Sparse Rational Univariate Representation

A. Mantzaflaris, E. Schost, E. Tsigaridas

RICAM-Report 2017-20

Sparse Rational Univariate Representation

Angelos Mantzaflaris
RICAM, Austrian Academy of Sciences
Altenberger Str. 69, 4040
Linz, Austria
angelos.mantzaflaris@oeaw.ac.at

Éric Schost
Computer Science Department
University of Waterloo
eschost@uwaterloo.ca

Elias Tsigaridas
Sorbonne Universités, UPMC Univ Paris 06, CNRS, INRIA, Laboratoire d'Informatique de Paris 6 (LIP6), Équipe PolSys, 4 place Jussieu, 75252 Paris Cedex 05, France
elias.tsigaridas@inria.fr

ABSTRACT

We present explicit worst case degree and height bounds for the rational univariate representation of the isolated roots of polynomial systems based on mixed volume. We base our estimations on height bounds of resultants and we consider the case of 0-dimensional, positive dimensional, and parametric polynomial systems.

CCS Concepts

•**Computing methodologies** → *Symbolic calculus algorithms;*

Keywords

rational univariate representation, separation bound, sparse resultant, DMM, polynomial system

1. INTRODUCTION

The algorithms for solving polynomial systems are in the heart of algebraic algorithms. An important question in this context is how to represent the solutions of 0-dimensional systems. A common representation consists in expressing each coordinate of the solutions as a rational function evaluated at the roots of a univariate polynomial. We find this representation and its variants with many different names in literature. For example, it appears as Kronecker representation, since Kronecker initiated it, as rational parametrization, or Geometric Resolution [28], see also [32, 38], rational univariate representation (RUR) [35], see also [1], or primitive element [10].

In this representation it is essential to estimate precise bounds for both the degree and the height of the involved (univariate) polynomials; especially in the case where the input polynomials have integer coefficients. Such bounds are important both from the theoretical and the practical point of view. On the theoretical side, we use them to estimate the arithmetic and Boolean complexity of the various algorithms for solving polynomial systems. From the practical point of view they are important because they affect the realization and the performance of multi-modular algorithms.

The existing estimates on the degree and the height of the polynomial involved in the representation are based on total degree or Bézout bounds, on the height theory of varieties and on the theory of Chow forms, [1, 15, 35]. This representation is related to the arithmetic Nullstellensätz [18, 30, 39, 40], see also [33] for the most recent approach, and the separation bounds of the polynomial systems [21]. There are also dedicated estimates for the special cases of bivariate [8, 34], bilinear [23], and multi-homogeneous [36] polynomial systems. Our references represent only the tip of the iceberg of the existing ones on the subject. We encourage the reader to refer to the references of the cited bibliography.

We follow a more elementary approach. We deduce the representation of the roots using resultant computations. This allows us to deduce precise degree and height (or bitsize) of the polynomials in the representation using the maximum bitsize of the coefficients of the input polynomials and the mixed volume of the system. The latter depends on the Newton polytopes, and thus on the sparsity of the input polynomials. This characteristic makes the bounds output sensitive, and to the best of our knowledge, these are the first bounds with this property. Moreover, our bounds encapsulate and generalize all the previous ones. Because of the use of mixed volume, which is closely connected with sparse resultants, we call the representation *Sparse Rational Univariate Representation* (SRUR).

The simplicity of our approach allows us to go further. We bound the SRUR for the isolated zeros of positive dimensional polynomial systems. Furthermore, we treat an even more generic case; we estimate bounds for the SRUR of the isolated zeros of parametric polynomial systems. We also study the complexity of computing SRUR, when resultant computations for polynomial systems are available. We give an example by studying the complexity of solving tensor-structured polynomial systems. For such systems there is matrix formula for the resultant, which we use for the resultant computation that SRUR needs. Other approaches rely on Gröbner basis computations. Our algorithms are of Monte Carlo type. The computation principle of our technique applies uniformly to all cases: 0-dimensional, positive dimensional, parametric polynomial systems.

1.1 Organization of the paper

The rest of the paper is organized as follows: In the next section we present the notation that we use throughout the paper and some results that we need for mixed volume of polynomial systems. In Sec. 2 we derive the SRUR of the roots of a 0-dimensional polynomial system using resultant computations, we present an algorithm to compute it (Sec. 2.2), and we estimate explicit degree and height bounds (Sec. 2.4). In Sec. 3 we use the bounds of SRUR to deduce the bit complexity of solving tensor-product polynomial systems. Finally, in Sec. 4 we consider the SRUR for positive dimensional systems and in Sec. 5 the SRUR for parametric polynomial systems.

ISSAC '17, July 25-28, 2017, Kaiserslautern, Germany

ACM ISBN 978-1-4503-5064-8/17/07.

DOI: http://dx.doi.org/10.1145/3087604.3087653

1.2 Notation and preliminaries

$\mathcal{O}$, resp. $\mathcal{O}_B$, means bit, resp. arithmetic, complexity and $\widetilde{\mathcal{O}}_B$, resp. $\widetilde{\mathcal{O}}$, means we are ignoring logarithmic factors. For a polynomial $f \in \mathbb{Z}[x_1, \dots, x_n]$, where $n \geq 1$, $\deg(f)$ denotes its total degree, while $\deg_{x_i}(f)$ denotes its degree with respect to x_i. Moreover, $\mathtt{lc}(f)$ stands for the the leading coeffcient wrt a total degree ordering. By $\mathtt{H}(f)$ we denote the height, that is the maximum magnitude of the coefficients, of f and by $\mathfrak{h}(f)$ we denote the maximum bitsize of the coefficients of f (including a bit for the sign), i.e., the number of bits to write them as binary integers. For $a \in \mathbb{Q}$, $\mathfrak{h}(a) \geq 1$ is the maximum bitsize of the numerator and denominator. We use $[D]$ to denote the set $\{1, \dots, D\}$.

Let $n > 1$ be the number of variables. Let $\mathbf{x}^{\mathbf{a}}$ denote the monomial $x_1^{a_1} \cdots x_n^{a_n}$, with $\mathbf{a} = (a_1, \dots, a_n) \in \mathbb{Z}^n$. In the multivariate case, the input is a system of *Laurent polynomials* $f_1, \dots, f_n \in K[x_1^{\pm}, \dots, x_n^{\pm}] = K[\boldsymbol{x}, \boldsymbol{x}^{-1}]$, where $K \subset \mathbb{C}$ is the coefficient field. Since it is possible to multiply Laurent polynomials by monomials without affecting their nonzero roots, in the sequel we assume there are no negative exponents. Let the polynomials be

$$f_i = \sum\nolimits_{j=1}^{m_i} c_{i,j} \boldsymbol{x}^{\mathbf{a}_{i,j}}, \quad 1 \leq i \leq n. \tag{1}$$

Let the total degree of f_i be d_i and $d = \max_{1 \leq i \leq d} d_i$. The set $\{\mathbf{a}_{i,1}, \dots, \mathbf{a}_{i,m_i}\} \subset \mathbb{Z}^n$ is the support of f_i; the Newton polytope Q_i is the convex hull of the support. Let $\mathsf{MV}(Q_1, \dots, Q_n) > 0$ be the *mixed volume* of convex polytopes $Q_1, \dots, Q_n \subset \mathbb{R}^n$.

We consider the well-constrained polynomial system

$$(\Sigma): \; f_1(\boldsymbol{x}) = f_2(\boldsymbol{x}) = \cdots = f_n(\boldsymbol{x}) = 0, \tag{2}$$

where $f_i \in \mathbb{Z}[\boldsymbol{x}]$. In Sec. 2 we assume that the corresponding variety is zero-dimensional and does not have any positive-dimensional components even at infinity. We will drop this assumption in Sec. 4. We are interested in the system's toric roots, which lie in $(\mathbb{C}^*)^n$.

Let Q_0 be the unit standard simplex. Let $\#Q_i$ denote the number of lattice points in the closed polytope Q_i and
$\mathsf{M}_i = \mathsf{MV}(Q_0, \dots, Q_{i-1}, Q_{i+1}, \dots, Q_n)$.

Wlog, assume $\dim \sum_{i=0}^{n} Q_i = n$ and $\dim \sum_{i \in I} Q_i \geq j$ for any $I \subset \{0, \dots, n\}$ with $|I| = j$, in other words the system is essential; otherwise, its roots would be defined by a smaller system [42].

We consider the *sparse (or toric) resultant* of a system of $n + 1$ polynomial equations in n variables, assuming we have fixed the $n + 1$ supports. It provides a condition on the coefficients for the solvability of the system, and generalizes the classical resultant of n homogeneous polynomials, by taking into account the supports of the polynomials. A standard way to study a well-constrained system (Σ) through resultants is to add a linear polynomial f_0 and consider the $u-$resultant of the overconstrained system; the latter is denoted by (Σ_0). The overconstrained system has Newton polytopes $Q_0, Q_1, \dots, Q_n$. The following well-known theorem relates the number of isolated toric solutions with the mixed volume.

THEOREM 1.1. *[5, 14, 27] For $f_1, \dots f_n \in \mathbb{C}[\mathbf{x}, \mathbf{x}^{-1}]$ with Newton polytopes $Q_1, \dots, Q_n$, the number of common isolated solutions in $(\mathbb{C}^*)^n$, multiplicities counted, does not exceed $\mathsf{M}_0 = \mathsf{MV}(Q_1, \dots, Q_n)$, independently of the corresponding variety's dimension.*

Let D be the the number of distinct roots $\in (\mathbb{C}^*)^n$ of (Σ), so $D \leq \mathsf{M}_0$. For $f_i \in \mathbb{Z}[\mathbf{x}^{\pm 1}]$, let $\mathfrak{h}(f_i) = \tau_i \leq \tau$, $1 \leq i \leq n$. Let $\mathrm{vol}(\cdot)$ stand for Euclidean volume, and $\#Q_i$ for the number of lattice points in Q_i; the inequality connecting $\#Q_i$ and polytope volume in Table 1 is in [6]. Table 1 summarizes some important notation and states certain immediate properties. We provide straightforward upper bounds for the various quantities using the total degrees of the input polynomials. We can use these rough estimates to obtain simpler, albeit less accurate, bounds for $\mathtt{SRUR}$.

$$D \leq M_0 \leq \prod_{i=1}^{n} d_i \leq d^n, \quad B \leq (n-2)\binom{D}{2} \leq n \prod_{i=1}^{n} d_i^2 \leq n d^{2n},$$

$$\mathsf{M}_i \leq \prod_{\substack{1 \leq j \leq n \\ j \neq i}} d_j, \leq d^{n-1} \quad \sum\nolimits_{i=1}^{n} \mathsf{M}_i \leq n d^{n-1},$$

$$\#Q_i \leq n! \,\mathrm{vol}(Q_i) + n \leq d_i^n + n \leq 2 d_i^n,$$

$$C = \prod_{i=1}^{n} \|f_i\|_{\infty}^{M_i} \leq 2^{\tau \sum_{i=1}^{n} \mathsf{M}_i} \leq 2^{n \tau d^{n-1}},$$

$$\varrho = \prod_{i=0}^{n} (\#Q_i)^{M_i} \leq 2^{2n \sum_{i=0}^{n} \mathsf{M}_i \lg(d_i)} \leq (2d^n)^{d^n}$$

Table 1: Notation and inequalities needed for various bounds.

2. THE SRUR IN THE 0-DIM CASE

Assume that (Σ), as in Eq. (2), is a 0-dimensional polynomial system. Since our goal is to derive worst case bounds, we further assume that the system has the maximum number of roots, that is M_0. Following the technique of u-resultant, we add an equation to (Σ) to obtain the system:

$$(\Sigma_0): \; f_0(\boldsymbol{x}) = f_1(\boldsymbol{x}) = \cdots = f_n(\boldsymbol{x}) = 0, \tag{3}$$

$$\text{where} \qquad f_0 = t - g_0(\boldsymbol{x}), \tag{4}$$

t is a new variable, and $g_0 = \sum_{i=1}^{n} c_i \, x_i + \sum_j b_j \boldsymbol{x}^{\mathbf{a}_j}$ is a polynomial of degree d_0 containing a linear form in the variables x_i and it is separating; that is if $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are two different roots of the system, then $g_0(\boldsymbol{\alpha}) \neq g_0(\boldsymbol{\beta})$. We assume that g_0 has integer coefficients.

The polynomial g_0, besides the separating property, should also have some other genericity properties that help us compute a representation of the roots of the system. These properties imply some restrictions on the coefficients of g_0 and eventually determine their height. We detail these conditions in the sequel.

Let $\boldsymbol{\alpha}_i = (\alpha_{i,1}, \dots, \alpha_{i,n}) \in \mathbb{C}^n$ for $i \in [D]$, with $D \leq \mathsf{M}_0$, be the number of distinct roots of the system. We denote by $\mathtt{mult}((\Sigma), \boldsymbol{\alpha}_i)$ the multiplicity of $\boldsymbol{\alpha}_i$ as a root of the system (Σ), or the $\mathtt{mult}(\boldsymbol{\alpha}_i)$ if the system we are referring to is clear from the context. Notice that $\sum_{i=1}^{D} \mathtt{mult}(\boldsymbol{\alpha}_i) = \mathsf{M}_0$.

We consider the resultant of (Σ_0) that eliminates $\boldsymbol{x}$. Then

$$R_1 = \mathtt{res}(f_0, f_1, \dots, f_n) = \mathtt{lc}(R_1) \prod_{i=1}^{D} (t - g_0(\boldsymbol{\alpha}_i))^{\mathtt{mult}(\boldsymbol{\alpha}_i)},$$

the square-free part of which, considering it as a polynomial in t, is

$$R(t) = \mathtt{SquareFree}(R_1) = \mathtt{lc}(R) \prod\nolimits_{i=1}^{D} (t - g_0(\boldsymbol{\alpha}_i)). \tag{5}$$

For an recent and extensive discussion on the leading coefficient of R_1 we refer the reader to [19]. The derivative of R w.r.t. t is

$$R'(t) = \frac{\partial}{\partial t} R = \mathtt{lc}(R) \sum_{i=1}^{D} \prod_{j \neq i} (t - g_0(\boldsymbol{\alpha}_i)). \tag{6}$$

Now we consider a different f_0. Let $f_0(\boldsymbol{x}) = t - g_0(\boldsymbol{x}) - s \, g_1(\boldsymbol{x})$, where s is a new variable, $g_0(\boldsymbol{x})$ is the same polynomial as before, and $g_1(\boldsymbol{x})$ is a new polynomial, to be specified in the sequel. In this case, when we eliminate $\boldsymbol{x}$ from the system we obtain

$$\begin{aligned} G_1(t,s) &= \mathtt{res}(t - g_0(\boldsymbol{x}) - s\, g_1(\boldsymbol{x}), f_1, \dots, f_n) \\ &= \mathtt{lc}(G_1) \prod_{i=1}^{D} (t - g_0(\boldsymbol{\alpha}_i) - s\, g_1(\boldsymbol{\alpha}_i))^{\mathtt{mult}(\boldsymbol{\alpha}_i)}. \end{aligned}$$

The square-free part of G_1, as a polynomial in t, is

$$G_2(t,s) = \mathtt{SquareFree}(G_1) = \mathtt{lc}(G_2) \prod_{i=1}^{D} (t - g_0(\boldsymbol{\alpha}_i) - s\, g_1(\boldsymbol{\alpha}_i)).$$

We consider the derivative of G_2 with respect to s, then

$$\frac{\partial}{\partial s} G_2(t,s) = -\mathtt{lc}(G_2) \sum_{i=1}^{D} g_1(\boldsymbol{\alpha}_i) \prod_{j \neq i} (t - g_0(\boldsymbol{\alpha}_j) - s\, g_1(\boldsymbol{\alpha}_j)),$$

which by specializing $s = 0$ becomes,

$$G(t) = \frac{\partial}{\partial s} G_2(t,s)|_{s=0} = -\mathtt{lc}(G_2) \sum_{i=1}^{D} g_1(\boldsymbol{\alpha}_i) \prod_{j \neq i} (t - g_0(\boldsymbol{\alpha}_j)). \tag{7}$$

The roots of R are the evaluations $g_0(\boldsymbol{\alpha}_i) \in \mathbb{C}$ for $i \in [D]$. Moreover, for any root $\boldsymbol{\alpha} \in \mathbb{C}^n$ of (Σ) it holds

$$R'(g_0(\boldsymbol{\alpha})) = \mathtt{lc}(R) \prod_{\boldsymbol{\beta} \neq \boldsymbol{\alpha}} (g_0(\boldsymbol{\alpha}) - g_0(\boldsymbol{\beta}))$$

and $\quad G(g_0(\boldsymbol{\alpha})) = -\mathtt{lc}(G_2)\, g_1(\boldsymbol{\alpha}) \prod_{\boldsymbol{\beta} \neq \boldsymbol{\alpha}} (g_0(\boldsymbol{\alpha}) - g_0(\boldsymbol{\beta})).$

Therefore,

$$g_1(\boldsymbol{\alpha}) = -\frac{\mathtt{lc}(R)}{\mathtt{lc}(G_2)} \frac{G(g_0(\boldsymbol{\alpha}))}{R'(g_0(\boldsymbol{\alpha}))} \tag{8}$$

where $g_0(\alpha) \in \mathbb{C}$ is a root of R. Consequently, if we choose $g_1(x) = x_i$, then we can recover the i-th coordinate of every root $\boldsymbol{\alpha}$. If we make this specific choice for g_1, then we use P_i, instead of G, to make clear the coordinate we are referring to.

2.1 Height bounds

To this end we have restated known results for the parametrization of the roots, see for example [1, 15]. However, we use resultant computations and especially the Poisson formula for the resultant to express the various quantities. Therefore we can use bounds on the height of the resultant to bound the height of the polynomials that appear in the parametrization of the roots.

Following the proof of the DMM bound in [21], the resultant R_1 is a univariate polynomial in t, with coefficients homogeneous polynomials in the coefficients of the polynomials in (Σ_0):

$$R_1(t) = \cdots + \varrho_k\, t^k\, \boldsymbol{c}_{0,k}^{\mathsf{M}_0 - k} \boldsymbol{c}_{1,k}^{\mathsf{M}_1} \boldsymbol{c}_{2,k}^{\mathsf{M}_2} \cdots \boldsymbol{c}_{n,k}^{\mathsf{M}_n} + \dots, \tag{9}$$

where $\varrho_k \in \mathbb{Z}$, $\boldsymbol{c}_{j,k}^{\mathsf{M}_j}$ denotes a monomial in coefficients of f_j with total degree M_j, and $\boldsymbol{c}_{0,k}^{\mathsf{M}_0 - k}$ denotes a monomial in the coefficients of f_0 of total degree $\mathsf{M}_0 - k$. The degree of R_1, with respect to t, is M_0 and corresponds to the number of solutions of the system. It is nonzero because we have assumed that the system has only isolated solutions, even at infinity. It holds that

$$\left| \boldsymbol{c}_{1,k}^{\mathsf{M}_1}\, \boldsymbol{c}_{2,k}^{\mathsf{M}_2}\, \dots\, \boldsymbol{c}_{n,k}^{\mathsf{M}_n} \right| \leq C = \prod\nolimits_{i=1}^{n} \|f_i\|_{\infty}^{\mathsf{M}_i} = 2^{\sum_{i=1}^{n} \tau_i \mathsf{M}_i}.$$

This bound already appeared in [21], see also [17, Lemma 3.10] [33] for more recent results. Moreover, $\boldsymbol{c}_{0,k}^{\mathsf{M}_0 - k} \leq \mathtt{H}(g_0)^{\mathsf{M}_0 - k}$ and $\varrho_k \leq \varrho$. As,

$$\mathtt{H}(R_1) \leq \varrho\, \mathtt{H}(g_0)^{\mathsf{M}_0} C,$$

we deduce that $\varrho = \prod_{i=0}^{n} (\#Q_i)^{M_i} \leq (2d^n)^{d^{n-1}}$, or the more accurately $\lg(\varrho) \leq 2n \sum_{i=0}^{n} \mathsf{M}_i \lg(d_i)$, where $\#Q_i$ is the number of lattice points in the Newton polytope of the polynomial f_i; see also Table 1 and the discussion in Sec. 1.2.

As R is a divisor of R_1, using Mignotte's bound, we have

$$\mathtt{H}(R) \leq 2^{D + \lg(\mathsf{M}_0 + 1)}\, \varrho\, \mathtt{H}(g_0)^{\mathsf{M}_0} C \tag{10}$$

and consequently

$$\mathtt{H}(R') \leq 2^{D + 3 \lg \mathsf{M}_0}\, \varrho\, \mathtt{H}(g_0)^{\mathsf{M}_0} C. \tag{11}$$

Moreover, $\deg(R) = D$.

Using the same arguments we bound the coefficients of G, and so

$$\mathtt{H}(G_1) \leq \varrho\, \mathtt{H}(g_0)^{\mathsf{M}_0} \mathtt{H}(g_1)^{\mathsf{M}_0} C,$$

$$\mathtt{H}(G) \leq 2^{D + 3 \lg \mathsf{M}_0}\, \varrho\, \mathtt{H}(g_0)^{\mathsf{M}_0} \mathtt{H}(g_1)^{\mathsf{M}_0} C. \tag{12}$$

2.2 The computation of SRUR

To compute SRUR we follow [23] that generalizes the method of Canny [10]. The general idea is to use resultant computations to obtain a representation of the roots of (Σ) using the primitive element, and then to convert this representation to the one of Eq. (8). We assume that we have a black-box that computes resultants. We also assume that g_0 contains a linear form in $\boldsymbol{x}$; therefore takes the form $g_0(\boldsymbol{x}) = c_1 x_1 + \cdots c_n x_n + g_r(\boldsymbol{x})$, where g_r is polynomial of degree d. The polynomial g_r has at most $\binom{d+n}{n} - n$ monomials, and so at most that many coefficients. We denote all these coefficients by $\boldsymbol{b}$ and we write $g_r(\boldsymbol{b}, \boldsymbol{x})$.

Let $\boldsymbol{\alpha}_m = (\alpha_{m,1}, \dots, \alpha_{m,n}) \in \mathbb{C}^n$ for $m \in [D]$ be the roots of the system. Following Eq. (5), the square-free part of the resultant is a product of factors of the form $t - \zeta_m$ where

$$\zeta_m = \sum\nolimits_{i=1}^{n} c_i \alpha_{m,i} + g_r(\boldsymbol{b}, \boldsymbol{\alpha}_m). \tag{13}$$

We have to choose n constants c_i and at most $\binom{d+n}{n} - n$ constants for $\boldsymbol{b}$. To indicate this choice of constants we denote the (square-free) part of the resultant by $R(t) = R(-t, c_1, \dots, c_n, \boldsymbol{b})$. Using this notation, we have $R(t) = \mathtt{lc}(R) \prod_m (t - \zeta_m)$, where m runs over all the distinct roots of R, and

$$R'(t) = \mathtt{lc}(R) \sum\nolimits_{1 \leq m \leq D} \prod\nolimits_{1 \leq \nu \leq D, \nu \neq m} (t - \zeta_\nu). \tag{14}$$

We also need the polynomials $\hat{A}_k^{+}(t)$ and $\hat{A}_k^{-}(t)$ where

$$\hat{A}_k^{\pm}(t) = R(-t, c_1, \dots, (c_k \pm 1), \dots, c_n, \boldsymbol{b}) \tag{15}$$

for $1 \leq k \leq n$. Let $A_k^{\pm}(t) = \mathtt{square\_free}(\hat{A}_k^{\pm})$. The separating conditions on g_0 imply that the degree of $A_k^{\pm}$ is D. The roots of $A_k^{\pm}$ are $\zeta_m \pm \alpha_{m,k}$ which induces the factorization

$$A_k^{\pm}(t) = \mathtt{lc}(A_k^{\pm}) \prod_{1 \leq m \leq D} (t - \zeta_m \mp \alpha_{m,k}).$$

Keeping k fixed, in the rest of this paragraph, we will simplify our formulas by writing $r_m = \zeta_m + \alpha_{m,k}$ and $s_m = \zeta_m - \alpha_{m,k}$, for $m = 1, \dots, D$.

Given the three polynomials R, A_k^{+}, A_k^{-}, we now show how to recover a parametrization that expresses $\alpha_{m,k}$ as a rational function of ζ_m. We will do it under the following assumption: for any ℓ, m, ν in $\{1, \dots, D\}$, the equality

$$2\zeta_\ell = (\zeta_m + \alpha_{m,k}) + (\zeta_\nu - \alpha_{\nu,k}) \tag{16}$$

holds if and only if $\ell = m = \nu$ (remark that this equality holds trivially when $\ell = m = \nu$). This condition holds generically. However, it implies certain restrictions on the coefficients of g_0. We detail on these restrictions in Sec. 2.3. Our goal is to compute the polynomial

$$T_k = \sum_{1\leq m\leq D} \alpha_{m,k} \prod_{1\leq m'\leq D, m'\neq m} (t-\zeta_{m'}).$$

Indeed, then $R/\mathtt{lc}(R)$ and $(T_1,\dots,T_n)$ form the univariate representation we wish to compute. In the sequel we present an algorithm, with arithmetic cost softly linear in D^2, to compute T_k.

First, consider the polynomial S of degree D^2 defined as

$$S(t) = \prod\nolimits_{1\leq m,\nu\leq D}(t - r_m - s_\nu); \tag{17}$$

it is known as the *composed sum* of A_k^+ and A_k^-, as its roots are all the sums of a root of A_k^+ with a root of A_k^- [7]. That reference shows that given A_k^+ and A_k^-, we can compute S in $\widetilde{\mathcal{O}}(D^2)$ base field operations. We will first recall the basics of this algorithm, as we will need to elaborate on it.

For $i \geq 0$, define τ_i^+ as the ith power sum of A_k^+, that is, $\tau_i^+ = \sum_{1\leq m\leq D} r_m^i$; we define τ_i^- as the analogue for A_k^-. Writing $\mathcal{Q} = \mathbb{C}[t,\theta]/\langle A_k^+(t), A_k^-(\theta)\rangle$, for any $P \in \mathbb{C}[t,\theta]$, we define as well $\tau_{\mathcal{Q}}(P) = \sum_{1\leq m,\nu\leq D} P(r_m, s_\nu)$; this is also known as the *trace* of P in $\mathcal{Q}$. The algorithm that computes S uses one family of traces, namely $u_i = \tau_{\mathcal{Q}}((t+\theta)^i)$, for $0 \leq i \leq D^2$. To compute T_k, we will also need $v_i = \tau_{\mathcal{Q}}((t-\theta)(t+\theta)^i) = v_i' + v_i''$, with $v_i' = \tau_{\mathcal{Q}}(t(t+\theta)^i)$ and $v_i'' = \tau_{\mathcal{Q}}(\tau(t+\theta)^i)$, for $0 \leq i < D^2$. Equivalently, we have

$$u_i = \sum\nolimits_{1\leq m,\nu\leq D}(r_m + s_\nu)^i$$

$$\text{and} \qquad v_i = \sum\nolimits_{1\leq m,\nu\leq D}(r_m - s_\nu)(r_m + s_\nu)^i.$$

We start by showing how to obtain these quantities. We first compute $\tau_0^+,\dots,\tau_{D^2}^+$, and $\tau_0^-,\dots,\tau_{D^2}^-$; this takes $\widetilde{\mathcal{O}}(D^2)$ base field operations [37]. Then, following [7], we use the equality between exponential generating series in $\mathbb{Q}[[z]]$,

$$\sum_{i\geq 0} \frac{1}{i!} u_i z^i = \sum_{i\geq 0} \frac{1}{i!} \tau_i^+ z^i \cdot \sum_{i\geq 0} \frac{1}{i!} \tau_i^- z^i;$$

this allows us to compute $u_0,\dots,u_{D^2}$ for the cost of one polynomial multiplication in degree D^2, plus $\mathcal{O}(D^2)$ other operations, for a total of $\widetilde{\mathcal{O}}(D^2)$ base field operations. In order to obtain $v_0,\dots,v_{D^2-1}$, we show here how to compute $v_0',\dots,v_{D^2-1}'$; by symmetry, the same idea will apply to $v_0'',\dots,v_{D^2-1}''$, from which $v_0,\dots,v_{D^2-1}$ will follow. To compute $v_0',\dots,v_{D^2-1}'$, we now use the generating series equality

$$\sum_{i\geq 0} \frac{1}{i!} v_i' z^i = \sum_{i\geq 0} \frac{1}{i!} \tau_{i+1}^+ z^i \cdot \sum_{i\geq 0} \frac{1}{i!} \tau_i^- z^i,$$

which was already used for similar purposes in [26]. Hence, knowing $\tau_0^+,\dots,\tau_{D^2}^+$ and $\tau_0^-,\dots,\tau_{D^2}^-$, we get $v_0,\dots,v_{D^2-1}$ using $\widetilde{\mathcal{O}}(D^2)$ base field operations.

We now know $u_0,\dots,u_{D^2}$, which are the first D^2 power sums of S; as a result, S can be recovered in time $\widetilde{\mathcal{O}}(D^2)$, by a fast algorithm based on Newton's identities (see [7]). To compute T_k, we use fact that the ordinary generating series $\sum_{i\geq 0} v_i z^i$ is equal

$$\frac{\sum_{1\leq m,\nu\leq D}(r_m - s_\nu)\prod_{(m',\nu')\neq(m,\nu)}(1-(r_{m'}+s_{\nu'})z)}{\prod_{1\leq m,\nu\leq D}(1-(r_m+s_\nu)z)};$$

this follows readily from the expression giving v_i. The denominator in this expression is simply the reverse polynomial of S. Since S is known, from the first D^2 values of v_i, we can recover the numerator in the above expression by means of one polynomial multiplication in degree D^2. Taking the reverse polynomial, we then finally obtain

$$U_k = \sum_{1\leq m,\nu\leq D} (r_m - s_\nu) \prod_{(m',\nu')\neq(m,\nu)} (z - (r_{m'} + s_{\nu'})).$$

To continue, let us write the factorization of S into two terms, corresponding to the diagonal, resp. off-diagonal terms in (17): $S = S_{\text{diag}} S_{\text{off}}$, with

$$S_{\text{diag}} = \prod_{1\leq m\leq D} (z - (r_m + s_m)) = \prod_{1\leq m\leq D} (z - 2\zeta_m)$$

and $S_{\text{off}} = \prod_{1\leq m,\nu\leq D, m\neq\nu}(z-(r_m+s_\nu))$. Using the assumption in (16), we deduce that S_{diag} and S_{off} are coprime. Let as also remark that S_{diag} can be deduced from R by a simple rescaling, and S_{off} by a division.

Now, for $m \neq \nu$, the summand $(r_m - s_\nu)\prod_{(m',\nu')\neq(m,\nu)}(z - (r_{m'} + s_{\nu'}))$ in U_k admits S_{diag} as a factor. On the other hand, for $m = \nu$, $r_m - s_\nu$ is simply equal to $2\alpha_{m,k}$, and that summand becomes

$$2\alpha_{m,k}\, S_{\text{off}} \prod_{1\leq m'\leq D, m'\neq m} (z - 2\zeta_{m'}).$$

As a result, the polynomial $U_k / S_{\text{off}} \bmod S_{\text{diag}}$ is equal to

$$2 \sum_{1\leq m\leq D} \alpha_{m,k} \prod_{1\leq m'\leq D, m'\neq m} (z - 2\zeta_{m'}).$$

After rescaling, we obtain the polynomial T_k we are looking for. Knowing U_k, all these last steps take quasi-linear time $\widetilde{\mathcal{O}}(D^2)$.

All the calculation can be performed modulo a prime p, using only divisions by $1,\dots,D^2$. Thm. 2.2 provides bounds on the height of P_k. Therefore, we perform all the computation using this number of bits. The degree of the polynomials involved is $\leq D$, which is the number of roots of (Σ), if we work $\mod R(t)$.

LEMMA 2.1. *Assuming an oracle for computing resultant, the arithmetic complexity of computing* $\mathtt{SRUR}$ *is* $\widetilde{\mathcal{O}}(n\, D^2)$.

2.3 Bounds on the height of g_0

To bound the height of the polynomials in the SRUR representation of the roots of (Σ) we need to bound the height of g_0. If it is a linear form, that is, if $g_0 = \sum_{i=1}^n c_i x_i$, where c_i are suitable generic coefficients that guarantee that g_0 is a separating linear form, then $\mathtt{H}(g_0) \leq B^{(n-1)}$ [21]. However, there are cases where g_0 is not linear. For example this is the case when we solve systems of bilinear polynomials using resultant matrices [23] or tensor-product polynomial systems, see Sec. 3. We bound the height of g_0 for the general case where it is a polynomial of degree d_0. We always assume that g_0 contains a linear form in $\boldsymbol{x}$, that is $g_0(\boldsymbol{x}) = c_1 x_1 + \cdots c_n x_n + g_r(\boldsymbol{x})$, where $\deg(g_r) = d_0 > 1$. This assumption allows us to recover the coordinates of the roots (cf. Sec. 2.2).

As g_0 is of degree d_0, it has at most $\binom{d_0+n}{n} \leq (d_0+1)^n$ monomials. Let i run over all these monomials, then we can write g_0 as $g_0(\boldsymbol{x}) = \sum_i g^{(i)} \boldsymbol{x}^{\mathbf{a}_i}$. Since g_0 is a separating polynomial, then for (any) two roots, $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{C}^n$, such that $\boldsymbol{\alpha} \neq \boldsymbol{\beta}$, it holds

$$\sum\nolimits_i g^{(i)} \boldsymbol{\alpha}^{\mathbf{a}_i} \neq \sum\nolimits_i g^{(i)} \boldsymbol{\beta}^{\mathbf{a}_i} \Rightarrow \sum\nolimits_i g^{(i)} (\boldsymbol{\alpha}^{\mathbf{a}_i} - \boldsymbol{\beta}^{\mathbf{a}_i}) \neq 0$$

where $g^{(i)}$ are the coefficients of g_0. If we set $g^{(i)} = t^i$ for some variable t, then the condition becomes $\sum_i t^i(\boldsymbol{\alpha}^{\mathbf{a}_i} - \boldsymbol{\beta}^{\mathbf{a}_i}) \neq 0$. In other words a univariate polynomial in t of degree at most $(d_0+1)^n$

should not vanish. Therefore, t should not be one of its roots, there are at most $(d_0+1)^n$. Hence, we can choose as t at least one of the integers in the interval $[0\mathinner{..}(d_0+1)^n]$.

There are at most $\binom{D}{2} \leq D^2$ pairs of roots $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ of the system. Thus, we can construct at most D^2 univariate polynomials in t, each of degree at most $(d_0+1)^n$. Multiplying these polynomials results a polynomial degree $(d_0+1)^n D^2$ in t.

There are also bad values for the constants c_i and $\boldsymbol{b}$ that induce Eq. (16) to vanish for different indices m, l, and ν. To bound the number of these values we change somewhat Eq. (13) to

$$\zeta_m = \sum\nolimits_i g^{(i)} \boldsymbol{\alpha}_m^{\mathbf{a}_i} \quad , \quad 1 \leq m \leq D\,. \tag{18}$$

The polynomial on the right hand side has at most $(d_0+1)^n$ terms. We set $g^{(i)} = t^i$. In this way ζ_m, ζ_ℓ, and ζ_ν become polynomials in t and Eq. (16) becomes a polynomial in t of degree at most $(d_0+1)^n$. For each k there are $\binom{D}{3}$ possible triplets, hence, overall we obtain $\binom{D}{3} n$ polynomials in t. The product of all of them results in a polynomial of degree at most $n\,(d_0+1)^n\, D^3$.

If we choose a value of t that is not a root of this polynomial then we are sure that Eq. (16) does not vanish for different indices m, ℓ, and ν. Multiplying all the polynomials together we obtain a polynomial of degree $(d_0+1)^n D^2 + n(d_0+1)^n D^3 \leq 2n(d_0+1)^n D^3$ in t. Consequently, there is at least one integer in the interval $[0\mathinner{..}2n(d_0+1)^n D^3]$ that we can choose t from to ensure that the polynomial $\sum_i t^i \boldsymbol{x}^{\mathbf{a}_i}$ is suitable candidate for g_0.

In the way that we parametrize the roots of (Σ) there are values for the coefficients of g_0 that allow the roots of (Σ), in (some) projective space, to annihilate the resultant. This is a consequence of the Poisson formula for the resultant; we refer to [19] for details. As there exist at most M_0 isolated roots, we can assume that a polynomial of degree M_0 in t encodes these bad values. Therefore, to cope with this case as well we choose t as one integer in the interval $[0\mathinner{..}3n(d_0+1)^n D^2 \mathsf{M}_0]$. This results in the upper bound

$$\mathtt{H}(g_0) \leq (D^2 \mathsf{M}_0)^{d_0^n}\, d_0^{3n d_0^n}\,. \tag{19}$$

As $D \leq \mathsf{M}_0$ we can replace D with M_0 in the previous inequality.

2.4 Bounds on the representation

Using the bounds from Eq. (10), (11), (12), and (19) we bound the degree and the height of the polynomials in the SRUR of the roots of (Σ) using the mixed volume.

THEOREM 2.2. *There is a representation of the coordinates of roots of* (Σ)*, using an* f_0 *as in Eq.* (4)*, for* $i \in [n]$*, as*

$$x_i = P_i(\theta)/R'(\theta), \text{ where } \theta \text{ is such that } R(\theta) = 0.$$

The univariate polynomials R*,* R'*, and* P_i *are as in Eq.* (5), (6), (7)*, respectively, and* P_i *indicates that we choose* $g_1(\boldsymbol{x}) = x_i$*. Their degrees are* $\deg(R), \deg(R'), \deg(P_i) \leq D$*, and their bitsizes are bounded as*

$$\mathfrak{h}(R) \leq D + \lg(\mathsf{M}_0+1) + 3n d_0^n \mathsf{M}_0 \lg(d_0^n \mathsf{M}_0) + \lg(\varrho\, C),$$

$$\mathfrak{h}(R'), \mathfrak{h}(P_i) \leq D + 3\lg(\mathsf{M}_0) + 3n d_0^n \mathsf{M}_0 \lg(d_0^n \mathsf{M}_0) + \lg(\varrho\, C).$$

If g_0 is a linear form and the bitsize of all the polynomials f_i is bounded by τ, using the bounds from DMM [21] we obtain

$$\mathfrak{h}(P_i) \leq D + 6(1+\mathsf{M}_0)\lg(\mathsf{M}_0) + (\tau + 2n)\sum_{i=1}^{n} \mathsf{M}_i \lg(2\,\#Q_i). \tag{20}$$

Assuming that we have an oracle that performs resultant computations, we need to use it $2n+1$ times to compute R and $\hat{A}_k^{\pm}$. Then we perform $2n$ square-free and GCD computations and finally within $\widetilde{\mathcal{O}}(D^2)$ arithmetic operations we compute SRUR. In total we perform $\widetilde{\mathcal{O}}(n\mathsf{M}_0^2 + nD^2)$ arithmetic operations.

From Thm. 2.2 we deduce that we need to compute with integers of bitsize $\eta = \widetilde{\mathcal{O}}(\mathsf{M}_0 d_0^n \lg(\mathsf{M}_0 d_0^n) + \sum_{i=1}^n (2n \lg \#Q_i + \tau_i)\mathsf{M}_i)$. Therefore the total bit complexity is $\widetilde{\mathcal{O}}_B((n\mathsf{M}_0^2 + nD^2)\eta)$. Using the inequality $D \leq \mathsf{M}_0$ we have the following:

LEMMA 2.3. *Assuming that there is an oracle for resultant computations, we can compute the* SRUR *of a 0-dim system in*

$$\widetilde{\mathcal{O}}_B(n\mathsf{M}_0^3 d_0^n \lg(\mathsf{M}_0 d_0^n) + n\mathsf{M}_0^2 \sum\nolimits_{i=1}^n (n \lg \#Q_i + \tau_i)\mathsf{M}_i).$$

3. USING RESULTANT MATRICES AND SRUR

The bounds on SRUR of Thm. 2.2 lead directly to an algorithm and complexity bounds for solving polynomial systems. In this section we assume that there exists a matrix, say M, of size $E \times E$ which we construct using the coefficients of the polynomials in (Σ_0) such that R_1 is a multiple of its determinant. Usually, $c \cdot R_1 = \det(M)$, where c is a constant; for example this is the case for the classical Macaulay resultant, the sparse resultant, etc. However, there are many cases where we have an exact determinantal representation for the resultant, for example the bilinear case [23], or the tensor-product polynomial systems that we consider next.

Under genericity assumptions that guarantee $c \neq 0$, we can compute R_1, and thus R, using determinant computations. The complexity is $\widetilde{\mathcal{O}}_B(E^3 \mathsf{M}_0 \tau)$, if we assume that the bitsize of the input polynomials is bounded by τ. For example, for sparse resultants E is the number of lattice points of the Minkowski sum of the Newton polytopes of the input polynomials. We can also opt for a complexity of $\widetilde{\mathcal{O}}_B(E^{\omega} \mathsf{M}_0 \tau)$, if we use the techniques for fast determinant computation [41], where ω is the exponent of matrix multiplication. We also refer to [9] for a recent result on the bit complexity of solving polynomial systems that exploits this computation. For sparse resultant matrix constructions, E stands for the number of lattice points of the polytope that is the Minkowski sum of the Newton polytopes of the input polynomials, e.g. [12]. We can bound E using the volume of the Newton polytopes.

Tensor-product systems

As a concrete example we consider the complexity of solving tensor-product polynomial systems. This interesting class of systems admits a determinantal formula for their resultant. Their polynomials are members of a tensor-product space of univariate polynomial rings. In particular, every polynomial in Eq. (3) contains all monomials up to a given degree $d_i \in \mathbb{N}$, with respect to the variable x_i. Therefore, the (common) support of all equations is characterized by the tuple $(d_1, \ldots, d_n) \in \mathbb{N}^n$ which corresponds to a Newton polytope that is an n-hypercube. To obtain a non-trivial resultant matrix with certainty, we assume that the monomials corresponding to the vertices of this hypercube (that is, the “extreme monomials”) are non-zero. In particular for f_0, which is going to play the role of the separating form, we assume non-zero coefficients for the extreme monomials, these correspond to the monomials of $g_0(\boldsymbol{x})$, as well as for the linear terms, cf. (4).

For any $0 \leq i \leq n$, the mixed volume M_i of the system $(\Sigma_0) \setminus \{f_i\}$, that is the number of solutions in $\left(\mathbb{P}^1\right)^{\times n}$, equals

$$\mathsf{M} := \mathsf{M}_0 = n! \prod\nolimits_{i=1}^n d_i.$$

The resultant, R, is a polynomial of total degree $\deg R = (n+1)\mathsf{M}$ in the coefficients of (Σ_0). Furthermore, R is homogeneous of degree exactly M with respect to the coefficients of each f_j, $j =$

$0,\dots,n$. To obtain the resultant matrix we consider degree vectors $(m_1,\dots,m_n)\in\mathbb{N}^n$ that satisfy

$$\prod_{i=1}^{n}(m_i+1)=(n+1)\mathsf{M} \text{ and } \prod_{i=1}^{n}(m_i-d_i+1)=\mathsf{M}\,. \quad (21)$$

Each solution of the Diophantine equation (21) gives rise to a Sylvester-type matrix, the determinant of which expresses the resultant of the system. Solutions to the above equations first appeared in [43], see also [22].

EXAMPLE 3.1. *Let $n=3$ and $(d_1,d_2,d_3)=(1,2,3)$. The mixed volume is $\mathsf{M}=36$. We check that $(m_1,m_2,m_3)=(3,5,5)$ satisfies (21). Therefore the resultant is given as the determinant of a 144×144 matrix. This matrix expresses the Sylvester map*

$$S\,:\,\mathcal{P}(2,3,2)^4\to\mathcal{P}(3,5,5)$$

$$\textit{with} \qquad S(g_0,g_1,g_2,g_3)=\sum\nolimits_{j=0}^{3} g_j f_j.$$

Here $\mathcal{P}(m_1,m_2,m_3)$ stands for the space of tensor-product polynomials of multidegree (m_1,m_2,m_3). We can check that the above map is generically surjective. Moreover, $\dim\mathcal{P}(2,3,2)=\mathsf{M}$ and $\dim\mathcal{P}(3,5,5)=\deg R=4\mathsf{M}$.

From the above example it is clear that the matrix is square of dimension $(n+1)\mathsf{M}\times(n+1)\mathsf{M}$. It consists of $n+1$ blocks; each has size $\mathsf{M}\times(n+1)\mathsf{M}$ and it is quasi-Toeplitz, i.e. it expresses the multiplication by a polynomial f_j of Σ_0. Under our assumption that the coefficients of the monomials $1, x_i^{d_i}, x_i^{d_i}x_j^{d_j},\dots,x_1^{d_1}\cdots x_n^{d_n}$ are non-zero in all polynomials, the above Sylvester-type matrix is generically non-singular. Moreover, it is always equal to the (sparse) resultant of the system, that is, it vanishes whenever the system has a common root in the toric variety $\left(\mathbb{P}^1\right)^{\times n}$. The bounds of Thm. 2.2 are tight for this class of systems.

We can compute the determinant of such matrices in $\widetilde{\mathcal{O}}(n\mathsf{M}^2)$ arithmetic operations by exploiting the algorithms for multivariate polynomial multiplication using Wiedemann's method e.g. [13, 23]. Taking into account the height bounds of SRUR, the bit complexity for computing the R is $\widetilde{\mathcal{O}}_B(n^2\mathsf{M}^4\tau)$. Alternatively, we can use [41], see also [9], to compute the determinant in $\widetilde{\mathcal{O}}_B((n\mathsf{M})^{\omega+1}\tau)$.

If we assume that $\mathfrak{h}(f_i)\le\tau$ for $i\in[n]$, and we notice that $\mathsf{M}_i=\mathsf{M}$, then following Thm. 2.3 we compute the SRUR in $\widetilde{\mathcal{O}}_B(n^2\mathsf{M}^3(d^n\lg(\mathsf{M}d^n)+n^3+n\tau))$, where $d=\max_i d_i$. The SRUR representation involves polynomials of maximum coefficient bitsize $\widetilde{\mathcal{O}}(\mathsf{M}(d^n\lg(\mathsf{M}d^n)+n^3+n\tau))$.

At this point we have computed the SRUR representation of zeros. If we also want to obtain approximations of the actual coordinates, then we have to approximate the roots of $R(\theta)$, refine them up to the separation bound of the system, and perform interval arithmetic with the rational function until we obtain disjoint hyperboxes; one for each zero of the system. We refer the reader to [23] for a more detailed description of this method.

4. POSITIVE DIMENSIONAL CASE

Assume that (Σ_0) has solutions at infinity, or it is positive dimensional, or that the coefficients of the polynomials are not generic enough and they cause the determinant of the resultant matrix to be identically zero. How do we compute the SRUR for the isolated roots in these cases? We use the technique of the toric generalized characteristic polynomial [11, 16]. The idea is to perturb the initial system, using a new parameter, say s, and, as in the previous section, we add a separating polynomial f_0 that introduces a new variable t. In the worst case we perturb all the coefficients of all the polynomials. The resultant of the system, say Q, is a bivariate polynomial in s and t. To recover the SRUR of the isolated roots of the system we need to obtain the polynomial corresponding to the first non-vanishing coefficient of Q, say k, considered as a polynomial in s. This is a univariate polynomial in t and is the R_1 polynomial of the previous section. Now R_1 encodes the isolated roots of the system, as well as points from the irreducible components of the variety defined by the zero set of the polynomials f_i.

More formally, we perturb (Σ_0) in (3) and we obtain:

$$(\widetilde{\Sigma}_0) \quad \left\{ \begin{array}{l} \widetilde{f}_0=f_0=0, \\ \widetilde{f}_i=f_i+p_i=0, \quad 1\le i\le n, \end{array}\right.$$

where $p_i=\sum_{\mathbf{a}\in\mathcal{D}_i}s^{\omega_i(\mathbf{a})}\mathbf{x}^{\mathbf{a}}$, $\omega_i(\cdot)$ are positive-valued linear forms, s a new parameter, and $\mathcal{D}_i$ is the subset of vertices in Q_i. In the worst case, $\mathcal{D}_i$ contains all vertices of Q_i. The perturbation does not alter the Newton polytopes of the f_i's. The correctness of the approach is due to the following theorem from [16, Theorem 3.6], see also [11, Theorem 3.2].

THEOREM 4.1. *Consider $f_i\in\mathbb{C}[t,u_1,\dots,u_\nu][\boldsymbol{x}]$ where $i\in[n+1]$ and their zero set $V=\mathbb{V}(f_0,f_1,\dots,f_n)\subset\mathbb{C}^m\times\mathcal{T}$ where $\mathcal{T}$ is the toric variety associated to the Newton polytopes of f_i, and let W be a proper component of V of dimension ν. Let $C(t,\boldsymbol{u})(s)$ be the generalized toric characteristic polynomial of the f_i, as polynomials in the x_j. Arranging the polynomial in powers of s, let $C_k(t,\boldsymbol{u})$ be its coefficient of lowest degree. If $\pi_t:\mathbb{C}\times\mathbb{C}^\nu\times\mathcal{T}\to\mathbb{C}\times\mathbb{C}^\nu$ denotes the projection on the $(t,\boldsymbol{u})$-coordinates, then $C_k(\pi_t(q))=0$ for all $q\in W$.*

The (terms of the) resultant of the system is as in Eq. (9). To obtain the bounds, we consider the worst case scenario where all the coefficients are perturbed. We assume that the coefficients of all the polynomials in $(\widetilde{\Sigma}_0)$ are integers of maximum bitsize τ, and the degree of all the polynomials in bounded by d. We need the following result on multivariate polynomial multiplication.

CLAIM 4.2 (POLYNOMIAL MULTIPLICATION). *Consider two multivariate polynomials, f_1 and f_2, in ν variables of total degrees δ and bitsize τ_1 and τ_2 respectively. Then their product is of degree 2δ and bitsize $\tau_1+\tau_2+2\,\nu\,\lg(\delta)$.*

We can prove using induction that the product of m polynomials, $\prod_{i=1}^m f_i$ of degree δ_i, results in a polynomial of total degree $\sum_{i=1}^m\delta_i$ and bitsize $\sum_{i=1}^m\tau_i+12\,\nu\,m\,\lg(m)\,\lg(\sum_{i=1}^m\delta_i)$. If we are interested in the m-th power of a polynomial, then a somewhat better bound on the bitsize could be computed, that is $m\tau+12\nu m\lg(\delta)$

It is an overestimation to assume that every factor of each term of the resultant of $(\widetilde{\Sigma}_0)$, see Eq. (9), corresponding to the polynomial $\widetilde{f}_i$ is of the form $(c_{i,j}+s)^{\mathsf{M}_i}$, where $c_{i,j}$ is a coefficient of f_i having the biggest magnitude. Then $\mathfrak{h}((c_{i,j}+s)^{\mathsf{M}_i})\le(\tau+12)\mathsf{M}_i$. The bitsize of the factor corresponding to all polynomials f_i is bounded by $(\tau+12)\sum_{i=0}^n\mathsf{M}_i+12(n+1)\lg(n+1)\lg\sum_{i=0}^n\mathsf{M}_i$. Recall that

$$\lg(\varrho)\le 2n\sum_{i=0}^{n}\mathsf{M}_i\lg(d_i)\le 2n\lg d\sum_{i=0}^{n}\mathsf{M}_i,$$

We have $\deg(R_1)\le\mathsf{M}_0$ and $\mathfrak{h}(R_1)\le\eta$, where

$$\eta=(\tau+2n\lg d+12)\sum_{i=0}^{n}\mathsf{M}_i+12(n+1)\lg(n+1)\lg\sum_{i=0}^{n}\mathsf{M}_i.$$

This leads to the bounds $\deg(R)\le\mathsf{M}_0$, $\mathfrak{h}(R)\le\eta+\mathsf{M}_0+\lg\mathsf{M}_0+1$, and $\mathfrak{h}(R')$, $\mathfrak{h}(P_i)\le\eta+\mathsf{M}_0+2\lg\mathsf{M}_0+1$.

Theorem 4.3. *Let (Σ) be an non-necessarily 0-dimensional, polynomial system with polynomials of maximum degree d and maximum coefficient bitsize τ. There is a representation of the isolated roots using an f_0 as in Eq. (4), for $i \in [n]$, as*

$$x_i = P_i(\theta)/R'(\theta), \text{ where } \theta \text{ is such that } R(\theta) = 0,$$

where $\deg(R), \deg(R'), \deg(P_i) \leq \mathsf{M}_0$, and their bitsizes are bounded as $\mathfrak{h}(R) \leq \eta + \mathsf{M}_0 + \lg \mathsf{M}_0 + 1$, $\mathfrak{h}(R'), \mathfrak{h}(P_i) \leq \eta + \mathsf{M}_0 + 2\lg \mathsf{M}_0 + 1$, and $\eta = (\tau + 2n\lg d + 12)\sum_{i=0}^{n} \mathsf{M}_i + 12(n+1)\lg(n+1)\lg\sum_{i=0}^{n}\mathsf{M}_i$.

If we assume an oracle for performing resultant computations, then we can compute SRUR, even in the positive dimensional case in $\widetilde{\mathcal{O}}(n\mathsf{M}_0^2)$ arithmetic operations, using the algorithm of Sec. 2.2 that supports Lem. 2.1. Therefore, the bit complexity is

$$\widetilde{\mathcal{O}}_B(n\mathsf{M}_0^2(n+\tau)\sum\nolimits_{i=0}^{n}\mathsf{M}_i).$$

Regarding g_0 the bounds of Eq. (19) hold in this case as well. Therefore, if $\mathfrak{h}(f_i) \leq \tau$ for $i \in [n]$, then it suffices to replace, for the purpose of estimating the bit complexity, τ with $\tau + d_0^n \lg(\mathsf{M}_0)$. To summarize, we have the following lemma

Lemma 4.4. *Assuming that there is an oracle to perform resultant computations, we can compute the SRUR of the isolated roots of a, not necessarily 0-dimensional, polynomial system in $\widetilde{\mathcal{O}}_B(n\mathsf{M}_0^2(n+\tau+d_0^n\lg(\mathsf{M}_0))\sum_{i=0}^{n}\mathsf{M}_i)$.*

We should note that the oracle supporting Lem. 4.4 is more powerful than the one that supports Lem. 2.3.

We present a straightforward algorithm for computing the resultant R_1, when there is a determinantal representation of the resultant of the system of size $n^c\mathsf{M}_0 \times n^c\mathsf{M}_0$, where c is a (small) constant. The determinant of the resultant matrix is a bivariate polynomial in s and t. It has M_0^2 terms, thus using interpolation we can recover it in $\widetilde{\mathcal{O}}_B(n^{2c}\mathsf{M}_0^4\eta)$. This bound is roughly $\widetilde{\mathcal{O}}_B(n^{2c}\mathsf{M}_0^5\tau)$, under the assumption on the size of the matrix. If this is not the case, that is, if no exact resultant matrix is available, then we can always use the sparse resultant matrix or its variants, but the bounds become more cumbersome as they involve the number of the lattice points in the Minkowski sum of the Newton polytopes. The complexity bound of computing the resultant dominates the bound of Lem. 4.4 and determines the complexity of the whole solving procedure. We refer the reader to [24, 25] for some further details.

We would have liked to have a dedicated algorithm to compute the determinant of a matrix with polynomial entries $\mod s^k$ with complexity proportional to $\lg k$. There is an efficient randomized procedure to estimate the value of k, that is the index of the first non-vanishing coefficient of Q, when we consider it as a univariate polynomial in s. Assuming that there is an oracle that answers whether a specific value of k is the correct one, we use exponential search. We test if k is $0, 2, 4, \ldots$, and after, if we identify a suitable interval, we perform binary search in this interval. We need to perform this test $\mathcal{O}(\lg k)$ times. As soon as we know the value k, we can compute the determinant of the resultant matrix $\mod s^{k+1}$. It remains to provide a realization for the oracle. We notice that for $l \leq k$, it holds $Q \mod s^l = 0$. Therefore, if we specialize $s = p$ for some random values p, then $Q = 0 \mod p^l$ for $l \leq k$ and $Q \neq 0 \mod p^l$ for $l > k$.

To find suitable values p we follow [29] that provides optimal certificates for linear algebra operations and we proceed as follows: The resultant has coefficients of bitsize 2^η. There are at most $\mathcal{O}(\lg\eta)$ primes that divide each of the coefficients and there are at most $\binom{\mathsf{M}_0}{2} \leq \mathsf{M}_0^2$ coefficients. If we choose a prime p uniformly at random from a set of $\mathcal{O}(m\mathsf{M}_0^2\lg(\eta))$ primes, then with probability $1 - 1/m$ the prime p does not divide any of the coefficients of the resultant.

Therefore, we obtain the correct value of k in $\widetilde{\mathcal{O}}_B(\mathsf{M}_0^3\tau)$. Whether we can obtain R within the same complexity bound or in $\widetilde{\mathcal{O}}_B(\mathsf{M}_0^4\tau)$ is an interesting open problem. In this direction, one should also exploit the recent results on linear algebra certificates [20].

5. PARAMETRIC SYSTEMS

The power of exploiting resultant computations for computing height bounds for the SRUR of the isolated roots is that we can consider coefficients in any field. For example we can consider polynomials with coefficients other multivariate polynomials. This allows us to obtain precise degree and height bounds for the SRUR of parametric polynomial systems. We refer the reader to [2, 4] for the details of a complete algorithm that exploits such an approach. For an algorithm based on geometric resolution we refer to [38].

In this section we assume that the coefficients of the polynomials in (Σ) are in $\mathbb{Z}[u_1, \ldots, u_\nu] = \mathbb{Z}[\boldsymbol{u}]$, where $u_1, \ldots, u_\nu$ are parameters. First, we assume that the system is 0-dimensional for (almost) any specialization of the parameters and does not have any solutions at infinity. If this is not the case we can resort to Sec. 4. We add a suitable polynomial f_0 to the system and we consider the Macaulay matrix, which we denote by M.

There is an algorithm that produces a constructible set A such that the rank of M is maximal for any specialization of the parameters taken form the set A. The constructible set is defined by polynomials in $\mathbb{Z}[\boldsymbol{u}]$ of degree at most $N\delta$ and maximum coefficient bitsize $N\tau$ where N is the size of the Macaulay matrix [2, 3, 4]. The theory behind this construction is due to Lazard [31, Theorem 4.1, 5.1, and 7.1]; he proved that the system is 0-dimensional if the Macaulay matrix is of full rank. Then we apply a parametric version of Gauss algorithm, e.g. [2, Sec. 2.4.1], to compute constructible sets of interest. The complexity of these algorithms is singly exponential with respect to the number of variables, but we do not elaborate further. We emphasize that we are not focusing on algorithm(s) to compute these constructible sets, but rather on computing explicit bounds on the degree and the height of the involved polynomials. It is enough for our purposes to assume that there exists an algorithm that constructs these sets. If the parameters lie in a connected component of these constructible sets, then the representation of the roots does not change. For each such constructible set the resultant of the system $R_1(t, \mathbf{u})$ is a homogeneous polynomial in the coefficients of the input polynomials and its terms are as in Eq. (9).

We consider R_1 as a univariate polynomial in t. To do so we consider the expansion of each of the terms of R_1 as univariate polynomials in t. We consider the partition of each term to $n+1$ factors, as Eq. (9) indicates, and we bound each factor separately.

We assume that the maximum bitsize of the coefficients in the polynomials f_i is τ, that is $\mathfrak{h}(f_i) \leq \tau$ for $i \in [n]$. Following Eq. (9) each $\boldsymbol{c}_{i,k}^{\mathsf{M}_i}$ corresponds to products of polynomials in $\mathbb{Z}[\boldsymbol{u}]$ of total degree M_i. To derive an upper bound we consider the worst case where $\mathbf{c}_{i,k}^{\mathsf{M}_i}$ corresponds to the M_i-th power of a polynomial in $\mathbb{Z}[\boldsymbol{u}]$; the latter has total degree δ and maximum coefficient bitsize τ. From Claim. 4.2 the expansion results in a polynomial in $\mathbb{Z}[\boldsymbol{u}]$ of total degree $\delta\,\mathsf{M}_i$ and bitsize $\mathsf{M}_i\tau + 12\nu\mathsf{M}_i\lg(\delta)$. Moreover $\boldsymbol{c}_{0,k}^{\mathsf{M}_0}$ is an integer of bitsize $\leq \mathfrak{h}(g_0)^{\mathsf{M}_0}$. Hence, each coefficient of t in R_1 is a polynomial in $\mathbb{Z}[\boldsymbol{u}]$ of degree $\delta\sum_{i=0}\mathsf{M}_i$ and it has bitsize bounded by

$$(\tau+12\nu\lg(\delta))\sum_{i=1}^{n}\mathsf{M}_i+12\nu n\lg(n)\lg(\delta\sum_{i=1}^{n}\mathsf{M}_i)+\mathsf{M}_0\mathfrak{h}(g_0)+\mathfrak{h}(\varrho).$$

Moreover,

$$\lg(\varrho C) + \mathsf{M}_0\mathfrak{h}(g_0) + 12\nu(\lg(\delta)\sum_{i=1}^{n} \mathsf{M}_i + \lg(n)\lg(\delta\sum_{i=1}^{n} \mathsf{M}_i))$$

and so

$$\begin{aligned}\mathfrak{h}(R) \leq \quad & \mathsf{M}_0 + \lg(\mathsf{M}_0+1) + \lg(\varrho C) + \mathsf{M}_0\mathfrak{h}(g_0) + \\ & 12\nu(\lg(\delta)\textstyle\sum_{i=1}^{n} \mathsf{M}_i + \lg(n)\lg(\delta\sum_{i=1}^{n} \mathsf{M}_i)).\end{aligned}$$

where $\mathfrak{h}(\varrho) \leq 2n\sum_{i=0}^{n} \mathsf{M}_i \lg(d_i)$ accounts for the lattice points in the corresponding Newton polytopes.

Using the height bounds on the resultant we can modify accordingly the approach of Sec. 2 to deduce bounds for the SRUR of parametric systems. We arrive at the following theorem

THEOREM 5.1. *There is a representation of the coordinates of the isolated zeros of the parametric polynomial system* (Σ)*, using an* f_0 *as in Eq.* (4)*, for* $i \in [n]$*, as*

$$x_i = \frac{P_i(\theta, \boldsymbol{u})}{R'(\theta, \boldsymbol{u})}, \text{ where } \theta \text{ is such that } R(\theta, \boldsymbol{u}) = 0,$$

where $\deg(R), \deg(P_i) \leq \mathsf{M}_0$ *and*

$$\begin{aligned}\mathfrak{h}(R), \mathfrak{h}(P_i) \leq \quad & \mathsf{M}_0 + 3\lg(\mathsf{M}_0) + \lg(\varrho C) + \mathsf{M}_0\mathfrak{h}(g_0) + \\ & 12\nu(\lg(\delta)\sum_{i=1}^{n} \mathsf{M}_i + \lg(n)\lg(\delta\sum_{i=1}^{n} \mathsf{M}_i)).\end{aligned}$$

We can combine the bounds of Thm. 5.1 with Thm. 4.1 to obtain bounds on the representation of the isolated roots even when the system becomes positive dimensional for certain values of the parameters. It suffices to perturb symbolically all the coefficients. The asymptotic behavior of the bounds remains the same. Following the approach of Sec. 2.2, see also Lem. 2.1, under the assumption that the corresponding resultants are provided, we can compute SRUR in $\widetilde{\mathcal{O}}(n\,\mathsf{M}_0^2)$ multiplications of multivariate polynomials in ν variables and degree $\delta\,\mathsf{M}_i$. The precise bit complexity bounds are rather cumbersome and we omit their presentation.

The most general case is to consider systems with polynomials in $(\mathbb{Z}[\alpha_1, \dots, \alpha_\mu][u_1, \dots, u_\nu])[x_1, \dots, x_n]$, where $u_1, \dots, u_\nu$ are parameters and α_i is an algebraic number of degree m_i, for $i \in [\mu]$.

Acknowledgments. The authors are grateful to the reviewers for their comments. ET is partially supported by HPAC (ANR ANR-11-BS02-013) and an FP7 Marie Curie Career Integration Grant.

References

[1] M. E. Alonso, E. Becker, M.-F. Roy, and T. Wörmann. Algorithms in algebraic geometry and applications. chapter Zeros, Multiplicities, and Idempotents for Zero-dimensional Systems, pages 1–15. 1996.

[2] A. Ayad. *Complexity of solving parametric polynomial systems*. PhD thesis, IRMAR, Rennes, 2006.

[3] A. Ayad. Complexity of solving parametric polynomial systems. *J. Math. Sc.*, 179(6):635–661, Dec. 2011.

[4] A. Ayad, A. Fares, and Y. Ayyad. An algorithm for solving zero-dimensional parametric systems of polynomial homogeneous equations. *J. Nonlinear Sciences Appl.*, 5(6):426—438, 2012.

[5] D. Bernstein. The number of roots of a system of equations. *Funct. Anal. and Appl.*, 9(2):183–185, 1975.

[6] H. F. Blichfeldt. A new principle in the geometry of numbers, with some applications. *Trans. AMS*, 15(3):227–235, 1914.

[7] A. Bostan, P. Flajolet, B. Salvy, and É. Schost. Fast computation of special resultants. *Journal of Symbolic Computation*, 41(1):1–29, 2006.

[8] Y. Bouzidi, S. Lazard, G. Moroz, M. Pouget, F. Rouillier, and M. Sagraloff. Improved algorithms for solving bivariate systems via Rational Univariate Representations. Tech. report, Inria, June 2015.

[9] C. Brand and M. Sagraloff. On the complexity of solving zero-dimensional polynomial systems via projection. In *Proc. of the ACM on International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 151–158. ACM, 2016.

[10] J. Canny. Some algebraic and geometric computations in PSPACE. In *Proc. 20th STOC*, pages 460–467. ACM, 1988.

[11] J. Canny. Generalised characteristic polynomials. *JSC*, 9(3):241–250, 1990.

[12] J. Canny and I. Emiris. A subdivision-based algorithm for the sparse resultant. *J. ACM*, 47(3):417–451, May 2000.

[13] J. F. Canny, E. Kaltofen, and L. Yagati. Solving systems of nonlinear polynomial equations faster. In *Proc. ISSAC*, pages 121–128, 1989.

[14] D. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Number 185 in GTM. Springer, New York, 2nd edition, 2005.

[15] X. Dahan and E. Schost. Sharp estimates for triangular sets. In *Proc. ACM ISSAC*, pages 103–110, 2004.

[16] C. D'Andrea and I. Emiris. Computing sparse projection operators. *Contemporary Mathematics*, 286:121–140, 2001.

[17] C. D'Andrea, A. Galligo, and M. Sombra. Quantitative Equidistribution for the Solutions of Systems of Sparse Polynomial Equations. *American Journal of Mathematics*, 136:1543–1579, 2014.

[18] C. D'Andrea, T. Krick, and M. Sombra. Heights of varieties in multiprojective spaces and arithmetic nullstellensätze. *Annales scientifiques de l'École Normale Supérieure*, 46(4):549–627, 2013.

[19] C. D'Andrea and M. Sombra. A Poisson formula for the sparse resultant. *Proceedings of the London Mathematical Society*, page pdu069, 2015.

[20] J.-G. Dumas, E. Kaltofen, E. Thomé, and G. Villard. Linear time interactive certificates for the minimal polynomial and the determinant of a sparse matrix. In *Proc. Int'l Symposium on Symbolic and Algebraic Computation*, ISSAC, pages 199–206. ACM, 2016.

[21] I. Emiris, B. Mourrain, and E. Tsigaridas. The DMM bound: Multivariate (aggregate) separation bounds. In S. Watt, editor, *Proc. 35th ACM Int'l Symp. on Symbolic & Algebraic Comp. (ISSAC)*, pages 243–250, Germany, 2010.

[22] I. Z. Emiris and A. Mantzaflaris. Multihomogeneous resultant formulae for systems with scaled support. *J. of Symbolic Computation*, 47(7):820–842, 2012.

[23] I. Z. Emiris, A. Mantzaflaris, and E. Tsigaridas. On the bit complexity of solving bilinear polynomial systems. In *Proc. of the ACM on International Symposium on Symbolic and Algebraic Computation*, ISSAC, pages 215–222, New York, NY, USA, 2016. ACM.

[24] I. Z. Emiris and V. Y. Pan. Symbolic and Numeric Methods for Exploiting Structure in Constructing Resultant Matrices. *JSC*, 33(4):393–413, Apr. 2002.

[25] I. Z. Emiris and V. Y. Pan. Improved algorithms for computing determinants and resultants. *J. of Complexity*, 21(1):43–71, Feb. 2005.

[26] L. D. Feo, J. Doliskani, and É. Schost. Fast arithmetic for the algebraic closure of finite fields. In *ISSAC'14*, pages 122–129, 2014.

[27] I. Gelfand, M. Kapranov, and A. Zelevinsky. *Discriminants, Resultants and Multidimensional Determinants*. Boston, Birkhäuser, 1994.

[28] M. Giusti, G. Lecerf, and B. Salvy. A Gröbner free alternative for polynomial system solving. *J. of Complexity*, 17(1):154–211, 2001.

[29] E. L. Kaltofen, M. Nehring, and B. D. Saunders. Quadratic-time certificates in linear algebra. In *Proc. 36th Int'l Symposium on Symbolic and Algebraic Computation*, ISSAC, pages 171–176, 2011.

[30] T. Krick, L. Pardo, and M. Sombra. Sharp estimates for the arithmetic Nullstellensatz. *Duke Math. J.*, 109(3):521–598, 2001.

[31] D. Lazard. Algèbre linéaire sur $k[x_1, \dots, x_n]$ et elimination. *Bull. Soc. Math. France*, 105(2):165–190, 1977.

[32] G. Lecerf. *Une alternative aux méthodes de réécriture pour la résolution des systèmes algébriques*. PhD thesis, École polytechnique, France, 2001.

[33] C. Martìnez and M. Sombra. An arithmetic Bernstein-Kusnirenko inequality. (submitted), 2016.

[34] E. Mehrabi and E. Schost. A softly optimal monte carlo algorithm for solving bivariate polynomial systems over the integers. *J. of Complexity*, 34:78 – 128, 2016.

[35] F. Rouillier. Solving zero-dimensional systems through the rational univariate representation. *J. of Appl. Algebra in Engin., Comm. and Computing*, 9(5):433–461, 1999.

[36] M. Safey El Din and É. Schost. Bit complexity for multi-homogeneous polynomial system solving application to polynomial minimization. *arXiv preprint arXiv:1605.07433*, 2016.

[37] A. Schönhage. The fundamental theorem of algebra in terms of computational complexity. Technical report, Univ. Tübingen, 1982.

[38] É. Schost. Computing parametric geometric resolutions. *Appl. Alg. in Eng., Comm. and Comp.*, 13(5):349–393, Feb. 2003.

[39] M. Sombra. *Estimaciones para el teorema de ceros de Hilbert*. PhD thesis, Universidad de Buenos Aires, 1998.

[40] M. Sombra. The height of the mixed sparse resultant. *Amer. J. Math.*, 126:1253–1260, 2004.

[41] A. Storjohann. The shifted number system for fast linear algebra on integer matrices. *Journal of Complexity*, 21(4):609–650, 2005.

[42] B. Sturmfels. On the newton polytope of the resultant. *Journal of Algebraic Combinatorics*, 3(2):207–236, 1994.

[43] B. Sturmfels and A. Zelevinsky. Multigraded resultants of Sylvester type. *Journal of Algebra*, 163(1):115–127, 1994.