Fast Algorithms for Polynomial Systems Solving

Alin Bostan

Gage, École polytechnique (France)

November 19, 2001

Summary by Frédéric Chyzak

Abstract

Solving a system of polynomial equations with a finite number of solutions can be reduced to linear algebra manipulations in an algebra A of finite type. We show how to accelerate this linear algebra phase in order to compute a "rational parameterization" of the zeros of the polynomial system. We propose new algorithmic solutions by extending ideas introduced by V. Shoup in the context of the factorization of polynomials over finite fields. The approach is based on the A-module structure of the dual of A, which translates algorithmically to techniques of the type "baby steps / giant steps." This is joint work with B. Salvy and É. Schost [1].

Given a zero-dimensional ideal I in some polynomial ring $k[X_1, \ldots, X_n]$, a nice form for a parameterization of the solution set $V(I)$ of I is of the type

$$(1) \qquad V(I) = \left\{ \left(\frac{g_1(a)}{g(a)}, \ldots, \frac{g_n(a)}{g(a)} \right) \ \middle|\ m(a) = 0 \right\}$$

for polynomials m, g, and g_i. In other words, solutions are indexed by the zeros of a univariate polynomial m, and the ith coordinate of the solutions is the evaluation of the fixed rational function g_i/g at those zeros. With additional technical constraints, (1) is called a *rational parameterization* of the variety $V(I)$. Note that by the algebraic nature of the problem, polynomials could be considered in place of rational functions with common denominators, but the choice of rational parameterizations proves useful to obtain compact expressions and algorithms with low complexity.

An algebraic quantity needed in several works to solve polynomial systems is the *minimal polynomial* of suitable elements u of the quotient algebra $A = k[X_1, \ldots, X_n]/I$. (By the minimal polynomial m_u of $u \in A$, we mean the unique monic polynomial of minimal degree such that $m(u) = 0$.) For example, in the algorithms below, obtaining the polynomials g and g_i of rational parameterizations indirectly requires to compute minimal polynomials.

The goal of this work is to accelerate the computation of minimal polynomials in A and of rational parameterizations of the variety $V(I)$. Additional motivation is given by the need for such calculations for polynomial factorization, in cryptography, effective Galois theory, effective theory of $\mathcal{D}$-modules, when counting and approximating zeros, etc. We present several probabilistic algorithms with several types of inputs and different complexity. A first class of algorithms takes as input one or a few matrices of multiplication by selected elements of A. In the case of the calculation of minimal polynomials, the result is that our approach gains when the degree δ of the minimal polynomial is relatively small, compared to the dimension D of A as a k-vector space. In the same way, our algorithm for the calculation of rational parameterizations gains when an a priori bound δ for the degree of minimal polynomials to be computed as a subtask is sufficiently smaller

than D. An additional gain by an order of $2^{-n}\sqrt{\delta}$ is made available by algorithms that take the whole multiplication table of A.

The algorithms presented here have been implemented in the computer algebra system Magma.

1. **Computation of Minimal Polynomials**

We focus on the computation of the minimal polynomial of an element u of the algebra A. The cost of the naive algorithm—express the powers of u in terms of a k-basis of A before looking for a linear dependency—is dominated by the calculations of the successive powers.

A first ingredient to improve the calculation of a minimal polynomial is by projection of powers, an idea already used in other contexts by Wiedemann and Shoup. Indeed, observe that when u satisfies an algebraic relation $a_d u^d + \cdots + a_0 = 0$, then

$$a_d \ell(u^{d+i}) + \cdots + a_0 \ell(u^i) = 0$$

for any k-linear form ℓ on A and any integer i, making the sequence of the $\ell(u^i)$ linear recurrent. One thus looks for the minimal polynomial $m_{u,\ell}$ of the sequence $\mathcal{L}_u = \big(\ell(u^i)\big)_{i \geq 0}$, which for "generic" ℓ is equal to m_u. (For unlucky choices of ℓ, it is only a divisor of m_u.)

Specifically, the algorithm chooses a k-linear form ℓ from the dual A^* of A, then determines the first 2δ terms of the scalar sequence $\mathcal{L}_u$. Using the Berlekamp–Massey algorithm, it next determines $m_{u,\ell}$, which merely amounts to computing a Padé approximant for the (truncated) series $\sum_{i=0}^{2\delta-1} \ell(u^i) U^i$, as one can prove the existence of a relation

$$(2) \qquad \sum_{i=0}^{2\delta-1} \frac{\ell(u^i)}{U^{i+1}} = \frac{G_{u,\ell}}{m_u}$$

for a polynomial $G_{u,\ell}$ of degree at most δ. Apart from the projection step, the complexity of this algorithm decreases from the complexity of linear algebra, in $O(\delta^\omega)$ for $2 < \omega \leq 3$, to $O(\delta^2)$. The overall complexity thus remains dominated by the calculations of the successive powers.

A second improvement consists in a better calculation of the powers, and follows a "baby steps / giant steps" approach: instead of computing the $\ell(u^i)$ in sequence for i to δ, one only computes $t = O(\sqrt{\delta})$ powers u^i for $1 \leq i \leq t$ and evaluates them at $O(\sqrt{\delta})$ forms of the form $x \mapsto \ell(u^{it}x)$. Computing those forms efficiently requires a better understanding of the structure of A^*. Specifically, for any $a \in A$, we consider the k-linear application of multiplication by a from A to itself, which by transposition induces a k-linear map of A^* to itself: this *transposed product* $a \cdot \ell$ maps an element $x \in A$ to $\ell(xa)$. The dual A^* thus turns out to be an A-module. Since (on suitable bases) the matrix of the (transposed) product by a in A^* is the transposed of the matrix of the multiplication by a in A, Tellegen's transposition principle predicts that the complexity of computing the transposed product by a is that of computing the multiplication by a. Guided by this heuristic, Bostan et al. have obtained an algorithm to compute all the projections and have gained essentially a factor of $\sqrt{\delta}$ on the naive complexity. For the sake of exposition, the description of this algorithm is postponed to Section 3.

Detailed complexity analysis leads to the following result.

Theorem 1. *Given $u \in A$, the minimal polynomial m_u (together with the polynomial $G_{u,\ell}$ in (2)) can be computed by a probabilistic algorithm:*

  1. *in $O(\delta D^2)$ operations in k if the matrix of multiplication by u is known;*
  2. *in $O(2^n \sqrt{\delta} D^2)$ operations in k if the multiplication table of A is known (and described in some specific way, see Section 3).*

This has to be compared with classical algorithms, respectively in $O(\delta D^2 + D^\omega)$ and $O_{\log}(D^\omega)$.

2. **Computations of Rational Parameterizations**

Elements of the quotient algebra $A = k[X_1, \ldots, X_n]/I$ can be viewed as functions on the variety $V(I)$. The idea behind the representation (1) is to distinguish two points by *distinct* values of a suitable polynomial function on $V(I)$. To this end, we introduce the notion of a *separating element* of A to refer to polynomial functions with this property.

The following central and, to the eye of the author of this summary, surprising result from [1] provides rational parameterizations as a by-product of computations of minimal polynomials.

Theorem 2. *Let u be a separating element of A of generic degree and ℓ be a linear form on A such that $m_{u,\ell} = m_u$. Then, a rational parameterization of $V(I)$ is given as*

$$V(I) = \left\{ \left(\frac{G_{u,x_1 \cdot \ell}(a)}{G_{u,\ell}(a)}, \ldots, \frac{G_{u,x_n \cdot \ell}(a)}{G_{u,\ell}(a)} \right) \;\middle|\; m_u(a) = 0 \right\}.$$

Then, an algorithm for rational parameterizations is the following. An element u is chosen in A, as well as a k-linear form ℓ in A^*. The minimal polynomial m_u and the related polynomial $G_{u,\ell}$ are computed by the algorithm of the previous section. The forms $x_j \cdot \ell$ are then computed. By projecting powers like in the previous section, the series $R_i = \sum_{i \geq 0} \ell(x_j u^i)/U^{i+1}$ are computed with precision $O(U^{1-\delta})$. The polynomials $G_{u,x_i \cdot \ell}$ are then obtained by mere polynomial products from the formula $G_{u,x_i \cdot \ell} = m_u R_i$, which assumes the generically verified identity $m_u = m_{u,\ell} = m_{u,x_i \cdot \ell}$. Again, the bottleneck of the calculation is the projection step.

Detailed complexity analysis leads to the following result. For non-zero characteristic, a technical condition is given in terms of the *radical* $\sqrt{I}$ of the ideal I, in other words, the set of polynomials that, when raised to some power, lie in the ideal I.

Theorem 3. *Given a separating element $u \in A$ of generic degree. Assuming that the field k is a perfect field of characteristic zero or at least $\min\{ s \mid \sqrt{I}^s \subset I \}$, a rational parameterization of $V(I)$ can be computed by a probabilistic algorithm:*

1. *in $O(\delta D^2 + n D^2)$ operations in k if the matrices of multiplication by u and the x_i are known;*
2. *in $O(n 2^n \sqrt{\delta} D^2)$ operations in k if the multiplication table of A is known (and described in some specific way, see Section 3).*

This has to be compared with Rouillier's RUR algorithm in $O(D^3 + n D^2)$.

3. **Algorithm for Effective Transposed Product**

In this section, we show that both multiplications in A and transposed products in A^* can be performed in $O(2^n D^2)$ operations in k when the multiplication table of A is known. More specifically, we require that the multiplication table be described in terms of a special vectorial basis of A. A basis $\{\omega_i\}_{i=1,\ldots,D}$ of the quotient algebra $A = k[X_1, \ldots, X_n]/I$ is called a *monomial basis* when the ω_i are given as $\omega_i = m_i + I$ for a collection M of monomials m_i "under the stairs" of a Gröbner basis for I. (In particular, if m is a monomial in this collection, all its monomial divisors are there as well; this property is in fact sufficient to obtain the subsequent results.) The property of being a monomial basis, not just any basis, has a strong consequence on products: the set $M \cdot M$ of the products $m_i m_j$ has cardinality bounded above by $2^n D$. After fixing orders on M and $M \cdot M$, the multiplication table of A is given as a $|M| \times |M \cdot M|$ matrix T.

In order to compute the multiplication of $u = \sum_{i=1}^{D} u_i \omega_i \in A$ and $v = \sum_{i=1}^{D} v_i \omega_i \in A$, just compute the product of $\sum_{i=1}^{D} u_i m_i$ with $\sum_{i=1}^{D} v_i m_i$ in $k[X_1, \ldots, X_n]$, write the vector V of the

coefficients of this product with respect to the basis $M \cdot M$, and compute the product TV to get the coefficients with respect to the basis M of the product uv. This uses $O(D^2)$ operations for the first step and $O(2^n D^2)$ for the second, so in total $O(2^n D^2)$ operations in k for multiplication in A.

We now turn to the computation of transposed products, in which certain truncations of generating series play a crucial role. We introduce the notation

$$S(\ell, C) = \sum_{m \in C} \ell(m + I)m \in k[X_1, \ldots, X_n]$$

for any form $l \in A^*$ and any collection C of monomials. One readily verifies that when $M = \{m_i\}_{i=1}^D$ corresponds to a monomial basis of A and with $u = \sum_{i=1}^D u_i m_i + I$, the polynomial $S(u \cdot \ell, M)$ is given as the part with support in M of the product

(3)
$$\left(\sum_{i=1}^D u_i m_i^{-1} \right) S(\ell, M \cdot M).$$

An algorithm to compute a transposed product $u \cdot \ell$ is thus the following. Write $\ell = \sum_{i=1}^D \ell_i \omega_i^*$ in terms of the dual basis $\{\omega_i^*\}_{i=1}^D$ of the monomial basis of A. Multiply this vector by the transposed of the matrix T that encodes the multiplication table of A. The resulting vector gives the coefficients of $S(\ell, M \cdot M)$ with respect to the basis $M \cdot M$. Compute the product in (3) and read off the coefficients of the form $u \cdot \ell$ with respect to the dual basis $\{\omega_i^*\}_{i=1}^D$. Again, this uses $O(2^n D^2)$ operations in k in total.

4. A Note on the Probabilistic Nature of the Algorithms

The algorithms of Sections 1 and 2 are randomized by the choice of the linear form ℓ. If the coordinates of ℓ are chosen in a finite subset F of k, then the probability of failure of the algorithms is bounded above by $\delta/|F|$ (uniformly in the input u in the case of the minimal polynomial computation).

Bibliography

[1] Bostan (Alin), Salvy (Bruno), and Schost (Éric). – Fast algorithms for zero-dimensional polynomial systems using duality. – To appear in *Applicable Algebra in Engineering, Communication and Computing*.