On the complexity exponent of polynomial system solving

Joris van der Hoeven, Grégoire Lecerf

On the complexity exponent of polynomial system solving

JORIS VAN DER HOEVEN[a], GRÉGOIRE LECERF[b]

CNRS (UMR 7161, LIX)
Laboratoire d'informatique de l'École polytechnique
Campus de l'École polytechnique
1, rue Honoré d'Estienne d'Orves
Bâtiment Alan Turing, CS35003
91120 Palaiseau, France

a. Email: vdhoeven@lix.polytechnique.fr
b. Email: lecerf@lix.polytechnique.fr

Final version of March 2020

We present a probabilistic Las Vegas algorithm for solving sufficiently generic square polynomial systems over finite fields. We achieve a nearly quadratic running time in the number of solutions, for densely represented input polynomials. We also prove a nearly linear bit complexity bound for polynomial systems with rational coefficients. Our results are obtained using the combination of the Kronecker solver and a new improved algorithm for fast multivariate modular composition.

KEYWORDS: Polynomial system solving, Geometric resolution, Complexity bounds

A.M.S. SUBJECT CLASSIFICATION: 14-04, 14Q20, 14B05, 68W30

1. INTRODUCTION

Let $\mathbb{K}$ be an *effective field*. This means that we have a data structure for representing elements of $\mathbb{K}$, together with algorithms for performing the field operations. Consider a homogeneous system of polynomial equations

$$f_1 = \cdots = f_n = 0,$$

where $f_i \in \mathbb{K}[x_0,\ldots,x_n]$. We are interested in the exact resolution of such a system through the computation of a *parametrization* of its zero-set by a so-called *primitive element*; see section 4.2 for a precise definition.

Throughout this paper, we assume that $f_1,\ldots,f_n$ are given in standard dense representation. The total degree of each f_i is written d_i, and we let $D_i := d_1 \cdots d_i$. Our algorithm requires the following *regularity conditions*:

$\mathbf{R_1}$. $f_1,\ldots,f_n$ is a *regular sequence*;

$\mathbf{R_2}$. The *intermediate ideals* $I_i := (f_1,\ldots,f_i)$ are *absolutely radical*, which means radical over the algebraic closure $\bar{\mathbb{K}}$ of $\mathbb{K}$, for $i = 1,\ldots,n$;

$\mathbf{R_3}$. $d_i \geqslant 2$, for $i = 1,\ldots,n$.

Conditions R_1 and R_2 formalize the idea of a "generic system" in the sense that they are likely to hold when the f_i admit random coefficients. Under these conditions it is well known that the system $f_1 = \cdots = f_n = 0$ admits exactly D_n geometrically regular solutions in the projective space $\mathbb{P}^n$ over $\bar{\mathbb{K}}$. Condition R_3 is included for the sake of simplicity. It is not restrictive because linear equations can be removed by means of linear Gaussian elimination, and by performing a suitable linear change of the variables; see Remark 5.6.

We prove new probabilistic complexity bounds for computing all the solutions in terms of a univariate parametrization by a primitive element. For finite fields $\mathbb{K}$, our bound is arbitrarily close to quadratic in the number of the solutions whenever $d_1 = \cdots = d_n$: see Corollary 5.5. For rational coefficients of bounded height, we deduce a complexity bound that is nearly linear in the expected bit size of the output of the algorithm: see Theorem 6.11. These results improve upon the best previously known complexity bounds.

Notations

We set $\bar{d} := \max(d_1, \ldots, d_n)$. In order to simplify the presentation of complexity bounds, we use *soft-Oh* notation: $f(n) \in \tilde{O}(g(n))$ means that $f(n) = g(n) \log_2^{O(1)}(g(n) + 3)$; see [21, chapter 25, section 7] for technical details. The least integer larger or equal to x is written $\lceil x \rceil$; the largest one smaller or equal to x is written $\lfloor x \rfloor$.

The $\mathbb{K}$-vector space of polynomials of degree $< d$ is written $\mathbb{K}[x]_{<d}$. The remainder of the division of a by b is written a rem b. In order to express that an expression A is explicitly computed modulo b, we write $A \bmod b$.

The *Galois ring* of characteristic p^κ and cardinality $p^{\kappa k}$, written $\mathrm{GR}(p^\kappa, k)$, is defined as

$$\mathrm{GR}(p^\kappa, k) := (\mathbb{Z}/p^\kappa\mathbb{Z})[z]/(\theta(z)),$$

with θ monic and irreducible modulo p.

1.1. Related work

The complexity of polynomial system solving is a central question in effective algebraic geometry, which is still open. Except in some very special cases, no softly linear time algorithms are known. In order to simplify the discussion, we assume that $d_i \geqslant 2$ for $i = 1, \ldots, n$.

One known favorable special case concerns systems of bounded degrees d_i over $\mathbb{K} = \mathbb{Q}$ that satisfy R_1 and R_2: in suitable bit complexity models (computation trees, or random access memory machines), the variant of the Kronecker solver designed by Giusti, Lecerf and Salvy [27] admits a softly linear running time for $n \to \infty$. This solver is probabilistic of Las Vegas type, with a well understood probability of failure that can easily be made arbitrarily small in practice; for details, see the recent work by Giménez and Matera [22]. The Kronecker algorithm has a long history, lying deep in Kronecker's work [44], but it emerged after a series of seminal articles by Giusti, Heintz, Matera, Morais, Pardo, and their collaborators in the nineties [24, 25, 26]; for the history of this algorithm along with references, we refer to the introduction of [18].

Assume that the input polynomials are given as a straight-line program of size L over $\mathbb{K}$ without division (see the definition in [13] for instance) and that we use the computation tree model for our complexity measures. If the cardinality of $\mathbb{K}$ is sufficiently large and the conditions R_1 and R_2 hold, then the Kronecker solver computes a parametrization of the solution set in terms of a primitive element using

$$L\,\tilde{O}(D_n^2) \tag{1.1}$$

operations in $\mathbb{K}$ and with a uniformly bounded probability of failure. The complexity bound (1.1) essentially comes from [27, Theorem 1] by taking into account that $D_{i+1} \geqslant 2D_i$; this is detailed in section 5 below. If the d_i are all bounded, then we notice that $L = (\log D_n)^{O(1)}$, so the bound (1.1) is essentially quadratic. In the other extreme case when n is bounded, we rather have $L \asymp D_n$, so the bound (1.1) becomes essentially cubic. In this paper, we show how to obtain a nearly quadratic bound for this case as well.

Gröbner bases constitute another cornerstone for polynomial system solving. Algorithms in this setting are usually deterministic, but their complexities are highly intricate in general: they depend on both the system and the admissible ordering attached to monomials. Complexities of unfavourable cases are doubly exponential in n; see a summary of known families in [21, chapter 21]. The first simply exponential bounds of the form $D_n^{O(1)}$ for zero dimensional systems are due to Giusti, Lazard and Lakshman in the eighties [23, 45, 46, 47, 48]. The constant hidden behind the latter "O" may be related to a feasible exponent ω for linear algebra, in the sense that any two square matrices of size $n \times n$ may be multiplied with $O(n^\omega)$ operations in their coefficient ring. Basically one may perform Gaussian elimination on Macaulay matrices: for instance, under R_1 and for the graduated reverse lexicographic ordering, the resolution requires $O\Big(n E \binom{n+E}{n}^\omega\Big)$ operations in $\mathbb{K}$, where $E = 1 + \sum_{i=1}^n (d_i - 1)$ is the Macaulay bound; see [4], or [8, chapter 26] for a proof from scratch, for instance.

At present time it is not known how to exploit the structure of Macaulay matrices to perform Gaussian elimination in time quadratic in D_n or even in $\bar{d}^n$. One contribution to this problem, due to Canny, Kaltofen, and Yagati [14], relies on sparse polynomial interpolation and the Wiedemann solver. Further important results based on structured linear algebra have been established by Mourrain, Pan, and Ruatta [56]: they proved that the number of roots and of real roots can be computed with $O(3^n D_n^2 \log D_n)$ floating point operations, and that all the δ (real) roots in a given box up to the error 2^{-b} can be computed with $O((\mu + \nu)\, n\, 3^n\, \delta\, D_n^2 \log D_n \log b)$ floating point operations, where μ and ν depend linearly on $\log b$ as well as on the conditioning of the system. Yet another important advance here is Faugère's F5 algorithm, which suppresses useless rows in Macaulay matrices. However, its worst case complexity $O(n\,(3\bar{d})^{3n})$ remains essentially cubic in D_n [3, 4]. Other methods derived or inspired by Macaulay's work have been developed by Mourrain and Trébuchet [57, 58, 59].

For Gröbner bases of zero dimensional systems, monomial orderings may be changed quite conveniently by means of linear algebra. This is known as the FGLM algorithm (named after the initials of its authors, Faugère, Gianni, Lazard and Mora [20]) and involves a cost $O(n D_n^3)$. Recently, Faugère, Gaudry, Huot and Renault [19] decreased it to $\tilde{O}(D_n^\omega)$. Nevertheless one must keep in mind that the best practical values for ω are only about 2.8, which is far from the best known theoretical bound <2.3728639, due to Le Gall [49].

Another aspect of the present paper concerns multivariate modular composition, that is the computation of $f(g_1, \ldots, g_n) \operatorname{rem} h$ where $f \in \mathbb{K}[x_1, \ldots, x_n]$, $h \in \mathbb{K}[x]$ has degree d, and $g_1, \ldots, g_n$ are in $\mathbb{K}[x]_{<d}$. When $n = 1$, the best known complexity bound $O(d^2)$ for any field $\mathbb{K}$ is due to Brent and Kung [11]. Their algorithm even yields a sub-quadratic cost $O(d^\varpi)$ when using fast linear algebra; see [40, p. 185]. Here the constant $\varpi > 1.5$ is such that a $\sqrt{n} \times \sqrt{n}$ matrix over $\mathbb{K}$ may be multiplied with another $\sqrt{n} \times n$ rectangular matrix with $O(n^\varpi)$ operations in $\mathbb{K}$. Huang and Pan proved that $\varpi < 1.667$; see [38, Theorem 10.1]. In [35, section 3] we extended Brent and Kung's result to the multivariate

setting. A major breakthrough for the modular composition problem is due to Kedlaya and Umans [41, 42], in the case when $\mathbb{A}$ is a finite field $\mathbb{F}_{p^k}$ and more generally a finite ring of the form $(\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ for any integer r and θ monic of degree k. For any fixed rational value $\varepsilon > 0$, they showed that $f(g_1, \ldots, g_n)$ can be computed modulo h in time $O(((d+1)^n k \log p)^{1+\varepsilon})$ whenever the partial degrees of f are $\leqslant d$, and under the condition $n = d^{o(1)}$; see [42, Theorem 7.1]. Extensions to compositions modulo triangular sets can be found in [63].

1.2. Our contributions

The first technical but important contribution of this paper concerns the refinement of Kedlaya and Umans' complexity result on modular composition [41, 42]. In fact, in Corollary 3.6 we achieve a sharper complexity bound in terms of the total degree d of f, that is softly linear in the bit size $O(k \log p)$ of the coefficients, and that holds without the assumption $n = d^{o(1)}$. We also handle fast evaluations of f in a larger algebra of the form $\mathbb{F}_{p^k}[e, y, t]/(e^2, y^M, Q(e, y, t))$, where $M \geqslant 1$ and Q is monic in t of degree d^n: this is a particular case of composition modulo triangular sets studied in [63], for which we also improve upon the dependency in the coefficient size. These new results are based on our recent advances on multivariate multi-point evaluation algorithms [36]. At a more abstract level, our algorithm applies to any field $\mathbb{K}$ over which fast multi-point multivariate polynomial evaluation exists. We recall that fast multivariate multi-point evaluation remains an important open problem: no efficient algorithms are known over a general abstract field $\mathbb{K}$ and we are not aware of any efficient implementations of Kedlaya and Umans' method; for some recent progress on special cases, we refer to [33, 34].

Our main contributions concern the complexity of polynomial system solving. We first prove a Las Vegas type probabilistic bit complexity bound $\tilde{O}(D_n \bar{d}^{(1+\varepsilon)(n-1)} k \log p)$ over the finite field $\mathbb{K} = \mathbb{F}_{p^k}$, for the dense representation of input polynomials, and where $\varepsilon > 0$ is any fixed rational value: see Corollary 5.5. Whenever $d_1 = \cdots = d_n = \bar{d}$, this bound is arbitrarily close to quadratic in the number of the roots $D_n = \bar{d}^n$ of the system. This improves upon previously known bounds. For example, the complexity bound (1.1) simplifies to $D_n^{2+O(\varepsilon)}$ whenever $L \leqslant D_n^{\varepsilon}$. With input polynomials given in dense representation, the quantity L is of the order $\sum_{i=1}^{n} \binom{d_i+n}{n}$. If all the d_i remain bounded and if n tends to infinity, then we have

$$\binom{d_i+n}{n} \sim \frac{n^{d_i}}{d_i!}.$$

Consequently, the expected complexity of the Kronecker solver becomes $\tilde{O}(D_n^2)$, in terms of the number of arithmetic operations in $\mathbb{K}$. However, if the d_i are no longer bounded, then $\binom{d_i+n}{n}$ may grow with $O(d_i^n)$ and the worst case complexity of the solver becomes $\tilde{O}(D_n^2 \bar{d}^n)$.

Our next main result concerns the case $\mathbb{K} = \mathbb{Q}$. We assume that $f_1, \ldots, f_n$ have integer coefficients of bit size $\leqslant \bar{h}$, and we achieve expected time $\tilde{O}(D_n \bar{d}^{(1+\varepsilon)n} \bar{h})$. Whenever $d_1 = \cdots = d_n = \bar{d}$, this time turns out to be nearly linear in the bit size of the output: see the precise bound in Theorem 6.11. This complexity bound admits major consequences for symbolic real solving, especially for algorithms based on the Kronecker solver; see for instance [2] and references therein. Once a Kronecker parametrization is known, it is also possible to appeal to univariate numerical solvers to deduce complex or real approximations of the roots of the system; see for instance the book [55].

Let us briefly describe the structure of this paper. Section 2 is devoted to prerequisites about the complexity model and fast multi-point polynomial evaluation. The next section 3 deals with various particular cases of multivariate modular composition involved in the Kronecker solver. Section 4 concerns the algebraic data structures needed by the solver, along with general position and some of the randomness aspects. The solver is itself recalled in section 5, that ends with our main result over the finite fields. Our main complexity bound over the rational numbers is stated in the final section 6.

It may be helpful to notice that the probability analysis of our algorithms contains two main stages. On the one hand, we need to put the system in general position and prepare the main data structures. This part is handled in section 4. On the other hand, for the sake of efficiency, some subalgorithms of the Kronecker solver involve additional random parameters that must be taken outside certain proper closed subvarieties (see Propositions 5.2 and 5.3), or integer parameters that must be coprime with a certain set of bad primes (see Lemma 6.7). The probability of picking suitable random parameters of this kind will be analyzed separately.

2. COMPLEXITY MODEL AND BASIC OPERATIONS

Throughout this paper, we will be working in the Turing machine model [61] with sufficiently many tapes (seven tapes are usually sufficient in order to implement basic routines on polynomials, series, and matrices in a natural way). The Kronecker algorithm is randomized, so it also requires a special instruction to generate a random symbol in one cell within constant time.

In some cases, algebraic structures have a natural bit size (e.g. modular integers, finite fields); in other cases the size is variable (e.g. arrays, polynomials). In both cases elements are represented on tapes as sequences of symbols, followed by a specific termination symbol "#". The reader must keep in mind that heads of the machine can just move one cell left or right at each time step. Algorithms must take care of using data in the most contiguous way as possible, and loop counters cannot be used for free.

The rest of the section gathers standard data types and elementary operations needed in the next sections. We freely use well known complexity bounds on polynomials and matrices from [21]; details about Turing machine implementations of these algorithms can be found in [65].

2.1. Basic data types

Integers

We use binary representation for integers. A modular integer in $\mathbb{Z}/r\mathbb{Z}$ is represented by its natural preimage in $\{0,\ldots,r-1\}$. Integers can be added in linear time and multiplied in softly linear time $O(n\log n)$; see [30] and historical references therein. Integer divisions and extended gcds in bit size $\leqslant n$ also take softly linear time [64]. We will also use truncated p-adic integers, for which we refer the interested reader to [6] for practical algorithms.

Arrays

One dimensional arrays are sequences of elements terminated by "#". For example the vector $(1,0,1)\in\mathbb{F}_2^3$ is stored as 1#0#1##.

For bidimensional arrays we use column-major representation. This means that an array $(A_{i,j})_{1\leqslant i\leqslant r,1\leqslant j\leqslant c}$ of size $r\times c$ (r rows and c columns), is stored as the vector of its columns, i.e. $((A_{1,1},\ldots,A_{r,1}),(A_{1,2},\ldots,A_{r,2}),\ldots,(A_{1,c},\ldots,A_{r,c}))$. Such arrays are allowed to contain elements of different types and sizes. For example the matrix $\left(\begin{smallmatrix}1&1\\0&0\end{smallmatrix}\right)$ over $\mathbb{F}_2$ is stored as 1#0##1#0###. In the Turing machine model, we recall that the transposition of a bidimensional array can be achieved in softly linear time:

LEMMA 2.1. [36, Lemma 2.3] *Let $A=(A_{i,j})$ be an $r\times c$ matrix. Let $b_{i,j}$ denote the size of $A_{i,j}$ for all i,j, and define $B:=\sum_{i,j}b_{i,j}$. Then A can be transposed in time $O((B+rc)\log\min(r,c))$.*

Univariate polynomials

For univariate polynomials we use dense representation, which means that a polynomial of degree d is stored as the vector of its $d+1$ coefficients from degrees 0 to d. Additions and subtractions take linear time in d. Multiplying two polynomials over finite fields may be achieved in softly linear time [15, 29, 31].

In the present paper we consider finite rings of the form $\mathbb{A}:=(\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ with $r\geqslant 2$ and $\theta\in(\mathbb{Z}/r\mathbb{Z})[z]$ monic of degree k. Elements of $\mathbb{A}$ are stored as their natural preimage in $\mathbb{Z}[z]_{<k}$ with coefficients in $\{0,\ldots,r-1\}$. Additions and subtractions in $\mathbb{A}$ take linear time, products take time $\tilde{O}(k\log r)$; see for instance [21, part II].

LEMMA 2.2. [36, adapted from Lemma 2.12] *Let $\mathbb{A}:=(\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ be as above and let $A_1,\ldots,A_m$ be polynomials in $\mathbb{A}[x]_{<d}$. For any sequence of points $\alpha_1,\ldots,\alpha_N$ in $\mathbb{A}$ we may compute $A_1(\alpha_1),\ldots,A_m(\alpha_1),\ldots,A_1(\alpha_N),\ldots,A_m(\alpha_N)$ in time $m\tilde{O}((d+N)k\log r)$.*

LEMMA 2.3. [36, adapted from Lemma 2.14] *Let $\mathbb{A}:=\mathrm{GR}(p^\kappa,k)$ be a Galois ring, let $\alpha_1,\ldots,\alpha_N$ be elements in $\mathbb{A}$ such that $\alpha_i-\alpha_j$ is invertible modulo p for all $i\neq j$, and let $b_1,\ldots,b_N$ be vectors in $\mathbb{A}^m$. We may compute the unique polynomials $A_1,\ldots,A_m$ in $\mathbb{A}[z]_{<N}$ such that $A_j(\alpha_i)=b_{i,j}$ for $i=1,\ldots,N$ and $j=1,\ldots,m$ in time $m\tilde{O}(N\kappa k\log p)$.*

Multivariate polynomials

For a polynomial $f\in\mathbb{A}[x_1,\ldots,x_n]$ we use the *recursive dense representation*, by regarding f as an element of $\mathbb{A}[x_1][x_2]\ldots[x_n]$. In particular, f admits the same representation as its expansion $f=f_0+f_1x_n+\cdots+f_{\ell_n-1}x_n^{\ell_n-1}\in\mathbb{A}[x_1,\ldots,x_{n-1}][x_n]$ as a univariate polynomial in x_n. In our algorithms, the number of variables n is not part of the representation of f, so it must be supplied as a separate parameter.

The *support* $\mathrm{supp}\,f$ of $f\in\mathbb{A}[x_1,\ldots,x_n]$ is defined as the set of monomials with non-zero coefficients and we write $|\mathrm{supp}\,f|$ for its cardinality. Assuming that, apart from the mandatory trailing "#" symbol, the representations of coefficients in $\mathbb{A}$ do not involve other "#" symbols (this can always be achieved through suitable renamings $\#\rightsquigarrow\#_{\mathbb{A}}$), we denote the number of "#" symbols involved in the representation of f by $|f|_\#$. We notice that $|\mathrm{supp}\,f|\leqslant|f|_\#$. For the remainder of this subsection, we assume that the size of elements in $\mathbb{A}$ is bounded by a constant $\mathsf{s}_{\mathbb{A}}$.

Example 2.4. The univariate polynomial $f(x)=c_0+c_1x+\cdots+c_dx^d$ of degree d is represented by $c_0\#c_1\#\ldots c_d\#\#$. The bivariate polynomial $f(x_1,x_2)=c_{0,0}+c_{0,1}x_1+(c_{1,0}+c_{1,1}x_1)x_2$ is represented by $c_{0,0}\#c_{0,1}\#\#c_{1,0}\#c_{1,1}\#\#\#$.

LEMMA 2.5. [36, Lemma 2.5] *Let $f\in\mathbb{A}[x_1,\ldots,x_n]$ be of partial degree $<\ell_i$ in x_i for $i=1,\ldots,n$. Then $|f|_\#\leqslant\sum_{i=1}^n\ell_1\cdots\ell_i+1\leqslant n\pi+1$, where $\pi:=\ell_1\cdots\ell_n\geqslant|\mathrm{supp}\,f|$.*

LEMMA 2.6. [36, Lemma 2.6] *Let $f \in \mathbb{A}[x_1,\ldots,x_n]$ be a non-zero polynomial of total degree $\leqslant d$. Then $|f|_\# \leqslant \sum_{i=0}^{n} \binom{d+i}{i} \leqslant n\rho+1$, where $\rho := \binom{d+n}{n} \geqslant |\mathrm{supp}\ f|$.*

LEMMA 2.7. [36, Lemma 2.7] *The partial degree bounds $\ell_1 = 1+\deg_{x_1} f, \ldots, \ell_n = 1+\deg_{x_n} f$ of a non-zero polynomial $f \in \mathbb{A}[x_1,\ldots,x_n]$ can be computed in time $O(|f|_\#(n+\log\pi)+|\mathrm{supp}\ f|\,\mathsf{s}_{\mathbb{A}})$, where $\pi := \ell_1 \cdots \ell_n$.*

LEMMA 2.8. [36, Lemma 2.8] *The total degree $d = \deg f$ of a polynomial $f \in \mathbb{A}[x_1,\ldots,x_n]$ can be computed in time $O(n|f|_\#\log(d+3)+|\mathrm{supp}\ f|\,\mathsf{s}_{\mathbb{A}})$, with the convention that $\deg 0 := -1$.*

LEMMA 2.9. *Let $\mathbb{A} := (\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ be as above. Any partial derivative of $f \in \mathbb{A}[x_1,\ldots,x_n]$ of total degree $\leqslant d$ can be computed in time $n\rho\,\tilde{O}(k\log r)$, where $\rho := \binom{d+n}{n}$.*

Proof. In order to compute the partial derivative in x_n, we run over all the coefficients of f in time $|\mathrm{supp}\ f|\,\tilde{O}(k\log r)+O(|f|_\#) = n\rho\,\tilde{O}(k\log r)$ by Lemma 2.6.

If we are interested in the derivative in x_i with $i<n$, then we reinterpret f as an element of $\mathbb{F}_{p^k}[x_1,\ldots,x_i][x_{i+1}\ldots,x_n]$ and its derivative requires time $|\mathrm{supp}\ f|\,\tilde{O}(k\log r)+O(|f|_\#) = n\rho\,\tilde{O}(k\log r)$. □

LEMMA 2.10. *Let $\mathbb{A} := (\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ be as above, let $f \in \mathbb{A}[x_1,\ldots,x_n]$ be of total degree d, and let $\alpha \in \mathbb{A}$. Then $f(\alpha,x_2,\ldots,x_n)$ may be computed in time $|\mathrm{supp}\ f|\,\tilde{O}(k\log r)+O(|f|_\#) = n\rho\,\tilde{O}(k\log r)$, where $\rho := \binom{d+n}{n}$.*

Proof. We use Horner's method to evaluate univariate polynomials over $\mathbb{A}$ at α in softly linear time. In total, the machine runs over the entire representation of f and the time spent in arithmetic operations is at most $\rho\,\tilde{O}(k\log r)$. □

LEMMA 2.11. [36, adapted from Lemma 2.13] *Let $\mathbb{A} := (\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ be as above, let $f \in \mathbb{A}[y]_{<M}[x_1,\ldots,x_n]$, and let $\alpha_1,\ldots,\alpha_N$ be elements in $\mathbb{A}$. Then $f(\alpha_1,x_1,\ldots,x_n),\ldots,f(\alpha_N,x_1,\ldots,x_n)$ can be computed in time*

$$\rho\,(M+N)\,(\tilde{O}(\log^2 M)+\min(\log\rho,\log N))\,\tilde{O}(k\log r)+O(N|f|_\#),$$

where ρ is the cardinality of the support of f in the variables $x_1,\ldots,x_n$.

At several places we will use the following bound on the cardinality of the support of a polynomial in n variables and total degree $d \geqslant 2$:

$$\frac{\binom{d+n}{n}}{d^n} = \prod_{i=1}^{n} \frac{d+i}{id} = \prod_{i=1}^{n} \left(\frac{1}{i}+\frac{1}{d}\right) \leqslant \frac{3}{2}. \tag{2.1}$$

For the sake of efficiency, a *homogeneous polynomial* f in the variables $x_0,\ldots,x_n$ has a dedicated representation made of $f^\flat(x_1,\ldots,x_n) := f(1,x_1,\ldots,x_n)$ and $\deg f$. In this way, f can be recovered as $x_0^{\deg f} f^\flat(x_1/x_0,\ldots,x_n/x_0)$.

2.2. Multivariate multi-point evaluation

The problem called *multivariate multi-point evaluation* over an effective ring $\mathbb{A}$ is the following: given $f \in \mathbb{A}[x_1,\ldots,x_n]$ and points $\alpha_1,\ldots,\alpha_N$ in $\mathbb{A}^n$, compute $f(\alpha_1),\ldots,f(\alpha_N)$. The first following statement concerns the cost of the naive evaluation algorithm in terms of the total degree of f.

LEMMA 2.12. [36, adapted from Lemma 3.8] *Let $\mathbb{A} := (\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ with θ monic of degree k be as above, let $f \in \mathbb{A}[x_1,\ldots,x_n]$ be of total degree $\leqslant d$. Then we may evaluate f at a point $(a_1,\ldots,a_n) \in \mathbb{A}^n$ in time $n\binom{d+n}{n}\tilde{O}(k\log r)$.*

In terms of the partial degrees of f, we will need the following better complexity bounds that refine previous work by Kedlaya and Umans [41, 42]. The first theorem takes partial degrees into account, while the second one is in terms of the total degree.

THEOREM 2.13. [36, Corollary 4.5] *Let $\varepsilon > 0$ be a fixed rational value. Let $\mathbb{A} := (\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ with θ monic of degree k, let $f \in \mathbb{A}[x_1,\ldots,x_n]$ be of partial degree $<\ell$ in x_i for $i=1,\ldots,n$, and let $\alpha_1,\ldots,\alpha_N$ be a sequence of points in $\mathbb{A}^n$. Then we may compute $f(\alpha_1),\ldots,f(\alpha_N)$ in time*

$$(1+\varepsilon)^n (N + (n\ell\log(n\ell))^n)\,\tilde{O}(n^9\ell^7 k\log r).$$

THEOREM 2.14. [36, Corollary 4.6] *Let $\varepsilon > 0$ be a fixed rational value. Let $\mathbb{A} := (\mathbb{Z}/r\mathbb{Z})[z]/(\theta(z))$ with θ monic of degree k, let $f \in \mathbb{A}[x_1,\ldots,x_n]$ be of total degree $\leqslant d$, and let $\alpha_1,\ldots,\alpha_N$ be a sequence of points in $\mathbb{A}^n$. Then we may compute $f(\alpha_1),\ldots,f(\alpha_N)$ in time*

$$(1+\varepsilon)^n (N + ((3d+2n)\log(3d+2n))^n)\,\tilde{O}(n^2 d\,(d+n)^6 k\log r).$$

2.3. Linear changes of variables

The Kronecker solver needs to perform a linear change of variables of matrix N into the input homogeneous polynomials. Let f be such a polynomial in $\mathrm{GR}(p^\kappa,k)[x_0,\ldots,x_n]$ of total degree d, and let $N = (N_{i,j})_{0\leqslant i\leqslant n,\, 0\leqslant j\leqslant n}$ be a $(n+1)\times(n+1)$ matrix over $\mathrm{GR}(p^\kappa,k)[x_0,\ldots,x_n]$. We need to compute

$$(f\circ N)(x_0,\ldots,x_n) := f(N_{0,0}x_0+\cdots+N_{0,n}x_n,\ldots,N_{n,0}x_0+\cdots+N_{n,n}x_n).$$

A natural idea is to interpolate $(f\circ N)(1,x_1,\ldots,x_n)$ from its values at S^n, where S is a subset of $\mathbb{F}_{p^k}$ of cardinality $d+1$. Such an interpolation is known to be computable in time $\tilde{O}((d+1)^n\kappa k\log p)$; see Lemma A.8 of the appendix. On the other hand the values of f at $N(\{1\}\times S^n)$ may be obtained in time $\tilde{O}((d+1)^{(1+\varepsilon)(n+1)}\kappa k\log p)$ by [36, Proposition 5.7], for any fixed rational value $\varepsilon > 0$. However, this approach suffers from two drawbacks: it takes neither the homogeneity of f nor the structure of $N(\{1\}\times S^n)$ into account. At the end of Appendix A, we show the following sharper complexity result:

PROPOSITION 2.15. *Let $f \in \mathrm{GR}(p^\kappa,k)[x_0,\ldots,x_n]$ be homogeneous of degree $d \geqslant 2$, and let N be an $(n+1)\times(n+1)$ matrix over $\mathrm{GR}(p^\kappa,k)$. If $p^k > d$ and if a LU-decomposition of N is given, then we may compute $f\circ N$ in time $\tilde{O}(d^n\kappa k\log p)$.*

3. FAST MULTIVARIATE MODULAR COMPOSITION

This section is devoted to multivariate modular composition, that is the evaluation of a multivariate polynomial in $\mathbb{A}[x_1,\ldots,x_n]$ at a point in $\mathbb{E}^n$, where $\mathbb{E} := \mathbb{A}[t]/(Q(t))$ with $Q(t)$ monic and $\deg Q \geqslant 1$. As recalled in the introduction, no fast algorithm is known for this task over a general ring or field $\mathbb{A}$. For the Kronecker solver presented in this paper, the following particular instances of $\mathbb{A}$ are needed: $\mathbb{F}_{p^k}[y]/(y^M)$ and $\mathbb{Z}/p^\kappa\mathbb{Z}$, along with tangent numbers over theses rings. In this section, we design fast algorithms for these cases, on the basis of Kedlaya and Umans' ideas, and new variants from [36].

If $\kappa = 1$, then $\mathrm{GR}(p^\kappa, k)$ is the finite field $\mathbb{F}_{p^k}$, whereas $k = 1$ corresponds to the ring $\mathbb{Z}/p^\kappa\mathbb{Z}$ of the p-adic integers truncated to precision κ. Until the end of the section we set

$$\mathbb{A} := \mathrm{GR}(p^\kappa, k)[e, y] / (e^2, y^M), \tag{3.1}$$

and

$$\mathbb{E} := \mathbb{A}[t] / (Q(t)),$$

where $Q \in \mathbb{A}[t]$ is a monic polynomial of degree $D \geqslant 1$ in t.

3.1. Reduction to evaluation

Let $\mathsf{E}(n, \ell, d, N)$ represent a cost function for the multi-point evaluation of n-variate polynomials over $\mathrm{GR}(p^\kappa, k)$ with partial degrees $< \ell$, total degree $\leqslant d$, and for $\leqslant N$ evaluation points. The following algorithm reduces modular composition to multi-point evaluation.

Algorithm 3.1

Input. $f \in \mathbb{A}[x_1, \ldots, x_n]$ of partial degrees $< \ell$ and of total degree $\leqslant d$; $A_1, \ldots, A_n$ in $\mathbb{E}$.

Output. $f(A_1, \ldots, A_n)$.

Assumptions. $p^k > d\,(\max(D, M) - 1)$.

1. Write
$$\bar{A}_i(t) =: \bar{A}_{i,0}(y, t) + \bar{A}_{i,1}(y, t)\, e$$
for the canonical preimage of A_i in $\mathrm{GR}(p^\kappa, k)[e, y, t]$, where $\bar{A}_{i,0}$ and $\bar{A}_{i,1}$ belong to $\mathrm{GR}(p^\kappa, k)[y, t]$ and have partial degrees $< M$ in y and $< D$ in t. In a similar manner write
$$\bar{f} = \bar{f}_0(y, x_1, \ldots, x_n) + \bar{f}_1(y, x_1, \ldots, x_n)\, e$$
for the preimage of f in $\mathrm{GR}(p^\kappa, k)[e, y, x_1, \ldots, x_n]$.
2. Build points $\alpha_0, \ldots, \alpha_{d(\max(D,M)-1)} \in \mathrm{GR}(p^\kappa, k)$ whose reductions modulo p are pairwise distinct.
3. For $i = 0, \ldots, d\,(M-1)$ and for $j = 0, \ldots, d\,(D-1)$, compute
$$\beta_{i,j} := (\bar{A}_{1,0}(\alpha_i, \alpha_j), \ldots, \bar{A}_{n,0}(\alpha_i, \alpha_j)) \text{ and } (\bar{A}_{1,1}(\alpha_i, \alpha_j), \ldots, \bar{A}_{n,1}(\alpha_i, \alpha_j)).$$
4. For $i = 0, \ldots, d\,(M-1)$ compute $\varphi_i(x_1, \ldots, x_n) := \bar{f}_0(\alpha_i, x_1, \ldots, x_n)$ and $\psi_i(x_1, \ldots, x_n) := \bar{f}_1(\alpha_i, x_1, \ldots, x_n)$.
5. For $i = 0, \ldots, d\,(M-1)$ and for $j = 0, \ldots, d\,(D-1)$, compute
$$\gamma_{i,j} := \varphi_i(\beta_{i,j}) \text{ and } \delta_{i,j} := \psi_i(\beta_{i,j}) + \sum_{k=1}^{n} \frac{\partial \varphi_i}{\partial x_k}(\beta_{i,j})\, \bar{A}_{k,1}(\alpha_i, \alpha_j).$$
6. Interpolate the polynomials h_0 and h_1 in $\mathrm{GR}(p^\kappa, k)[y, t]$, of partial degrees $\leqslant d\,(M-1)$ in y and $\leqslant d\,(D-1)$ in t, and such that $h_0(\alpha_i, \alpha_j) = \gamma_{i,j}$ and $h_1(\alpha_i, \alpha_j) = \delta_{i,j}$ for all $i = 0, \ldots, d\,(M-1)$ and $j = 0, \ldots, d\,(D-1)$.
7. Return $h_0(y, t) + h_1(y, t)\, e$ reduced modulo (e^2, y^M, Q).

PROPOSITION 3.1. *Algorithm 3.1 is correct and takes time*

$$(n+2)\,dM\,\mathsf{E}(n,\ell,d,dD)+\tilde{O}(n\,d^2MD\,\kappa k\log p)+\tilde{O}(Md^{n+1}\,\kappa k\log p),$$

whenever $d\geqslant 2$.

Proof. First we verify that

$$\begin{aligned}\bar{f}(e,y,\bar{A}_1,\ldots,\bar{A}_n) &= \bar{f}_0(y,\bar{A}_1,\ldots,\bar{A}_n)+\bar{f}_1(y,\bar{A}_1,\ldots,\bar{A}_n)\,e \bmod e^2\\ &= \bar{f}_0(y,\bar{A}_{1,0},\ldots,\bar{A}_{n,0})+\left(\bar{f}_1(y,\bar{A}_{1,0},\ldots,\bar{A}_{n,0})+\sum_{k=1}^{n}\frac{\partial\bar{f}_0}{\partial x_k}(y,\bar{A}_{1,0},\ldots,\bar{A}_{n,0})\,\bar{A}_{k,1}\right)e \bmod e^2.\end{aligned}$$

Therefore, for all $i=0,\ldots,d\,(M-1)$ and $j=0,\ldots,d\,(D-1)$ we obtain

$$\begin{aligned}&\bar{f}(e,y,\bar{A}_1,\ldots,\bar{A}_n)(e,\alpha_i,\alpha_j)\\ &= \varphi_i(\beta_{i,j})+\left(\psi_i(\beta_{i,j})+\sum_{k=1}^{n}\frac{\partial\varphi_i}{\partial x_k}(\beta_{i,j})\,\bar{A}_{k,1}(\alpha_i,\alpha_j)\right)e \bmod e^2\\ &= h_0(\alpha_i,\alpha_j)+h_1(\alpha_i,\alpha_j)\,e \bmod e^2.\end{aligned}$$

Since $\bar{f}(e,y,\bar{A}_1,\ldots,\bar{A}_n) \bmod e^2$ has partial degrees $\leqslant d\,(M-1)$ in y and $\leqslant d\,(D-1)$ in t, we deduce

$$\bar{f}(e,y,\bar{A}_1,\ldots,\bar{A}_n) \bmod e^2 = h_0(y,t)+h_1(y,t)\,e,$$

whence the correctness of the algorithm.

The rewriting in step 1 takes linear time. The construction of the points in step 2 costs $\tilde{O}(d\max(D,M)\,\kappa k\log p)$. In step 3 we appeal to Lemma 2.2 in order to obtain

$$\bar{A}_1(e,\alpha_0,t),\ldots,\bar{A}_n(e,\alpha_0,t),\ldots,\bar{A}_1(e,\alpha_{d(M-1)},t),\ldots,\bar{A}_n(e,\alpha_{d(M-1)},t)$$

in time $\tilde{O}(n\,dMD\,\kappa k\log p)$, and then to obtain

$$\bar{A}_1(\alpha_i,\alpha_0),\ldots,\bar{A}_n(\alpha_i,\alpha_0),\ldots,\bar{A}_1(\alpha_i,\alpha_{d(D-1)}),\ldots,\bar{A}_n(\alpha_i,\alpha_{d(D-1)})$$

for $i=0,\ldots,d\,(M-1)$ in time $\tilde{O}(n\,d^2MD\,\kappa k\log p)$. Notice that the required data reorganizations can be done in softly linear time thanks to Lemma 2.1.

The cost of step 4 is provided by Lemma 2.11,

$$\rho\,dM\,\tilde{O}(\log^2 M+\log(dM))\,\tilde{O}(\kappa k\log p)+O(dM\,|f|_{\#}),$$

and simplifies to $\tilde{O}(Md^{n+1}\,\kappa k\log p)$ by (2.1) and Lemma 2.6.

The derivations in step 5 incur $\tilde{O}(Md^n\,\kappa k\log p)$ by Lemma 2.9. Taking into account data reorganizations *via* Lemma 2.1, the total cost of step 5 is

$$(n+2)\,dM\,\mathsf{E}(n,\ell,d,dD)+\tilde{O}(n\,d^2MD\,\kappa k\log p)+\tilde{O}(Md^n\,\kappa k\log p).$$

In step 6, the interpolations amount to $\tilde{O}(d^2MD\,\kappa k\log p)$ by Lemma 2.3. Finally step 7 takes additional time $\tilde{O}(d^2MD\,\kappa k\log p)$. □

3.2. Main complexity bound

Our main complexity bound for modular composition is presented in the next theorem, under the condition that sufficiently many evaluation points are at our disposal. For convenience we recall the following lemma.

LEMMA 3.2. [36, Lemma 4.7] *For all positive integers n, d we have*

$$\log \binom{d+n}{n} \leqslant n \log\left(1+\frac{d}{n}\right) + d \log\left(1+\frac{n}{d}\right).$$

THEOREM 3.3. *Let $\varepsilon > 0$ be a fixed rational value, assume that $d \geqslant 2$ and that*

$$p^k > \max\left(d, (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n\, d^{\varepsilon/8}\right)(\max(M, d^n)-1). \tag{3.2}$$

Let $f \in \mathbb{A}[x_1, \ldots, x_n]$ be of total degree $\leqslant d$ and let $A_1, \ldots, A_n$ be in $\mathbb{E} := \mathbb{A}[t]/(Q(t))$, with Q monic in t of degree $\leqslant d^n$. Then $f(A_1, \ldots, A_n)$ can be computed in time

$$\tilde{O}(M d^{(1+3\varepsilon)n} \kappa k \log p).$$

Proof. Let $\gamma > 0$ be such that

$$\log(1+\gamma) + \gamma \log(1+1/\gamma) \leqslant \log(1+\varepsilon).$$

First we examine the case $d \leqslant \gamma n$. Lemma 3.2 and the fact that the function

$$\gamma \mapsto \log(1+\gamma) + \gamma \log(1+1/\gamma)$$

is nondecreasing yield

$$\log \binom{d+n}{n} \leqslant n\left(\log\left(1+\frac{d}{n}\right) + \frac{d}{n}\log\left(1+\frac{n}{d}\right)\right) \leqslant (\log(1+\gamma) + \gamma \log(1+1/\gamma))\, n,$$

so we have $\binom{d+n}{n} \leqslant (1+\varepsilon)^n$. By adapting Lemma 2.12 to $\mathbb{E}$, the cost of the naive modular evaluation is $(1+\varepsilon)^n \tilde{O}(M d^n \kappa k \log p) = \tilde{O}(M d^{(1+2\varepsilon)n} \kappa k \log p)$, since $d \geqslant 2$.

From now we assume that $\gamma n < d$ holds. We set $m := \lceil 8/\varepsilon \rceil$. If d is bounded then so is n and the above adaptation of Lemma 2.12 may again be used to achieve a naive evaluation cost in $\tilde{O}(M d^n \kappa k \log p)$. Consequently, we may further assume that d is sufficiently large to satisfy

$$(3+2/\gamma)\log((3+2/\gamma)\, d) \leqslant d^{\varepsilon},\ d+1 \leqslant d^{1+\varepsilon},\ \text{and}\ m \leqslant \log d. \tag{3.3}$$

We perform the multivariate Kronecker segmentation of f into $\hat{f}$ which has $\hat{n} := mn$ variables and partial degrees $< \hat{\ell}_i$ as follows. We let

$$\ell := d+1,\ \hat{\ell}_i := \lfloor \ell^{1/m} \rfloor \text{ for } i = 1, \ldots, m-1, \text{ and } \hat{\ell}_m := \lceil \ell / (\hat{\ell}_1 \cdots \hat{\ell}_{m-1}) \rceil,$$

and introduce the Kronecker substitution map

$$\begin{aligned} K_{\hat{\ell}_1, \ldots, \hat{\ell}_m} \colon \mathrm{GR}(p^\kappa, k)[x_{1,1}, \ldots, x_{1,m}, \ldots, x_{n,1}, \ldots, x_{n,m}] &\to \mathrm{GR}(p^\kappa, k)[x_1, \ldots, x_n] \\ x_{i,j} &\mapsto x_i^{\hat{\ell}_1 \cdots \hat{\ell}_{j-1}} \text{ for } i = 1, \ldots, n,\ j = 1, \ldots, m. \end{aligned}$$

We set $\hat{f} := K^{-1}_{\hat{\ell}_1, \ldots, \hat{\ell}_m}(f)$ and $\hat{d} := (\hat{\ell}_1 + \cdots + \hat{\ell}_m)\, n - \hat{n}$. Observe that we have $\hat{\ell}_1 = \cdots = \hat{\ell}_{m-1} \leqslant \hat{\ell}_m$ and $\ell \leqslant \hat{\ell}_1 \cdots \hat{\ell}_m$. From

$$\ell^{1/m} - 1 < \hat{\ell}_i \leqslant \ell^{1/m} \text{ for } i = 1, \ldots, m-1$$

and

$$\ell / (\hat{\ell}_1 \cdots \hat{\ell}_{m-1}) \leqslant \hat{\ell}_m < \ell / (\hat{\ell}_1 \cdots \hat{\ell}_{m-1}) + 1$$

we deduce that

$$\hat{\ell}_1 \cdots \hat{\ell}_m < \ell + \ell^{(m-1)/m} = \ell\,(1+\ell^{-1/m}).$$

Still by allowing d to be sufficiently large we may further assume that

$$\hat{\ell}_1 \cdots \hat{\ell}_m \leqslant (1+\varepsilon)^m\,\ell \quad \text{and} \quad \hat{\ell}_m \leqslant (1+\varepsilon)\,d^{1/m}. \tag{3.4}$$

On the one hand, by [36, Proposition 5.1] and (3.3), the computation of $\hat{f}$ takes time

$$(1+\varepsilon)^{mn}\tilde{O}(\ell^n \kappa k \log p) = (1+\varepsilon)^{mn}\tilde{O}(d^{(1+\varepsilon)n}\kappa k \log p).$$

On the other hand, for $i = 1, \ldots, n$ we compute

$$(\hat{A}_{i,1}, \ldots, \hat{A}_{i,m}) := \left(A_i, A_i^{\hat{\ell}_1}, \ldots, A_i^{\hat{\ell}_1 \cdots \hat{\ell}_{m-1}}\right),$$

by binary powering, in time $\tilde{O}(M d^n \kappa k \log p)$. We may now compute $f(A_1, \ldots, A_n)$ as

$$f(A_1, \ldots, A_n) = \hat{f}(\hat{A}_{1,1}, \ldots, \hat{A}_{1,m}, \ldots, \hat{A}_{n,1}, \ldots, \hat{A}_{n,m}).$$

Since (3.4) ensures

$$\hat{d} \leqslant (\hat{\ell}_1 + \cdots + \hat{\ell}_m)\,n \leqslant (1+\varepsilon)\,m n d^{1/m} \leqslant (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n d^{\varepsilon/8},$$

the condition (3.2) allows us to combine Proposition 3.1 and Theorem 2.13. This yields the complexity bound

$$\begin{aligned}
& (\hat{n}+2)\,\hat{d}\,M\,\mathsf{E}(\hat{n}, \hat{\ell}, \hat{d}, \hat{d}d^n) + \tilde{O}(\hat{n}\hat{d}^2 M d^n \kappa k \log p) + \tilde{O}(M \hat{d}^{n+1} \kappa k \log p) \\
\leqslant\ & (\hat{n}+2)\,\hat{d}\,M\,\mathsf{E}(\hat{n}, \hat{\ell}, \hat{d}, \hat{d}d^n) + \tilde{O}(\hat{d}^2 M d^n \kappa k \log p) \\
\leqslant\ & (1+\varepsilon)^{mn}\,\hat{d}\,M\,(\hat{d}d^n + (mn\,\hat{\ell}_m \log(mn\,\hat{\ell}_m))^{mn})\,\tilde{O}((mn)^{10}\,\hat{\ell}_m^7 \kappa k \log p) \\
& + \tilde{O}(\hat{d}^2 M d^n \kappa k \log p) \\
=\ & (1+\varepsilon)^{mn}\,M\,(d^{1/m} + (mn\,(1+\varepsilon)\log((1+\varepsilon)\,mn\,d^{1/m}))^{mn}) \\
& \times\ \tilde{O}(d^{n+\varepsilon}\kappa k \log p) \qquad \text{(using (3.4))} \\
\leqslant\ & (1+\varepsilon)^{mn}\,M\,(d^{1/m} + (cn^c \log d)^{mn})\,\tilde{O}(d^{n+\varepsilon}\kappa k \log p),
\end{aligned}$$

for a suitable constant c. If $d^{\varepsilon n} \geqslant (cn^c \log d)^{mn}$, then this bound further simplifies to

$$(1+\varepsilon)^{mn}\tilde{O}(M d^{(1+\varepsilon)n+\varepsilon}\kappa k \log p).$$

Otherwise we have $d^{\varepsilon n} < (cn^c \log d)^{mn}$ which implies that $d = n^{O(1)}$. In this case, the condition (3.2) allows us to combine Proposition 3.1 and Theorem 2.14, in order to obtain $f(A_1, \ldots, A_n)$ in time

$$\begin{aligned}
& (n+2)\,d\,M\,\mathsf{E}(n, \ell, d, dd^n) + \tilde{O}(d^2 M d^n \kappa k \log p) \\
\leqslant\ & (n+2)(1+\varepsilon)^n\,d\,M\,(d^{n+1} + ((3d+2n)\log(3d+2n))^n)\,\tilde{O}(n^2 d\,(d+n)^6 \kappa k \log p) \\
& + \tilde{O}(M d^{n+2}\kappa k \log p) \\
=\ & (1+\varepsilon)^n\,((3+2/\gamma)\log((3+2/\gamma)\,d))^n\,\tilde{O}(M d^n \kappa k \log p),
\end{aligned}$$

which simplifies to $(1+\varepsilon)^n\tilde{O}(M d^{(1+\varepsilon)n}\kappa k \log p)$ thanks to (3.3).

Overall we have proved a complexity bound $(1+\varepsilon)^{mn}\tilde{O}(M d^{(1+2\varepsilon)n}\kappa k \log p)$. Taking into account that $m \leqslant \log d$, from (3.3), it follows that $(1+\varepsilon)^{mn} \leqslant d^{\varepsilon n}$, which concludes the proof. □

3.3. Extension of the residue field

In order to generalize Theorem 3.3 to small residue fields, namely whenever inequality (3.2) does not hold, we extend the cardinality of the ground ring. We will rely on the following construction.

LEMMA 3.4. *Let* $\mathrm{GR}(p^\kappa,k) = (\mathbb{Z}/p^\kappa\mathbb{Z})[z]/(\theta(z))$ *be a Galois ring, and let* l *be an integer that is coprime with* k. *Then we may compute a monic irreducible polynomial* $\rho \in \mathbb{F}_p[y]$ *of degree* l *in time* $\tilde{O}(l^2 \log^2 p)$ *with any fixed arbitrarily low probability of failure, or in expected time* $\tilde{O}(l^2 \log^2 p)$ *with a Las Vegas algorithm. Then we may build the Galois ring* $\mathrm{GR}(p^\kappa, k\,l)$ *as* $(\mathbb{Z}/p^\kappa\mathbb{Z})[z]/(\nu(z))$, *such that the following isomorphism holds:*

$$\begin{aligned} \Phi\colon (\mathbb{Z}/p^\kappa\mathbb{Z})[y,z]/(\rho(y),\theta(z)) &\to (\mathbb{Z}/p^\kappa\mathbb{Z})[u]/(\nu(u)) \\ y &\mapsto \gamma(u) \\ z &\mapsto \eta(u) := u/\gamma(u), \end{aligned}$$

for some polynomials γ *and* η *in* $(\mathbb{Z}/p^\kappa\mathbb{Z})[u]_{<kl}$. *The polynomials* ν, γ *and* η *can be computed in time* $\tilde{O}(l^{\omega+2}\kappa k \log p)$. *Each application of* Φ *and* Φ^{-1} *takes time* $\tilde{O}(l^2\kappa k \log p)$.

Proof. The polynomial ρ can be obtained by using Ben-Or's randomized algorithm [21, Theorem 14.42]. Since k and l are coprime, a natural candidate for $\nu(z)$ is the *composed product* of θ and ρ (usually written $\theta \odot \rho$),

$$\nu(z) := \prod_{\theta(\zeta)=0} \prod_{\rho(\xi)=0} (z - \zeta\,\xi) = \prod_{\rho(\xi)=0} \xi^k \theta(\xi^{-1} z). \tag{3.5}$$

It is well known that ν is irreducible of degree $k\,l$. Let $\tilde{\theta}(u,y) := \theta(y^{-1}u) \bmod \rho(y)$, that can be computed in time $\tilde{O}(l\,\kappa k \log p)$. We construct ν as the monic part of the resultant $\mathrm{Res}_y(\rho(y), \tilde{\theta}(u,y))$ in y and we write $A(u)\,y - B(u)$ for the corresponding subresultant of degree 1 in y. For subresultants over Galois rings, we need to compute the necessary minors of the Sylvester matrix by means of Berkowitz' algorithm [5]. In this manner ν, A, and B can be obtained in time $\tilde{O}(l^{\omega+2}\kappa k \log p)$.

It is well known that A is invertible modulo ν, so we compute $\gamma(u) := A(u)^{-1}B(u) \bmod \nu(u)$ and $\eta(u) := u/\gamma(u)$ in time $\tilde{O}(l\,\kappa k \log p)$.

Let us consider an element $\alpha \in \mathrm{GR}(p^\kappa,k)$ represented by $a(z) \in (\mathbb{Z}/p^\kappa\mathbb{Z})[z]_{<k}$, and

$$A(u) := \mathrm{Tr}_{\mathrm{GR}(p^\kappa,l)/(\mathbb{Z}/p^\kappa\mathbb{Z})}\left(\left(a(y^{-1}u)\left(\frac{\nu(u)}{\tilde{\theta}(u,y)}\right)^{-1} \bmod \tilde{\theta}(u,y)\right)\frac{\nu(u)}{\tilde{\theta}(u,y)}\right),$$

where $\mathrm{GR}(p^\kappa,l) = (\mathbb{Z}/p^\kappa\mathbb{Z})[y]/(\rho(y))$. Equivalently, we have

$$A(u) = \sum_{\rho(\xi)=0} \left(a(\xi^{-1}u)\left(\frac{\nu(u)}{\theta(\xi^{-1}u)}\right)^{-1} \bmod \theta(\xi^{-1}u)\right)\frac{\nu(u)}{\theta(\xi^{-1}u)},$$

which corresponds to the usual Chinese remaindering formula associated to the factorization (3.5) of ν.

We observe that $A(u) \operatorname{rem} \theta(\xi^{-1}u) = a(\xi^{-1}u)$ for all roots ξ of ρ. Consequently, for all roots $\zeta\,\xi$ of ν with $\theta(\zeta) = 0$, we have $A(\zeta\,\xi) = a(\zeta)$. It follows that A actually represents $\Phi(\alpha)$. It can be computed in time $\tilde{O}(l^2\kappa k \log p)$.

In the other direction, given a preimage $A(u)$ of $a \in (\mathbb{Z}/p^\kappa\mathbb{Z})[u]/(\nu(u))$, we obtain $\Phi^{-1}(a)$ as $A(yz) \bmod (\rho(y),\theta(z))$, in time $\tilde{O}(l^2\kappa k \log p)$. □

Remark 3.5. In the latter proof we could have appealed to faster algorithms to build irreducible polynomials, such as the ones of [16, 60, 67]. In addition, faster algorithms are known to obtain ν in specific cases; see [9, 28, 51]. For simplicity we have preferred to restrict to general well known algorithms, because the complexity bound of Lemma 3.4 is to be used only when l is rather small.

The following corollary aims at completing Theorem 3.3 for small residue fields. It will not be used in the sequel because the extension of the residue field will instead be handled at the top level of the Kronecker solver.

COROLLARY 3.6. *Let $\varepsilon > 0$ be a fixed rational value, let $\mathbb{A}$ be as in (3.1), and assume that $d \geqslant 2$. Let $f \in \mathbb{A}[x_1, \ldots, x_n]$ be of total degree $\leqslant d$ and let $A_1, \ldots, A_n$ be in $\mathbb{E} := \mathbb{A}[t]/(Q(t))$, with Q monic in t of degree $\leqslant d^n$. Then $f(A_1, \ldots, A_n)$ can be computed using a probabilistic Las Vegas algorithm in expected time*

$$\tilde{O}(M d^{(1+\varepsilon)n} \kappa k \log p).$$

Proof. Notice that the total degree of f may be obtained in time $\tilde{O}(M d^n \kappa k \log p)$ by Lemmas 2.6, 2.8 and inequality (2.1). It remains to handle the case when (3.2) does not hold in Theorem 3.3. In other words, assume that

$$k \log p = O(n \log d + \log M).$$

We first compute the smallest integer $\underline{l}$ such that

$$p^{k\underline{l}} > \max\left(d, (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n d^{\varepsilon/8}\right)(\max(M, d^n) - 1),$$

that is

$$\underline{l} := \left\lceil \frac{\log\left(\max\left(d, (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n d^{\varepsilon/8}\right)(\max(M, d^n)-1)\right)}{k \log p} \right\rceil = O\left(\frac{n \log d + \log M}{k \log p}\right),$$

and we set

$$l := \min\{ik + 1 : i \in \mathbb{N}, ik + 1 \geqslant \underline{l}\},$$

so l is coprime to k and

$$kl \log p = O((n \log d + \log M)^2).$$

We next apply Lemma 3.4, from which we borrow the notation: $\rho \in \mathbb{F}_p[z]$ of degree l is obtained in expected time $\tilde{O}(l^2 \log^2 p) = \tilde{O}((n \log d + \log M)^2)$; the computation of ν, γ, and η requires time $\tilde{O}(l^{\omega+2} \kappa k \log p) = \tilde{O}((n \log d + \log M)^{2(\omega+2)} \kappa)$.

With this construction in hand we set

$$\bar{\mathbb{A}} := \mathrm{GR}(p^\kappa, kl)[e, y]/(e^2, y^M),$$

and

$$\bar{\mathbb{E}} := \bar{\mathbb{A}}[t]/(Q(t)),$$

and we proceed as follows for the modular composition problem:

- We cast Q into $\bar{\mathbb{A}}[t]$ in time $\tilde{O}(M d^n l^2 \kappa k \log p) = \tilde{O}(M d^n \kappa)$;
- We cast f into $\bar{\mathbb{A}}[x_1, \ldots, x_n]$ in time $\tilde{O}(M d^n l^2 \kappa k \log p) = \tilde{O}(M d^n \kappa)$;
- We cast $A_1, \ldots, A_n$ into $\bar{\mathbb{E}}$ in time $\tilde{O}(n M d^n l^2 \kappa k \log p) = \tilde{O}(M d^n \kappa)$;

- By Theorem 3.3, we can evaluate $f(A_1,\ldots,A_n)$ in $\bar{\mathbb{E}}$ in time

$$\tilde{O}(M d^{(1+3\varepsilon)n} \kappa k \log p).$$

- We cast $f(A_1,\ldots,A_n)$ into $\mathbb{E}$ in time $\tilde{O}(M d^n \kappa)$.

Adding up, we obtain the claimed complexity bound—up to replacing ε by $\varepsilon/3$ from the outset. □

4. DATA STRUCTURES AND RANDOMNESS ISSUES

Consider a polynomial system $f_1 = \cdots = f_n = 0$ over an effective field $\mathbb{K}$, which satisfies the regularity assumptions R_1, R_2, and R_3. In the next section, we will recall the Kronecker algorithm that solves such a system. But before, we prepare the terrain by presenting a few concepts and data structures that will be necessary, and by discussing how to put the system into a sufficiently general position *via* a random linear change of coordinates.

Under our regularity assumptions, and letting $I_i := (f_1,\ldots,f_i)$, the coordinate ring $\mathbb{B}_i := \mathbb{K}[x_0,\ldots,x_n]/I_i$ has dimension $r = n-i$ and degree $D_i := d_1 \cdots d_i$ by the Bézout theorem. In particular the system $f_1 = \cdots = f_n = 0$ admits D_n distinct solutions in $\mathbb{P}^n$, which are all regular. After applying a generic affine change of coordinates, the Kronecker solver first solves the system $f_1 = 0$ then $f_1 = f_2 = 0$, ..., and so on until $f_1 = \cdots = f_n = 0$. We first study how to perform this change of variables. Next we explain how positive dimensional solution sets of the intermediate systems $f_1 = \cdots = f_i = 0$ are represented.

4.1. Noether position

An homogeneous ideal I of $\mathbb{K}[x_0,\ldots,x_n]$ is said to be in *Noether position* when $I \cap \mathbb{K}[x_0,\ldots,x_r] = (0)$ and $\mathbb{B} := \mathbb{K}[x_0,\ldots,x_n]/I$ is an integral ring extension of $\mathbb{K}[x_0,\ldots,x_r]$, where $r := \dim \mathbb{B}$. Let X be the column vector with entries $x_0,\ldots,x_n$. If M is an invertible $(n+1)\times(n+1)$ matrix over $\mathbb{K}$, then we write

$$I \circ M := (f(MX) : f \in I).$$

If I is generated by a proper regular sequence $f_1,\ldots,f_n$, then we say that a matrix M puts the intermediate ideals $I_i \circ M$ in *simultaneous Noether position* if $I_i \circ M$ is in Noether position for all $i = 1,\ldots,n$. Let us now examine the probability that this happens for an upper triangular matrix M with ones on the diagonal and random entries above the diagonal.

LEMMA 4.1. *Let $f_1,\ldots,f_n$ satisfy R_1, R_2, and R_3. Given a finite subset S of $\mathbb{K}$ of cardinality $|S|$, consider the matrix*

$$M := \begin{pmatrix} 1 & \alpha_{1,0} & \alpha_{2,0} & \cdots & \alpha_{n-1,0} & \alpha_{n,0} \\ & 1 & \alpha_{2,1} & & \vdots & \\ & & 1 & \ddots & & \vdots \\ & & & \ddots & \alpha_{n-1,n-2} & \\ & & & & 1 & \alpha_{n,n-1} \\ & & & & & 1 \end{pmatrix}$$

where the $\alpha_{i,j}$ are taken at random in S. Then the probability that M puts $I_1 \circ M,\ldots,I_n \circ M$ into simultaneous Noether position is at least $1 - \frac{2D_n}{|S|}$.

Proof. We use the incremental method described in the proof of [18, Theorem 1.9] as follows (for historical references, we also refer to [18]). The values $(\alpha_{n,0},\dots,\alpha_{n,n-1})$ are taken such that the coefficient of $x_n^{D_1}$ in $f_1(x_0+\alpha_{n,0}x_n,\dots,x_{n-1}+\alpha_{n,n-1}x_n,x_n)$ is non-zero. For such values the ideal $I_1\circ M$ is in Noether position.

Then we consider the determinant c_2 of the multiplication endomorphism by f_2 in $\mathbb{K}(x_0,\dots,x_{n-1})[x_n]/(I_1\circ M)$: it belongs to $\mathbb{K}[x_0,\dots,x_{n-1}]\cap(I_2\circ M)$ and has degree D_2. The values $(\alpha_{n-1,0},\dots,\alpha_{n-1,n-2})$ such that the coefficient of $x_{n-1}^{D_2}$ in $c_2(x_0+\alpha_{n-1,0}x_{n-1},\dots,x_{n-2}+\alpha_{n-1,n-2}x_{n-1},x_{n-1})$ is non-zero put $I_2\circ M$ into Noether position.

By induction we consider the determinant c_{i+1} of the multiplication endomorphism by f_{i+1} in $\mathbb{K}(x_0,\dots,x_{n-i})[x_{n-i+1},\dots,x_n]/(I_i\circ M)$: it belongs to $\mathbb{K}[x_0,\dots,x_{n-i-1}]\cap(I_{i+1}\circ M)$ and has degree D_{i+1}. The values $(\alpha_{n-i,0},\dots,\alpha_{n-i,n-i-1})$ such that the coefficient of $x_{n-i}^{D_{i+1}}$ in $c_{i+1}(x_0+\alpha_{n-i,0}x_{n-i},\dots,x_{n-2}+\alpha_{n-i,n-i-1}x_{n-i},x_{n-i})$ is non-zero put $I_{i+1}\circ M$ into Noether position. By taking into account the fact that $D_{i+1}\geqslant 2D_i$, the Schwartz–Zippel lemma [66, 69] leads to a probability of success at least

$$\left(1-\frac{D_1}{|S|}\right)\left(1-\frac{D_2}{|S|}\right)\cdots\left(1-\frac{D_n}{|S|}\right)\geqslant 1-\frac{1}{|S|}(D_1+\cdots+D_n)\geqslant 1-\frac{2D_n}{|S|}.$$ □

4.2. Primitive element

Let I be an absolutely radical ideal of $\mathbb{K}[x_1,\dots,x_n]$ such that the coordinate ring $\mathbb{B}:=\mathbb{K}[x_1,\dots,x_n]/I$ is zero dimensional—here we consider the affine case when I is not necessarily homogeneous. A polynomial $u\in\mathbb{K}[x_1,\dots,x_n]$ is said to be a *primitive element* for $\mathbb{B}$ if the projections of its powers generate $\mathbb{B}$ as a $\mathbb{K}$-vector space. Let D represent the degree of $\mathbb{B}$ (i.e. its dimension as a $\mathbb{K}$-vector space).

LEMMA 4.2. *Under the above assumptions, given a finite subset S of $\mathbb{K}$, the probability that a random vector $(\lambda_1,\dots,\lambda_n)\in S^n$ induces a primitive element $u=\lambda_1x_1+\cdots+\lambda_nx_n$ of $\mathbb{B}$ is $>1-2D^2/|S|$.*

Proof. The minimal polynomial $q(t)$ of u in $\mathbb{B}$ actually belongs to $\mathbb{K}[\lambda_1,\dots,\lambda_n][t]$ when the λ_i are regarded as parameters, and the total degree in the λ_i is $\leqslant D$; see for instance [18, Corollary 3.4]. The points $(\lambda_1,\dots,\lambda_n)$ to be avoided are the zeros of the discriminant of q, seen as a polynomial of total degree $\leqslant D(2D-1)$ in $\lambda_1,\dots,\lambda_n$. We conclude by the aforementioned Schwartz–Zippel lemma. □

For a primitive element u there exist unique polynomials $q,v_1,\dots,v_n\in\mathbb{K}[t]$ such that $\deg q=D$, q is monic and separable, $\deg v_i<D$, and

$$I=(q(u),x_1-v_1(u),\dots,x_n-v_n(u)).$$

The polynomials $q,v_1,\dots,v_n$ are called the *parametrization* of I by u. Equivalently, we may define $w_i=q'v_i \operatorname{rem} q$ for $i=1,\dots,n$, whence

$$I=(q(u),q'(u)x_1-w_1(u),\dots,q'(u)x_n-w_n(u)).$$

These polynomials $q,w_1,\dots,w_n$ are called the *Kronecker parametrization* of I by u; such a parametrization is uniquely determined by u.

LEMMA 4.3. *Let $f_1, \ldots, f_n$ satisfy* R_1, R_2, *and* R_3 *and such that the I_i are simultaneously in Noether position for $i = 1, \ldots, n$. Given a finite subset S of $\mathbb{K}$, the probability that a vector $(\lambda_1, \ldots, \lambda_n) \in S^n$ induces a primitive element $u = \lambda_1 x_1 + \cdots + \lambda_n x_n$ common to all $\mathbb{E}_i := \mathbb{K}(x_0, \ldots, x_{n-i})[x_{n-i+1}, \ldots, x_n]/I_i$ for $i = 1, \ldots, n$ is $> 1 - 4D_n^2/|S|$.*

Proof. Each $\mathbb{E}_i$ has dimension D_i as a $\mathbb{K}(x_0, \ldots, x_{n-i})$-vector space [18, section 2.4 and the relationship to the geometric definition of the degree of an algebraic variety in Corollary 3.10]. By the previous lemma and the fact that $D_{i+i} \geqslant 2D_i$, the probability that u is primitive for all $\mathbb{B}_i$ is at least

$$\left(1 - \frac{2D_1^2}{|S|}\right)\left(1 - \frac{2D_2^2}{|S|}\right)\cdots\left(1 - \frac{2D_n^2}{|S|}\right) \geqslant 1 - \frac{2}{|S|}(D_1^2 + \cdots + D_n^2) > 1 - \frac{4D_n^2}{|S|}. \qquad \square$$

4.3. Lifting fiber

Let $f_1, \ldots, f_n$ satisfy R_1, R_2, and R_3, be in simultaneous Noether position, and let

$$\mathbb{E}_i := \mathbb{K}(x_0, \ldots, x_{n-i})[x_{n-i+1}, \ldots, x_n]/I_i$$

for $i = 1, \ldots, n$. Thanks to Noether positions, the system $f_1 = \cdots = f_n = 0$ has all its roots in the affine space. More generally given a point $(a_1, \ldots, a_{n-i}) \in \mathbb{K}^{n-i}$ all the roots of $I_i + (x_1 - a_1 x_0, \ldots, x_{n-i} - a_{n-i} x_0)$ are in the affine space. A point $(a_1, \ldots, a_{n-i}) \in \mathbb{K}^{n-i}$ is called a *lifting point* if $I_i + (x_1 - a_1 x_0, \ldots, x_{n-i} - a_{n-i} x_0)$ is absolutely radical.

LEMMA 4.4. *Under the latter conditions, given a finite subset S of $\mathbb{K}$, the probability that a vector $(a_1, \ldots, a_n) \in S^n$ is a simultaneous lifting point for $I_1, \ldots, I_n$ is $> 1 - 4D_n^2/|S|$.*

Proof. Without loss of generality we may assume that $\mathbb{K}$ is algebraically closed, so we may consider a primitive element u that is common to all $\mathbb{E}_i$, thanks to Lemma 4.3. The minimal polynomial $q(t)$ of u in $\mathbb{E}_i$ is homogeneous in $\mathbb{K}[x_0, \ldots, x_{n-i}, t]$ of degree D_i, and is monic and separable in t. Any point $(a_1, \ldots, a_{n-i})$ such that the specialization of q at $x_1 = a_1, \ldots, x_{n-i} = a_{n-i}$ is separable ensures that $I_i + (x_1 - a_1 x_0, \ldots, x_{n-i} - a_{n-i} x_0)$ is absolutely radical; see for instance [18, Proposition 3.6]. The probability of picking such a point at random in S^{n-i} is at least $1 - 2D_i^2/|S|$ by the Schwartz–Zippel lemma. The probability that $(a_1, \ldots, a_n) \in S^n$ satisfies the requested property is thus at least

$$\left(1 - \frac{2D_1^2}{|S|}\right)\left(1 - \frac{2D_2^2}{|S|}\right)\cdots\left(1 - \frac{2D_n^2}{|S|}\right) \geqslant 1 - \frac{2}{|S|}(D_1^2 + \cdots + D_n^2) > 1 - \frac{4D_n^2}{|S|},$$

again by using $D_{i+1} \geqslant 2D_i$. $\square$

4.4. Random parameter summary

Assume that we are given a polynomial system $f_1 = \cdots = f_n = 0$ that satisfies R_1, R_2 and R_3. In order the make the Kronecker algorithm work, the following conditions are required for $i = 1, \ldots, n$:

V_1. I_i is in Noether position,

V_2. the ideal $\dot{I}_i := I_i + (x_0 - 1, x_1, \ldots, x_{n-i})$ of $\mathbb{K}[x_{n-i+1}, \ldots, x_n]$ is absolutely radical.

LEMMA 4.5. *Let $f_1,\dots,f_n$ satisfy* R_1, R_2 *and* R_3, *and let S be a finite subset of $\mathbb{K}^n$. If we take $(\alpha_{i,j})_{0\leqslant i<j\leqslant n}$ in $S^{n(n+1)/2}$ at random, as well as $(a_1,\dots,a_n)$ in S^n, and if we let*

$$N := \begin{pmatrix} 1 & \alpha_{1,0} & \alpha_{2,0} & \cdots & \alpha_{n-1,0} & \alpha_{n,0} \\ & 1 & \alpha_{2,1} & & \vdots & \\ & & 1 & \ddots & & \vdots \\ & & & \ddots & \alpha_{n-1,n-2} & \\ & & & & 1 & \alpha_{n,n-1} \\ & & & & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ a_1 & 1 & & \\ \vdots & & \ddots & \\ a_n & & & 1 \end{pmatrix},$$

then V_1 *and* V_2 *are satisfied with probability* $>1-5\,D_n^2/|S|$ *after replacing $f_1,\dots,f_n$ by $f_1\circ N,\dots,f_n\circ N$.*

Proof. We first pick up a matrix M at random as in Lemma 4.1 in order to get simultaneous Noether positions. Then comes the choice of the lifting point, for which we use the strategy presented in the proof of Lemma 4.4. □

4.5. A posteriori verification

At the end of the resolution, unless the execution has failed, we expect to obtain an invertible $(n+1)\times(n+1)$ matrix N over $\mathbb{K}$ and polynomials $q, v_1, \dots, v_n$ such that q is separable of degree D_n, the v_j have degree $<D_n$, and

$$I\circ N + (x_0-1) = (q(u), x_0-1, x_1-v_1(u), \dots, x_n-v_n(u)).$$

On the other hand, I being absolutely radical, its roots are all regular, q is separable, and the value of the Jacobian matrix of $f_1\circ N,\dots,f_n\circ N$ in $x_1,\dots,x_n$ at $1, v_1(t),\dots,v_n(t)$ is invertible modulo $q(t)$; see for instance [18, section 4.2].

One natural question at the end of a probabilistic resolution process is to determine whether the computed solution set is correct or not. In the case of sufficiently generic systems, namely satisfying R_1 and R_2, it is sufficient to verify that D_n regular solutions have been found. This idea is formalized in the following proposition, which turns a probabilistic resolution algorithm into a Las Vegas one.

PROPOSITION 4.6. *Consider homogeneous polynomials $f_1,\dots,f_n$ in $\mathbb{K}[x_0,\dots,x_n]$, an invertible $(n+1)\times(n+1)$ matrix N, a linear form $u\in\mathbb{K}[x_1,\dots,x_n]$, and polynomials $q, v_1,\dots,v_n$ in $\mathbb{K}[t]$ that satisfy the following conditions:*

- *q is separable of degree D_n, the v_j have degree $<D_n$,*
- *$u(v_1(t),\dots,v_n(t)) = t$,*
- *$(f_i\circ N)(1, v_1(t),\dots,v_n(t)) = 0 \operatorname{rem} q(t)$ for all $i=1,\dots,n$,*
- *the Jacobian matrix in $x_1,\dots,x_n$ of $f_1\circ N,\dots,f_n\circ N$ is invertible at $(1, v_1(t),\dots,v_n(t))$ modulo $q(t)$.*

Then, R_1 *and* R_2 *are satisfied, the ideal $I\circ N = (f_1\circ N,\dots,f_n\circ N)$ is in Noether position, and q, $v_1,\dots,v_n$ form a parametrization of $I\circ N + (x_0-1)$ by u.*

Let $\varepsilon>0$ be a fixed rational value, and assume that R_3 *holds. If $\mathbb{K}=\mathbb{F}_{p^k}$ and if*

$$p^k > \max\left(\bar{d}, (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n\,\bar{d}^{\varepsilon/8}\right)(\bar{d}^n-1),$$

then the above conditions can be checked in time $\tilde{O}(\bar{d}^{(1+3\varepsilon)n} k \log p)$.

Proof. The assumptions ensure that the system $f_1 \circ N = \cdots = f_n \circ N = 0$ admits D_n distinct isolated regular solutions in the affine subspace defined by $x_0 = 1$. By Bézout's theorem, this system has therefore no solution at infinity, namely in the hyperplane defined by $x_0 = 0$.

For $i = 1, \ldots, n$, let $\mathcal{V}(f_1, \ldots, f_i)$ denote the variety of zeros of $f_1 = \cdots = f_i = 0$. Let us first show that R_1 is satisfied. If it were not, then there would exist a smallest index $i < n$ for which $\dim \mathcal{V}(f_1, \ldots, f_i) = n - i$ and $\dim \mathcal{V}(f_1, \ldots, f_{i+1}) = n - i$. In other words $\mathcal{V}(f_1, \ldots, f_{i+1})$ would have a (non-empty) equidimensional component W of dimension $n - i$ and a possibly empty component W' of dimension $n - i - 1$. Heintz' version of the Bézout theorem [32, Theorem 1] would imply that

$$\deg W + \deg W' \leqslant D_i d_{i+1} = D_{i+1},$$

whence $\deg W' < D_{i+1}$. Therefore the number of isolated roots of $f_1 = \cdots = f_n = 0$ would be

$$\deg(W' \cap \mathcal{V}(f_{i+2}, \ldots, f_n)) < D_{i+1} d_{i+2} \cdots d_n \leqslant D_n,$$

which contradicts the fact that $f_1 = \cdots = f_n = 0$ admits D_n distinct isolated regular solutions.

Let i be in {1,...,n}. We consider a non-zero homogeneous polynomial $g(x_0, x_i)$ in $I \circ N$. We write g into $g(x_0, x_i) = c(x_0)\, h(x_0, x_i)$ with h primitive as a polynomial of $\mathbb{K}[x_0][x_i]$ and $c(x_0)$ a monomial in x_0. In particular h is monic in x_i and h vanishes at all the points of $\mathcal{V}(I \circ N)$, so it belongs to $\sqrt{I \circ N}$. This shows that the class of x_i in $\mathbb{K}[x_0, \ldots, x_n] / (I \circ N)$ is integral over $\mathbb{K}[x_0]$ when seen as an element of $\mathbb{K}[x_0, \ldots, x_n] / (I \circ N)$. We deduce that $I \circ N$ is in Noether position; for instance by using [18, Theorem 1.12].

Now let us show that R_2 is satisfied. Consider an irreducible component W of $\mathcal{V}(f_1, \ldots, f_i)$ for some index $i \leqslant n$ and then a point P in $W \cap \mathcal{V}(f_{i+1}, \ldots, f_n)$, which is non-empty in the projective setting. This point P is a solution of $f_1 = \cdots = f_n = 0$, and as such, the value of the Jacobian matrix of $f_1, \ldots, f_n$ at P has full rank by hypothesis. Therefore, the Jacobian matrix of $f_1, \ldots, f_i$ has full rank over a Zariski dense subvariety of W. It follows that the primary component associated to W is necessarily prime. On the other hand, R_1 implies that I_i is unmixed by [53, chapter 7, section 17]; in other words, I_i has no embedded primary ideal. Consequently I_i is absolutely radical.

Assume now that $\mathbb{K} = \mathbb{F}_{p^k}$, and let us examine complexities. Testing the separability of q boils down to computing its gcd with q', which takes time $\tilde{O}(D_n k \log p)$. We compute $f_1 \circ N, \ldots, f_n \circ N$ in time $\tilde{O}(\bar{d}^n k \log p)$ by Proposition 2.15. Then we compute the Jacobian matrix J in $x_1, \ldots, x_n$ of $f_1 \circ N, \ldots, f_n \circ N$ in time $\tilde{O}(\bar{d}^n k \log p)$ by Lemma 2.9. The evaluations of all the $f_i \circ N$ and of J at $(1, v_1(t), \ldots, v_n(t))$ in $\mathbb{F}_{p^k}[t] / (q(t))$ can be done in time $\tilde{O}(\bar{d}^{(1+3\varepsilon)n} k \log p)$ by means of Theorem 3.3—recall that these homogeneous polynomials to be evaluated are already represented by their specialization at $x_0 = 1$. Finally the determinant of the latter value of J is obtained without division thanks to Berkowitz' algorithm [5] in time $n^{\omega+1} \tilde{O}(D_n k \log p)$. Testing the invertibility of this determinant simply requires additional time $\tilde{O}(D_n k \log p)$. □

5. THE KRONECKER SOLVER

Let $\mathbb{K}$ be an effective field and let $f_1, \ldots, f_n$ be homogeneous polynomials in $\mathbb{K}[x_0, \ldots, x_n]$. Throughout this section, we assume that conditions R_1, R_2, R_3, V_1, and V_2 are satisfied. The Kronecker solver is incremental in the number of equations to be solved. More precisely, at stage i, we assume that a parametrization of the ideal

$$\dot{I}_i \;=\; (f_1, \ldots, f_i) + (x_0 - 1, x_1, \ldots, x_{n-i}) \;\subseteq\; \mathbb{K}[x_{n-i+1}, \ldots, x_n]$$

by a primitive element $u = \lambda_{n-i+1} x_{n-i+1} + \cdots + \lambda_n x_n$ is given, so that

$$\dot{I}_i \;=\; (q(u), x_{n-i+1} - v_{n-i+1}(u), \ldots, x_n - v_n(u)).$$

We will build primitive elements for all intermediate ideals from a unique tuple $(\lambda_1, \ldots, \lambda_n) \in \mathbb{K}^n$. In order to obtain a similar parametrization for $\dot{I}_{i+1}$, we apply the two following steps.

Lifting step. The parametrization of $\dot{I}_i$ is lifted into a parametrization of

$$\tilde{I}_i \;:=\; (f_1, \ldots, f_i) + (x_0 - 1, x_1, \ldots, x_{n-i-1}) \;\subseteq\; \mathbb{K}(x_{n-i})[x_{n-i+1}, \ldots, x_n]$$

of the form

$$\tilde{I}_i \;=\; (\tilde{q}(u), \tilde{q}'(u)\, x_{n-i+1} - \tilde{w}_{n-i+1}(u), \ldots, \tilde{q}'(u)\, x_n - \tilde{w}_n(u))$$

where $\tilde{q}$ is a monic polynomial of degree D_i in $\mathbb{K}(x_{n-i})[t]$, and the $\tilde{w}_i$ are in $\mathbb{K}(x_{n-i})[t]_{<D_i}$. It turns out that $\tilde{q}, \tilde{w}_{n-i+1}, \ldots, \tilde{w}_n$ actually belong to $\mathbb{K}[x_{n-i}][t]$ and have total degrees $\leqslant D_i$; see for instance [18, Corollary 3.4]. We will compute them by a variant of the Newton–Hensel lifting strategy over $\mathbb{K}[[x_{n-i}]]$.

Intersection step. Geometrically speaking $\tilde{I}_i$ represents the affine curve $\mathcal{V}(I_i) \cap \mathcal{V}(x_0 - 1, x_1, \ldots, x_{n-i-1})$. The intersection of this curve with the hypersurface $\mathcal{V}(f_{i+1})$ corresponds to $\mathcal{V}(\dot{I}_{i+1})$. The minimal polynomial of $U := \lambda_{n-i} x_{n-i} + \cdots + \lambda_n x_n$ modulo $\dot{I}_{i+1}$ will be computed as a resultant, and the rest of the parametrization of $\dot{I}_{i+1}$ is completed by means of a suitable deformation technique. We define $Q, V_{n-i}, \ldots, V_n$ to be such that this parametrization is of the form

$$\dot{I}_{i+1} \;=\; (Q(U), x_{n-i} - V_{n-i}(U), \ldots, x_n - V_n(U)).$$

5.1. Lifting step

The lifting step essentially relies on a multivariate variant of the Hensel lemma (or, equivalently, on a multivariate power series analogue of Newton's method). The Jacobian matrix of $f_1, \ldots, f_i$ in $x_{n-i+1}, \ldots, x_n$ is written

$$J_i := \begin{pmatrix} \frac{\partial f_1}{\partial x_{n-i+1}} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_i}{\partial x_{n-i+1}} & \cdots & \frac{\partial f_i}{\partial x_n} \end{pmatrix}.$$

The identity matrix of size $i \times i$ is written Id_i.

Algorithm 5.1

Input. $f_1, \ldots, f_i \in \mathbb{K}[x_0, \ldots, x_n]$ and the parametrization $q, v_{n-i+1}, \ldots, v_n \in \mathbb{K}[t]$ of $\dot{I}_i$ by $u = \lambda_{n-i+1} x_{n-i+1} + \cdots + \lambda_n x_n$.

Output. The Kronecker parametrization $\tilde{q}, \tilde{w}_{n-i+1}, \ldots, \tilde{w}_n$ in $\mathbb{K}(x_{n-i})[t]$ of $\tilde{I}_i$ by u.

Assumptions. $f_1, \ldots, f_n$ satisfy R_1, R_2, R_3, V_1, V_2.

1. Initialize $M := 1$, $\tilde{q} := q$, $\tilde{v}_{n-i+1} := v_{n-i+1}, \ldots, \tilde{v}_n := v_n$.
2. Initialize B with the inverse of $A := J_i(1, 0, \ldots, 0, v_{n-i+1}(t), \ldots, v_n(t))$ modulo $q(t)$.
3. While $M \leqslant D_i + 1$ do the following:
   a. update $A := J_i(1, 0, \ldots, 0, x_{n-i}, \tilde{v}_{n-i+1}(t), \ldots, \tilde{v}_n(t))$ rem $\tilde{q}(t)$
      and $B := B - B\,(A\,B - \mathrm{Id}_i)$ rem $\tilde{q}(t)$ over $\mathbb{K}[[x_{n-i}]] / (x_{n-i}^M)$;
   b. set $M := \min(2M, D_i + 1)$;

c. compute $\begin{pmatrix} \hat{v}_{n-i+1}(t) \\ \vdots \\ \hat{v}_n(t) \end{pmatrix} := \begin{pmatrix} \tilde{v}_{n-i+1}(t) \\ \vdots \\ \tilde{v}_n(t) \end{pmatrix} - B \begin{pmatrix} f_1(1,0,\ldots,0,x_{n-i},\tilde{v}_{n-i+1}(t),\ldots,\tilde{v}_n(t)) \\ \vdots \\ f_i(1,0,\ldots,0,x_{n-i},\tilde{v}_{n-i+1}(t),\ldots,\tilde{v}_n(t)) \end{pmatrix} \operatorname{rem} \tilde{q}(t)$
over $\mathbb{K}[[x_{n-i}]]/(x_{n-i}^M)$;

d. compute $\Delta(t) := \lambda_{n-i+1}\hat{v}_{n-i+1}(t) + \cdots + \lambda_n \hat{v}_n(t) - t$;

e. update $\tilde{q}(t) := \tilde{q}(t) - (\tilde{q}'(t)\,\Delta(t) \operatorname{rem} \tilde{q}(t))$ computed over $\mathbb{K}[[x_{n-i}]]/(x_{n-i}^M)$;

f. for j from $n-i+1$ to n,
update $\tilde{v}_j(t) := \hat{v}_j(t) - (\tilde{v}_j'(t)\,\Delta(t) \operatorname{rem} \tilde{q}(t))$ computed over $\mathbb{K}[[x_{n-i}]]/(x_{n-i}^M)$.

4. For j from $n-i+1$ to n, compute $\tilde{w}_j := \tilde{q}'\,\tilde{v}_j \operatorname{rem} \tilde{q}$ over $\mathbb{K}[[x_{n-i}]]/(x_{n-i}^{D_i+1})$.

5. Return $\tilde{q}, \tilde{w}_{n-i+1}, \ldots, \tilde{w}_n$ truncated to order $x_{n-i}^{D_i+1}$ and then regarded in $\mathbb{K}[x_{n-i}][t]$.

PROPOSITION 5.1. *Algorithm 5.1 is correct. Let $\varepsilon > 0$ be a fixed rational value. If $\mathbb{K} = \mathbb{F}_{p^k}$ and if*

$$p^k > \max\left(\bar{d}, (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n\, \bar{d}^{\varepsilon/8}\right)(\bar{d}^n - 1),$$

then it takes time $\tilde{O}(D_i \bar{d}^{(1+3\varepsilon)i} k \log p) + \tilde{O}(\bar{d}^n k \log p)$.

Proof. This algorithm directly comes from [27, section 4]. We do not repeat the correctness proof but focus on the complexity analysis in the Turing model for the dense representation of polynomials.

The partial evaluations of $f_1, \ldots, f_i$ at $x_0 = 1, x_1 = 0, \ldots, x_{n-i-1} = 0$ are achieved in time $\tilde{O}(\bar{d}^n k \log p)$ by Lemma 2.10 and (2.1)—recall that the homogeneous polynomials are actually represented by their specialization at $x_0 = 1$ and their total degree. All the partial derivatives needed for the Jacobian matrix can be obtained in time $\tilde{O}(\bar{d}^n k \log p)$ by Lemma 2.9.

In step 2, we compute A in time

$$\tilde{O}(\bar{d}^{(1+3\varepsilon)i} k \log p),$$

by Theorem 3.3. Then the inversion of A is performed by Berkowitz' algorithm in time $i^{\omega+1}\tilde{O}(D_i k \log p) = \tilde{O}(\bar{d}^i k \log p)$.

In step 3 we regard $f_j(1,0,\ldots,0,x_{n-i},\ldots,x_n)$ and the entries of $J_i(1,0,\ldots,0,x_{n-i},\ldots,x_n)$ as polynomials in $\mathbb{F}_{p^k}[[x_{n-i}]]/(x_{n-i}^{D_i+1})$: their evaluations at $(\tilde{v}_{n-i+1}(t),\ldots,\tilde{v}_n(t))$ take time $\tilde{O}(D_i \bar{d}^{(1+3\varepsilon)i} k \log p)$ by Theorem 3.3. The remaining computations in steps 3 until 5 take time $\tilde{O}(D_i^2 k \log p)$. □

5.2. Intersection step

For the intersection step we follow the method designed in [27, section 6]. We first compute a deformation of the parametrization of $\tilde{I}_i$ by the primitive element

$$u_{\mathfrak{e}} := u + e_{n-i+1}x_{n-i+1} + \cdots + e_n x_n$$

where the e_i are new variables satisfying $e_i e_j = 0$ for all i, j. Let $\mathfrak{e} := (e_{n-i+1}, \ldots, e_n)$ and let $\mathbb{K}_{\mathfrak{e}} := \mathbb{K}[\mathfrak{e}]/(\mathfrak{e}^2)$ be a shorthand for $\mathbb{K}[e_{n-i+1},\ldots,e_n]/(e_{n-i+1},\ldots,e_n)^2$. The extension $\tilde{I}_{\mathfrak{e},i}$ of $\tilde{I}_i$ to $\mathbb{K}_{\mathfrak{e}}(x_{n-i})[x_{n-i+1},\ldots,x_n]$ is parametrized by $u_{\mathfrak{e}}$ as follows

$$\tilde{I}_{\mathfrak{e},i} = (\tilde{q}_{\mathfrak{e}}(u_{\mathfrak{e}}), \tilde{q}_{\mathfrak{e}}'(u_{\mathfrak{e}})\,x_{n-i+1} - \tilde{w}_{\mathfrak{e},n-i+1}(u_{\mathfrak{e}}), \ldots, \tilde{q}_{\mathfrak{e}}'(u_{\mathfrak{e}})\,x_n - \tilde{w}_{\mathfrak{e},n}(u_{\mathfrak{e}})),$$

where $q_{\mathfrak{e}} \in \mathbb{K}_{\mathfrak{e}}[x_{n-i}][t]$ is monic in t of degree D_i, and the $w_{\mathfrak{e},j} \in \mathbb{K}_{\mathfrak{e}}[x_{n-i}][t]$ have degree $< D_i$ in t, for $j = n-i+1, \ldots, n$. Of course, $\tilde{q}_{\mathfrak{e}}, \tilde{w}_{\mathfrak{e},n-i+1}, \ldots, \tilde{w}_{\mathfrak{e},n}$ respectively coincide with $\tilde{q}$, $\tilde{w}_{n-i+1}, \ldots, \tilde{w}_n$ modulo $\mathfrak{e}$. In addition we know that the total degrees of $\tilde{q}_{\mathfrak{e}}, \tilde{w}_{\mathfrak{e},n-i+1}, \ldots, \tilde{w}_{\mathfrak{e},n}$ in x_{n-i} and t is $\leqslant D_i$; see [18, section 3].

Algorithm 5.2

Input. The Kronecker parametrization $u, \tilde{q}, \tilde{w}_{n-i+1}, \ldots, \tilde{w}_n$ of $\tilde{I}_i$ and $b \in \mathbb{K}$.

Output. The Kronecker parametrization $u_{\mathfrak{e}}, \tilde{q}_{\mathfrak{e}}, \tilde{w}_{\mathfrak{e},n-i+1}, \ldots, \tilde{w}_{\mathfrak{e},n}$ of $\tilde{I}_{\mathfrak{e},i}$.

Assumptions. $f_1, \ldots, f_n$ satisfy R_1, R_2, R_3, V_1, V_2.

1. Compute $\tilde{q}_{\mathfrak{e}}(t) := \tilde{q}(t) - \sum_{j=n-i+1}^{n} e_j \tilde{w}_j(t)$.
2. Compute $\Delta(t) := \left(\sum_{j=n-i+1}^{n} e_j \tilde{w}_j(t)\right) / \tilde{q}'(t) \bmod \tilde{q}(t) \bmod (x_{n-i}-b)^{D_i+1}$.
3. For j from $n-i+1$ to n do the following:
   a. compute $\tilde{v}_j(t) := \tilde{w}_j(t) / \tilde{q}'(t) \bmod \tilde{q}(t) \bmod (x_{n-i}-b)^{D_i+1}$;
   b. compute $\tilde{v}_{\mathfrak{e},j}(t) := \tilde{v}_j(t) + (\tilde{v}_j'(t)\,\Delta(t) \bmod \tilde{q}(t) \bmod (x_{n-i}-b)^{D_i+1})$;
   c. compute $\tilde{w}_{\mathfrak{e},j}(t) := \tilde{q}_{\mathfrak{e}}'(t)\,\tilde{v}_{\mathfrak{e},j}(t) \bmod \tilde{q}_{\mathfrak{e}}(t) \bmod (x_{n-i}-b)^{D_i+1}$.
4. Return $u_{\mathfrak{e}}, \tilde{q}_{\mathfrak{e}}, \tilde{w}_{\mathfrak{e},n-i+1}, \ldots, \tilde{w}_{\mathfrak{e},n}$ truncated to $(x_{n-i}-b)^{D_i+1}$ and regarded in $\mathbb{K}_{\mathfrak{e}}[x_{n-i}][t]$.

PROPOSITION 5.2. *Except for $<2D_i^2$ values of b in $\mathbb{K}$, Algorithm 5.2 is correct. If $\mathbb{K} = \mathbb{F}_{p^k}$, then it takes time $\tilde{O}(D_i^2 k \log p)$.*

Proof. This algorithm corresponds to [27, Algorithm 8]. Its correctness is explained in [27, section 6.4]. We adapt the complexity analysis to our context. In steps 2 and 3.a the inverse of $\tilde{q}'$ modulo $\tilde{q}$ is only required modulo $\mathfrak{e}$. If we were performing this modular inversion over $\mathbb{F}_{p^k}(x_{n-i})$ then we would have to handle rational functions of degrees growing with D_i^2. In order to avoid this coefficient swell we use truncated power series instead of rational functions. In fact, we compute the inverse of $\tilde{q}'$ modulo $\tilde{q}$ for x_{n-i} specialized to b, then we use classical Hensel lifting to recover $\tilde{q}'^{-1} \bmod \tilde{q} \bmod (x_{n-i}-b)^{D_i+1}$. This method works fine whenever the resultant of $\tilde{q}'$ and $\tilde{q}$ does not vanish at b. This resultant has degree $<2D_i^2$ in x_{n-i}.

Algorithm 5.2 needs to shift x_{n-i} by $-b$ in the input and by b in the output, which can be done in time $\tilde{O}(D_i^2 k \log p)$ using a divide and conquer algorithm; see for instance [7, Lemma 7]. Then $\tilde{q}'^{-1} \bmod \tilde{q} \bmod (x_{n-i}-b)^{D_i+1}$ is obtained in time $\tilde{O}(D_i^2 k \log p)$. The rest of the algorithm performs polynomial products of cost $\tilde{O}(D_i^2 k \log p)$. □

If $P(t)$ is a polynomial in $\mathbb{K}_{\mathfrak{e}}[x_{n-i}][t]$ of total degree $\leqslant D_i$ in x_{n-i} and t, and assuming $\lambda_{n-i} \neq 0$, then we write

$$P(t)|_{x_{n-i} \mapsto (y-t)/(\lambda_{n-i}+e_{n-i})}$$

for the polynomial in $\mathbb{K}_{\mathfrak{e}}[y][t]$ obtained after replacing x_{n-i} by $(y-t)/(\lambda_{n-i}+e_{n-i})$. This substitution can be performed efficiently as follows:

1. Compute $\bar{P}(t) := P(t)|_{x_{n-i} \mapsto y/(\lambda_{n-i}+e_{n-i})}$ with a softly linear number of operations in $\mathbb{K}$;
2. Compute $\bar{P}(t)|_{y \mapsto y-t}$. This substitution is applied to each homogeneous component of $\bar{P}$ seen in $\mathbb{K}_{\mathfrak{e}}[y,t]$. In fact if $\bar{P}_i$ is the homogeneous component of degree i then we are led to compute $\bar{P}_i(y-t,t)$, which basically reduces to computing $\bar{P}_i(y-1,1)$, and that corresponds to a univariate polynomial shift. Such a shift is known to take softly linear time, as mentioned above.

If $\mathbb{K} = \mathbb{F}_{p^k}$ then this substitution in P takes time $\tilde{O}(D_i^2 k \log p)$. We are now ready to complete the intersection step.

Algorithm 5.3

Input. The Kronecker parametrization $u_{\mathfrak{e}}, \tilde{q}_{\mathfrak{e}}, \tilde{w}_{\mathfrak{e},n-i+1}, \ldots, \tilde{w}_{\mathfrak{e},n}$ of $\tilde{I}_{\mathfrak{e},i}$ and $b \in \mathbb{K}$.

Output. The Kronecker parametrization $U, Q, W_{n-i}, \ldots, W_n$ of $\dot{I}_{i+1}$.

Assumptions. $f_1, \ldots, f_n$ satisfy R_1, R_2, R_3, V_1, V_2.

1. Compute $\hat{q}_{\mathfrak{e}} := \tilde{q}_{\mathfrak{e}}(t)|_{x_{n-i} \mapsto (y-t)/(\lambda_{n-i}+e_{n-i})}$.
2. Compute $\hat{o}_{\mathfrak{e}} := \tilde{q}'_{\mathfrak{e}}(t)|_{x_{n-i} \mapsto (y-t)/(\lambda_{n-i}+e_{n-i})}$ and $\hat{p}_{\mathfrak{e}} := \hat{o}_{\mathfrak{e}}^{-1} \bmod \hat{q}_{\mathfrak{e}} \bmod (y-b)^{D_{i+1}+1}$.
3. For j from $n-i+1$ to n compute:
   a. $\hat{w}_{\mathfrak{e},j} := \tilde{w}_{\mathfrak{e},j}|_{x_{n-i} \mapsto (y-t)/(\lambda_{n-i}+e_{n-i})}$;
   b. $\hat{v}_{\mathfrak{e},j} := \hat{p}_{\mathfrak{e}} \hat{w}_{\mathfrak{e},j} \bmod \hat{q}_{\mathfrak{e}} \bmod (y-b)^{D_{i+1}+1}$.
4. Let $\hat{h}_{\mathfrak{e}}(t) := \hat{o}_{\mathfrak{e}}^{d_{i+1}} f_{i+1}\Big(1, 0, \ldots, 0, \frac{y-t}{\lambda_{n-i}+e_{n-i}}, \hat{v}_{\mathfrak{e},n-i+1}(t), \ldots, \hat{v}_{\mathfrak{e},n}(t)\Big) \bmod \hat{q}_{\mathfrak{e}} \bmod (y-b)^{D_{i+1}+1}$.
5. Compute $Q_{\mathfrak{e}} := \mathrm{Res}_t(\hat{h}_{\mathfrak{e}}(t), \hat{q}_{\mathfrak{e}}(t)) / \mathrm{Res}_t(\hat{o}_{\mathfrak{e}}(t), \hat{q}_{\mathfrak{e}}(t))^{d_{i+1}} = Q - e_{n-i} W_{n-i} - \cdots - e_n W_n$ with $Q, W_{n-i}, \ldots, W_n$ in $\mathbb{K}[[y-b]] / ((y-b)^{D_{i+1}+1})$.
6. Return $U := \lambda_{n-i} x_{n-i} + \cdots + \lambda_n x_n, Q, W_{n-i}, \ldots, W_n$ truncated modulo $(y-b)^{D_{i+1}+1}$ and seen as polynomials in $\mathbb{K}[y]$.

PROPOSITION 5.3. *Let S be a finite subset of $\mathbb{K}$. If we take $(\lambda_{n-i}, \ldots, \lambda_n)$ in S^{i+1} and then b in S at random, then Algorithm 5.3 works correctly as specified with probability $> 1 - 6 D_{i+1}^3 / |S|$. Let $\varepsilon > 0$ be a fixed rational value, and assume that $\bar{d} \geqslant 2$. If $\mathbb{K} = \mathbb{F}_{p^k}$, and if*

$$p^k > \max\left(\bar{d}, (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n \bar{d}^{\varepsilon/8}\right)(\bar{d}^n - 1),$$

then Algorithm 5.3 takes time $\tilde{O}(D_{i+1} \bar{d}^{(1+3\varepsilon)i} k \log p) + \tilde{O}(\bar{d}^n k \log p)$.

Proof. The algorithm is borrowed from [27, Algorithm 7]. Again we only focus on the complexity and probability analyses in our context.

First we need to compute $\mathrm{Res}_t(\hat{o}_{\mathfrak{e}}(t), \hat{q}_{\mathfrak{e}}(t))$ along with the corresponding Bézout relation that yields $\hat{p}_{\mathfrak{e}} := \hat{o}_{\mathfrak{e}}^{-1} \bmod \hat{q}_{\mathfrak{e}}$. We wish to apply a fast subresultant algorithm, namely either the one of [21, chapter 11] or the one of [51]. For this purpose, we need to ensure the following properties:

- the leading coefficient (in degree D_i) of $\hat{q}_{\mathfrak{e}}$ is non-zero modulo $\mathfrak{e}$,
- the leading coefficient, in degree $D'_i \leqslant D_i - 1$, of $\hat{o}_{\mathfrak{e}}$ is non-zero modulo $\mathfrak{e}$,
- the non-zero subresultant coefficients from degree 0 to $D_i - 2$ are non-zero modulo $\mathfrak{e}$.

In fact, these coefficients are the only quantities that need to be inverted during the execution of the aforementioned fast subresultant algorithms.

We claim that these properties are met for sufficiently generic values of the λ_i. In order to prove and quantify this claim, we introduce

$$u_\Lambda := \Lambda_{n-i+1} x_{n-i+1} + \cdots + \Lambda_n x_n$$

where the Λ_i are new independent variables, and set $\mathbb{K}_\Lambda := \mathbb{K}(\Lambda_{n-i+1}, \ldots, \Lambda_n)$. The extension $\tilde{I}_{\Lambda,i}$ of $\tilde{I}_i$ to $\mathbb{K}_\Lambda(x_{n-i})[x_{n-i+1}, \ldots, x_n]$ is parametrized by u_Λ as follows

$$\tilde{I}_{\Lambda,i} = (\tilde{q}_\Lambda(u_\Lambda), \tilde{q}'_\Lambda(u_\Lambda)\, x_{n-i+1} - \tilde{w}_{\Lambda,n-i+1}(u_\Lambda), \ldots, \tilde{q}'_\Lambda(u_\Lambda)\, x_n - \tilde{w}_{\Lambda,n}(u_\Lambda)),$$

where

- $\tilde{q}_\Lambda, \tilde{w}_{\Lambda,n-i+1}, \ldots, \tilde{w}_{\Lambda,n}$ actually belong to $\mathbb{K}[\Lambda_{n-i+1}, \ldots, \Lambda_n][x_{n-i}, t]$,
- $\tilde{q}_\Lambda$ is monic, separable, and of degree D_i in t,

- $\tilde{w}_{\Lambda,n-i+1},\ldots,\tilde{w}_{\Lambda,n}$ have degree $<D_i$ in t,
- $\tilde{q}_\Lambda,\tilde{w}_{\Lambda,n-i+1},\ldots,\tilde{w}_{\Lambda,n}$ have total degree $\leqslant D_i$ in x_{n-i} and t,
- $\tilde{q}_\Lambda,\tilde{w}_{\Lambda,n-i+1},\ldots,\tilde{w}_{\Lambda,n}$ have total degree $\leqslant D_i$ in $\Lambda_{n-i+1},\ldots,\Lambda_n$.

For the proofs of these facts we refer the reader to [18, section 3]. It follows that $\tilde{q}_\mathfrak{e}$, $\tilde{w}_{\mathfrak{e},n-i+1},\ldots,\tilde{w}_{\mathfrak{e},n}$ are the respective specializations of $\tilde{q}_\Lambda,\tilde{w}_{\Lambda,n-i+1},\ldots,\tilde{w}_{\Lambda,n}$ at $\Lambda_{n-i+1}=\lambda_{n-i+1}+e_{n-i+1},\ldots,\Lambda_n=\lambda_n+e_n$. We further introduce

$$\hat{q}_\Lambda := \tilde{q}_\Lambda(t)|_{x_{n-i}\mapsto(y-t)/\Lambda_{n-i}} \quad\text{and}\quad \hat{o}_\Lambda := \tilde{q}'_\Lambda(t)|_{x_{n-i}\mapsto(y-t)/\Lambda_{n-i}}.$$

By the specialization property, the subresultants of $\hat{o}_\mathfrak{e}(t)$ and $\hat{q}_\mathfrak{e}(t)$, seen as polynomials of respective degrees $\deg\hat{o}_\Lambda\leqslant D_i-1$ and $\deg\hat{q}_\Lambda=D_i$ in t, are the specializations of those of $\hat{o}_\Lambda(t)$ and $\hat{q}_\Lambda(t)$ at $\Lambda_{n-i}=\lambda_{n-i}+e_{n-i},\ldots,\Lambda_n=\lambda_n+e_n$. Such a subresultant coefficient of $\hat{o}_\Lambda(t)$ and $\hat{q}_\Lambda(t)$ is a specific minor of the Sylvester matrix of $\hat{o}_\Lambda(t)$ and $\hat{q}_\Lambda(t)$, of size at most $(2D_i-1)\times(2D_i-1)$. Therefore such a subresultant coefficient is a (non-reduced) rational function with denominator $\Lambda_{n-i}^{D_i(2D_i-1)}$ and numerator in $\mathbb{K}[y][\Lambda_{n-i},\ldots,\Lambda_n]$ of total degree $\leqslant 2D_i(2D_i-1)$ in $\Lambda_{n-i},\ldots,\Lambda_n$. Consequently the product of all these non-zero numerators yields a non-zero polynomial $\mathfrak{L}(\Lambda_{n-i},\ldots,\Lambda_n)$ of total degree

$$\leqslant\ 2D_i(2D_i-1)(D_i-1)\ <\ 4D_i^3,$$

such that $\lambda_{n-i}\,\mathfrak{L}(\lambda_{n-i},\ldots,\lambda_n)\neq 0$ implies that the D_i-1 non-zero subresultant coefficients of $\hat{o}_\mathfrak{e}(t)$ and $\hat{q}_\mathfrak{e}(t)$ are invertible; in particular the resultant is invertible and so are the leading coefficients of $\hat{o}_\mathfrak{e}(t)$ and $\hat{q}_\mathfrak{e}(t)$.

The fast computation of $\mathrm{Res}_t(\hat{h}_\mathfrak{e}(t),\hat{q}_\mathfrak{e}(t))$ raises the same issue, and requires the same kind of probability analysis. We thus introduce

$$\hat{h}_\Lambda := \left((\tilde{q}'_\Lambda(t))^{d_{i+1}}f_{i+1}\left(1,0,\ldots,0,x_{n-i},\frac{\tilde{w}_{\Lambda,n-i+1}(t)}{\tilde{q}'_\Lambda(t)},\ldots,\frac{\tilde{w}_{\Lambda,n}(t)}{\tilde{q}'_\Lambda(t)}\right)\right)\Bigg|_{x_{n-i}\mapsto(y-t)/\Lambda_{n-i}}$$

which belongs to $\mathbb{K}(\Lambda_{n-i})[\Lambda_{n-i+1},\ldots,\Lambda_n][y,t]$, has degree $\leqslant D_{i+1}$ in t, $\leqslant D_{i+1}$ in y, and total degree $\leqslant D_{i+1}$ in $\Lambda_{n-i+1},\ldots,\Lambda_n$. In this way

$$\hat{o}_\mathfrak{e}^{d_{i+1}}f_{i+1}\left(1,0,\ldots,0,\frac{y-t}{\lambda_{n-i}+e_{n-i}},\hat{v}_{\mathfrak{e},n-i+1}(t),\ldots,\hat{v}_{\mathfrak{e},n}(t)\right)$$

is the specialization of $\hat{h}_\Lambda$ at $\Lambda_{n-i}=\lambda_{n-i}+e_{n-i},\ldots,\Lambda_n=\lambda_n+e_n$.

A subresultant coefficient of $\hat{h}_\Lambda(t)$ and $\hat{q}_\Lambda(t)$, seen as polynomials of respective degrees $\deg_t\hat{h}_\Lambda(t)\leqslant D_{i+1}$ and D_i, is a (non-reduced) rational function with denominator $\Lambda_{n-i}^{2D_iD_{i+1}}$ and numerator in $\mathbb{K}[y][\Lambda_{n-i},\ldots,\Lambda_n]$ of total degree $\leqslant 4D_iD_{i+1}$ in $\Lambda_{n-i},\ldots,\Lambda_n$. Consequently there exists a non-zero polynomial $\mathfrak{H}(\Lambda_{n-i},\ldots,\Lambda_n)$ of total degree

$$\begin{aligned}&\leqslant\ 4D_{i+1}D_i^2+\deg_\Lambda(\Lambda_{n-i}^{D_{i+1}}\hat{h}_\Lambda(t))+\deg_\Lambda(\Lambda_{n-i}^{D_i}\hat{q}_\Lambda(t))\\&\leqslant\ 4D_{i+1}D_i^2+2D_{i+1}+2D_i\\&\leqslant\ 6D_{i+1}D_i^2,\end{aligned}$$

such that $\lambda_{n-i}\,\mathfrak{H}(\lambda_{n-i+1},\ldots,\lambda_n)\neq 0$ implies that all the non-zero subresultant coefficients of $\hat{h}_\mathfrak{e}(t)$ and $\hat{q}_\mathfrak{e}(t)$ and the leading coefficients of $\hat{h}_\mathfrak{e}(t)$ and $\hat{q}_\mathfrak{e}(t)$ are invertible.

As in the proof of Lemma 4.2, there exists a non-zero polynomial $\mathfrak{D}(\Lambda_{n-i},\ldots,\Lambda_n)$ of total degree $<2D_{i+1}^2$ such that $\mathfrak{D}(\lambda_{n-i},\ldots,\lambda_n)\neq 0$ implies that U is a primitive element for $\dot{I}_{i+1}$. This ensures the correctness of the output.

Now assume that $\lambda_{n-i}\,\mathfrak{L}(\lambda_{n-i},\ldots,\lambda_n)\,\mathfrak{H}(\lambda_{n-i+1},\ldots,\lambda_n)\,\mathfrak{D}(\lambda_{n-i},\ldots,\lambda_n)\neq 0$. This happens with probability

$$\geqslant 1-\frac{4D_i^3+6D_{i+1}D_i^2+2D_{i+1}^2}{|S|}\geqslant 1-\frac{3D_{i+1}^3}{|S|},$$

whenever $(\lambda_{n-i},\ldots,\lambda_n)$ is taken at random in S^{i+1}.

The subresultant coefficients of $\hat{o}_\Lambda(t)$ and $\hat{q}_\Lambda(t)$ have degree $\leqslant D_i\,(2\,D_i-1)$ in y, so except for $\leqslant D_i^2\,(2\,D_i-1)<D_{i+1}^3/4$ values of b, we can compute $\mathrm{Res}_t(\hat{o}_{\mathfrak{e}}(t),\hat{q}_{\mathfrak{e}}(t))$ modulo $(y-b)^{D_{i+1}+1}$ in time $\tilde{O}(D_{i+1}D_i k\log p)$, by means of the fast subresultant algorithm. A subresultant coefficient of $\hat{h}_\Lambda(t)$ and $\hat{q}_\Lambda(t)$ has degree $\leqslant 2D_{i+1}D_i$ in y, so except for $\leqslant 2D_{i+1}D_i^2+D_i+D_{i+1}<2D_{i+1}^3$ values of b, $\mathrm{Res}_t(\hat{h}_{\mathfrak{e}}(t),\hat{q}_{\mathfrak{e}}(t))$ can be computed by means of the fast subresultant algorithm modulo $(y-b)^{D_{i+1}+1}$ in time $\tilde{O}(D_{i+1}\,D_i\,k\log p)$. Consequently a suitable value for b is found at random in S with probability of success

$$\geqslant 1-\frac{3D_{i+1}^3}{|S|}.$$

All the substitutions $x_{n-i}\mapsto (y-t)/(\lambda_{n-i}+e_{n-i})$ take $\tilde{O}(D_i^2 k\log p)$ time according to the above discussion. All the shifts by $-b$ and b in y totalize time $\tilde{O}((D_{i+1}+D_i^2)\,k\log p)$.

The steps 3.b take time $\tilde{O}(D_i\,D_{i+1}\,k\log p)$ in total. In step 4 we first specialize the variables $x_0,\ldots,x_{n-i-1}$ to $1,0,\ldots,0$ in time $\tilde{O}(\bar{d}^n k\log p)$, after which the rest of the evaluation of

$$f_{i+1}\Big(1,0,\ldots,0,\frac{y-t}{\lambda_{n-i}+e_{n-i}},\hat{v}_{\mathfrak{e},n-i+1}(t),\ldots,\hat{v}_{\mathfrak{e},n}(t)\Big)\bmod \hat{q}_{\mathfrak{e}}\bmod (y-b)^{D_{i+1}+1}$$

is decomposed into $i+1$ evaluations in $\mathbb{E}_j:=\mathbb{F}_{p^k}[e_j,y,t]/(e_j^2,(y-b)^{D_{i+1}},\hat{q}_{\mathfrak{e}})$ for $j=n-i,\ldots,n$. More precisely, for the evaluation in $\mathbb{E}_j$ we ignore the variables e_k for $k\neq j$ and we may thus appeal to Theorem 3.3 in order to achieve time

$$\tilde{O}(D_{i+1}\,\bar{d}^{(1+3\varepsilon)i}k\log p).$$ □

5.3. Total complexity

Putting together the propositions of this and the previous sections, we obtain the following complexity bound.

THEOREM 5.4. *Let $\varepsilon>0$ be a fixed rational value, let $\mathbb{K}=\mathbb{F}_{p^k}$ and let $f_1,\ldots,f_n$ be homogeneous polynomials in $\mathbb{K}[x_0,\ldots,x_n]$ of total degrees $d_1,\ldots,d_n$ such that:*

- R_1, R_2, *and* R_3 *are satisfied;*
- $p^k\geqslant \max\Big(\max\Big(\bar{d},(1+\varepsilon)\Big(\frac{8}{\varepsilon}+1\Big)n\,\bar{d}^{\varepsilon/8}\Big)(\bar{d}^n-1),100\,D_n^3\Big)$.

Then an invertible matrix N, a primitive element u and a Kronecker parametrization of $I_n\circ N+(x_0-1)$ can be computed in time $\tilde{O}(D_n\bar{d}^{(1+3\varepsilon)(n-1)}k\log p)$, with a probability of failure that is bounded by $<1/2$.

Proof. This result is a consequence of the Kronecker solver that was recalled at the beginning of this section. First we pick up an invertible matrix N of size $(n+1)\times(n+1)$ with entries at random in $\mathbb{F}_{p^k}$ as described in Lemma 4.5, in order to ensure that after replacing $f_1,\ldots,f_n$ by $f_1\circ N,\ldots,f_n\circ N$ conditions V_1 and V_2 hold with a probability at least

$$1-5\,D_n^2/p^k\;\geqslant\;1-1/40.$$

Computing $f_1\circ N,\ldots,f_n\circ N$ takes time $\tilde{O}(\bar{d}^n k\log p)=\tilde{O}(D_n\bar{d}^{n-1}k\log p)$ by Proposition 2.15.

On the other hand the probability that $\lambda_1, \ldots, \lambda_n$ are suitable for all the intersection steps is at least

$$1 - \sum_{i=1}^{n-1} \frac{6 D_{i+1}^3}{p^k} \geqslant 1 - 12 D_n^3 / p^k \geqslant 1 - 1/8,$$

by Proposition 5.3. The probability that a random value of b is suitable for all stages of the solver is at least

$$1 - \sum_{i=1}^{n-1} \frac{6 D_{i+1}^3 + 2 D_i^2}{p^k} \geqslant 1 - 14 D_n^3 / p^k \geqslant 1 - 1/7,$$

by Propositions 5.2 and 5.3.

Overall the probability of failure is bounded by $1/2$. Notice that the constant 100 is not intended to be optimal. The total complexity follows by combining Propositions 5.1, 5.2 and 5.3. □

COROLLARY 5.5. *Let $\varepsilon > 0$ be a fixed rational value, let $\mathbb{K} = \mathbb{F}_{p^k}$ and let $f_1, \ldots, f_n$ be homogeneous polynomials in $\mathbb{K}[x_0, \ldots, x_n]$ of total degrees $d_1, \ldots, d_n$ such that* R_1, R_2, *and* R_3 *are satisfied.*

Then the Kronecker algorithm computes an integer $l = O(n \log \bar{d})$, the finite field $\mathbb{F}_{p^{kl}}$, the images of $f_1, \ldots, f_n$ into $\mathbb{F}_{p^{kl}}[x_0, \ldots, x_n]$, an invertible matrix N with entries in $\mathbb{F}_{p^{kl}}$, a primitive element $u \in \mathbb{F}_{p^{kl}}$ and a Kronecker parametrization of $(f_1, \ldots, f_n) \circ N + (x_0 - 1)$ over $\mathbb{F}_{p^{kl}}$ in time $\tilde{O}(D_n \bar{d}^{(1+\varepsilon)(n-1)} k \log p)$ with any arbitrarily small fixed probability of failure, or in expected time $\tilde{O}(D_n \bar{d}^{(1+\varepsilon)(n-1)} k \log p)$ with a Las Vegas algorithm.

Proof. The case

$$p^k \geqslant B := \max\left(\max\left(\bar{d}, (1+\varepsilon)\left(\frac{8}{\varepsilon} + 1\right) n \bar{d}^{\varepsilon/8}\right)(\bar{d}^n - 1), 100 D_n^3\right).$$

is already handled in Theorem 5.4, and corresponds to $l = 1$. So we focus on the opposite case $p^k < B$, for which $k \log p = O(n \log \bar{d})$. We compute the first integer $\underline{l}$ such that

$$p^{k\underline{l}} > B,$$

that is

$$\underline{l} := \left\lceil \frac{\log B}{k \log p} \right\rceil = O\left(\frac{n \log \bar{d}}{k \log p}\right).$$

Then we set

$$l := \min\,(ik + 1 : i \in \mathbb{N},\ ik + 1 \geqslant \underline{l}),$$

so l is coprime to k and we have $lk \log p = O((n \log \bar{d})^2)$. Now we appeal to Lemma 3.4: we construct $\mathbb{F}_{p^{kl}}$ in time $\tilde{O}(l^2 \log^2 p + l^{\omega+2} k \log p)$ with probability of failure $< 1/4$; then the casts between $\mathbb{F}_{p^k}$ and $\mathbb{F}_{p^{kl}}$ can be achieved in time $\tilde{O}(l^2 k \log p)$.

With this construction at hand we proceed as follows:

- We cast $f_1, \ldots, f_n$ into $\mathbb{F}_{p^{kl}}[x_0, \ldots, x_n]$ in time $\tilde{O}(\bar{d}^n l^2 k \log p) = \tilde{O}(\bar{d}^n k \log p)$;
- We call the Kronecker solver over $\mathbb{F}_{p^{kl}}$. In view of Theorem 5.4, the resolution over $\mathbb{F}_{p^{kl}}$ takes time $\tilde{O}(D_n \bar{d}^{(1+3\varepsilon)(n-1)} k \log p)$ with a probability of failure $< 1/2$.

Using Proposition 4.6, the correctness of the result can be verified in time $\tilde{O}(\bar{d}^{(1+3\varepsilon)n} k \log p)$ as well. We are thus able to reach any arbitrarily small fixed probability of failure by repeating the resolution sufficiently many times. Altogether, this leads to a Las Vegas algorithm that satisfies the claimed expected complexity bound—of course, ε needs to be replaced by $\varepsilon/3$ from the outset to conclude the proof. □

Remark 5.6. If some of the d_i are 1, then it is far more efficient to compute:

- a permutation τ of $\{1,\ldots,n\}$,
- $t \in \mathbb{N}$, linear forms $l_0,\ldots,l_n \in \mathbb{K}\,y_0 + \cdots + \mathbb{K}\,y_{n-t}$,
- homogeneous polynomials $g_1,\ldots,g_{n-t} \in \mathbb{K}[y_0,\ldots,y_{n-t}]$ of total degrees $\geqslant 2$,

such that $f_{\tau(i)}(l_0,\ldots,l_n) = 0$ for $i = 1,\ldots,t$ and $g_i(y_0,\ldots,y_{n-t}) = f_{\tau(t+i)}(l_0,\ldots,l_n)$ for $i = 1,\ldots,n-t$. In this way we are reduced to solve a "smaller" system with all degrees $\geqslant 2$. This simplification in the input system can be achieved in time $\tilde{O}((n^\omega + \bar{d}^n)\,k \log p)$ by means of Proposition 2.15. We leave out the details for the sake of conciseness.

6. SYSTEMS OVER THE RATIONAL NUMBERS

From now we turn to solving polynomial systems $f_1 = \cdots = f_n = 0$ with coefficients in $\mathbb{Z}$, still under assumptions R_1, R_2, and R_3. We still write $I := (f_1,\ldots,f_n)$, and we appeal to the algorithm designed in [27, section 4.6]: we first perform the resolution modulo a suitable prime number p, and then lift the parametrization of the solution set over the p-adic numbers by means of Newton–Hensel lifting. Roughly speaking, once the bit size exceeds twice the maximum bit size of the integers of the Kronecker parametrization over $\mathbb{Q}$, then this parametrization can be reconstructed from its p-adic expansion.

In order to formalize this approach, we first need to estimate the bit size of the Kronecker parametrizations in terms of the input size. Then we design a probabilistic algorithm that generates suitable values for p with high probability.

For a scalar, a vector, a matrix, or a polynomial A over $\mathbb{C}$ we write $\|A\|_\infty$ for the maximal norm of its entries or coefficients. Then the height of A, written $\mathrm{ht}\,A$, is defined as

$$\mathrm{ht}\,A := \max(0, \log \|A\|_\infty).$$

Until the end of the text we set

$$H_n := \sum_{i=1}^{n} \frac{\mathrm{ht}(f_i)}{d_i}.$$

6.1. *p*-adic Hensel lifting

In section 5.1 we have described an algorithm for lifting power series in a Kronecker parametrization. In the following paragraph we adapt the method to lift p-adic integers instead. The algorithm originates from [27, section 4.6]. The Jacobian matrix of $f_1,\ldots,f_n$ in $x_1,\ldots,x_n$ is still written

$$J_n := \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}.$$

Algorithm 6.1

Input. $f_1,\ldots,f_n \in \mathbb{Z}[x_0,\ldots,x_n]$ and the parametrization $q, v_1,\ldots,v_n \in (\mathbb{Z}/p\mathbb{Z})[t]$ of $(f_1,\ldots,f_n) + (x_0 - 1) \bmod p$ by $u = \lambda_1 x_1 + \cdots + \lambda_n x_n$; an integer $\kappa > 0$.

Output. The Kronecker parametrization $\tilde{q}, \tilde{w}_1, \ldots, \tilde{w}_n$ in $(\mathbb{Z}/p^\kappa\mathbb{Z})[t]$ of $(f_1,\ldots,f_n) + (x_0 - 1) \bmod p^\kappa$ by u.

Assumptions. $f_1,\ldots,f_n$ satisfy R_1, R_2, R_3, V_1, V_2.

1. Initialize $\varkappa := 1$, $\tilde{q} := q$, $\tilde{v}_1 := v_1,\ldots,\tilde{v}_n := v_n$.
2. Initialize B with the inverse of $A := J_n(1, v_1(t),\ldots,v_n(t))$ modulo $q(t)$.

3. While $\nu < \kappa$ do the following:
   a. update $A := J_n(1, \tilde{v}_1(t), \ldots, \tilde{v}_n(t)) \bmod \tilde{q}(t)$ and $B := B - B\,(A\,B - \mathrm{Id}_n) \bmod \tilde{q}(t)$, over $\mathbb{Z}/p^{\nu}\mathbb{Z}$;
   b. set $\nu := \min(2\nu, \kappa)$;
   c. compute $\begin{pmatrix} \hat{v}_1(t) \\ \vdots \\ \hat{v}_n(t) \end{pmatrix} := \begin{pmatrix} \tilde{v}_1(t) \\ \vdots \\ \tilde{v}_n(t) \end{pmatrix} - B \begin{pmatrix} f_1(1, \tilde{v}_1(t), \ldots, \tilde{v}_n(t)) \\ \vdots \\ f_n(1, \tilde{v}_1(t), \ldots, \tilde{v}_n(t)) \end{pmatrix} \bmod \tilde{q}(t)$ over $\mathbb{Z}/p^{\kappa}\mathbb{Z}$;
   d. compute $\Delta(t) := \lambda_1 \hat{v}_1(t) + \cdots + \lambda_n \hat{v}_n(t) - t$ over $\mathbb{Z}/p^{\nu}\mathbb{Z}$;
   e. replace $\tilde{q}(t) := \tilde{q}(t) - (\tilde{q}'(t)\,\Delta(t) \bmod \tilde{q}(t))$ computed over $\mathbb{Z}/p^{\nu}\mathbb{Z}$;
   f. for j from 1 to n,
      replace $\tilde{v}_j(t) := \hat{v}_j(t) - (\tilde{v}_j'(t)\,\Delta(t) \bmod \tilde{q}(t))$ computed over $\mathbb{Z}/p^{\nu}\mathbb{Z}$.
4. For j from 1 to n, compute $\tilde{w}_j := \tilde{q}'\,\tilde{v}_j \bmod \tilde{q}$ over $\mathbb{Z}/p^{\kappa}\mathbb{Z}$.
5. Return $\tilde{q}, \tilde{w}_1, \ldots, \tilde{w}_n$.

PROPOSITION 6.1. *Algorithm 6.1 is correct. Let $\varepsilon > 0$ be a fixed rational value. If*

$$p > \max\left(\bar{d}, (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n\, \bar{d}^{\varepsilon/8}\right)(\bar{d}^n - 1),$$

then Algorithm 6.1 takes time $\tilde{O}(\bar{d}^{(1+3\varepsilon)n}\,\kappa \log p + \bar{d}^n\,(\bar{h}+1))$, where $\bar{h} := \max(\mathrm{ht}\, f_1, \ldots, \mathrm{ht}\, f_n)$.

Proof. This algorithm directly comes from [27, section 4.6]. We do not repeat the proof of the correctness but focus directly on its bit complexity for the dense representation of input polynomials.

Before all we may compute the partial derivatives of the f_i in time $\tilde{O}(\bar{d}^n\,(\bar{h} + \kappa \log p))$ by Lemma 2.9 to any precision κ. In step 2, we compute A in time

$$\tilde{O}(\bar{d}^{(1+3\varepsilon)n} \log p),$$

by Theorem 3.3. Then the inversion of A modulo q is performed by Berkowitz' algorithm in time $n^{\omega+1}\tilde{O}(D_n \log p) = \tilde{O}(\bar{d}^n \log p)$.

In step 3 the evaluations of $f_1, \ldots, f_n$ and J_n at $(1, \tilde{v}_1(t), \ldots, \tilde{v}_n(t))$ at a given precision ν take time $\tilde{O}(\bar{d}^{(1+3\varepsilon)n}\,\nu \log p + \bar{d}^n\,\kappa \log p)$, again in view of Theorem 3.3. Since we keep doubling ν until we reach the precision κ, the total cost is bounded by $\tilde{O}(\bar{d}^{(1+3\varepsilon)n}\,\kappa \log p)$. The rest of the computations in steps 3 and 4 take time $\tilde{O}(n^{\omega} D_n\, \kappa \log p) = \tilde{O}(D_n\, \kappa \log p)$. □

6.2. Chow form and height

Let V be a zero dimensional subvariety of $\mathbb{P}^n$. "The" *Chow form* of V is the polynomial $\mathcal{F} \in \mathbb{C}[\Lambda_0, \ldots, \Lambda_n]$ that satisfies

$$\mathcal{F}(\lambda_0, \ldots, \lambda_n) = 0 \quad \Leftrightarrow \quad V \cap \mathcal{U}_{\mathbb{P}^n}(\lambda_0 x_0 + \lambda_1 x_1 + \cdots + \lambda_n x_n) \neq \emptyset,$$

where the λ_i are taken in $\mathbb{C}$. The Chow form is defined up to a scalar factor in $\mathbb{C}$, and is homogeneous of total degree $\deg V$.

From now we assume that V is the variety of $I \circ N := (f_1 \circ N, \ldots, f_n \circ N)$, where N is an invertible $(n+1) \times (n+1)$ integer matrix such that $I \circ N$ is in Noether position. In this setting, V is included in the affine subspace $\mathbb{A}_n$ defined by $x_0 = 1$, and we have $\deg \mathcal{F} = D_n$. We then define the *normalized Chow form* C of V as the $\mathbb{C}$-multiple of $\mathcal{F}$ having the coefficient of $\Lambda_0^{D_n}$ equal to 1. The normalized Chow form belongs to $\mathbb{Q}[\Lambda_0, \ldots, \Lambda_n]$.

In order to bound bit sizes of Kronecker parametrizations of $I \circ N$, we first need to bound the bit size of the coefficients of C. For this purpose we follow a construction due to Philippon [62], and introduce

$$\mathcal{H} := m(C; S_{n+1}) + D_n \sum_{i=1}^{n} \frac{1}{2i},$$

where $m(C; S_{n+1})$ denotes the S_{n+1}-Mahler measure of C defined as

$$m(C; S_{n+1}) := \int_{S_{n+1}} \log |C(\lambda_0, \ldots, \lambda_n)| \, \mu_{n+1}(\lambda_0, \ldots, \lambda_n)$$

where μ_{n+1} is the usual Haar measure of mass 1 over the sphere

$$S_{n+1} := \{(z_0, \ldots, z_n) \in \mathbb{C}^{n+1} : |z_0|^2 + \cdots + |z_n|^2 = 1\}.$$

So $m(C; S_{n+1})$ is to be understood as a real $(2n+1)$-dimensional integral.

On the other hand, the usual Mahler measure of C is defined as an n-dimensional complex contour integral

$$m(C) := \int_0^1 \cdots \int_0^1 \log |C(\mathrm{e}^{2\pi \mathrm{i} t_0}, \ldots, \mathrm{e}^{2\pi \mathrm{i} t_n})| \, \mathrm{d}t_0 \cdots \mathrm{d}t_n.$$

The relationship between these two measures is given in [52, Theorem 4]:

$$0 \leqslant m(C) - m(C; S_{n+1}) \leqslant D_n \sum_{i=1}^{n} \frac{1}{2i},$$

which implies

$$m(C) \leqslant \mathcal{H}. \tag{6.1}$$

Then we appeal to [43, Corollary 2.10, p. 553] which ensures the existence of a rational number c such that $c\,C$ has all its coefficients in $\mathbb{Z}$ and

$$\log |c| + \mathcal{H} \leqslant \left(\sum_{i=1}^{n} \frac{\mathrm{ht}(f_i \circ N)}{d_i} + 2n \log(n+1) \right) D_n. \tag{6.2}$$

To relate the Mahler measure to the actual bit size of $c\,C$ we use [43, Lemma 1.1, p. 529]:

$$|m(C) - \log \|C\|_\infty| \leqslant \log(n+2)\, D_n. \tag{6.3}$$

Combining the three latter inequalities leads to

$$\begin{aligned}
\mathrm{ht}(c\,C) \;=\; \log \|c\,C\|_\infty \;&\leqslant\; \log |c| + \log \|C\|_\infty \\
&\leqslant\; \log |c| + m(C) + \log(n+2)\, D_n && \text{(by (6.3))} \\
&\leqslant\; \log |c| + \mathcal{H} + \log(n+2)\, D_n && \text{(by (6.1))} \\
&\leqslant\; \left(\sum_{i=1}^{n} \frac{\mathrm{ht}(f_i \circ N)}{d_i} + 2n \log(n+1) \right) D_n + \log(n+2)\, D_n && \text{(by (6.2))} \\
&\leqslant\; \left(\sum_{i=1}^{n} \frac{\mathrm{ht}(f_i \circ N)}{d_i} + (2n+1) \log(n+2) \right) D_n. && (6.4)
\end{aligned}$$

6.3. Bit size of Kronecker parametrizations

Let $q, w_1, \ldots, w_n$ be a Kronecker parametrization of $I \circ N + (x_0 - 1)$ by the primitive element $u = \lambda_1 x_1 + \cdots + \lambda_n x_n$ taken in $\mathbb{Z}[x_1, \ldots, x_n]$. It relates to the Chow form C of $I \circ N$ as follows:

$$\begin{aligned}
q(t) \;&=\; C(t, -\lambda_1, \ldots, -\lambda_n), \\
w_i(t) \;&=\; \frac{\partial C}{\partial \Lambda_i}(t, -\lambda_1, \ldots, -\lambda_n) \text{ for } i = 1, \ldots, n.
\end{aligned}$$

Since $c\,C$ belongs to $\mathbb{Z}[\Lambda_0,\ldots,\Lambda_n]$, all the coefficients of $c\,q$ and of $c\,w_i$ for $i=1,\ldots,n$ also belong to $\mathbb{Z}$, and their bit size can be bounded as follows. By $D_n \geqslant 2$ and inequality (2.1), the Chow form C admits at most $\binom{D_n+n}{n} \leqslant \frac{3}{2}D_n^n$ terms, whence

$$\|c\,q\|_\infty \;\leqslant\; \frac{3}{2}D_n^n\,\|c\,C\|_\infty\,\|\lambda\|_\infty^{D_n}, \tag{6.5}$$

$$\|c\,q'\|_\infty \;\leqslant\; \frac{3}{2}D_n^{n+1}\,\|c\,C\|_\infty\,\|\lambda\|_\infty^{D_n}, \tag{6.6}$$

$$\|c\,w_i\|_\infty \;\leqslant\; \frac{3}{2}D_n^{n+1}\,\|c\,C\|_\infty\,\|\lambda\|_\infty^{D_n} \quad \text{for } i=1,\ldots,n, \tag{6.7}$$

where $\|\lambda\|_\infty \geqslant 1$ is a shorthand for $\|(\lambda_1,\ldots,\lambda_n)\|_\infty$. Now we estimate how the coefficients of f increase because of the change of variables N.

LEMMA 6.2. *Let $f \in \mathbb{Z}[x_0,\ldots,x_n]$ be homogeneous of total degree $d \geqslant 2$, and let N be a $(n+1)\times(n+1)$ matrix over $\mathbb{Z}$. Then we have*

$$\|f\circ N\|_\infty \leqslant \frac{3}{2}d!\,d^n\,\|f\|_\infty\,\|N\|_\infty^d.$$

Proof. The proof is immediate by using $\|(x_0+\cdots+x_n)^d\|_\infty \leqslant d!$ and inequality (2.1). □

LEMMA 6.3. *For all $d \geqslant 1$ we have $\log(d!) \leqslant 2\,d\log d$.*

Proof. This is a well known unconditional variant of Stirling's formula:

$$\log(d!) \;\leqslant\; \int_2^{d+1}\log t\,\mathrm{d}t \;=\; (d+1)\log(d+1)-(d+1)-2\log 2+2 \;\leqslant\; 2\,d\log d. \qquad \square$$

LEMMA 6.4. *Assume that* R_1, R_2, *and* R_3 *hold, and let N be an $(n+1)\times(n+1)$ matrix over $\mathbb{Z}$ such that $f_1\circ N,\ldots,f_n\circ N$ is in Noether position. Then we have*

$$\mathrm{ht}(f_i\circ N) \leqslant \mathrm{ht}(f_i) + d_i\log\|N\|_\infty + 4\,n\,d_i\log d_i, \quad \textit{for } i=1,\ldots,n.$$

In addition, for a Kronecker parametrization $u,q,w_1,\ldots,w_n$ of $(f_1\circ N,\ldots,f_n\circ N)+(x_0-1)$, and with c as above, then we have

$$\begin{aligned} K \;&:=\; \max\left(\mathrm{ht}(c\,q),\mathrm{ht}(c\,q'),\mathrm{ht}(c\,w_1),\ldots,\mathrm{ht}(c\,w_n)\right)\\ &\leqslant\; (H_n + n\log\|N\|_\infty + \log\|\lambda\|_\infty + 16\,n\log D_n)\,D_n. \end{aligned}$$

Proof. From Lemmas 6.2 and 6.3 we obtain

$$\begin{aligned} \mathrm{ht}(f_i\circ N) \;&\leqslant\; \mathrm{ht}(f_i) + d_i\log\|N\|_\infty + \log\left(\frac{3}{2}d_i!\,d_i^n\right)\\ &\leqslant\; \mathrm{ht}(f_i) + d_i\log\|N\|_\infty + 4\,n\,d_i\log d_i. \end{aligned}$$

Consequently,

$$\begin{aligned} \sum_{i=1}^n \frac{\mathrm{ht}(f_i\circ N)}{d_i} \;&\leqslant\; \sum_{i=1}^n \frac{\mathrm{ht}(f_i) + d_i\log\|N\|_\infty + 4\,n\,d_i\log d_i}{d_i}\\ &\leqslant\; H_n + n\log\|N\|_\infty + 4\,n\log D_n. \end{aligned}$$

Combining this bound with (6.4)–(6.7) and using $\log D_n \leqslant D_n$, we deduce

$$\begin{aligned} K \;&\leqslant\; \log\frac{3}{2} + (n+1)\log D_n + \log\|c\,C\|_\infty + D_n\log\|\lambda\|_\infty\\ &\leqslant\; \log\frac{3}{2} + (n+1)\log D_n + \left(\sum_{i=1}^n \frac{\mathrm{ht}(f_i\circ N)}{d_i} + (2\,n+1)\log(n+2)\right)D_n + D_n\log\|\lambda\|_\infty\\ &\leqslant\; (H_n + n\log\|N\|_\infty + \log\|\lambda\|_\infty + 4\,n\log D_n + (3\,n+3)\log(n+2))\,D_n. \end{aligned}$$

Finally we use $n+2 \leqslant 2^{n+1} \leqslant D_n^2$ to bound $(3n+3)\log(n+2)$ by $12\,n \log D_n$, which concludes the proof. □

6.4. Lucky primes

Given an input polynomial system $f_1 = \cdots = f_n = 0$ satisfying R_1, R_2, and R_3, and having all its coefficients in $\mathbb{Z}$, we are led to characterize prime numbers p for which the system admits D_n isolated simple roots when considered modulo p. We will rely on the *a posteriori* verification criterion from Proposition 4.6, in order to ensure that a resolution modulo p can be lifted over the rational numbers.

LEMMA 6.5. *For all $i \geqslant 1$ we have* $\mathrm{ht}((1+z+\cdots+z^{k-1})^i) \leqslant (i-1)\log k$.

Proof. The inequality is an equality for $i=1$. For $i>1$ we have

$$\|(1+z+\cdots+z^{k-1})^i\|_\infty \leqslant k\,\|(1+z+\cdots+z^{k-1})^{i-1}\|_\infty,$$

whence $\|(1+z+\cdots+z^{k-1})^i\|_\infty \leqslant k^{i-1}$ by induction on i. □

LEMMA 6.6. (HADAMARD'S INEQUALITY) *Let M be a $n \times n$ matrix over the real numbers, then we have $|\det M| \leqslant \|M_1\|_2 \cdots \|M_n\|_2 \leqslant n^{n/2}\|M\|_\infty^n$, where M_i represents the i-th column vector of M.*

LEMMA 6.7. *Let $f_1, \ldots, f_n$ be homogeneous polynomials in $\mathbb{Z}[x_0, \ldots, x_n]$ which satisfy R_1, R_2, and R_3. There exists a positive integer $\mathfrak{A}$ such that*

$$\mathrm{ht}\,\mathfrak{A} \leqslant n\bar{d}\,(5H_n + 77\,n\log D_n)\,D_n^2,$$

and for any p that does not divide $\mathfrak{A}$ the sequence $f_1, \ldots, f_n$ satisfies R_1, R_2 and R_3 over $\mathbb{Z}/p\,\mathbb{Z}$.

Proof. By Lemma 4.5 there exists an invertible $(n+1)\times(n+1)$ matrix N with integer entries such that $\|N\|_\infty \leqslant 5D_n^2$ and $f_1 \circ N, \ldots, f_n \circ N$ is in Noether position. On the other hand by Lemma 4.2 there exists a primitive element $u = \lambda_1 x_1 + \cdots + \lambda_n x_n$ for $(f_1 \circ N, \ldots, f_n \circ N)$ with integer coefficients and such that $\|\lambda\|_\infty \leqslant 2D_n^2$. Let $q, w_1, \ldots, w_n$ represent the corresponding Kronecker parametrization, and let c and C be as above. Let φ_i be a non-zero coefficient in f_i of a term of degree d_i for $i = 1, \ldots, n$, let $J \in (\mathbb{Z}[x_0, \ldots, x_n])^{n\times n}$ be the Jacobian matrix of $f_1 \circ N, \ldots, f_n \circ N$ with respect to $x_1, \ldots, x_n$, and

$$\begin{aligned}
\mathfrak{A}_q &:= |\mathrm{Disc}(cq)| = |\mathrm{Res}(cq, cq')|,\\
\mathfrak{A}_J &:= |\mathrm{Res}(\det J(cq', cw_1, \ldots, cw_n), cq)|,\\
\mathfrak{A} &:= \varphi_1 \cdots \varphi_n \mathfrak{A}_q \mathfrak{A}_J.
\end{aligned}$$

By construction, $\mathfrak{A}_q$, $\mathfrak{A}_J$ and thus $\mathfrak{A}$ are positive integers. Now if p does not divide $\mathfrak{A}$, then $\deg(f_i \bmod p) = d_i$ for $i = 1, \ldots, n$, and the system $f_1 = \cdots = f_n = 0$ admits D_n isolated regular roots modulo p. Consequently $f_1, \ldots, f_n$ satisfy R_1, R_2, and R_3 modulo p by Proposition 4.6.

To conclude the proof it remains to bound $\mathfrak{A}$. First, we may further rewrite a bound for the quantity K introduced in Lemma 6.4, by making use of $D_i \geqslant 2^i$:

$$\begin{aligned}
K &\leqslant (H_n + n\log\|N\|_\infty + \log\|\lambda\|_\infty + 16\,n\log D_n)\,D_n\\
&\leqslant (H_n + n\log(5D_n^2) + \log(2D_n^2) + 16\,n\log D_n)\,D_n\\
&\leqslant (H_n + n\log(D_n^5) + \log(D_n^3) + 16\,n\log D_n)\,D_n\\
&\leqslant (H_n + 24\,n\log D_n)\,D_n.
\end{aligned}$$

By applying Lemma 6.6 on the Sylvester matrix of cq and cq', we obtain

$$\mathfrak{A}_q \leqslant \|cq\|_2^{D_n-1}\|cq'\|_2^{D_n} \leqslant (D_n+1)^{D_n}\|cq\|_\infty^{D_n-1}\|cq'\|_\infty^{D_n},$$

whence

$$\begin{aligned}\operatorname{ht}\mathfrak{A}_q &\leqslant D_n\log(D_n+1)+2KD_n\\ &\leqslant 2D_n\log D_n+2\,(H_n+24n\log D_n)\,D_n^2\\ &\leqslant 2\,(H_n+25n\log D_n)\,D_n^2.\end{aligned}\tag{6.8}$$

As to $\mathfrak{A}_J$, we deduce from Lemma 6.4 that

$$\begin{aligned}\operatorname{ht}(f_i\circ N) &\leqslant \operatorname{ht}(f_i)+d_i\log\|N\|_\infty+4nd_i\log d_i\\ &\leqslant \operatorname{ht}(f_i)+d_i\log(5D_n^2)+4nd_i\log d_i\\ &\leqslant \operatorname{ht}(f_i)+2d_i\log D_n+7nd_i\log d_i\\ &\leqslant \operatorname{ht}(f_i)+9nd_i\log D_n.\end{aligned}$$

On the one hand, we have

$$\deg\left(\frac{\partial(f_i\circ N)}{\partial x_j}(cq',cw_1,\ldots,cw_n)\right)\leqslant(d_i-1)\,(D_n-1),$$

hence

$$\deg(\det J(cq',cw_1,\ldots,cw_n))\leqslant n\,(\bar d-1)\,(D_n-1).$$

On the other hand, Lemma 6.5 gives us

$$\operatorname{ht}((cq')^{e_0}(cw_1)^{e_1}\cdots(cw_n)^{e_n}) \leqslant (e_0+\cdots+e_n)\,(K+\log D_n).$$

From Lemma 6.4 and $d_i\binom{d_i-1+n}{n}=d_i\frac{d_i}{d_i+n}\binom{d_i+n}{n}\leqslant\frac{3}{2}d_i^{n+1}$, it follows that

$$\begin{aligned}&\operatorname{ht}\left(\frac{\partial(f_i\circ N)}{\partial x_j}(cq',cw_1,\ldots,cw_n)\right)\\ &\leqslant \operatorname{ht}\left(d_i\binom{d_i-1+n}{n}\right)+\operatorname{ht}(f_i\circ N)+(d_i-1)\,(K+\log D_n)\\ &\leqslant \operatorname{ht}\left(\frac{3}{2}d_i^{n+1}\right)+\operatorname{ht}(f_i)+9nd_i\log D_n+(d_i-1)\,((H_n+24n\log D_n)\,D_n+\log D_n)\\ &\leqslant 12nd_i\log D_n+d_i\,(2H_n+25n\log D_n)\,D_n\\ &\leqslant d_i\,(2H_n+37n\log D_n)\,D_n.\end{aligned}$$

Let $\mathfrak{S}_n$ denote the group of permutations of $\{1,\ldots,n\}$. Then we have

$$\begin{aligned}\operatorname{ht}(\det J(cq',cw_1,\ldots,cw_n)) &\leqslant \operatorname{ht}\left(\sum_{\sigma\in\mathfrak{S}_n}\operatorname{sgn}(\sigma)\prod_{i=1}^n\frac{\partial(f_i\circ N)}{\partial x_{\sigma(i)}}(cq',cw_1,\ldots,cw_n)\right)\\ &\leqslant \operatorname{ht}(n!)+\sum_{i=1}^n d_i\,(2H_n+37n\log D_n)\,D_n\\ &\leqslant n\bar d\,(2H_n+38n\log D_n)\,D_n.\end{aligned}$$

Again by Hadamard bound we deduce that

$$\begin{aligned}\mathfrak{A}_J &\leqslant \|cq\|_2^{n(\bar d-1)(D_n-1)}\|\det J(cq',cw_1,\ldots,cw_n)\|_2^{D_n}\\ &\leqslant (D_n+1)^{n(\bar d-1)(D_n-1)/2}\|cq\|_\infty^{n(\bar d-1)(D_n-1)}\\ &\quad\times(n\,(\bar d-1)\,(D_n-1)+1)^{D_n/2}\|\det J(cq',cw_1,\ldots,cw_n)\|_\infty^{D_n},\end{aligned}$$

whence

$$\begin{aligned}\operatorname{ht}\mathfrak{A}_J &\leqslant n\,(\bar{d}-1)\,(D_n-1)\left(\frac{1}{2}\log(D_n+1)+(H_n+24\,n\log D_n)\,D_n\right)\\ &\quad+\ D_n\left(\frac{1}{2}\log(n\,(\bar{d}-1)\,(D_n-1)+1)+n\,\bar{d}\,(2\,H_n+38\,n\log D_n)\,D_n\right)\\ &\leqslant n\,\bar{d}\,(H_n+25\,n\log D_n)\,D_n^2+n\,\bar{d}\,(2\,H_n+39\,n\log D_n)\,D_n^2\\ &= n\,\bar{d}\,(3\,H_n+64\,n\log D_n)\,D_n^2. \qquad (6.9)\end{aligned}$$

The conclusion follows from adding right-hand sides of (6.8), (6.9), and $\operatorname{ht}(f_1)+\cdots+\operatorname{ht}(f_n)\leqslant\bar{d}\,H_n$. □

Remark 6.8. Theorem 2.1 of [17] provides a more general statement than Lemma 6.7 for the affine case. Extensions to the quasi-projective case can be found in [22].

LEMMA 6.9. *Assume that $d_i\geqslant 2$ for $i=1,\ldots,n$. Let N be an invertible $(n+1)\times(n+1)$ matrix over $\mathbb{Z}$ and let u be a linear form in $\mathbb{Z}[x_1,\ldots,x_n]$. Let p be a prime number, and let $q,w_1,\ldots,w_n$ be polynomials in $\mathbb{Z}[t]$ such that:*

- *q is monic of degree D_n and $\deg w_i<D_n$ for $i=1,\ldots,n$;*
- *q is separable modulo p;*
- *$u(w_1(t),\ldots,w_n(t))=t\,q'(t) \bmod (p,q(t))$;*
- *$(q'(t):w_1(t):\ldots:w_n(t))$ is a root of $f_1\circ N=\cdots=f_n\circ N=0$ modulo $(p,q(t))$;*
- *The value of the Jacobian matrix of $f_1\circ N,\ldots,f_n\circ N$ in $x_1,\ldots,x_n$ at $(q'(t):w_1(t):\ldots:w_n(t))$ is invertible modulo $(p,q(t))$.*

Then $f_1\circ N,\ldots,f_n\circ N$ satisfy R_1, R_2, *and are in Noether position (over $\mathbb{Q}$), u is a primitive element for $(f_1\circ N,\ldots,f_n\circ N)+(x_0-1)$, and $q,w_1,\ldots,w_n$ coincide modulo p with the Kronecker parametrization of $(f_1\circ N,\ldots,f_n\circ N)+(x_0-1)$ over $\mathbb{Q}$ by u.*

Proof. By Proposition 4.6, the residues modulo p of $u,q,w_1,\ldots,w_n$ form a Kronecker parametrization of $(f_1\circ N,\ldots,f_n\circ N)+(x_0-1)$. We may use Algorithm 6.1 from a theoretical point of view in order to prove the existence of univariate polynomials $Q,W_1,\ldots,W_n$ over the p-adic numbers $\mathbb{Q}_p$, satisfying the following properties:

- Q is monic of degree D_n and $\deg W_i<D_n$ for $i=1,\ldots,n$;
- $Q,W_1,\ldots,W_n$ have nonnegative valuations in p and coincide with $q,w_1,\ldots,w_n$ modulo p;
- $u(W_1(t),\ldots,W_n(t))=t\,Q'(t) \bmod Q(t)$;
- $(Q'(t):W_1(t):\ldots:W_n(t))$ is a root of $f_1\circ N=\cdots=f_n\circ N=0$ modulo $Q(t)$;
- The value of the Jacobian matrix of $f_1\circ N,\ldots,f_n\circ N$ in $x_1,\ldots,x_n$ at $(Q'(t):W_1(t):\ldots:W_n(t))$ is invertible modulo $Q(t)$.

Consequently, Proposition 4.6 ensures that $Q,W_1,\ldots,W_n$ is a Kronecker parametrization of $(f_1\circ N,\ldots,f_n\circ N)+(x_0-1)$ over $\mathbb{Q}_p$ by the primitive element u. By uniqueness, it coincides with the image over $\mathbb{Q}_p$ of the Kronecker parametrization of $(f_1\circ N,\ldots,f_n\circ N)+(x_0-1)$ over $\mathbb{Q}$ by u. Using again Proposition 4.6 with the parametrization over the rationals concludes the proof. □

Now we address the question of obtaining a suitable prime number.

LEMMA 6.10. *There exists a probabilistic algorithm which, given positive integers $\mathfrak{A}$ and B such that $6\,\mathrm{ht}\,\mathfrak{A} \leqslant B$, computes a prime p in the range $\{B+1,\ldots,2B\}$, which does not divide $\mathfrak{A}$, with probability of success $\geqslant 1/2$, and in time $\log^{O(1)} B$.*

Proof. This is a consequence of [21, Theorem 18.10, part (i)]: we just appeal to the deterministic primality test designed in [1] in order to ensure that p is always prime in return. This test works in polynomial time. □

6.5. Top level algorithm over the rational numbers

The idea behind the Kronecker solver over $\mathbb{Q}$ is as follows. For a suitable prime p, we use the Kronecker algorithm to solve the input system over $\mathbb{Z}/p\mathbb{Z}$. If the algorithm finishes with success, then we appeal to the *a posteriori* verification criterion of Lemma 6.9. If the criterion passes, then we may lift the Kronecker parametrization over the p-adic integers to arbitrary precision κ by using Algorithm 6.1, and we know that the p-adic parametrization is the image of the rational one. Consequently, as soon as κ is sufficiently large, we can reconstruct the parametrization over $\mathbb{Q}$.

Let us recall that a rational number a/b, with $\gcd(a,b)=1$ and $b>0$, is uniquely determined by its p-adic expansion to precision κ whenever $|a| < \sqrt{p^\kappa/2}$ and $|b| < \sqrt{p^\kappa/2}$; see for instance [68, section 4]. Using the fast Euclidean algorithm from [21, Theorem 5.26] this reconstruction takes $O(\mathsf{I}(\kappa\log p)\log(\kappa\log p)) = \tilde{O}(\kappa\log p)$. The choices of p and the precision κ needed by the solver are made precise in the following algorithm. We recall that ε is a fixed positive rational number, thought to be small.

Algorithm 6.2

Input. $f_1,\ldots,f_n$ homogeneous in $\mathbb{Z}[x_0,\ldots,x_n]$.

Output. An invertible $(n+1)\times(n+1)$ matrix N over $\mathbb{Z}$ such that $(f_1\circ N,\ldots,f_n\circ N)$ is in Noether position, a linear form $u\in\mathbb{Z}[x_1,\ldots,x_n]$, and a Kronecker parametrization $q,w_1,\ldots,w_n$ by u of $(f_1\circ N,\ldots,f_n\circ N)+(x_0-1)$.

Assumption. $f_1,\ldots,f_n$ satisfy R_1, R_2, and R_3.

1. For each i from 1 to n, compute the maximum bit size h_i of the coefficients of f_i. Compute D_n and a positive integer $\tilde{H}_n$ such that $\tilde{H}_n - 2 \leqslant \left(\sum_{i=1}^n \frac{h_i}{\bar{d}_i}\right)/\log 2 \leqslant \tilde{H}_n$. Compute $B\in\mathbb{Z}$ such that
$$B-2 \leqslant \max\left\{\begin{array}{l}\max\left(\bar{d}, (1+\varepsilon)\left(\frac{8}{\varepsilon}+1\right) n\,\bar{d}^{\varepsilon/8}\right)(\bar{d}^n-1),\\ 100\,D_n^3,\\ 6\,n\,\bar{d}\,(5\tilde{H}_n + 77\,n\log D_n)\,D_n^2\end{array}\right\} \leqslant B.$$
2. Compute a prime number p at random in $\{B,\ldots,2B\}$ *via* Lemma 6.10.
3. Call the Kronecker algorithm underlying Theorem 5.4 with the images $\bar{f}_1,\ldots,\bar{f}_n$ of $f_1,\ldots,f_n$ in $(\mathbb{Z}/p\mathbb{Z})[x_0,\ldots,x_n]$. If the resolution fails then go to step 2.
4. Let $\bar{N},\bar{u},\bar{q},\bar{w}_1,\ldots,\bar{w}_n$ be the data returned by the Kronecker algorithm in step 3. If $\deg\bar{q} < D_n$, or if $\bar{q}$ is not separable, or if $\bar{u}(\bar{w}_1(t),\ldots,\bar{w}_n(t)) \neq t\,\bar{q}'(t) \bmod \bar{q}(t)$, or if one of the $\bar{f}_i\circ\bar{N}$ does not vanish at $(\bar{q}'(t):\bar{w}_1(t):\ldots:\bar{w}_n(t))$ modulo $\bar{q}(t)$, or if the value of the Jacobian matrix of $\bar{f}_1\circ\bar{N},\ldots,\bar{f}_n\circ\bar{N}$ in $x_1,\ldots,x_n$ at $(\bar{q}'(t):\bar{w}_1(t):\ldots:\bar{w}_n(t))$ is not invertible modulo $\bar{q}(t)$, then go to step 2.

5. Let N (resp. u) be the canonical preimage of $\bar{N}$ (resp. of $\bar{u}$) with entries in $\{0,...,p-1\}$. Let κ be the first integer such that

$$\log \sqrt{2p^\kappa} \geqslant (\tilde{H}_n + (n+1)\log p + 16\, n \log D_n)\, D_n.$$

   Compute $f_1 \circ N,..., f_n \circ N$ over $\mathbb{Z}/p^\kappa\mathbb{Z}$, and use Algorithm 6.1 to obtain the Kronecker parametrization $\tilde{q}, \tilde{w}_1, \ldots, \tilde{w}_n$ by u of $(f_1 \circ N, \ldots, f_n \circ N) + (x_0 - 1)$ over $\mathbb{Z}/p^\kappa\mathbb{Z}$.
6. By rational reconstruction, compute the unique polynomials $q, w_1,..., w_n$ in $\mathbb{Q}[t]$ that coincide with $\tilde{q}, \tilde{w}_1, \ldots, \tilde{w}_n$ modulo p^κ.

Notice that H_n is 0 whenever all the f_i have their coefficients in $\{-1, 0, 1\}$. Therefore in the following complexity estimates we shall write $H_n + 1$ in order to make complexity bounds valid in this particular case.

THEOREM 6.11. *Algorithm 6.2 is correct, as a Las Vegas algorithm, and takes expected time*

$$\tilde{O}(D_n \bar{d}^{(1+\varepsilon)n} (H_n + 1)).$$

Proof. By Lemma 6.10 the prime p determined in step 2 does not divide the integer $\mathfrak{A}$ defined in Lemma 6.7 with probability $\geqslant 1/2$. In other words, the sequence $\bar{f}_1, \ldots, \bar{f}_n$ satisfy properties R_1, R_2, and R_3 with probability $\geqslant 1/2$. Thanks to $p \geqslant B$, Theorem 5.4 ensures that the Kronecker solver modulo p succeeds with probability $\geqslant 1/2$. It follows that the expected number of times that the algorithm repeats steps 2 to 4 is bounded.

Once the algorithm passes step 4, Lemma 6.9 ensures that the parametrization modulo p may be lifted to any arbitrary precision, and that the parametrization over $\mathbb{Q}$ may be recovered whenever the precision is sufficiently large. According to the preceding discussion on rational reconstruction, we need that

$$\log \sqrt{2p^\kappa} \geqslant K,$$

with K defined in Lemma 6.4; it satisfies

$$K \leqslant (H_n + n \log \|N\|_\infty + \log \|\lambda\|_\infty + 16\, n \log D_n)\, D_n,$$

with $\|N\|_\infty$ and $\|\lambda\|_\infty$ being $< p \leqslant 2B$. This concludes the proof of correctness.

The computation of h_i needs time $\tilde{O}(d_i^n h_i) = \tilde{O}(D_n \bar{d}^n (H_n + 1))$ by Lemmas 2.6 and 2.8. Then D_n, $\tilde{H}_n$ and B can be computed in time $\tilde{O}(\log B)$, which is negligible since

$$\log B = O(n \log \bar{d} + \log(H_n + 1)) = O(n \log D_n + \log(H_n + 1)).$$

The cost of step 2 is $\log^{O(1)} B$ by Lemma 6.10, which is again negligible.

By Theorem 5.4 step 3 takes time

$$\tilde{O}(D_n \bar{d}^{(1+3\varepsilon)(n-1)} \log p) = \tilde{O}(D_n \bar{d}^{(1+3\varepsilon)(n-1)} \log(H_n + 1))$$

since $\log p = O(\log B)$. Proposition 4.6 then provides us with the cost

$$\tilde{O}(\bar{d}^{(1+3\varepsilon)n} \log p) = \tilde{O}(D_n \bar{d}^{(1+3\varepsilon)n} \log(H_n + 1))$$

for step 4. Computing $f_1 \circ N, \ldots, f_n \circ N$ over $\mathbb{Z}/p^\kappa\mathbb{Z}$ in step 5 takes $\tilde{O}(\bar{d}^n \kappa \log p)$ by Proposition 2.15. The cost of the p-adic Hensel lifting is given in Proposition 6.1:

$$\tilde{O}(\bar{d}^{(1+3\varepsilon)n} \kappa \log p) + \tilde{O}(\bar{d}^n \bar{h}) = \tilde{O}(\bar{d}^{(1+3\varepsilon)n} \kappa \log p) + \tilde{O}(D_n \bar{d}^n (H_n + 1)),$$

where $\bar{h} = \max(h_1, \ldots, h_n) = O(\bar{d}(H_n + 1))$. We verify that

$$\kappa \log p = O(\log B + (H_n + (n+1)\log p + 16\, n \log D_n) D_n) = \tilde{O}(D_n (H_n + 1)).$$

Step 6 requires additional time $D_n \tilde{O}(\kappa \log p) = \tilde{O}(D_n^2 (H_n + 1))$. □

The version of the Kronecker solver summarized in Algorithm 6.2 can be optimized by using ideas previously developed for the case of input polynomials given by straight-line programs [27]; see implementation details in the C++ library GEOMSOLVEX of MATHEMAGIX [37].

A first optimization consists in minimizing the height of the parametrization over the rationals. For this purpose it is worth finding a matrix N and a primitive element of small height. This search may be achieved efficiently modulo p, and we refer the reader to [50] for the description of the underlying algorithms.

A second optimization concerns the early termination of the lifting stage over the p-adic integers. In fact one may try to perform rational reconstruction before the bound on the precision is actually reached. If all the reconstructions succeed, then it suffices to verify that the Kronecker parametrization actually describes the solution set. A probabilistic method to do this consists in evaluating the equations at the parametrization modulo an independent random prime number p' with a bit size similar to the one of p. This verification may also be done deterministically over the rationals, but at a higher cost.

For an input system in n variables of degrees $d_i = 2$, the quantity $\log_2(100 D_n^3)$ is of the order $6.6 + 3n$. Therefore with $n = 15$, Algorithm 6.2 involves primes p of bit size close to 52 bits. In fact, the bounds used in the design of Algorithm 6.2 have been simplified and overestimated for the sake of the presentation. Sharper bounds should be considered in practice, in order to ensure that p fits into 32 bits for small and moderate systems, and 64 bits for larger ones.

Let us finally mention that the present bound for the bit size of the Kronecker parametrization over the rationals turns out to be essentially sharp for generic systems, according to [54, Theorem 3.1]. Consequently the complexity reached within Theorem 6.11 is nearly linear in the "expected" size of the output, from the asymptotic point of view. This fact is illustrated by the following example.

Example 6.12. Consider the system, adapted from [12],

$$\begin{cases} f_1 &= x_n^d - x_1^d \\ f_2 &= x_1 x_n^{d-1} - x_2^d \\ &\vdots \\ f_{n-1} &= x_{n-2} x_n^{d-1} - x_{n-1}^d \\ f_n &= x_{n-1} x_n^{d-1} - a\, x_0^d, \end{cases}$$

where $d \geqslant 2$ and $a \in \mathbb{N}_{>0}$. Let ζ_n be a root of

$$x_n^{d^n} - a^{d^{n-1}} = 0,$$

and set

$$\zeta_{n-1} := \frac{a}{\zeta_n^{d-1}}, \quad \zeta_{n-2} := \frac{\zeta_{n-1}^d}{\zeta_n^{d-1}}, \quad \ldots, \quad \zeta_1 := \frac{\zeta_2^d}{\zeta_n^{d-1}},$$

so that $(1 : \zeta_1 : \ldots : \zeta_n)$ is clearly a zero of $f_2 = \cdots = f_n = 0$. For $i = n-1, n-2, \ldots, 1$, we observe that

$$\frac{\zeta_i}{\zeta_n} = \frac{a^{d^{n-i-1}}}{\zeta_n^{d^{n-i}}}.$$

It follows that $(1 : \zeta_1 : \ldots : \zeta_n)$ is also a zero of f_1. The value of the Jacobian matrix of $f_1, \ldots, f_n$ in $x_1, \ldots, x_n$ at $(1 : \zeta_1 : \ldots : \zeta_n)$ is

$$A := \begin{pmatrix} -d\,\zeta_1^{d-1} & & & d\,\zeta_n^{d-1} \\ \zeta_n^{d-1} & -d\,\zeta_2^{d-1} & & (d-1)\,\zeta_1\,\zeta_n^{d-2} \\ & \ddots & \ddots & \vdots \\ & & \zeta_n^{d-1} & -d\,\zeta_{n-1}^{d-1} \quad (d-1)\,\zeta_{n-2}\,\zeta_n^{d-2} \\ & & & \zeta_n^{d-1} \quad (d-1)\,\zeta_{n-1}\,\zeta_n^{d-2} \end{pmatrix}.$$

By expanding $\det A$ with respect to the last column, we obtain

$$\begin{aligned}
\det A &= (-1)^{n-1} d\,\zeta_n^{n(d-1)} + \sum_{i=1}^{n-1} (-1)^{n-1-i} (d-1)\,\zeta_i\,\zeta_n^{d-2} (-d\,\zeta_1^{d-1}) \cdots (-d\,\zeta_i^{d-1})\,(\zeta_n^{d-1})^{n-1-i} \\
&= (-1)^{n-1} d\,\zeta_n^{n(d-1)} + (-1)^{n-1-i} \zeta_n^{n(d-1)} (d-1) \sum_{i=1}^{n-1} d^i \frac{\zeta_i}{\zeta_n} \left(\frac{\zeta_1}{\zeta_n}\right)^{d-1} \cdots \left(\frac{\zeta_i}{\zeta_n}\right)^{d-1} \\
&= (-1)^{n-1} \zeta_n^{n(d-1)} \left(d + (d-1) \sum_{i=1}^{n-1} d^i \frac{a^{d^{n-i-1}}}{\zeta_n^{d^{n-i}}} \left(\frac{a^{d^{n-2}}}{\zeta_n^{d^{n-1}}}\right)^{d-1} \cdots \left(\frac{a^{d^{n-i-1}}}{\zeta_n^{d^{n-i}}}\right)^{d-1} \right) \\
&= (-1)^{n-1} \zeta_n^{n(d-1)} \left(d + (d-1) \sum_{i=1}^{n-1} d^i \frac{a^{d^{n-i-1}}}{\zeta_n^{d^{n-i}}} \left(\frac{a^{d^{n-1}-d^{n-i-1}}}{\zeta_n^{d^n-d^{n-i}}}\right) \right) \\
&= (-1)^{n-1} \zeta_n^{n(d-1)} \left(d + (d-1) \sum_{i=1}^{n-1} d^i \frac{a^{d^{n-1}}}{\zeta_n^{d^n}} \right) \\
&= (-1)^{n-1} \zeta_n^{n(d-1)} \left(d + (d-1) \sum_{i=1}^{n-1} d^i \right) \\
&= (-1)^{n-1} \zeta_n^{n(d-1)} d^n.
\end{aligned}$$

Consequently $\det A$ is invertible. From

$$\frac{\zeta_i}{\zeta_n} = \frac{a^{d^{n-i-1}}}{\zeta_n^{d^{n-i}}} = \frac{\zeta_n^{d^n}}{a^{d^{n-1}}} \frac{a^{d^{n-i-1}}}{\zeta_n^{d^{n-i}}} = \frac{\zeta_n^{d^n-d^{n-i}}}{a^{d^{n-1}-d^{n-i-1}}},$$

Proposition 4.6 thus ensures that $(f_1, \ldots, f_n)$ satisfies R_1 and R_2, that it is in Noether position, and that it is parametrized by the primitive element $u = x_n$, as follows:

$$(f_1, \ldots, f_n) + (x_0 - 1) = \left(x_n^{d^n} - a^{d^{n-1}}, x_1 - \frac{x_n^{d^n-d^{n-1}+1}}{a^{d^{n-1}-d^{n-1-1}}}, \ldots, x_{n-1} - \frac{x_n^{d^n-d+1}}{a^{d^{n-1}-1}} \right).$$

7. CONCLUSION

The Kronecker solver was already known to be competitive both in theory and practice for polynomial systems represented by straight-line programs; see for instance timings in [2, 27]. Under suitable genericity assumptions and for suitable ground fields, we have shown in this paper that this solver also leads to improved probabilistic asymptotic complexity bounds for the dense representation of input polynomials.

Our main results concern finite fields and rational numbers, but our methods can in principle be extended to any ground field over which fast multivariate multi-modular composition is available. Unfortunately, it is still a major open problem to design an algorithm for modular composition over arbitrary ground fields.

Another major problem concerns the practical efficiency of algorithms for modular composition. This holds in particular for the recent fast algorithms for finite fields designed by Kedlaya and Umans [42], and revisited in [36]; we refer the reader to the concluding section of [36] for a quantitative discussion. Consequently we do not expect our Theorem 3.3 to be of practical use, and we cannot claim our new main complexity bounds for polynomial system solving to be relevant in practice.

Nevertheless, it is important to point out that Theorem 3.3 uses modular composition as a blackbox. Whenever faster algorithms for this task *do* become available, they can directly be used in our algorithms. As one possible research direction, it is worthwhile to investigate the replacement of generic modular compositions by specific ones, such as those studied in [33, 34] and which feature more promising performances.

The Kronecker solver admits different extensions for when genericity assumptions R_1 and R_2 are no longer satisfied: see for instance [39, 50], and more references in [18]. In this case, the system might have an infinite number of solutions, and one may express them in terms of a union of equidimensional or irreducible algebraic varieties. We expect that these extensions could be revisited for the dense representation of input polynomials, and plan to investigate the details in future work.

Yet another important question for systems over the rational numbers concerns the complexity of finding numerical approximations of the roots. In generic cases we have proved in Theorem 6.11 how to compute Kronecker representations of solution sets in quasi-optimal time. This implies that numerical roots may also be computed in quasi-linear time but whenever the requested precision is "sufficiently large", by means of fast univariate polynomial solvers. It would be interesting to quantify the latter "sufficiently large", and to compare to the state of the art of numerical polynomial solvers.

APPENDIX A. LINEAR CHANGES OF VARIABLES

This appendix is devoted to subjecting a multivariate polynomial f to a linear change of variables. More precisely, given $f \in \mathbb{A}[x_1,\ldots,x_n]$ and an $n \times n$ matrix $N = (N_{i,j})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant n}$ over a commutative ring $\mathbb{A}$, then we wish to compute

$$(f \circ N)(x_1,\ldots,x_n) := f(N_{1,1}x_1 + \cdots + N_{1,n}x_n, \ldots, N_{n,1}x_1 + \cdots + N_{n,n}x_n).$$

The fast algorithms that we propose below do not seem to be available in the literature. They are suitable for any coefficient ring with sufficiently many elements, and they are also well suited for homogeneous polynomials.

A.1. Algebraic complexity model

In this subsection we focus on the algebraic model (computation trees for instance), we let $\mathbb{A}$ be an effective commutative ring, and M is a cost function such that two polynomials in $\mathbb{A}[x]_{<\ell}$ may be multiplied with cost $\mathsf{M}(\ell)$. The evaluation of a multivariate polynomial at points in a block of points S^n, where S is a finite subset of $\mathbb{A}$, is usually achieved by the successive use of fast univariate evaluations, as recalled in the following lemma.

LEMMA A.1. *Let $\ell \geqslant 1$, let $f \in \mathbb{A}[x_1, \ldots, x_n]$ be of partial degree $<\ell$ in x_i for $i = 1, \ldots, n$, and let S be a subset of $\mathbb{A}$ of cardinality ℓ. Then, all the values of f at S^n can be computed with $O(n\,\ell^{n-1}\mathsf{M}(\ell)\log\ell)$ arithmetic operations in $\mathbb{A}$.*

Proof. We interpret $f \in \mathbb{A}[x_1, \ldots, x_n]$ as a univariate polynomial in x_n,

$$f(x_1, \ldots, x_n) = f_0(x_1, \ldots, x_{n-1}) + \cdots + f_{\ell-1}(x_1, \ldots, x_{n-1})\, x_n^{\ell-1}.$$

We evaluate $f_0, \ldots, f_{\ell-1}$ at S^{n-1} recursively. Then, for each $(\alpha_1, \ldots, \alpha_{n-1}) \in S^{n-1}$, we evaluate $f(\alpha_1, \ldots, \alpha_{n-1}, x_n)$ at all the points of S, with a total cost $O(\ell^{n-1}\mathsf{M}(\ell)\log\ell)$. Denoting by $\mathsf{T}(n, \ell)$ the cost of the algorithm in terms of operations in $\mathbb{A}$, we thus obtain

$$\mathsf{T}(n, \ell) \;=\; \ell\,\mathsf{T}(n-1, \ell) + O(\ell^{n-1}\mathsf{M}(\ell)\log\ell).$$

By induction over n, it follows that

$$\mathsf{T}(n, \ell) \;=\; O(n\,\ell^{n-1}\mathsf{M}(\ell)\log\ell),$$

which implies the claimed bound. □

The next lemma, also well known, concerns the corresponding interpolation problem.

LEMMA A.2. *Let $\ell \geqslant 1$, let $\alpha_1, \ldots, \alpha_\ell$ be pairwise distinct points in $\mathbb{A}$ such that $\alpha_i - \alpha_j$ is invertible whenever $i \neq j$, let $\beta_{i_1,\ldots,i_n}$ be a family of values in $\mathbb{A}$ for $(i_1, \ldots, i_n)$ running over $\{1, \ldots, \ell\}^n$. The unique polynomial $f \in \mathbb{A}[x_1, \ldots, x_n]$ of partial degrees $<\ell$ and such that $f(\alpha_{i_1}, \ldots, \alpha_{i_n}) = \beta_{i_1,\ldots,i_n}$ for all $(i_1, \ldots, i_n) \in \{1, \ldots, \ell\}^n$ can be computed with $O(n\,\ell^{n-1}\mathsf{M}(\ell)\log\ell)$ arithmetic operations in $\mathbb{A}$, including inversions.*

Proof. Again we interpret $f \in \mathbb{A}[x_1, \ldots, x_n]$ as a univariate polynomial in x_n,

$$f(x_1, \ldots, x_n) = f_0(x_1, \ldots, x_{n-1}) + \cdots + f_{\ell-1}(x_1, \ldots, x_{n-1})\, x_n^{\ell-1}. \tag{A.1}$$

For all $(i_1, \ldots, i_{n-1}) \in \{0, \ldots, \ell-1\}^{n-1}$ we interpolate the values $f_0(\alpha_{i_1}, \ldots, \alpha_{i_{n-1}}), \ldots, f_{\ell-1}(\alpha_{i_1}, \ldots, \alpha_{i_{n-1}})$ with $\ell^{n-1}\,O(\mathsf{M}(\ell)\log\ell)$ operations in $\mathbb{A}$. We then recursively interpolate $f_0, \ldots, f_{\ell-1}$ and form f as in (A.1). The total cost is obtained as in the proof of the previous lemma. □

The aim of the following proposition is the fast evaluation of f at a set of points of the form $N(S^n) + B$, for any matrix N and any vector B.

PROPOSITION A.3. *Let $\ell \geqslant 1$, let $f \in \mathbb{A}[x_1, \ldots, x_n]$ be of partial degree $<\ell$ in x_i for $i = 1, \ldots, n$, let $S = \{\alpha_1, \ldots, \alpha_\ell\}$ be a subset of $\mathbb{A}$ of cardinality ℓ such that $\alpha_i - \alpha_j$ is invertible whenever $i \neq j$, let N be a $n \times n$ matrix over $\mathbb{A}$, and let $B \in \mathbb{A}^n$. Let X be the column vector with entries $x_1, \ldots, x_n$. If an LU-decomposition of N is given, then $f(N(S^n) + B)$ and $f(NX + B)$ can be computed with $O(n\,\ell^{n-1}\mathsf{M}(\ell)\log\ell + n^\omega)$ arithmetic operations in $\mathbb{A}$, including inversions.*

Proof. We write $B =: (\beta_1, \ldots, \beta_n)$. We first assume that $N = (N_{i,j})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant n}$ is upper triangular, and we partition $N(S^n) + B$ into

$$\begin{aligned} N(S^n) + B &= \bigsqcup_{i=1}^{\ell} N(S^{n-1} \times \{\alpha_i\}) + B \\ &= \bigsqcup_{i=1}^{\ell} (\tilde{N}(S^{n-1}) + \tilde{B}_i) \times \{N_{n,n}\alpha_i + \beta_n\}, \end{aligned}$$

where $\tilde{N} := (N_{i,j})_{1 \leqslant i \leqslant n-1,\, 1 \leqslant j \leqslant n-1}$ and $\tilde{B}_i := \alpha_i \begin{pmatrix} N_{1,n} \\ \vdots \\ N_{n-1,n} \end{pmatrix} + \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_{n-1} \end{pmatrix}$. We compute

$$g_i(x_1, \ldots, x_{n-1}) := f(x_1, \ldots, x_{n-1}, N_{n,n}\alpha_i + \beta_n)$$

for $i = 1, \ldots, \ell$ using $O(\ell^{n-1}\mathsf{M}(\ell)\log \ell)$ operations in $\mathbb{A}$. For $i = 1, \ldots, \ell$, we then evaluate $g_i(x_1, \ldots, x_{n-1})$ at $\tilde{N}(S^{n-1}) + \tilde{B}_i$ by induction. The base case $n = 0$ takes constant time $O(1)$. Consequently, for any n, the total number of operations in $\mathbb{A}$ is $O(n\,\ell^{n-1}\mathsf{M}(\ell)\log \ell)$, by the same argument as in the proof of Lemma A.1. We recover $f(N(x_1, \ldots, x_n) + B)$ with $O(n\,\ell^{n-1}\mathsf{M}(\ell)\log \ell)$ operations in $\mathbb{A}$ by Lemma A.2.

If N is lower triangular then we may revert of the variables in f and the columns of N in order to reduce to the upper triangular case. Alternatively, we may adapt the latter decomposition of the set of points, as follows:

$$\begin{aligned} N(S^n) + B &= \bigsqcup_{i=1}^{\ell} N(\{\alpha_i\} \times S^{n-1}) + B \\ &= \bigsqcup_{i=1}^{\ell} \{N_{1,1}\alpha_i + \beta_1\} \times (\tilde{N}(S^{n-1}) + \tilde{B}_i), \end{aligned}$$

where $\tilde{N} := (N_{i,j})_{2 \leqslant i \leqslant n,\, 2 \leqslant j \leqslant n}$ and $\tilde{B}_i := \alpha_i \begin{pmatrix} N_{2,1} \\ \vdots \\ N_{n,1} \end{pmatrix} + \begin{pmatrix} \beta_2 \\ \vdots \\ \beta_n \end{pmatrix}$. So we compute

$$g_i(x_2, \ldots, x_n) := f(N_{1,1}\alpha_i + \beta_1, x_2, \ldots, x_n)$$

and evaluate $g_i(x_2, \ldots, x_n)$ at $\tilde{N}(S^{n-1}) + \tilde{B}_i$ by induction, for $i = 1, \ldots, \ell$.

Finally if N is general, then it suffices to use the given LU-decomposition, where L is lower triangular with 1 on the diagonal, and U is upper triangular. In fact we have $f(LU(S^n) + B) = (f \circ L)(U(S^n) + L^{-1}B)$, so we compute $f \circ L$ and then $(f \circ L)(U(S^n) + L^{-1}B)$ and $(f \circ L)(UX + L^{-1}B)$. □

In the next lemma the same technique is adapted to homogeneous polynomials.

LEMMA A.4. *Let $f \in \mathbb{A}[x_0, \ldots, x_n]$ be homogeneous of degree $d \geqslant 1$, let N be a $(n+1) \times (n+1)$ matrix over $\mathbb{A}$, and let $S = \{\alpha_0, \ldots, \alpha_d\}$ be a subset of $\mathbb{A}$ of cardinality $d+1$ such that $\alpha_i - \alpha_j$ is invertible whenever $i \neq j$. If an LU-decomposition of N is given, then $f \circ N$ can be computed with $O(n\,(d+1)^{n-1}\mathsf{M}(d)\log d)$ arithmetic operations in $\mathbb{A}$.*

Proof. Assume first that $N = (N_{i,j})_{0 \leqslant i \leqslant n,\, 0 \leqslant j \leqslant n}$ is lower triangular and let $\tilde{N} := (N_{i,j})_{1 \leqslant i \leqslant n,\, 1 \leqslant j \leqslant n}$. We are led to compose $f(N_{0,0}, x_1, \ldots, x_n)$ with

$$\tilde{N} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} N_{1,0} \\ \vdots \\ N_{n,0} \end{pmatrix}$$

by means of Proposition A.3. If N is upper triangular then it suffices to revert the variables $x_0,\ldots,x_n$ in f, and the columns of N, in order to reduce to the lower triangular case. Alternatively, we may set $\tilde{N} := (N_{i,j})_{0\leqslant i\leqslant n-1,\ 0\leqslant j\leqslant n-1}$ and compose $f(x_0,\ldots,x_{n-1},N_{n,n})$ with

$$\tilde{N}\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} N_{0,n} \\ \vdots \\ N_{n-1,n} \end{pmatrix},$$

in order to obtain $(f\circ N)(x_0,\ldots,x_{n-1},1)$. Finally, for any N, it suffices to use the given LU-decomposition. □

PROPOSITION A.5. *Let $f\in\mathbb{A}[x_0,\ldots,x_n]$ be homogeneous of degree $d\geqslant 2$, let N be an $(n+1)\times(n+1)$ matrix over $\mathbb{A}$, and let $S=\{\alpha_0,\ldots,\alpha_d\}$ be a subset of $\mathbb{A}$ of cardinality $d+1$ such that $\alpha_i-\alpha_j$ is invertible whenever $i\neq j$. If an LU-decomposition of N is given, then $f\circ N$ can be computed with $O(n^2d^{n-1}\mathsf{M}(d)\log d)$ arithmetic operations in $\mathbb{A}$.*

Proof. The total number of coefficients in f is $O(d^n)$ by inequality (2.1). We decompose

$$f = x_0g_0 + x_1g_1 + \cdots + x_ng_n, \tag{A.2}$$

where $x_ng_n(x_0,x_1,\ldots,x_n)$ is made of the terms of f which are multiple of x_n, then $x_{n-1}g_{n-1}$ is made of the terms of $f-x_ng_n$ which are multiple of x_{n-1},..., and finally x_0g_0 is made of the terms of $f-(x_1g_1+\cdots+x_ng_n)$ which are multiple of x_0 (that is a $\mathbb{A}$-multiple of a power of x_0). In this way, we are led to compute $g_i\circ N$ for $i=0,\ldots,n$, with g_i of degree $\leqslant d-1$; this requires $O(n^2d^{n-1}\mathsf{M}(d)\log d)$ operations in $\mathbb{A}$, by Lemma A.4. Then $f\circ N$ can be recovered with further $\tilde{O}(n^2d^n)$ operations. □

Remark A.6. If one can use specific sequences of points α_i, for instance in geometric progressions, then multipoint evaluations and interpolations in one variable and in degree d over $\mathbb{A}$ cost $O(\mathsf{M}(d))$ by means of [10], that saves a factor of $\log d$ in the above complexity estimates.

A.2. Coefficients in a Galois ring

For the purpose of the present paper, we need to adapt the results of the previous subsection to the case when $\mathbb{A}$ is the Galois Ring $\mathrm{GR}(p^\kappa,k)$, and in the context of Turing machines. In the next lemmas we use the lexicographic order on $\mathbb{N}^n$, written $<_{\mathrm{lex}}$, defined by

$$\alpha <_{\mathrm{lex}} \beta \;\Leftrightarrow\; (\exists j\in\{1,\ldots,n\},\ \alpha_n=\beta_n\wedge\cdots\wedge\alpha_{j+1}=\beta_{j+1}\wedge\alpha_j<\beta_j).$$

In terms of Turing machines, we need the following variants of Lemmas A.1 and A.2.

LEMMA A.7. *Let $\ell\geqslant 1$, let $f\in\mathrm{GR}(p^\kappa,k)[x_1,\ldots,x_n]$ be of partial degree $<\ell$ in x_i for $i=1,\ldots,n$, and let $\alpha_1,\ldots,\alpha_\ell$ be values in $\mathrm{GR}(p^\kappa,k)$. Then, the values $f(\alpha_{i_1},\ldots,\alpha_{i_n})$ for $(i_1,\ldots,i_n)$ running over $\{1,\ldots,\ell\}^n$ in the lexicographic order $<_{\mathrm{lex}}$ can be computed in time*

$$n\,\ell^n\tilde{O}(\log^2\ell\,\kappa k\log p).$$

Proof. The proof follows the one of Lemma A.1 while taking data reorganizations into account. More precisely, using one $\ell^{n-1}\times\ell$ matrix transposition, we reorganize the values of the f_i after the recursive calls into the sequence of

$$f_0(\alpha_{i_1},\ldots,\alpha_{i_{n-1}}),\ldots,f_{\ell-1}(\alpha_{i_1},\ldots,\alpha_{i_{n-1}})$$

for $(i_1,\ldots,i_{n-1})$ running over $\{1,\ldots,\ell\}^{n-1}$ in the lexicographic order $<_{\text{lex}}$. Then, after the multi-point evaluations of $f(\alpha_{i_1},\ldots,\alpha_{i_{n-1}},x_n)$, we need to transpose the $\ell\times\ell^{n-1}$ array made of the values of f, in order to ensure the lexicographic ordering in the output. The cost of these transpositions is $O(\ell^n\log\ell\,\kappa k\log p)$ by Lemma 2.1, which is negligible. □

LEMMA A.8. *Assume $\ell\geqslant 1$ and $p^k\geqslant\ell$. Let $\alpha_1,\ldots,\alpha_\ell$ be pairwise distinct values in $\mathrm{GR}(p^\kappa,k)$ such that $\alpha_i-\alpha_j$ is invertible modulo p for all $i\neq j$, and let $\beta_{i_1,\ldots,i_n}$ be a family of values in $\mathrm{GR}(p^\kappa,k)$ for $(i_1,\ldots,i_n)$ running over $\{1,\ldots,\ell\}^n$ in the lexicographic order $<_{\text{lex}}$. The unique polynomial $f\in\mathrm{GR}(p^\kappa,k)[x_1,\ldots,x_n]$ of partial degree $<\ell$ in x_i for $i=1,\ldots,n$, and such that $f(\alpha_{i_1},\ldots,\alpha_{i_n})=\beta_{i_1,\ldots,i_n}$ for all $(i_1,\ldots,i_n)$ in $\{1,\ldots,\ell\}^n$, can be computed in time*

$$n\,\ell^n\,\tilde{O}(\log^2\ell\,\kappa k\log p).$$

Proof. The proof follows the one of Lemma A.2, by doing the data reorganizations in the opposite direction from the one in the proof of Lemma A.7. □

From now, for convenience, we discard the case $\ell=1$. In this way, whenever $\ell\geqslant 2$, we may use $n^{O(1)}=\log^{O(1)}(\ell^n)$.

PROPOSITION A.9. *Assume $\ell\geqslant 2$ and $p^k\geqslant\ell$. Let $f\in\mathrm{GR}(p^\kappa,k)[x_1,\ldots,x_n]$ be of partial degree $<\ell$ in x_i for $i=1,\ldots,n$, and let N be a $n\times n$ matrix over $\mathrm{GR}(p^\kappa,k)$. If an LU-decomposition of N is given, then $f\circ N$ can be computed in time $\tilde{O}(\ell^n\,\kappa k\log p)$.*

Proof. We first generate a subset $S:=\{\alpha_1,\ldots,\alpha_\ell\}$ of $\mathrm{GR}(p,k)$ of cardinality ℓ in time $\tilde{O}(\ell k\log p)$; this ensures the invertibility of $\alpha_i-\alpha_j$ for $i\neq j$. The proof then follows the one of Proposition A.3 while taking data reorganizations into account. When N is upper triangular, the computation of $g_1,\ldots,g_\ell$ requires the multi-point evaluation of f regarded in $\mathrm{GR}(p^\kappa,k)[x_1,\ldots,x_{n-1}][x_n]$: we may simply appeal to the fast univariate algorithm because it only involves additions, subtractions and products by elements of $\mathrm{GR}(p^\kappa,k)$ over the ground ring $\mathrm{GR}(p^\kappa,k)[x_1,\ldots,x_{n-1}]$. Consequently $g_1,\ldots,g_\ell$ may be obtained in time $\ell^{n-1}\,\tilde{O}(\ell\,\kappa k\log p)$, by Lemma 2.5. In addition, the $\ell^{n-1}\times\ell$ array of values of the g_i must be transposed at the end, in order to guarantee the lexicographic ordering necessary to interpolate $f\circ N$.

When N is lower triangular, the data reorganization costs essentially the same, except that the computation of $g_1,\ldots,g_\ell$ takes time $\ell^{n-1}\,\tilde{O}(\ell\,\kappa k\log p)$ by Lemmas 2.11 and 2.5. □

Before achieving the proof of Proposition 2.15, we further need the following lemma in order to change the representation of a homogeneous polynomial.

LEMMA A.10. *Let f be a homogeneous polynomial of degree $d\geqslant 2$ in $\mathrm{GR}(p^\kappa,k)[x_0,\ldots,x_n]$, represented as before by $f^\flat(x_1,\ldots,x_n):=f(1,x_1,\ldots,x_n)$ and d, and let $i\in\{0,\ldots,n\}$. Then, for any $\alpha\in\mathrm{GR}(p^\kappa,k)$ we can compute $f^\diamond(x_0,\ldots,x_{i-1},x_{i+1},\ldots,x_{n-1}):=f(x_0,\ldots,x_{i-1},\alpha,x_{i+1},\ldots,x_{n-1})$ in time $\tilde{O}(d^n\,\kappa k\log p)$.*

Proof. For simplicity the proof is done for $i=n$, but it extends in a coefficientwise manner to any i. A sparse representation of f is made of a sequence of pairs of coefficients and vector exponents. More precisely, if $f=\sum_{e\in\mathbb{N}^{n+1}}f_e x_0^{e_0}\cdots x_n^{e_n}$ then a sparse representation of it is the sequence of the pairs (f_e,e), for all the non-zero coefficients f_e. The bit size of a vector exponent is $O(n+\log d)$, and therefore the bit size of a sparse representation of f is $O(d^n\,(n+\log d)\,\kappa k\log p)$ by (2.1).

In order to prove the lemma, we first convert f, given in dense representation, into a sparse representation. When $n=1$ the sparse representation of $f^\flat$ may be obtained in time $O(d \log d\, \kappa k \log p)$. Otherwise $n \geqslant 2$ and we regard $f^\flat$ in $\mathrm{GR}(p^\kappa,k)[x_1,\ldots,x_{n-1}][x_n]$,

$$f^\flat(x_1,\ldots,x_n) = f_0^\flat(x_1,\ldots,x_{n-1}) + \cdots + f_d^\flat(x_1,\ldots,x_{n-1})\, x_n^d,$$

and recursively compute the sparse representation of $f_i^\flat$ for $i=0,\ldots,d$. These representations may naturally be glued together into a sparse representation of $f^\flat$, in time $O(d^n\,(n+\log d)\,\kappa k \log p)$, by adding the exponent of x_n into each exponent vector. A straightforward induction leads to a total time $O(d^n\,(n+\log d)\,\kappa k \log p)$ for the change of representation of $f^\flat$. Then the sparse representation of f may be deduced with additional time $O(d^n\,(n+\log d)\,\kappa k \log p)$ by appending the exponent of x_0 needed for homogenization.

Second, from the latter sparse representation of f we may simply discard the exponents of x_n and multiply the coefficients with the corresponding powers of α, in order to obtain a sparse representation of f° in time $\tilde{O}(d^n \kappa k \log p)$.

Finally it remains to construct the dense representation of f° from its sparse representation. To this aim we sort the sparse representation in increasing lexicographic order on the exponent vectors in time $O(d^n \log(d^n)\,(n+\log d)\,\kappa k \log p)$. We next compute the dense representation by induction over n. Writing

$$f^\circ(x_0,\ldots,x_{n-1}) = f_0^\circ(x_0,\ldots,x_{n-2}) + \cdots + f_{\ell-1}^\circ(x_0,\ldots,x_{n-2})\, x_{n-1}^{\ell-1},$$

the sparse representations of $f_0^\circ,\ldots,f_{\ell-1}^\circ$ are computed by induction, after removal of the powers of x_{n-1}. The induction ends when $n=0$, in which case the conversion to dense representation requires time $O(d \log d\, \kappa k \log p)$. In total, the dense representation of f° can be computed in time $O(d^n \log(d^n)\,(n+\log d)\,\kappa k \log p)$. □

Proof of Proposition 2.15. We follow the proofs of Lemma A.4 and Proposition A.5, still while taking into account the cost of data reorganizations.

In the proof of Lemma A.4, the cost of obtaining $f(N_{0,0},x_1,\ldots,x_n)$ and $f(x_0,\ldots,x_{n-1},N_{n,n})$ is given by Lemma A.10, that is $\tilde{O}(d^n \kappa k \log p)$.

In the proof of Proposition A.5 we first need to compute the decomposition (A.2) of f. The polynomial

$$g_n(x_0,\ldots,x_n) = f_1(x_0,\ldots,x_{n-1}) + f_2(x_0,\ldots,x_{n-1})\, x_n + \cdots + f_d(x_0,\ldots,x_{n-1})\, x_n^{d-1}$$

is represented by

$$g_n^\flat(x_1,\ldots,x_n) := f_1^\flat(x_1,\ldots,x_{n-1}) + f_2^\flat(x_1,\ldots,x_{n-1})\, x_n + \cdots + f_d^\flat(x_1,\ldots,x_{n-1})\, x_n^{d-1}$$

and $d-1$. Consequently $g_n^\flat$ may be easily obtained in time $O(d^n \kappa k \log p)$. Then the rest of the decomposition $g_{n-1}^\flat,\ldots,g_0^\flat$ is obtained from $f_0^\flat(x_1,\ldots,x_{n-1})$, recursively. The total cost for obtaining all the $g_i^\flat$ is therefore bounded by $\tilde{O}(d^n \kappa k \log p)$.

For any $c \in \mathrm{GR}(p^\kappa,k)$, any $i \in \{0,\ldots,n\}$, and any $j \in \{1,\ldots,n\}$, the computations of $c\,(g_i \circ N)(1,x_1,\ldots,x_n)$ and of $c\,x_j\,(g_i \circ N)(1,x_1,\ldots,x_n)$ take time $d^n \tilde{O}(\kappa k \log p)$ since their supports have cardinality $O(d^n)$ by (2.1).

Finally, from

$$f \circ N = \sum_{i=0}^{n} \left(\sum_{j=0}^{n} N_{i,j} x_j \right) (g_i \circ N)$$

we obtain the representation of $f \circ N$ as

$$(f \circ N)(1, x_1, \ldots, x_n) = \sum_{i=0}^{n} \left(N_{i,0} + \sum_{j=1}^{n} N_{i,j} x_j \right) (g_i \circ N)(1, x_1, \ldots, x_n),$$

using additional time $\tilde{O}(d^n \kappa k \log p)$. The cost of the data reorganizations in the proof of Proposition A.5 is negligible. □

BIBLIOGRAPHY

[1] M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. *Ann. Math.*, pages 781–793, 2004.

[2] B. Bank, M. Giusti, J. Heintz, G. Lecerf, G. Matera, and P. Solernó. Degeneracy loci and polynomial equation solving. *Found. Comput. Math.*, 15(1):159–184, 2015.

[3] M. Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie*. PhD thesis, Université Pierre et Marie Curie - Paris VI, 2004. https://tel.archives-ouvertes.fr/tel-00449609.

[4] M. Bardet, J.-C. Faugère, and B. Salvy. On the complexity of the F_5 Gröbner basis algorithm. *J. Symbolic Comput.*, 70:49–70, 2015.

[5] S. J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Inform. Process. Lett.*, 18:147–150, 1984.

[6] J. Berthomieu, J. van der Hoeven, and G. Lecerf. Relaxed algorithms for p-adic numbers. *J. Théor. Nombres Bordeaux*, 23(3), 2011.

[7] J. Berthomieu, G. Lecerf, and G. Quintin. Polynomial root finding over local rings and application to error correcting codes. *Appl. Alg. Eng. Comm. Comp.*, 24(6):413–443, 2013.

[8] A. Bostan, F. Chyzak, M. Giusti, R. Lebreton, G. Lecerf, B. Salvy, and É Schost. *Algorithmes Efficaces en Calcul Formel*. Frédéric Chyzak (self-published), Palaiseau, 2017. Electronic version available from https://hal.archives-ouvertes.fr/AECF.

[9] A. Bostan, Ph. Flajolet, B. Salvy, and É. Schost. Fast computation of special resultants. *J. Symbolic Comput.*, 41(1):1–29, 2006.

[10] A. Bostan and É. Schost. Polynomial evaluation and interpolation on special sets of points. *J. Complexity*, 21(4):420–446, 2005.

[11] R. P. Brent and H. T. Kung. Fast algorithms for manipulating formal power series. *J. ACM*, 25(4):581–595, 1978.

[12] W. D. Brownawell. Bounds for the degrees in the Nullstellensatz. *Annal. of Math.*, 126(3):577–591, 1987.

[13] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic complexity theory*, volume 315 of *Grundlehren der Mathematischen Wissenschaften*. Springer-Verlag, 1997.

[14] J. F. Canny, E. Kaltofen, and L. Yagati. Solving systems of nonlinear polynomial equations faster. In *Proceedings of the ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation*, ISSAC '89, pages 121–128, New York, NY, USA, 1989. ACM.

[15] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Infor.*, 28:693–701, 1991.

[16] J.-M. Couveignes and R. Lercier. Fast construction of irreducible polynomials over finite fields. *Israel J. Math.*, 194(1):77–105, 2013.

[17] C. D'Andrea, A. Ostafe, I. E. Shparlinski, and M. Sombra. Reduction modulo primes of systems of polynomial equations and algebraic dynamical systems. *Trans. Amer. Math. Soc.*, 371(2):1169–1198, 2019.

[18] C. Durvye and G. Lecerf. A concise proof of the Kronecker polynomial system solver from scratch. *Expo. Math.*, 26(2):101–139, 2008.

[19] J.-C. Faugère, P. Gaudry, L. Huot, and G. Renault. Sub-cubic change of ordering for Gröbner basis: A probabilistic approach. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, ISSAC '14, pages 170–177, New York, NY, USA, 2014. ACM.

[20] J.-C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional Gröbner bases by change of ordering. *J. Symbolic Comput.*, 16(4):329–344, 1993.

[21] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, New York, 3rd edition, 2013.

[22] N. Giménez and G. Matera. On the bit complexity of polynomial system solving. *J. Complexity*, 51:20–67, 2019.

[23] M. Giusti. Some effectivity problems in polynomial ideal theory. In J. Fitch, editor, *EUROSAM 84: International Symposium on Symbolic and Algebraic Computation Cambridge, England, July 9–11, 1984*, pages 159–171, Berlin, Heidelberg, 1984. Springer Berlin Heidelberg.

[24] M. Giusti, K. Hägele, J. Heintz, J. L. Montaña, J. E. Morais, and L. M. Pardo. Lower bounds for Diophantine approximations. *J. Pure Appl. Algebra*, 117/118:277–317, 1997.

[25] M. Giusti, J. Heintz, J. E. Morais, J. Morgenstern, and L. M. Pardo. Straight-line programs in geometric elimination theory. *J. Pure Appl. Algebra*, 124(1-3):101–146, 1998.

[26] M. Giusti, J. Heintz, J. E. Morais, and L. M. Pardo. When polynomial equation systems can be "solved" fast? In *Applied algebra, algebraic algorithms and error-correcting codes (Paris, 1995)*, volume 948 of *Lecture Notes in Comput. Sci.*, pages 205–231. Springer-Verlag, 1995.

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

[28] B. Grenet, J. van der Hoeven, and G. Lecerf. Deterministic root finding over finite fields using Graeffe transforms. *Appl. Alg. Eng. Comm. Comp.*, 27(3):237–257, 2016.

[29] D. Harvey and J. van der Hoeven. Faster polynomial multiplication over finite fields using cyclotomic coefficient rings. *J. Complexity*, 54:101404, 2019.

[30] D. Harvey and J. van der Hoeven. Integer multiplication in time $O(n \log n)$. Technical report, HAL, 2019. http://hal.archives-ouvertes.fr/hal-02070778.

[31] D. Harvey and J. van der Hoeven. Polynomial multiplication over finite fields in time $O(n \log n)$. Technical report, HAL, 2019. http://hal.archives-ouvertes.fr/hal-02070816.

[32] J. Heintz. Definability and fast quantifier elimination in algebraically closed fields. *Theor. Comput. Sci.*, 24(3):239–277, 1983.

[33] J. van der Hoeven and G. Lecerf. Modular composition via complex roots. Technical report, CNRS & École polytechnique, 2017. http://hal.archives-ouvertes.fr/hal-01455731.

[34] J. van der Hoeven and G. Lecerf. Modular composition via factorization. *J. Complexity*, 48:36–68, 2018.

[35] J. van der Hoeven and G. Lecerf. Accelerated tower arithmetic. *J. Complexity*, 55:101402, 2019.

[36] J. van der Hoeven and G. Lecerf. Fast multivariate multi-point evaluation revisited. *J. Complexity*, 56:101405, 2020.

[37] J. van der Hoeven, G. Lecerf, B. Mourrain, et al. Mathemagix, From 2002. http://www.mathemagix.org.

[38] Xiaohan Huang and V. Y. Pan. Fast rectangular matrix multiplication and applications. *J. Complexity*, 14(2):257–299, 1998.

[39] G. Jeronimo and J. Sabia. Effective equidimensional decomposition of affine varieties. *J. Pure Appl. Algebra*, 169(2–3):229–248, 2002.

[40] E. Kaltofen and V. Shoup. Fast polynomial factorization over high algebraic extensions of finite fields. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*, ISSAC '97, pages 184–188, New York, NY, USA, 1997. ACM.

[41] K. S. Kedlaya and C. Umans. Fast modular composition in any characteristic. In *FOCS'08: IEEE Conference on Foundations of Computer Science*, pages 146–155, Washington, DC, USA, 2008. IEEE Computer Society.

[42] K. S. Kedlaya and C. Umans. Fast polynomial factorization and modular composition. *SIAM J. Comput.*, 40(6):1767–1802, 2011.

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

[44] L. Kronecker. Grundzüge einer arithmetischen Theorie der algebraischen Grössen. *J.reine angew. Math.*, 92:1–122, 1882.

[45] Y. N. Lakshman. On the complexity of computing a Gröbner basis for the radical of a zero dimensional ideal. In *Proceedings of the Twenty-second Annual ACM Symposium on Theory of Computing*, STOC '90, pages 555–563, New York, NY, USA, 1990. ACM.

[46] Y. N. Lakshman. A single exponential bound on the complexity of computing Gröbner bases of zero dimensional ideals. In T. Mora and C. Traverso, editors, *Effective Methods in Algebraic Geometry*, pages 227–234, Boston, MA, 1991. Birkhäuser Boston.

[47] Y. N. Lakshman and D. Lazard. On the complexity of zero-dimensional algebraic systems. In T. Mora and C. Traverso, editors, *Effective Methods in Algebraic Geometry*, pages 217–225, Boston, MA, 1991. Birkhäuser Boston.

[48] D. Lazard. Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In J. A. Hulzen, editor, *Computer Algebra: EUROCAL'83, European Computer Algebra Conference London, England, March 28–30, 1983 Proceedings*, pages 146–156. Springer Berlin Heidelberg, 1983.

[49] F. Le Gall. Powers of tensors and fast matrix multiplication. In K. Nabeshima, editor, *ISSAC'14: International Symposium on Symbolic and Algebraic Computation*, pages 296–303, New York, NY, USA, 2014. ACM.

[50] G. Lecerf. Computing the equidimensional decomposition of an algebraic closed set by means of lifting fibers. *J. Complexity*, 19(4):564–596, 2003.

[51] G. Lecerf. On the complexity of the Lickteig–Roy subresultant algorithm. *J. Symbolic Comput.*, 92:243–268, 2019.

[52] P. Lelong. Mesure de Mahler et calcul de constantes universelles pour les polynomes de N variables. *Math. Ann.*, 299(1):673–695, 1994.

[53] H. Matsumura. *Commutative ring theory*, volume 8 of *Cambridge Studies in Advanced Mathematics*. Cambridge university press, 1989.

[54] D. McKinnon. An arithmetic analogue of Bezout's theorem. *Compos. Math.*, 126(2):147–155, 2001.

[55] J. M. McNamee and V. Y. Pan. *Numerical Methods for Roots of Polynomials, Part II*, volume 16 of *Studies in Computational Mathematics*. Elsevier, 2013.

[56] B. Mourrain, V. Y. Pan, and O. Ruatta. Accelerated solution of multivariate polynomial systems of equations. *SIAM J. Comput.*, 32(2):435–454, 2003.

[57] B. Mourrain and Ph. Trébuchet. Solving projective complete intersection faster. In *Proceedings of the 2000 International Symposium on Symbolic and Algebraic Computation*, ISSAC '00, pages 234–241, New York, NY, USA, 2000. ACM.

[58] B. Mourrain and Ph. Trébuchet. Generalized normal forms and polynomial system solving. In *Proceedings of the 2005 International Symposium on Symbolic and Algebraic Computation*, ISSAC '05, pages 253–260, New York, NY, USA, 2005. ACM.

[59] B. Mourrain and Ph. Trébuchet. Border basis representation of a general quotient algebra. In *Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation*, ISSAC '12, pages 265–272, New York, NY, USA, 2012. ACM.

[60] A. K. Narayanan. Fast computation of isomorphisms between finite fields using elliptic curves. In L. Budaghyan and F. Rodríguez-Henríquez, editors, *Arithmetic of Finite Fields. 7th International Workshop, WAIFI 2018, Bergen, Norway, June 14–16, 2018, Revised Selected Papers*, volume 11321 of *Lecture Notes in Comput. Sci.*, pages 74–91. Springer, Cham, 2018.

[61] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[62] P. Philippon. Sur des hauteurs alternatives. I. *Math. Ann.*, 289(1):255–283, 1991.

[63] A. Poteaux and É. Schost. On the complexity of computing with zero-dimensional triangular sets. *J. Symbolic Comput.*, 50:110–138, 2013.

[64] A. Schönhage. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Informatica*, 1(2):139–144, 1971.

[65] A. Schönhage, A. F. W. Grotefeld, and E. Vetter. *Fast algorithms: A multitape Turing machine implementation*. B. I. Wissenschaftsverlag, Mannheim, 1994.

[66] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.

[67] V. Shoup. New algorithms for finding irreducible polynomials over finite fields. *Math. Comp.*, 54(189):435–447, 1990.

[68] P. S. Wang. A p-adic algorithm for univariate partial fractions. In *Proceedings of the Fourth ACM Symposium on Symbolic and Algebraic Computation*, SYMSAC '81, pages 212–217, New York, NY, USA, 1981. ACM.

[69] R. Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings EUROSAM' 79*, number 72 in Lect. Notes Comput. Sci., pages 216–226. Springer-Verlag, 1979.