"

http://www.ccas.ru/ca/conference

is organized in Moscow from 28 to 29 June 2021 jointly by the Dorodnicyn Computing Centre (Federal Research Center "Computer Science and Control") of Russian Academy of Science and the Russian University of Peoples' Friendship.

The first, second and third conferences were held in Moscow in 2016, 2017 and 2019:

http://www.ccas.ru/ca/conference2016,
http://www.ccas.ru/ca/conference2017,
http://www.ccas.ru/ca/conference2019.

Computer algebra algorithms are focused on the exact solutions of mathematical and applied problems using a computer. The participants of this conference present new results obtained in this field.

Program and Organizing Committees of the conference

Contents

Invited talks

Contributed talks

Invited talks

Level Lines of a Polynomial in the Plane

A.D. Bruno[1], A.B. Batkhin[1,2]

[1]*Keldysh Institute of Applied Mathematics of RAS, Russia*

[2]*Moscow Institute of Physics and Technology, Russia*

e-mails: abruno@keldysh.ru, batkhin@gmail.com

Abstract. We propose a method for computing the position of all level lines of a real polynomial in the real plane. To do this, it is necessary to compute its critical points and critical lines (there are finite number of them), and then its critical values of the polynomial. Now finite number of critical levels and one representative of noncritical level corresponding to a value between two neighboring critical ones enough to compute. Software for these computations is discussed. A nontrivial example is considered.

Keywords: polynomial, critical point, critical curve, level line

1. Introduction

Let $X = (x_1, x_2) \in \mathbb{R}^2$. Consider the real polynomial $f(X)$. For a constant $c \in \mathbb{R}$, the curve in the plane $\mathbb{R}^2$

$$f(X) = c \tag{1}$$

is the *level line* of the polynomial $f(X)$.

Our task is to describe all level lines of the polynomial $f(X)$ on the real plane $X \in \mathbb{R}^2$. Let $C_* = \inf f(X)$ and $C^* = \sup f(X)$ for $X \in \mathbb{R}^2$. The main result is as follows:

Theorem 1. *There is a finite set of critical values of c:*

$$C_* < c_1^* < c_2^* < \cdots < c_m^* < C^*, \tag{2}$$

to which there correspond the critical level lines

$$f(X) = c_j^*, \quad j = 1, \ldots, m, \tag{3}$$

and for c values from each of the $m + 1$ intervals

$$I_0 = (C_*, c_1^*)\,,\ I_j = \left(c_j^*, c_{j+1}^*\right), j = 1, \ldots, m-1,\ I_m = (c_m^*, C^*) \tag{4}$$

level lines are topologically equivalent. If $C_* = c_1^*$ or $C^* = c_m^*$, there are no I_0 or I_m intervals.

Therefore, to identify the location of all level lines of the polynomial $f(X)$ we need to find all critical values c_j^*, figure m of critical level lines (3) and one level line each for any value c of $m + 1$ interval (4). The way to compute these level lines is described in [1] and partly in [2, Ch. I, § 2] using power geometry. The local structure of the polynomial level lines was considered in [2, Ch. I, § 3]. Here some results from [2, Ch. I, § 3] are extended.

2. Critical points and critical lines

A point $X = X^0$ is called *simple* for a polynomial $f(X)$ if at least one of the partial derivatives $\partial f/\partial x_1$, $\partial f/\partial x_2$ is non-zero in it.

Definition 1. Point $X = X^0$ for a polynomial $f(X)$ is called *critical of order* k if at the point $X = X^0$ all partial derivatives of $f(X)$ to order k are zero, that is, all

$$\frac{\partial^l f}{\partial x_1^i \partial x_2^j}(X^0) = 0,\ 1 \leqslant i + j = l \leqslant k,$$

and at least one partial derivative of order $k + 1$ is non-zero.
Definition 2. Line

$$g(X) = 0 \tag{5}$$

is called *critical* for a polynomial $f(X)$ if

1. it lies on some level line (1) and
2. on it $\partial f/\partial x_1 \equiv 0$, or $\partial f/\partial x_2 \equiv 0$.

Values of the constant $c = f(X)$ at critical points $X = X^0$ and at critical lines (5) are called *critical* and denote c_j^* according to (2).

3. Local and global level line analysis

Near the point $X = X^0$ we will consider analytic invertible coordinate substitutions

$$y_i = x_i^0 + \varphi_i\left(x_1 - x_1^0, x_2 - x_2^0\right),\ i = 1, 2, \tag{6}$$

where φ_i are analytic functions of $X - X^0$.
Lemma 1. [2, Ch. I, § 3] *If the point X^0 is simple and it has $\partial f/\partial x_2 \neq 0$, then there exists a substitution* (6) *that transforms equation* (1) *into the form*

$$f(X) = y_2 = c. \tag{7}$$

Level lines (7) are lines parallel to the y_1 axis.
Consider solutions to the equation (1) near the critical point $X^0 = 0$ of order 1. Then

$$f(X) = f_0 + ax_1^2 + bx_1x_2 + cx_2^2 + \dots$$

The discriminant Δ of the written above quadratic form is $\Delta = b^2 - 4ac$.
Lemma 2. [2, Ch. I, § 3] *If at the first-order critical point $X^0 = 0$ the discriminant $\Delta \neq 0$, then there exists a substitution* (6) *that reduces equation* (1) *to the form*

$$f(X) = f_0 + \sigma y_1^2 + y_2^2 = c, \tag{8}$$

where $\sigma = 1$ (if $\Delta < 0$) or $\sigma = -1$ (if $\Delta > 0$).
Lemma 3. *If at the first-order critical point $X^0 = 0$ the discriminant $\Delta = 0$, then there is a substitution* (6) *that brings the equation* (1) *to the form*

$$f(X) = f_0 + y_2^2 + \tau y_1^n = c, \tag{9}$$

where the integer $n > 2$ and the number $\tau \in \{-1, 0, +1\}$.

The expressions (8) and (9) are *normal forms* of the polynomial $f(X)$ near its critical point of the first order $X^0 = 0$. Now let the critical point $X^0 = 0$ be of order $k > 1$. According to [1, p. 5], its corresponding level line either has no branches going to the critical point $X^0 = 0$, or has several such branches. In the first case, the critical level line consists

of this point $X^0 = 0$, and other level lines are closed curves around it and correspond to one sign of the difference $c - f(X^0)$.

In the second case, the critical level line consists of a finite number of branches of different multiples entering the critical point $X^0 = 0$. They divide the vicinity of this critical point into curvilinear sectors. The remaining level lines fill these sectors, remaining at some distance from the critical point $X^0 = 0$. In neighboring sectors, they correspond to different signs of the difference $c - f(X^0)$ if the branch separating them has odd multiplicity, and to one sign of this difference if the branch separating them has even multiplicity.

Lemma 4. *For all values of constant c from one of the* $m + 1$ *intervals* (4) *the level lines* (1) *are topologically equivalent.*

Theorem 1 follows from Lemmas 1–4 and the properties of level lines near the critical point $X^0 = 0$ of order $k > 1$ described above.

4. Building level line sketches

To build a level line sketch, you can use any computer algebra system that has programs for building isolines (isosurfaces) for two- or three-dimensional scalar fields. These programs use different computational algorithms based on finite elements that triangulate some part of the plane (usually with triangular or square finite elements). Then the function values (1) in the vertices are calculated and interpolated over the whole finite element. Such algorithms deal well with the situation when the level line has no singularities. The presence of singularities makes us significantly reduce the partitioning step and, accordingly, increase the volume of calculations.

In some cases it is possible to improve the quality of the level line sketch if for a certain value of c the equation (1) can be factored into multipliers and zeros of those multipliers (or part of them) define an algebraic curve of genus 0. In this case we can compute a rational parametrization of such a curve, and use it to construct a sketch with any accuracy. The package `algcurves` of `Maple` does a good job with such a problem. This package, in particular, allows to study planar algebraic curves. It can be used to represent a sketch of the curve $f(x_1, x_2) = 0$ by numerically integrating the corresponding differential equation $\dfrac{\partial f}{\partial x_1} + \dfrac{\partial f}{\partial x_2} \cdot \dfrac{dx_2}{dx_1} = 0$ for some set of initial conditions, defined by points where at least one partial derivative of the function $f(x_1, x_2)$ is zero. Using this package to study a set of curves with different orders of singularities showed that in the case of high-order singularities, the quality of the sketch is not very high.

5. Example

Consider computing the level lines of the polynomial

$$\begin{aligned} f(X) =& x_1^5 + 2x_1^4x_2 + 4x_1^4 + x_1^3x_2^2 + x_1^3x_2 + 4x_1^3 + x_1^2x_2^3 - 6x_1^2x_2^2 - \\ & - 12x_1^2x_2 + 2x_1x_2^4 + x_1x_2^3 - 12x_1x_2^2 - 12x_1x_2 + x_2^5 + 4x_2^4 + 4x_2^3. \end{aligned} \tag{10}$$

Here $C_* = -\infty$, $C^* = +\infty$. They are reached at $x_1 = 0$.

To compute the critical points, we will use the Gröbner basis for the system consisting of the polynomial itself (10) and its two partial derivatives on the variables x_1, x_2, choosing the lexicographic order of the variables as $x_1 < x_2 < c$. The first polynomial of the computed basis is $g_1 = c\,(c+1)\,(3125c^2 + 56736c + 54000)$. Among its roots, we select those critical

values c_j^* for which the corresponding values are real:

$$c^* \in \left\{ -\left(28368 + 3036\sqrt{69}\right)/3125, -\left(28368 - 3036\sqrt{69}\right)/3125, 0 \right\}.$$

The value $c_1^* \approx -17.1478$ corresponds to the critical point $(x_1^{(1)} = x_2^{(1)} = (3+\sqrt{69})/10)$. It has the discriminant $\Delta \approx -14221.2 < 0$ and is isolated. The value $c_2^* \approx -1.0077$ corresponds to the critical point $(x_1^{(2)} = x_2^{(2)} = (3-\sqrt{69})/10)$. It has the discriminant $\Delta \approx 1.3415 > 0$ and two branches passing through it. The value $c_3^* = 0$ corresponds to the critical point $(x_1^{(3)} = x_2^{(3)} = 0)$. Here the discriminant $\Delta \approx 144 > 0$ and two branches pass through it. Finally, at $c_3^* = 0$ there is a critical line $x_1 + x_2 + 2 = 0$.

The critical level lines for the polynomial (10) are shown in Figure 1. The critical points $X^{(1)}, X^{(2)}$ and $X^{(3)} = 0$ and the critical line $x_1 + x_2 + 2 = 0$ are shown as bold. The level line for the critical value $c_1^* \approx -17.14$ is shown as dash-dotted line. The level line for the critical value $c_2^* \approx -1$ is shown as dashed line. It has three components: an oval in the first quadrant, two intersecting branches above the critical line, and one curve below it. The level line for the critical value $c_3^* = 0$ is shown as a solid line. It consists of two components: the critical straight line and the Descartes folium. Four types of non-critical level lines are easily recovered as lying entirely between adjacent critical ones. They are shown in Fig. 1 as dotted lines for $c = -25$ and $c = 25$.

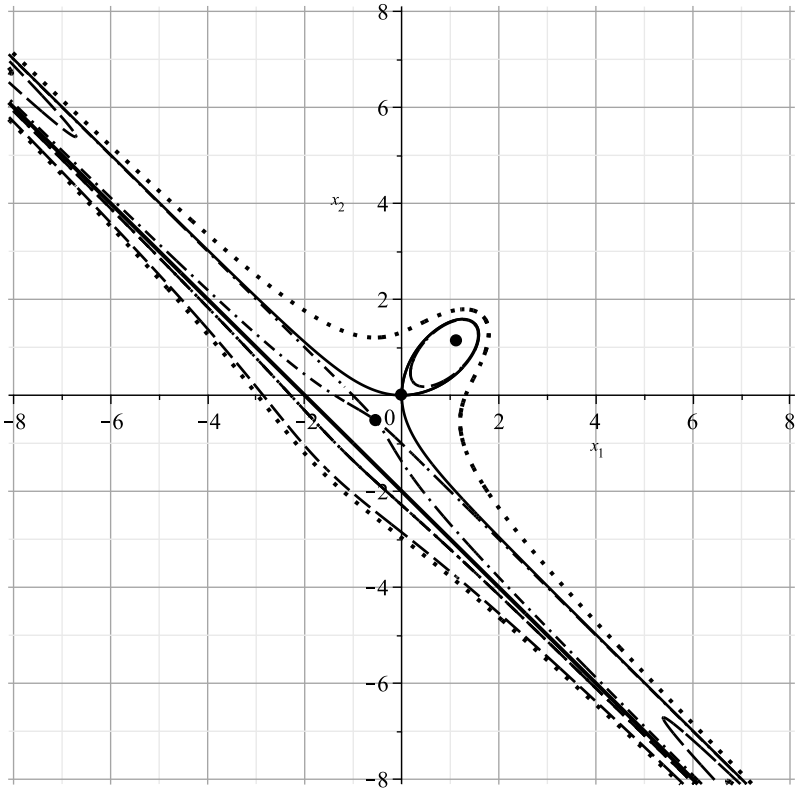


Figure 1. Level lines of the polynomial (10).

References

1. *Bruno A.D., Batkhin A.B.* Introduction to nonlinear analysis of algebraic equations. KIAM Preprints. 2020. No. 87. DOI: 10.20948/prepr-2020-87 (in Russian).

2. *Bruno A.D.* Local Methods in Nonlinear Differential Equations. Berlin – Heidelberg – New York – London – Paris – Tokyo. Springer–Verlag, 1989.

What's New in Maple 2021

J. Gerhard[1]

[1] *Maplesoft, Canada*

e-mail: jgerhard@maplesoft.com

Abstract. We will give an overview over the new features of Maple 2021, including limits and asymptotic expansions, automatic plotting domain and range selection, a new Student:-ODEs package, approximate polynomial algebra, and improved LaTeX export.

Keywords: Maple 2021, computer algebra system

Contributed talks

On Infinite Sequences and Difference Operators

S.A. Abramov[1,2], M. A. Barkatou[3], M. Petkovšek[4]

[1] *Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of RAS, Russia*

[2] *Faculty of Computational Mathematics and Cybernetics, Moscow State University, Russia*

[3] *University of Limoges ; CNRS ; XLIM UMR 7252 ; MATHIS, France*

[4] *University of Ljubljana, Faculty of Mathematics and Physics, Slovenia*

e-mail: sergeyabramov@mail.ru, moulay.barkatou@unilim.fr, Marko.Petkovsek@fmf.uni-lj.si

Abstract. Some properties of linear difference operators whose coefficients have the form of infinite two-sided sequences over a field of characteristic zero are considered. In particular, it is found that such operators are deprived of some properties that are natural for differential operators over differential fields. In addition, we discuss decidability of certain problems arising in connection with the algorithmic representation of infinite sequences.

Keywords: linear difference operators, infinite sequences, dimension of solution spaces

1. Introduction

The need to consider linear difference operators with coefficients in the form of sequences (or of equivalence classes of sequences) arises, in particular, in connection with the universal Picard-Vessiot extensions of difference fields (cf. [3]). The point that difference-ring extensions of difference fields have to be considered in this context was first noticed by C. H. Franke in [2]. Thus questions arise naturally about properties of difference operators having their coefficients in a difference ring.

It is not surprising that difference operators over rings, in particular, over rings of sequences, lack some properties, enjoyed by differential operators over differential fields (cf. [3, Appx. A] or [5]). Below, we present some such properties, and demonstrate undecidability of several problems related to linear difference operators with sequences as coefficients.

2. Preliminaries

In the sequel, R denotes the ring $\mathbb{Q}^{\mathbb{Z}}$ of two-sided sequences having rational-number terms, with termwise addition and multiplication, and σ is the shift operator on R defined by $(\sigma c)(k) = c(k+1)$ for all $k \in \mathbb{Z}$. Clearly, σ is an automorphism of R, and the field of rational numbers $\mathbb{Q}$ is the field of constants of R. For any $k, m \in \mathbb{Z}$, $m \geq 1$, define $\delta_k, \omega_m \in R$ by

$$\delta_k(n) = \begin{cases} 1, & n = k, \\ 0, & \text{otherwise}, \end{cases} \qquad \omega_m(n) = \begin{cases} 1, & n \equiv m \pmod{m+1}, \\ 0, & \text{otherwise}. \end{cases}$$

The ring $R[\sigma]$ is the ring of linear difference operators with coefficients in R. The *order* of $L \in R[\sigma]$, denoted by $\operatorname{ord} L$, is the non-negative integer equal to the degree of the (skew) polynomial L in σ; conventionally, $\operatorname{ord} 0 = -\infty$. For $L \in R[\sigma]$, the $\mathbb{Q}$-linear space of all $f \in R$ s.t. $L(f) = 0$ will be denoted by V_L.

3. On dimension of solution spaces

3.1. A useful lemma

Lemma 1. *Let $L \in R[\sigma]$ have order $m \in \mathbb{N}$. If there are a two-sided sequence $f \in V_L$ and a one-sided sequence of indices $\nu \in \mathbb{Z}^{\mathbb{N}}$ such that*

1. $\forall n \in \mathbb{Z}, n \geq \nu_0 \colon (f(n) \neq 0 \iff \exists k \in \mathbb{N} \colon n = \nu_k)$, *and*

2. $\forall k \in \mathbb{N} \colon \nu_{k+1} \geq \nu_k + m + 1$,

then $\dim V_L = \infty$.

Example 1. Let $L = \sum_{i=0}^{m} a_k(n)\sigma^k \in R[\sigma]$ be such that $(L(\omega_m))(n) = 0$. As $\omega_m((m+1)k) = \omega_m((m+1)k+1) = \cdots = \omega_m((m+1)k+m-1) = 0$ and $\omega_m((m+1)k+m) = 1$ for all $k \in \mathbb{Z}$, it follows that for each $n \in \mathbb{Z}$ and $i \in \{0, 1, \ldots, m\}$ exactly one term $\omega_m(n+i)$ is nonzero, namely that for which $n = (m+1)k + m - i$ for some $k \in \mathbb{Z}$. Hence $\omega_m \in V_L$ implies that $a_i((m+1)k+m-i) = 0$ for all $k \in \mathbb{Z}$ and $i \in \{0, 1, \ldots, m\}$. Now it is not difficult to see that $\delta_{(m+1)k+m} \in V_L$ for all $k \in \mathbb{Z}$. Since $\{\delta_{(m+1)k+m};\ k \in \mathbb{Z}\}$ is an infinite $\mathbb{Q}$-linearly independent subset of R, it follows that $\dim V_L = \infty$.

3.2. Annihilating operators

Proposition 1. *For any positive integer m there exist $f_1, \ldots, f_m \in R$ such that if for some $L \in R[\sigma] \setminus \{0\}$,* ord $L \leq m$, *the equalities*

$$L(f_i) = 0, \quad i = 1, \ldots, m,$$

hold then $\dim V_L = \infty$.

Proof. This is a consequence of Lemma 1. For example, any $f_1, \ldots, f_m \in R$ such that $f_1 = \omega_m$ possess the stated property. □

Recall that in the differential case we can find an operator L, ord $L \leq m$, annihilating given $f_1, \ldots, f_m$ such that $\dim V_L$ equals the maximum number of linearly independent elements of the set $\{f_1, \ldots, f_m\}$.

Note that an even stronger form of the statement of Lemma 1 is possible:

Lemma 1*. *There exist sequences such that if an operator L of arbitrary order annihilates any one of them, then* $\dim V_L = \infty$.

Accordingly, the statement of Proposition 1 can be strengthened as well by skipping the restriction ord $L \leq m$.

3.3. Least common left multiple

Definition 1. For $L_1, L_2 \in R[\sigma]$, we define $\operatorname{lclm}(L_1, L_2)$ as the set of all operators $L \in R[\sigma]$ such that

- $L \neq 0$,
- L is a common left multiple of L_1 and L_2,
- there is no operator M such that $M \neq 0$, M is a common left multiple of L_1 and L_2, and ord $M <$ ord L. □

Note that it is possible that for some $L_1, L_2 \in R[\sigma]$ their only common left multiple is 0.

Example 2. Consider two operators of order 0: $L_1 = c = \omega_1$, i.e., $c(2k) = 0$, $c(2k+1) = 1$ for all $k \in \mathbb{Z}$, and $L_2 = d = \sigma(\omega_1)$, i.e., $d(2k) = 1$, $d(2k+1) = 0$ for all $k \in \mathbb{Z}$. It is easy to see that the only common left multiple of c and d is the zero sequence.

Proposition 2. *There exist first-order operators L_1, L_2 such that*

(i) $\dim V_{L_1} = \dim V_{L_2} = 1$,

(ii) there exists a second-order common left multiple L of L_1, L_2,

(iii) for any common left multiple M of L_1, L_2 with $\operatorname{ord} M \leq 2$, *we have* $\dim V_M = \infty$.

It follows from Proposition 2 that for $L \in \operatorname{lclm}(L_1, L_2)$, the equality

$$V_L = V_{L_1} + V_{L_2} \tag{1}$$

need not hold in general.

Recall that in the differential case, when L_1, L_2 are two operators over a differential field $\mathbb{K}$, equality (1) holds if we consider solutions from a Picard-Vessiot extension of $\mathbb{K}$. Similarly, this holds also in the case of differential systems (see [1]).

In the scalar difference case, equality (1) is valid if the coefficients belong to a difference field possessing some special properties (cf. [4]).

4. Some undecidable problems

The problem of representing infinite sequences is an important one in computer algebra. A general formula for the n-th element of a sequence is not always available, and may even not exist. A natural way for representing a sequence is by an algorithm for computing its elements from their indices. We will call the sequence *computable* if such an algorithm exists. The algorithmic representation of a sequence is, of course, not unique, which is one of the reasons for undecidability of the zero-testing problem for computable sequences.

4.1. A consequence of a classical result by Turing

Below, we discuss some undecidable problems related to operators with coefficients in R. Their proofs are, in general, based on the following consequence of the well-known Turing's result [6] on undecidability of the halting problem:

Let S be a set with two or more elements, such that there exists an algorithm to test for equality of any two given elements from S. Then there exists no algorithm to decide whether a given computable sequence, either one-sided $(c(0), c(1), c(2), \dots)$ or two-sided $(\dots, c(-1), c(0), c(1), \dots)$, has all of its elements equal to a given $u \in S$; the same holds for the question of whether such a sequence has none of its elements equal to a given $u \in S$.

4.2. On testing for invertibility and divisibility in $R[\sigma]$

Lemma 2. *The question of whether a given computable sequence $c \in R$ and a given nonnegative integer r satisfy the statement*

$$\exists k \in \mathbb{N}\ \forall n \in \mathbb{Z}\colon\ c(n)c(n+r)c(n+2r)\dots c(n+kr) = 0$$

is undecidable.

Proposition 3. *There exists no algorithm for testing for an arbitrary $L \in R[\sigma] \setminus \{0\}$ whether L is invertible in $R[\sigma]$ or not.*

Remark 1. For each integer $r \geq 0$, there is an invertible operator $L_r \in R[\sigma]$ of order r. Indeed, let $\mathbf{1}$ denote the sequence all of whose elements are equal to 1. If $r = 0$ then $L_0 = \mathbf{1}$ is invertible since $\mathbf{1} \cdot \mathbf{1} = \mathbf{1}$. Otherwise, $L_r = \delta_0(n)\sigma^r + \mathbf{1}$ is invertible since

$$(\delta_0(n)\sigma^r + \mathbf{1})(-\delta_0(n)\sigma^r + \mathbf{1}) \;=\; -\delta_0(n)\delta_0(n+r)\sigma^{2r} + \mathbf{1} \;=\; \mathbf{1}.$$

Proposition 4. *There exists no algorithm for testing for arbitrary $L_1, L_2 \in R[\sigma] \setminus \{0\}$ whether L_1 is right-divisible by L_2 in $R[\sigma]$.*

Proof. If such an algorithm existed then one could use it to test for invertibility of a given $L \in R[\sigma] \setminus \{0\}$. Indeed, take any $M \in R[\sigma]$. If L is invertible then $M = ML^{-1}L$ and $M + \mathbf{1} = (M + \mathbf{1})L^{-1}L$ are both right-divisible by L. Conversely, if there are $A, B \in R[\sigma]$ s.t. $M = AL$ and $M + \mathbf{1} = BL$ then $\mathbf{1} = (B - A)L$, so L is invertible. Thus, L is invertible iff M and $M + \mathbf{1}$ are both right-divisible by L, and one could use this to test for invertibility if one could test for divisibility from the right. But by Proposition 3, this is impossible. □

4.3. On the existence of a nonzero common left multiple

Here we discuss the problem of possibility or impossibility of algorithms for testing for the existence of a nonzero common left multiple of two given operators $L_1, L_2 \in R[\sigma]$.

Notice that the impossibility of a general algorithm for this problem can be shown easily by considering the case of zero-order operators, i.e., the case of sequences.

Proposition 5. *Let $r, s, \in \mathbb{N}$. There is no algorithm for testing the existence of a non-zero common left multiple of two given operators $L_1, L_2 \in R[\sigma]$,* ord $L_1 = r,$ ord $L_2 = s$.

As a consequence we get the following: *There is no algorithm for testing the existence of a non-zero common left multiple of two given operators $L_1, L_2 \in R[\sigma]$.*

Acknowledgments: The first author was supported in part by the Russian Foundation for Basic Research, project no. 19-01-00032. The third author was supported in part by the Ministry of Education, Science and Sport of Slovenia research programme P1-0294.

References

1. *Abramov S., Barkatou M., Petkovšek M.* Matrices of scalar differential operators: divisibility and spaces of solutions. Computational Mathematics and Mathematical Physics. 2020. Vol. 60, No 1. P. 109–118.
2. *Franke C.H.* Picard-Vessiot theory of linear homogeneous difference equations. Trans. Amer. Math. Soc. 1963. Vol. 108. P. 491–515.
3. *Hendriks P., Singer M.* Solving difference equation in finite terms. J. Symbolic Comput. 1999. Vol. 27(3). P. 239–259.
4. *van Hoeij M.* Finite singularities and hypergeometric solutions of linear recurrence equations. J. Pure Appl. Algebra. 1999. Vol. 139. P. 109–131.
5. *Petkovšek M.* Symbolic computation with sequences. Programming and Computer Software. 2006. Vol. 32. P. 65–70.
6. *Turing A.* On computable numbers, with an application to the Entscheidungsproblem. Proceedings of the London Mathematical Society. Series 2. 1936. Vol. 42. P. 230–265.

Quadratization of ODE Systems

F. Alauddin[1], A. Bychkov[2], G. Pogudin[3]

[1] *Trinity School, New York, USA*

[2] *Higher School of Economics, Moscow, Russia*

[3] *LIX, CNRS, École Polytechnique, Institute Polytechnique de Paris, Palaiseau, France*

e-mail: foyez.alauddin21@trinityschoolnyc.org, abychkov@edu.hse.ru, gleb.pogudin@polytechnique.edu

Abstract. Quadratization, that is a transformation of an ODE system with polynomial right-hand side into an ODE system with at most quadratic right-hand side via the introduction of new variables, has been recently used for model order reduction, synthesis of chemical reaction networks, and numeric algorithms. We will discuss some recent results on optimal (i.e. with the smallest number of variables) quadratizations: a practical algorithm for finding optimal monomial quadratizations and some results on arbitrary quadratizations of scalar ODEs. We will conclude with a list of open problems.

Keywords: differential equations, quadratization, order reduction, combinatorial optimization

Introduction to quadratization

The *quadratization* problem is, given a system of ordinary differential equations (ODEs) with polynomial right-hand side, transform it into a system with at most quadratic right-hand side. We give a formal definition below, but start with a simple example of a scalar ODE:

$$x' = x^5. \tag{1}$$

The right-hand side has degree larger than two but if we introduce a new variable $y := x^4$, then we can write:

$$x' = xy, \quad \text{and} \quad y' = 4x^3x' = 4x^4y = 4y^2. \tag{2}$$

The right-hand sides of (2) are of degree at most two, and every solution of (1) is the x-component of some solution of (2). Such a transformation has recently appeared in model order reduction algorithms [4, 7, 8], in synthesis of chemical reaction networks [6], and in solving differential equations numerically [5].

Definition. *Consider a system of ODEs*

$$x_1' = f_1(\bar{x}), \quad \ldots, \quad x_n' = f_n(\bar{x}), \tag{3}$$

where $\bar{x} = (x_1, \ldots, x_n)$ *and* $f_1, \ldots, f_n \in \mathbb{C}[\mathbf{x}]$. *Then, a list of new variables*

$$y_1 = g_1(\bar{x}), \ldots, y_m = g_m(\bar{x}), \tag{4}$$

is said to be a quadratization *of (3) if there exist polynomials* $h_1, \ldots, h_{m+n} \in \mathbb{C}[\bar{x}, \bar{y}]$ *of degree at most two such that*

- $x_i' = h_i(\bar{x}, \bar{y})$ *for every* $1 \leqslant i \leqslant n$;
- $y_j' = h_{j+n}(\bar{x}, \bar{y})$ *for every* $1 \leqslant j \leqslant m$.

The number m will be called the order of quadratization. *A quadratization of the smallest possible order will be called* an optimal quadratization.

Definition. *If all the polynomials $g_1, \dots, g_m$ are monomials, the quadratization is called* a monomial quadratization. *If a monomial quadratization of a system has the smallest possible order among all the monomial quadratizations of the system, it is called* an optimal monomial quadratization.

Our results

In our recent paper [2] we design and implement an algorithm for finding an optimal monomial quadratization for a given ODE system. This is a combinatorial optimization problem with an infinite search space (the set of all new monomial variables). The prior approaches to this computation (e.g., [3, 4, 6]) restricted themselves to the new variables which are monomials of the form $x_1^{d_1} \cdot \ldots \cdot x_n^{d_n}$ such that

$$d_i \leqslant \max_{1 \leqslant j \leqslant n} \deg_{x_i} f_j.$$

It is known that one can always find (not necessarily optimal) monomial quadratization of this form. This restriction allowed to use powerful techniques based on the SAT-solvers. We have designed an algorithm based on the branch-and-bound approach and not relying on this restriction, and used it to show that allowing monomials of arbitrary degrees makes it possible to find much better quadratizations for some of the benchmarks. Therefore, our algorithm is the first practical algorithm for monomial quadratization with the optimality guarantees. We implemented the algorithm in Python, the implementation is available at https://github.com/AndreyBychkov/QBee.

Once we have a practical algorithm for finding an optimal monomial quadratization, it is natural to ask *can we find a more optimal quadratization if we allow new variables to take a more general polynomial form?* The first step towards answering this question was made in [1] for scalar ODEs, that is equations of the form $x' = p(x)$, where p is a polynomial. The main results of this paper are:

1. Full characterization is given of polynomials $p(x)$ such that the equation $x' = p(x)$ can be quadratized using only one new variable. It is shown that, for any number N, one can find such $p(x)$ that the optimal monomial quadratization will be of order at least N while the optimal quadratization will be of order one.

2. Every equation $x' = p(x)$ with $\deg p \leqslant 6$ can be quadratized with only two new variables. This is not true for monomial variables (one may need three) and the form of these new variables is quite nontrivial (was obtained using Gröbner bases). More precisely, an equation

$$x' = p_6x^6 + p_5x^5 + p_4x^4 + p_3x^3 + p_2x^2 + p_1x + p_0$$

with $p_6 \neq 0$ can be always quadratized with the following two new variables

$$z_1 := (\sqrt[6]{p_6} \cdot x + \frac{p_5}{6 \cdot \sqrt[6]{p_6^5}})^5 + (\frac{25p_5^3}{216\sqrt{p_6^5}} - \frac{5p_5p_4}{12\sqrt{p_6^3}} + \frac{5p_3}{8\sqrt{p_6}})(\sqrt[6]{p_6} \cdot x + \frac{p_5}{6 \cdot \sqrt[6]{p_6^5}})^2,$$

$$z_2 := (\sqrt[6]{p_6} \cdot x + \frac{p_5}{6 \cdot \sqrt[6]{p_6^5}})^3.$$

Open problems

In this section, we would like to state some interesting open questions about quadratizations of differential equations which we expect to be of interest to the computer algebra community.

Problem 1: theoretical bounds. It is still not so clear how large the optimal quadratization may be compared to the size of input. Conjecture 1 in [6] gives a sequence of system for which it is conjectured that the number of new variables as a function of the number of monomials in the original system grows exponentially but we are not aware of any proved non-polynomial lower bound.

Problem 2: Laurent-monomial quadratizations. In the contrast to the conjectured exponential number of new variables, it has been shown in Proposition 1 in [2] that, if the new variables are allowed to be Laurent monomials, then there always exists a quadratization of order not exceeding the number of monomials in the original system. However, we are not aware of any algorithm computing an optimal Laurent-monomial quadratization. In particular, we do not know whether such a optimal Laurent-monomial quadratization can be found as a subset of the quadratization given in the proof of Proposition 1 in [2].

Problem 3: pre-processing for monomial quadratization. It has been observed in [1] that the size of an optimal monomial quadratization may change dramatically after a linear change of variables. Therefore, it would be interesting to enhance the existing algorithms for finding monomial quadratizations with a pre-processing step that tries to perform a linear change of variables aiming at decreasing the number of new variables required for quadratization. An example of such pre-processing would be any change of variables strictly reducing the support of the right-hand sides.

Problem 4: general polynomial case. As has been observed in [1], allowing general polynomial (not necessarily monomial) quadratizations may dramatically reduce the number of required variables. Therefore, understanding such general quadratizations is an important problem. At the moment, it is open even in the scalar case, that is a single ODE $x' = p(x)$.

Problem 5: PDEs. In the context of applications to model order reduction problem, quadratization of PDEs is also of great interest (see [8]).

References

1. *Alauddin F.* Quadratization of ODEs: Monomial vs. Non-Monomial. SIAM Undergraduate Research Online. 2021. Vol. 14.
2. *Bychkov A. and Pogudin G.* Optimal monomial quadratization for ODE systems. Accepted to the International Workshop on Combinatorial Algorithms, 2021.
3. *Carothers D., Parker G., Sochacki J. and Warne P.* Some properties of solutions to polynomial systems of differential equations. Electron. J. Differ. Eq. 2005. No. 40. P. 1–17.
4. *Gu C.*. QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems. IEEE Trans. Comput.-Aided Design Integr. Circuits Syst. 2011. Vol. 30. P. 1307–1320.
5. *Guillot L., Cochelin B., and Vergez C.* A generic and efficient Taylor series-based continuation method using a quadratic recast of smooth nonlinear systems. Int. J. Numer. Methods Eng. 2019. Vol. 119. P. 261–280.

6. *Hemery M., Fages F., and Soliman S.* On the complexity of quadratization for polynomial differential equations. In: CMSB 2020. The 18th International Conference on Computational Methods in Systems Biology, Konstanz, Germany, 2020.

7. *Kramer B. and Willcox K.* Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition. AIAA J. 2019. Vol. 57, No. 6.

8. *Qian E., Kramer B., Peherstorfer B., and Willcox K.* Lift & Learn: Physics-informed machine learning for large-scale nonlinear dynamical systems. Physica D: Nonlinear Phenomena. 2020. Vol. 406.

Asymptotic Expansions of Solutions to the Hierarchy of the Fourth Painlevé Equation

V.I. Anoshin[1], A.D. Beketova[1], A.V. Parusnikova[1]

[1]*National Research University "Higher School of Economics", Russia*

e-mail: vianoshin@edu.hse.ru, adbeketova@edu.hse.ru, avparusnikova@hse.ru

Abstract. In this paper, we study the asymptotic expansions of the solutions of the hierarchies [2] of the fourth Painlevé equation using power geometry methods. To find these expansions, we need to build a Newton polygon and find solutions to the truncated equations using the rules given below. Hierarchies of Painlevé equations are used in physics, geometry, and other fields.

Keywords: Newton polygon, Painlevé hierarchies

The aim of the present work is to investigate asymptotic expansions of solutions to the hierarchies of the fourth Painlevé equation. Hierarchies of Painlevét equations – mathematical objects obtained as a result of isomondromic deformations of Painlevé equations. Painlevé functions are used in statistical physics, quantum field theory, geometry of minimal surfaces, number theory, and other areas. Using the methods of Power Geometry, we study the asymptotic solutions of the hierarchies of the fourth Painlevé equation.

In this work, we use the methods of Power Geometry [1] to study asymptotic expansions of solutions to the hierarchies of the fourth Painlevé equation [3]:

$$(y_{xx} - 2xy - 2y^3 - \beta)y^2y_{xxxx} - \frac{1}{2}y^2y_{xxx}^2 + (2y^2 + 8y^3y_x + 4yy_xx - y_xy_{xx} + \beta y_x)yy_{xxx} -$$

$$\frac{4}{3}yy_{xx}^3 + \left(3xy^2 + 3\beta y - \frac{3}{2}y^4 + \frac{3}{2}y_x^2\right)y_{xx}^2 + (\beta y^4 - 2y_xy^2 - 12y_x^2y^3 - 2\beta^2y + 10xy^5$$

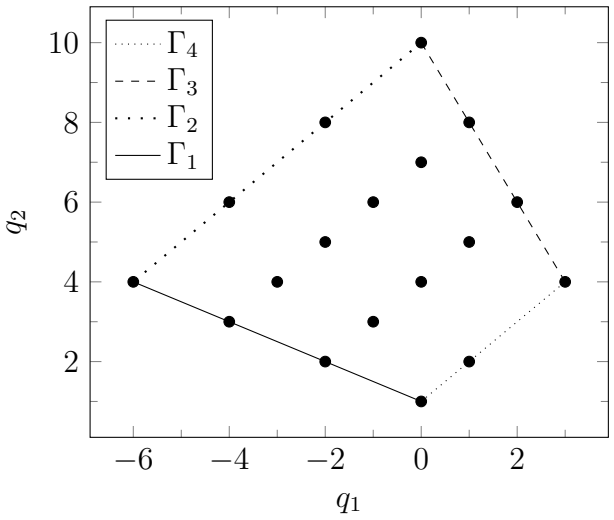
$$-3\beta y_x^2 + 10y^7 - 4xyy_x^2 - 4\beta xy^2)y_{xx} + 2(\beta - 4y^3)y^2y_x + \left(4\beta xy + 8xy^4 + \frac{3}{2}\beta^2 + 12\beta y^3\right)y_x^2 -$$

$$\frac{10}{3}y^{10} - 8xy^8 - 2\beta y^7 - 6x^2y^6 - 2x\beta y^5 + \left(\frac{1}{2}\beta^2 - 2 + 9\delta - \frac{4}{3}x^3\right)y^4 + x\beta^2y^2 + \frac{1}{3}\beta^3y = 0.$$

First, we need to build a Newton polygon for this equation. In order to do it, we should align the vector exponent $Q(a) = (q_1, q_2) \in \mathbb{R}^2$ with each differential monomial $y_{xx}y^2y_{xxxx}, -2xyy^2y_{xxxx}, -\beta y^2y_{xxxx}$ ... using the rules described in [3].

Then we get 18 vector exponents:
$(-4;3), (-6;4), (-3;4), (-4;6), (-2;5), (-2;2), (-1;6), (-2;8), (-1;3), (0;10), (1;8), (0;7), (2;6), (-1;5), (0;4), (3;4), (1;2), (0;1)$.

We put them on the coordinate plane (q_1, q_2) and construct the convex hull of the set containing these points. The resulting figure is called Newton polygon. To build polygons, we use a program by K.V. Romanov.



Each face $\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_4$ of a given polygon can be associated with points Q_i that belong to it. Each point corresponds to a monomial, from these monomials you can compose a function of the truncated sum for a given face of the polygon: $\hat{f}_j^{(d)}(X) = \sum a_i(X)$ with $Q(a_i) \in S_j^{(d)}$. For our equation we get:

For each face, we construct the normal vector directed outward the polygon: $N_1 = (-1; -2)$, $N_2 = (-1; 1)$, $N_3 = (2; 1)$, $N_4 = (1; -1)$.

The general form of the coordinates of the normal: $\lambda\omega(1, r)$, where $\lambda > 0$. If $\omega > 0$, then $x \to 0$, if $\omega < 0$, then $x \to \infty$.

The general view of the asymptotic expansion of the solution to the equation is $y = cx^r$.

If we need to find the next terms of the asymptotics, it is necessary to substitute y that is equal to the sum of the obtained term of the asymptotics and y1 (the required term of the asymptotics) into the original equation.

For this equation, we can build a Newton polygon. The second term of the asymptotics can be obtained by constructing a truncated equation for an edge whose normal has the following properties:

1) The value to which x tends, corresponding to the given normal, coincides with the value for the first term of the asymptotics.

2) The exponent x for a given normal is greater than the exponent for the first term of the asymptotics.

The method for finding the exponent x and the value to which x tends is described above.

For finding the general form of the terms of the asymptotic expansion, it is necessary:

The found asymptotic expansions are presented below:

$$W1 = \left\{ y = -0.1\beta x^2 + \frac{\beta}{80}x^5 + \sum_{k=2}^{\infty} c_k x^{2+3k}, \omega = -1, \beta \in C \right\}$$

$$W2 = \left\{ y = 0.5\beta x^2 + 0.025(2\beta \pm 4.24264\beta\sqrt{\sigma})x^5 + \sum_{k=2}^{\infty} c_k x^{2+3k}, \omega = -1, \beta \in C \right\}$$

$$W3 = \left\{ y = 2x^{-1} + \left(\frac{\beta - 2}{20}\right) x^2 + \sum_{k=2}^{\infty} c_k x^{3k-1}, \omega = -1 \right\}$$

$$W4 = \left\{ y = -2x^{-1} + \left(\frac{\beta + 2}{20}\right) x^2 + \sum_{k=2}^{\infty} c_k x^{3k-1}, \omega = -1 \right\}$$

$$W5 = \left\{ y = x^{-1} \pm \frac{\sqrt{-8-8\beta-2\beta^2+9\sigma}}{4\sqrt{2}}x^2 + \sum_{k=2}^{\infty} c_k x^{3k-1}, \omega = -1, \beta \in C, \sigma \in C \right\}$$

$$W6 = \left\{ y = -x^{-1} \pm \frac{\sqrt{8-8\beta+\beta^2-9\sigma}}{4\sqrt{2}}x^2 + \sum_{k=2}^{\infty} c_k x^{3k-1}, \omega = -1, \beta \in C, \sigma \in C \right\}$$

$$W7 = \left\{ y = i\sqrt{x} + (\frac{\beta}{4} \pm \frac{\sqrt{\beta-(\beta^2-18\sigma)}}{4})\frac{1}{x} + \sum_{k=1}^{\infty} c_k x^{-1-\frac{3}{2}k}, \omega = 1, \beta \in C \right\}$$

$$W8 = \left\{ y = -i\sqrt{x} + (\frac{\beta}{4} \pm \frac{\sqrt{\beta+(\beta^2-18\sigma)}}{4})\frac{1}{x} + \sum_{k=1}^{\infty} c_k x^{-1-\frac{3}{2}k}, \omega = 1, \beta \in C \right\}$$

$$W9 = \left\{ y = \sqrt{0.4}i\sqrt{x} - 0.5\beta\frac{1}{x} + \sum_{k=1}^{\infty} c_k x^{-1-\frac{3}{2}k}, \omega = 1, \beta \in C \right\}$$

$$W10 = \left\{ y = -\sqrt{0.4}i\sqrt{x} - 0.5\beta\frac{1}{x} + \sum_{k=1}^{\infty} c_k x^{-1-\frac{3}{2}k}, \omega = 1, \beta \in C \right\}$$

$$W11 = \left\{ y = \beta\frac{1}{x} + 0.5(-7\beta - 5\beta^3 + 6\beta\sigma)\frac{1}{x^4} + \sum_{k=2}^{\infty} c_k x^{-1-3k}, \omega = 1, \beta \in C \right\}$$

$$W12 = \left\{ y = -0.5\beta\frac{1}{x} \pm 0.75\beta\sqrt{\sigma}\frac{1}{x^4} + \sum_{k=2}^{\infty} c_k x^{-1-3k}, \omega = 1, \beta \in C \right\}$$

References

1. *Bruno A.D.* Asymptotics and expansions of solutions to an ordinary differential equation. Uspekhi Matem. Nauk. 2004. Vol. 59(3). P. 31–80 (in Russian) = Russian Mathem. Surveys. 2004. Vol. 59(3). P. 429–480 (in English)

2. *Pickering A.* Painlevé hierarchies and the Painlevé test. Theoret. and Math. Phys. 2003. Vol. 137(3). P. 445–456 (in Russian) = Theoret. and Math. Phys. 2003. Vol. 137(3). P. 1733–1742 (in English)

3. *Kudryashov N.A.* About the fourth Painlevé hierarchy. Theoret. and Math. Phys. 2003. Vol. 134(1). P. 101–109 (in Russian) = Theoret. and Math. Phys. 2003. Vol. 134(1). P 86–93 (in English)

Algorithms for Solving a Polynomial Equation in One or Two Variables

A.B. Batkhin[1,2], A.D. Bruno[1]

[1]*Keldysh Institute of Applied Mathematics of RAS, Russia*

[2]*Moscow Institute of Physics and Technology, Russia*

e-mail: batkhin@gmail.com, abruno@keldysh.ru

Abstract. Here we demonstrate two new methods of solution of polynomial equations, based on constructing a convex polygon, and provide description of corresponding software. The first method allows to find approximate roots of a polynomial by means of the Hadamard polygon. The second one allows to compute branches of an algebraic curve near its singular point and near infinity by means of the Newton polygon and to draw sketches of real algebraic curves in the plane. Computer algebra algorithms are specified, which allow to investigate any complex cases.

Keywords: convex polygon, polynomial roots, algebraic curve, computer algebra software

1. Introduction

Here we present two new methods for solving polynomial equations based on the construction of a convex polygon from a polynomial. The first method allows one to find approximate roots of a polynomial using the Hadamard broken line (section 3). The second method allows one to find branches of an algebraic curve near its singular point and near infinity with the Newton polygon (section 4). It also allows one to construct sketches of real algebraic curves in the plane. These methods can be generalized to higher dimensions [1].

All algorithms are provided with descriptions of their software implementation in various computer algebra systems.

New points of this work are the following:

- the concept of the cone of the problem is actively used ;
- the application of Newton's polygon to find branches of a curve near infinity is given;
- the theory of Hadamard's broken line method is given;
- computer algebra software is discussed for all algorithms.

For extended version of this work with many examples, listings of program for computer algebra systems `Maple`, `Sympy`, `Mathematica` and for detailed references see [2].

2. Polyhedron and normal cone

Let in $\mathbb{R}^n$ be given several points $\{Q_1, \dots, Q_k\} = \mathbf{S}$. Their convex hull

$$\Gamma(\mathbf{S}) = \left\{ Q = \sum_{i=1}^{k} \mu_i Q_i, \mu_i \geqslant 0, \sum \mu_i = 1 \right\}$$

is a polyhedron. Its boundary $\partial\Gamma$ consists of vertices $\Gamma_j^{(0)}$, edges $\Gamma_j^{(1)}$ and faces $\Gamma_j^{(d)}$ of different dimensions $d : 1 < d \leqslant n-1$. If the real n-vector $P = (p_1, \ldots, p_n)$ is given, then the maximum and minimum of the scalar product $\langle P, Q\rangle = p_1q_1 + \ldots + p_nq_n$ on $\mathbf{S}$ are reached at points Q_i that lie on the boundary $\partial\Gamma$. For each boundary element $\Gamma_j^{(d)}$ (including vertices $\Gamma_j^{(0)}$ and edges $\Gamma_j^{(1)}$), we identify the set of vectors P whose maximum $\langle P, Q\rangle$ is reached on points $Q_i \in \Gamma_j^{(d)}$. This will be its *normal cone*

$$\mathbf{U}_j^{(d)} = \left\{P : \langle P, Q'\rangle = \langle P, Q''\rangle > \langle P, Q'''\rangle \text{ for } \ Q', Q'' \in \Gamma_j^{(d)}, Q''' \in \Gamma\backslash\Gamma_j^{(d)}\right\}.$$

The vector P lies in the space $\mathbb{R}_*^n$, dual to the space $\mathbb{R}^n$.

Let us be interested not in the whole boundary $\partial\Gamma$, but only a part of it corresponding to some set $\mathcal{K}$ of directions P. Then let us call the set $\mathcal{K}$ the *cone of the problem*. It is not necessarily convex. By $\partial\Gamma(\mathcal{K})$ we denote the part of the boundary $\partial\Gamma$ for whose elements $\Gamma_j^{(d)}$ their normal cones $\mathbf{U}_j^{(d)}$ intersect with the cone of problem $\mathcal{K}$. Let us call $\partial\Gamma(\mathcal{K})$ as *boundary of the problem*.

Software for convex hull and normal cones computation

The `Qhull` package is used in many application software packages, both commercial and free. The main feature of the package is that the calculations are performed using real numbers rather than in the field of rational numbers, which is convenient when working with the Hadamard polyhedron. When calculating the Newton polyhedron, additional steps are required to bring the results of the calculations to rational values.

Since the 2015 version, the `Maple` computer algebra system includes the `PolyhedralSets` package. In this package all calculations are performed in the field of rational numbers, which somewhat simplifies its use for the study of the Newton polyhedron, but makes it useless when working with the Hadamard polyhedron. Note that `PolyhedralSets` has extremely low performance compared to `Qhull`.

3. Hadamard broken line method

Let us describe a new method for computing approximate values of roots of the polynomial

$$f_m(x) = \sum_{k=0}^{m} a_k x^k. \tag{1}$$

To do this, the points in the real plane q_1, q_2 are plotted $\check{Q}_k = (q_1, q_2) = (k, \ln|a_k|)$, where $\ln 0 = -\infty$, $k = 0, \ldots, m$, forming the *supersupport* $\check{\mathbf{S}} = \left\{\check{Q}_0, \ldots, \check{Q}_m\right\}$, and their convex hull is constructed $\Gamma(\check{\mathbf{S}}) = \left\{\check{Q} = \sum\limits_{k=0}^{m} \mu_k \check{Q}_k, \mu_k \geqslant 0, \sum_{k=0}^{m} \mu_k = 1\right\} \stackrel{\text{def}}{=} \mathbf{H}(f_m)$, which is called *Hadamard's polygon* [3] (Hadamard, 1893). The boundary $\partial\mathbf{H}$ is a broken line. Each edge $\Gamma_j^{(1)}$ and vertex $\Gamma_j^{(0)}$ of this boundary $\partial\mathbf{H}$ corresponds to a boundary subset $\mathbf{S}_j^{(d)}$ of points $\check{Q}_k$ lying on $\Gamma_j^{(d)}$, and the truncated polynomial

$$\check{f}_j^{(d)}(x) = \sum a_k x^k \text{ over } \check{Q}_k \in \mathbf{S}_j^{(d)}. \tag{2}$$

If $\Gamma_j^{(d)}$ is a vertex ($d = 0$), then the truncated polynomial (2) is a monomial that has no nonzero root. If $\Gamma_j^{(d)}$ is an edge ($d = 1$), then the truncated polynomial (2) has nonzero

roots, which give approximate values for the roots of the full polynomial (1). Except very special cases, the truncated polynomials (2) are significantly simpler than the original polynomial (1), and their roots are easier to compute.

Since the vector $(p_1, 1)$ lies in the upper half-plane of the dual plane $\mathbb{R}^2_*$, the cone of the problem here $\mathcal{K} = \{P = (p_1, p_2) : p_2 > 0\}$, i.e. this is the upper half-plane. It corresponds to the upper part of the boundary $\partial\mathbf{H}$. It will be called *Hadamard's broken line* and denoted by $\widetilde{\mathbf{H}}$. Examples with successful application of the Hadamard broken line method see in [1, 2].

4. Plane algebraic curve

Let $f(x_1, x_2)$ be a polynomial with real or complex coefficients. The set of solutions x_1, x_2 of the equation

$$f(x_1, x_2) = 0 \tag{3}$$

in $X = (x_1, x_2) \in \mathbb{R}^2$ or $\mathbb{C}^2$ is called a *plane algebraic curve* $\mathcal{F}$.

A point $X = X^0$, $f(X^0) = 0$ is called the *simple* point of a curve $\mathcal{F}$ if the vector $(\partial f/\partial x_1, \partial f/\partial x_2)$ in it is nonzero. Otherwise, the point X^0 is called *singular*. By a shift, move the point X^0 to the origin of coordinates.

For local analysis of a simple point one can use

Theorem 1 (Cauchy). *If at $X^0 = 0$ the derivative $\partial f/\partial x_i \neq 0$, then all solutions to the equation* (3) *near point $X^0 = 0$ are contained in the expansion*

$$x_i = \sum_{k=1}^{\infty} b_k x_j^k, \tag{4}$$

where b_k — constants, and $j = 3 - i$.

Further consider local analysis of the singular point $X^0 = 0$ and points at infinity. Let's write the polynomial $f(X)$ as

$$f(X) = \sum f_Q X^Q \text{ over } Q \geqslant 0, \quad Q \in \mathbb{Z}^2, \tag{5}$$

where $X = (x_1, x_2)$, $Q = (q_1, q_2)$, $X^Q = x_1^{q_1} x_2^{q_2}$, $f_Q \in \mathbb{C}$ are constants. Let $\mathbf{S}(f) = \{Q : f_Q \neq 0\} \subset \mathbb{R}^2$. The set $\mathbf{S}$ is called the *support* of the polynomial $f(X)$. Let it consist of points $Q_1, \ldots, Q_k$. The convex hull of the support $\mathbf{S}(f)$ is the set

$$\Gamma(\mathbf{S}) = \left\{ Q = \sum_{j=1}^{k} \mu_j Q_j,\ \mu_j \geqslant 0,\ \sum_{j=1}^{k} \mu_j = 1 \right\} \stackrel{\text{def}}{=} \mathbf{N}(f),$$

which is called *Newton's polygon*. The boundary $\partial\mathbf{N}(f)$ consists of vertices $\Gamma_j^{(0)}$ and edges $\Gamma_j^{(1)}$, where j is the number.

Each generalized face $\Gamma_j^{(d)}$ corresponds to: its *boundary subset* $\mathbf{S}_j^{(d)} = \mathbf{S} \cap \Gamma_j^{(d)}$, its a *truncated* polynomial $\hat{f}_j^{(d)}(X) = \sum f_Q X^Q$ by $Q \in \mathbf{S}_j^{(d)}$ and its *normal cone* $\mathbf{U}_j^{(d)} = \{P : \langle P, Q' \rangle = \langle P, Q''' \rangle > \langle P, Q'' \rangle, Q', Q'' \in \Gamma_j^{(d)}, Q''' \in \Gamma \backslash \Gamma_j^{(d)}\}$, where $P = (p_1, p_2) \in \mathbb{R}^2_*$, and the plane $\mathbb{R}^2_*$ is conjugate to the plane $\mathbb{R}^2$.

We will look for solutions to the equation

$$f(X) \stackrel{\text{def}}{=} \sum_{Q \in \mathbf{S}} f_Q X^Q = 0 \tag{6}$$

in the form of expansion

$$x_2 = b_1 x_1^{p_1} + b_2 x_1^{p_2} + b_3 x_1^{p_3} + \cdots , \tag{7}$$

where f_Q, $b_k = \text{const} \in \mathbb{C}$, $Q \in \mathbb{R}^2$, $p_k = \text{const} \in \mathbb{R}$, $\omega p_k > \omega p_{k+1}$. In these expansions, the exponents of degree p_k increase with k if $x_1 \to 0$ ($\omega = -1$), and decrease if $x_1 \to \infty$ ($\omega = 1$).

Theorem 2. *For solutions* (7) *of equation* (6), *the truncated solution* $x_2 = b_1 x_1^{p_1}$ *is the solution to the truncated equation*

$$\hat{f}_j^{\,(1)}(X) = 0, \tag{8}$$

corresponding to the boundary element $\Gamma_j^{(1)}$ *with the external normal vector* $\omega(1, p_1) \in \mathbf{U}_j^{(1)}$.

The truncated equation (8) uniquely determines the sign of ω and the index of degree p_1. If in the sum (5) all vector indexes of degree $Q = (q_1, q_2)$ have rational components q_1 and q_2, then the index p_1 is rational. For the coefficient b_1 we obtain the algebraic equation $\hat{f}_j^{(1)}(1, b_1) = 0$.

Theorem 3. *For the polynomial equation* (6), *all solutions* $x_2(x_1)$ *are expanded into a series of the form* (7), *where all exponents of degree* p_k *are rational numbers with a common denominator.*

For the neighborhood of the point $X = 0$, Theorem 3 is that of V. Puiseau, 1850, i.e., for the cone problem $\mathcal{K} = \{P = (p_1, p_2) : p_1, p_2 < 0\}$. The corresponding part (lower left) of the boundary $\partial\mathbf{N}$ is called Newton's *broken line*. The expansions of Theorem 1 converge, so all expansions (7) for solutions of polynomial equations (6) also converge.

Software for plane curve investigation

The `Maple` system has an excellent package `algcurves` that allows to study planar algebraic curves: build their sketches with high precision, calculate their genus, find singular points; for curves of genus 0 find rational parameterization, for elliptic curves bring to Weierstrass normal form. The package allows to construct a sketch of the real curve $f(x, y) = 0$ by numerical integration of the differential equation $f'_x + f'_y y' = 0$ for some set of initial conditions defined by points in which at least one of the partial derivatives of the function $f(x, y)$ is equal to zero.

Since version 12, `Wolfram Mathematica` has included an `AsymptoticSolve` procedure that implements an asymptotic representation of solutions to equations or systems of equations (not necessarily algebraic) in the form of either Taylor, Laurent, or Puiseau series near finite or infinite points. If the point is singular, the procedure tries to calculate the asymptotic expansions of all branches. In this case we can specify that we should restrict ourselves to real expansions only.

References

1. *Bruno A.D.* Algorithms for solving an algebraic equation. Programming and Computer Software. 2018. Vol. 44, No. 6. P. 533–545. DOI: 10.1134/S0361768819100013.
2. *Bruno A.D., Batkhin A.B.* Introduction to nonlinear analysis of algebraic equations. Keldysh Institute Preprints. 2020. No. 87. 31 p. (in Russian) DOI: 10.20948/prepr-2020-87.
3. *Hadamard J.* Etude sur les propriétés des fonctions entières et en particulier d'une fonction considérée par Riemann. Journal de mathématiques pures et appliquées 4^e série. 1893. Vol. 9. P. 171–216.

Role of Monomial Orderings in Efficient Gröbner Basis Computation in Parameter Identifiability Problem

M. Bessonov[1], I. Ilmer[2], T. Konstantinova[3], A. Ovchinnikov[2,3,4], G. Pogudin[5]

[1]*CUNY NYC College of Technology, Department of Mathematics, New York, USA*

[2]*Ph.D. Program in Computer Science, CUNY Graduate Center, New York, USA*

[3]*Department of Mathematics, CUNY Queens College, New York, USA*

[4]*Ph.D. Program in Mathematics, CUNY Graduate Center, New York, USA*

[5]*LIX, CNRS, École Polytechnique, Institute Polytechnique de Paris, France*

Abstract. We present empirical runtime and memory use improvements for computing Gröbner bases of ideals generated by polynomials that appear in solving the parameter identifiability problems for ODE models by the SIAN algorithm. Such differential-algebraic systems may also occur some other in prolongation-based algorithms and efficiently computing Gröbner bases can be critical. The main speed-up is achieved by automatically choosing problem-specific monomial orderings.

Keywords: structural parameter identifiability, Gröbner bases, monomial orderings

Structural identifiability properties of ODE systems determine whether a parameter value can be recovered form experimental data. If recovered value is unique, we say that parameters are uniquely identifiable. In case of finitely many values, we say that parameters are locally identifiable. If there are infinitely many such values, a parameter is said to be non-identifiable.

Structural identifiability queries can be solved using methods of differential algebra. Among available tools for determining parameter identifiability such as [2, 3, 7, 8, 10, 11]. In this work, we focus our attention on SIAN, Structural Identifiability Analyzer presented and detailed in [6] and [7]. This package has multiple notable advantages most notably a typical computational speed advantage when compared to other similar tools, [7].

SIAN uses Gröbner bases to analyze structural identifiability of the input differential system. Gröbner bases can have degrees that are doubly exponential in the input size [4] thus leading to significant resource requirements for computing them. While in practice these computations can be efficient, especially with resent developments in hardware and software [1, 5], one may nonetheless encounter computationally hard problems that may take up to several days to resolve.

It is crucial to seek empirical approaches that can lead to improvements in computing Göbner bases. For instance, it is well-known that a choice of monomial ordering could significantly impact the speed and that reverse degree lexicographic order of variables is often the most efficient in practice [12].

In this work, we show several observations on monomial orderings for polynomial systems produced by the SIAN algorithm from ordinary differential equations. Similar polynomial systems arise in other prolongation-based algorithms, see for instance [9]. These observations, if taken into account, significantly reduce runtime and memory use of the SIAN and its Gröbner basis calculation compared to degree reverse lexicographic ordering.

Acknowledgments

This work was partially supported by the NSF grants CCF-1564132, CCF-1563942, DMS-1760448, DMS-1853650, and DMS-1853482, and by the Paris Île-de-France Region.

References

1. *Bardet M., Faugère J.-C., and Salvy B.* On the complexity of the F5 Gröbner basis algorithm. Journal of Symbolic Computation. 2015. Vol. 70. P. 49–70.
2. *Bellu G., Saccomani M.P., Audoly S., and D'Angiò L.* DAISY: A new software tool to test global identifiability of biological and physiological systems. Computer Methods and Programs in Biomedicine. 2007. Vol. 88(1). P. 52–61.
3. *Chis O., Banga J. R., and Balsa-Canto E.* GenSSI: a software toolbox for structural identifiability analysis of biological models. Bioinformatics. 2011. Vol. 27(18). P. 2610–2611.
4. *Dubé T. W.* The structure of polynomial ideals and Gröbner bases. SIAM Journal on Computing. 1990. Vol. 19(4). P. 750–773.
5. *Eder C. and Faugère J.-C.* A survey on signature-based algorithms for computing Gröbner bases. Journal of Symbolic Computation. 2017. Vol. 80. P. 719–784.
6. *Hong H., Ovchinnikov A., Pogudin G., and Yap C.* Global identifiability of differential models. Communications on Pure and Applied Mathematics. 2020. Vol. 73(9). P. 1831–1879.
7. *Hong H., Ovchinnikov A., Pogudin G., and Yap C.* SIAN: software for structural identifiability analysis of ODE models. Bioinformatics. 2019. Vol. 35(16). P. 2873–2874.
8. *Ligon T. S. et al.* GenSSI 2.0: multi-experiment structural identifiability analysis of SBML models. Bioinformatics. 2018. Vol. 34(8). P. 1421–1423.
9. *Ovchinnikov A., Pogudin G., and Vo T.N.* Bounds for elimination of unknowns in systems of differential-algebraic equations. International Mathematics Research Notices, 2021, rnaa302 1073-7928.
10. *Saccomani M.P., Bellu G., Audoly S., and D'Angió L.* A new version of DAISY to test structural identifiability of biological models. International Conference on Computational Methods in Systems Biology. 2019. P. 329–334.
11. *Villaverde A.F., Barreiro A., and Papachristodoulou A.* Structural identifiability of dynamic systems biology models. PLoS Computational Biology. 2016. Vol. 12(10). e1005153.2
12. *Cox D., Little J., O'Shea D., Sweedler M.* Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra. Springer Science & Business Media, 2013.

Amoebas of Multivariate Hypergeometric Polynomials

D.V. Bogdanov[1], T.M. Sadykov[2]
[1]*Moscow Center of Technological Modernization of Education, Russia*
[2]*Plekhanov Russian University*
125993, Moscow, Russia.
e-mail: BogdanovDV1@edu.mos.ru

Abstract. With any integer convex polytope $P \subset \mathbb{R}^n$ we associate a multivariate hypergeometric polynomial whose set of exponents is $\mathbb{Z}^n \cap P$. This polynomial is defined uniquely up to a constant multiple and satisfies a holonomic system of partial differential equations of Horn's type. We prove that under certain nondegeneracy conditions the zero locus of any such polynomial is optimal in the sense of [4], i.e., that the topology of its amoeba [6] is as complex as it could possibly be. Using this, we derive optimal properties of several classical families of multivariate hypergeometric polynomials.

Keywords: polynomial amoeba, hypergeometric polynomial.

Zeros of hypergeometric functions are known to exhibit highly complicated behavior. The univariate case has been extensively studied both classically (see, e.g., [5, 7]) and recently (see [1, 2, 10] and the references therein). Already the distribution of zeros of polynomial instances of the simplest non-elementary hypergeometric function ${}_2F_1\left(a, b; c; x\right)$ is far from being clear. When one of the parameters a, b equals a nonpositive integer, say $a = -d$, the series representing ${}_2F_1$ terminates and the hypergeometric function is a polynomial of degree d in x (see [1]). By letting the parameters a, b, c assume values in various ranges, one can obtain a wide variety of shapes. Some of them are highly regular (see, e.g., Fig. 1) while other are nearly chaotic.

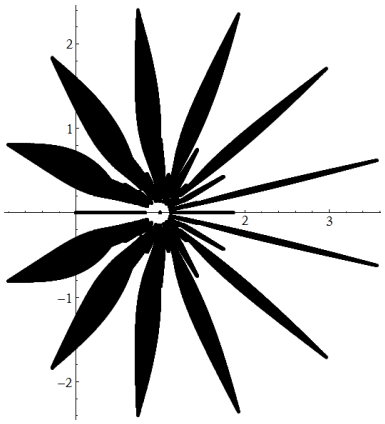


Figure 1: The hypergeometric aster: zeros of the polynomials ${}_2F_1\left(-12, b; c; x\right)$ with $b, c \in \left\{\frac{k}{1000} : k = 100, \ldots, 4000\right\}$

In the talk, we will consider a definition of a multivariate hypergeometric polynomial in $n \geq 2$ complex variables that is coherent with the properties of classical hypergeometric polynomials. This polynomial is defined by an integer convex polytope $P \subset \mathbb{R}^n$, its set of exponents is $\mathbb{Z}^n \cap P$. For this polynomial to be "truly hypergeometric" in the sense made precise below, we need to assume that any pair of points in $\mathbb{Z}^n \cap P$ can be connected by a polygonal line with unit sides and integer vertices. This assumption does not affect the

generality of the results since any polytope that does not satisfy this condition gives rise to a finite number of hypergeometric polynomials that can be considered independently.

The following definition is central and brings together the intrinsic properties of the classical families of hypergeometric polynomials: the denseness, convexity, and irreducibility of the support, as well as the property of being a solution to a suitable system of linear differential equations with polynomial coefficients.

Definition 1. For $n \geq 2$ let $P \subset \mathbb{R}^n$ be an integer convex polytope such that any two points in $P \cap \mathbb{Z}^n$ can be connected by a polygonal line with unit sides. Let $\langle B_i, s\rangle + c_i = 0$, $i = 1, \ldots, q$ be the equations of the hyperplanes containing the faces of P with B_i being the outer normal to P at the respective face with integer relatively prime components.

The polynomial

$$\sum_{s \in P \cap \mathbb{Z}^n} \frac{x^s}{\prod_{i=1}^{q} \Gamma(1 - \langle B_i, s\rangle - c_i)}$$

will be called *the hypergeometric polynomial* defined by the polytope P.

The hypergeometric polynomial associated with the polytope P is defined uniquely up to a constant multiple and satisfies a holonomic system of partial differential equations of Horn's type [8, 9]. We will prove that under certain nondegeneracy conditions any such polynomial is optimal in the sense of [4]. Generally speaking, this means that the topology of the amoeba [4, 6] of such a polynomial is as complicated as it could possibly be. This property is the multivariate counterpart of the property of having different absolute values of the roots for a polynomial in a single variable. In the talk, we will show various families of classically known multivariate polynomials to be optimal (possibly after a monomial change of variables): a biorthogonal basis in the unit ball, certain polynomial instances of the Appel F_1 function, bivariate Chebyshev polynomials of the second kind etc.

References

1. *Dominici D., Johnston S.J., and Jordaan K.* Real zeros of ${}_2F_1$ hypergeometric polynomial. Journal of Comput. and Appl. Math. 2013. Vol. 247. P. 152–161.
2. *Driver K.A. and Johnston S.J.* Asymptotic zero distribution of a class of hypergeometric polynomials. Quaestiones Mathematicae. 2007. Vol. 30, No. 2. P. 219–230.
3. *Dunkl C.F. and Xu Y.* Orthogonal Polynomials of Several Variables. Cambridge University Press, 2014.
4. *Forsberg M., Passare M., and Tsikh A.K.* Laurent determinants and arrangements of hyperplane amoebas. Adv. Math. 2000. Vol. 151. P. 45–70.
5. *Klein F.* Über die Nullstellen der hypergeometrischen Reihe. (German) Math. Ann. 1890. Vol. 37, No. 4. P. 573–590.
6. *Mikhalkin G.* Real algebraic curves, the moment map and amoebas. Ann. Math. 2000. Vol 151 P. 309–326.
7. *Nørlund N.E.* Hypergeometric functions. Acta Math. 1955. Vol. 94. P. 289–349.
8. *Sadykov T.M.* On a multidimensional system of hypergeometric differential equations. Siberian Math. J. 1998. Vol. 39. P. 986–997.
9. *Sadykov T.M. and Tanabe S.* Maximally reducible monodromy of bivariate hypergeometric systems. Izv. Math. 2016. Vol. 80, No. 1. P. 221–262.

10. *Zhou J.-R., Srivastava H.M., and Wang Z.-G.* Asymptotic distribution of the zeros of a family of hypergeometric polynomials. Proc. of the AMS. 2012. Vol. 140, No. 7. P. 2333–2346.

Symbolic Integration of Differential Forms

Shaoshi Chen[1]

[1]*KLMM, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, 100190, China*

e-mail: schen@amss.ac.cn

Abstract. Symbolic integration of differential forms is a higher dimensional analogue of symbolic integration of elementary functions. This note will present a Liouville-style theorem for integration of closed differential forms with rational-function coefficients.

Keywords: Closed forms, elementary functions, symbolic integration

The integration problem of elementary functions is as old as calculus and differentiation, whose history can be traced back at least to the work of Euler and others on elliptic integrals [2]. Abel in his great parisian memoir in 1826 studied the integrals of general algebraic functions, which are now called abelian integrals [1]. Liouville in the period of 1833-1841 established the fundamental theorem of elementary integration that is if the integral of an elementary function in a field K which is an elementary extension of $\mathbb{C}(x)$ is still an elementary function over $\mathbb{C}(x)$, then it must be the sum of an elementary function in K and a linear combination of logarithms of elementary functions in K (see [6, Chapter IX] for a detailed historical overview). In 1948, Ritt wrote a book on integration in finite terms that presents Liouville's theory of elementary integration systematically [8]. Based on Liouville's theorem and some developments in differential algebra [9], Risch in 1970 finally solved the integration problem of elementary functions by giving a complete algorithm [7]. After Risch's work, more efficient algorithms have been given due to the emerging developments in symbolic computation [5, 10, 3, 4], which makes Symbolic Integration as an active topic in symbolic computation. The long-term goal of this project is developing theory and algorithms for symbolic integration of differential forms, which can be viewed as a higher dimensional analogue of symbolic integration of elementary functions.

Let k be an algebraically closed field of characteristic zero and $K = k(x_1, \ldots, x_m)$ be the field of rational functions in $x_1, \ldots, x_m$ over k. Let ∂_i denote the usual partial derivation ∂/∂_{x_i} on K which satisfies that $\partial_i\partial_j(f) = \partial_j\partial_i(f)$ for all $f \in K$ and $\partial_i(c) = 0$ for all $c \in k(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m)$. Then $(K, \{\partial_1, \ldots, \partial_m\})$ forms a differential field. In a differential field extension E of K, an element $t \in E$ is said to be *logarithmic* over K if there exists $u \in K \setminus \{0\}$ such that $\partial_i(t) = \partial_i(u)/u$ (we can symbolically write that $t = \log(u)$). The module of all differential forms of E over k is denoted by $\Omega_{E/k}$. This module can also be viewed as a vector space over k. Any derivation ∂ on E can be uniquely lifted as a linear map ∂^* of $\Omega_{E/k}$ such that $\partial^*(fdg) = \partial(f)dg + fd\partial(g)$. By an abuse of notation, we use the same symbol ∂ for the lifted operator ∂^*. We can write any p-form $\omega \in \Omega^p_{E/k}$ as

$$\omega = \sum_{1\le i_1<\cdots<i_p\le m} f_{i_1,\ldots,i_p} dx_{i_1}\cdots dx_{i_p}, \quad \text{where } f_{i_1,\ldots,i_p} \in E.$$

The exterior derivation is defined as

$$d\omega = \sum_{1\le i_1<\cdots<i_p\le m} \left(\sum_{j=1}^{m} \partial_j(f_{i_1,\ldots,i_p})dx_j\right) dx_{i_1}\cdots dx_{i_p} \in \Omega^{p+1}_{E/k}.$$

A p-form $\omega \in \Omega^p_{E/k}$ is said to be *closed* if $d\omega = 0$ and is said to be *exact* if there exists $\eta \in \Omega^{p-1}_{E/k}$ such that $\omega = d\eta$. By definition, any exact form is always closed. The fundamental problem on differential forms is to decide whether a given closed form is exact or not.

Let us first study the integration of closed 1-forms with coefficients in $K = F(x_m)$ with $F = k(x_1, \ldots, x_{m-1})$. The following lemma was first claimed without a proof in Abel's "Parisian Mémoire" in 1826 [1].

Lemma 1. *Let $\omega = f_1 dx_1 + \cdots + f_m dx_m$ be a closed 1-form with $f_i \in K$. Then $\omega = dg$ for some g of the form*

$$g = a + \sum c_i \log(b_i),$$

where $a \in K$, $c_i \in k$, and $b_i \in F[x_m]$ are pairwise coprime polynomials of positive degree.

Remark 2. *If the field k is not algebraically closed, then the above constants c_i are in $\bar{k}$ and $b_i \in F(c_i)[x_m]$.*

We recall some notation from the first chapter of the book [11]. For any $f \in E$, we define

$$d_s(f) = \partial_1(f)dx_1 + \cdots + \partial_s(f)dx_s \quad \text{and} \quad d^s(f) = \partial_s(f)dx_s \quad \text{for } s \in \{1, 2, \ldots, m\},$$

We can extend d_p and d^s to the module $\Omega_{E/k}$ as d. By definition, we have the relation

$$d = d_{m-1} + d^m.$$

Let $\omega \in \Omega^p_{E/k}$, we can always decompose ω into

$$\omega = \omega_{m-1} + \omega^m,$$

where all monomials $f_{i_1,\ldots,i_p} dx_{i_1} \cdots dx_{i_p}$ of ω_{m-1} do not involve dx_m and $\omega^m = \mu dx_m$ with $\mu \in \Omega^{p-1}_{E/k}$ and free of dx_m. To abbreviate the expression, we will write $f_I dx_I$ for the monomial $f_{i_1,\ldots,i_p} dx_{i_1} \cdots dx_{i_p}$. We now extend Abel's claim to the case of general closed p-forms.

Theorem 3. *Let ω be a closed p-form with coefficients in $K = k(x_1, \ldots, x_m)$. Then*

$$\omega = d_p(\Psi^p) + \cdots + d_m(\Psi^m),$$

where for each $i = p, \ldots, m$, the coefficients $f_{i,j}$ of Ψ^i are of the form

$$f_{i,k} = b_{i,k,0} + \sum_j c_{i,k,j} \log b_{i,k,j}$$

with $b_{i,k,0} \in k(x_1, \ldots, x_i)$, $c_{i,k,j} \in \overline{k(x_1, \ldots, x_{i-1})}$ and $b_{i,k,j} \in k(x_1, \ldots, x_{i-1})(c_{i,k,j})[x_i]$.

Remark 4. *The above theorem says that the claim of Abel can be extended to the general p-form if one allows more general coefficients appear in the linear combination of logarithms with a recursive pattern.*

The following example shows that all of the combination factors in the logarithmic part may not be constants, which means the above theorem is optimal in some sense.

[1] Thanks to David A. Cox for asking me to give a detailed proof of this result and also proposed to study the general problem on p-forms.

Example 5. *Let* $K = \mathbb{C}(x, y)$ *and* $\omega = 1/(xy)dxdy$. *If* $\omega = d(Ady - Bdx)$ *with*

$$A = a_0 + \sum \lambda_i \log a_i \quad \textit{and} \quad B = b_0 + \sum \mu_i \log b_i,$$

where $a_i, b_i \in \mathbb{C}(x, y)$ *and* λ_i, μ_i *are constants in* $\mathbb{C}$, *then* $1/(xy) = \partial_x(A) + \partial_y(B)$. *Taking the residues at* $y = 0$ *(viewing the functions in* y *over* $\overline{\mathbb{C}(x)}$*) on the both sides of the above identity leads to*

$$\frac{1}{x} = \partial_x(res_{y=0}(a_0)) + c \quad \textit{for some constant } c \in \mathbb{C},$$

which contradicts with the fact that $\log(x)$ *is not algebraic over* $\mathbb{C}(x)$. *But we have*

$$\omega = d\left(\frac{1}{y}\log(x)dy\right) = d\left(-\frac{1}{x}\log(y)dx\right).$$

References

1. *Niels Henrik Abel.* Œuvres complètes. Tome I. Éditions Jacques Gabay, Sceaux, 1992. Edited and with a preface by L. Sylow and S. Lie, Reprint of the second (1881) edition.
2. *Shreeram S. Abhyankar.* Historical ramblings in algebraic geometry and related algebra. Amer. Math. Monthly. 1976. Vol. 83(6). P. 409–448.
3. *Manuel Bronstein.* Integration of elementary functions. J. Symbolic Comput. 1990. Vol. 9(2). P. 117–173.
4. *Manuel Bronstein.* Symbolic Integration I: Transcendental Functions. Springer-Verlag, Berlin, 2005.
5. *James Harold Davenport.* On the Integration of Algebraic Functions, volume 102 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1981.
6. *Jesper Lützen.* Joseph Liouville 1809–1882: master of pure and applied mathematics, volume 15 of Studies in the History of Mathematics and Physical Sciences. Springer-Verlag, New York, 1990
7. *Robert H. Risch.* The solution of the problem of integration in finite terms. Bull. Amer. Math. Soc. 1970. Vol. 76. P. 605–608.
8. *Joseph Fels Ritt.* Integration in Finite Terms. Liouville's Theory of Elementary Methods. Columbia University Press, New York, N. Y., 1948.
9. *Joseph Fels Ritt.* Differential Algebra. American Mathematical Society Colloquium Publications, Vol. XXXIII. American Mathematical Society, New York, N. Y., 1950.
10. *Barry M. Trager.* Integration of Algebraic Functions. PhD thesis, MIT, 1984.
11. *Steven H. Weintraub.* Differential forms. Elsevier/Academic Press, Amsterdam, second edition, 2014. Theory and practice.

A Maple Implementation of the Finite Element Method for Solving Metastable State Problems for Systems of Second-Order Ordinary Differential Equations

G. Chuluunbaatar[1,2], A.A. Gusev[1,3], V.L. Derbov[4], S.I. Vinitsky[1,2], O. Chuluunbaatar[1,5]

[1] *Joint Institute for Nuclear Research, Dubna, Russia*

[2] *Peoples' Friendship University of Russia (RUDN University), Moscow, Russia*

[3] *Dubna State University, Dubna, Russia*

[4] *N.G. Chernyshevsky Saratov National Research State University, Saratov, Russia*

[5] *Institute of Mathematics and Digital Technology, Mongolian Academy of Sciences, Ulaanbaatar, Mongolia*

e-mail: galmandakh@mail.ru, gooseff@jinr.ru, derbovvl@gmail.com, vinitsky@theor.jinr.ru, chuka@jinr.ru

Abstract. We present a new algorithm for systems of second-order ordinary differential equations to calculate metastable states with complex eigenvalues of energy or to find bound states with homogeneous boundary conditions depending on a spectral parameter. The boundary-value problems is discretized by means of the FEM using the Hermite interpolation polynomials with arbitrary multiplicity of the nodes, which preserves the continuity of derivatives of the desired solutions. For the solution of the relevant algebraic problems the Newton iteration scheme is implemented.

Keywords: finite element method, interpolation Hermite polynomials, boundary-value problem, metastable state, system of ordinary differential equations, Newton iteration scheme

1. Statement of the problem

The proposed approach implemented as program KANTBP 5M is intended for solving BVPs for systems of the ODEs with respect to unknown functions $\mathbf{\Phi}(z)=(\Phi_1(z),\ldots,\Phi_N(z))^T$ of independent variable $z\in\Omega(z^{\min},z^{\max})$ numerically using the FEM (see for details [1,2]):

$$(\mathbf{D}-E\,\mathbf{I})\,\mathbf{\Phi}(z)\equiv\left(-\frac{1}{f_B(z)}\mathbf{I}\frac{d}{dz}f_A(z)\frac{d}{dz}+\mathbf{V}(z)-E\,\mathbf{I}\right)\mathbf{\Phi}(z)=0. \tag{1}$$

Here $f_B(z)>0$ and $f_A(z)>0$ are continuous or piecewise continuous positive functions, $\mathbf{I}$ is the unit matrix, $\mathbf{V}(z)$ is a symmetric matrix, $V_{ij}(z)=V_{ji}(z)$. The elements of these matrices are continuous or piecewise continuous real or complex-valued coefficients from the Sobolev space $\mathcal{H}_2^{s\geq 1}(\Omega)$, providing the existence of nontrivial solutions $\mathbf{\Phi}(z)$ subjected to homogeneous Dirichlet, Neumann or Robin BCs at the boundary points of the interval $z\in\{z^{\min},z^{\max}\}$ with given symmetric real or complex-valued $N\times N$ matrix $\mathbf{G}(z)$

$$\mathbf{\Phi}(z^t)=0,\qquad \lim_{z\to z^t}f_A(z)\frac{d}{dz}\mathbf{\Phi}(z)=0,\qquad \lim_{z\to z^t}\mathbf{I}\frac{d}{dz}\mathbf{\Phi}(z)=\mathbf{G}(z^t)\mathbf{\Phi}(z^t), \tag{2}$$

where the superscript $t=\min,\max$ labels the boundary points of the interval.

Table. Eigenvalues E_i, $i=1,2,3$ of bound states and E_i^M, $i=1,...,4$ of metastable states obtained by solving the BVP with Neumann BC (E) and with Robin BC by Newton method (N) and method of matching fundamental solutions (M)

E	-2.12846503065	-0.925565889437	0.835126562953
N	-2.12846503036	-0.925565881437	0.835126980234
M	-2.12846503156	-0.925565883542	0.835126979072
N	$1.35989392876-\imath 0.00016253897$	$2.43040517408-\imath 0.0789059067115$	
M	$1.35989392695-\imath 0.00016253895$	$2.43040517183-\imath 0.0789059070893$	
N	$6.32021061134-\imath 0.00326071312$	$7.50608788873-\imath 0.0194121454599$	
M	$6.32021060910-\imath 0.00326071319$	$7.50608789245-\imath 0.0194121442796$	

Eigenfunctions $\mathbf{\Phi}_m(z)$ obey the normalization and orthogonality conditions

$$(\mathbf{\Phi}_m|\mathbf{\Phi}_{m'}) = \int_{z^{\min}}^{z^{\max}} f_B(z)(\mathbf{\Phi}^{(m)}(z))^T\mathbf{\Phi}^{(m')}(z)dz = \delta_{mm'}.$$

For bound states with real eigenvalues E: $E_1 \le E_2 \le ...$ the Dirichlet or Neumann BC (2) follow from asymptotic expansions. For metastable states with complex eigenvalues $E = \Re E + \imath\Im E$, $\Im E < 0$: $\Re E_1 \le \Re E_2 \le ...$ the Robin BC follow from outgoing wave fundamental asymptotic solutions that correspond to the Siegert outgoing wave BCs [2].

For the set of ODEs (1) with $f_B(z)=f_A(z)=1$ and constant effective potentials $V_{ij}(z)=V_{ij}^{L,R}$ in the asymptotic region, asymptotic solutions $\mathbf{X}_i^{(*)}(z\to\pm\infty)$ are as follows. For bound states:

$$\mathbf{X}_{i_c}^{(c)}(z\to\pm\infty)\to\exp\left(-\sqrt{\lambda_{i_c}^{L,R}-E_i}|z|\right)\mathbf{\Psi}_{i_c}^{L,R},\quad \lambda_{i_c}^{L,R}\ge E,\quad i_c=1,\dots,N,$$

and for metastable states:

$$\mathbf{X}_{i_o}^{(\overrightarrow{\leftarrow})}(z\to\infty)\to\exp\left(+\imath\sqrt{E-\lambda_{i_o}^{L,R}}|z|\right)\mathbf{\Psi}_{i_o}^{L,R},\quad \lambda_{i_o}^{L,R}<\Re E,\quad i_o=1,...,N_o^{L,R},$$

$$\mathbf{X}_{i_c}^{(c)}(z\to\infty)\to\exp\left(-\sqrt{\lambda_{i_c}^{L,R}-E}|z|\right)\mathbf{\Psi}_{i_c}^{L,R},\quad \lambda_{i_c}^{L,R}\ge\Re E,\quad i_c=N_o^{L,R}+1,\dots,N.$$

In the considered case matrix $\mathcal{R}(z^t)$ of logarithmic derivatives for the corresponding Robin BC takes the form

$$\mathcal{R}(z^t) = \mathbf{\Psi}^{L,R}\mathbf{F}^{L,R}\left(\mathbf{\Psi}^{L,R}\right)^{-1},$$

where $\mathbf{F}^{L,R}=\mathrm{diag}(...,\pm\sqrt{\lambda_{i_c}^{L,R}-E},...,\mp\imath\sqrt{E-\lambda_{i_o}^{L,R}},...)$, $\lambda_i^{L,R}$ and $\mathbf{\Psi}_i^{L,R}=\{\Psi_{1i}^{L,R},...,\Psi_{Ni}^{L,R}\}^T$ are solutions of the algebraic eigenvalue problems with matrix $\mathbf{V}^{L,R}$ of dimension $N\times N$ for the entangled channels [4]

$$\mathbf{V}^{L,R}\mathbf{\Psi}_i^{L,R} = \lambda_i^{L,R}\mathbf{\Psi}_i^{L,R},\quad (\mathbf{\Psi}_i^{L,R})^T\mathbf{\Psi}_j^{L,R} = \delta_{ij}.$$

Note that $\lambda_i^{L,R} = V_{ii}^{L,R}$ and $\mathbf{\Psi}_i^{L,R} = \delta_{ji}$, if $V_{i\ne j}^{L,R} = 0$, i.e. in the conventional case of orthogonal channels.

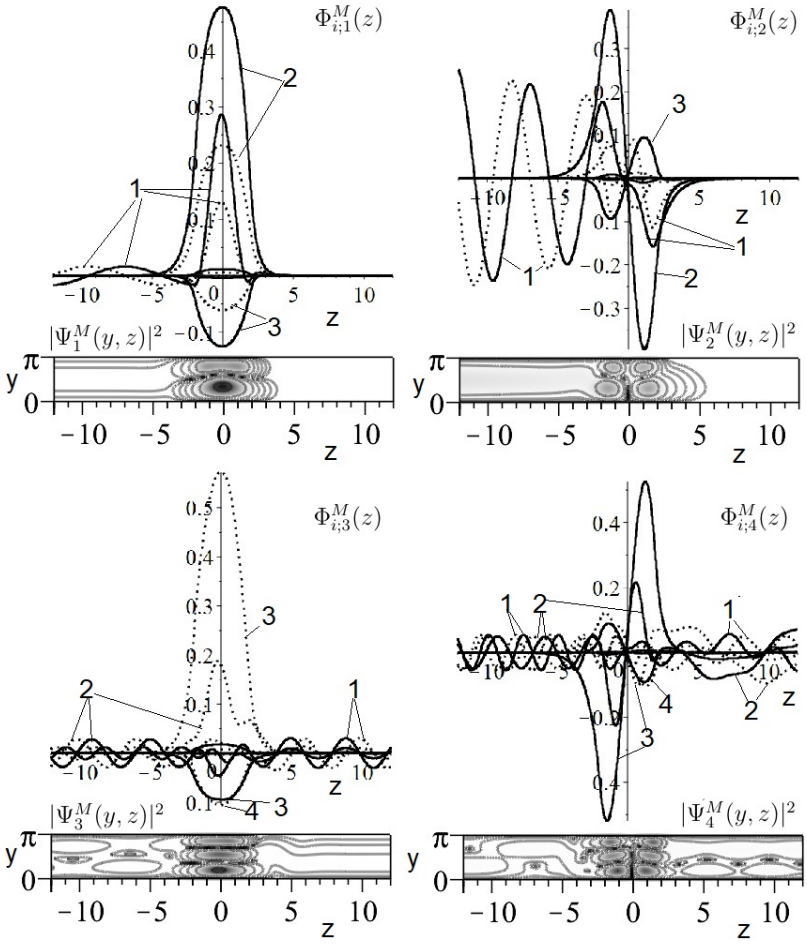


Figure. The real (solid lines) and imagine (dotted lines) parts of components $\Phi^M_{i;n}(z)$ (marked by label i) of metastable state eigenfunction and the corresponding probability density $|\Psi^M_n(y,z)|^2$ (the values $0.25/5^i$, $i=0,...,6$ are marked with isolines).

2. Example.

Consider, e.g., a BVP similar to that of Ref. [3] for the Schrödinger equation in 2D domain Ω_{yz}=$\{y\in(0,\pi), z\in(-\infty,+\infty)\}$, with potential

$$V(y,z)=\{0, z<-2; -2y, |z|\le 2; 2y, z>2\}\,.$$

We seek the solution in the form of expansion $\Psi(y,z)=\sum_{i=1}^{N} B_i(y)\Phi_i(z)$ in a set of basis functions $B_i(y)=\frac{\sqrt{2}}{\sqrt{\pi}}\sin(iy)$, which leads to Eqs. (1) with $f_B(z)=f_A(z)=1$, and effective potentials

$$V_{ij}(z)=i^2\delta_{ij}+\left\{0, z<-2; -2, |z|\le 2; 2, z>2\right\}\times\left\{\pi/2, i=j; 0, \text{ even } i-j; \frac{-8ij}{\pi(i^2-j^2)^2}, \text{ odd } i-j\right\}.$$

For example, let us choose $N=6$. The considered system has sets of threshold energies that differ in the left and right asymptotic regions of the z-axis: $\lambda_i^{(L)}=\{1,4,9,16,25,36\}$ and $\lambda_i^{(R)}=\{3.742260,\ 7.242058,\ 12.216485,\ 19.188688,\ 28.173689,39.286376\}$, respectively. So, we have different numbers of open channels and entangled channels in the right-hand asymptotic region.

Table presents the calculated energies of bound and metastable states. The bound states were calculated on a grid [-25.78125, -18.1875, -13.125, -9.75, -7.5, -6(1)6] built up a geometric

progression of steps in accordance with a slow exponential decay of solutions at $z < -6$ subject to the Neumann BC. The metastable states were found by Newton method on a grid [-4(1)4] with the Robin BC dependent on the eigenvalue. As initial data, the solution obtained on a grid [-2(1)2] with the Neumann BC was taken. In both cases, IHPs of the sixth order $(\kappa_1^{\max}, ..., \kappa_5^{\max}) = (2, 1, 1, 1, 2)$ were used. The computation time was 62 sec for solutions of the eigenvalue problem and 70 sec per iteration when using the Newton method. The calculations were performed by KANTBP 5M code using MAPLE 2019 on PC Intel Pentium 987 2x1.5GHz, 4Gb, 64bit Windows 8. The results obtained by matching the fundamental solutions [3] and FEM are seen to coincide with an accuracy of $10^{-9} \div 10^{-7}$.

The real (solid lines) and imaginary (dottet lines) parts of components $\Phi_{i;n}^M(z)$ of metastable state eigenfunctions $\Psi_n^M(y,z) = \sum_{i=1}^N B_i(y)\Phi_{i;n}^M(z)$ and the corresponding probability density $|\Psi_n^M(y,z)|^2$ are shown in the figure.

Conclusion

We presented the FEM scheme and showed its efficiency by benchmark examples of using the KANTBP 5M program (an upgrade of KANTBP 4M [4]) implemented and executed in MAPLE. The new type of FEM discretization is implemented using IHPs *with an arbitrary multiplicity of IHPs nodes*, which preserves the continuity of derivatives of the sought solutions and Gauss quadratures. To calculate metastable states with complex eigenvalues, or to solve a bound state problem with Robin BC depending on the spectral parameter, the Newtonian iteration scheme is implemented and applied for calculation of beryllium dimer spectrum [5].

The work was partially supported by the RFBR and MECSS, project number 20-51-44001, the Ministry of Education and Science of the Russian Federation, grant number 075-10-2020-117, the Bogoliubov-Infeld program, the Hulubei-Meshcheryakov program, the RUDN University Strategic Academic Leadership Program, grant of Plenipotentiary of the Republic of Kazakhstan in JINR, the Foundation of Science and Technology of Mongolia, grant number SST_18/2018.

References

1. *Gusev A.A. et al.* Symbolic-numerical solution of boundary-value problems with self-adjjoint second-order differential equation using finite element method with interpolation hermite polynomials. Lecture Notes in Computer Sci. 2014. Vol. 8660. P. 138–154.
2. *Gusev A.A. et al.* Symbolic-numerical solution of boundary-value problems for Schrödinger equation using the finite element method: Scattering problem and resonance states. Lecture Notes in Computer Sci. 2015. Vol. 9301. P. 182–197.
3. *Chuluunbaatar G. et al.* KANTBP 4M: Program for solving the scattering problem for a system of ordinary second-order differential equations. EPJ Web of Conferences, 2020. Vol. 226. P. 02008.
4. *Gusev A.A. et al.* KANTBP 4M — program for solving boundary problems of the self-adjoint system of ordinary second order differential equations. Program library JINRLIB, https://wwwinfo.jinr.ru/programs/jinrlib/kantbp4m/indexe.html
5. *Derbov, V.L., et al.* Spectrum of beryllium dimer in ground $X^1\Sigma_g^+$ state. Journal of Quantitative Spectroscopy and Radiative Transfer. 2021. Vol. 262. P. 107529-1-10.

From Prony's Exponential Fitting to sub-Nyquist Signal Processing Using Computer Algebra Techniques

Annie Cuyt[1,3] and Wen-shin Lee[2]

[1] *University of Antwerp, Antwerp, Belgium*

[2] *University of Stirling, Scotland, UK*

[3] *Shenzhen University, Shenzhen, China*

e-mail: annie.cuyt@uantwerpen.be, wen-shin.lee@stir.ac.uk

Abstract. The classical Prony's exponential fitting method is closely related to sparse polynomial interpolation. Using techniques from computer algebra, a sub-Nyquist version of Prony's method has been developed, which offers new potentials in signal processing, including an application in antenna design.

Keywords: sparse polynomial, Prony's method, Ben-Or/Tiwari algorithm.

1. From computer algebra to signal processing

Sparse polynomial interpolation is at the core of computer algebra because polynomials are one of the fundamental objects in symbolic computation, as well as computational mathematics in general. In 1988, Ben-Or and Tiwari gave a sparse interpolation algorithm that is based on the Berlekamp/Massey algorithm from coding theory [1]. Interestingly, this exact algorithm is closely related to a classical exponential fitting method proposed by de Prony at the end of the 18th century [4].

Prony's method interpolates a univariate function $f(t)$ as a sum of exponential functions. Essentially, a substantial amount of effort in the field of signal processing is dedicated to the analysis of exponential functions whose exponents are complex. In order to unambiguously recover the imaginary parts of exponents, the target exponential function needs to be sampled at a rate greater than twice the bandwidth, the Nyquist sampling rate [9, 10].

Using techniques from computer algebra, we developed a sub-Nyquist version of Prony's method that can correctly recover the complex exponents from measurements sampled at a rate lower than the Nyquist rate [3]. This development offers new possibilities in signal processing, including the direction of arrival (DOA) estimation of incoming signals through a sparse array antenna system [8].

2. Prony's method and sparse polynomial interpolation

We briefly describe Prony's method [4]. Consider a univariate exponential function

$$f(t) = \sum_{j=1}^{n} \alpha_j \exp(\phi_j t), \qquad \alpha_j, \phi_j \in \mathbb{C}.$$

For a given $\Delta \in \mathbb{R}_{>0}$, Prony's method recovers α_j and $\exp(\phi_j \Delta)$ from $2n$ evenly spaced evaluations of $f(t)$:

$$f_0 = f(0), f_1 = f(\Delta), f_2 = f(2\Delta), \ldots, f_{2n-1} = f((2n-1)\Delta).$$

Let $\Omega_j = \exp(\phi_j \Delta)$ and consider the polynomial

$$\Lambda(z) = \prod_{j=1}^{n} (z - \Omega_j) = z^n + \beta_{n-1} z^{n-1} + \cdots \beta_1 z + \beta_0.$$

Since Ω_j are the zeros of $\Lambda(z)$, for any $k \geq 0$ we have

$$\begin{aligned} 0 &= \sum_{j=1}^n \alpha_j \Omega_j^k \cdot \underbrace{\Lambda(\Omega_j)}_{=0} = \sum_{j=1}^n \alpha_j \Omega_j^k \underbrace{\left(\Omega_j^n + \beta_{n-1}\Omega_j^{n-1} + \cdots + \beta_0\right)}_{\Lambda(\Omega_j)=0} \\ &= \sum_{j=1}^n \alpha_j \Omega_j^{n+k} + \beta_{n-1} \sum_{j=1}^n \alpha_j \Omega_j^{n-1+k} + \cdots + \beta_0 \sum_{j=1}^n \alpha_j \Omega_j^k \\ &= f_{k+n} + \beta_{n-1} f_{k+n-1} + \cdots + \beta_0 f_k. \end{aligned} \tag{1}$$

Based on (1), the coefficients β_j in $\Lambda(z)$ can be obtained by solving a Hankel system whose entries are the evaluations of $f(t)$

$$\begin{bmatrix} f_0 & \cdots & f_{n-1} \\ \vdots & \iddots & \vdots \\ f_{n-1} & \cdots & f_{2n-2} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_{n-1} \end{bmatrix} = - \begin{bmatrix} f_n \\ \vdots \\ f_{2n-1} \end{bmatrix}. \tag{2}$$

Then Ω_j are computed by rooting the polynomial $\Lambda(z) = z^n + \beta_{n-1} z^{n-1} + \cdots + \beta_0$.

On the other hand, Ω_j can also be directly computed as the generalised eigenvalues of two shifted Hankel systems [6]. That is, Ω_j are the n solutions to z in

$$\begin{bmatrix} f_1 & \cdots & f_n \\ \vdots & \iddots & \vdots \\ f_n & \cdots & f_{2n-1} \end{bmatrix} v = z \begin{bmatrix} f_0 & \cdots & f_{n-1} \\ \vdots & \iddots & \vdots \\ f_{n-1} & \cdots & f_{2n-2} \end{bmatrix} v. \tag{3}$$

Note that either (2) or (3) requires $2n$ evaluations of $f(t)$.

Once the values of Ω_j are known, α_j can be computed from solving the Vandermonde system

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \Omega_1 & \Omega_2 & \cdots & \Omega_n \\ \vdots & \vdots & \vdots & \vdots \\ \Omega_1^{n-1} & \Omega_2^{n-1} & \cdots & \Omega_n^{n-1} \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{n-1} \end{bmatrix}.$$

In Section 2, we will discuss when $\phi_j \in \mathbb{C}$ can be uniquely recovered from $\Omega_j = \exp(\phi_j \Delta)$.

In computer algebra, the Ben-Or/Tiwari sparse interpolation algorithm [1] recover the coefficients α_j and exponents $(d_{j_1}, \ldots, d_{j_n})$ in the given polynomial

$$p(x_1, \ldots, x_n) = \sum_{j=1}^n \alpha_j x_1^{d_{j_1}} x_2^{d_{j_2}} \cdots x_n^{d_{j_n}}$$

from its evaluations at powers of $(\omega_1, \omega_2, \ldots, \omega_n)$:

$$p_k = p(\omega_1^k, \omega_2^k, \ldots, \omega_n^k) = \sum_{j=1}^n \alpha_j \omega_1^{k \cdot d_{j_1}} \cdots \omega_n^{k \cdot d_{j_n}} = \sum_{j=1}^n \alpha_j (\omega_1^{d_{j_1}} \cdots \omega_n^{d_{j_n}})^k.$$

Suppose $\overline{\Omega}_j = \omega_1^{d_{j_1}} \cdots \omega^{d_{j_n}}$. The Ben-Or/Tiwari algorithm can be viewed as applying Prony's method to recovers α_j and $\overline{\Omega}_j = \omega_1^{d_{j_1}} \cdots \omega_n^{d_{j_n}}$ from

$$p_k = p(\omega_1^k, \omega_2^k, \ldots, \omega_n^k) = \sum_{j=1}^n \alpha_j \overline{\Omega}_j^k.$$

For appropriately chosen $\omega_1, \ldots, \omega_n$, the multivariate exponents $(d_{j_1}, d_{j_2}, \ldots, d_{j_n})$ in polynomial $p(x_1, \ldots, x_n)$ can be uniquely recovered from $\overline{\Omega}_j$. For example,

- $\omega_1, \ldots, \omega_n$ are integers and $\gcd(\omega_k, \omega_\ell) = 1$ whenever $k \neq \ell$ [1], or
- $p_1, \ldots, p_n$ are integers, $\gcd(p_k, p_\ell) = 1$ whenever $k \neq \ell$ and $\omega_k = \exp(2\pi \mathtt{i}/p_k)$ [5].

Both of the above exploit the fact that the exponents $(d_{j_1}, \ldots, d_{j_n})$ in a polynomial are all integers.

So far we assume that the number n of terms is known. When n is not given in a polynomial $p(x_1, \ldots, x_n)$, through randomization n can be determined with high probability in exact arithmetic [7]. The situation of an exponential function in floating point arithmetic is much more complicated. We refer to [2] for further discussions and the solutions to some situations.

3. Sub-Nyquist signal processing

As explained in Section 1, Prony's method can recover α_j and Ω_j from the evaluations of the target exponential function at evenly spaced values $f(0), f(\Delta), f(2\Delta), \ldots, f((2n-1)\Delta)$. Now we investigate how to uniquely recover $\phi_j \in \mathbb{C}$ from $\Omega_j = \exp(\phi_j \Delta)$. Unlike the integer exponents in a polynomial, we cannot assume a minimum distance between ϕ_j in an exponential function.

We use $\Im(\phi_j)$ to represent the imaginary part of ϕ_j. As stated in the Shannon-Nyquist sampling theorem [9, 10], if

$$\Delta = \frac{2\pi}{M} \quad \text{and} \quad |\Im(\phi_j)| < \frac{M}{2} \quad \text{for} \quad j = 1, \ldots, n,$$

then each ϕ_j can be uniquely recovered from $\Omega_j = \exp(\phi_j \Delta)$.

Sampling an exponential function at a rate below the Nyquist rate causes aliasing, meaning different frequencies to become indistinguishable. To be more precise, for a positive integer $r \in \mathbb{Z}_{>0}$, from the value of $\exp(\phi_j r \Delta)$ there are r many solutions to the imaginary part of ϕ_j (within the restricted range).

In signal processing, a co-prime technique can be used to uniquely recover ϕ_j from two sub-sampled values $\exp(\phi_j r_1 \Delta) = \Omega_j^{r_1}$ and $\exp(\phi_j r_2 \Delta) = \Omega_j^{r_2}$ when $\gcd(r_1, r_2) = 1$ (for example, see [11]). However, if Prony's method is applied to the corresponding sub-sampled evaluations, then from the values

$$\begin{aligned} & \bar{\Omega}_1 = \Omega_1^{r_1}, \quad \bar{\Omega}_2 = \Omega_2^{r_1}, \quad \ldots, \quad \bar{\Omega}_n = \Omega_n^{r_1} \\ \text{and} \quad & \tilde{\Omega}_1 = \Omega_1^{r_2}, \quad \tilde{\Omega}_2 = \Omega_2^{r_2}, \quad \ldots, \quad \tilde{\Omega}_n = \Omega_n^{r_2} \end{aligned}$$

one still needs to correctly match $\bar{\Omega}_j$ with $\tilde{\Omega}_j$ for each $j = 1, \ldots, n$ [12].

Using computer algebra techniques, we offer a method that can directly compute the match [3]. Applying Prony's method to $f(0), f(r_1\Delta), f(2r_1\Delta), \ldots, f((2n-1)r_1\Delta)$, we obtain $\bar{\Omega}_1 = \Omega_1^{r_1}, \ldots, \bar{\Omega}_n = \Omega_n^{r_1}$ and $\alpha_1, \ldots, \alpha_n$. Once $\Omega_j^{r_1}$ are known, with additional n evaluations $f(kr_1\Delta + r_2\Delta)$ for $k = 0, 1, \ldots, n-1$, we solve $\overline{\alpha}_j$ in

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \Omega_1^{r_1} & \Omega_2^{r_1} & \cdots & \Omega_n^{r_1} \\ \vdots & \vdots & \vdots & \vdots \\ \Omega_1^{(n-1)r_1} & \Omega_2^{(n-1)r_1} & \cdots & \Omega_n^{(n-1)r_1} \end{bmatrix} \begin{bmatrix} \overline{\alpha}_1 = \alpha_1 \Omega_1^{r_2} \\ \overline{\alpha}_2 = \alpha_2 \Omega_2^{r_2} \\ \vdots \\ \overline{\alpha}_n = \alpha_n \Omega_n^{r_2} \end{bmatrix} = \begin{bmatrix} f(r_2\Delta) \\ f(r_1\Delta + r_2\Delta) \\ \vdots \\ f((n-1)r_1\Delta + r_2\Delta) \end{bmatrix}.$$

From $\overline{\alpha}_j$ we can compute $\Omega_j^{r_2}$ as $\overline{\alpha}_j/\alpha_j$ and directly match $\Omega_j^{r_1}$ to $\Omega_j^{r_2}$ for each $j = 1, \ldots, n$.

4. Sparse array antenna systems

Prony's method and its variants are viewed as high-resolution time-frequency analysis tools and have been used to estimate the directions of incoming signals from a uniform linear antenna array [6]. For array antenna systems, the sub-Nyquist version of Prony's method allows for larger spacing between individual antenna elements, leading to an improved angular resolution and reduced mutual coupling effects between the receivers [8].

References

1. *Ben-Or M. and Tiwari P.* A deterministic algorithm for sparse multivariate polynomial interpolation. STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing. P. 301–309. New York, NY, USA, 1988. ACM.
2. *Cuyt A., Tsai M.-n., Verhoye M., and Lee W.-s.* Faint and clustered components in exponential analysis. Appl. Math. Comput. 2018. Vol. 327. P. 93–103.
3. *Cuyt A. and Lee W.-s.* How to get high resolution results from sparse and coarsely sampled data. Appl. Comput. Harmon. Anal. 2020. Vol. 48. P. 1066–1087.
4. *de Prony R.* Essai expérimental et analytique sur les lois de la dilatabilité des fluides élastiques et sur celles de la force expansive de la vapeur de l'eau et de la vapeur de l'alkool, à différentes températures. J. Ec. Poly. 1795. Vol. 1(22). P. 24–76.
5. *Giesbrecht M., Labahn G., and Lee W.-s.* Symbolic-numeric sparse interpolation of multivariate polynomials. J. Symbolic Comput. 2009. Vol. 44(8). P. 943–959.
6. *Hua Y. and Sarkar T.K.* Matrix pencil method for estimating parameters of exponentially damped/undamped sinusoids in noise. IEEE Trans. Acoust., Speech, Signal Process. 1990. Vol. 38. P. 814–824.
7. *Kaltofen E. and Lee W.-s.* Early termination in sparse interpolation algorithms. J. Symbolic Comput. 2003. Vol. 36(3-4). P. 365–400.
8. *Knaepkens F., Cuyt A., Lee W.-s., and de Villiers D.I.L.* Regular sparse array direction of arrival estimation in one dimension. IEEE Trans. Antennas Propag. 2020. Vol. 68. P. 3997–4006.
9. *Nyquist H.* Certain topics in telegraph transmission theory. Trans. Am. Inst. Electr. Eng. 1928. Vol. 47(2). P. 617–644.
10. *Shannon C.E.* Communication in the presence of noise. Proc. IRE. 1949. Vol. 37. P. 10–21.
11. *Vaidyanathan P.P. and Pal P.* Sparse sensing with co-prime samplers and arrays. IEEE Transactions on Signal Processing. 2011. Vol. 59(2). P. 573–586.
12. *Weng Z. and Djurić P.M.* A search-free doa estimation algorithm for coprime arrays. Digital Signal Processing. 2014. Vol. 24. P. 27–33.

The Integrability Condition in the Normal Form Method[1]

V.F. Edneral[1,2]

[1] *Skobeltsyn Institute of Nuclear Physics of Lomonosov Moscow State University, Russia*

[2] *Peoples' Friendship University of Russia (RUDN University), Russia*

e-mail: edneral@theory.sinp.msu.ru, edneral-vf@rudn.ru

Abstract. The purpose of this report is to demonstrate the search for integrability conditions by the normal form method. We have chosen an equation of the Liénard-type as the object of the demonstration. We presented the equation as a dynamical system and parameterized it. We constructed polynomial equations in the system parameters which should be satisfied for the integrable cases.

Keywords: Liénard's equation, resonance normal form, integrability

Problem

We will check our method on the example of the Liénard-like equation

$$\ddot{x} = f(x)\dot{x} + g(x). \tag{1}$$

We suppose $f(x)$ is a quadratic polynomial and $g(x)$ is polynomial of fourth-order. Usually in the Liénard equation it supposed $f(x)$ is an odd function [1]. Opposite, we suppose $g(x)$ is an voluntary function and say about the Liénard-like equation. Equation (1) is equivalent to the dynamical system

$$\begin{aligned} \dot{x} &= y, \\ \dot{y} &= (a_0 + a_2 x^2)\, y + d_0(1 + d_1 x)(1 + d_2 x)(1 + d_3 x)(1 + d_4 x), \end{aligned} \tag{2}$$

here x and y are functions in time and parameters $a_0, a_2, d_0, d_1, d_2, d_3, d_4$ are real. If exclude trivial case $d_0 = 0$ it is possible to put $d_0 = 1$.

The problem is to construct the integral of motion of the system (2).

Method

The main idea of the discussed method is a search of conditions on the system parameters when the system is locally integrable near its stationary points. The local integrability means we have enough number (one for an autonomic plane system) of the local integrals near each stationary point. Local integrals can be different for each such point, but for the existence of the global integral, the local integrals should exist in all stationary points. This is a necessary condition. We have an algebraic condition for local integrability. It is the condition **A** [2]. We look for sets of parameters at which the condition **A** is satisfied at all stationary points simultaneously. Such sets of parameters are good candidates for the existence of global integrals. These integrals we look for by other methods.

The symmetric notation of the g polynomial in (2) allows to get the condition **A** for all stationary points $(x = -1/d_1, y = 0), \ldots, (x = -1/d_4, y = 0)$ from the normal form for the single point by a permutation of d-parameters. This procedure eliminates the solutions which correspond to a single point only.

[1] This paper has been supported by the RUDN University Strategic Academic Leadership Program.

Conditions of the Integrability

The condition **A** is some infinite sequence of polynomial equations in coefficients of the system [3]. Near each of the stationary points are its equations. The normal form has a nontrivial form in the resonance case only. The eigenvalues of the linear part of system (2) are (at $d_0 = 1$)

$$\frac{1}{2}\left(a_0 \mp \sqrt{a_0^2 + 4(d_1 + d_2 + d_3 + d_4)}\right),$$

so we can choose the parameter a_0 to have the resonance $1 : N$ by solving the equation

$$\begin{aligned}&\tfrac{1}{2}\left(a_0 - \sqrt{a_0^2 + 4(d_1 + d_2 + d_3 + d_4)}\right) = \\ &-\tfrac{N}{2}\left(a_0 + \sqrt{a_0^2 + 4(d_1 + d_2 + d_3 + d_4)}\right).\end{aligned}$$

We calculated the lowest contributions in conditions A for resonances 1:1, 1:2 and 1:3 and added the ones with permutations of d parameters. We got very complicated algebraic equations in parametric space. Now we are searching for rational solutions.

Results

For resonance 1:1 we received at least one rational solution of the truncated condition A. It is

$$a_2 = -a_0 d_1^2, d_0 = 1, d_2 = -d_1, d_3 = 0, d_4 = 0.$$

System (2) at these parameters has a form

$$\begin{aligned}&\dot{x} = y, \\ &\dot{y} = (1 + a_0 y)\left(1 - d_1^2 x^2\right)\end{aligned} \tag{3}$$

or

$$\ddot{x} = (1 + a_0\dot{x})\left(1 - d_1^2 x^2\right)$$

and the corresponded integral of motion is

$$I(x, y) = \frac{a_0 y - \log(a_0 y + 1)}{a_0^2} + \frac{1}{3} d_1^2 x^3 - x.$$

Equation (3) corresponds to the form of item 1.3.3 in book [4].

Our equations are satisfied for the form of item $1.3.3 - 2.1$ in the book [4]. In this sample, we have resonance (1:2) [4]

$$\ddot{x} = y(ax + 3b) - abx^2 - 2b^2 x + cx^3,$$

but in the non-resonance case $1.3.3 - 2.2$

$$\ddot{x} = y(3ax + b) - a^2 x^3 - abx^2 + cx$$

these equations are satisfied at resonance cases only.We checked cases $c = 2b^2$ (resonance $1 : 2$), $c = \frac{3b^2}{4}$ (resonance $1 : 3$), $c = \frac{4b^2}{9}$ (resonance $1 : 4$). We need to learn all polynomial examples from the wonderful book [4].

The case with resonance 1:3 has been explored in [5] by the Puiseux series method. There is the nontrivial first integral at rational numerical values of parameters. These parameter values satisfy the received here equations.

References

1. *Liénard A.* Etude des oscillations entretenues. Revue générale de l'électricité. 1928. Vol. 23. P. 901–912 and 946–954.

2. *Bruno A.D.* Analytical form of differential equations (I, II). Trudy Moskov. Mat. Obsc. 1971. Vol. 25. P. 119–262. 1972. Vol. 26. P. 199–239 (in Russian). = Trans. Moscow Math. Soc. 1971. Vol. 25. P. 131–288. 1972. Vol. 26. P. 199–239 (in English).
Bruno A.D. Local Methods in Nonlinear Differential Equations. Nauka, Moscow 1979 (in Russian) = Springer-Verlag, Berlin. 1989. P. 348.

3. *Edneral V.F.* About integrability of the degenerate system. Computer Algebra in Scientific Computing (CASC 2019), M. England et al. Lecture Notes in Computer Science. 2019. Vol. 11661. Springer International Publishing, Springer Nature, Switzerland AG. P. 140–151.

4. *Polyanin A.D., Zaitsev V.F.* Handbook of exact solutions for ordinary differential equations–2nd ed. Chapman & Hall/CRC Press. 2003. P. 803.

5. *Demina M.V.* Private communication.

Deciding Cryptographically Important Properties of Finite Quasigroups

A.V. Galatenko[1], A.E. Pankratiev[1], V.M. Staroverov[1]

[1]*Lomonosov Moscow State University, Russia*

e-mail: agalat@msu.ru, apankrat@intsys.msu.ru, staroverovvl@imscs.msu.ru

Abstract. Finite quasigroups are a promising structure for design of various cryptographic primitives. Due to security reasons it makes sense to select quasigroups with a number of properties, such as polynomial completeness or at least non-affinity and poor quasigroup structure. We describe algorithms that decide these properties, analyse algorithm complexity and outline the results of practical efficiency analysis.

Keywords: finite quasigroup, polynomial completeness, simplicity, affinity, subquasigroup

1. Introduction

Non-commutative and non-associative algebraic structures are attracting attention as promising building blocks for cryptographic primitives [1]. Finite quasigroups are an example of such structure. A survey on quasigroup-based cryptographic algorithms can be found e.g. in [2]. In order to provide cryptographic strength researchers impose additional requirements on quasigroups. Some of these requirements are hard to formalize (e.g. "fractile/non-fractile" output structure in [3]), some can be trivially checked based on well-known methods (e.g. degrees on polynomials in the multivariate representation). V. A. Artamonov pointed out the importance of polynomially complete quasigroups ([4]), since the problem of solvability of equations over polynomially complete algebras is NP-complete ([5]), thus a large class of algebraic attacks becomes infeasible. Another important property is poor quasigroup structure (otherwise quasigroup-based transformations can degrade).

We focus on deciding polynomial completeness and the presence of non-trivial subquasigroups in case of quasigroups specified by Cayley tables. Polynomial completeness of quasigroups is known to be equivalent to simplicity and non-affinity ([6]), so the first problem is reduced to two subproblems, namely deciding simplicity and deciding non-affinity.

2. Basic definitions

Definition 1. *A finite quasigroup is a pair (Q, f), where Q is a finite set and $f\colon Q\times Q \to Q$ is invertible in both variables.*

We consider only finite quasigroups, so for the sake of brevity hereinafter the word "finite" will be omitted.

Without loss of generality we assume that $Q = \{0, \ldots, k-1\}$ for some $k \in \mathbb{N}$ and f belongs to the set of function of k-valued logic endowed with standard operations of superposition and closure ($f \in P_k$). Let P_k^0 be the set of all constants, $[F]$ be the closure of a set $F \subseteq P_k$.

Definition 2. *A quasigroup (Q, f) is polynomially complete if $[\{f\} \cup P_k^0] = P_k$.*

In other words, polynomial completeness means that an arbitrary function of k-valued logic can be generated by the set of all constants and the operation f using a finite number of superpositions.

Let $\sim$ be an equivalence relation on Q, $g \in P_k$ be an n-ary function. The function g preserves the relation $\sim$, if for any $a_1, \ldots, a_n, b_1, \ldots, b_n$ from Q such that $a_i \sim b_i$, $i = 1, \ldots, n$, it holds that $g(a_1, \ldots, a_n) \sim g(b_1, \ldots, g_n)$. This definition can be naturally generalized to the case of an arbitrary set of functions.

Definition 3. *A quasigroup (Q, f) is simple, if the function f preserves no non-trivial equivalence relations on Q. In this case f is also referred to as simple.*

Definition 4. *A quasigroup (Q, f) is affine, if there exists an Abelian group $(Q, +)$, automorphisms α, β of this group and $c \in Q$ such that the identity $f(x, y) = \alpha(x) + \beta(y) + c$ holds for all $x, y \in Q$. In this case f is also referred to as affine.*

Definition 5. *Suppose that $Q' \subset Q$, $1 \leq |Q'| < |Q|$. If $f(Q') = Q'$, then the quasigroup (Q, f) is said to contain the proper subquasigroup Q' (formally the subquasigroup is the pair (Q', f'), where f' is the restriction of f to the set $Q' \times Q'$; however for the sake of brevity we will simplify the notation). If additionally it holds that $|Q'| \geq 2$, then the subquasigroup is non-trivial.*

In the framework of complexity analysis elementary operations are reading and writing data from/to a cell in memory (in particular, getting the result of the quasigroup operation represented by its Cayley table) and arithmetic operations in $\mathbb{Z}_k$.

3. Deciding simplicity

We use the following algorithm from [7]:

```
cycle through all pairs (0, i), i = 1, ..., k - 1
  build the transitive closure of the relation 0 ~ i under the action of f
  if the relation is non-trivial
    return ‘‘non-siple’’
  endif
endcycle
return ‘‘simple’’
```

Theorem 1. *The algorithm decides simplicity with complexity $O(|Q|^3)$.*

Note that in case of quasigroups only uniform relations, i.e. relations with equivalence classes of equal sizes, may be preserved, so in case of quasigroups of prime orders the answer is always “simple”.

The algorithm can be easily parallelized, since the iterations of the outermost cycle are independent. As it is shown in [7], MPI-based solutions scale well; OpenMP-based solutions scale worse due to the bottleneck formed by memory lookups; CUDA-based solutions for quasigroups of reasonable orders proved to be slow, as the Cayley table does not fit into “fast” memory.

The results of the OpenMP implementation tested using a 8-core workstation (i7–3770 CPU @3.40GHz) with 32 gigabytes of RAM are shown in the Table 1 ([7]).

As one can see, even in case of large quasigroups processing takes reasonable time.

4. Deciding non-affinity

Non-affinity is decided using the following algorithm from [7]:

Table 1. Program runtime (in seconds) for simplicity decision. N is the number of cores utilized; the remaining fields of the first row are the quasigroup orders.

N	1024	2048	4096	8192	16384	32768
1	1	14	143	1292	10853	86195
2	1	9	79	779	5431	48654
4	1	5	56	510	3776	37201
8	1	4	51	519	4403	34154

```
reorder the rows and columns of the Cayley table so that the first
                row and the first column specify the identical permutation
check whether the resulting table specifies an Abelian group
if not, return ‘‘non-affine’’
set A equal to the column of the original Cayley table starting with 0
set B equal to the row of the original Cayley table starting with 0
if A or B are not automorphisms with respect to +, return ’’non-affine’’
set C equal to the upper left element of the original Cayley table
check that the identity f(x,y) = A(x) + B(y) + C holds
if it does then return ‘‘affine’’
else return ‘‘non-affine’’
```

Theorem 2. *The algorithm decides non-affinity with complexity* $O\left(|Q|^3\right)$.

The hardest part here is associativity check; this part, as well as other complex steps (e.g. commutativity check or automorphism verification) can be easily parallelized. Performance of the implementations is similar to the case of simplicity decision. The results of the OpenMP implementation obtained in the similar way are shown in Table 2 ([7]):

Table 2. Program runtime (in seconds) for non-affinity decision. N is the number of cores utilized; the remaining fields of the first row are the quasigroup orders.

N	1024	2048	4096	8192	16384	32768
1	1	11	104	776	10809	86293
2	1	5	49	335	5245	42495
4	1	3	28	206	2715	21890
8	1	2	21	250	1995	16331

As one can see, in practice deciding non-affinity is faster than deciding simplicity.

5. Subquasigroup existence check

Let (Q, f) be a quasigroup, $Q' \subset Q$. Denote by $[Q']$ the set of all constants generated by f from Q'. Note that Q' is a subset of any proper subquasigroup if and only if $[Q'] \neq Q$. A straightforward method for deciding whether there exist proper subquasigroups of the size at least k is to exhaust all k-element subsets Q' of Q and to compute $[Q']$ for all of them. The complexity of this procedure for a fixed value of k obviously equals $O\left(|Q|^{k+2}\right)$. This bound can be essentially improved using the following idea: first compute "partial closures" of all Q' up to a certain size; if a subquasigroup is found, stop, else build a system of representatives that generate minimum elements in the set of "partial closures" (due to monotonicity of

the operation $[Q']$ reduction to the set of representatives is correct). Finally compute "full closures" of all representatives. Appropriate parameter selection allows one to essentially reduce temporal complexity (at the cost of slightly increased spatial complexity).

Theorem 3. *The algorithm proposed decides whether there exist non-trivial subquasigroups with temporal complexity* $O\left(|Q|^3 \log |Q|\right)$ *and spatial complexity* $O\left(|Q|^{5/2}\right)$. *Existence of arbitrary proper subquasigroups is decided with temporal complexity* $O\left(|Q|^{7/3} \cdot (\log |Q|)^{2/3}\right)$ *and spatial complexity* $O\left(|Q|^2\right)$.

Parallelization of the algorithm is not as straightforward as in the previous problems; however our OpenMP implementation showed fairly decent scalability. The case of arbitrary proper subquasigroups is processed almost instantly, so we only list results for the case of non-trivial subquasigroups. Testing environment was the same as in the previous experiments. Program execution times are shown in Table 3. Missing values indicate that program execution was terminated because computation time exceeded the threshold.

Table 3. Program runtime (in seconds) for non-trivial subquasigroup decision. N is the number of cores utilized; the remaining fields of the first row are the quasigroup orders.

N	1024	2048	4096	8192	16384
1	2	27	273	2254	
2	1	5	49	454	
4	1	4	32	282	
8	1	5	27	228	2401

As one can see, execution time is acceptable for quasigroups that fit into memory.

References

1. *Markov V.T., Mikhalev A.V., Nechaev A.A.* Nonassociative Algebraic Structures in Cryptography and Coding. Journal of Mathematical Sciences. 2020. Vol. 245, No. 2. P. 178–196.
2. *Shcherbacov V.A.* Quasigroups in cryptology. Computer Science Journal of Moldova. 2009. Vol. 17, No. 2(50). P. 193–228.
3. *Dimitrova V., Markovski S.* Classification of quasigroups by image patterns. Proc. of the Fifth International Conference for Informatics and Information Technology, Bitola, 2007. P. 152–160.
4. *Artamonov V.A., Chakrabarti S., Gangopadhyay S., Pal S.K.* On Latin squares of polynomially complete quasigroups and quasigroups generated by shifts. Quasigroups and Related Systems. 2013. Vol. 21. P. 117–130.
5. *Horváth G., Nehaniv C.L., Szabó Cs.* An assertion concerning functionally complete algebras and NP-completeness. Theoretical Computer Science. 2008. Vol. 407, No. 1. P. 591–595.
6. *Hagemann J., Herrmann C.* Arithmetical locally equational classes and representation of partial functions. Universal Algebra, Esztergom (Hungary). 1982. Vol. 29. P. 345–360.
7. *Galatenko A.V., Pankratiev A.E., Staroverov V.M.* Efficient verification of polynomial completeness of quasigroups. Lobachevskii Journal of Mathematics. 2020. Vol. 41, No. 8. P. 1444–1453.

Symbolic Implementation of Multivector Algebra in Julia Language

M.N. Gevorkyan[1], D.S. Kulyabov[1,2], A.V. Korolkova[1], A.V. Demidova[1], T.R. Velieva[1,3]

[1]*Peoples' Friendship University of Russia (RUDN University), Russian Federation*

[2]*Joint Institute for Nuclear Research, Russian Federation*

[3]*Plekhanov Russian University of Economics, Russian Federation*

e-mail: {gevorkyan-mn, kulyabov-ds, korolkova-av, demidova-av, velieva-tr}@rudn.ru

Abstract. In this work, we will briefly present the main facts from the theory of polyvectors and multivectors, generalizing the known mathematical constructions. In the report, we will consider the solution of a number of geometric examples using the Grassmann.jl library. This is implementation of the multivector algebra in the Julia language. It mixes computer algebra calculations with numerical calculations for better performance.

Keywords: Clifford algebra, Grassman algebra, multivectors, Julia Language, Reduce computer algebra system

Introduction

Geometric algebra originates from the works of G. G. Grassman, W. K. Clifford, and W. R. Hamilton. It studies the space of multivectors and the geometric product operation of vectors, which is defined through scalar and external multiplication of vectors. The geometric product defines the structure of the Clifford algebra [1], generalizing the Grassmann algebra [2,3] and the Hamilton quaternion algebra [4].

Multivector algebra generalizes many geometric and algebraic objects. In particular, for a three-dimensional space, the vector and scalar triple products become a special case of the external one. The algebra of multivectors of a special kind is isomorphic to the algebra of complex numbers (in the case of a two-dimensional space) and the algebra of quaternions (in the case of a three-dimensional space).

The framework of geometric algebra is not so widely known. Although its foundations were laid out in the works of Grassman and Clifford, but they did not become famous. Only at the beginning of the current century, works devoted to this area began to appear [3,5,6]. It is worth to note the applied focus of these works: on computer geometry [7] and geometric methods of mathematical physics.

1. Polyvector algebra

It is necessary to distinguish multivectors and polyvectors. Polyvectors are contravariant anti-symmetric tensors of rank $(0, p)$. More often, the term p-vector is used. It is used to denote p-vectors in the same way as ordinary vectors. The rank in some cases is indicated at the bottom of the letter, for example: $\underset{p}{\mathbf{u}}$ [3]. For $p = 2$, the term bivector is used, for $p = 3$ — trivector, and so on.

With respect to the operation of addition and multiplication by a scalar, the set of p-vectors form linear space, denoted as $\Lambda^p(L)$. The space L is a linear space of contravariant vectors with the basis $\langle \mathbf{e}_1, \dots, \mathbf{e}_n \rangle$. We will assume that L is defined over the field of real

numbers $\mathbb{R}$. The linear spaces of polyvectors of rank 0 are identified with the scalar field $\Lambda^0(L) = \mathbb{R}$, and the spaces of items of rank 1 with the space L.

The binary operation $\wedge$ is called *outer product* or *wedge product* if for any p-vectors $\mathbf{u} \in \Lambda^p(L)$, $\mathbf{v} \in \Lambda^q(L)$ a number of properties are satisfied, including bilinearity, associativity, and anticommutativity for odd ranks.

- $\operatorname{rank}(\underset{p}{\mathbf{u}} \wedge \underset{q}{\mathbf{v}}) = p + q$.
- $\mathbf{u} \wedge (\mathbf{v} \wedge \mathbf{w}) = (\mathbf{u} \wedge \mathbf{v}) \wedge \mathbf{w}$ associativity.
- $1 \wedge \mathbf{u} = \mathbf{u} \wedge 1 = \mathbf{u}$, where 1 scalar unit (identity).
- $\alpha \wedge \beta = \alpha \cdot \beta$ for scalars $\wedge$ equivalent to simple multiplication.
- $\underset{p}{\mathbf{u}} \wedge \underset{q}{\mathbf{v}} = (-1)^{pq} \underset{q}{\mathbf{v}} \wedge \underset{p}{\mathbf{u}}$ anticommutativity for odd ranks.
- $(\mathbf{u} + \mathbf{v}) \wedge \mathbf{w} = \mathbf{u} \wedge \mathbf{w} + \mathbf{v} \wedge \mathbf{w}$ distributivity (right).
- $\mathbf{w} \wedge (\mathbf{u} + \mathbf{v}) = \mathbf{w} \wedge \mathbf{u} + \mathbf{w} \wedge \mathbf{v}$ distributivity (left).
- $\mathbf{u} \wedge (\alpha \mathbf{v}) = (\alpha \mathbf{u}) \wedge \mathbf{v} = \alpha(\mathbf{u} \wedge \mathbf{v})$ this property, coupled with distributivity, gives the bilinearity of the operation $\wedge$.

From the bilinearity and associativity of the $\wedge$ operation, it follows that $\Lambda(L)$ is endowed with the structure of an algebra called the *outer algebra* of the L space or the *Grassmann algebra*.

The number of components of a p-vector depends on the dimension n of the space L and is equal to C_n^p. For example, a bivector in three-dimensional space has three significant components, which makes it possible to identify it with a certain vector, which in analytical geometry is called a pseudovector (for example, the unit normal vector). A trivector, in three-dimensional space, has one significant component and is called a pseudoscalar (for example, the torsion of a spatial curve).

In general, the rank of any p - vector embedded in the space L coincides with the rank of the $n - p$-vector. This allows you to define a one-to-one mapping between the space $\Lambda^p(L)$ and $\Lambda^{n-p}(L)$, called the *complement* mapping. For a three-dimensional space L, the vector product of two vectors can be reduced to the operation of the complement of the outer product of these vectors. Similarly, the mixed product is reduced to the operation of the complement of the outer product of three vectors.

In the general case, the n-vector in the n-dimensional space plays the role of a unit hyper-volume (area, volume in the case of $n = 2$ and 3).

2. Multivector algebra

Polyvectors already allow us to generalize a number of operations of vector algebra, which without this generalization are limited only to the special case of three-dimensional and two-dimensional Euclidean spaces. However, an even more general multivector algebra can be introduced.

Multivector is an object from $\Lambda(L) = \bigoplus_p \Lambda^p(L)$ consisting of elements of different dimensions: scalars, vectors, bivectors, trivectors, etc.

$$\mathbf{M} = \alpha + \mathbf{u} + \underset{2}{\mathbf{V}} + \underset{3}{\mathbf{W}} + \dots$$

In this expression, the summation sign is used in the same sense as the sign of the sum in complex numbers and quaternions.

Geometric product of two 1-vectors $\mathbf{u}$ and $\mathbf{v}$ is defined as:

$$\mathbf{uv} = (\mathbf{u}, \mathbf{v}) + \mathbf{u} \wedge \mathbf{v}.$$

This operation makes it possible to determine the divisions of a vector by a vector. The inverse of an arbitrary vector $\mathbf{u}$ is defined as $\mathbf{u}/\|\mathbf{u}\|^2$, and the division operation is defined as

$$\mathbf{u}/\mathbf{v} = \mathbf{u}\mathbf{v}^{-1} = \mathbf{u}\mathbf{v}/\|\mathbf{v}\|^2.$$

Here are some examples of generalizations of known mathematical objects.

For a two-dimensional Euclidean space L with the basis $\langle \mathbf{e}_1, \mathbf{e}_2 \rangle$, the set of multivectors of the form $a + b\mathbf{i}$ is isomorphic to the set of complex numbers $a + \mathrm{i}b$, where a, b are real numbers, and $\mathbf{i} = \mathbf{e}_1\mathbf{e}_2$.

For a three-dimensional space L, the set of multivectors of the form $a + \mathbf{v}$, where $\mathbf{v}$ is a bivector, is isomorphic to the set of quaternions.

The multivector $\mathbf{R} = \mathbf{ba}$, where $\mathbf{a}$ and $\mathbf{b}$ are some vectors, is called *rotor*. Acting as a rotation operator on an arbitrary vector $\mathbf{v}$

$$\mathbf{v}' = \mathbf{R}\mathbf{v}\mathbf{R}^{-1}$$

we get a new vector $\mathbf{v}'$, which is the result of rotation of the vector $\mathbf{v}$ by the angle 2θ, where θ — the angle between $\mathbf{a}$ and $\mathbf{b}$. It is noteworthy that the formula $\mathbf{R}\mathbf{v}\mathbf{R}^{-1}$ remains valid for any dimension of the space L.

3. Grassmann.jl package

The Grassmann.jl [8] package for the Julia language implements the multivector data type, allows you to set the bases of Euclidean and pseudo-Euclidean spaces, and perform various manipulations on multivectors in a given basis. An interesting feature of this package is the ability to combine both numerical and symbolic calculations. Symbolic calculations in this module are implemented using the Reduce computer algebra system.

The package implements operations of external, internal, and geometric products, as well as multivectors in Euclidean, pseudo-Euclidean, and projective spaces.

Conclusion

As can be seen from the above, multivectors have great generalizing power. This is especially evident in the software implementation of the multivector algebra. In the future, we will demonstrate the use of multivectors in various geometric problems, in particular in the problems of rotation and reflection in three-dimensional space.

Acknowledgments

The theoretical part of the work was done under the RUDN University Strategic Academic Leadership Program (M.N. Gevorkyan, A.V. Demidova). The numerical study was performed with the support of the Russian Foundation for Basic Research (RFBR) according to the research project No 19-01-00645 (D.S. Kulyabov, A.V. Korolkova). The symbolical study was performed with the support of the Russian Science Foundation grant No. 19-71-30008 (T.R. Velieva).

References

1. *Clifford W. K.* Applications of Grassmann's Extensive Algebra. American Journal of Mathematics. 1878. Vol. 1, No. 4. P. 350–358. ISSN 00029327,10806377.
2. *Grassmann H. G.* Die Mechanik nach den Principien der Ausdehnungslehre. Mathematische Annalen. 1877. Vol. 12. P. 222–240. DOI: 10.1007/BF01442659.
3. *Browne J.* Grassmann Algebra : Exploring extended vector algebra with Mathematica. 2009. Incomplete draft Version 0.50.
4. *Hamilton William Rowan S.* Elements of quaternions. — London : Longmans, Green, & co, 1866.
5. *Chisolm E.* Geometric Algebra. 2012.
6. *Hitzer E.* Introduction to Clifford's Geometric Algebra. SICE Journal of Control. 2011. Vol. 4, No. 1.
7. *Lengyel E.* Foundations of Game Engine Development. V. 1. Mathematics. Lincoln, California : Terathon Software LLC, 2016. 195 P. ISBN 9780985811747. URL: http://foundationsofgameenginedev.com.
8. Leibniz-Grassmann-Clifford differential geometric algebra / multivector simplicial complex. 2021. URL: https://github.com/chakravala/Grassmann.jl.

Application of Symbolic Computations for Investigation of the Equilibria of the System of Connected Bodies Moving on a Circular Orbit

S.A. Gutnik[1,2], V.A. Sarychev[3]

[1]*Moscow State Institute of International Relations (MGIMO University), Russia*

[2]*Moscow Institute of Physics and Technology, Russia*

[3]*Keldysh Institute of Applied Mathematics RAS, Russia*

e-mail: s.gutnik@inno.mgimo.ru, vas31@rambler.ru

Abstract. Computer algebra methods were used to solve a system of 12 algebraic equations that determines the equilibrium orientations for a system of two bodies, connected by a spherical hinge, that moves on a circular orbit around the Earth. To determine the equilibria the algebraic system was decomposed using algorithms for Gröbner basis construction. Evolution of the conditions for equilibria existence in the dependence of the parameter of the problem was investigated. The effectiveness of the algorithms for Gröbner basis construction was analyzed depending on the number of parameters of the problem.

Keywords: satellite–stabilizer, gravitational torque, circular orbit, Lagrange equations, algebraic system, equilibrium orientation, Computer algebra, Gröbner basis

1. Introduction

We consider the dynamics of a system of two bodies (satellite and stabilizer) connected by a spherical hinge. The system moves in a central Newtonian force field along a circular orbit. The problem is of practical importance for designing passive gravitational orientation systems of satellites, that can stay on the orbit for a long time without energy consumption. The dynamics of various composite schemes for satellitestabilizer gravitational orientation systems was presented in detail in [1]. The planar equilibrium orientations of two bodies system were found in papers [2] and [3] in the special cases, when the spherical hinge is located at the intersection of the satellite and stabilizer principal central axis of inertia and when the hinge is located on the intersection line of the principal central planes of inertia of the satellite and stabilizer. In paper [4], some classes of spatial equilibrium orientations of the satellite–stabilizer system in the orbital reference frame were analyzed, using the combination of computer algebra and linear algebra methods. In [5] the main attention was paid to the study of the conditions of existence of the equilibrium orientations of the system of two bodies refers to special cases when one of the principal axes of inertia of each of the two bodies coincides with either the normal of the orbital plane, the radius vector or the tangent to the orbit.

At the previous works we tried to separate the initial algebraic system of 12 equations into set of more simple algebraic subsystems using special conditions for the system parameters. At the present paper we solve the algebraic system by reducing the number of system parameters. We study the spatial equilibrium orientations of the satellite-stabilizer system in the orbital coordinate frame for the certain combinations of values of inertial and geometrical characteristic of the connected bodies when equations of stationary motions of the system depends on one parameter, using Gröbner basis construction method [6] and CAS

Maple [7], and Mathematica [8]. The spatial equilibrium orientations were determined by analyzing the roots of algebraic equations from the constructed Gröbner basis. Some results of this work were presented at the CASC 2020 conference.

2. Equations of motion

Let us consider the system of two bodies connected by a spherical hinge that moves along a circular orbit [4]. To write the equations of motion of two bodies, we introduce the following right-handed Cartesian coordinate systems: $OXYZ$ is the orbital coordinate system, the OZ axis is directed along the radius vector connecting the Earth center of mass C and the center of mass O of the two–body system, the OX axis is directed along the linear velocity vector of the center of mass O, and the OY axis coincides with the normal to the orbital plane. The axes of coordinate systems $O_1x_1y_1z_1$ and $O_2x_2y_2z_2$, are directed along the principal central axes of inertia of the first and the second body, respectively. The orientation of the coordinate system $Ox_iy_iz_i$, with respect to the orbital coordinate system is determined using the pitch (α_i), yaw (β_i), and roll (γ_i) angles, and the direction cosines in the transformation matrix between the orbital coordinate system $OXYZ$ and $Ox_iy_iz_i$ can be expressed in terms of aircraft angles [1]. Suppose that (a_i, b_i, c_i) are the coordinates of the spherical hinge P in the body coordinate system $Ox_iy_iz_i$, A_i, B_i, C_i are principal central moments of inertia; $M = M_1M_2/(M_1 + M_2)$; M_i is the mass of the ith body; p_i, q_i, and r_i are the projections of the absolute angular velocity of the ith body onto the axes Ox_i, Oy_i and Oz_i; and ω_0 is the angular velocity for the center of mass of the two-body system moving along a circular orbit. Then, using expressions for kinetic energy and force function, which determines the effect of the Earth gravitational field on the system of two bodies connected by a hinge [1], the equations of motion for this system can be written as Lagrange equations of the second kind by symbolic differentiation in the Maple system [7] in the case when $b_1 = b_2 = c_1 = c_2 = 0$ and the coordinates of the spherical hinge P in the body coordinate systems are $(a_i, 0, 0)$:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{x}_i} - \frac{\partial T}{\partial x_i} - \frac{\partial U}{\partial x_i} = 0, \quad i = \overline{1,6}, \tag{1}$$

where $x_1 = \alpha_1$, $x_2 = \alpha_2$, $x_3 = \beta_1$, $x_4 = \beta_2$, $x_5 = \gamma_1$, $x_6 = \gamma_2$ and

$$\begin{gathered} T = 1/2\big(A_1p_1^2 + (B_1 + Ma_1^2)q_1^2 + (C_1 + Ma_1^2)r_1^2\big) \\ +1/2\big(A_2p_2^2 + (B_2 + Ma_2^2)q_2^2 + (C_2 + Ma_2^2)r_2^2\big) - Ma_1a_2\big((r_1a_{12} - q_1a_{13})(r_2b_{12} - q_2b_{13}) \\ +(r_1a_{22} - q_1a_{23})(r_2b_{22} - q_2b_{23}) + (r_1a_{32} - q_1a_{33})(r_2b_{32} - q_2b_{33})\big) \end{gathered} \tag{2}$$

is the kinetic energy of the system,

$$\begin{gathered} U = 3/2\omega_0^2\big((C_1 - A_1 + Ma_1^2)a_{31}^2 + (C_1 - B_1)a_{32}^2\big) + 3/2\omega_0^2\big((C_2 - A_2 + Ma_2^2)b_{31}^2 \\ +(C_2 - B_2))b_{32}^2\big) - Ma_1a_2\omega_0^2(a_{11}b_{11} + a_{21}b_{21} + a_{31}b_{31}) \end{gathered} \tag{3}$$

is the force function, which determines the effect of the Earth gravitational field on the system of two connected by a hinge bodies. Differential equations (1) have the form indicated in [4].

The kinematic Euler equations have the form

$$\begin{gathered} p_1 = (\dot{\alpha_1} + 1)a_{21} + \dot{\gamma_1}, \quad p_2 = (\dot{\alpha_2} + 1)b_{21} + \dot{\gamma_2}, \\ q_1 = (\dot{\alpha_1} + 1)a_{22} + \dot{\beta_1}\sin\gamma_1, \quad q_2 = (\dot{\alpha_2} + 1)b_{22} + \dot{\beta_2}\sin\gamma_2, \\ r_1 = (\dot{\alpha_1} + 1)a_{23} + \dot{\beta_1}\cos\gamma_1, \quad r_2 = (\dot{\alpha_2} + 1)b_{23} + \dot{\beta_2}\cos\gamma_2. \end{gathered} \tag{4}$$

3. Investigation of Equilibrium Orientations

Assuming the initial conditions $x_i = \text{const}$, also $A_i \neq B_i \neq C_i$, we obtain from (1) and (4) the stationary equations

$$\begin{gathered} a_{22}a_{23} - 3a_{32}a_{33} = 0, \\ (a_{23}a_{21} - 3a_{33}a_{31}) + m_1(a_{23}b_{21} - 3a_{33}b_{31}) = 0, \\ (a_{22}a_{21} - 3a_{32}a_{31}) - n_1(a_{22}b_{21} - 3a_{32}b_{31}) = 0, \\ b_{22}b_{23} - 3b_{32}b_{33} = 0, \\ (b_{23}b_{21} - 3b_{33}b_{31}) + m_2(b_{23}a_{21} - 3b_{33}a_{31}) = 0, \\ (b_{22}b_{21} - 3b_{32}b_{31}) - n_2(b_{22}a_{21} - 3b_{32}a_{31}) = 0, \end{gathered} \tag{5}$$

which allow us to determine the equilibrium orientations of the system of two bodies connected by a spherical hinge in the orbital coordinate system. In (5): $m_1 = Ma_1a_2/\big((A_1-C_1)-Ma_1^2\big)$; $m_2 = Ma_1a_2/\big((A_2-C_2)-Ma_2^2\big)$; $n_1 = Ma_1a_2/\big((B_1-A_1)+Ma_1^2\big)$; $n_2 = Ma_1a_2/\big((B_2-A_2)+Ma_2^2\big)$.

We consider system (5) as the system of six algebraic equations for 12 direction cosines unknowns. To solve algebraic equations (5) we add to this system six orthogonality conditions for the direction cosines

$$\begin{gathered} a_{21}^2 + a_{22}^2 + a_{23}^2 - 1 = 0, \quad a_{31}^2 + a_{32}^2 + a_{33}^2 - 1 = 0, \\ a_{21}a_{31} + a_{22}a_{32} + a_{23}a_{33} = 0, \quad b_{21}^2 + b_{22}^2 + b_{23}^2 - 1 = 0, \\ b_{31}^2 + b_{32}^2 + b_{33}^2 - 1 = 0, \quad b_{21}a_{31} + b_{22}b_{32} + b_{23}b_{33} = 0, \end{gathered} \tag{6}$$

and obtain closed algebraic system of 12 equations for 12 unknowns.

For system (5), (6) the following problem is formulated: for given four parameters, determine all twelve direction cosines. The other six direction cosines (a_{1i} and b_{1i}) can be obtained from the orthogonality conditions.

In [2, 3], planar oscillations of the two-body system were analyzed, all equilibrium orientations were determined, and sufficient conditions for the stability of the equilibrium orientations were obtained, using the energy integral as a Lyapunov function. In [4], system (5), (6) was decomposed into homogeneous subsystems, using linear algebra methods and algorithms for constructing Gröbner basis. Some classes of spatial equilibrium solutions were obtained from the algebraic equations included in the Gröbner basis.

In [5] and system (5), (6) was solved in the special cases, when one of the principal axes of inertia of each of the two bodies coincides with either the normal to the orbital plane, the radius vector or the tangent to the orbit. Algebraic system (5), (6) was divided on the set of nine subsystems, then some distinct solutions of these subsystems was founded.

Solving the system of 12 algebraic equations (5) and (6) depending on four parameters by applying methods for constructing Gröbner bases is a very complicated algorithmic problem. Experiments on the construction of a Gröbner basis for the system of polynomials (5) and (6) by applying the `Groebner[Basis]` package implemented in Maple [7] and `GroebnerBasis` package implemented in CAS Wolfram Mathematica [8] were performed on a personal computer with 8 GB of RAM and 2.9 GHz Intel Core i7 CPU running 64-bit MS Windows 10. The results of the experiments are presented in the Table 1. In the general case, we failed to construct a Gröbner basis for this system.

We carry out a detailed solution of the system of algebraic equations (5) and (6) for the case when $m_1 = m_2 = n_1 = n_2 = m$. To solve this algebraic system, the `Groebner[Basis]` package for Maple 18 and `GroebnerBasis` package in Wolfram Mathematica 11 were used.

The number of polynomials in the constructed Gröbner basis in Maple was 160, the size of which occupies more than 1 million 120 character lines. Then we repeated this result of the construction the Gröbner basis using the Wolfram Mathematica 11 `GroebnerBasis` package. The first polynomial in the Gröbner basis that depends only on one variable $x = a_{33}$ after factorization has the form

$$P(x) = x(x^2 - 1)P_1(x)P_2(x)P_3(x)P_4(x)P_5(x)P_6(x)P_7(x)P_8(x) = 0, \tag{7}$$

where P_i are the quadratic or biquadratic polynomials. It is necessary to consider two cases $a_{33} = 0$, $a_{33}^2 = 1$, and eight cases $P_i(a_{33}) = 0$ $(i = 1, 2, 3, 4, 5, 6, 7, 8)$ to investigate the solutions of system (5), (6). In our report the investigations of these cases are presented. Equation (7) and system (5), (6) make it possible to determine all the spatial equilibrium configurations of the satellite–stabilizer, due to the action of the gravity torque for the given values of system parameter m.

We have estimated the size of a problem that can be solved by the application of the Gröbner basis construction method. The obtained results can be used at the stage of preliminary design of gravitational systems to control the orientation of the artificial Earth satellites.

Table 1. Time required to compute Gröbner basis for particular cases of the system parameters in Maple and Mathematica

Ex	Case of the problem	*Maple*	Type of the Maple algorithm	*Mathematica*
1	$m_1 = m; n_1 = m_2 = n_2 = 1$	12324 s	Walk/ *lexdeg* option	6520 s
2	$m_1 = n_1 = m; m_2 = n_2 = 1$	5503 s	F4/ *tdeg* option	18445 s
3	$m_1 = n_1 = m_2 = n_2 = m$	24 h	FGLM/ *plex* option, HPC	31 h

References

1. *Sarychev V.A.* Problems of orientation of satellites. Itogi Nauki i Tekhniki, Ser. "Space Research", Vol. 11. Moscow. VINITI, (1978). (In Russian)
2. *Sarychev V.A.* Relative equilibrium orientations of two bodies connected by a spherical hinge on a circular orbit. Cosmic Research. 1967. Vol. 5. P. 360–364.
3. *Gutnik S.A., Sarychev V.A.* Symbolic investigation of the dynamics of a system of two connected bodies moving along a circular orbit. In: England M., et al. (eds.) CASC 2019. LNCS, 2019. Vol. 11661. P. 164–178.
4. *Gutnik, S.A., Sarychev, V.A.* Application of computer algebra methods to investigate the dynamics of the system of two connected bodies moving along a circular orbit. Programming and Computer Software. 2019. Vol. 45, No. 2. P. 51–57.
5. *Gutnik, S.A., Sarychev, V.A.* Application of computer algebra methods to investigation of stationary motions of a system of two connected bodies moving in a circular orbit. Computational Mathematics and Mathematical Physics. 2020. Vol. 60, No. 1. P. 75–81.
6. *Buchberger B.* A theoretical basis for the reduction of polynomials to canonical forms. SIGSAM Bulletin. 1976. Vol 10, No. 3. P. 19–29.
7. Maple online help.– `ULR: http://maplesoft.com/support/help/`
8. *Wolfram S.* The Mathematica Book, 5-th edn. Wolfram media, Inc. Champaign, 2003.

Revisiting Geometric Integrators in Mechanics

Aziz Hamdouni[2], Vladimir Salnikov[1,2]
[1] *CNRS – National Center for Scientific Research, France*
[2] *La Rochelle University, France*
e-mail: aziz.hamdouni@univ-lr.fr, vladimir.salnikov@univ-lr.fr

Abstract. We address the question of efficient construction of geometric integrators – numerical methods preserving some internal geometric structure of the system of equations. Such methods are of particular importance for modelling and simulation of mechanical systems, where these structures permit to control the conservation of physically relevant quantities. We focus our attention on the so called generalized geometry, for which we present an approach to design higher order Runge–Kutta style numerical methods.

Keywords: geometric integrators, constraint mechanics, Dirac structures, Runge–Kutta methods.

Motivation / Introduction

In this contribution we study the structure preserving numerical methods, also often called *geometric integrators* appearing naturally in the context of robust and reliable simulation of mechanical systems. The key idea is that the equations governing mechanical systems have some intrinsic description using the objects from modern differential and algebraic geometry, those objects serve as a "proxy" to mimic physical properties of the systems: symmetries, conservation laws, qualitative behaviour, etc... The strategy itself is not exactly new, it somehow dates back almost to the middle of the 20th century in the context of integrable systems. However very often the implementation of it amounts to some "do-it-yourself" constructions. What we discuss in this text is a part of a big project of bringing "order and method" to this strategy, namely we work on explicit descriptions of the classes of mechanical systems with the corresponding geometric structures, for which then we formulate clear algorithmic approaches to construction of appropriate numerical methods. A recent overview of the state of the art can be found in [1].

Symplectic integrators

As mentioned above, one of the folkloric examples of geometric integrators are symplectic numerical methods in the context of Hamiltonian systems. One considers the phase space of a mechanical system on which a symplectic form is naturally defined – locally this is a skew-symmetric non-degenerate (constant) bilinear form[1] ω – a multidimensional generalization of the oriented area. Given a smooth function H, this ω permits to define a Hamiltonian vector field X_H governing the dynamics of the system. It is easy to show that ω is invariant by the flow of X_H, but a more interesting property is sort of converse: a vector field preserving ω will respect the level sets of H.

Phrased this way the symplectic property naturally gives the idea of a numerical method: if a discretized flow of the system of differential equations (better) preserves the symplectic

[1] In this text we will only give qualitative descriptions of the necessary geometric objects, skipping technical details, they may sound vague, but are globally correct. For more precise definitions and details a motivated reader may consult [2].

form, it conserves the energy of the system (better). However, to the best of our knowledge, the symplectic integrators were not constructed this way, they were merely discovered by chance. In some simulation of planetary systems it has been observed that neither explicit nor implicit methods produced satisfactory results in terms of stability, while a semi-implicit method did. And only after, for example in [3], the result was interpreted as above.

Let us, for pedagogical reasons, deduce the form of the symplectic Euler method. Consider a Hamiltonian system

$$\dot{q} = \frac{\partial H}{\partial p}, \quad \dot{p} = -\frac{\partial H}{\partial q},$$

defined by the Hamiltonian H and the symplectic form $\omega = dp \wedge dq$. Here q are the coordinates of the system, and p are its momenta; they are both multidimensional (vector) variables, but we omit the indeces not to overload the presentation. To solve this system consider a family of first order methods

$$q^{n+1} = q^n + h\frac{\partial H}{\partial p}(q, p), \tag{1}$$

$$p^{n+1} = p^n - h\frac{\partial H}{\partial q}(q, p), \tag{2}$$

where h is the timestep, and (q, p) in the right hand sides is a point to be determined. More precisely, let

$$q = aq^n + bq^{n+1} \text{ and } p = cp^n + dp^{n+1}$$

with unknown coefficients a, b, c, d, in principal allowed to be all different. We want the symplectic form to be conserved, thus compute $dp^{n+1} \wedge dq^{n+1} - dp^n \wedge dq^n$ and determine the conditions for it to vanish up to the maximal possible power of h, for arbitrary choice of H. Plugging in the (implicit) expressions (1) and (2), and using the Taylor expansion for the right hand sides, one obtains for the linear term the condition

$$a + b - c - d = 0,$$

which is trivially satisfied due to the consistency of the method: $a + b = 1$, $c + d = 1$. But already the quadratic term adds to this a non-trivial condition

$$a - d = 0,$$

which precisely means that the method should be neither purely explicit nor implicit. It is satisfied by the standard symplectic Euler method, where $a = d = 1$, $b = c = 0$. The computation may go further and potentially produces other conditions. The same strategy can eventually be applied for higher order methods that replace (1) and (2), and as mentioned, is relevant for other geometric structures – we describe them in the next section.

Dirac structure based methods

The Hamiltonian–symplectic formalism described above is appropriate for conservative isolated mechanical systems. Since its establishment several other directions have been explored, here in particular we discuss systems with constraints. It has been observed ([4]) that the relevant geometry for those is related to Dirac structures, not going into technical details, let us just give an idea of those. For classical mechanics one can describe the system using coordinates and either velocities or momenta – that gives Lagrangian or Hamiltonian

picture respectively. Dirac structures, roughly speaking use both of them simultaneously, so the considered space is enlarged; but they also take into account that velocities and momenta are not independent, so the space is restricted back in a non-trivial way. Hence, if the system is subject to some constraints, those can be formulated in terms of velocities, or momenta, or both.

The main message of [4], reviewed in [5], is that considering the Dirac structure associated to the constraint distribution, one can apply the techniques of variational integrators to design a numerical method which preserves the constraints better than the usual one. In [5, 6] we have constructed an improvement of such a Dirac based method, and considered some applications of it. But as mentioned, this has been done rather by an "educated guess", than by a generalizable procedure. In this text, we present a more algorithmic approach, in the style of the above symplectic discussion.

The starting point of [4, 5] is the data of a Lagrangian $L = L(q, v)$, and constraints $\varphi(q, v) = 0$, again all the variables are of appropriate dimension, but the indeces are dropped. The dynamics will however be viewed in the (q, p)-space:

$$\dot{q} = v, \quad \dot{p} = \frac{\partial L}{\partial q} + \lambda\alpha, \tag{3}$$

where $\alpha = d\varphi$ – the generators of the vanishing ideal for the constraints, λ is the set of Lagrange multipliers. The constraints are rewritten as

$$\alpha(v) = 0, \tag{4}$$

and the relation of v and p is given by the Legendre transform

$$p = \frac{\partial L}{\partial v}. \tag{5}$$

These two (algebraic) conditions as well as the differential equations (3) are deduced directly from the Dirac structure.

We will now apply the Runge–Kutta type methods to solve (3). Recall that for an equation $\dot{y} = f(y)$ the method reads

$$y^{n+1} = y^n + h\sum_{i=1}^{s} b_i k_i, \quad k_i = f(y^n + h\sum_{j=1}^{s} a_{ij}k_j). \tag{6}$$

This general form of solution is applied to both equations in (3), obviously with different f. Note that the method is a priori allowed to be implicit, and the coefficients a_{ij} and b_i can and will eventually be different for the two equations.

The same procedure as for the symplectic form above is now applied to the equations (4) – (5): suppose that they are satisfied at the n-th step — compute the approximation of them for the $(n + 1)$-st one, using the Taylor expansion — force it to be satisfied up to the maximal possible power of h. As a result, for $s = 1$, i.e. for the simplest first order Runge–Kutta method, one reproduces up to h^2 the Dirac-1 integrator spelled-out explicitly in [6]. Moreover one sees that holonomic constraints, that is not depending explicitly on v, are better preserved.

Higher order methods ($s > 1$) will be presented in detail in the extended version of this paper ([7]), together with careful benchmark tests and examples of application to mechanical systems. But already here it is important to note, that in contrast to the original approach of [4] their derivation sketched here is rather straightforward. The only arising complication is because of lengthy formal computation, potentially treated by computer algebra tools.

Conclusions / Outlook

Let us mention several remarks in conclusion.

First, the main framework discussed above – systems with constraints – seem to be a very particular class of systems. This is true from the mechanical point of view, but not exactly from the geometric perspective: for instance, apparently the Dirac structures constructed from the constraint distribution behave similarly to the ones from symplectic foliations of Poisson manifolds, this makes us think about Poisson integrators.

Second, Dirac structures also appear naturally for dissipative and coupled systems – the preservation of it by the continuous or discrete flow corresponds to power balance. We discussed some open questions on that in [2], and we can now add one more near at hand direction to them – constructing appropriate higher order discretizations.

Third, and probably conceptually the most important, is the relation of the above discussion to so called graded geometry. In fact all the above mentioned geometric constructions have a uniform and rather convenient description in terms of differential graded manifolds. We have sketched their description in [1] and [2], and mentioned some open computer algebra problems. Note here, that the discretization in the "graded world" is also a totally unexplored field that we plan to address in the nearest future.

Acknowledgments. This work is supported by the CNRS 80 Prime project "GraNum".

References

1. *Salnikov V., Hamdouni A., Loziienko D.* Generalized and graded geometry for mechanics: a comprehensive introduction. Mathematics and Mechanics of Complex Systems. 2021. Vol. 9, Issue 1.
2. *Salnikov V., Hamdouni A.* Differential Geometry and Mechanics — a source of problems for computer algebra. Programming and Computer Software. 2020. Vol. 46, Issue 2.
3. *Verlet L.* Computer "Experiments" on Classical Fluids. Phys. Rev. 1967. Vol. 159(1). P. 98–103.
4. *Yoshimura H., Marsden J.E.* Dirac Structures in Lagrangian Mechanics. Part I: Implicit Lagrangian Systems. Journal of Geometry and Physics. 2006. Vol. 57 P. 133–156.
5. *Salnikov V., Hamdouni A.* From modelling of systems with constraints to generalized geometry and back to numerics. Z Angew Math Mech. 2019.
6. *Razafindralandy D., Salnikov V., Hamdouni A., Deeb A.* Some robust integrators for large time dynamics. AMSES. 2019. Vol. 6(5).
7. *Loziienko D., Salnikov V., Hamdouni A.* Dirac–Runge–Kutta numerical methods, final preparation, 2021.

Automatic Confirmation of Exhaustive Use of Information on a Given Equation

D.E. Khmelnov[1], A.A. Ryabenko[1], S.A. Abramov[1,2]

[1] *Dorodnicyn Computing Center, Federal Research Center "Computer Science and Control" of RAS, Russia*

[2] *Faculty of Computational Mathematics and Cybernetics, Moscow State University, Russia*

e-mail: dennis_khmelnov@mail.ru, anna.ryabenko@gmail.com, sergeyabramov@mail.ru

Abstract. Algorithms were previously proposed that allow one to find truncated Laurent solutions to linear differential equations with coefficients in the form of truncated formal power series. Below are suggested some automatic means of confirming the impossibility of obtaining a larger number of terms of such solutions without some additional information on a given equation. The confirmation has the form of a counterexample to the assumption about the possibility of obtaining some additional terms of the solution.

Keywords: linear differential equations, power series, formal Laurent series, numbers of obtained terms of solutions, computer algebra systems

1. Problem Statement

In [1–3], we considered linear ordinary differential equations with coefficients given as truncated power series. We discussed the question of what can be learned from equations given in this way about their Laurent solutions, i.e. solutions belonging to the field of formal Laurent series. We were interested in the maximum possible information about these solutions, that is invariant with respect to possible prolongations of the truncated series which are the coefficients of the given equation (a *prolongation* of a truncated series is a series, possibly also truncated, whose initial terms coincide with the known initial terms of the original truncated series; correspondingly, the prolongation of an equation with truncated-series coefficients is an equation, whose coefficients are prolongations of the coefficients of the original equation). Algorithms for constructing such invariant truncated Laurent solutions were presented in the mentioned papers. In other words, the presented algorithms provide exhaustive use of information on a given equation. Maple [4] was chosen as a tool of the implementation.

Now we are focusing on the question of automatic confirmation of such exhaustive use of information on a given equation, i.e. the confirmation that it is not possible to add any additional terms to the constructed truncated solutions that are invariant with respect to prolongations of the given equation. In order to confirm this, it is sufficient to demonstrate a counterexample with two different prolongations of the given equation which lead to the appearance of different additional terms in the solutions.

Below, preliminary versions of procedures for searching for counterexample prolongations are presented. The procedures are based on finding Laurent solutions with *literals*, i.e., symbols used to represent the unspecified coefficients of the series involved in the equations (see [3]). Those symbols are coefficients of the terms, the degrees of which are greater than the degree of the series truncation. Finding Laurent solutions using literals means expressing the subsequent (not invariant to all possible prolongations) terms of the series in the solution as formulas in literals, i.e. via unspecified coefficients. This allows one to clarify the influence of unspecified coefficients on the subsequent terms of the series in the solutions.

Differential equations in the sequel are written using the operator $\theta = x\frac{d}{dx}$.

2. Examples

The confirmation of exhaustive use of the information on a given equation in the truncated Laurent solution is implemented as the Maple procedure `ExhaustiveUseConfirmation`.

Example 1. Consider the following equation with the truncated-series coefficients and construct its Laurent solution using the `TruncatedSeries` package [1–3] :

```
> eq := (-1+x+x^2+O(x^3))*theta(y(x),x,2)+(-2+O(x^3))*theta(y(x),x,1)+
        (x+6*x^2+O(x^4))*y(x);
```

$$eq := \left(-1 + x + x^2 + O(x^3)\right)\theta(y(x), x, 2) + \left(-2 + O(x^3)\right)\theta(y(x), x, 1) + \left(x + 6x^2 + O(x^4)\right)y(x)$$

```
> sol := TruncatedSeries:-LaurentSolution(eq,y(x));
```

$$sol := \left[\frac{\_c_1}{x^2} - \frac{5\_c_1}{x} + \_c_2 + O(x), \_c_2 + \frac{x\_c_2}{3} + \frac{5x^2\_c_2}{6} + \frac{13x^3\_c_2}{30} + O(x^4)\right]$$

The invocation of the procedure `ExhaustiveUseConfirmation` confirms exhaustive use of the information on the given equation with presenting two different prolongations of the equation that lead to two different prolongations of the solution. The procedure prints out details on two different equation prolongations and their solutions. It is shown that the provided solutions are different prolongations of the solution of the given equation with presenting different additional terms in the solutions.

```
> ExhaustiveUseConfirmation(sol, eq, y(x));
```

`The equation prolongation #1`

$$\left(-1 + x + x^2 - x^3 + O(x^4)\right)\theta(y(x), x, 2) + \left(-2 - x^3 + O(x^4)\right)\theta(y(x), x, 1) + \left(x + 6x^2 - x^4 + O(x^5)\right)y(x)$$

Additional term(s) in the equation prolongation:

$$y(x)(-x^4 + O(x^5)) + \theta(y(x), x, 1)(-x^3 + O(x^4)) + \theta(y(x), x, 2)(-x^3 + O(x^4))$$

The equation solution:

$$\left[\frac{\_c_1}{x^2} - \frac{5\_c_1}{x} + \_c_2 + x\left(\frac{\_c_2}{3} - \frac{37\_c_1}{3}\right) + O(x^2), \_c_2 + \frac{x\_c_2}{3} + \frac{5x^2\_c_2}{6} + \frac{13x^3\_c_2}{30} + \frac{11x^4\_c_2}{24} + O(x^5)\right]$$

Additional term(s) in the equation solution:

$$\left[x\left(\frac{\_c_2}{3} - \frac{37\_c_1}{3}\right) + O(x^2), \frac{11x^4\_c_2}{24} + O(x^5)\right]$$

`The equation prolongation #2`

$$\left(-1 + x + x^2 + x^3 + O(x^4)\right)\theta(y(x), x, 2) + \left(-2 + x^3 + O(x^4)\right)\theta(y(x), x, 1) + \left(x + 6x^2 + x^4 + O(x^5)\right)y(x)$$

Additional term(s) in the equation prolongation:

$$y(x)(x^4+O(x^5))+\theta(y(x),x,1)(x^3+O(x^4))+\theta(y(x),x,2)(x^3+O(x^4))$$

The equation solution:

$$\left[\frac{\_c_1}{x^2}-\frac{5\_c_1}{x}+\_c_2+x\left(\frac{\_c_2}{3}-11\_c_1\right)+O(x^2),\ \_c_2+\frac{x\_c_2}{3}+\frac{5x^2\_c_2}{6}\right.$$
$$\left.+\frac{13x^3\_c_2}{30}+\frac{43x^4\_c_2}{72}+O(x^5)\right]$$

Additional term(s) in the equation solution:

$$\left[x\left(\frac{\_c_2}{3}-11\_c_1\right)+O(x^2),\ \frac{43x^4\_c_2}{72}+O(x^5)\right]$$

Example 2. Consider a prolongation of the given equation with other additional terms (we use an auxiliary procedure `ConstructProlongation`) and construct its Laurent solution:

```
> eq1 := ConstructProlongation(theta(y(x), x, 1)*x^3, eq, y(x))
```

$$\left(-1+x+x^2+O(x^3)\right)\theta(y(x),x,2)+\left(-2+x^3+O(x^4)\right)\theta(y(x),x,1)+\left(x+6x^2+O(x^4)\right)y(x)$$

```
> TruncatedSeries:-LaurentSolution(eq1,y(x));
```

$$\left[\frac{\_c_1}{x^2}-\frac{5\_c_1}{x}+\_c_2+O(x),\ \_c_2+\frac{x\_c_2}{3}+\frac{5x^2\_c_2}{6}+\frac{13x^3\_c_2}{30}+O(x^4)\right]$$

We see that the solution is the same as the solution of the given equation `eq`. It shows that it is not sufficient just to construct the solutions of two random different prolongations for confirming exhaustive use of the information on a given equation. Supplementary information provided by the additional terms in a random prolongation does not necessarily lead to appearance of some additional terms in the equation solutions, so such a prolongation may not be used as a counterexample.

Example 3. Consider one more equation and its Laurent solution:

```
> eq := (x + O(x^2))*theta(y(x), x, 1) + O(x^2)*y(x);
```

$$eq := \left(x+O(x^2)\right)\theta(y(x),x,1)+O(x^2)y(x)$$

```
> sol := TruncatedSeries:-LaurentSolution(eq,y(x));
```

$$sol := [\_c_1+O(x)]$$

Instead of using procedure `ExhaustiveUseConfirmation`, it is possible to check exhaustive use of the information on the given equation using two additional implemented procedures step by step. This way may be a better than using the text printed by the procedure `ExhaustiveUseConfirmation` when, for example, the details of the counterexample are needed in some further algorithmic processing.

First, the invocation of the procedure `DifferentProlongationExtras` gives two different additional terms to construct two different prolongations of the given equation:

```
> dp := DifferentProlongationExtras(eq, y(x));
```

$$dp := \left[y(x)\left(-x^2 + O(x^3)\right), y(x)\left(x^2 + O(x^3)\right)\right]$$

Next, the procedure `ConstructProlongation` is applied twice to construct the equation prolongations.

```
> eq1 := ConstructProlongation(dp[1], eq, y(x));
```

$$eq1 := \left(x + O(x^2)\right)\theta(y(x), x, 1) + y(x)\left(-x^2 + O(x^3)\right)$$

```
> eq2 := ConstructProlongation(dp[2], eq, y(x));
```

$$eq2 := \left(x + O(x^2)\right)\theta(y(x), x, 1) + y(x)\left(x^2 + O(x^3)\right)$$

Finally, the Laurent solutions of each equation prolongation are constructed:

```
> sol1 := TruncatedSeries:-LaurentSolution(eq1, y(x))
```

$$sol1 := \left[\_c_1 + x\_c_1 + O(x^2)\right]$$

```
> sol2 := TruncatedSeries:-LaurentSolution(eq2, y(x))
```

$$sol2 := \left[\_c_1 - x\_c_1 + O(x^2)\right]$$

We can see that the different equation prolongations lead to two different solution prolongations.

Acknowledgments: We are grateful to Maplesoft (Waterloo, Canada) for consultations and discussions.

Funding: This work was supported by the Russian Foundation for Basic Research, project no. 19-01-00032.

References

1. *Abramov S.A, Khmelnov D.E., Ryabenko A.A.* Laurent solutions of linear ordinary differential equations with coefficients in the form of truncated power series. In: COMPUTER ALGEBRA, Moscow, June 17–21, 2019, International Conference Materials. P. 75–82.
2. *Abramov S.A, Khmelnov D.E., Ryabenko A.A.* Linear Ordinary Differential Equations and Truncated Series. Computational Mathematics and Mathematical Physics. 2019. Vol. 59, No. 10. P. 1649–1659.
3. *Abramov S.A, Khmelnov D.E., Ryabenko A.A.* Procedures for searching Laurent and regular solutions of linear differential equations with the coefficients in the form of truncated power series. Programming and Computer Software. 2020. Vol. 46, No. 2. P. 67–75.
4. Maple online help. http://www.maplesoft.com/support/help/

On Semantics of Names in Formulas and References in Object-Oriented Languages

And.V. Klimov[1]

[1]*Keldysh Institute of Applied Mathematics of RAS, Russia*

e-mail: klimov@keldysh.ru

Abstract. The long-standing problem of a formal semantics of free and bounded names in mathematical formulas, as well as the semantics of references in object-oriented languages, is discussed. We review the history of the topic and related works including recent contributions to the theory of names. An outline of a constructive denotational semantics of a functional language with a new/fresh name generator is presented.

Keywords: formal languages, free and bound variables, fresh names, constructive denotational semantics, nominal techniques, contextual equivalence, observational equivalence

1. Introduction: Names and References as Data

Computer algebra systems manipulate formulas with variables, which may be free or bound by quantifiers in nested scopes. Here, variables become elements of the data domain, while in normal use they belong to the language of mathematics, a meta-level compared to the level of ordinary mathematical values. Thus, we need a theory of data containing variables or names in order to prove correctness of symbolic manipulation.

It so happens that a similar problem has arisen in post-Algol programming languages which contain references to mutable objects as plain data. The concept of an object and reference is the cornerstone of object-oriented languages. Everything else is just syntactic constructs. Thus, the task of developing a mathematical semantics of modern languages requires a theory of references.

In fact, the domains of variables, names and references are one and the same. Classical mathematics and its set-theoretic foundation deal with values, which are significantly different from objects and references. Value equality and reference identity are quite different concepts. In Russian, we have special words to distinguish them in everyday life: “takoi zhe” and “tot zhe samyi”. Interestingly, the English word “same” fulfills both roles, and there is no such clear distinction. Speaking in Russian, we can say that classical mathematics is the theory of the “takoi zhe” relation between values, while computations in object-oriented languages deal with the “tot zhe samyi” relation between references and objects.

Below we explain the problem and some subtleties of developing a theory of names and references, point out the recent achievements of a group of mathematicians led by Andrew Pitts [6, 8, 9, 10] and present an outline of what can be called an *operational denotational semantics* of a functional language with names as data.

2. History and Related Works

The subtleties of manipulating terms and formulas with free and bounded variables are actually well-known. It is worth mentioning the trick in N.Bourbaki’s treatise [3, see the first page of Chapter 1] with bars (referred to as *links* in [3]) above lines of text in the basic formal language (referred to as *assembly* in [3]). This turns it upside down by using at the foundation of mathematics the non-trivial notion of a graph, which requires its own

definition. Set theory, being the heart of modern mathematics, does not capture the graph notions directly. Only a tree is a natural datum in classical set theory and traditional mathematics, everything else is a construct.

For a while in history, name manipulation was considered only at the meta-level, at the level of a mathematical or programming language (not counting the rare special construction of passing an argument by name as in Algol-60), rather than at the level of ordinary data as first-class citizens. The problem with variable names has become more acute in software systems for processing mathematical expressions. While developing a computer algebra system AUTOMATH, N.G. de Bruijn invented indices, later named after him, which are the representation of a bound variable by an integer denoting the number of lambdas from an occurrence of a variable to its lambda [4]. Needless to say, this is more of a programming trick for this particular case than an adequate formalization that reflects the essence.

The idea of a proper notion of a name and a reference to a mutable object as data gradually penetrated into programming languages in the 1960s. The first properly designed language with references to mutable objects as plain data was Simula-67 [5]. It established the main concepts of object-oriented programming.

In the 1980s, when object-oriented programming became widespread, it was realized that objects and references are a significantly different kind of data than values, such as numbers, strings, lists, etc. One of the first papers (as far as I know) in which this question was explicitly raised and discussed was [7]. Classical mathematics deals only with values, it lacks the concepts to directly talk about names and references. Thus, the problem of a semantics of names actually arose, but for a long time it was not given due attention.

Since the early 1990s, category theory mathematician Andrew Pitts has pioneered research into the proper mathematical foundations of names and references as data. The papers [6, 8, 9, 10] are just some of the many interesting works describing the results by his group. In 2019 the contribution of Murdoch Gabbay and Andrew Pitts was recognized with the ACM Alonzo Church Award "for their ground-breaking work introducing the theory of nominal representations" [1]. They developed two approaches to the semantics of names: in Set Theory based on Fraenkel-Mostowski permutation model and in Category Theory. We hypothesize that the categorical approach is more promising as a foundation for proving properties of symbolic manipulation and object-oriented program verification.

3. Constructive Denotational Semantics of a Language with Names

Let us show the essence of the topic by presenting an outline of a semantics of a language with names or references. Consider an arbitrary pure functional language to your taste – be it statically typed (as Haskell) or untyped (as Lisp). A first-order subset (without lambda expressions) is sufficient for our purposes. To write examples, we use common mathematical notation for values, terms, and function definitions. Any of the usual operational semantics of a functional language is appropriate.

Now consider the following extension of the language (like FreshML in [10]). Extend the data domain with names that are atomic in the sense that each one has no other property than being equal to itself and unequal to other data. The values passed as arguments and returned by functions are like the usual ones, but possibly containing names at the tips.

Extend the functional language with a **new** operator, which generates a new, fresh name each time it is called, which by definition is unequal to any name already encountered in computation. For example, a term $f(\mathbf{new})$ returns a pair of two different new names where the function f is defined as follows (the sign ":=" means "is by definition"):

$$f(x) := \langle \mathbf{new}, x \rangle$$

What are the properties of such a language? First, the fundamental property of the mathematical language and pure functional programming languages, called *referential transparency*, is violated. This means that several evaluations of a term can produce unequal results, e.g., $\mathbf{new} \neq \mathbf{new}$ and $f(x) \neq f(x)$ for any x, where f is as above.

Second, the usual *set-theoretic denotational semantics* of functional languages does not work, since each term must be assigned a single *denotation*, which can be used in place of all occurrences of the term, preserving the meaning. But the result of a function call, which contain new names, e.g., **new** in the simpest case, cannot serve as such denotation. Moreover, the data domain with names is not a set in the classical sense. Of course, non-standard set-theoretical models of this domain can be constructed, but this complicates the situation. Such theories have been developed by the group led by Andrew Pitts [6, 8, 9, 10].

Have any properties been preserved after such an extension? Indeed, instead of equality, there is the *observational* or *contextual equivalence* of the results of several evaluations of the same term. It means that values are equivalent if they cannot be distinguished by any predicate defined in the language. Denoting this equivalence with the sign $\approx$, we have $\mathbf{new} \approx \mathbf{new}$, $f(x) \approx f(x)$ for any x; moreover, for any f, x, y:

$$x \approx y \Rightarrow f(x) \approx f(y)$$

Based on this property, various nice semantics can be defined for a language with names: in set theory in a certain non-standard way, in category theory, as well as operational semantics that does not require computation time to define the generation of fresh names. Let us give an idea and an outline of the latter.

Let the denotation of a closed term T be not the result value y, but a term of the following form obtained from y by considering as variables all new names $n_1, \ldots, n_k$ produced during evaluation of T; the list of names is prepended to y along with a sign ν:

$$\nu\, n_1 \ldots n_k .\, y$$

Different results of each act of evaluation of the term T are obtained from this denotation by evaluating the following lambda expression:

$$(\lambda\, n_1 \ldots n_k .\, y)\, \mathbf{new} \ldots \mathbf{new}$$

which means: substitute unique new names for all variables $n_1, \ldots, n_k$ in y.

Next, let the denotation of a function f be a set of terms of the following form (rather than a set of argument-value pairs, as usual):

$$\lambda\, m_1 \ldots m_l .\, x \;\mapsto\; \nu\, n_1 \ldots n_k .\, y$$

Such function denotation determines the result of evaluation of a function call $f(a)$ as follows. Find an element of the denotation with left-hand side $\lambda\, m_1 \ldots m_l .\, x$ and names $b_1, \ldots, b_l$ occurring in a such that $a = (\lambda\, m_1 \ldots m_l .\, x)\, b_1 \ldots b_l$. Then return the result of the following lambda expression:

$$(\lambda\, m_1 \ldots m_l .\, x)\, ((\lambda\, n_1 \ldots n_k .\, y)\, \mathbf{new} \ldots \mathbf{new})\, b_1 \ldots b_l .$$

Operationally, the denotations of functions of a program are gradually collected during computation by constructing such elements from argument-value pairs: $m_1 \ldots m_l$ are all the names occurring in the argument, $n_1 \ldots n_k$ are the names generated during evaluation of the function call. It can be proved that plain evaluation and computation using these denotations produces contextually equivalent results.

We see that this semantics satisfies the usual requirements to the denotational semantics: each term and function in a program has a denotation, although denotations are constructive terms rather than elements of the set-theoretical world. Therefore, it is referred to as a *constructive denotational semantics*.

4. Conclusion

We considered the problem of adequate formalization of names in mathematical formulas and references in object-oriented languages. The intention was to have a formal semantics of free and bound names that is simpler than the usual formalization using imperative concepts of time and sequential processing when it comes to generating fresh names. The essence of the problem and its possible solutions were shown for a functional language extended by a new/fresh name generator.

To formulate the object under study, an outline of a constructive denotational semantics for such a language was presented. It is based on the recent achievements in the field of *nominal techniques* of the group led by Andrew Pitts.

The formalization of names as data objects in a functional language is a prerequisite for the full semantics of objects with mutable states in object-oriented languages. This topic was not discussed above; this is our further work. We study it with the aim of developing a deterministic parallel programming language that lies between purely functional and object-oriented languages [2].

References

1. *ACM Special Interest Group on Logic and Computation.* Winners of the 2019 Alonzo Church Award. 2019. URL: https://siglog.org/winners-of-the-2019-alonzo-church-award/
2. *Adamovich A.I., Klimov And.V.* How to create deterministic by construction parallel programs? Problem Statement and Survey of Related Works. Program Systems: theory and Applications. 2017. Vol. 8, No. 4 (35). P. 221–244.
3. *Bourbaki N.* Elements of Mathematics. Theory of Sets. Moscow: Mir Publishers, 1965. (In Russian; transl. from French)
4. *de Bruijn N.G.* Lambda calculus notation with nameless dummies: A tool for automatic formula manipulation, with application to the Church-Rosser theorem. Indagationes Mathematicae. 1972. Vol. 34. P. 381–392.
5. *Dahl O.-J., Myhrhaug B., Nygaard K.* SIMULA-67, a universal programming language. Moscow: Mir Publishers, 1969. (In Russian; transl. from English)
6. *Gabbay M.J., Pitts A.M.* A new approach to abstract syntax with variable binding. Formal Aspects of Computing. 2002. Vol. 13, No. 3. P. 341–363.
7. *MacLennan B.J.* Values and objects in programming languages. SIGPLAN Not. 1982. Vol. 17, No. 12. P. 70–79.
8. *Pitts A.M.* Nominal logic, a first order theory of names and binding. Information and Computation. 2003. Vol. 186, No. 2. P. 165–193.
9. *Pitts A.M.* Nominal sets: names and symmetry in computer science. Cambridge Tracts in Theoretical Computer Science. 2013. Vol. 57.
10. *Pitts A.M., Gabbay M.J.* A metalanguage for programming with bound names modulo renaming. Lecture Notes in Computer Science. 2000. Vol. 1837. P. 230–255.

Subsystems in Finite Quantum Mechanics

V.V. Kornyak[1]

[1]*Laboratory of Information Technologies*
Joint Institute for Nuclear Research, Dubna, Russia
e-mail: vkornyak@gmail.com

Abstract. Any Hilbert space with composite dimension can be represented as a tensor product of smaller Hilbert spaces. This allows to decompose a quantum system into subsystems. We propose a model based on finite quantum mechanics for the constructive study of such decompositions.

Keywords: quantum mereology, finite quantum mechanics, ontic basis, energy basis

Introduction

Mereology is the study of the part-to-whole and part-to-part relations within the whole. In *quantum mereology*, the whole is an isolated quantum system ("the universe") in a given pure state, undergoing a given unitary (Schrödinger) evolution. Quantum mereology studies the interrelations between singled out subsystems of the universe ("observable system", "observer", "environment", etc.), the emergence of geometry from quantum entanglement, and other fundamental issues of quantum mechanics.

We develop and implement algorithms based on computer algebra techniques to perform the following. An isolated quantum system, constructed in the framework of finite quantum mechanics, is decomposed into a tensor product of subsystems. By reducing the "universe" quantum state, we obtain mixed states for subsystems. This allows us to study energy interactions and quantum correlations between subsystems and their time evolution.

Decomposition of a Quantum System

Tensor product of Hilbert spaces. The (*global*) Hilbert space $\mathcal{H}$ of a K-component quantum system is the tensor product of the component (*local*) Hilbert spaces $\mathcal{H}_k$

$$\mathcal{H} = \mathcal{H}_1 \otimes \cdots \otimes \mathcal{H}_k \otimes \cdots \otimes \mathcal{H}_K. \tag{1}$$

If $\dim \mathcal{H} = \mathcal{N}$ and $\dim \mathcal{H}_k = d_k$, then $\mathcal{N} = d_1 \cdots d_K$. For any d-dimensional Hilbert space, the ith orthonormal basis element is denoted by $|i\rangle$, that is, $|0\rangle = (1, 0, \ldots)^\top$, $|1\rangle = (0, 1, 0, \ldots)^\top, \ldots, |d-1\rangle = (0, 0, \ldots, 1)^\top$. Tensor monomials of local basis elements form an orthonormal basis in the global Hilbert space

$$|i\rangle = |i_1\rangle \otimes \cdots \otimes |i_k\rangle \otimes \cdots \otimes |i_K\rangle\,, \tag{2}$$

where $|i\rangle \in \mathcal{H}$, $|i_k\rangle \in \mathcal{H}_k$ and

$$i = i_1\,(d_2 \cdots d_K) + \ldots + i_k\,(d_{k+1} \cdots d_K) + \ldots + i_K. \tag{3}$$

Tensor factorization of a Hilbert space. We can reverse the procedure, since (2) is a bijection: the sequence $i_1, \ldots, i_K$ is uniquely recovered from i by (3). Given an orthonormal basis in an $\mathcal{N}$-dimensional Hilbert space $\mathcal{H}$ and a decomposition $\mathcal{N} = d_1 \cdots d_K$, we can construct a particular isomorphism of the form (1). When constructing an isomorphism, we must take into account the freedom in the choice of basis: a unitary transformation of the global space $\mathcal{H}$ changes the isomorphism. Thus, to specify a factorization of a Hilbert space $\mathcal{H}$ we need a *decomposition* of $\dim \mathcal{H}$ and a *unitary transformation* U that fixes a basis in $\mathcal{H}$.

Decomposition of a pure quantum state. Since any mixed state of a quantum system can be obtained from a pure state in a larger Hilbert space by taking a partial trace, it is natural to assume that at a fundamental level the state of an isolated system must be pure. For a given factorization $\dim \mathcal{H} = d_1 \cdots d_K$, we introduce the set (of *"geometric points"*)

$$X = \{1, \ldots, K\}\,. \tag{4}$$

Subsystems are identified with subsets $A \subseteq X$. The density matrix of the pure state $|\psi\rangle \in \mathcal{H}$ of the entire system is $\rho = \dfrac{|\psi\rangle\langle\psi|}{\langle\psi \mid \psi\rangle}$. To take into account the unitary freedom, we introduce the matrix $\rho^U = U\rho U^\dagger$. According to the laws of quantum mechanics, the statistical behavior of the subsystem A is correctly described by the reduced density matrix $\rho^U_A = \mathrm{tr}_{X\setminus A}\,\rho^U$ calculated by taking the partial trace over the complement to A.

Finite Quantum Mechanics

We use a version of quantum theory [1] in which the groups of unitary evolutions are replaced by linear representations of finite groups, and the field of complex numbers is replaced by its dense constructive subfields that naturally arise from the non-negative integers and roots of unity.

Permutation Hilbert space *Any* linear (hence *unitary*) representation of a finite group is a subrepresentation of some permutation representation. This implies that the formalism of quantum mechanics can be completely[1] reproduced based on permutations of some set

$$\Omega = \{e_1, \ldots, e_{\mathcal{N}}\} \cong \{1, \ldots, \mathcal{N}\}$$

of primary (*"ontic"*) objects on which a permutation group $G \leq \mathsf{S}_{\mathcal{N}}$ acts.

The Hilbert space on Ω, needed for calculations in quantum theory, can be most economically constructed on the basis of two primitive concepts: (a) *natural numbers* $\mathbb{N} = \{0, 1, \ldots\}$, abstraction of *counting*, and (b) *roots of unity*, abstraction of *periodicity*.

To construct a field $\mathcal{F}$ sufficient for all the needs of the quantum formalism, we can proceed as follows. We extend the semiring $\mathbb{N}$ to the ring $\mathbb{N}[\zeta_\ell]$, where ζ_ℓ is the ℓth primitive root of unity, and ℓ is the LCM of the periods of the elements of G. The *algebraic integer* ζ_ℓ can be written in complex form as $\zeta_\ell = \mathrm{e}^{2\pi\mathbf{i}/\ell}$. Finally, constructing the *quotient field* of the ring $\mathbb{N}[\zeta_\ell]$, we arrive at the *cyclotomic extension* of the rationals $\mathcal{F} = \mathbb{Q}\big(\mathrm{e}^{2\pi\mathbf{i}/\ell}\big)$. For $\ell > 2$, the field $\mathcal{F}$, being a dense subfield of $\mathbb{C}$, is empirically indistinguishable from $\mathbb{C}$.

Treating the set Ω as a basis, we obtain an $\mathcal{N}$-dimensional Hilbert space $\mathcal{H}_{\mathcal{N}}$ over $\mathcal{F}$. The action of G on Ω determines the *permutation representation* $\mathcal{P}$ in $\mathcal{H}_{\mathcal{N}}$ by the matrices $\mathcal{P}(g)_{i,j} = \delta_{ig,j}$, where ig denotes the (right) action of $g \in G$ on $i \in \Omega$.

Decomposition of permutation representation The permutation representation of any group G has the *trivial* one-dimensional subrepresentation in the space spanned by the all-ones vector $|\omega\rangle = (1, 1, \ldots, 1)^\top$. The complement to the trivial subrepresentation is called the *standard representation*. The operator of projection onto the $(\mathcal{N}-1)$-dimensional *standard space* $\mathcal{H}_\star$ has the form

$$\mathrm{P}_\star = \mathbb{1}_{\mathcal{N}} - \frac{|\omega\rangle\langle\omega|}{\mathcal{N}}\,.$$

Quantum mechanical behavior (interference, etc.) manifests itself precisely in $\mathcal{H}_\star$. Banks made a profound observation [2] that the projection of classical permutation evolutions in the whole $\mathcal{H}_{\mathcal{N}}$ leads to truly quantum evolutions in the subspace $\mathcal{H}_\star$. Banks also showed that the choice $G = \mathsf{S}_{\mathcal{N}}$, where $\mathcal{N}$ is the number of fundamental (Planck) elements,[2] "can accurately reproduce all of the results of conventional quantum mechanics".

[1] *Modulo* empirically insignificant elements of traditional formalism such as infinities of various kinds.

[2] By the current cosmological data, the number $\mathcal{N}$ is estimated as $\sim \mathrm{Exp}(\mathrm{Exp}(20))$ and $\sim \mathrm{Exp}(\mathrm{Exp}(123))$ for 1 cm^3 of matter and for the entire Universe, respectively.

Ontic vectors. $\mathsf{S}_\mathcal{N}$ is a *rational-representation group*, i.e., its every irreducible representation (the standard representation is one of them) is realizable over $\mathbb{Q}$. This means that to describe evolutions in $\mathcal{H}_\star$, it is sufficient to consider only vectors with rational components (complex numbers – nontrivial elements of cyclotomic extensions – may be needed only in problems that require splitting representations of some proper subgroups of $\mathsf{S}_\mathcal{N}$ into irreducible components). It is easy to show that any quantum state in $\mathcal{H}_\star$ can be obtained by projection of vectors from $\mathcal{H}_\mathcal{N}$ with natural components. To build constructive models, we need to select a finite subset in the set of natural vectors.

Ontic vectors are vectors with coordinates from the set $\{0,1\}$, i.e., bit strings of length $\mathcal{N}$. These vectors are attractive for both ontological and computational reasons. Interpreting ontic vector $|q\rangle$ as a *characteristic function*, we can identify it with the subset $q \subset \Omega$. The complete set of ontic vectors is $Q = 2^\Omega \setminus \{\emptyset, \Omega\}\,, \quad |Q| = 2^\mathcal{N} - 2\,.$ The number of ontic vectors depends exponentially on $\mathcal{N}$, i.e., they represent quantum states fairly well at large $\mathcal{N}$.

The inner product of ontic vectors $|q\rangle$ and $|r\rangle$ in $\mathcal{H}_\mathcal{N}$ is $\langle q \mid r\rangle = \langle q \& r\rangle$, where $\&$ is the bitwise AND for bit strings, and $\langle\,\cdot\,\rangle$ denotes *population number* (or *Hamming weight*). The inner product of normalized projections onto $\mathcal{H}_\star$ is $S(q,r) = \dfrac{\mathcal{N}\langle q\,\&\,r\rangle - \langle q\rangle\,\langle r\rangle}{\sqrt{\langle q\rangle\,\langle\sim q\rangle\,\langle r\rangle\,\langle\sim r\rangle}}$, where $\sim$ is *bitwise inversion*. The identities $\langle\sim a\rangle = \mathcal{N} - \langle a\rangle$ and $\langle a\,\&\,b\rangle + \langle a\,\&\sim b\rangle = \langle a\rangle$ imply the folowing symmetry with respect to transpositions of subsets of the ontic set and their complements $S(q,r) = -S(\sim q, r) = -S(q, \sim r) = S(\sim q, \sim r)\,.$

Ontic and Energy Bases

Ontic basis. The original permutation basis in the space $\mathcal{H}_\mathcal{N}$, i.e., the set Ω, will be referred as the *ontic basis*. In this basis, the pure density matrix in $\mathcal{H}_\star$ associated with the ontic state $|q\rangle \in \mathcal{H}_\mathcal{N}$ has the form

$$\rho^o_q = \frac{\mathrm{P}_\star\,|q\rangle\langle q|\,\mathrm{P}_\star}{\langle q\,|\mathrm{P}_\star|\,q\rangle} = \frac{1}{\mathcal{N}}\frac{(|q\rangle - \alpha\,|\omega\rangle)\,(\langle q| - \alpha\,\langle\omega|)}{\alpha\,(1-\alpha)}\,, \tag{5}$$

where $\alpha = \langle q\rangle\,/\mathcal{N}$ is the *population density*. There is an obvious duality: the expression for the density matrix $\rho^o_{\sim q}$ is obtained from (5) by replacements $q \to \sim q$ and $\alpha \to 1-\alpha$.

Energy basis. In continuous QM, the evolution of an isolated system is described by the one-parameter unitary group $U_t = \mathrm{e}^{-\mathbf{i}Ht}$ generated by the Hamiltonian H whose eigenvalues are called *energy eigenvalues*. In finite QM, the evolution is described by a cyclic group $U(g)^t$ generated by an element $U(g) \in \mathcal{P}(G)$, where t is an integer parameter. We call the *energy basis* an orthonormal basis in which the matrix $U(g)$ is diagonal.

Any permutation is a product of disjoint cycles. One can show that the total number of cycles of length ℓ in the whole group $\mathsf{S}_\mathcal{N}$ is $\mathcal{N}!/\ell$, and, therefore, the expected number of ℓ-cycles in a single permutation is $1/\ell$. That is, high-energy evolutions are more common.

The ℓ-cycle matrix has the form $(C_\ell)_{ij} = \delta_{i-j+1 \pmod\ell}$. The diagonal form of this matrix is $F_\ell C_\ell F_\ell^{-1} = \mathrm{diag}\big(1, \zeta_\ell, \zeta_\ell^2, \ldots, \zeta_\ell^{\ell-1}\big)$, where $\zeta_\ell = \mathrm{e}^{2\pi\mathbf{i}/\ell}$ is the ℓth primitive root of unity, and $(F_\ell)_{ij} = \zeta_\ell^{-(i-1)(j-1)}/\sqrt{\ell}$ is the *Fourier transform* matrix. F_ℓ is both unitary and symmetric, therefore $F_\ell^{-1} = F_\ell^\dagger = F_\ell^* \implies \big(F_\ell^{-1}\big)_{ij} = \zeta_\ell^{(i-1)(j-1)}/\sqrt{\ell}$. In general, the matrix of the permutation representation of an element $g \in \mathsf{S}_\mathcal{N}$ is the direct sum of cyclic matrices $U(g) = \bigoplus_{m=1}^{M} C_{\ell_m}$, and the corresponding diagonalizing matrix is $F = \bigoplus_{m=1}^{M} F_{\ell_m}$, which is the transition matrix from the *ontic basis* to the *energy basis*. The density matrix in the energy basis can be calculated from (5) by the formula $\rho^e_q = F\rho^o_q F^*$.

Entanglement Measures

Quantitatively, quantum correlations are described by *measures of entanglement*, which are based on the concept of entropy. The most commonly used in physics is the *von Neumann entropy* $S_1(\rho) = -\operatorname{tr}(\rho \log \rho)$. Also often used are entropies from the *Rényi family*

$$S_\alpha(\rho) = \frac{1}{1-\alpha} \log \operatorname{tr}(\rho^\alpha), \quad \alpha \geq 0, \ \alpha \neq 1.$$

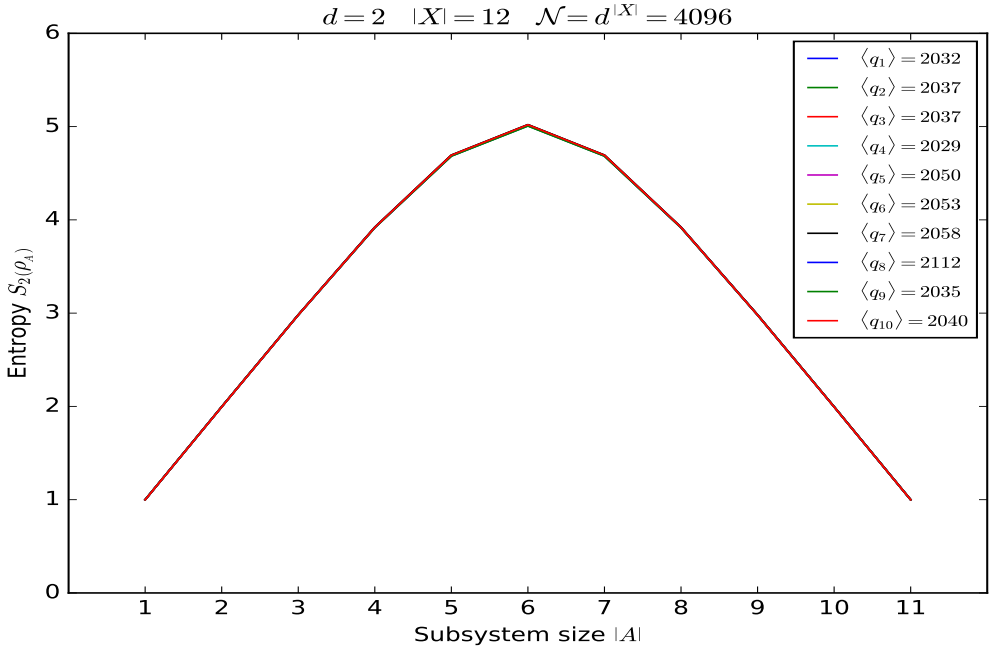
In our calculations, we use the 2nd Rényi entropy $S_2(\rho) = -\log \operatorname{tr}\left(\rho^2\right)$ (also called the *collision entropy*) for the following reasons:

(a) It is easy to calculate: $\operatorname{tr}\left(\rho^2\right) = \sum_{i=1}^{n} \rho_{ii}^2 + 2 \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} |\rho_{ij}|^2$.

(b) $\operatorname{tr}\left(\rho^2\right)$ coincides with the Born probability: "*the system observes itself*".

(c) $\operatorname{tr}\left(\rho^2\right)$ is squared *Frobenius* (*Hilbert-Schmidt*) *norm* of the matrix ρ.

Example of Calculation



The figure presents the values of entropy $S_2(\rho_A)$ computed in the ontic basis for the decomposition $\mathcal{N} = 2^{12}$. Data for subsystems of all possible sizes, computed for ten randomly generated ontic vectors, demonstrate the following features:

(a) Weak dependence on the quantum state: visually, all graphs are almost identical. This behavior arises for a sufficiently large number of decomposition components (4).

(b) Symmetry $S_2(\rho_A) = S_2\left(\rho_{X \setminus A}\right)$ is a manifestation of the *Schmidt decomposition* of a pure state: both matrices ρ_A and $\rho_{X \setminus A}$ have identical sets of nonzero eigenvalues.

(c) For $|A|$ noticeably smaller than $|X|/2$, the reduced state is close to the *maximally mixed* state: $S_2(\rho_A) \approx |A| \log d$. In our example, $d = 2$.

References

1. *Kornyak V.V.* Quantum models based on finite groups. IOP Conf. Series: Journal of Physics: Conf. Series. **965**, 012023, 2018. https://arxiv.org/pdf/1803.00408.pdfarXiv:1803.00408 [physics.gen-ph]
2. *Banks T.* Finite Deformations of Quantum Mechanics. https://arxiv.org/pdf/2001.07662.pdfarXiv:2001.07662 [hep-th], 20 p., 2020.

About Big Matrix Inversion

G. Malaschonok[1], I. Tchaikovsky[2]
[1]*National University of Kyiv-Mohyla Academy, Ukraine*
[2]*Lviv Polytechnic National University, Ukraine*
e-mail: malaschonok@ukma.edu.ua, ihortch@yahoo.com

Abstract. We consider three types of inverse matrices: inverse, pseudoinverse, and generalized inverse. And we discuss algorithms, which are applicable for commutative domains. The research is motivated by modern problems of supercomputing.

Keywords: inverse matrix, pseudoinverse matrix, generalized inverse matrix, commutative domain, supercomputing

1. On the boundary between big data and small data

From the point of view of computational mathematics, the size of the input problem largely determines the accumulation of computational error.

When the calculation error reaches the value of the sought numbers, then a revision of the computational algorithm is required. If algorithms demonstrate such an error starting from a certain matrix size, then this size can be considered a borderline between large and small data.

In paper [1], all matrix algorithms are classified into three classes:
(MA_1) the rational matrix algorithms,
(MA_2) the irrational matrix algorithms, which are expressed in radicals,
(MA_3) the irrational iterative matrix algorithms, which are not expressed in radicals.

For the experiment, the Cholesky algorithm for decomposition of a symmetric positive definite matrix was chosen, which belongs to class MA_2. To obtain the initial matrix for decomposition, triangular matrices with integer coefficients between 1 and 9 were randomly chosen and multiplied by their transposed matrix. Thus, the correct decomposition result was known in advance. The calculations were performed with double precision and 100 experiments were performed for matrices of the same size. Subtracting the matrix obtained in the decomposition from the exact one, the error was found as the largest number in absolute value.

Table 1 shows the results of such experiments. The first line shows the maximum error per 100 experiments, and the second line shows the average error.

Table 1. The value of the calculation error in the Cholesky decomposition algorithm

Matrix size	4×4	$8x8$	16×16	32×32	64×64
Maximum error value	$2 \cdot 10^{-13}$	10^{-10}	$3 \cdot 10^{-6}$	0.6	142
Average error value	$6 \cdot 10^{-15}$	$4 \cdot 10^{-12}$	$6 \cdot 10^{-8}$	0.01	7.9

As can be seen from the table, the average error reaches the value of the input numbers already for matrices of size 64.

These experiments show that size 64 is the boundary between big data and small data for this triangular decomposition problem.

2. Three types of inverse matrices

We consider three types of inverse matrices: inverse, pseudoinverse, and generalized inverse. We are interested in class 1 algorithms, since for these algorithms it is possible to overcome the problem of accumulating computational error if all calculations are carried out in the commutative domain. So we discuss algorithms,which are applicable for commutative domains.

Definition 1. *Matrix A^{-1} is the inverse of a square nondegenerate matrix A if two equalities*

$$A^{-1}A = AA^{-1} = I \tag{1}$$

are true.

Definition 2. *Matrix $A^{\times}$ is a pseudo inverse for matrix A if two equalities*

$$A^{\times} = A^{\times}AA^{\times},\ A = AA^{\times}A \tag{2}$$

are true.

Definition 3. *Matrix A^{+} is a generalized inverse (generalized inverse Moore-Penrose matrix) for matrix A if four equalities*

$$A^{+} = A^{+}AA^{+},\ A = AA^{+}A,\ (A^{+}A)^{T} = A^{+}A,\ (AA^{+})^{T} = AA^{+} \tag{3}$$

are satisfied.

If matrix A is a square and nondegenerate matrix, then all three types of inverse matrices coincide

$$A^{+} = A^{\times} = A^{-1}.$$

Let a matrix A of size $n \times m$ be decomposed as follows: $A = B \cdot C$, $B \in F^{n \times k}$, $C \in F^{k \times m}$, $rank(A) = rank(B) = rank(C) = k$.

It is easy to check that matrix

$$A^{+} = C^{T}(CC^{T})^{-1}(B^{T}B)^{-1}B^{T}, \tag{4}$$

is the generalized inverse matrix for the matrix A. This idea was first expressed by Vera Kublanovskaya in 1965 [2].

3. Algorithms for commutative domains

The first exact matrix inversion algorithm, in which the inverse of the matrix is represented as the ratio of the adjoint matrix and the determinant, and only n^3 integer operations were used, was presented in 1981 and published in 1983 [3].

The most famous of the previous works is [4]. However, the algorithm proposed in [4] did not allow finding the adjoint matrix or the numerators and denominators of fractions, which are elements of the inverse matrix or the solution of a system of linear equations, due to the presence of many unnecessary factors. Apparently, this fact is not generally known, since the work [4] is still the most cited, and the work [3] is little known to specialists.

With regard to additional permutations, in this issue, algorithm [3] is similar to the standard Gaussian elimination algorithm. As you know, if the Gaussian elimination algorithm is applied to a matrix of the form $[A, I]$, here I is the identity matrix, then it allows you to find the inverse matrix for the matrix A. If at the same time a leading element equal to zero

is encountered, then it is necessary to perform permutations and preserve the permutation matrices. At the end of the calculation, these permutation matrices are applied to obtain the desired adjoint matrix for matrix A.

If the original matrix is a dense integer matrix jf size n and standard algorithms for integers are used, then the total bit complexity will be $\sim n^5$, and if you use the Chinese remainder theorem, then the total bit complexity will be $\sim n^4$.

Just like the Gauss algorithm, this algorithm is efficient for sequential algorithms and for parallel algorithms that are used in computers with shared memory.

However, it is not efficient for modern supercomputers with tens and hundreds of thousands of processors, as finding a pivot and moving rows and columns create delays that can destroy all the advantages of a supercomputer.

AdjDet algorithm

The search for an algorithm effective for a supercomputer continued for a long time. A new block-recursive algorithm $AdjDet$ was published in [5] and [6].

The main advantage of the new algorithm was the rejection of permutations of rows and columns of the matrix in favor of local multiplication by the permutation matrix for a separate block.

It should be noted that there are other advantages of this algorithm: less complexity, the ability to calculate the kernel of the matrix operator and the generalized inverse matrix.

Let A be a matrix of rank r. If A is a square matrix of size r, then $A^{-1} = Adj(A)/Det(A)$. If this is not the case, then we can calculate the generalized inverse matrix.

Algorithm $AdjDet$ returns permutation matrices P and Q that move the largest nondegenerate minor to the upper left corner. Let A_0 be such nondegenerate minor of size r, $d = \det(A_0)$, located in the upper left corner. Let A_U be a submatrix formed by the first r rows, A_L be a submatrix formed by the first r columns. Then the matrix A is decomposed into the product of three matrices: $A = (1/d)A_L Adj(A_0)A_U$ Hence,

$$A^+ = (1/d)PA_U^*(A_U A_U^*)^{-1}A_0(A_L^* A_L)^{-1}A_L^* Q.$$

Here matrices $A_U A_U^*$ and $A_L^* A_L$ of size $r \times r$ are invertible.

LDU algorithm

Another algorithm for calculating inverse matrices was recently obtained [7]. This is a block-recursive LDU triangular decomposition algorithm: $A = LDU$. Triangular matrices L and U belong to the same commutative domain as matrix A, and the matrix of weighted permutations D has the same rank as the matrix A.

This algorithm computes the inverse matrices M and W to the matrices L and U: $LdM = I$ and $WdU = I$. All these matrices have full rank. Matrix d is a matrix of weighted permutations and it associated with D. All other matrices belong to the commutative domain.

If matrix A is invertible, then its inverse is calculated as follows:

$$A^{-1} = U^{-1}D^+L^{-1} = WdD^+dM = WDM.$$

If matrix A is not invertible, then its psevdo inverse matrix is equal $A^\times = WDM$. We can easily check the fulfillment of equalities (2):

$$A^{\times}AA^{\times} = U^{-1}D^{+}L^{-1}LDUU^{-1}D^{+}L^{-1} = U^{-1}D^{+}L^{-1} = A^{\times}$$
$$AA^{\times}A = LDUU^{-1}D^{+}L^{-1}LDU = LDU = A.$$

However, calculating the generalized inverse matrix requires additional effort.

Let the matrix A be decomposed into the product of three matrices $A = BDC$, that have the same rank r. Where D is the matrix of weighted permutations, C is the matrix of columns corresponding to the matrix D, B is the matrix of strings corresponding to the matrix D. Then we can find the generalized inverse matrix thanks to the basic form:

$$A^{+} = C^{*}(CC^{*})^{-1}D^{+}(B^{*}B)^{-1}B^{*}. \tag{5}$$

We denote by I and J diagonal matrices, whose rank is the same as the rank of the matrix D, and which correspond to nonzero rows and columns of matrix D: $IDJ = D$.

Then we can write $A = (LI)D(JU) = BDC$ with $LI = B$ and $JU = C$ and use the basic form (5), in which matrices CC^{*} and $B^{*}B$ are invertible matrices of rank r.

References

1. *Malaschonok G.* Recursive matrix algorithms, distributed dynamic control, scaling, stability. Proc. of 12th Int. Conf. on Comp. Sci. and Information Technologies (CSIT-2019). September, Yerevan. P. 23–27.
2. *Kublanovskaya V.N.* Evaluation of a generalized inverse matrix and projector. USSR Computational Mathematics and Mathematical Physics. 1966. Vol. 6, No. 2. P. 179–188. (In Russian)
3. *Malashonok G.I.* Solution of a system of linear equations in an integral domain. USSR J. of Comput. Math. and Math. Phys. 1983. Vol. 23, No. 6. P. 1497–1500. arXiv:1711.09452
4. *Bareiss E.N.* Sylvester's identity and multistep integer-preserving Gaussian elimination. Math. Comput. 1968. Vol. 22. P. 565–578.
5. *Malaschonok G.I.* On computation of kernel of operator acting in a module. Tambov University Reports. Natural and Technical Sciences. 2008. Vol. 13, part. 1. P. 129–131. (In Russian)
6. *Malaschonok G. and Ilchenko E.* Recursive matrix algorithms in commutative domain for cluster with distributed memory. 2018 Ivannikov Memorial Workshop (IVMEM), Yerevan, Armenia. P. 40–46. doi: 10.1109/IVMEM.2018.00015. arXiv:1903.04394
7. *Malaschonok G.* LDU-factorization. E-print 2011.04108. 2020. P. 1–16. arXiv:2011.04108

Solving the Hyperbolic Equation in Elementary Functions

M.D. Malykh[1], K.Yu. Malyshev[2]

[1]*Peoples' Friendship University of Russia, Russia*

[2]*Lomonosov Moscow State University*

Skobeltsyn Institute of Nuclear Physics (SINP MSU)

e-mail: malykh_md@pfur.ru, kmalyshev08102@mail.ru

Abstract. The classical problem about oscillation of music strings is considered. Its solution can be represented as Fourier series, and also as elementary functions. Such a representation is valid only over the field of real numbers and doesn't belong Liouville theory. This circumstance lids to several difficulties at work with the solutions in CAS. A typical example from CAS Maple is considered.

Keywords: Liouville class, d'Alembert method, series summation

1. Description of string oscillations in finite terms

In classical mathematical physics there are several problems that can be solved in elementary functions but can't be solved in such a form in computer algebra systems (CAS). The simplest one of them is follows.

Let the variable x belong the interval $[0, l]$, and the variable t be positive. We want to find a smooth function u of x, t that verify to the problem

$$\begin{cases} \dfrac{\partial^2 u}{\partial x^2} = \dfrac{1}{c^2}\dfrac{\partial^2 u}{\partial t^2}, \\ u|_{x=0} = u|_{x=l} = 0, \\ u|_{t=0} = \varphi(x), \ \left.\dfrac{\partial u}{\partial t}\right|_{t=0} = 0, \end{cases} \tag{1}$$

which describe, for example, oscillations of strings. If we denote the odd $2l$-periodic extension of the function φ as $\hat{\varphi}$, then we can write the solution of the problem by d'Alembert formula:

$$u = \frac{\hat{\varphi}(x + ct) + \hat{\varphi}(x - ct)}{2}. \tag{2}$$

To construct the odd $2l$-periodic extension of the given elementary function we need to have two functions: the odd $2l$-periodic function $\theta_1(x)$ which is equal to 1 on the interval $0 < x < l$ and the even $2l$-periodic function $\theta_2(x)$ which is equal to x on the interval $0 < x < l$. Then the required extension can be written as

$$\hat{\varphi}(x) = \theta_1(x)\varphi(\theta_2(x)).$$

Mathematicians of XIX century investigated analytic functions over the field $\mathbb{C}$ by default. Thus elementary functions in Liouvillian theory are investigated as analytic. Furthermore, in that time there is a discussion about piecewise expressions like $|x|$, more authors did not want to consider them as function at all [1]. The functions θ_1 and θ_2 are not monogenic analytical functions and thus are not elementary.

In modern point of view piecewise expressions like $|x|$ define functions correctly. In our problem the values of the variable x belong to $\mathbb{R}$. Using standard definition of radical and arc-functions we can describe the θ_1 and θ_2 as elementary: we can take, for example,

$$\theta_1(x) = \frac{\sqrt{\sin^2 \frac{\pi x}{l}}}{\sin \frac{\pi x}{l}}, \quad \theta_2 = \frac{1}{2} + \frac{1}{\pi} \arcsin \sin \frac{\pi(2x-l)}{2l}.$$

Thus the problem (1) have the solution in elementary function if ϕ is elementary function.

2. An example

Let's take an example, actually. Let l be equal to 1 and φ be equal to $x^2(1-x)$. Then

$$\hat{\varphi}(x) = \frac{\sqrt{\sin^2 \pi x}}{\sin \pi x} \left(\frac{1}{2} + \frac{1}{\pi} \arcsin \sin \frac{\pi(2x-1)}{2} \right)^2 \left(\frac{1}{2} - \frac{1}{\pi} \arcsin \sin \frac{\pi(2x-1)}{2} \right)$$

and the solution can be calculated by Eq. (2). The proof that the function constructed by such a formula is a classical solution of the oscillation equation is given in [2]. It should be noted that there are several representations for the functions θ_1 and θ_2, using elementary functions or `abs` and `floor`, having in any CAS [3]. It is easy to implement such kind of solutions in modern computer algebra systems.

Until that happens, users try to write the solution in infinite Fourier series, which is very popular in textbooks. For the example it is the series

$$u = \sum_{n=1}^{\infty} \frac{8 \cdot (-1)^{n+1} - 4}{\pi^3 n^3} \sin(\pi n x) \cos(\pi n c t). \tag{3}$$

Maple'2019 is able to convert the infinite series in symbolic expression, namely

$$\begin{aligned} u = {} & \frac{2i}{\pi^3} \left(\mathrm{Li}_3(-e^{i\cdot\pi(x+ct)}) - \mathrm{Li}_3(-e^{-i\cdot\pi(x+ct)}) + \mathrm{Li}_3(-e^{i\cdot\pi(x-ct)}) - \mathrm{Li}_3(-e^{-i\cdot\pi(x-ct)}) \right) \\ & - \frac{i}{\pi^3} \left(\mathrm{Li}_3(e^{-i\cdot\pi(x+ct)}) - \mathrm{Li}_3(e^{i\cdot\pi(x+ct)}) + \mathrm{Li}_3(e^{-i\cdot\pi(x-ct)}) - \mathrm{Li}_3(e^{i\cdot\pi(x-ct)}) \right). \end{aligned} \tag{4}$$

Here $\mathrm{Li}_3(z)$ is Euler's polylogarithm [4]. This representation for the solution is certainly non-elementary.

4. Conclusion

First at all the problem (1) is interesting for computer algebra as the example of the problem which have the elementary solution outside of Liouvillian theory. In practice it seems that its implementation meets some difficulties due to computer algebra systems work over $\mathbb{C}$ by default.

References

1. *Laptev B.L., Markushevich A.I.* Mathematics of the XIX century. Geometry. Theory of analytic functions. Moscow. Nauka. 1981. (In Russian)
2. *Dolya P.G.* Solution to the homogeneous boundary value problems of free vibrations of a finite string. Zh. Mat. Fiz. Anal. Geom. 2008. Vol. 4(2). P. 237–251.

3. *Malykh M.D. et al.* Computer methods of mathematical physics. Moscow. RUDN. 2020. (In Russian)

4. *Andrews G., Askey R., Roy R.* Special functions. Cambridge University Press. (2000). (Translation from English into Russian, edited by Yu.A. Neretin, Moscow, MCCME, 2013)

On a Machine-Checked Proof for an Optimized Method to Multiply Polynomials

S.D. Meshveliani[1]

[1]*A. K. Ailamazyan Program systems institute of RAS, Russia*

e-mail: mechvel@botik.ru

Abstract. The Karatsuba method to multiply univariate polynomials is simple to program and has essentially smaller run-time cost order than the simplest method of "multiply each monomial by each and sum". Here it is shortly described the design for a provable program in the Agda language for this Karatsuba method. In this program, computing the polynomial product is expressed in the same function together with the machine-checked proof for that the method is equivalent to the simplest multiplication. This is a part of the library DoCon-A of provable programs for algebra developed by the author.

Keywords: computer algebra, Karatsuba method for polynomials, machine-checked proofs, Agda language

1. Introduction

Below the word "library" means our library `DoCon-A` [1] of provable programs for computer algebra written in the `Agda` language [2]. The words "mc-proof", "witness" mean a machine-checked proof.

The simplest method to multiply polynomials f, g is to multiply each monomial in f by each monomial in g and sum the products. This costs $O(n^2)$ operations on coefficients, where n is the bound for the operand degrees. In 1960 A. A. Karatsuba demonstrated his fast method to multiply integer numbers [3]. A similar method for multiplying univariate polynomials costs $O(n^{log_2 3})$ operations on coefficients ([4], paragraph 8.1). Later it has been invented even a faster method based on the fast Fourier transformation. Here we deal with the Karatsuba method for polynomials, because it is simpler and more usable on practice, and other methods can be considered later.

Our library deals with formal proofs for mathematical objects, in particular, for polynomial arithmetic. A polynomial is represented *sparsely*: as a list of monomials having nonzero coefficients, and this list is ordered decreasingly by the monomial exponents. Adding polynomials is defined as a well-known method of merging the two ordered monomial lists together with summing the monomials having the same exponent, and with deleting the appearing zero monomials. As a definition of polynomial multiplication it is taken the following simplest method: $f * g$ is the sum of the products $m * g$ for all monomials m in f, and $m * g$ is formed as the list of nonzero products of m by the monomials in g.

For these representation and functions, the library proves many lemmata, and also proves that the domain of polynomials over any `DecCommutativeRing` (a commutative ring with a decidable equality) satisfies the laws of `DecCommutativeRing`.

The goal of this part of investigation was to implement the Karatsuba method for polynomials over any `DecCommutativeRing` in `Agda` as such a program that

* has a performance corresponding closely to the estimation of $O(n^{log_2 3})$, and is essentially faster on practice than the simplest method,

* includes an mc-proof of the method correctness, that is that the method returns the product polynomial equal to the one returned by the simplest method,

* is expressed as a source code of an admissible volume,
* is compiled in an admissible time.

The commutative ring laws for polynomials are easier to prove for the case of the simplest multiplication, and this has been done first. Then, from the proved correctness statement (the equivalence of the two multiplication functions) and from the proofs for the simplest multiplication it is easy to compose mc-proofs for the same laws for the polynomial arithmetic with the optimized multiplication: associativity, commutativity, distributivity, and such.

The program availability: the library archive referred by [1] has the file `Pol/Karatsuba.agda` for the Karatsuba method, `Pol/KTest.agda` for its demonstration and test. Installation and running is described in `install.txt`.

On other works on the subject:
currently we discovered an impressive work [5] which includes the proof design in the `Coq` system for the Karastuba method for polynomials (Section 1.4.1). We shall have to compare the two approaches.

2. On polynomial arithmetic in a provable program

A monomial (type `Mon`) is represented as a record containing the coefficient — an element of the carrier (type `C`) of a coefficient ring `R`, and the exponent (degree) of type $\mathbb{N}$.

A polynomial is represented as a record containing a) a list mons of monomials, b) a witness for the coefficients in mons being nonzero, c) a witness for that the exponents in mons are ordered decreasingly. For example, the polynomial $-2*x^4+1$ over integers is represented as

```
Pol ((Mon -2 4) :: (Mon 1 1) :: []) nzCoefs ordExps,
```

where nzCoefs is a witness for -2, 1 being nonzero in $\mathbb{Z}$,
ordExps a witness for decreasing orderedness of the list $4 :: 1 :: []$.

The polynomial addition and multiplication are defined earlier. The cost of addition is estimated as $O(n)$, where n is a bound for the operand degrees. The cost of multiplication is estimated as $O(n^2)$, and its degree cannot be smaller for the simplest multiplication.

Our library includes explicitly these definitions, and proves the structure of `DecCommutativeRing` for `Pol R` for any `DecCommutativeRing R`. This is expressed as the program modules `Pol/Over-decComMonoid, Pol/Over-abelianGroup, Pol/Over-decComRing`.

3. The Karatsuba method to multiply polynomials

To start with, let the polynomials f and g be represented in the dense form — with allowing zero monomials. And let $\deg f = \deg g = 2k$, where k is a power of 2. Represent

$$f = x^k f_1 + f_2, \quad g = x^k g_1 + g_2,$$

where $\deg f_1 = \deg f_2 = \deg g_1 = \deg g_2 = k$. Then

$$f * g = x^{2k} f_1 g_1 + x^k * (f_1 g_2 + f_2 g_1) + f_2 g_2$$

This formula has four multiplications for the polynomials of degree k. It can be rewritten equivalently as

$$f * g = x^{2k} f_1 g_1 + x^k * (ss' - f_1 g_1 - f_2 g_2) + f_2 g_2, \tag{1}$$

where $s = f_1 + f_2,\ \ s' = g_1 + g_2$. This has only three multiplications for the degree k; each of the products $f_1g_1,\ f_2g_2,\ ss'$ to be evaluated once and then to substitute to the expression (1) to the needed places, and they are evaluated by applying recursively the formula (1).

The general case is reduced to this case of k being a power of two by adding a zero monomial of degree e to both operands, where e is the least power of 2 which is not less than $\deg f$ and $\deg g$.

In [4] Theorem 8.3, it is proved the estimation $O(n^{log_2 3})$ for this method.

In our program **a)** polynomials are represented sparsely (for example, f_2 can have a smaller degree than k), **b)** instead of a single adding of a zero monomial it is applied the following algorithm.

Method Kara.

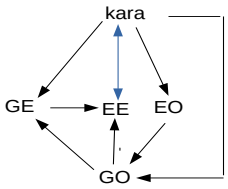
Denote $e = \deg f,\ e' = \deg g,$ and order the operands to satisfy the inequality $e \le e'$.

Then it is applied the function kara. It takes nonzero polynomials f, g, such that $e = \deg f \le \deg g = e',$ and returns $f * g$. It considers the cases of

(EE) $e = e'$ is even, (GE) $e < e',\ e'$ is even,

(GO) $e < e',\ e'$ is odd, (EO) $e = e'$ is odd.

Each of these cases is implemented as the corresponding function in the program. All these functions call each other as shown on the following diagram:



In the case EE it is first applied the function splitAtDegree. It splits f to the higher part higher which monomials have degrees not less than $k = e/2$, and the lower part f_2, which monomials have degrees less than k. higher is represented as $x^k * f_1,$ and $f = x^k * f_1 + f_2$. The same splitting is applied to $g:\ \ g = x^k * g_1 + g_2$. This splitting spends $O(e)$ operations. Then it is applied the formula (1), where the products $f_1g_1,\ f_2g_2,\ ss'$ are evaluated by recursively applying the function kara, with first ordering the operands according to their degrees (due to the sparse representation, the degrees of f_2, g_2 may differ).

In the case GE put $F = x^{e'} + f$ and evaluate $f * g = (F - x^{e'}) * g = \ \mathrm{EE}(F, g) - x^{e'}g,$ where $x^{e'}g$ takes not more than e' operations on monomials.

In the case GO $p = e' - 1$ is even. Put $g = m + g',$ where m is the leading monomial in g. And put $f * g = m * f + h,$ where

$h =$ `case` $(\deg g' =?\ p,\ \deg f =?\ p)$ `of` `(True, True)` -> $\mathrm{EE}(f, g')$
`(True, False)` -> $\mathrm{GE}(f, g')$
`(False, True)` -> $\mathrm{GE}(g', f)$
`(False, False)` -> $\mathrm{GE}(f, x^p + g') - x^p f$

(in the last pattern, $\deg f, \deg g' < p$).

In the case EO represent $f = m + f'$ where m is the leading monomial in f, and evaluate

$$f * g = \ m * g + \mathrm{GO}(f', g)$$

For deriving the cost estimation for this method with sparse polynomials we use the following inequality for a polynomial:

the number of monomials in $f\ \le\ 1 + (\deg f)$.

About mc-proofs: in the corresponding `Agda` function, it occurs practically impossible to separate the proper result evaluation from the needed proof. So that the proper evaluation is intermingled recursively with the proof within the same function. And this does not reduce the run-time performance any essentially. This effect with performance can always be reached due to the "lazy" evaluation by default in `Agda`, and due to certain other reasons.

It is also important that this function includes the termination proof: this requires proving certain inequalities with degrees related to the arithmetic operations with polynomials.

Proofs cost: one of the goals of the investigation was to find of what are the expenses caused by adding the mc-proofs. In the example of the Karatsuba method and the systems `Glasgow Haskell 8.8.3` vs `Agda 2.6.1, MAlonzo` the result is as follows.

* The run-time performance is 2 times lower.

* The source program volume is increased 8 times.

* The executable program is "made" 100 times slower, but it still takes a reasonable time of several minutes.

The weakest point is the effort needed for designing mc-proofs, it needs to be reduced. This can be done by developing a library of various provers for each particular subject domain.

Performance demonstration: It is implemented in the module `Pol/KTest` as powering the polynomial $x + 1$ into the degree $n = 2^k$ by the binary method, so that polynomial multiplication is applied k times. To get rid of the effect of growing size of coefficients, the coefficient domain is set $\mathbb{Z}$`/(b)` (integers modulo `b`), `b = 99991`. The timing is compared for various n and with switching between the simplest multiplication `*p` and the `karatsuba` function. The table of this test shows the cost orders about respectively n^2 and $n^{1.63}$.

The corresponding program for both methods has also been written in the `Haskell` language (of course, without proofs) and run under the Glasgow Haskell system `ghc-8.8.3`.

Acknowledgement: this investigation was partially supported by Russian Academy of Sciences, research project No AAAA-A19-119020690043-9.

References

1. *Meshveliani S. D.* `DoCon-A`, version 3.2rc. A provable algebraic domain constructor. A source program and a short manual. Pereslavl-Zalessky. April 2021. `http://www.botik.ru/pub/local/Mechveliani/docon-A/3.2rc/`

2. *Norell U.* Dependently typed programming in Agda. AFP 2008: Advanced Functional Programming, Lecture Notes in Computer Science. 2008. Vol. 5832. P. 230–266. Springer, Berlin–Heidelberg.

3. *Karatsuba A. A., Ofman Yu. P.* Multiplication of multidigit numbers on automata. Cybernetics and control theory. 1963. Vol. 7, No. 7. `https://www.researchgate.net/publication/234346907_Multiplication_of_Multidigit_Numbers_on_Automata`

4. *Gathen J., Gerhard J.* Modern computer algebra. Third edition. Cambridge University Press, 2013.

5. *Mörtberg A.* Formalizing refinements and constructive algebra in type theory. PhD thesis. 2014. `https://www-sop.inria.fr/members/Anders.Mortberg/thesis/doc/v0.1/karatsuba.html`

Symbolic Computation in Studying the Stability of Periodic Motion of the Swinging Atwood Machine

Alexander N. Prokopenya[1]

[1] *Warsaw University of Life Sciences – SGGW, Poland*

e-mail: alexander_prokopenya@sggw.edu.pl

Abstract. The swinging Atwood machine under consideration is a conservative Hamiltonian system with two degrees of freedom. Although its equations of motion are essentially nonlinear one can construct a periodic solution in the form of power series in a small parameter if the masses difference is sufficiently small. We have investigated perturbations of periodic solution and proved its stability in linear approximation. All tedious symbolic computations are performed with the aid of the computer algebra system Wolfram Mathematica.

Keywords: symbolic computation, swinging Atwood machine, periodic solution, stability, characteristic exponents

The Atwood machine is a well-known device that was designed to demonstrate the uniformly accelerated motion of a system (see [1]). The swinging Atwood machine (SAM) to be studied consists of two small massless pulleys and two bodies of masses $m_1 \leq m_2$ attached to opposite ends of a massless inextensible thread (see Fig. 1). The body m_1 is allowed to swing in vertical plane and it behaves like a pendulum of variable length while the body m_2 is constrained to move only along a vertical. It is a conservative Hamiltonian system with two degrees of freedom but its equations of motion are nonlinear and their general solution cannot be written in symbolic form. Numerical analysis of the equations of motion has shown that, depending on the mass ratio and initial conditions, the SAM can demonstrate different types of motion (see, for example, [2-4]). In particular, the system has a dynamic equilibrium state described by a periodic solution of the equations of motion that may be constructed in the form of a power series in a small parameter (see [5]). The main purpose of this paper is to investigate stability of this periodic solution. All the relevant symbolic computations are performed with the computer algebra system Wolfram Mathematica (see [6]).

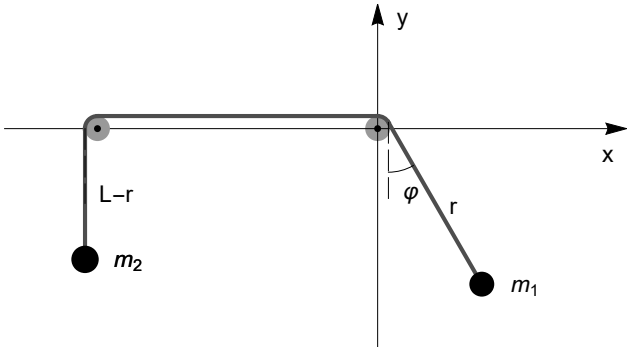


Figure 1. The SAM with two small pulleys

To simplify the calculations it is expedient to introduce dimensionless variables. As we expect the body m_1 in the state of dynamic equilibrium behaves like a pendulum of a length R_0, the distance r can be made dimensionless by using R_0 as a characteristic distance, whereas the time t can be made dimensionless by using the inverse of the pendulum's natural

frequency $\sqrt{g/R_0}$. Then we can write the Lagrangian of the system in the form

$$\mathcal{L} = \frac{2+\varepsilon}{2}\dot{r}^2 + \frac{1}{2}r^2\dot{\varphi}^2 - (1+\varepsilon)r + r\cos\varphi, \tag{1}$$

where the dot over a symbol denotes the total derivative of the corresponding function with respect to time, and parameter $\varepsilon = (m_2 - m_1)/m_1$ represents the ratio of the masses difference to the mass m_1. Note that the Lagrangian (1) depends on a single dimensionless parameter ε which we shall assume to be small ($0 \leq \varepsilon \ll 1$). Using (1) and doing standard symbolic computation, we obtain the equations of motion in the form

$$(2+\varepsilon)\ddot{r} = -\varepsilon - (1-\cos\varphi) + r\dot{\varphi}^2,$$

$$r\ddot{\varphi} = -\sin\varphi - 2\dot{r}\dot{\varphi}. \tag{2}$$

One can easily check that the system (2) has an equilibrium solution $r = const$, $\varphi = 0$ only in the case of equal masses ($\varepsilon = 0$). This equilibrium state is unstable, and the system leaves it as soon as the mass m_1 gets even very small initial velocity (see [4]). From the other side, in case of different masses the constant term $\varepsilon > 0$ in the right-hand side of the first equation (2) causes the uniformly accelerated motion of the Atwood machine in the absence of oscillations as it is in the classical Atwood machine (see [1]). However, if the masses difference is sufficiently small one can expect that an averaged value of the oscillating functions in the right-hand side of the first equation (2) compensates the constant ε, and the smaller oscillating mass m_1 can balance the larger mass m_2. Indeed, such a state of dynamical equilibrium of the system exists and it is described by the periodic solution of the system (2) given by (see [5])

$$r_p(t) = 1 + \frac{\varepsilon}{16}(1 - 6\cos 2t) - \frac{3\varepsilon^2}{2048}(87 - 92\cos 2t + 35\cos 4t) +$$

$$+\frac{\varepsilon^3}{131072}(4275 - 8166\cos 2t + 5067\cos 4t - 1510\cos 6t) + ..., \tag{3}$$

$$\varphi_p(t) = \sqrt{\varepsilon}\left(2\sin t + \frac{53\varepsilon}{192}\sin 3t + \frac{\varepsilon^2}{81920}(14795\sin t - 8495\sin 3t + \frac{5813}{5}\sin 5t) + ...\right). \tag{4}$$

The existence of periodic solution (3)-(4) means that for given value of parameter ε one can choose such initial conditions that the system is in the state of dynamical equilibrium when the bodies oscillate near some equilibrium positions. Note that for $\varepsilon > 0$ the system under consideration has no a static equilibrium state when the coordinates $r(t)$, $\varphi(t)$ are some constants. So it is natural to investigate whether the system will remain in the neighborhood of the equilibrium if the initial conditions are perturbed or whether the periodic solution (3)-(4) is stable.

First of all, using (1) and doing the Legendre transformation, we define the Hamiltonian of the system

$$\mathcal{H} = \frac{p_r^2}{2(2+\varepsilon)} + \frac{p_\varphi^2}{2r^2} + (1+\varepsilon)r - r\cos\varphi, \tag{5}$$

where p_r, p_φ are the conjugate momenta to r, φ, respectively.

Then we define new canonical variables q_1, q_2, p_1, p_2 according to the rules

$$r \to r_p + q_1,\ \varphi \to \varphi_p + q_2,\ p_r \to p_{r0} + p_1,\ p_\varphi \to p_{\varphi 0} + p_2, \tag{6}$$

where the unperturbed momenta $p_{r0} = (2+\varepsilon)\dot{r}_p, p_{\varphi 0} = r_p^2\dot{\varphi}_p$ are obtained from (5) in a standard way. Doing the canonical transformation (6) and expanding the Hamiltonian (5)

into power series in terms of q_1, q_2, p_1, p_2 up to second order inclusive, we represent it in the form

$$\mathcal{H}_2 = \frac{p_1^2}{2(2+\varepsilon)} + \frac{3p_{\varphi 0}^2}{2r_p^4}q_1^2 + \frac{p_2^2}{2r_p^2} + \frac{r_p}{2}\cos\varphi_p q_2^2 - -\frac{2p_{\varphi 0}}{r_p^3}q_1 p_2 + q_1 q_2 \sin\varphi_p. \tag{7}$$

Note that the quadratic term $\mathcal{H}_2$ is the first non-zero term in the Hamiltonian (5) expansion.

The quadratic part (7) of the Hamiltonian determines the linearized equations of the perturbed motion which is convenient to write in the matrix form

$$\dot{x} = J \cdot H(t,\varepsilon)x, \tag{8}$$

where $x^T = (q_1, q_2, p_1, p_2)$ is a 4-dimensional vector, $J = \begin{pmatrix} 0 & E_2 \\ -E_2 & 0 \end{pmatrix}$, E_2 is the second-order identity matrix, and $H(t,\varepsilon)$ is the fourth-order matrix-function the elements of which are obtained by differentiation of $\mathcal{H}_2$:

$$H_{i,j} = \frac{\partial^2 \mathcal{H}_2}{\partial x_i \partial x_j}, \quad i,j = 1,2,3,4. \tag{9}$$

It is clear that matrix $H(t,\varepsilon)$ is periodic function of time, and so the perturbed motion of the system is described by the linear system (8) of four differential equations with periodic coefficients.

The systems of linear differential equations with periodic coefficients and their general properties have been studied quite well (see [7]). The behavior of solutions to system (8) is determined by its characteristic multipliers which are the eigenvalues of the monodromy matrix $X(2\pi,\varepsilon)$, where $X(t,\varepsilon)$ is a fundamental matrix for system (8) satisfying the initial condition $X(0,\varepsilon) = E_4$. As periodic solution (3)-(4) is represented by power series in ε, the matrix $H(t,\varepsilon)$ can be also represented in the form of power series

$$H(t,\varepsilon) = H_0(t) + \sqrt{\varepsilon}H_1(t) + \varepsilon H_2(t) + \varepsilon^{3/2}H_3(t) + ..., \tag{10}$$

where $H_k(t), k = 0,1,2,...$, are continuous periodic fourth-order square matrices which are obtained by substitution of solution (3)-(4) into (7), (9) and expanding each element of the matrix $H(t,\varepsilon)$ into power series in ε.

The fundamental matrix $X(t,\varepsilon)$ can be sought in the form of power series

$$X(t,\varepsilon) = X_0(t) + \sqrt{\varepsilon}X_1(t) + \varepsilon X_2(t) + \varepsilon^{3/2}X_3(t) + ..., \tag{11}$$

where $X_k(t), k = 0,1,2,...$, are continuous matrix functions. On substituting (10)-(11) into (8) and collecting coefficients of equal powers of ε, we obtain the following sequence of differential equations:

$$\dot{X}_0 = JH_0X_0(t), \tag{12}$$

$$\dot{X}_k - JH_0X_k = \sum_{j=1}^{k} JH_j(t)X_{k-j}(t), \quad (k \geq 1). \tag{13}$$

The functions $X_k(t)$ must satisfy the following initial conditions:

$$X_0(0) = E_4, \quad X_k(0) = 0 \quad (k \geq 1). \tag{14}$$

Equations (12), (13) are solved in succession but the corresponding computations are very bulky and we do not show them here. Note only that all the calculations are performed with the computer algebra system Wolfram Mathematica. Finally, the monodromy matrix

$X(2\pi,\varepsilon)$ has been computed up to the second order in ε, and the corresponding characteristic equation determining the characteristic multipliers for the system (8) is given by

$$\det(X(2\pi,\varepsilon)-\rho E_4)=(\rho-1)^2(\rho^2+B\rho+1)=0, \tag{15}$$

where

$$B=-2+3\pi^2\varepsilon-\frac{3\pi^2}{16}(17+4\pi^2)\varepsilon^2+\frac{3\pi^2}{5120}(4845+2720\pi^2+128\pi^4)\varepsilon^3.$$

Solving (15), we obtain

$$\rho_{1,2}=1,\ \rho_{3,4}=1\pm i\pi\sqrt{3\varepsilon}-\frac{3\pi^2}{2}\varepsilon\mp i\frac{\pi\sqrt{3}}{32}(17+16\pi^2)\varepsilon^{3/2}. \tag{16}$$

Note that two characteristic multipliers $\rho_{1,2}=1$ determine two independent 2π-periodic solutions to system (8). One can readily check that the absolute value of the second couple of the characteristic multipliers $\rho_{3,4}$ is equal to 1. They are complex conjugate and determine two purely imaginary characteristic exponents

$$\lambda_{3,4}=\frac{1}{2\pi}\log\rho=\pm i\frac{\sqrt{3\varepsilon}}{2}\left(1-\frac{17}{32}\varepsilon+\frac{85}{256}\varepsilon^2\right). \tag{17}$$

According to Floquet-Lyapunov theory (see [7]), four linearly independent solutions to system (8) with 2π-periodic matrix may be represented in the form

$$x_1(t)=f_1(t),\ x_2(t)=f_2(t),\ x_3(t)=\exp(\lambda_3 t)f_3(t), x_4(t)=\exp(\lambda_4 t)f_4(t), \tag{18}$$

where $f_k(t),(k=1,2,3,4)$ are 2π-periodic vector-functions. Therefore, in case of $\varepsilon>0$ solutions (18) describe the perturbed motion of the system in the bounded domain in the neighborhood of the periodic solution (3)-(4). It means this solution is stable in linear approximation, and so the SAM is an example of mechanical system in which the equilibrium state is stabilized by oscillations.

References

1. *Atwood G.* A treatisa on the rectilinear motion and rotation of bodies. Cambridge University Press, 1784.
2. *Tufillaro N.B., Abbott T.A., Griffiths D.J.* Swinging Atwood's machine. American Journal of Physics. 1984. Vol. 52, No. 10. P. 895–903.
3. *Nunes A., Casasayas J., Tufillaro N.B.* Periodic orbits of the integrable swinging Atwood's machine. American Journal of Physics. 1995. Vol. 63, No. 2. P. 121–126.
4. *Prokopenya A.N.* Motion of a swinging Atwood's machine: simulation and analysis with Mathematica. Mathematics in Computer Science. 2017. Vol. 11, No. 3. P. 417–425.
5. *Prokopenya A.N.* Construction of a periodic solution to the equations of motion of generalized Atwood's machine using computer algebra. Programming and Computer Software. 2020. Vol. 46, No. 2. P. 120–125.
6. *Wolfram S.* An elementary introduction to the Wolfram Language. 2nd ed. Champaign, IL, USA, Wolfram Media, 2017.
7. *Yakubovich V.A., Starzhinskii V.M.* Linear differential qquations with periodic coefficients. John Wiley, New York, 1975.

Quadrature Formula for the Double Layer Potential

I.O. Reznichenko[1,2], P.A. Krutitskii[2]
[1]*Moscow State University, Russia*
[2]*Keldysh Institute of Applied Mathematics, Russia*
e-mail: io.reznichenko@physics.msu.ru, krutitsk@mail.ru

Abstract. In this work, a quadrature formula for the double layer potential is derived in the case of the Helmholtz equation with continuous density given on a smooth closed or open surface. This quadrature formula gives higher computational accuracy than standard quadrature formula, which is confirmed by numerical tests. The advantage of the new quadrature formula is especially noticeable near the surface, where the standard quadrature formula diverges rapidly, while the new formula provides acceptable accuracy for points that are distant from the surface at distances comparable to the integration step and more.

Keywords: computer algebra, implementation, applied aspects

1. Introduction

The double layer potential is used to solve boundary value problems for the Helmholtz equation by the method of integral equations. Such problems arise in various fields of mathematical physics, for example, in the theory of scattering of acoustic and electromagnetic waves by obstacles, in geophysical hydrodynamics, when studying the diffraction of tidal waves on islands, in the theory of heat waves in thermodynamics, etc.

Numerical solution of boundary value problems using the double layer potential discussed in [1,2,3] and consists of two steps. At the first stage, by numerically solving the boundary integral equation, one finds the potential density. At the second stage, by substituting the numerical value density into a quadrature formula, one finds a solution to the boundary value problem at any point in the region. However, quadrature formulas for potentials used in engineering calculations [4] do not give a uniform approximation of the potential in the region and do not preserve the continuity property of the potentials up to the region boundary. Moreover, near certain points on the boundary of the region, the quadrature formulas diverge and tend to infinity, although the potentials themselves are limited near the boundary.

When using standard quadrature formulas, to improve the accuracy, you either have to decrease the step or carry out additional constructions near the border of the region, which increases the cost of computations. Therefore, there exists a problem of obtaining improved quadrature formulas that provide increased accuracy near the border.

In the two-dimensional case, the improved quadrature formula for the potential of a simple layer with density, given on open curves and having power singularities at the ends of the curves, is constructed in [5,6]. This formula can be applied when finding numerical solutions to boundary value problems for the Helmholtz equation outside cuts and open curves on a plane using the method potentials and boundary integral equations. Such problems were studied by the indicated method in [7,8,9,10,11]. In [12], an improved quadrature formula was proposed for the potential of a simple layer in the three-dimensional case, providing uniform approximation and uniform convergence in the domain. Moreover, this formula preserves the property of continuity of the potential of a simple layer when crossing the boundary of the region.

2. Problem statement

Let us introduce in space the Cartesian coordinate system $x = (x_1, x_2, x_3) \in R^3$. Let Γ be a simple smooth closed surface of class C^2, or a simple smooth bounded open oriented surface of class C^2, containing its limit points [13, Chapter 14, § 1]. If the surface Γ is closed, then it must bound the volumetric simply connected inner region [14, pp. 201]. Suppose the surface Γ is parametrized so that a rectangle is mapped onto it:

$$y = (y_1, y_2, y_3) \in \Gamma, \quad y_1 = y_1(u, v),\ y_2 = y_2(u, v),\ y_3 = y_3(u, v); \quad u \in [0, A],\ \ v \in [0, B];$$

$$y_j(u, v) \in C^2([0, A] \times [0, B]), \quad j = 1, 2, 3. \tag{1}$$

Sphere, ellipsoid surface, smooth surfaces of figures of revolution, torus surface and many other more complex surfaces can be parametrized in this way. Introduce N points u_n with step h on the segment $[0, A]$ and M points v_m on the segment $[0, B]$ and consider a partition of the rectangle $[0, A] \times [0, B]$, which is mapped to the surface Γ

$$A = Nh, \quad B = MH, \quad u_n = (n + 1/2)h, \quad n = 0, ..., N - 1;$$

$$v_m = (m + 1/2)H, \quad m = 0, ..., M - 1.$$

We demand the following condition so that $|\eta(y(u, v))| \in C^1((0, A) \times (0, B))$.

$$|\eta(y(u, v))| > 0, \quad \forall\ (u, v) \in ((0, A) \times (0, B)). \tag{2}$$

The double layer potential for the Helmholtz equation is used to solve boundary value problems by the method of integral equations. Let $x \notin \Gamma$. By $\mathbf{n}_y$ we denote the unit normal at the point $y \in \Gamma$, that is, $\mathbf{n}_y = \eta(y)/|\eta(y)|$. Consider the double layer potential for the Helmholtz equation with a given on the surface Γ with density $\mu(y) \in C^0(\Gamma)$

$$\mathcal{W}_k[\mu](x) = \frac{1}{4\pi} \int_\Gamma \mu(y) \frac{\partial}{\partial \mathbf{n}_y} \frac{e^{ik|x-y|}}{|x - y|} ds_y \approx \frac{1}{4\pi} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \mu_{nm} \exp(ik|x - y(u_n, v_m)|) \times$$

$$\times (ik|x - y(u_n, v_m)| - 1) \int_{u_n - h/2}^{u_n + h/2} du \int_{v_m - H/2}^{v_m + H/2} dv \sum_{j=1}^{3} \frac{\eta_j(y(u, v))(y_j(u, v) - x_j)}{|x - y(u, v)|^3}, \tag{3}$$

where $\mu_{nm} = \mu(y(u_n, v_m))$ and $k \geq 0$. It can be shown that the constants in the estimates of the functions: $\mu(y(u, v))$, $|x - y(u, v)|$ and $\exp(ik|x - y(u, v)|)$ are of order $O(h + H)$ and do not depend on n, m and from the location of $x \notin \Gamma$. Thus, to obtain a quadrature formula for the double layer potential at $x \notin \Gamma$, it is necessary to calculate the integral in the (3), which we will call the canonical integral $K_{nm}(x)$.

Expanding expressions with $x - y_j$ in the numerator and denominator in a Taylor series, we arrive at a double integral. The number of terms in the Taylor series was chosen so that the canonical integral could be found in elementary functions without going over to the elliptic case. After introducing the appropriate substitutions, we come to the integral

$$J_{12} = \int_{-h/2}^{h/2} dU \frac{S_1 U + S_0}{(C_2 U^2 + C_1 U + C_0)\sqrt{B_2 U^2 + B_1 U + B_0}}, \tag{4}$$

the calculation of which depends on the sign of the discriminant of the polynomial $C_2U^2 + C_1U + C_0$ in the denominator. One of the cases involves integration in the complex plane. It can also be shown that in the case when the root of one of the polynomials in the denominator

falls into the integration limit, the contribution to the sum from this piece of the surface can be equated to zero. Let us formulate the main result in the form of a theorem.

Theorem 1. *Let Γ — simple smooth closed surface of class C^2, bounding a simply connected volumetric internal region, or a simple smooth bounded open oriented surface class C^2 containing its limit points. Let Γ admit parametrization (1) with property (2), and $\mu(y) \in C^0(\Gamma)$. Then for the double layer potential for $x \notin \Gamma$ and $k \geq 0$ the quadrature formula holds*

$$\mathcal{W}_k[\mu](x) \approx \frac{1}{4\pi} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \mu_{nm} \exp(ik|x - y(u_n, v_m)|)(ik|x - y(u_n, v_m)| - 1)K_{nm}(x), \qquad (5)$$

where the integral $K_{nm}(x)$ is calculated explicitly.

3. Numerical tests

Testing the improved and standard quadrature formulas was carried out in the case when the surface Γ is a sphere of unit radius. Here we showcase one of the tests, where we used the potential density $\mu(y(u, v)) = k$, and the double layer potential for the Helmholtz equation has the form

$$\mathcal{W}_k[\mu](x) = \begin{cases} (1 - ik) \exp(ik) \dfrac{\sin(k|x|)}{|x|} & \text{when } |x| < 1, \\ (\sin k - k \cos k) \dfrac{\exp(ik|x|)}{|x|} & \text{when } |x| > 1, \end{cases} \qquad (6)$$

where $k = 1$. In the Table 1 the calculated maximum values of the absolute errors are given.

Table 1. Maximum absolute error of quadrature formulas in the test

Inner spheres

ΔR	$M = N = 25$	$M = N = 50$	$M = N = 100$
-0.1	0.029; 0.0089	0.0079; 0.0024	0.0021; 0.00062
-0.06	0.13; 0.012	0.020; 0.0038	0.0055; 0.0010
-0.03	1.04; 0.034	0.13; 0.0065	0.020; 0.0019
-0.01	12.1; 0.12	2.74; 0.031	0.49; 0.0061

External spheres

ΔR	$M = N = 25$	$M = N = 50$	$M = N = 100$
0.1	0.028; 0.0077	0.0077; 0.0021	0.0020; 0.00054
0.06	0.15; 0.014	0.020; 0.0035	0.0055; 0.00093
0.03	1.08; 0.042	0.14; 0.0069	0.020; 0.0018
0.01	12.1; 0.24	2.76; 0.036	0.50; 0.0063

The first number in the table cells is the error of the standard quadrature formula, and the second number is the error of the improved quadrature formula. It can be seen from the tables that the improved quadrature formula provides higher computational accuracy near the Γ boundary than the standard one. In addition, the standard formula diverges rapidly when approaching the boundary. Tests show that the improved formula gives good

computational accuracy for all points located at a distance H and more from the boundary Γ. In this case, the improved quadrature formula has the second order of convergence and provides the maximum computational error of the order of $O(hH)$. At distances of the order of hH to the boundary, the improved formula gives an error of $O(H)$, and the standard formula diverges.

References

1. *Belotserkovsky S.M., Lifanov I.K.* Numerical methods in singular integral equations. Moscow: Nauka, 1985. (In Russian)
2. *Lifanov I.K.* Method of singular integral equations and numerical experiment. M.: LLP Janus, 1995. (In Russian)
3. *Setukha A.V.* Numerical methods in integral equations and their applications. M.: Argamak-media, 2016. (In Russian)
4. *Brebbia C.A., Telles J.C.F., and Wrobel L.C.* Boundary element techniques. Berlin-Heidelberg-New York-Tokyo: Springer-Verlag, 1984.
5. *Krutitskii P.A., Kwak D.Y., Hyon Y.K.* Numerical treatment of a skew-derivative problem for the Laplace equation in the exterior of an open arc. Journal of Engineering Mathematics. 2007. Vol. 59. P. 25–60.
6. *Krutitsky P.A., Kolybasova V.V.* A numerical method for solving integral equations in a problem with an oblique derivative for the Laplace equation outside open curves. Differential Equations. 2016. Vol. 52, No. 9. P. 1262–1276. (In Russian)
7. *Krutitskii P.A.* Wave propagation in a 2-D external domain with cuts. Applicable Analysis. 1996. Vol. 62, No. 3–4. P. 297–309.
8. *Krutitskii P.A.* The Neumann problem for the 2-D Helmholtz equation in a multiply connected domain with cuts. Zeitschrift fur Analysis und ihre Anwendungen. 1997. Vol. 16, No. 2. P. 349–361.
9. *Krutitskii P.A.* The mixed problem for the Helmholtz equation in a multiply connected region. Comp. Maths. Math. Phys. 1996. Vol. 36, No. 8. P. 1087–1095.
10. *Krutitskii P.A.* The Dirichlet problem for the 2-D Helmholtz equation in a multiply connected domain with cuts. ZAMM. 1997. Vol. 77, No. 12. P. 883–890.
11. *Krutitskii P.A.* The Helmholtz equation in the exterior of slits in a plane with different impedance boundary conditions on opposite sides of the slits. Quarterly of applied mathematics. 2009. Vol. 67, No. 1. P. 73–92.
12. *Krutitsky P.A., Fedotova A.D., Kolybasova V.V.* Quadrature formula for the potential of a simple layer. Differential Equations. 2019. Vol. 55. P. 1226–1241.
13. *Butuzov V.F., Krutitskaya N.Ch., Medvedev G.N., Shishkin A.A.* Mathematical analysis in questions and problems. M .: Fizmatlit, 2000. (In Russian)
14. *Ilyin V.A., Poznyak E.G.* Fundamentals of mathematical analysis. Part 2. M .: Fizmatlit, 1973. (In Russian)

A Plain Note on Binary Solutions to Large Systems of Linear Equations

A.V. Seliverstov[1]

[1]*Institute for Information Transmission Problems of RAS (Kharkevich Institute), Russia*
e-mail: slvstv@iitp.ru

Abstract. A generic-case algorithm is proposed to recognize systems of linear equations without any binary solution, when the number of equations is close to the number of unknowns. This problem corresponds to a well-known optimization problem, i.e., the multidimensional knapsack problem. In 1994 Nikolai Kuzyurin discovered an average-case polynomial-time optimization algorithm. His proof is based on binomial tail bounds. Contrariwise, our algebraic approach allows to specify the structure of the set of inconvenient inputs. For any fixed dimension, this set is included in the set of zeros of an explicit nonzero multivariate polynomial.

Keywords: binary solution, linear equation, generic-case complexity

Introduction

Let us consider the decision problem whether there exists a binary solution (also known as a $(0,1)$-solution) to a system of inhomogeneous linear equations with integer coefficients. The problem is NP-complete and can be reduced to its particular case containing only one linear equation [1]. In some cases, the equation has small integer coefficients [2, 3]. Furthermore, a binary solution to one linear equation can be found using a pseudopolynomial-time algorithm [1, 4–7]. Without any restriction on the coefficients, Horowitz and Sahni [8] had introduced the meet-in-the-middle approach and gave an exact $O^*(2^{n/2})$ time and space algorithm. A few years later, Schroeppel and Shamir [9] improved the space complexity to $O^*(2^{n/4})$. Recently a probabilistic $O^*(2^{0.86n})$ time and polynomial-space algorithm was found [10]. The O^* notation suppresses a factor that is polynomial in the input size. There is also known a polynomial upper bound on the average-case complexity of the multidimensional knapsack problem [11].

By means of Gaussian elimination, searching for a binary solution to a system of m linearly independent linear equations in n unknowns is reduced to a parallel check whether a binary solution to a subsystem in $n-m$ unknowns can be extended to a binary solution to the whole system of equations in n unknowns. Hence, the initial problem is polynomial-time solvable when the difference between the number of unknowns and the number of linearly independent equations is bounded by a function of the type $n - m = O(\log n)$. Let us consider the case when the difference between the number of unknowns n and the number of equations m is bounded by a function of the type $n - m = O(\sqrt{n})$. So, the previously obtained estimate is improved, although the proposed method is generally useless for one equation.

An easy generalization of this problem is searching for binary solutions to a system of linear equations over an arbitrary field $(K, 0, 1, +, -, \times, ()^{-1}, =)$ of characteristic zero. Let us define $0^{-1} = 0$. In contrast to previous works [11], we consider not only ordered fields but also arbitrary fields of characteristic zero, including the field of complex numbers. Let us use either generalized register machines [12] or BSS-machines over reals [13]. These machines over an algebraic extension of the field of rational numbers naturally correspond to the idea

of symbolic computations. Every register contains an element of K. The machine also has index registers containing non-negative integers. The running time is polynomial when the total number of operations performed by the machine is bounded by a polynomial in the number of registers containing the input. Initially, this number is written in the zeroth index register.

A predicate holds almost everywhere when it holds on every instance x satisfying an inequality of the type $f(x) \neq 0$, where f denotes a nonzero polynomial [14]. This restriction is more rigorous than any upper bound on the measure. Let us consider so-called generic generalized register machines over K. The machine halts at every input and gives a meaningful answer at almost every input, but it can abandon the calculation using explicit notification, that is, there exists the vague halting state. More precisely, a generalized register machine over K is called generic when two conditions hold: (1) the machine halts at every input and (2) for every positive integer k and for almost all inputs, each of which occupies exactly k registers, the machine accepts or rejects the input, but does not halt in the vague state. Generic machines that compute non-trivial output in registers are defined similarly. If the machine halts in the vague state, then the output recorded in the registers is considered meaningless. Note that the machine does not make any error. For detailed description of generic-case computation on classical computational models refer to [15–16].

Results

Let us consider systems of linear equations of the type $x_j = \ell_j(1, x_1, \cdots, x_{n-m})$, where $j > n - m$ and every $\ell_j(x_0, x_1, \cdots, x_{n-m})$ denotes a linear form over K.

Theorem. *There exists a polynomial-time generic generalized register machine over K such that for all positive integers n and m satisfying the inequality $2n \geq (n-m+1)(n-m+2)$, and for almost every m-tuple of linear forms $\ell_j(x_0, \cdots, x_{n-m})$, where $j > n - m$, if the machine accepts the input, then there exists no binary solution to the system of all equations of the type $x_j = \ell_j(1, x_1, \cdots, x_{n-m})$. Moreover, for every n, there exists a polynomial of degree at most $2n$ in coefficients of all the linear forms ℓ_j such that if the machine halts in the vague state, then the polynomial vanishes.*

Proof. If $2n < (n-m+1)(n-m+2)$, then the machine rejects the input. Else, in accordance with Theorem 1, some polynomial time generic machine calculates numbers $\lambda_1, \ldots, \lambda_n$ such that the equality

$$\sum_{k=1}^{n-m} \lambda_k x_k (x_k - x_0) + \sum_{j=n-m+1}^{n} \lambda_j \ell_j (\ell_j - x_0) = x_0^2$$

holds. On the other hand, if there exists a binary solution to the system of all the equations $x_j = \ell_j(1, x_1, \cdots, x_{n-m})$, then the left-hand polynomial vanishes at the binary solution. Therefore, an affirmative answer confirms that there is no binary solution to the system. Otherwise, the machine halts in the vague state.

The set $\{\lambda_k\}$ is a solution to an inhomogeneous system of linear equations in n unknowns $\lambda_1, \ldots, \lambda_n$. The system contains only one inhomogeneous equation. Let us denote by r the number of all the equations, that is, $r = \frac{1}{2}(n-m+1)(n-m+2) \leq n$. The sufficient condition for the solvability is the full rank of a $r \times n$ matrix. If $r = n$, then it is sufficient that the determinant does not vanish. If $r < n$, then it is sufficient that some $r \times r$ minor does not vanish. For example, let us pick up the leading principal minor. In any case, it is a polynomial of degree r in matrix entries. Every entry is a polynomial of degree at most two

in coefficients of some ℓ_j. Thus, the minor is a polynomial of degree at most $2r \leq 2n$. To complete the proof, we need to show that this polynomial does not vanish identically. $\square$

Remark 1. Over the field of rational numbers, not only the arithmetic complexity but also the bit complexity is polynomial because the rank can be easily computed [1]. So, there is a generic-case polynomial-time algorithm. Moreover, the rank can be computed in $O(\log^2 n)$ operations over an arbitrary field using a polynomial number of processors [17–18].

Remark 2. Our method can be generalized using higher degree forms. For example, let us consider a general straight line L in the projective plane. There exist four $(0,1)$-points with homogeneous coordinates $(1:0:0)$, $(1:0:1)$, $(1:1:0)$, and $(1:1:1)$, respectively. Our goal is a sufficient condition such that no $(0,1)$-point belongs to L. Every ternary quadratic form vanishing at every $(0,1)$-point is one of the type $\lambda_1 x_1(x_1 - x_0) + \lambda_2 x_2(x_2 - x_0)$. These forms span a linear space of dimension two. Ternary cubic forms vanishing at every $(0,1)$-point span a linear space of dimension six [19]. All binary cubic forms span a linear space of dimension four. The restriction of a ternary cubic form to the straight line L defines a linear map from the linear space of ternary cubic forms to the linear space of binary cubic forms. The kernel of the map is spanned by forms vanishing identically at whole L. Every such form is reducible and has a linear factor corresponding to L. Consequently, the dimension of the kernel equals the dimension of the space of some ternary quadratic forms.

Let the image of a ternary cubic form vanishing at every $(0,1)$-point be its restriction to the general straight line L. The dimension of the kernel of the linear map equals two. The dimension of the image of the linear map equals four and coincides with the dimension of the space of all binary cubic forms. Consequently, the map is surjective. Obviously, its surjectivity is a sufficient condition for the absence of any $(0,1)$-point belonging to L.

Conclusion

We have considered a decision problem. The binary search allows to find binary solutions to sufficiently large systems of linear equations when such a solution exists and some generality assumption holds. So, the proposed method can be used to solve some combinatorial optimization problems that can be reduced to Boolean programming. In particular, such problems arise in bioinformatics.

References

1. *Schrijver A.* Theory of linear and integer programming. John Wiley & Sons, New York, 1986.
2. *Seliverstov A.V.* Binary solutions to some systems of linear equations. In: Eremeev A., Khachay M., Kochetov Y., Pardalos P. (eds) Optimization Problems and Their Applications. OPTA 2018. Communications in Computer and Information Science. Vol. 871. Springer, Cham, 2018. P. 183–192.
3. *Seliverstov A.V.* On binary solutions to systems of equations. Prikladnaya Diskretnaya Matematika. 2019. No. 45. P. 26–32 (in Russian).
4. *Smolev V.V.* On an approach to the solution of a Boolean linear equation with positive integer coefficients. Discrete Mathematics and Applications. 1993. Vol. 3, No. 5. P. 523–530.

5. *Koiliaris K., Xu C.* Faster pseudopolynomial time algorithms for subset sum. ACM Transactions on Computation Theory. 2019. Vol. 15, No. 3. Article 40.

6. *Curtis V.V., Sanches C.A.A.* An improved balanced algorithm for the subset-sum problem. European Journal of Operational Research. 2019. Vol. 275. P. 460–466.

7. *Mucha M., Węgrzycki K., Włodarczyk M.* A subquadratic approximation scheme for partition. In: Chan N.M. (ed.) Proceedings of the 2019 Annual ACM-SIAM Symposium on Discrete Algorithms. 2019. P. 70–88.

8. *Horowitz E., Sahni S.* Computing partitions with applications to the knapsack problem. Journal of the Association for Computing Machinery. 1974. Vol. 21, No. 2. P. 277–292.

9. *Schroeppel R., Shamir A.* A $T = O(2^{n/2})$, $S = O(2^{n/4})$ algorithm for certain NP-complete problems. SIAM Journal on Computing. 1981. Vol. 10, No. 3. P. 456–464.

10. *Bansal N., Garg S., Nederlof J., Vyas N.* Faster space-efficient algorithms for subset sum, k-sum, and related problems. SIAM Journal on Computing. 2018. Vol. 47, No. 5. P. 1755–1777.

11. *Kuzyurin N.N.* An algorithm that is polynomial in the mean in integer linear programming. Sibirskii Zhurnal Issledovaniya Operatsii. 1994. Vol. 1, No. 3. P. 38–48. (In Russian)

12. *Neumann E., Pauly A.* A topological view on algebraic computation models. Journal of Complexity. 2018. Vol. 44. P. 1–22.

13. *Blum L., Shub M., Smale S.* On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines. Bulletin of the American Mathematical Society (N.S.) 1989. Vol. 21, No. 1. P. 1–46.

14. *Seliverstov A.V.* Symmetric matrices whose entries are linear functions. Computational Mathematics and Mathematical Physics. 2020. Vol. 60, No. 1. P. 102–108.

15. *Miasnikov A., Ushakov A.* Generic case completeness. Journal of Computer and System Sciences. 2016. Vol. 82, No. 8. P. 1268–1282.

16. *Rybalov A.N.* On generic complexity of the subset sum problem for semigroups of integer matrices. Prikladnaya Diskretnaya Matematika. 2020. No. 50. P. 118–126. (In Russian)

17. *Chistov A.L.* Fast parallel calculation of the rank of matrices over a field of arbitrary characteristic. In: Budach L. (eds) Fundamentals of Computation Theory. FCT 1985. Lecture Notes in Computer Science, vol 199. Springer, Berlin, Heidelberg. 1985.

18. *Mulmuley K.* A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. Combinatorica. 1987. Vol. 7, No. 1. P. 101–104.

19. *Seliverstov A.V., Lyubetsky V.A.* About forms equal to zero at each vertex of a cube. Journal of Communications Technology and Electronics. 2012. Vol. 57, No. 8. P. 892–895.

Refinements on Bounds for Polynomial Roots

D. Ştefănescu[1]

[1] *Department of Theoretical Physics and Mathematics University of Bucharest, Romania*

e-mail: doru.stefanescu@gmail.com

Abstract. We discuss the efficiency of the computation of bounds for polynomial roots.

Keywords: computer polynomial algebra, bounds for roots

Hypergeometric Type Power Series

B. Teguia Tabuguia[1], W. Koepf[1]
[1]*Mathematics and Natural Sciences, University of Kassel, Germany*
e-mail: {bteguia, koepf}@mathematik.uni-kassel.de

Abstract. For three decades now, the second author proposed a symbolic general-purpose approach to compute formal power series. This was originally presented in three main steps that reduce the problem to solving a holonomic recurrence equation (RE) for the coefficients of the power series sought. However, for linear combinations of Laurent-Puiseux series having hypergeometric term coefficients, one needs to compute so-called m-fold hypergeometric term solutions of holonomic REs. We give an overview of a new algorithm, called **mfoldHyper**, that achieves this purpose and allows linearity in computing hypergeometric type power series.

Keywords: holonomic recurrence equations, m-fold hypergeometric terms, hypergeometric type power series

1. Introduction

Let $\mathbb{K}$ be a field of characteristic zero containing the rationals. A function is holonomic over $\mathbb{K}$ if it satisfies a homogeneous differential equation with polynomial coefficients over $\mathbb{K}$. The class of holonomic functions encompasses a wide family of expressions among which one has the hypergeometric functions. We are interested by this connection to hypergeometric functions with the aim of computing power series. Let f be an analytic (or meromorphic) expression in the indeterminate variable z. A power series representation of f is obtained by: finding a holonomic differential equation (DE) satisfied by f; converting that DE into a holonomic recurrence equation (RE) for the power series coefficients of f; solving that RE and finally use initial values to find a linear combination corresponding to the power series expansion of f. Koepf's initial algorithm could already recover many power series related to the generalized hypergeometric series (see [1]). This is how the terminology "*hypergeometric type*" appeared in the first place. However, in our context the scope of this terminology is wider.

Definition 1. *A term a_n is said to be m-fold hypergeometric, for a positive integer m, if the term ratio a_{n+m}/a_n is a rational function over $\mathbb{K}$. When $m = 1$ one talks about a hypergeometric term. When used without specifying the value of m, m-fold hypergeometric term denotes a sequence with this property, i.e. such an m exists.*

Observe that if a_n is an m-fold hypergeometric term then the sequence (a_n) is an interlacing of m subsequences (a_{mn+l}), $l = 0 \dots m-1$ whose term ratios can be deduced from each other. This is one reason why **mfoldHyper** computes m-fold hypergeometric term solutions of holonomic REs for the specific case $l = 0$ corresponding to a_{mn}.

Consider $f(z) = \cos(z)+\sin(z)$. A holonomic RE satisfied by the power series coefficients of $f(z)$ is

$$(n+1)\ (n+2)\ a_{n+2} + a_n = 0. \tag{1}$$

Therefore $f(z)$ has a 2-fold hypergeometric term coefficient with an even part and an odd part corresponding, respectively, to the coefficient of $\cos(z)$ and the one of $\sin(z)$. Finally using two initial values we get the power series of $f(z)$ as a linear combination of the series of

$\cos(z)$ and the one of $\sin(z)$. Note that $\cos(z)$ and $\sin(z)$ are both related to the generalized hypergeometric series provided a shift and some argument changes (see [1]). As one can see, $f(z)$ is not a generalized hypergeometric series but rather a "*type of*" it. We then say that $f(z)$ is a hypergeometric type function with type $m = 2$. However, the algorithm we use to compute holonomic REs does not guarantee of finding REs of least order possible, though developed with this purpose. For example, $f(z) = \sqrt{1+z} + 1/\sqrt{1+z}$ leads to

$$4\,(n+1)\;a_{n+1} + 6n\,a_n + (2n-3)\;a_{n-1} = 0, \tag{2}$$

which is of second order whereas the series coefficient of $f(z)$ also satisfies the following simple first-order RE

$$2\,(n-1)\;(n+1)\;a_{n+1} + n\;(2n-1)\;a_n = 0, \tag{3}$$

which could be obtained by other means. From (3) it follows that the series of $f(z)$ has a hypergeometric term coefficient.

With the arrival of Petkovšek's (1993) and van Hoeij's (1998) algorithms (see [2,3]) for finding hypergeometric term solutions of holonomic REs, the case of $f(z)$ could already be solved. However, there comes an extension of the initial hypergeometric type family, because using Petkovšek's or van Hoeij's algorithm permits to find many hypergeometric term solutions of the same RE. A simple example of this category is $\exp(z) + 1/(1+z)$ whose power series coefficients satisfy the RE

$$2\,(n+1)\;(n+2)\;a_{n+2} + (n+1)\;(3n+1)\;a_{n+1} + \left(n^2 - 3n - 3\right)\;a_n - n a_{n-1} = 0, \tag{4}$$

which has the following basis of hypergeometric term solutions

$$\left\{(-1)^n\,, \frac{1}{n\,!}\right\}. \tag{5}$$

In a more general sense, the RE taken into consideration could have m-fold hypergeometric term solutions with several different values of m. In this case Petkovšek's and van Hoeij's algorithms will not detect the solutions for $m \geqslant 2$ since they are only designed for hypergeometric terms ($m = 1$). This theme is approached differently in [4].

In the first author Ph.D. thesis [5], a new algorithm, called **mfoldHyper**, to efficiently compute a basis of the subspace of m-fold hypergeometric term solutions of holonomic REs was proposed (see [6]). The algorithm is implemented in the computer algebra systems (CAS) Maple and Maxima into packages of name `FPS` (see [7]). The latter is also the name of our main command used to compute formal power series whose hypergeometric type series is a subcase. `FPS` will be incorporated into Maple 2022 and its incorporation into Maxima is in progress.

Mark van Hoeij's algorithm is accessible in the CAS Maple through `LREtools[hypergeomsols]`, and Koepf's original algorithm is implemented in Maple 2021 by `convert/FormalPowerSeries` with an internal call to `LREtools[hypergeomsols]` when needed. Let $f(z) = \log(1 + z + z^2 + z^3)$. The power series coefficients of $f(z)$ satisfy the holonomic RE:

$$\begin{aligned}&(n+1)\;(n+2)\;a_{n+2} + (n+1)\;(3n-1)\;a_{n+1} + 2\,n\;(3n-5)\;a_n\\ &+2\;(n-1)\;(3n-5)\;a_{n-1} + (n-2)\;(5n-11)\;a_{n-2} + 3\,(n-3)^2 a_{n-3} = 0\end{aligned} \tag{6}$$

`LREtools[hypergeomsols]` finds no solution over the rationals whereas using `FPS[mfoldHyper]` we get the following basis of m-fold hypergeometric term solutions

$$\left[\left[1, \left\{\frac{(-1)^n}{n}\right\}\right], \left[2, \left\{\frac{(-1)^n}{n}\right\}\right]\right]. \tag{7}$$

Since there is a 2-fold hypergeometric solution, we should compute the remaining interlacing term. This can be done with our Maxima implementation as `mfoldHyper(RE,a[n],2,1)` (assuming the package is already loaded) and in Maple as `FPS[mfoldHyper](RE,a(n),ml=[2,1])`. This yields

$$\left\{\frac{(-1)^n}{2n+1}\right\}. \tag{8}$$

Finally using initial values (around zero) we solve a linear system whose solution leads to the following linear combination

$$\sum_{n=0}^{\infty} \frac{(-1)^n z^{2(n+1)}}{n+1} + \sum_{n=0}^{\infty} \frac{(-1)^n z^{n+1}}{n+1}, \tag{9}$$

which is the power series representation of $f(z)$ sought.

2. Definition and key results

Let us now give a definition of "hypergeometric type power series" that meets the wide family of formal power series that can be computed thanks to **mfoldHyper**.

Definition 2. *For an expansion around $z_0 \in \mathbb{K}$, a series $s(z)$ is said to be of hypergeometric type if it can be written as*

$$s(z) := T(z) + \sum_{j=1}^{J} s_j(z), \;\; s_j = \sum_{n=n_{j,0}}^{\infty} a_{j,n}(z-z_0)^{n/p_j} \tag{10}$$

where n is the summation variable, $T(z) \in \mathbb{K}[z, 1/z, \ln(z)]$, $n_{j,0} \in \mathbb{Z}$, $J, p_j \in \mathbb{N}$, and $a_{j,n}$ is an m_j-fold hypergeometric term.

Thus a hypergeometric type power series is a linear combination of Laurent-Puiseux series whose coefficients are m-fold hypergeometric terms. A hypergeometric function is a function that can be expanded as a hypergeometric type power series. T is called the Laurent polynomial part of the expansion, and the p_j's are its Puiseux numbers.

Our main result is the theorem upon which **mfoldHyper** is based. This is accompanied by other lemmas and theorems to determine all the remaining data in (10): the Laurent polynomial part, the Puiseux numbers, and the linear combination whose coefficients are absorbed by the m-fold hypergeometric terms in (10).

The following two definitions are two prerequisites for understanding Theorem 1 (see [5, 6]).

Definition 3. *Let m be a positive integer. A holonomic RE is said to be m-fold holonomic if it has at least two non-zero terms, and the difference between every pair of indices in the equation is a multiple of m.*

Provided a change of variable, an m-fold holonomic recurrence equation can always be transformed into a 1-fold holonomic recurrence equation. (1) is a 2-fold holonomic RE.

Definition 4. *Let m be a positive integer. Two m-fold holonomic REs are said to be m-fold distinct, if the difference between any index taken from one and another taken from the second is not a multiple of m.*

Theorem 1. *Let $m \in \mathbb{N}$, $\mathbb{K}$ a field of characteristic zero, and h_n be an m-fold hypergeometric term which is not u-fold hypergeometric over $\mathbb{K}$ for all positive integers $u < m$. Then h_n is a solution of a given holonomic recurrence equation, if that equation can be written as a linear combination of m-fold holonomic recurrence equations; such that h_n is solution of each of the m-fold distinct holonomic recurrence equations of that linear combination.*

As a corollary, **mfoldHyper** is an iterative algorithm for m from 1 to the order of the given RE, which computes m-fold hypergeometric term solutions of m-fold distinct REs appearing in the linear combination encountered in each iteration.

3. Examples

Below we give some examples of hypergeometric type power series around $z_0 = 0$ that are linearly automatically computed with our new approach.

$$z\,\cos(z^{3/2}) + \arcsin(z^{1/3})^2 = \sum_{n=0}^{\infty} \frac{(-1)^n\, z^{3\,n+1}}{(2n)\,!} + \sum_{n=0}^{\infty} \frac{2\,4^n\,n\,!^2\, z^{\frac{2(n+1)}{3}}}{(2\,(n+1))\,!} \tag{11}$$

$$\begin{aligned}\frac{1}{(p-z^2)(q-z^3)} = &\sum_{n=0}^{\infty} -\frac{\left(qp^{-1-n/2} - pq^{-1/3-n/3}\right) z^n}{p^3-q^2} + \sum_{n=0}^{\infty} -\frac{\left(p^{3/2}-q\right) p^{-n-3/2} z^{2\,n+1}}{p^3-q^2} \\ &+ \sum_{n=0}^{\infty} -\frac{q^{-1-n} p\left(q^{2/3}-p\right) z^{3\,n}}{p^3-q^2} + \sum_{n=0}^{\infty} \frac{\left(q^{2/3}-p\right) q^{-n-2/3} z^{3\,n+1}}{p^3-q^2}\end{aligned} \tag{12}$$

$$\operatorname{arcsech}(z) + \exp(z^3) = \sum_{n=0}^{\infty} -\frac{4^{-n-1}\,(2\,n+1)\,!\,z^{2(n+1)}}{(n+1)\,!^2} + \sum_{n=0}^{\infty} \frac{z^{3n}}{n\,!} - \log(z) + \log(2) \tag{13}$$

References

1. *Koepf W.* Power series in computer algebra. J. Symb. Comput. 1992. Vol. 13, No. 6. P. 581–603.
2. *Koepf W.* Hypergeometric summation. An algorithmic approach to summation and special function odentities. Springer, London, 2014.
3. *Teguia Tabuguia B.* A variant of van Hoeij's algorithm to compute hypergeometric term solutions of holonomic recurrence equations. arXiv:2012.11513 [cs.SC] preprint, 2020. (Submitted)
4. *Ryabenko A.A.* Formal solutions of linear ordinary differential equations containing m-hypergeometric series. Programming and Computer Software. 2002. Vol. 28. P. 92–101.
5. *Teguia Tabuguia B.* Power series representations of hypergeometric types and non-holonomic functions in computer algebra. Ph.D. thesis, University of Kassel, 2020. *URL: https://kobra.uni-kassel.de/handle/123456789/11598*
6. *Teguia Tabuguia B. and Koepf W.* Symbolic computation of hypergeometric type and non-holonomic power series. arXiv:2102.04157 [cs.SC] preprint, 2021. (Submitted)

7. *Teguia Tabuguia B. and Koepf W.* Power series representations of hypergeometric type functions. To appear in: Corless R., Gerhard J., Kotsireas I. (eds): Maple in Mathematics Education and Research. MC 2020. Communications in Computer and Information Science, Springer.

High Accuracy Trigonometric Approximations of the Real Bessel Functions of the First Kind

Min Wu[1]

[1]*East China Normal University, Shanghai, P.R.China*

e-mail: mwu@sei.ecnu.edu.cn

Abstract. We construct high accuracy trigonometric interpolants from equidistant evaluations of the Bessel functions $J_n(x)$ of the first kind and integer order. The trigonometric models are cosine or sine based depending on whether the Bessel function is even or odd. The main novelty lies in the fact that the frequencies in the trigonometric terms modelling $J_n(x)$ are also computed from the data in a Prony-type approach. Hence the interpolation problem is a nonlinear problem. Some existing compact trigonometric models for the Bessel functions $J_n(x)$ are hereby rediscovered and generalized.

Keywords: trigonometric approximation, Prony's method, Prony-like method, Bessel functions

1. Bessel functions of the first kind

The Bessel functions $J_\nu(x)$ of the first kind and order ν satisfy the second order differential equation

$$x^2\frac{d^2y(x)}{dx^2} + x\frac{dy(x)}{dx} + (x^2-\nu^2)y(x) = 0, \qquad \nu \in \mathbb{C}.$$

They are therefore especially important in many scientific computing problems involving wave propagation and static potentials. Among others, we mention signal processing, electromagnetics, acoustical radiation and vibration analysis.

We are interested in Bessel functions of the first kind of positive integer order and real arguments. Note that for negative integer order we have the relation

$$J_{-n}(x) = (-1)^n J_n(x) = J_n(-x), \qquad n \in \mathbb{N}, \quad x \in \mathbb{R}.$$

In the literature, various approximations of the Bessel functions $J_n(x)$ can be found, including power series, asymptotic series, Fourier series expansions and quadrature formulas applied to integral representations, where the latter three ones are trigonometric approximations. Mostly these approximations are constructed to provide simple compact models guaranteeing a few significant digits on a finite interval $0 \le x \le B$.

The aim of the current work is to provide equally compact formulas constructed from discrete data, using suitably selected frequencies and delivering high accuracy approximants.

From the graphs of the Bessel functions $J_n(x)$ of the first kind for integer orders, one notices that they behave very much like decaying trigonometric functions.

Since our trigonometric approximations are constructed from a finite number of uniformly collected samples of $J_n(x)$, we focus on the behaviour of $J_n(x)$ and its approximations on a finite interval $[0, B]$ with $B > 0$. This restriction doesn't inherently change the behaviour of the Bessel function since the function $J_n(B; x)$ is a somewhat magnified version of $J_n(x)$ for $B > 1$ and $0 < x \le B$.

For increasing B, we find that the new approximations introduced in this work have a far better overall behaviour, meaning a smaller uniform norm of the relative error, than the existing power series and asymptotic series expansions which only perform well at either end of the interval [7, 3].

1.1 Finite sum cosine approximations

In this work, we obtain the following formulas for the construction of an m-term cosine interpolant with coefficients α_k and frequencies ϕ_k, satisfying the interpolation conditions

$$\sum_{k=1}^{m} \alpha_k \cos(\phi_k j\Delta) = f_j, \qquad j = 0, \ldots, 2m-1, \tag{1}$$

where the sampled (restricted) Bessel function is an even function.

The idea of our work is as follows. We first solve (1) for the unknown nonlinear parameters ϕ_k in the interpolant and afterwards for the unknown linear coefficients α_k. The nonlinear problem of computing α_k's and ϕ_k's can be converted into solving a generalized eigenvalue problem to obtain $\phi_k\Delta$ and then solving a linear system involving α_k's. The values $\cos(\phi_k\Delta), k = 1, \ldots, m$ are the generalized eigenvalues of [5, 4]

$$C_m^{(1)}\, v_k = \cos(\phi_k\Delta)\, C_m^{(0)}\, v_k, \qquad k = 1, \ldots, m, \tag{2}$$

where the v_k are the generalized eigenvectors. Under the condition that

$$0 \le \max_k \phi_k\Delta \le \pi \tag{3}$$

the frequencies $\phi_k, k = 1, \ldots, m$ can be unambiguously extracted from the generalized eigenvalues $\cos(\phi_k\Delta), k = 1, \ldots, m$. With the ϕ_k identified, the interpolation problem (1) can be solved for the coefficients α_k. In an exact mathematical setting it suffices to consider a subset of m interpolation conditions of the $2m$ imposed ones, as the remaining m conditions have become linearly dependent because of the generalized eigenvalue relation satisfied by the ϕ_k. With respect to (3) we make the following remarks and observations. With $\Delta = B/(2m-1)$ condition (3) amounts to

$$0 \le \max_k \phi_k \le \frac{(2m-1)\pi}{B},$$

which implies that the choice of B and m determines which frequencies in $J_n(B; x)$ or $J_n(x)$ can be identified without aliasing effect. In other words, the choice of B and m limits the frequency range of the parameters ϕ_k in the models (1) and (4).

1.2 Finite sum sine approximations

Similarly, from [4], we find similar formulas for the construction of an m-term sine interpolant for $J_n(B; x)$, satisfying

$$\sum_{k=1}^{m} \alpha_k \sin(\phi_k j\Delta) = f_j, \qquad j = 1, \ldots, 2m, \tag{4}$$

where the sampled (restricted) Bessel function of the first kind is an odd function. The values $\cos(\phi_k\Delta)$ are the generalized eigenvalues of the generalized eigenvalue problem

$$S_m^{(1)}\, v_k = \cos(\phi_k\Delta)\, S_m^{(0)}\, v_k, \qquad k = 1, \ldots, m. \tag{5}$$

Under the condition that $|\max_k \phi_k\Delta| \le \frac{\pi}{2}$, or equivalently

$$\max_k |\phi_k| \le \frac{(2m-1)\pi}{2B}, \tag{6}$$

the frequencies ϕ_k can be uniquely extracted from the generalized eigenvalues $\cos(\phi_k \Delta)$ computed from (5) and then the coefficients $\alpha_k, k = 1, \ldots, m$ can be computed from the interpolation conditions (4) as above. Our recommendation now is to respect $2B < (2m - 1)\pi$.

To compare the approximation methods, we use the formula

$$\frac{|J_n(B;x) - R_m(x)|}{1 + |J_n(B;x)|} \text{ or } \frac{|J_n(x) - R_m(x)|}{1 + |J_n(x)|}$$

for the relative error from approximating the function value $J_n(B;x)$ or $J_n(x)$ by the trigonometric m-term sum $R_m(x)$. The denominator takes care of any occurrence of zeroes: in their neighbourhood the relative error is gradually replaced by the absolute error.

We compare the new Prony-type approximants to the power series and asymptotic series expansions of the Bessel functions of the first kind. The experiments show that the cosine and sine sums are, as expected, much better when regarding a wider range for x, while the power series and asymptotic series expansions only perform well for either small or large values of the argument, respectively.

Furthermore, the choice for B satisfies the recommendations formulated in the discussion of (3) and (6) for most cases. When using $2B \geq (2m - 1)\pi$ in the sine model, then the frequency range for the ϕ_k is reduced and introduces an unwanted aliasing effect from the frequencies in the range $[(2m - 1)\pi/(2B), 1]$.

2. Simplified procedure

When inspecting the computed frequencies ϕ_k in the models (1) and (4) of the previous section, we observe that:

- all (or almost all) the $\phi_k, k = 1, \ldots, m$ are real and lie in the interval $[0, 1]$;
- their distribution becomes denser towards the interval endpoint 1.

This behaviour is reminiscent of the behaviour of Chebyshev zeroes and extrema, when restricting them to the interval $[0, 1]$. We also note that the frequencies appearing in the simple models coincide with the extrema of a Chebyshev polynomial of the first kind restricted to $[0, 1]$.

Therefore we now investigate the accuracy of the simplified approximants $\sum_{k=1}^{m} \alpha_k \cos(\tilde{\phi}_k x)$ and $\sum_{k=1}^{m} \alpha_k \sin(\tilde{\phi}_k x)$ where the $\tilde{\phi}_k$ are not obtained from the solution of the generalized eigenvalue problems (2) or (5) but are fixed a priori, preferably as some Chebyshev extrema or zeroes.

For the $\tilde{\phi}_k$ we consider 5 different schemes, where $T_n(x)$ and $U_n(x)$ respectively denote the Chebyshev polynomials of the first and second kind:

1. from the zeroes of $T_{2m}(x)$: $\tilde{\phi}_k = \cos\left(\frac{(2k-1)\pi}{4m}\right), \qquad k = 1, \ldots, m,$ (7)
2. from the zeroes of $U_{2m}(x)$: $\tilde{\phi}_k = \cos\left(\frac{k\pi}{2m+1}\right), \qquad k = 1, \ldots, m,$
3. from the zeroes of $T_{2m+1}(x)$: $\tilde{\phi}_k = \cos\left(\frac{(2k-1)\pi}{2(2m+1)}\right), \qquad k = 1, \ldots, m,$ (8)
4. from the extrema of $T_{2m}(x)$: $\tilde{\phi}_k = \cos\left(\frac{k\pi}{2m}\right), \qquad k = 1, \ldots, m,$

5. from the extrema of $T_{2(m-1)}(x)$: $\tilde{\phi}_k = \cos\left(\frac{k\pi}{2(m-1)}\right), \qquad k = 0, \ldots, m-1.$ (9)

The numerical experiments show that the simplified approximants using frequencies $\tilde{\phi}_k$ from either (7), (8) or (9) deliver the smallest truncation errors and follow best the optimum trend of (1). The conclusion for the sine case is similar.

3. Conclusion

The proposed Prony-like method generates quite high accuracy approximants, which we believe are unexplored so far. In view of the many physics and engineering applications involving Bessel functions of the first kind of integer order, these exploratory results may offer interesting opportunities.

In the future, a more complete comparison of the newly introduced trigonometric approximants with different representations of the Bessel functions of the first kind, should also be conducted.

Precise statements for general order functions $J_n(x)$ on a guaranteed truncation error bound for these interpolants in the interval $[0, B]$ and the exact behaviour of the frequencies present in the oscillating and decaying functions $J_n(x)$ form interesting topics for future research.

References

1. *Andrusyk A.* Infinite series representations for Bessel functions of the first kind of integer order. Tech. rep., Institute for Condensed Matter Physics, Lviv, Ukraine, 2012. arXiv preprint arXiv:1210.2109
2. *Blachman N.M., Mousavinezhad S.H.* Trigonometric approximations for Bessel functions. IEEE Trans. Aerosp. Electron. Syst. AES-22. 1986. P. 2–7.
3. *Cuyt A., Brevik Petersen V., Verdonk B., Waadeland H., Jones W.B.* Handbook of continued fractions for special functions. Springer, Berlin, 2008.
4. *Cuyt A., Lee W.s.* Sparse trigonometric and sinc spectral analysis. Tech. rep., Universiteit Antwerpen, 2019, in preparation.
5. *Giesbrecht M., Labahn G., Lee W.s.* Symbolic-numeric sparse polynomial interpolation in Chebyshev basis and trigonometric interpolation. In: Proc. Workshop on Computer Algebra in Scientific Computation (CASC). 2004. P. 195–204.
6. *Hildebrand F.B.* Introduction to numerical analysis. Dover Publications, Inc., second edn., 1987.
7. *Oldham K., Myland J., Spanier J.* An atlas of ffunctions. Springer, New York, NY, second edn., 2009. https://doi.org/10.1007/978-0-387-48807-3
8. *de Prony R.* Essai expérimental et analytique sur les lois de la dilatabilité des fluides élastiques et sur celles de la force expansive de la vapeur de l'eau et de la vapeur de l'alkool, à différentes températures. J. Ec. Poly. 1795. Vol. 1. P. 24–76.
9. *Sneddon I.N.* The use of integral transforms. Mc-Graw Hill, New York, 1972.

Machine Learning for Bratu's Problem: Solution and Parameter Estimation

M. Youssef[1], R. Pulch[2]

[1]*Institute of Applied Analysis and Numerical Simulation, University of Stuttgart, Germany*

[2]*Institute of Mathematics and Computer Science, University of Greifswald, Germany*

e-mail: maha.youssef-ismail@mathematik.uni-stuttgart.de, roland.pulch@uni-greifswald.de

Abstract. We investigate a parametric nonlinear Bratu equation. Our aim is to determine cheap and efficient approximations of the solution and the parameter values. We define the truth solution as the Poly-Sinc collocation solution. We arrange a set of samples including basis coefficients of the truth solution. The target is to approximate the mapping from the parameter domain to the basis coefficients and vice versa. We apply machine learning with artificial neural networks for these approximations. We present results of numerical computations for both the forward problems and inverse problems. The calculations include one-dimensional and higher-dimensional models.

Keywords: Bratu's problem, Parametric PDEs, Poly-Sinc Collocation method, Nonlinear BVPs, Machine Learning, Neural Network, Inverse Problem.

References

1. *Abbott J.P.* An efficient algorithm for the determination of certain bifurcation points. J. Comp. Appl. Math. 1978. Vol. 4(19).
2. *Stenger F.* Handbook of Sinc Methods. CRC Press, 2011.
3. *Youssef M., Baumann G.* Troesch's problem solved by Sinc methods. Math. Comput. Simulat. 2019. Vol. 162. P. 31–44. https://doi.org/10.1016/j.matcom.2019.01.003
4. *Higham C.F., Higham D.J.* Deep Learning: An introduction for applied mathematicians. SIAM REVIEW. 2019. Vol. 61(4). P. 860–891.

On the Structure of Polynomial Solutions of Gosper's Key Equation

E.V. Zima[1]

[1] *Wilfrid Laurier University, Waterloo, Canada*

e-mail: ezima@wlu.ca

Abstract. The structure of polynomial solutions to the Gosper's key equation is analyzed. A method for rapid "extraction" of simple high-degree factors of the solution is given. It is shown that in cases when equation corresponds to a summable non-rational hypergeometric term the Gosper's algorithm can be accelerated by removing non-essential dependency of its running time on the value of dispersion of its rational certificate.

Keywords: Indefinite hypergeometric summation, accelerated Gosper's algorithm, factorial polynomials.

Let $\mathbb{K}$ be a field of characteristic zero, x – an independent variable, E – the shift operator with respect to x, i.e., $Ef(x) = f(x+1)$ for an arbitrary $f(x)$. Recall that a nonzero expression $F(x)$ is called a hypergeometric term over $\mathbb{K}$ if there exists a rational function $r(x) \in \mathbb{K}(x)$ such that $F(x+1)/F(x) = r(x)$. Usually $r(x)$ is called the rational *certificate* of $F(x)$. The problem of indefinite hypergeometric summation (anti-differencing) is: given a hypergeometric term $F(x)$ find a hypergeometric term $G(x)$, which satisfies the first order linear difference equation

$$(E-1)G(x) = F(x). \tag{1}$$

If found, write $\sum_x F(x) = G(x) + c$, where c is an arbitrary constant.

An important notion widely used in the context of algorithmic summation is the *dispersion set* of polynomials $p(x)$ and $q(x)$, which is the set of positive integers h such that $\deg(\gcd(p(x+h), q(x))) > 0$. Another important notion is the largest element of the dispersion set known as the *dispersion* [1].

One more piece of standard terminology required here is the notion of *shift equivalence* of polynomials: two polynomials $u(x), v(x) \in \mathbb{K}[x]$ are shift equivalent if there exists $h \in \mathbb{Z}$, such that $u(x+h) = v(x)$. Given several distinct shift equivalent polynomials $u_1(x), \ldots, u_t(x)$ it is easy to list them in order from "leftmost" to "rightmost", i.e. reorder them as $v_1(x), \ldots, v_t(x)$ in such a way that for any indices $i, j \in \{1, \ldots, t\}, i < j$, $v_j(x + h_{ij}) = v_i(x)$ and $h_{ij} > 0$. We refer to $v_1(x)$ as the smallest and to $v_t(x)$ as the largest element of shift equivalence class formed by $u_1(x), \ldots, u_t(x)$.

Finally, following [5] define the *factorial polynomial* (a generalization of the falling factorial) for $p(x) \in \mathbb{K}[x]$ as

$$[p(x)]_k = p(x) \cdot p(x-1) \cdot \ldots \cdot p(x-k+1) \tag{2}$$

for $k > 0$ and $[p(x)]_0 = 1$.

Although the algorithmic treatment of symbolic summation has a long history [1,3], standard algorithms and their implementations in computer algebra systems suffer from the fact that they can unnecessarily require a running time which is exponential in the input size. This is due to the well know effect of "intermediate expression swell". In algorithms for hypergeometric summation the value of dispersion of the rational certificate of a hypergeometric term plays crucial role. The dispersion can be exponentially large in the size of the summand, and the running time of Gosper's algorithm [3] effectively depends on the dispersion. This means that even when the closed form sum is small, the intermediate results can

be exponentially large, and the performance of the algorithm deteriorates. This makes the instances of summation problem with large dispersion effectively intractable.

In what follows we describe simple modifications of the Gosper's algorithm that remove non-essential dependency of the running time of Gosper's procedure on the dispersion.

Consider $R \in \mathbb{K}(x)$. If $z \in \mathbb{K}$ and monic polynomials $A, B, C \in \mathbb{K}[x]$ satisfy
(i) $R = z \cdot \frac{A}{B} \cdot \frac{EC}{C}$,
(ii) A and $E^k B$ are relatively prime for all $k \in \mathbb{N}$,
then (z, A, B, C) is a *polynomial normal form* (PNF) of R. If in addition,
(iii) A is relatively prime to C and B is relatively prime to EC,
then (z, A, B, C) is a *strict* (PNF) of R (see [6,2] for details).

For example, for the rational function $\frac{(x+100)\left(2x^2+1\right)}{x}$ the PNF is

$$2, x^2 + \frac{1}{2}, 1, (x+99) \cdot (x+98) \cdot (x+97) \cdot \cdots \cdot (x+1) \cdot x$$

with polynomial $C = [x+99]_{100}$ of degree 100. Note, that factorial polynomials naturally appear in the last component of PNF, assuming that the dispersion of the numerator and denominator of a given rational function is non-zero.

In [3] Gosper described a decision procedure for the summation of a hypergeometric term, which is widely adopted and used in computer algebra systems. The algorithm is based on simple observation that if a given hypergeometric term $F(x)$ has a hypergeometric anti-difference $G(x)$ (i.e. if it is summable), then the terms $G(x)$ and $F(x)$ are *similar*: i.e., there exists $Y(x) \in \mathbb{K}(x)$ such that $G(x) = Y(x)F(x)$ (in other words, the anti-difference is a rational function multiple of the summand). This reduces the original summation problem to the problem of finding a rational function $Y(x)$ solving equation

$$Y(x+1)r(x) - Y(x) = 1, \tag{3}$$

where $r(x)$ is the rational certificate of the summand.

In order to solve (3), the rational certificate is transformed to the Gosper-Petkovšek form (or PNF): (z, A, B, C), i.e. one finds $z \in \mathbb{K}$ and polynomials $A(x), B(x), C(x)$ such that

$$r(x) = z\frac{C(x+1)A(x)}{C(x)B(x)}.$$

reducing the search for the sum to the search of a polynomial solution $y(x)$ of the *key equation*:

$$zA(x)y(x+1) - B(x-1)y(x) = C(x). \tag{4}$$

If $y(x)$ is found, then

$$G(x) = F(x)\frac{B(x-1)y(x)}{C(x)}. \tag{5}$$

In order to solve (4) one can find a degree bound of the solution and use (for example) the method of undetermined coefficients to reduce the problem to standard linear algebra routine. One of the factors influencing this bound (and as a consequence the size of linear system to be solved) is the degree of $C(x)$ in (4).

Sometimes the polynomial $C(x)$ from (4) is called *the universal denominator*. One well-known problem with Gosper's algorithm is that the universal denominator can have pessimistically large degree (i.e., $C(x)$ and $y(x)$ can have very large degree common factor, which will cancel after substituting the solution $y(x)$ into (5)). This in turn can lead to the exponential dependency of the running time on the size of the input, even when input and output is small. We will show that it is possible (after obtaining the degree bound for the

solution of (4)) to remove extraneous terms in this universal denominator before solving the key equation.

Assume that the given hypergemetric term $F(x)$ is not a rational function and that it is summable. Cancellation in (5) happens when solution $y(x)$ and right hand side $C(x)$ of (4) have common factor. Suppose $C(x) = [p(x)]_k \tilde{C}(x)$ and solution $y(x)$ also has factor $[p(x)]_k$. Then substitution $y(x) = \tilde{y}(x)[p(x)]_{k+1}$ into (4) gives new equation

$$zA(x)p(x+1)\tilde{y}(x+1) - B(x-1)p(x-k)\tilde{y}(x) = \tilde{C}(x) \tag{6}$$

which has polynomial solution $\tilde{y}(x)$ and $y(x)$ in (5) can be replaced by $\tilde{y}(x)$, $C(x)$ in (5) can be replaced by $\tilde{C}(x)$ (effectively realizing cancellation of unnecessary common factor in the numerator and denominator of (5)). If the degree bound for $y(x)$ in (4) is N, then the degree bound for $\tilde{y}(x)$ in (6) is $N - k \deg p(x)$.

Note, that due to the nature of PNF, factorial polynomials appear only in the right-hand side of the key equation (4) and they are the only candidates for cancellation. Moreover, the number of these factorial polynomials is bounded by the degrees of the numerator and the denominator of the rational certificate $r(x)$ and does not depend on the value of the dispersion. On the other hand, each term of the form $[p(x)]_k$ contributes the value of $k \deg p(x)$ towards the upper bound of the degree of the solution $y(x)$, which can be as large as the value of the dispersion.

In what follows let $[p(x)]_k$ be one of the factors of $C(x)$ in (4). Our approach is based on a succinct representation of the factorial polynomials appearing in the Gosper-Petkovšek form, lazy evaluation of consecutive values of $y(x)$ in (4) and very simple properties of the structure of the solutions of the equation (4):

1. The term $[p(x)]_k$ vanishes at any root α of $p(x)$ and also at $\alpha+1, \ldots, \alpha+k-1$.
2. Neither $A(x)$ nor $B(x-1)$ can be equal to zero at $\alpha+1, \ldots, \alpha+k-1$.
3. If a solution $y(x)$ of (4) is equal to zero at any of $\alpha, \alpha+1, \ldots, \alpha+k$ (where α is a root of $p(x)$), then $y(x)$ is equal to zero at all these points. This means that $[p(x)]_{k+1}$ is a factor of $y(x)$ and the factorial polynomial term $[p(x)]_k$ in $C(x)$ cancels after substituting $y(x)$ into (5).
4. If neither $A(x)$ nor $B(x-1)$ contains a factor shift equivalent to $p(x)$, then the term $[p(x)]_k$ is a factor of the solution $y(x)$. This property holds only for non-rational hypergeometric summands and is based on the fact that the homogeneous equation

   $$zA(x)y(x+1) - B(x-1)y(x) = 0$$

   corresponding to equation (4) can only have trivial solution if $z, A(x)$ and $B(x)$ come from PNF for non-rational hypergeometric term (as shown in [4]).
5. Any shift equivalent to $p(x)$ factor of $A(x)$ (resp. $B(x-1)$) from (4) provides initial value for the solution of $y(x)$ at β (resp. at $\beta+1$), where β is a root of this factor.
6. The evaluations required to detect equality or non-equality of $y(x)$ to zero at the consecutive points starting at β can be done lazily. The expanded form of $[p(x)]_k$ or the knowledge of the root β itself are not required for this test.

For example, let $C(x) = [p(x)]_k \tilde{C}(x)$ in (4), $b(x)$ is a factor of $B(x-1)$ shift equivalent to $p(x)$ (i.e., $p(x+h) = b(x), h > 0$). Reducing equation (4) modulo $b(x)$ gives us $y(x+1) = A(x)^{-1}\tilde{C}(x)[p(x)]_k \mod b(x)$. Using this as initial value we can write

$$y(x+2) = A(x+1)^{-1}\left[B(x)A(x)^{-1}\tilde{C}(x)p(x-k+1) + \tilde{C}(x+1)p(x+1)\right][p(x)]_{k-1} \mod b(x).$$

The value of $y(\beta+2)$ will be equal to zero iff the expression in square brackets above is zero polynomial. In order to test this equality we do not need to fully evaluate right hand side. Continuing to use similar scheme we can find if $y(x+h)$ is equal to zero, and if it is – we have detected that factor $[p(x)]_k$ is part of the solution of (4) (so, it can be removed as described before). Note, that due to the nature of PNF polynomials $A(x),\dots,A(x+h)$ are all invertible modulo $b(x)$.

One can try to minimize the number of intermediate lazy evaluation steps in cases when both $A(x)$ and $B(x-1)$ contain factors shift equivalent to $p(x)$. For example, let $C(x) = [p(x)]_k\tilde{C}(x)$ in (4), $b(x)$ is a factor of $B(x-1)$ shift equivalent to $p(x)$, and $a(x)$ is a factor of $A(x)$ shift equivalent to $p(x)$. Note that $a(x)$ is the leftmost element and $b(x)$ is the rightmost element in the group of shift-equivalent polynomials $a(x), p(x-k+1), p(x-k+2),\dots,p(x),b(x)$. If one starts evaluations at a root of $b(x)$ (respectively root of $a(x)$) - the number of evaluation steps needed to detect if solution $y(x)$ is zero at a root of $p(x)$ is equal to dispersion of $p(x)$ and $b(x)$ (respectively dispersion of $a(x)$ and $p(x-k+1)$). The smaller of these values gives fewer lazy evaluation steps.

Described above properties allow us to incorporate simple and efficient changes into Gosper's decision procedure, which do not worsen the total asymptotic complexity of the procedure, but can lead to tremendous savings in the running time for summable terms with large dispersion of the rational certificate (effectively, in this case modified Gosper's procedure has running time which is polynomial in the size of the input [7]). Implementation of described technique in Maple shows practical improvements in the running time for summable non-rational hypergeometric terms with large dispersion of the rational certificate. Note, that there are situations when given hypergeometric term is known to be summable in advance (for example in reduction based creative telescoping).

For the non-summable terms proposed reductions of the universal denominator can be used for factors $[p(x)]_k$ for which neither $A(x)$ nor $B(x-1)$ contains a factor shift equivalent to $p(x)$. If (4) has no polynomial solution then (6) can not have polynomial solution, but equation (6) has lower degree of the right hand side. This means that all such factors of the right-hand side of (4) are redundant and their removal will improve time to decide that there is no solution. The question if other factorial polynomials can be safely removed from the universal denominator requires further investigation.

References

1. *Abramov S.A.* The summation of rational functions. U.S.S.R. Computational Mathematics and Mathematical Physics. 1971. Vol. 11(4). P. 324–330.
2. *Abramov S.A., Petkovšek M.* Rational normal forms and minimal decompositions of hypergeometric terms. Journal of Symbolic Computation. 2002. Vol. 33(5). P. 521–543.
3. *Gosper R.W.* Decision procedure for indefinite hypergeometric summation. Proc. Nat. Acad. Sci. U.S.A. 1978. Vol. 75(1). P. 40–42.
4. *Lisonek P., Paule P., Strehl V.* Improvement of the degree setting in Gosper's algorithm. Journal of Symbolic Computation. 1993. Vol. 16. P. 243–258.
5. *Moenck R.* On computing closed forms for summations. In Proceedings of the 1977 MACSYMA Users' Conference. 1977. P. 225–236.
6. *Petkovšek M.* Hypergeometric solutions of linear recurrences with polynomial coefficients. Journal of Symbolic Computation. 1992. Vol. 14(2). P. 243–264.

7. *Zima E.V.* Accelerating indefinite hypergeometric summation algorithms. ACM Commun. Comput. Algebra. 2019. Vol. 52(3). P. 96–99.

Author index

Научное издание

КОМПЬЮТЕРНАЯ АЛГЕБРА

Материалы 4-й международной конференции

Москва, 28–29 июня 2021 г.

Издательство «МАКС Пресс»
Главный редактор: *Е. М. Бугачева*

Напечатано с готового оригинал-макета
Подписано в печать 28.05.2021 г.
Формат 60x90 1/16. Усл.печ.л. 7,75.
Тираж 300 (1–80) экз. Заказ 86.

Издательство ООО “МАКС Пресс”
Лицензия ИД N 00510 от 01.12.99 г.
119992, ГСП-2, Москва, Ленинские горы, МГУ им. М.В. Ломоносова,
2-й учебный корпус, 527 к.
Тел. 8(495)939-3890/91. Тел./Факс 8(495)939-3891.

Отпечатано в полном соответствии с качеством
предоставленных материалов в ООО «Фотоэксперт»
115201, г. Москва, ул. Котляковская, д.3, стр. 13.