A Gröbner Free Alternative for Polynomial System Solving

Marc GIUSTI[1], Grégoire LECERF[1] and Bruno SALVY[2]

[1]UMS MEDICIS, Laboratoire GAGE, École polytechnique,
F-91128 Palaiseau Cedex, France

[2]Projet ALGO, INRIA Rocquencourt,
F-78153 Le Chesnay Cedex, France

January 18, 2000

Abstract

Given a system of polynomial equations and inequations with coefficients in the field of rational numbers, we show how to compute a geometric resolution of the set of common roots of the system over the field of complex numbers. A geometric resolution consists of a primitive element of the algebraic extension defined by the set of roots, its minimal polynomial and the parametrizations of the coordinates. Such a representation of the solutions has a long history which goes back to Leopold Kronecker and has been revisited many times in computer algebra.

We introduce a new generation of probabilistic algorithms where all the computations use only univariate or bivariate polynomials. We give a new codification of the set of solutions of a positive dimensional algebraic variety relying on a new global version of Newton's iterator. Roughly speaking the complexity of our algorithm is polynomial in some kind of degree of the system, in its height, and linear in the complexity of evaluation of the system.

We present our implementation in the Magma system which is called Kronecker in homage to his method for solving systems of polynomial equations. We show that the theoretical complexity of our algorithm is well reflected in practice and we exhibit some cases for which our program is more efficient than the other available software.

Keywords. Polynomial system solving, elimination, geometric resolution.

Contents

1 Introduction

We are interested in solving systems of polynomial equations, possibly including inequations. Let $f_1, \ldots, f_n$ and g be polynomials in $\mathbb{Q}[x_1, \ldots, x_n]$ such that the system $f_1 = \cdots = f_n = 0$ with $g \neq 0$ has only a finite set of solutions over the field $\mathbb{C}$ of complex numbers. We show how to compute a representation of this set in the form

$$q(T) = 0, \quad \left\{ \begin{array}{lcl} x_1 & = & v_1(T), \\ & \vdots & \\ x_n & = & v_n(T), \end{array} \right. \tag{1}$$

where q is a univariate polynomial with coefficients in $\mathbb{Q}$ and the v_i, $1 \leq i \leq n$, are univariate rational functions with coefficients in $\mathbb{Q}$.

Let us sketch our algorithm, which is incremental in the number of equations to be solved. At step i we have a resolution

$$q(T) = 0, \quad \left\{ \begin{array}{lcl} x_{n-i+1} & = & v_{n-i+1}(T), \\ & \vdots & \\ x_n & = & v_n(T), \end{array} \right. \tag{2}$$

of the solution set of

$$f_1 = \cdots = f_i = 0, \quad g \neq 0, \qquad x_1 = a_1, \ldots, x_{n-i} = a_{n-i},$$

where q is a univariate polynomial over $\mathbb{Q}$, the v_j are univariate rational functions over $\mathbb{Q}$ and the a_j are chosen generic enough in $\mathbb{Q}$. The variable T represents a linear form separating the solutions of the system: the linear form takes different values when evaluated on two different points that are solution of the system. From there, two elementary steps are performed. The first step is a Newton-Hensel lifting of the variable x_{n-i} in order to obtain a geometric resolution

$$Q(x_{n-i}, T) = 0, \quad \left\{ \begin{array}{lcl} x_{n-i+1} & = & V_{n-i+1}(x_{n-i}, T), \\ & \vdots & \\ x_n & = & V_n(x_{n-i}, T), \end{array} \right. \tag{3}$$

of the 1-dimensional solution set of

$$f_1 = \cdots = f_i = 0, \quad g \neq 0, \qquad x_1 = a_1, \ldots, x_{n-i-1} = a_{n-i-1},$$

where Q is polynomial in T and rational in x_{n-i} and the V_j are bivariate rational functions over $\mathbb{Q}$. The second step is the intersection of this 1-dimensional set with the solution set of the next equation $f_{i+1} = 0$, which leads to a geometric resolution like (2) for step $i + 1$.

At step i the system $f_1, \ldots, f_i$ defines a positive dimensional variety, the new codification of its resolution we propose here consists of a specialization of some variables and a resolution of the zero-dimensional specialized system. This representation makes the link between the positive and zero dimensions and relies on two main ideas: the *Noether position* and the *lifting fiber* (see §3).

History

The representation of a variety in the form of (1) above has a long history. To the best of our knowledge the oldest trace of this representation is to be found in Kronecker's work at the end of the 19th century [51] and a few years later in König's [48]. Their representation is naturally defined for positive dimensional algebraic varieties, for instance for a variety of codimension i it has the form:

$$q(x_1, \dots, x_{n-i}, T) = 0, \quad \left\{ \begin{array}{lcl} \frac{\partial q}{\partial T} x_{n-i+1} & = & w_{n-i+1}(x_1, \dots, x_{n-i}, T), \\ & \vdots & \\ \frac{\partial q}{\partial T} x_n & = & w_n(x_1, \dots, x_{n-i}, T), \end{array} \right. \tag{4}$$

where $q, w_{n-i+1}, \dots, w_n$ are polynomials in $x_1, \dots, x_{n-i}$ and T with coefficients in $\mathbb{Q}$ and such that q is square free. A good summary of their work can be found in Macaulay's book [58].

This representation has been used in computer algebra as a tool to obtain complexity results by many authors, in the particular zero-dimensional case: Chistov, Grigoriev [19], Canny [17], Gianni, Mora [32], Kobayashi, Fujise, Furukawa [47], Heintz, Roy, Solerno [46], Lakshman, Lazard [54], Renegar [65], Giusti, Heintz [34], Alonso, Becker, Roy, Wörman [6] and many others. From a practical point of view, the computation of such a representation is always relying on Gröbner basis computations [84], either with a pure lexicographical elimination order, or with an algorithm of change of basis [28, 20] or from any basis using a generalization of Newton's formulæ by Rouillier [66, 67].

In 1995, Giusti, Heintz, Morais and Pardo [36, 64] rediscovered Kronecker's approach without any prior knowledge of it and improved the space complexity but not the running time complexity. A first breakthrough was obtained by Giusti, Hägele, Heintz, Montaña, Morais, Morgenstern and Pardo [33, 35]: there exists an algorithm with a complexity roughly speaking polynomial in the degree of the system, in its height and in the number of the variables. Then, in [37], it is announced that the height of the integers does not appear in the complexity if the integers are represented by straight-line programs. For exact definitions and elementary properties of the notion of straight-line programs we refer to [77, 82, 76, 44]. A good historical presentation of all these works can be found in [18] and a didactic presentation of the algorithm is in [62]. We recall the main statement of these works:

Theorem [37] *Let g and $f_1, \dots, f_n$ be polynomials in $\mathbb{Q}[x_1, \dots, x_n]$. Suppose that $f_1, \dots, f_n$ define a reduced regular sequence in the open subset $\{g \neq 0\}$ of $\mathbb{C}^n$ and are of degree at most d, coded by straight-line programs of size at most L and height at most h. There is a bounded error Turing machine that outputs a geometric resolution of $\mathcal{V}(f_1, \dots, f_n) \backslash \mathcal{V}(g)$. The time complexity of the execution is in $L(nd\delta h)^{\mathcal{O}(1)}$ if we represent the integers of the output by straight-line programs.*

The *reduced* and *regular* hypothesis means that each variety

$$\mathcal{V}_i = \overline{\mathcal{V}(f_1, \dots, f_i) \backslash \mathcal{V}(g)}, \qquad 1 \le i \le n,$$

has dimension $n - i$ and for each $1 \le i \le n-1$ the localized quotient

$$\big(\mathbb{Q}[x_1, \dots, x_n]/(f_1, \dots, f_i)\big)_g$$

is reduced. Using the Jacobian criterion, this is equivalent to the situation when the Jacobian matrix of $f_1, \ldots, f_i$ has full rank at each generic point of $\mathcal{V}_i$. This condition is not really restrictive since we can perform a generic linear combination of the equations to recover this situation, as showed in [49, Proposition 37]. The number δ is defined as $\max(\deg(\mathcal{V}_1), \ldots, \deg(\mathcal{V}_{n-1}))$, it is bounded by d^n, by Bézout's theorem [43] (see also [81] and [30]). A precise definition of geometric resolutions is given in §3.2.

The geometric resolution returned by the algorithm underlying the above theorem has its integers represented by means of straight-line programs, the manipulation of such a representation has been studied in [41] and [40]. The size of the integers of the intermediate computations is bounded by the one of the output. In [18, Theorem 20] this result has been refined, showing how to compute efficiently only the solutions of height bounded by a given value.

Contributions

We have transformed this algorithm in order to obtain a new and simpler one, as above, without using straight-line programs anymore, neither for multivariate polynomials nor for integer numbers. We give a new estimate of the exponents of the complexity of Theorem [37] above improving the results of [45].

One main step of this transformation is obtained by a technique reminiscent of the deforestation [83], that we had already used in [39] to replace straight-line programs by an efficient use of specialization. We only need polynomials in at most two variables. From a geometrical point of view our algorithm only needs to compute the intersection of two curves. This improvement has been independently discovered in [45, Remark 13].

The second step is the use of Kronecker's form (4) to represent geometric resolutions, leading to a lower total degree complexity in the positive dimensional case. In [66, 67], this representation has also been used and its good behavior in practice in the zero dimensional case has been observed.

The third step is the use of a global Newton iterator presented in §2.2.1 and §4. This improves the original algorithm of [62, §4.2.1] by avoiding to compute a geometric resolution of each $\mathcal{V}_i$ from a *lifting fiber* (see §3.4) by means of primitive elements computations in two variables [62, Lema 54].

The fourth simplifying step is the use of a simple technique to intersect a variety by a hypersurface, which was already present in Kronecker's method presented in §2.2.2 and §6. This improves [45, §4.2] by avoiding the use of primitive elements computations in two variables which is used twice in [62]. This technique first appeared in [34] and developed in [50].

The last step is the intensive use of modular arithmetic: the resolution is computed modulo a small prime number, the integers are lifted at the end by our global Newton iterator. Hence the cost of integer manipulations is quite optimal: we never use integers more than twice as large as the ones contained in the output.

Results

We present three new results: the first one gives a new arithmetic complexity in terms of number of operations in the base field $\mathbb{Q}$, the second one is more realistic and takes care of the bit length of the integers and the third one consists in an implementation of our algorithm which demonstrates its tractability and efficiency.

For our complexity measurement we use the class of functions $\mathtt{M}$ defined by $\mathtt{M}(n) = \mathcal{O}(n \log^2(n) \log\log(n))$. As recalled in §3.5, if R is any unitary ring, it represents the complexity of the arithmetic operations in $R[T]$ for polynomials of degree at most n in terms of operations in R: addition, multiplication, division, resultant (if R is integral), greatest common divisor and interpolation (if R is a field). It is also the bit complexity of the arithmetic operations of the integers of bit-size at most n: addition, multiplication, division, greatest common divisor. The class $\mathcal{O}(n^\Omega)$ represents the complexity of the arithmetic operations of the matrix with coefficients in R of size $n \times n$ in terms of arithmetic operations in R: addition, multiplication, determinant and adjoint. We know that Ω is less than 4.

Theorem 1 *Let k be a field of characteristic zero, let $f_1, \ldots, f_n, g$ be polynomials in $k[x_1, \ldots, x_n]$ of degree at most d and given by a straight-line program of size at most L, such that $f_1, \ldots, f_n$ defines a reduced regular sequence in the open subset $\{g \neq 0\}$. The geometric resolution of the variety $\mathcal{V}(f_1, \ldots, f_n) \setminus \mathcal{V}(g)$ can be computed with $\mathcal{O}\big(n(nL + n^\Omega)(\mathtt{M}(d\delta))^2\big)$ arithmetic operations in k, where $\delta = \max(\deg(\mathcal{V}_1), \ldots, \deg(\mathcal{V}_{n-1}))$. There is a probabilistic algorithm performing this computation. Its probability of returning correct results relies on choices of elements of k. Choices for which the result is not correct are enclosed in strict algebraic subsets.*

The fact that bad choices are enclosed in strict algebraic subsets implies that almost all random choices lead to a correct computation. In this sense we can say that our probabilistic algorithm has a low probability of failure. Our algorithm is not Las Vegas, but it satisfies a weaker property: one can check that the geometric resolution it returns satisfies the input equations; if it does some of the solutions have been found but not necessarily all of them. In the special case when the output contains $\deg(f_1)\deg(f_2)\cdots\deg(f_n)$ solutions Bézout's theorem implies that all of them have been found.

In order to compare the complexity of our algorithm to Gröbner bases computations we apply our complexity theorem to the case of systems of polynomials $f_1, \ldots, f_n$ given by their dense representation:

Corollary 1 *Let $f_1, \ldots, f_n$ be a reduced regular sequence of polynomials of $k[x_1, \ldots, x_n]$ of degree at most d. Assume that d is at least n, then the geometric resolution of $\mathcal{V}(f_1, \ldots, f_n)$ can be computed with $\mathcal{O}(d^{3(n+\mathcal{O}(1))})$ arithmetic operations in k with the probabilistic algorithm of Theorem 1.*

Proof By Bézout's inequality $d\delta$ is at most d^n, so $\mathtt{M}(d\delta)$ is in $d^{n+\mathcal{O}(1)}$. And L is at most $n\binom{d+n}{n}$, which is in $d^{n+\mathcal{O}(1)}$. □

Our algorithm does not improve drastically the worst case complexity in case of dense input systems; its efficiency fully begins when either the complexity of evaluation of the input system is small or when the hypersurface $g = 0$ contains several components of each $\mathcal{V}_i$, i.e. δ is small with respect to d^n.

These results are proved for a field of characteristic 0 and are not valid for fields of positive characteristic. However, when k is equal to $\mathbb{Q}$, it is tempting to compute resolutions in $\mathbb{Z}/p\mathbb{Z}$ for some prime numbers p. We have a result in this direction: from a resolution computed modulo a *lucky prime number* p we can deduce the resolution in $\mathbb{Q}$, and p can be chosen small with respect to the integers of the output.

Theorem 2 *Assume that k is $\mathbb{Q}$, $\mathcal{V} = \overline{\mathcal{V}(f_1, \ldots, f_n) \setminus \mathcal{V}(g)}$ is zero-dimensional and $\big(\mathbb{Q}[x_1, \ldots, x_n]/(f_1, \ldots, f_n)\big)_g$ is reduced.*

Let u be a primitive element of the extension $\mathbb{Q} \longrightarrow \mathbb{Q}[\mathcal{V}]$, $q(T)$ its monic minimal polynomial in $\mathbb{Q}[T]$. Let D be the degree of q, D is equal to $\deg(\mathcal{V})$. Let $w_i(T)$, $1 \leq i \leq n$, be polynomials of $\mathbb{Q}[T]$ of degree strictly less than D such that $q'(u)x_i - w_i(u)$ is equal to zero in $\mathbb{Q}[\mathcal{V}]$.
If we are given

- *η the bit-size of the integers of the polynomials q and w_i;*
- *a prime number p not dividing any denominator appearing in q and the w_i and such that $\log(p) < \eta$;*
- *q_p and $w_{1,p}, \ldots, w_{n,p}$ polynomials in $\mathbb{Z}/p\mathbb{Z}[T]$, images of q and $w_1, \ldots, w_n$,*

such that

- *q_p' is invertible modulo q_p;*
- *for each i, $1 \leq i \leq n$, $f_i(w_{1,p}/q_p', \ldots, w_{n,p}/q_p') \equiv 0\ [q_p]$;*
- *the Jacobian matrix J of the f_i is invertible: $\det(J(w_{1,p}/q_p', \ldots, w_{n,p}/q_p'))$ is invertible modulo q_p,*

then the polynomials q and the w_i can be reconstructed in the bit-complexity

$$\mathcal{O}\big((nL + n^{\Omega})\mathtt{M}(D)\mathtt{M}(\eta)\big).$$

From a practical point of view we combine the algorithms related to Theorems 1 and 2 in the following way: first choose at random a small prime number, compute a geometric resolution of the input system modulo p and then lift the integers to get the geometric resolution in $\mathbb{Q}$.

The problem of choosing a prime number for which this algorithm leads to a correct result is similar to the problem of computing the greatest common divisor of two univariate polynomials over $\mathbb{Q}$ by means of modular computations and Hensel's lifting (for example see [23, §4.1.1] or [31, §7.4]). The description of the probability of choosing a *lucky* p is out of the scope of this work but such considerations are as in [40, 41, 42].

The probability of failure of the algorithm given in [45] has been studied using Zippel-Schwartz's zero test [85, 72] for multivariate polynomials. We could use the same analysis here to quantify the probability mentioned in Theorem 1, but this has no practical interest without the quantification of the probability of choosing a lucky prime number p. These probabilities will be studied in forthcoming works.

Implementation: the Kronecker Package

One aim of this article is to demonstrate that our algorithm has a practical interest and is competitive with the other methods. We have implemented our algorithm within the computer algebra system Magma [1, 16, 12], the package has been called Kronecker [55] and is available with its documentation at `http://www.gage.polytechnique.fr/~lecerf/software/kronecker/`.

We compare our implementation to Gröbner bases computations for total degree orders and algorithms of change of bases. Given a Gröbner basis of a zero-dimensional polynomial equation system one can deduce a Rational Univariate Representation of the zeros via the algorithm proposed in [66, 67]. We also compare our implementation to the one of [66].

This article is organized as follows. The next section is devoted to an informal presentation of the whole algorithm reflecting the actual computations performed in a generic situation. We then give definitions and introduce our encoding of the solutions. The next three sections are devoted to the formal presentation and proofs of our Newton iterator and the intersection algorithm. Section 7 presents the whole algorithm and specifies the random choices. The last part provides some practical aspects of our implementation in the Magma system and comparisons with other methods for solving systems of polynomial equations.

Acknowledgment: We greatly thank L. M. Pardo who has exhumed Kronecker's work and brought us very useful comments, J. Cannon and all the Magma team for their very efficient support, and É. Schost for having been the first user of Kronecker.

2 Description of the Algorithm

We first give an informal presentation of the probabilistic aspects of our algorithm. Then we show the actual computations that are performed in a generic case, forgetting the modular computational aspects for the moment. All these points are detailed in the next sections.

2.1 Outlook of the Probabilistic Aspects

Let k be a field of characteristic zero, and $f_1, \ldots, f_n, g$ be polynomials in the ring $k[x_1, \ldots, x_n]$ under the hypotheses of Theorem 1. The system

$$\mathcal{S} = \{f_1 = \cdots = f_n = 0, g \neq 0\}$$

has only a finite set of solutions in the n-affine space over an algebraic closure of k. Our algorithm is parametrized by three parameters $\mathcal{N}$, $\mathcal{L}$, $\mathcal{C}$, called respectively the *Noether points*, *lifting points* and *Cayley points*: they are functions returning tuples of integers (see §7.2). Once the parameters are fixed this specifies a deterministic algorithm $\mathcal{A}_{\mathcal{N},\mathcal{L},\mathcal{C}}$ for the resolution of $\mathcal{S}$. For a proper choice of these parameters, the algorithm $\mathcal{A}_{\mathcal{N},\mathcal{L},\mathcal{C}}$ computes a resolution of the set of solutions of the system in the form:

$$q(T) = 0, \quad \begin{cases} x_1 &= & v_1(T), \\ & \vdots & \\ x_n &= & v_n(T), \end{cases} \tag{5}$$

where $q, v_1, \ldots, v_n \in k[T]$ and T represents a k-linear form in the x_i.

The time complexity of the execution of $\mathcal{A}_{\mathcal{N},\mathcal{L},\mathcal{C}}$ for such a proper choice is $L(ndh\delta)^{\mathcal{O}(1)}$. It has been shown in [33] that the choices of the parameters can be done using Correct Test Sequences of size polynomial in the sequential complexity of the algorithm. In [45, Theorem 5] it is shown using Zippel-Schwartz's equality test [85, 72] that the choices can be done at random in a set of integers of size polynomial in the sequential complexity of $\mathcal{A}_{\mathcal{N},\mathcal{L},\mathcal{C}}$ with a uniformly bounded probability of failure less than 1/2.

In the case of our algorithm, we precise these parameters in §7 and we show that we can choose them in a Zariski open subset of the space of choices. In particular this means that any random choice suits the input system.

We say that our algorithm is *semi-numerical* since it is parametrized by some initial choices in the same way as some numerical algorithms are. Our advantage over numerical algorithms is the *certification* of the result. In [18] a comparison is made between our method and the numerical approach using homotopy and the approximate zero theory introduced by Smale [74, 75].

2.2 Description of the Computations

We present now the actual computations performed by our algorithm in a generic case. Our algorithm is incremental in the number of equations to be solved. Let $\mathcal{S}_i$ be the system of polynomial equations

$$x_1 = \cdots = x_{n-i} = f_1 = \cdots = f_i = 0, \quad g \neq 0.$$

The algorithm solves $\mathcal{S}_1, \ldots, \mathcal{S}_n$ in sequence. We enter step i with a solution of $\mathcal{S}_i$ in the form

$$q(T) = 0, \quad \begin{cases} x_{n-i+1} &= & T, \\ x_{n-i+2} &= & v_{n-i+2}(T), \\ & \vdots & \\ x_n &= & v_n(T), \end{cases} \tag{6}$$

with the property that x_{n-i+1} separates the points of $\mathcal{S}_i$. We want to compute such a solution for $\mathcal{S}_{i+1}$. The computation divides into three main parts: the lifting, the intersection and the cleaning steps.

2.2.1 The Lifting Step

Starting from (6), we compute a solution of the system $\mathcal{S}_i'$

$$x_1 = \cdots = x_{n-i-1} = f_1 = \cdots = f_i = 0, \quad g \neq 0,$$

in the form

$$Q(x_{n-i},T)=0,\quad \left\{\begin{array}{lcl} x_{n-i+1} & = & T, \\ \frac{\partial Q}{\partial T}(x_{n-i},T)x_{n-i+2} & = & W_{n-i+2}(x_{n-i},T), \\ & \vdots & \\ \frac{\partial Q}{\partial T}(x_{n-i},T)x_{n} & = & W_{n}(x_{n-i},T), \end{array}\right. \tag{7}$$

such that the W_j and Q are polynomials in x_{n-i} and T. The solution $\mathbf{v} = (T, v_{n-i+2}(T), \ldots, v_n(T))$ from (6) can be seen as an approximated solution of $\mathcal{S}'_i$ at precision $\mathcal{O}(x_{n-i})$. We now show how it can be lifted to a solution at precision $\mathcal{O}(x_{n-i}^2)$.

We first compute, with the classical Newton method,

$$\begin{pmatrix} V_{n-i+1}(x_{n-i},T) \\ \vdots \\ V_n(x_{n-i},T) \end{pmatrix} = \mathbf{v}^t - J(0,\ldots,0,x_{n-i},\mathbf{v})^{-1} \begin{pmatrix} f_1(0,\ldots,0,x_{n-i},\mathbf{v}) \\ \vdots \\ f_i(0,\ldots,0,x_{n-i},\mathbf{v}) \end{pmatrix},$$

modulo $q(T)$ and at precision $\mathcal{O}(x_{n-i}^2)$, where J is the Jacobian matrix of $f_1, \ldots, f_i$ with respect to $x_{n-i+1}, \ldots, x_n$. The parametrization

$$q(T)=0,\quad \left\{\begin{array}{lcl} x_{n-i+1} & = & V_{n-i+1}(x_{n-i},T), \\ & \vdots & \\ x_{n} & = & V_{n}(x_{n-i},T), \end{array}\right. \tag{8}$$

is a solution of $\mathcal{S}'_i$ at precision $\mathcal{O}(x_{n-i}^2)$. The expression V_{n-i+1} can also be written

$$V_{n-i+1}(x_{n-i},T) = T + x_{n-i}\Delta(T) + \mathcal{O}(x_{n-i}^2).$$

Hence

$$T = x_{n-i+1} - x_{n-i}\Delta(x_{n-i+1}) + \mathcal{O}(x_{n-i}^2).$$

Substituting the right-hand side for T in q and the V_j we get:

$$q(x_{n-i+1}) - x_{n-i}(q'(x_{n-i+1})\Delta(x_{n-i+1}) \bmod q(x_{n-i+1})) + \mathcal{O}(x_{n-i}^2) = 0$$

and

$$x_j = V_j(x_{n-i},x_{n-i+1}) - x_{n-i}(\frac{\partial V_j}{\partial T}\Delta(x_{n-i+1}) \bmod q(x_{n-i+1})) + \mathcal{O}(x_{n-i}^2),$$

for $n-i+1 \leq j \leq n$, which is an approximated solution of $\mathcal{S}'_i$ at precision $\mathcal{O}(x_{n-i}^2)$. We continue this process up to a certain precision. At the end, multiplying both sides of the parametrization of the coordinates by the derivative of q with respect to T and reducing the right-hand side with respect to q, we get the resolution (7) exactly. Section 4 gives the full description of this method.

Compared to the original algorithm in [62], this method shortcuts the reconstruction of the whole geometric resolution from a fiber by means of primitive element computations in two variables. Compared to [45], we only need to perform the lifting at precision the degree of the current variety. This method also applies for integers to lift a geometric resolution known modulo a prime number p, see §4.6.

2.2.2 The Intersection Step

To the solution (7) of $\mathcal{S}'_i$ we add the new equation $f_{i+1} = 0$. Let X be a new variable, we first perform the following change of variables in the power series ring $k[[t]]$:

$$x_{n-i} = X - t x_{n-i+1} + \mathcal{O}(t^2).$$

This leads to a new parametrization in the form

$$Q_t(X,T) = 0, \qquad \left\{ \begin{array}{lcl} x_{n-i+1} & = & T, \\ x_{n-i+2} & = & V_{t,n-i+2}(X,T), \\ & \vdots & \\ x_n & = & V_{t,n}(X,T), \end{array} \right. \tag{9}$$

where Q_t is a polynomial in X and T and the $V_{t,j}$ are polynomial in T and rational in X with coefficients in $k[[t]]$ at precision $\mathcal{O}(t^2)$. Then we compute

$$\begin{array}{ll} A(X) & = \mathrm{Resultant}_T(Q_t(X,T), \\ & f_{i+1}(0,\ldots,0,X - tT, T, V_{t,n-i+2}(X,T),\ldots,V_{t,n}(X,T))). \end{array}$$

The resultant $A(X)$ is indeed in $k[X][[t]]$ and replacing X by $x_{n-i} + t x_{n-i+1}$ in

$$A(X) = a_0(X) + t a_1(X) + \mathcal{O}(t^2) = 0,$$

we get:

$$a_0(x_{n-i}) = 0, \qquad a_0'(x_{n-i}) x_{n-i+1} + a_1(x_{n-i}) = 0,$$

which gives the desired resolution of $\mathcal{S}_i \cup \{f_{i+1} = 0\}$. If a_0 is not relatively prime with its first derivative a_0', we replace a_0 by its square free part a_s and let $a_m = a_0/a_s$, a_m divides a_0' and a_1. The parametrization becomes: $a_0'/a_m x_{n-i+1} + a_1/a_m = 0$. Then a_0'/a_m is relatively prime with a_0. These computations are described in more detail in §6.

This method simplifies considerably the ones given in the original algorithm [62] and [45] relying on primitive element computations in two variables.

2.2.3 The Cleaning Step

We now have a resolution of $\mathcal{S}_i \cup \{f_{i+1} = 0\}$ in the form

$$q(T) = 0, \qquad \left\{ \begin{array}{lcl} x_{n-i} & = & T, \\ x_{n-i+1} & = & v_{n-i+1}(T), \\ x_{n-i+2} & = & v_{n-i+2}(T), \\ & \vdots & \\ x_n & = & v_n(T), \end{array} \right. \tag{10}$$

where q and the v_j are new polynomials in T. To get a resolution of $\mathcal{S}_{i+1}$ we must remove the points contained in the hypersurface $g = 0$. To do this, we compute the greatest common divisor:

$$c(T) = \gcd_T(q, g(0,\ldots,0,T,v_{n-i+1},\ldots,v_n)).$$

Then we just have to replace q by q/c and reduce the parametrizations v_j by the new polynomial q. This algorithm relies on Proposition 8: it simplifies [62, §4.3.1].

The rest of this article is devoted to the justifications of these computations and to the comparison of practical results with some other methods.

3 Definitions and Basic Statements

One key feature of our algorithm is an effective use of the *Noether Normalization Lemma* also seen geometrically as a *Noether Position*. It allows us to represent a positive dimensional variety as a zero-dimensional one.

3.1 Noether Position, Primitive Element

Let k be a field of characteristic 0. Let $x_1, \ldots, x_n$ be indeterminates over k. Let $\mathcal{V}$ be a r-dimensional k-variety in $\overline{k}^n$, where $\overline{k}$ is the algebraic closure of k and $\mathfrak{I} = \mathfrak{I}(\mathcal{V})$ the annihilating ideal of $\mathcal{V}$.

We say that a subset of variables $Z = \{x_{i_1}, \ldots, x_{i_k}\}$ is *free* when $\mathfrak{I} \cap k[x_{i_1}, \ldots, x_{i_k}] = (0)$. A variable is *dependent* or *integral* with respect to a subset of variables Z if there exists in $\mathfrak{I}(\mathcal{V})$ a monic polynomial annihilating it and whose coefficients are polynomial in the variables of Z only.

A *Noether normalization* of $\mathcal{V}$ consists of a k-linear change of variables, transforming the variables $x_1, \ldots, x_n$ into new ones, $y_1, \ldots, y_n$, such that the linear map from $\overline{k}^n$ to $\overline{k}^r$ $(r \leq n)$ defined by the forms $y_1, \ldots, y_r$ induces a finite surjective morphism of affine varieties $\pi \ : \ \mathcal{V} \longrightarrow \overline{k}^r$. This is equivalent to the fact that the variables $y_1, \ldots, y_r$ are free and $y_{r+1}, \ldots, y_n$ dependent with respect to the first ones. In this situation we say that $y_1, \ldots, y_n$ are in Noether position.

If B is the coordinate ring $k[\mathcal{V}]$, then a Noether normalization induces an integral ring extension $R := k[y_1, \ldots, y_r] \longrightarrow B$. Let K be the field of fractions of R and B' be $K \otimes_R B$, B' is a finite-dimensional K-vector space.

Example 1 Consider $f = x_1x_2$ in $\mathbb{Q}[x_1, x_2]$, f defines a hypersurface in the affine space of dimension two over the complex numbers. The variable x_1 is free but x_2 is not integral over x_1. This hypersurface is composed of two irreducible components $x_1 = 0$ and $x_2 = 0$. When specializing the variable x_1 to any value p_1 in k^*, $f(p_1, x_2)$ has one irreducible factor only. Let us take $y_1 = x_1 - x_2$ and $y_2 = x_2$ then f becomes $(y_1 + y_2)y_2 = y_2^2 + y_1y_2$. The variable y_2 is integral over y_1: we have a Noether position of this hypersurface; we can specialize y_1 to 0 in f, there remains two irreducible components.

Example 2 Consider the hypersurface given by the equation $x_2 - x_1^2 = 0$. The variables x_1, x_2 are in Noether position but when specializing x_1 to a point of k, for instance 0, the fiber contains only one point while the hypersurface has degree 2. The vector space B' is $k(x_1)[x_2]/(x_2 - x_1^2)$ and has dimension one only.

The degeneration of the dimension of B' in the last example does not occur when working with projective varieties, so if we want to avoid it in affine spaces we need a kind of stronger Noether position.

We say that the variables $y_1, \ldots, y_n$ are in *projective Noether position* if they define a Noether position for the projective algebraic closure of $\mathcal{V}$. More precisely, let x_0 be a new variable, to any polynomial f of $k[x_1, \ldots, x_n]$, we write

$f^h(x_0, \ldots, x_n)$ the homogenization of f with respect to x_0, $\mathfrak{I}^h$ denotes the ideal of the homogenized polynomial of $\mathfrak{I}$ and $\mathcal{V}^h$ the variety associated to $\mathfrak{I}^h$, which corresponds to the projective closure of $\mathcal{V}$. We say that the variables $y_1, \ldots, y_n$ are in projective Noether position with respect to $\mathcal{V}$ when $x_0, y_1, \ldots, y_n$ are in Noether position with respect to $\mathcal{V}^h$.

In the rest of the paper we only use projective Noether positions, so we only say Noether position. We write $\mathfrak{I}'$ for the extension of $\mathfrak{I}$ in $K[y_{r+1}, \ldots, y_n]$. $\mathfrak{I}'$ is a zero-dimensional radical ideal. We are interested in some particular bases of B':

Definition 1 *A k-linear form $u = \lambda_{r+1}y_{r+1} + \cdots + \lambda_n y_n$ such that the powers $1, u, \ldots, u^{\deg(\mathcal{V})-1}$ form a basis of the vector space B' is called a* primitive element *of the variety $\mathcal{V}$.*

In general we do not know any efficient way to compute in B. Even when it is a free module we do not know bases of small size [5]. The next two propositions give some properties of computations in B'.

Proposition 1 *With the above notations, assume that $\mathcal{V}$ is r-equidimensional. If the variables $x_1, \ldots, x_n$ are in projective Noether position with respect to $\mathcal{V}$ then the dimension of B' is the degree of $\mathcal{V}$.*

We recall a result [68, Proposition 1], itself a continuation of [15, Remark 9]:

Proposition 2 *Let $\mathfrak{I}$ be a radical ideal of $k[x_1, \ldots, x_n]$ such that $\mathcal{V} = \mathcal{V}(\mathfrak{I})$ is r-equidimensional and the variables $x_1, \ldots, x_n$ are in Noether position. Let f be an element of $k[x_1, \ldots, x_n]$ and $\overline{f}$ its class in the quotient ring B. Let T be a new variable, then there exists a monic polynomial $F \in R[T]$ which satisfies $F(\overline{f}) = 0$ and whose total degree is bounded by $\deg(\mathcal{V})\deg(f)$.*

An alternative proof of this proposition is given in [62, Corolario 21]. The next corollary expresses that minimal and characteristic polynomials in B' have their coefficients in R.

Corollary 2 *Let $\mathfrak{I}$ be a radical ideal of $k[x_1, \ldots, x_n]$ such that $\mathfrak{I}$ is r-equidimensional and the variables $x_1, \ldots, x_n$ are in Noether position. Let f be a polynomial in $k[x_1, \ldots, x_n]$. Then the characteristic polynomial χ of the endomorphism of multiplication by f in B' belongs to $k[x_1, \ldots, x_r][T]$. Its coefficient of degree i in T has degree at most $(\delta - i)\deg(f)$, where $\delta = \dim(B')$. In the case when $f = u$ is a primitive element of $\mathcal{V}$ we have $\chi(\overline{u}) = 0$.*

Proof Let F be an integral dependence relation of f modulo the ideal $\mathfrak{I}$ of degree bounded by $\deg(\mathcal{V})\deg(f)$ from Proposition 2, M_f be the endomorphism of multiplication by f in B' and μ its minimal polynomial. First we note that $F(M_f) = 0$, thus μ divides F. The polynomials μ and F being monic we deduce using Gauss lemma that μ is in $R[T]$ and so is χ. If $f = u$, $\deg_T(\mu) = \deg_T(\mathcal{V})$ and thus $\mu = F$.

Let us now prove the bound on the degrees, to do this we homogenize the situation: let x_0 be a new variable and f^h denotes the homogenized polynomial of f, $\mathfrak{I}^h$ the homogenized ideal of $\mathfrak{I}$. Let now B' be $k(x_0, \ldots, x_r)[x_{r+1}, \ldots, x_n]/\mathfrak{I}^h$ and $\chi(T)$ the characteristic polynomial of the endomorphism of multiplication by f^h in B'. It is sufficient to prove that the coefficient of degree i in T of χ is homogeneous of degree $(\delta - i)\deg(f)$. To do this let $\overline{K}$ be the algebraic closure of $k(x_0, \dots, x_r)$ and $Z_1, \dots, Z_\delta$ be the zeroes of $\mathfrak{I}^h$ in $\overline{K}$. The following formula holds:

$$\chi(x_0, \dots, x_r, T) = \prod_{i=1}^{\delta} (T - f^h(x_0, \dots, x_r, Z_i)).$$

Hence, if t is a new variable we have

$$\begin{aligned}\chi(tx_0, \dots, tx_r, T) &= \prod_{i=1}^{\delta} (T - f^h(tx_0, \dots, tx_r, tZ_i)) \\ &= \prod_{i=1}^{\delta} (T - t^{\deg(f)} f^h(x_0, \dots, x_r, Z_i)).\end{aligned}$$

Expanding this last expression, we get the claimed bound on the degrees of the coefficients in T of χ, this concludes the proof. □

3.2 Geometric Resolutions

Let $\mathcal{V}$ be a r-equidimensional algebraic variety and $\mathfrak{I}$ its annihilator ideal in the ring $k[x_1, \dots, x_n]$. A *geometric resolution* of $\mathcal{V}$ is given by:

- an invertible $n \times n$ square matrix M with entries in k such that the new coordinates $y = M^{-1}x$ are in Noether position with respect to $\mathcal{V}$;
- a primitive element $u = \lambda_{r+1} y_{r+1} + \dots + \lambda_n y_n$ of $\mathcal{V}$;
- the minimal polynomial $q(T) \in R[T]$ of u in B', monic in T, and
- the *parametrization* of $\mathcal{V}$ by the zeros of q, given by polynomials

  $$v_{r+1}(y_1, \dots, y_r, T), \dots, v_n(y_1, \dots, y_r, T) \in K[T],$$

  such that $y_j - v_j(y_1, \dots, y_r, u) \in \mathfrak{I}'$ for $r + 1 \leq j \leq n$, where $\mathfrak{I}'$ is the extension of $\mathfrak{I}$ in $k(y_1, \dots, y_r)[y_{r+1}, \dots, y_n]$ and $\deg_T(v_j) < \deg_T(q)$.

Given a primitive element u, its minimal polynomial q is uniquely determined up to a scalar factor. The parametrization can be expressed in several ways. In the definition of geometric resolutions the parametrization of the algebraic coordinates has the form

$$y_j = v_j(T), \qquad r + 1 \leq j \leq n.$$

However, given any polynomial $p(T) \in K[T]$ relatively prime with $q(T)$ another parametrization is given by:

$$p(T) y_j = v_j(T) p(T), \qquad r + 1 \leq j \leq n.$$

One interesting choice is to express the parametrization in the following way:

$$\frac{\partial q}{\partial T}(T) y_j = w_j(T), \qquad r + 1 \leq j \leq n, \tag{11}$$

with $\deg_T w_j < \deg_T q$.

Definition 2 *We call a parametrization in the form of Equation (11) a* Kronecker parametrization.

Proposition 3 *The polynomial q has its coefficients in R and in a Kronecker parametrization such as (11) the polynomials w_i have also their coefficients in R instead of K. The* total *degree of q and the w_i is bounded by $\deg_T(q)$. Moreover $q(u)$ and $\frac{\partial q}{\partial T}(u)x_j - w_j(u)$ belong to $\mathfrak{I}$, for $r+1 \le j \le n$.*

In particular the discriminant of q with respect to T is a multiple of any denominator appearing in any kind of parametrization.

> **Example 3** Let $f_1 = x_3^2 + x_1x_2 + 1$ and $f_2 = x_2^2 + x_1x_3$, the variables x_1, x_2, x_3 are in Noether position, x_2 is a primitive element and we have the following Kronecker parametrization
>
> $$x_2^4 + x_1^3x_2 + x_1^2 = 0,$$
>
> $$(4x_2^3 + x_1^3)x_3 = 4x_1x_2 + 3x_1^2x_2^2.$$

The following is a fundamental result.

Proposition 4 *Given a Noether position and a primitive element, any r-equidimensional algebraic variety $\mathcal{V}$ admits a unique geometric resolution.*

The proofs of the last two propositions are given in the next section.

> **Example 4** Here is an example of an ideal which is not Cohen-Macaulay: in $k[x_1, x_2, x_3, x_4]$, consider
>
> $$\mathfrak{I} = (x_2x_4, x_2x_3, x_1x_4, x_1x_3).$$
>
> $\mathfrak{I}$ is radical 2-equidimensional. A Noether position is given by $x_1 = y_3 - y_1, x_2 = y_4 - y_2, x_3 = y_3, x_4 = y_4$. The generating equations become $y_4^2 - y_2y_4, y_3y_4 - y_2y_3, y_3y_4 - y_1y_4, y_3^2 - y_1y_3$. For any $\lambda_3, \lambda_4 \in k$ and $u = \lambda_3y_3 + \lambda_4y_4$ we have $u^2 - \lambda_4y_2u - \lambda_3y_1u \in \mathfrak{I}$.

3.3 Generic Primitive Elements

Assume that $\mathfrak{I}$ is radical and equidimensional of dimension r and the variables $x_1, \ldots, x_n$ are in Noether position. The minimal polynomial of a generic primitive element is of great importance in algebraic geometry and computer algebra. It was already used by Kronecker as an effective way to compute geometric resolutions.

Let Λ_i be new variables, $i = r+1, \ldots, n$, $k_\Lambda = k(\Lambda_{r+1}, \ldots, \Lambda_n)$, and $R_\Lambda = k_\Lambda[x_1, \ldots, x_r]$. Let $\mathfrak{I}_\Lambda$ be the extension of $\mathfrak{I}$ in $k_\Lambda[x_1, \ldots, x_n]$. Let $u_\Lambda = \Lambda_{r+1}x_{r+1} + \cdots + \Lambda_nx_n$. The objects indexed with Λ are related to objects defined over k_Λ. The generic linear form u_Λ is a primitive element of $\mathfrak{I}_\Lambda$, let U_Λ be its characteristic polynomial in $B'_\Lambda := k_\Lambda(x_1, \ldots, x_r)[x_{r+1}, \ldots, x_n]/\mathfrak{I}_\Lambda$. From Corollary 2, the polynomial $U_\Lambda(x_1, \ldots, x_r, T)$ is square free, monic in T, of total degree equal to the degree of the variety corresponding to $\mathfrak{I}$, has its coefficients in R_Λ and we have

$$U_\Lambda(x_1, \ldots, x_r, u_\Lambda) \in \mathfrak{I}_\Lambda.$$

Differentiating U_Λ with respect to $\Lambda_{r+1}, \ldots, \Lambda_n$, we deduce the following geometric resolution of $\mathfrak{I}_\Lambda$:

$$U_\Lambda(x_1, \ldots, x_r, T) = 0,$$
$$\left\{ \begin{array}{rcl} \frac{\partial U_\Lambda}{\partial T}(x_1, \ldots, x_r, T)x_{r+1} & = & -\frac{\partial U_\Lambda}{\partial \Lambda_{r+1}}(x_1, \ldots, x_r, T), \\ & \vdots & \\ \frac{\partial U_\Lambda}{\partial T}(x_1, \ldots, x_r, T)x_n & = & -\frac{\partial U_\Lambda}{\partial \Lambda_n}(x_1, \ldots, x_r, T). \end{array} \right. \tag{12}$$

This proves Propositions 3 and 4.

Example 5 In the previous example with $\lambda_3\lambda_4 \neq 0$, we deduce the parameterization

$$u^2 - \lambda_4 y_2 u - \lambda_3 y_1 u = 0, \quad \left\{ \begin{array}{rcl} (2u - \lambda_4 y_2 - \lambda_3 y_1)y_3 & = & y_1 u, \\ (2u - \lambda_4 y_2 - \lambda_3 y_1)y_4 & = & y_2 u. \end{array} \right.$$

3.4 Lifting Fibers

Instead of processing the representation of univariate polynomials over the free variables we make an intensive use of specialization. Thanks to our lifting process presented in §4 we do not lose anything.

From now on we assume that $\mathcal{V}$ is an r-equidimensional variety which is a sub-variety of $\mathcal{V}(f_1, \ldots, f_{n-r})$, where $f_1, \ldots, f_{n-r}$ define a reduced regular sequence of polynomials at each generic point of $\mathcal{V}$. We call such a sequence of polynomials a *lifting system* of $\mathcal{V}$. Let $y_1, \ldots, y_n$ be new coordinates bringing $\mathcal{V}$ into a Noether position. We recall that π represents the finite projection morphism onto the free variables.

Definition 3 *A point $p = (p_1, \ldots, p_r)$ in k^r is called a* lifting point *of $\mathcal{V}$ with respect to the lifting system $f_1, \ldots, f_{n-r}$ if the Jacobian matrix of $f_1, \ldots, f_{n-r}$ with respect to the dependent variables $y_{r+1}, \ldots, y_n$ is invertible at each point of $\pi^{-1}(p)$.*

Our encoding of the geometric resolution is given by a specialization of the geometric resolution at a lifting point.

Definition 4 *A* lifting fiber *of $\mathcal{V}$ is given by:*

- *a lifting system $\mathbf{f} = (f_1, \ldots, f_{n-r})$ of $\mathcal{V}$;*
- *an invertible $n \times n$ square matrix M with entries in k such that the new coordinates $y = M^{-1}x$ are in Noether position with respect to $\mathcal{V}$;*
- *a lifting point $p = (p_1, \ldots, p_r)$ for $\mathcal{V}$ and the lifting system;*

- *a primitive element* $u = \lambda_{r+1} y_{r+1} + \cdots + \lambda_n y_n$ *of* $\mathcal{V}_p = \pi^{-1}(p)$*;*
- *the minimal polynomial* $q(T) \in k[T]$ *annihilating* u *over the points of* $\mathcal{V}_p$*;*
- $n - r$ *polynomials* $\mathbf{v} = (v_{r+1}, \ldots, v_n)$ *of* $k[T]$*, of degree strictly less than* $\deg_T(q)$*, giving the* parametrization *of* $\mathcal{V}_p$ *by the zeros of* q*:* $y_j - v_j(u) = 0$ *for all* $r+1 \leq j \leq n$ *and all roots* u *of* q*.*

We have the following relations between the components of the lifting fiber:

$$u(v_{r+1}(T), \ldots, v_n(T)) = T,$$

$$\mathbf{f} \circ M(p_1, \ldots, p_r, v_{r+1}(T), \ldots, v_n(T)) \equiv 0 \bmod q(T).$$

The following proposition explains the one to one correspondence between geometric resolutions and lifting fibers. The specialization of the free variables at a lifting point constitutes the main improvement of complexity of our algorithm: compared to rewriting techniques such as Gröbner bases computations, we do not have to store multivariate polynomials, but only univariate ones.

Proposition 5 *For any lifting fiber encoding a variety* $\mathcal{V}$ *there exists a unique geometric resolution of* $\mathcal{V}$ *for the same Noether position and primitive element. The specialization of the minimal polynomial and the parametrization of this geometric resolution on the lifting point gives exactly the minimal polynomial and the parametrization of the lifting fiber. We have* $\deg(\mathcal{V}_p) = \deg(\mathcal{V})$*.*

Proof First, the equality $\deg(\mathcal{V}_p) = \deg(\mathcal{V})$ is a direct consequence of the definition of the degree and the choice of p.

Suppose now that the primitive element u for $\mathcal{V}_p$ is not primitive for $\mathcal{V}$. We can choose a primitive element u' of $\mathcal{V}$ which is also a primitive element for $\mathcal{V}_p$. The specialization of the corresponding Kronecker parametrization of $\mathcal{V}$ with respect to u' gives a parametrization of $\mathcal{V}_p$. Using the powers of u' as a basis of B', we can compute the minimal polynomial of u, of degree strictly less than δ. Its denominators do not vanish at p, hence its specialization at p gives an annihilating polynomial of u for $\mathcal{V}_p$ of degree strictly less than δ. This leads to a contradiction. This concludes the proof. □

We now show that lifting points and primitive elements can be chosen at random with a low probability of failure in practice.

Lemma 1 *With the above notations and assumptions, the points*

$$(p_1, \ldots, p_r, \lambda_{r+1}, \ldots, \lambda_n) \in k^n$$

such that $(p_1, \ldots, p_r)$ *is not a lifting point or* $u = \lambda_{r+1} y_{r+1} + \cdots + \lambda_n y_n$ *is not a primitive element for* $\mathcal{V}_p$ *are enclosed in a strict subset of* k^n *which is algebraic.*

Proof Let J be the Jacobian matrix of $f_1, \ldots, f_{n-r}$ with respect to the variables $y_{r+1}, \ldots, y_n$ and $F(T)$ be an integral dependency relation of $\det(J)$ modulo $\mathcal{V}$. By hypothesis $\det(J)$ is not a zero divisor in B. Hence the constant coefficient $A(y_1, \ldots, y_r)$ of F is not zero and satisfies $A \in \mathfrak{I} + (\det(J))$. Each point p such that $A(p) \neq 0$ is a lifting point.

Now fix a lifting point p and consider U_Λ of §3.3 for $\mathcal{V}_p$, then any point $\Lambda_{r+1} = \lambda_{r+1}, \ldots, \Lambda_n = \lambda_n$ such that the discriminant of U_Λ does not vanish is a primitive element of $\mathcal{V}_p$. □

Notations for the Pseudo-Code: For the pseudo-code of the algorithms we use the following notations. If F denotes the lifting fiber: $F_{ChangeOfVariables}$ is M, $F_{PrimitiveElement}$ is u, $F_{LiftingPoint}$ is p, $F_{MinimalPolynomial}$ is q, $F_{Parametrization}$ is $\mathbf{v}$ and $F_{Equations}$ is $\mathbf{f}$. We assume we have the following functions on F:

Dimension: Lifting Fiber$\longrightarrow$ Integers: $F \mapsto r$ and

Degree: Lifting Fiber$\longrightarrow$ Integers: $F \mapsto \deg_T(F_{MinimalPolynomial})$.

3.5 Complexity Notations

We now discuss the complexity of integer and polynomial arithmetic. In the whole paper $\mathtt{M}(n)$ denotes $\mathcal{O}(n\log^2(n)\log\log(n))$ and represents the bit-complexity of the arithmetic operations (addition, multiplication, quotient, remainder and gcd) of the integers of bit-size n and the complexity of the arithmetic operations of the polynomials of degree n in terms of number of operations in the base ring. Many authors have contributed to these topics. Some very good historical presentations can be found in the books of Aho, Hopcroft, Ullman [4], Bürgisser, Clausen, Shokrolahi [14], Bini, Pan [10] among others.

Let R be a unitary commutative ring, the Schönhage-Strassen polynomial multiplication [71, 70, 63] of two polynomials of $R[T]$ of degree at most n can be performed in $\mathcal{O}(n\log(n)\log\log(n))$ arithmetic operations in R. The division of polynomials has the same complexity as the multiplication [11, 78]. The greatest common divisor of two polynomials of degree at most n over a field K can be computed in $\mathtt{M}(n)$ arithmetic operations in K [61]. The resultant, the sub-resultants and the interpolation can also be computed within the same complexity [57, 29].

The Schönhage-Strassen algorithm [71] for multiplying two integers of bit-size at most n has a bit-complexity in $\mathcal{O}(n\log(n)\log\log(n))$. The division has the same complexity as the multiplication [73]. The greatest common divisor has complexity $\mathtt{M}(n)$ [69].

Let R be a unitary ring, the multiplication of two $n \times n$ matrices can be done in $\mathcal{O}(n^\omega)$ arithmetic operations in R. The exponent ω can be taken less than 2.39 [21]. If R is a field, Bunch and Hopcroft showed that matrix inversion is not harder than the multiplication [13]. According to [13], the converse fact is due to Winograd.

In our case, R is a k-algebra $k[T]/q(T)$, where q is a square-free monic polynomial of $k[T]$, so we can not apply the results of [13] to compute the inverse of a matrix. In the whole paper $\mathcal{O}(n^\Omega)$ denotes the complexity of the elementary operations on $n \times n$ matrices over any commutative ring R in terms of arithmetic operations in R: addition, multiplication, determinant and adjoint matrix. In fact, Ω can be taken less than 4 [3, 9, 22, 56], see also [79, 59].

4 Global Newton Lifting

In this section we present the new global Newton-Hensel iterator. First, through an example, we recall the Newton-Hensel method in its local form and show the slight modification we make in order to globalize it. Then we give a formal description and proof of the method. We apply it in the case of lifting fibers in order to compute lifted curves. In the case $k = \mathbb{Q}$, we present a method to compute a geometric resolution in $\mathbb{Q}$, knowing one over $\mathbb{Z}/p\mathbb{Z}$, for a prime integer p.

4.1 Local Newton Iterator

We recall here the classical Newton iterator, along with an example. Let

$$\left\{ \begin{array}{lll} f_1(x_1, x_2, t) & = & (x_1 - 1)^2 + (x_2 - 1)^2 - 4 - t - t^2, \\ f_2(x_1, x_2, t) & = & (x_1 + 1)^2 + (x_2 + 1)^2 - 4 - t. \end{array} \right.$$

Suppose that we have solved the zero-dimensional system obtained by specializing t to 0. The variable x_1 is a primitive element and we thus have the geometric resolution

$$T^2 - 1 = 0, \; \left\{ \begin{array}{l} x_1 = T, \\ x_2 = -T. \end{array} \right. \tag{13}$$

Let $\mathbb{Q}[a]$ be the extension $\mathbb{Q}[T]/(T^2 - 1)$ of $\mathbb{Q}$. In $\mathbb{Q}[a]$ the point $\mathbf{X}_0 = (a, -a)$ is a solution of the system $f_1 = f_2 = 0$ for $t = 0$. Hence in the formal power series ring $\mathbb{Q}[a][[t]]$, it is a solution of the system at precision $\mathcal{O}(t)$. If the Jacobian matrix of f_1 and f_2 with respect to the variables x_1 and x_2 evaluated at $\mathbf{X}_0$ is invertible, the classical Newton method lifts the solution to a solution at an arbitrary precision by computing the sequence $\mathbf{X}_n$ given by

$$\mathbf{X}_{n+1} := \mathbf{X}_n - J(\mathbf{X}_n)^{-1}\mathbf{f}(\mathbf{X}_n), \qquad n \geq 0.$$

Then $\mathbf{X}_n$ is the solution of the system at the precision $\mathcal{O}(t^{2^n})$. In our example we have

$$\mathbf{X}_2 := \begin{pmatrix} a + \frac{1}{4}\, at + \left(-\frac{1}{8} + \frac{3}{32}\, a\right) t^2 - \frac{3}{128}\, at^3 + \mathcal{O}\left(t^4\right) \\ \\ -a - \frac{1}{4}\, at - \left(\frac{1}{8} + \frac{3}{32}\, a\right) t^2 + \frac{3}{128}\, at^3 + \mathcal{O}\left(t^4\right) \end{pmatrix}.$$

4.2 From Local to Global Lifting

The above method allows a local study of the positive dimensional variety in the neighborhood of $t = 0$ but does not lead to a finite representation of a solution of the input system, since the parametrization is given by infinite series over an algebraic extension of $\mathbb{Q}$. The variety $\mathcal{V}(f_1, f_2)$ has the resolution

$$T^2 - 1 - \frac{1}{2}t + \left(\frac{1}{4}T - \frac{1}{4}\right)t^2 + \frac{1}{32}t^4 = 0, \; \left\{ \begin{array}{l} x_1 = T, \\ x_2 = -T - \frac{1}{4}t^2. \end{array} \right. \tag{14}$$

We now show how we perform the lifting on this example. We lift our resolution (13) when $t = 0$ step by step to get (14).

After the first step of Newton's iterator, when $T^2 - 1 = 0$, $\mathbf{X}_1$ is $(T(1 + t/4 + \mathcal{O}(t^2)), -T(1 + t/4 + \mathcal{O}(t^2)))$. We deduce that $T = x_1(1 - t/4 + \mathcal{O}(t^2))$ and thus

$$x_1^2 - 1 - \frac{1}{2}t + \mathcal{O}(t^2) = 0 \qquad \text{and} \qquad x_2 = -x_1 + \mathcal{O}(t^2),$$

which is the approximation of (14) at precision $\mathcal{O}(t^2)$.

We repeat this technique with the new resolution

$$q(T) = T^2 - 1 - \frac{1}{2}t = 0, \; \left\{ \begin{array}{l} x_1 = T, \\ x_2 = -T. \end{array} \right.$$

We perform another step of Newton's iterator over $\mathbb{Q}[[t]][T]/q(T)$ at the point $(T,-T)$ at precision $\mathcal{O}(t^4)$. We get the following refinement of the parametrization

$$\begin{cases} x_1 = T + \left(\frac{1}{8}T - \frac{1}{8}\right)t^2 - \frac{1}{16}t^3T + \mathcal{O}(t^4) \\ x_2 = -T + \left(-\frac{1}{8}T - \frac{1}{8}\right)t^2 + \frac{1}{16}t^3T + \mathcal{O}(t^4) \end{cases}$$

Thus

$$T = x_1 + \left(\frac{1}{8} - \frac{1}{8}x_1\right)t^2 + \frac{1}{16}x_1t^3 + \mathcal{O}(t^4)$$

and we deduce:

$$T^2 - 1 - \frac{1}{2}t + \left(\frac{1}{4}T - \frac{1}{4}\right)t^2 + \mathcal{O}(t^4) = 0, \quad \begin{cases} x_1 = T, \\ x_2 = -T - \frac{1}{4}t^2 + \mathcal{O}(t^4). \end{cases}$$

Finally, the next step leads to the resolution

$$T^2 - 1 - \frac{1}{2}t + \left(\frac{1}{4}T - \frac{1}{4}\right)t^2 + \frac{1}{32}t^4 + \mathcal{O}(t^8) = 0,$$
$$\begin{cases} x_1 = T, \\ x_2 = -T - \frac{1}{4}t^2 + \mathcal{O}(t^8), \end{cases}$$

which is the desired resolution, we can remove the $\mathcal{O}(t^8)$. In general, to decide when the lifting is finished, there are two solutions: either we know the required precision in advance, this is the case in §4.5, or no *a priori* bound is known, this the case in §4.6. In the last case, the only way to decide if the resolution is correct is to check whether the lifting equations vanish on the resolution or not.

4.3 Description of the Global Newton Algorithm

Let R be a commutative integral ring, I an ideal of R. We now give a formal presentation of our lifting process passing from a resolution known at precision I to one at precision I^2.

The lifting algorithm takes as input:

(I1) $\mathbf{f} = (f_1, \ldots, f_n)$, n polynomials in $R[x_1, \ldots, x_n]$;

(I2) $u = \lambda_1 x_1 + \cdots + \lambda_n x_n$ a linear form in the x_i, with λ_i in R;

(I3) $q(T)$ a monic polynomial of degree $\delta \geq 1$ in $R[T]$;

(I4) $\mathbf{v} = (v_1(T), \ldots, v_n(T))$, n polynomials of degrees strictly less than δ in $R[T]$.

Let J be the Jacobian matrix of $f_1, \ldots, f_n$ with respect to the variables $x_1, \ldots, x_n$:

$$J_{(i,j)} = \frac{\partial f_i}{\partial x_j}.$$

In $(R/I)[T]/(q(T))$, we make the following assumptions:

(H1) $\mathbf{f}(\mathbf{v}) \equiv \mathbf{0}$;

(H2) $T \equiv u(\mathbf{v})$;

Algorithm 1: Global Newton Iterator

procedure GlobalNewton($\mathbf{f}, \mathbf{x}, u, q, \mathbf{v}$,StopCriterion)

\# $\mathbf{x}$ is the list of variables,
\# $\mathbf{f}, u, q, \mathbf{v}$ are the ones of (I1), (I2), (I3), (I4) and
\#   satisfy (H1), (H2) and (H3).
\# StopCriterion is a function returning a boolean.
\#   Its arguments are taken from the local variables $\mathbf{f}$,
\# $\mathbf{x}$, u, Q, $\mathbf{V}$ and k below.
\#   It returns whether the lifted parametrization
\# $Q(u) = 0$, $\mathbf{x} = \mathbf{V}$ at precision k is sufficient of not.

\# The procedure returns Q a polynomial and $\mathbf{V}$ as in (O1)
\# and (O2), giving a solution of $\mathbf{f}$ modulo I^κ, where
\# κ is implicitely fixed by StopCriterion.

  $J \leftarrow$ JacobianMatrix($\mathbf{f}$,$\mathbf{x}$);
  $k \leftarrow 1$; $Q \leftarrow q$; $\mathbf{V} \leftarrow \mathbf{v}$;
  while not StopCriterion($\mathbf{f}, \mathbf{x}, u, Q, \mathbf{V}, k$) **do**
    $k \leftarrow 2k$;
    $\mathbf{V} \leftarrow \mathbf{V} - J(\mathbf{V})^{-1}\mathbf{f}(\mathbf{V}) \bmod Q$;
    $\Delta \leftarrow u(\mathbf{V}) - T$;
    $\mathbf{V} \leftarrow \mathbf{V} - (\frac{\partial \mathbf{V}}{\partial T}\Delta \bmod Q)$;
    $Q \leftarrow Q - (\frac{\partial Q}{\partial T}\Delta \bmod Q)$;
  od;
  return($Q, \mathbf{V}$);
end;

(H3) $J(\mathbf{v})$ is invertible.

Then the following objects exist and we give formulæ to compute them:

(O1) Q, a monic polynomial of degree δ, such that $Q \equiv q \bmod R[T]I$;

(O2) $\mathbf{V} = (V_1, \ldots, V_n)$, n polynomials in $R[T]$ of degrees strictly less than δ such that for all i, $1 \leq i \leq n$, we have $V_i \equiv v_i \bmod R[T]I$, and verifying

$$\mathbf{f}(\mathbf{V}) \equiv 0 \text{ and } T \equiv u(\mathbf{V}) \text{ in } (R/I^2)[T]/(Q(T)).$$

The coefficients of Q and V are uniquely determined by the above conditions modulo I^2.

Proof This process is summarized in Algorithm 1, the notations being the ones of the end of §3.4. The proof divides into two parts and is just the formalization of the computations of §4.2.

First we perform a classical Newton step to compute the vector of n polynomials $\mathbf{w} = (w_1, \ldots, w_n)$, of degrees strictly less than δ in $R[T]$ such that:

(C) $\mathbf{w} \equiv \mathbf{v} \bmod R[T]I$ and $\mathbf{f}(\mathbf{w}) \equiv 0$ in $(R/I^2)[T]/(q(T))$.

We recall that this can be done by writing the first order Taylor expansion of $\mathbf{f}$ between the points $\mathbf{v}$ and $\mathbf{w}$. The condition (C) implies that:

$$\mathbf{f}(\mathbf{w}) \equiv \mathbf{f}(\mathbf{v}) + J(\mathbf{v}) \cdot (\mathbf{w} - \mathbf{v}) \text{ in } (R/I^2)[T]/(q(T)).$$

According to hypothesis (H3), we deduce the existence and uniqueness of $\mathbf{w}$ modulo $R[T]I$:

$$\mathbf{w} \equiv \mathbf{v} - J(\mathbf{v})^{-1} \cdot \mathbf{f}(\mathbf{v}) \text{ in } (R/I^2)[T]/(q(T)).$$

According to hypothesis (H2) we can write $u(\mathbf{w})$ as

$$u(\mathbf{w}) = T + \Delta(T),$$

where $\Delta(T)$ is a polynomial in $R[T]$ of degree strictly less than δ, with all its coefficients in I.

The second part is a consequence of the following equality between ideals in $R/I^2[T, U, x_1, \ldots, x_n]$:

$$\begin{aligned}(q(T), U - T - \Delta(T), x_1 - w_1(T), \ldots, x_n - w_n(T))\\ = (Q(U), T - U + \Delta(U), x_1 - V_1(U), \ldots, x_n - V_n(U)),\end{aligned}$$

where

$$\begin{aligned} Q(U) &= q(U) - (q'(U)\Delta(U) \bmod q(U)), \\ V_i(U) &= w_i(U) - (w_i'(U)\Delta(U) \bmod q(U)), \qquad i = 1, \ldots, n. \end{aligned}$$

□

We now turn to the evaluation of the complexity of Algorithm 1. Let $\mathtt{a}(h)$ be the cost of the arithmetic operations in R/I^h, where h is a positive integer. Recall that $\mathtt{M}$ is the complexity of the arithmetic operations in $R[T]$ in terms of operations in the base ring R, where R denotes here any commutative ring. Let L be the number of operations required to evaluate $f_1, \ldots, f_n$. Using the notations of §3.5, we have the following complexity estimate:

Lemma 2 *According to the above notations and assumptions, the complexity of Algorithm 1 returning a solution of $f_1, \ldots, f_n$ at precision I^κ (where κ is a power of 2) is in*

$$\mathcal{O}\big((nL + n^\Omega)\mathtt{M}(\delta) \sum_{j=0}^{\log_2(\kappa)} \mathtt{a}(2^j)\big).$$

Proof Thanks to [8], we only need at most $5L$ operations to evaluate the gradient of a straight-line program of size L. Thus the evaluation of the polynomials $\mathbf{f}$ and the Jacobian matrix J of Algorithm 1 has complexity $\mathcal{O}(nL)$. Then, the core of the loop requires $\mathcal{O}(n^\Omega)$ operations to compute the inverse of the Jacobian matrix and $\mathcal{O}(n^2)$ other operations to update Q and $\mathbf{V}$, so at step k of the loop $\mathcal{O}(nL + n^\Omega)$ arithmetic operations are done in $R/I^k[T]$ modulo Q. □

In practice there are many possible improvements. An important one consists in taking better care of the precision, for instance to compute the solution at precision $2k$, we just need to know the value of the Jacobian matrix at precision k, since the value of $f_1, \ldots, f_n$ has valuation at least k. Another one can be obtained by inverting the value of the Jacobian matrix by means of a Newton iterator: let J_k be the value of the Jabobian matrix at step k and J_k^{-1} be its inverse then we have $J_{2k}^{-1} = J_k^{-1} + J_k^{-1}(\mathrm{Id}_n - J_{2k}J_k^{-1})$. These techniques are described in [86].

4.4 Recovering a Geometric Resolution

Our iterator allows to compute a whole geometric resolution from a lifting fiber.

In the frame of §3.4, taking $R = k[y_1 - p_1, \ldots, y_r - p_r]$ and $I = (y_1 - p_1, \ldots, y_r - p_r)$ we can apply our iterator with a lifting fiber, in order to lift the parametrization using the lifting equations. But in this case by Propositions 3 and 5 we know that there exists a parametrization of the variety with total degree bounded by $\delta = \deg(\mathcal{V})$, in the form

$$q(T) = 0, \quad \begin{cases} \frac{\partial q}{\partial T} y_{r+1} & = & w_{r+1}(T), \\ & \vdots & \\ \frac{\partial q}{\partial T} y_n & = & w_n(T). \end{cases} \tag{15}$$

We can compute q and the w_i in the following way: first we apply our iterator until precision $\delta + 1$ is reached and get a resolution in the form

$$Q(T) + \mathcal{O}(I^{\delta+1}) = 0, \quad \begin{cases} y_{r+1} & = & V_{r+1}(T) + \mathcal{O}(I^{\delta+1}), \\ & \vdots & \\ y_n & = & V_n(T) + \mathcal{O}(I^{\delta+1}). \end{cases}$$

Then let

$$W_i = V_i(T)\frac{\partial Q}{\partial T} \bmod Q(T), \qquad r+1 \le i \le n,$$

the unicity of the geometric resolution lying over the lifting fiber implies that $Q - q, W_{r+1} - w_{r+1}, \ldots, W_n - w_n \in I^{\delta+1}$, whence we deduce q and the w_i.

In practice we are not interested in the lifting of a lifting fiber to its corresponding geometric resolution since it would imply storing multivariate polynomials. Indeed we do not need to lift the fiber over the whole space of the free variables but just over one line containing the lifting point.

4.5 Lifted Curves

Let F be a lifting fiber of the variety $\mathcal{V}$ as in §3.4, δ its degree and $p' \in k^r$ a point different from p. We are interested in computing the geometric resolution of $\pi^{-1}(D)$, where D denotes the line (pp').

First we notice that the variety $\mathcal{V}_D = \pi^{-1}(D)$ is 1-equidimensional of degree $\delta = \deg(\mathcal{V})$. The restriction $\pi_D : \mathcal{V}_D \to D$ is a finite surjective morphism of degree δ, smooth for $t = 0$.

Definition 5 *The variety $\mathcal{V}_D = \pi^{-1}(D)$ is called a* lifted curve *of the lifting fiber F.*

Algorithm 2: Lift Curve

procedure LiftCurve(F,p')

\# F is a lifting fiber of dimension r,
\# p' is a point in k^r different from the lifting point of F.

\# The procedure returns the Kronecker parametrization
\# $q, \mathbf{w}$ of the geometric resolution of the lifted curve
\# for the line (pp'), as in §4.5.

$r \leftarrow$ Dimension(F);
$\delta \leftarrow$ Degree(F);
$\mathbf{g} \leftarrow F_{Equations} \circ F_{ChangeOfVariables}$;
$\mathbf{h} \leftarrow \mathbf{g}((p'_1 - p_1)t + p_1, \ldots, (p'_r - p_r)t + p_r, y_{r+1}, \ldots, y_n)$;
StopCriterion $\leftarrow \big((k) \mapsto k > \delta\big)$
Q,$\mathbf{V}$:=GlobalNewton($\mathbf{h}$, $[y_{r+1}, ..., y_n]$, $F_{PrimitiveElement}$,
$F_{MinimalPolynomial}$, $F_{Parametrization}$,StopCriterion);
$\mathbf{W} \leftarrow [z\frac{\partial Q}{\partial T} \bmod Q : \ z \in \mathbf{V}]$;
$q \leftarrow$ Truncate($Q, t^{\delta+1}$);
$\mathbf{w} \leftarrow$ [Truncate($z, t^{\delta+1}$) : $z \in \mathbf{W}$];
return($q, \mathbf{w}$);
end;

Let $g_1, \ldots, g_{n-r}$ be the equations of F expressed in the Noether coordinates y_i:

$$g_j = f_j \circ M(y_1, \ldots, y_n)^t.$$

Let also $h_1, \ldots, h_{n-r}$ be the polynomials in $k[t, y_{r+1}, \ldots, y_n]$ defined by:

$$h_i = g_i((p'_1 - p_1)t + p_1, \ldots, (p'_r - p_r)t + p_r, y_{r+1}, \ldots, y_n).$$

From the lifting fiber F we deduce a lifting fiber of $\mathfrak{I}_D$ directly.

Proposition 6 *The variables $t, y_{r+1}, \ldots, y_n$ are in Noether position for $\mathcal{V}_D$, the polynomials h_i define a lifting system for $\mathcal{V}_D$, $t = 0$ is a lifting point and the primitive element of F is primitive for the fiber $t = 0$.*

We can apply the method of the previous section and get the geometric resolution of $\mathcal{V}_D$ in the form

$$q(t,T) = 0, \quad \left\{ \begin{array}{lcl} \frac{\partial q}{\partial T} y_{r+1} & = & w_{r+1}(t,T), \\ & \vdots & \\ \frac{\partial q}{\partial T} y_n & = & w_n(t,T). \end{array} \right. \tag{16}$$

This process is summarized in Algorithm 2.

In order to evaluate the complexity of this algorithm, let L be the number of operations required to evaluate $f_1, \ldots, f_n$ and let the notations be as in §3.5.

For technical reasons we have to assume that there exists a constant C such that $C\mathtt{M}(X) \geq \mathtt{M}(2X) \geq 2\mathtt{M}(X)$ for all $X > 0$ large enough; this is not really restrictive since it is verified for $\mathtt{M}(X) = X\log(X)\log\log(X)$; then we have the following complexity estimate:

Lemma 3 *Using the above notations and assumptions, the number of operations that Algorithm 2 performs on elements of R is in*

$$\mathcal{O}\big((nL + n^{\Omega})\mathtt{M}(\delta)^2\big).$$

Proof We just apply Lemma 2 to the case $\mathtt{a} = \mathtt{M}$. We have to bound the sum:

$$\sum_{j=0}^{\log_2(\kappa)} \mathtt{M}(2^j) \leq \mathtt{M}(\kappa) \sum_{j=0}^{\log_2(\kappa)} 1/2^j \in \mathcal{O}(\mathtt{M}(\kappa)).$$

The precision κ of the last step verifies $\delta < \kappa \leq 2\delta$. Hence $\mathtt{M}(\kappa) \leq \mathtt{M}(2\delta) \in \mathcal{O}(\mathtt{M}(\delta))$. □

Of course, in practice we take κ the biggest power of two less than $\delta + 1$, $\kappa \leq \delta + 1 < 2\kappa$, we lift C up to precision κ and the last step of the lifting is performed at precision $\delta + 1$ only.

4.6 Lifting the Integers

We assume here that $k = \mathbb{Q}$. The lifting of the free variables of the previous section can be used for integers as well. If we have a geometric resolution of a zero dimensional variety computed modulo a prime number p we can lift it to precision p^k. If there exists a geometric resolution with rational coefficients lying over the modular one, then the lifting process can stop and we can recover the rational numbers of the geometric resolution.

Here we take $R = \mathbb{Z}$ and $I = p\mathbb{Z}$ where p is a prime number. We assume that we have computed a geometric resolution of a zero dimensional $\mathbb{Q}$-variety in $\mathbb{Z}/p\mathbb{Z}$, that we have $f_1, \ldots, f_n$, n polynomials in $\mathbb{Q}[x_1, \ldots, x_n]$ such that their Jacobian matrix is invertible over the modular resolution, and that the degree of the modular resolution is δ, the degree of the $\mathbb{Q}$-variety. In this case there exists a unique rational geometric resolution lying over the modular one; the lifting process gives the p-adic expansions of its rational coefficients at any required precision.

In [24] Dixon gave a Padé approximant method for integers, see also [41] and [40] for related results.

Proposition 7 *[24] Let $s, h > 1$ be integers and suppose that there exist integers f, g such that*

$$gs \equiv f(\mathrm{mod}\ h) \quad \text{and} \quad |f|, |g| \leq \lambda\sqrt{h},$$

where $\lambda = 0.618\ldots$ is a root of $\lambda^2 + \lambda - 1 = 0$. Let w_i/v_i $(i = 1, 2, \ldots)$ be the convergents to the continued fraction of s/h and put $u_i = v_i s - w_i h$. If k is the least integer such that $|u_k| < \sqrt{h}$, then $f/g = u_k/v_k$.

We assume we have a function called *RationalReconstruction* computing the unique rational f/g for any s in $\mathbb{Z}/p^k\mathbb{Z}$ with bit complexity in $\mathcal{O}(\mathtt{M}(k\log(p)))$. Such a complexity can obtained combining Dixon's algorithm [24] and a fast

Algorithm 3: Lifting of Integers

procedure LiftIntegers(F)

\# F is a zero-dimensional geometric resolution over $\mathbb{Z}/p\mathbb{Z}$.

\# The procedure returns F', the geometric
\# resolution over $\mathbb{Q}$ lying over F, if it exists.

```
δ ← Degree(F);
f ← F_Equations ∘ F_ChangeOfVariables;
StopCriterion ← ((f, x, u, Q, V, k) ↦
    q ← RationalReconstruction(Q);
    w ← RationalReconstruction([z ∂q/∂T mod q : z ∈ V]);
    if f(w/(∂q/∂T)) mod q = 0 then
        Q← q;V← w;
        return true;
    else return false;
    fi; )
q, w← GlobalNewton(f, x, F_PrimitiveElement,
    F_MinimalPolynomial, F_Parametrization,StopCriterion);
F' ← F;
F'_MinimalPolynomial ← q;
F'_Parametrization ← w;
return(F');
end;
```

Gcd algorithm for integers as discussed in §3.5, see [10, p.247]. This function returns an error if no such rational number exists. Thus we can stop the lifting when the rational reconstruction of each coefficient of the current resolution leads to a parametrization over $\mathbb{Q}$ of $\mathcal{V}$ satisfying all the equations f_i. This process is summarized in Algorithm 3.

Lemma 4 *Assume that the geometric resolution lying over the modular one has height at most η with $\log(p) \leq \eta$, then it can be computed in bit complexity*

$$\mathcal{O}((nL + n^{\Omega})\mathtt{M}(\delta)\mathtt{M}(\eta)).$$

Proof We apply Lemma 2 with $\mathtt{a}(k)$ being the bit complexity of the arithmetic in $\mathbb{Z}/p^k\mathbb{Z}$: we can take $\mathtt{a}(k) = \mathtt{M}(k\log(p))$. Choose κ a power of two, such that $4\eta \geq \kappa\log(p) > 2\eta$ and apply Algorithm 1 until precision $k = \kappa$: since $\sum_{j=0}^{\log_2(\kappa)} \mathtt{M}(\log(p)2^j) \in \mathtt{M}(\kappa\log_2(p))$, then the complexity is in $\mathcal{O}((nL + n^{\Omega})\mathtt{M}(\delta)\mathtt{M}(\eta))$. The rational reconstruction for each coefficient of the Kronecker parametrization is in $\mathcal{O}(n\delta\mathtt{M}(\eta))$ □

Theorem 2 is a direct corollary of this lemma.

This result does not give the complexity of Algorithm 3 because it forgets the verification that the rational reconstructed parametrization satisfies the equations. This verification could be done in $\mathbb{Q}[T]/q(T)$ but it would involve a growth of the size of the integers in the intermediate computations. So, in practice, we prefer to choose another prime number $p' \neq p$ and we perform this verification in $\mathbb{Z}/p'\mathbb{Z}$. The study of the probability of success of this method is out of the scope of this work (see [41] and [40] for results related to this question).

5 Changing a Lifting Fiber

From any given lifting fiber one can change it to another one, more precisely we can make any linear change of the free variables, or compute a lifting fiber for another lifting point or for another primitive element. These three operations on lifting fibers are crucial for the algorithm since it may appear that a given lifting fiber may not be generic enough for computing the intersection of its corresponding variety by a given hypersurface. In this section we assume we are given a lifting fiber with the same notations as in §3.4.

5.1 Changing the Free Variables

Let $\mathcal{V}$ be a r-equidimensional variety given by a lifting fiber and f be a given polynomial in $k[x_1, \ldots, x_n]$. We are interested in having a Noether position of $\mathcal{V} \cap \mathcal{V}(f)$.

Lemma 5 *Let $\mathcal{V}$ be a r-equidimensional variety of degree δ such that the variables $x_1, \ldots, x_n$ are in Noether position, and f a polynomial in $k[x_1, \ldots, x_n]$ of total degree d such that $\mathcal{V}(f)$ intersects $\mathcal{V}$ regularly. For almost all choices of $(p_1, \ldots, p_{r-1}) \in k^{r-1}$ the change of variables $x_1 = y_1 + p_1 y_r, \ldots, x_{r-1} = y_{r-1} + p_{r-1} y_r, x_r = y_r, \ldots, x_n = y_n$ brings the new coordinates y_i into a Noether position with respect to $\mathcal{V} \cap \mathcal{V}(f)$.*

Proof Let $\mathfrak{I} = \mathfrak{I}(\mathcal{V})$, x_0 be a new variable, the exponent ^{h} is related to the homogenized objects as in §3.1. The ideal $\mathfrak{I}^h$ is in Noether position, let F be an integral dependence relation for f^h given by Proposition 2. Its total degree is bounded by δd and $F(f^h)$ belongs to $\mathfrak{I}^h$. Let $A \in k[x_0, \ldots, x_r]$ be the constant coefficient of F, it belongs to $\mathfrak{I}^h + (f^h)$. Since f intersects $\mathcal{V}$ regularly, F can be chosen such that $A \neq 0$. Let m be the valuation of A with respect to x_0, we define B by A/x_0^m, B is in $(\mathfrak{I}^h + (f^h)) : x_0^\infty$, which is the homogenized ideal of $\mathfrak{I} + (f)$. Let B_0 be the constant coefficient of B with respect to x_0, it is homogeneous and not zero, we can choose a point $p = (p_1, \ldots, p_{r-1})$ in k^{r-1} such that $B_0(p_1, \ldots, p_{r-1}, 1)$ is not zero. Then the change of variables $x_1 = y_1 + p_1 y_r, \ldots, x_{r-1} = y_{r-1} + p_{r-1} y_r, x_r = y_r, \ldots, x_n = y_n$, is such that the new variable y_r is integral over $x_0, y_1, \ldots, y_{r-1}$ for $\mathcal{V} \cap \mathcal{V}(f)$. We deduce that the variables $x_0, y_1 \ldots, y_n$ are in Noether position with respect to $\mathcal{V} \cap \mathcal{V}(f)$. $\square$

The operations to perform such a change of variables are described in Algorithm 4. Its complexity is in $\mathcal{O}(n^\Omega)$, the complexity of performing linear algebra in dimension n, it is not significative in the whole algorithm.

5.2 Changing the Lifting Point

We are now interested in computing a lifting fiber F' on another given lifting point p', assuming that the primitive element of F remains primitive for F'.

Algorithm 4: Change Free Variables

procedure ChangeFreeVariables(F, p)

\# F is a Lifting Fiber of dimension r
\# p is a point in k^{r-1}

\# The procedure performs the linear change of the free variables of
\# F: $y_1 \leftarrow y_1 + p_1 y_r, \ldots, y_{r-1} \leftarrow y_{r-1} + p_{r-1} y_r$.

$r \leftarrow$ Dimension(F);
$N \leftarrow \mathrm{Id}_n \in$ SquareMatrix(n);
for i **from** 1 **to** $r-1$ **do**$N[i,r] \leftarrow p_i$; **od**;
$F_{ChangeOfVariables} \leftarrow F_{ChangeOfVariables} \circ N$;
$N \leftarrow$ SubMatrix($N, 1..r, 1..r$);
$F_{LiftingPoint} \leftarrow N^{-1} F_{LiftingPoint}$;
end;

We use the method of §4.5 to compute the geometric resolution of the lifted curve corresponding to the line (pp') in the form of Equation (16). The specialization of this parametrization for $t = 1$ is the one of F'. The method is summarized in Algorithm 5. Its complexity is the same as in Lemma 3.

Algorithm 5: Change Lifting Point

procedure ChangeLiftingPoint(F,p')

\# F is a lifting fiber of dimension r,
\# $p' \in k^r$ is a new lifting point, such that
\# $F_{PrimitiveElement}$ remains primitive over p'.

\# At the end F contains the lifting fiber for p'.

$q, \mathbf{w} \leftarrow$ LiftCurve(F, p');
$q, \mathbf{w} \leftarrow$ subs($t = 1, q, \mathbf{w}$);
$\mathbf{v} \leftarrow [z/q' \bmod q \ : \ z \in \mathbf{w}]$;
$F_{MinimalPolynomial} \leftarrow q$;
$F_{Parametrization} \leftarrow \mathbf{v}$;
$F_{LiftingPoint} \leftarrow p'$;
end;

Algorithm 6: Change Primitive Element

procedure ChangePrimitiveElement(F,u')

\# F is a lifting fiber of dimension r,
\# u' is a lucky new primitive element.

\# At the end F contains the lifting fiber for u'.

$q \leftarrow F_{MinimalPolynomial}$;
$\mathbf{v} \leftarrow F_{Parametrization}$;
$u \leftarrow F_{PrimitiveElement}$;
\# Let t be a new variable the computations are in $k[t]/(t^2)$.
$u'_t \leftarrow u' + tu$;
$U'_t \leftarrow \text{Resultant}_T(q, S - u'_t(\mathbf{v}))$;
$Q \leftarrow \text{subs}(t = 0, U'_t)$;
$V \leftarrow -\text{Coefficient}(U'_t, t)/Q' \bmod Q$;
$\mathbf{V} \leftarrow [z(V) \bmod Q \ : \ z \in \mathbf{v}]$;
$F_{MinimalPolynomial} \leftarrow Q$;
$F_{Parametrization} \leftarrow \mathbf{V}$;
$F_{PrimitiveElement} \leftarrow u'$;
end;

5.3 Changing the Primitive Element

We show how we compute a lifting fiber F' for another given primitive element $u' = \lambda'_{r+1} y_{r+1} + \cdots + \lambda'_n y_n$. The method is summarized in Algorithm 6.

Let t be a new variable, we extend the base field k to the rational function field $k_t = k(t)$. Let $u'_t = u' + tu$ and $\mathfrak{I}_t$ the extension of $\mathfrak{I}$ in k_t. We can compute the characteristic polynomial U'_t of u'_t such that $U'_t(u'_t) \in \mathfrak{I}_t$ and deduce the Kronecker parametrization of $\mathfrak{I}_t$ with respect to u_t in the same way as in §3.3. The characteristic polynomial can be computed by means of a resultant:

$$U'_t(S) = \text{Resultant}_T(q(T), S - u'_t(v_{r+1}, \ldots, v_n)).$$

In order to get the new parametrization, we only need to know the first order partial derivative with respect to t at the point $t = 0$. So the resultant can be computed modulo t^2. If we use a resultant algorithm performing no division on its base ring, this specialization over the non-integral ring $k[t]/(t^2)[S]$ does not create any problem.

A problem comes from the fact that we are interested in using resultant algorithms for integral rings since they have better complexity. In order to explain how this can work under some genericity conditions, we come back to the notations of §3.3. Then we take u' generic: $u_\Lambda = \Lambda_{r+1} x_{r+1} + \cdots + \Lambda_n x_n$, the Λ_i being new variables, we can compute

$$U_\Lambda(S) = \text{Resultant}_T(q(T), S - u_\Lambda(v_{r+1}, \ldots, v_n)),$$

in the integral ring $k[\Lambda_{r+1},\ldots,\Lambda_n][S]$. Let Φ be the ring morphism of specialization:

$$\begin{array}{rrcl} \Phi: & k[\Lambda_{r+1},\ldots,\Lambda_n][S] & \to & k[t]/(t^2)[S], \\ & \Lambda_i & \mapsto & \lambda'_i + t\lambda_i \end{array}$$

If u' is chosen generic enough the specialization Φ commutes with the resultant computation. The justification of this fact is based on the remark that the specialization commutes when all the equality tests on elements of $k[t]/(t^2)$ can be done on the coefficients of valuation 0 and give the same answer as the corresponding test in $k[\Lambda_{r+1},\ldots,\Lambda_n]$. The λ'_i for which this condition does not apply satisfy algebraic equations in $k[\Lambda_{r+1},\ldots,\Lambda_n]$. A choice of u' such that the specialization Φ commutes with a given resultant algorithm is said to be *lucky* for this computation. One can find in [31, §7.4] a systematic discussion about this question.

In order to estimate the complexity of this method, recall that $\mathtt{M}(\delta)$ is the complexity of the resultant of two univariate polynomials of degrees at most δ in terms of arithmetic operations in the base ring and also the complexity of the arithmetic operations on univariate polynomials of degree δ, as in §3.5.

Lemma 6 *Let u' be a lucky primitive element for Algorithm 6, then the complexity of Algorithm 6 is in $\mathcal{O}(n\delta\mathtt{M}(\delta))$.*

Proof In the resultant computation of U'_t the variable S if free thus its specialization commutes with the resultant. The degree of U'_t in S is δ. So we can compute U'_t for $\delta+1$ distinct values of S and interpolate in k the polynomials q' and v'. The cost of interpolation in degree δ is in $\mathtt{M}(\delta)$ [10, p. 25].

Then the computation of $\mathbf{v}'$ requires to compute the powers $v^2,\ldots,v^{\delta-1}$ modulo q', this involves a cost in $\mathcal{O}(\delta\mathtt{M}(\delta))$. Finally we perform n linear combinations of these powers, which takes $\mathcal{O}(n\delta^2)$ operations. □

6 Computation of an Intersection

We show in this section how we compute a lifting fiber of the intersection by a hypersurface of a r-equidimensional variety given a lifting fiber. We use Kronecker's method: when performing an elimination, the parametrization of the coordinates are given at the same time as the eliminating polynomial. The computational trick consists in a slight change of variables called Liouville's substitution [58, p.15] and the use of first order Taylor expansions.

Example 6 Suppose we want a geometric resolution of two equations f_1 and f_2, intersecting regularly, in $k[x_1,x_2]$. Let Λ_1 and Λ_2 be new variables and $u_\Lambda = \Lambda_1x_1+\Lambda_2x_2$. We can compute $U_\Lambda(T)$, the eliminating polynomial of u_Λ:

$$U_\Lambda(T) = \mathrm{Resultant}_{x_1}\Big(f_1(x_1,\frac{T-\Lambda_1x_1}{\Lambda_2}), f_2(x_1,\frac{T-\Lambda_1x_1}{\Lambda_2})\Big).$$

The expression $U_\Lambda(u_\Lambda)$ belongs to the ideal (f_1,f_2), and f_1, f_2 have a common root if and only if $U_\Lambda(u_\Lambda)$ vanishes. Taking the first derivatives in the Λ_i we deduce that

$$\frac{\partial U_\Lambda}{\partial T} x_1 + \frac{\partial U_\Lambda}{\partial \Lambda_1} \in (f_1, f_2),$$

and

$$\frac{\partial U_\Lambda}{\partial T} x_2 + \frac{\partial U_\Lambda}{\partial \Lambda_2} \in (f_1, f_2).$$

If U_Λ is square free, then the common zeros of f_1 and f_2 are parameterized by

$$U_\Lambda(T) = 0, \quad \begin{cases} \frac{\partial U_\Lambda}{\partial T}(T) x_1 &= -\frac{\partial U_\Lambda}{\partial \Lambda_1}(T), \\ \frac{\partial U_\Lambda}{\partial T}(T) x_2 &= -\frac{\partial U_\Lambda}{\partial \Lambda_2}(T). \end{cases} \tag{17}$$

For almost all values λ_1, λ_2 in k of Λ_1, Λ_2, the specialization of (17) gives a geometric resolution of f_1, f_2. So, letting $\Lambda_i = \lambda_i + t_i$, in order to get a geometric resolution we only need to know U_Λ at precision $\mathcal{O}((t_1, t_2)^2)$.

Our aim is to generalize the method of this example for the intersection of a lifted curve with an hypersurface.

Let $\mathfrak{I}$ be a 1-equidimensional radical ideal in $k[y, x_1, \ldots, x_n]$ such that the variables $y, x_1, \ldots, x_n$ are in Noether position and assume that we have a geometric resolution in the form

$$q(y, T) = 0, \quad \begin{cases} \frac{\partial q}{\partial T}(y, T) x_1 &= w_1(y, T), \\ &\vdots \\ \frac{\partial q}{\partial T}(y, T) x_n &= w_n(y, T). \end{cases} \tag{18}$$

The variable T represents the primitive element u. Let f be a given polynomial in $k[y, x_1, \ldots, x_n]$ intersecting $\mathfrak{I}$ regularly, which means that $\mathfrak{I} + (f)$ is 0-dimensional. We want to compute a geometric resolution of $\mathfrak{I} + (f)$.

6.1 Characteristic Polynomials

In the situation above one can easily compute an eliminating polynomial in the variable y, using any elimination process. First we invert q' modulo q and compute $v_i(y, T) = w_i(y, T) q'^{-1}(y, T) \bmod q(y, T)$, for $1 \leq i \leq n$. The elimination process we use is given in the following:

Proposition 8 *The characteristic polynomial of the endomorphism of multiplication by f in $B' = k(y)[x_1, \ldots, x_n]/\mathfrak{I}$ belongs to $k[y][T]$ and its constant coefficient with respect to T is given by*

$$A(y) = \mathrm{Resultant}_T(q, f(y, v_1, \ldots, v_n)),$$

up to its sign. Moreover the set of roots of $A(y)$ is exactly the set of values of the projection on the coordinate y of the set of roots of $\mathfrak{I} + (f)$.

Proof We already know from Corollary 2 that A belongs to $k[y]$ and has degree bounded by $\deg(f)\delta$, $\delta = \deg(\mathcal{V})$. Let π be the finite projection onto the coordinate y. Let y_0 be a point of $\overline{k}$ and $\{Z_1, \ldots, Z_s\} = \pi^{-1}(y_0)$ of respective multiplicity $m_1, \ldots, m_s$, $s \leq \delta$ and $m_1 + \cdots + m_s = \delta$, where the multiplicity of Z_i is defined as $m_i = \dim_k(k[y, x_1, \ldots, x_n]/(\mathfrak{I} + (y - y_0))_{Z_i})$. First we prove that

$$A(y) \in \mathfrak{I} + (f), \tag{19}$$

which implies that any root of $\mathfrak{I} + (f)$ cancels A, and then the formula

$$A(y_0) = \prod_{j=1}^{s} f(Z_j)^{m_j}, \tag{20}$$

which implies that when y_0 annihilates A at least one point in the fiber annihilates f.

The ideal $\mathfrak{I}$ being 1-equidimensional and the variables being in Noether position, the finite $k[y]$-module $B = k[y, x_1, \ldots, x_n]/\mathfrak{I}$ is free of rank δ (combine [53, Example 2, p.187] and the proof of [38, Lemma 3.3.1] or [7, Lemma 5]). Since any basis of B induces a basis for $B' = k(y) \otimes B$, the characteristic polynomials of the endomorphism of multiplication by f in B and B' coincide, Cayley-Hamilton theorem applied in B implies (19).

For the formula (20), let $B_0 = \overline{k}[y, x_1, \ldots, x_n]/(\mathfrak{I} + (y - y_0))$, B_0 is a $\overline{k}$-vector space of dimension δ. Let $e_1, \ldots, e_\delta$ be a basis of B, their specialization for $y = y_0$ leads to a set of generators of B_0 of size δ thus it is a basis of B_0. We deduce that $A(y_0)$ is the constant coefficient of the characteristic polynomial of the endomorphism of multiplication by f in B_0, whence formula (20). □

From a computational point of view, the variable y belongs to $k(y)$ and if we take $p \in k$ such that the denominators of the v_i do not vanish at p we can perform the computation of the resultant in $k[[y - p]]/((y - p)^{\delta d + 1})$, since A has degree at most δd. This method works well if we use a resultant algorithm performing no test and no division. So we are in the same situation as in §5.3, we want to use an algorithm with tests and divisions in order to get a better complexity, and this is possible if p is generic enough. The values of p for which this computation gives the good result are said to be *lucky*. Unlucky p are contained in a strict algebraic closed subset of k. In Algorithm 7 we suppose that the last coordinate of the lifting point is lucky.

As in §3.5, $\mathtt{M}$ denotes respectively the complexity of univariate polynomial arithmetic and the resultant computation.

Lemma 7 *Let L be the complexity of evaluation of f, d the total degree of f and δ the degree of q, then $A(y)$ can be computed in $\mathcal{O}\big((L+n^2)\mathtt{M}(\delta)\mathtt{M}(d\delta)\big)$ arithmetic operations in k.*

Proof Let $p \in k$ be generic enough, we perform the computation with y in $k[[y - p]]$ at precision $\mathcal{O}((y - p)^{d\delta + 1})$. First we have to compute each v_i from the w_i, this is done by performing an extended GCD between q and $\frac{\partial q}{\partial T}$. The cost of the extended GCD is the same as $\mathtt{M}$. Then we evaluate f modulo q, and perform the resultant computation, whence the complexity. □

6.2 Liouville's Substitution

We are now facing two questions: first the variable y is probably not a primitive element of $\sqrt{\mathfrak{I}+(f)}$, so we are looking for an eliminating polynomial of $\lambda y+u$ and secondly we want the parametrization of the coordinates with respect to the linear form $\lambda y+u$, for the same cost. Liouville's substitution answers both problems, for almost all $\lambda \in k$.

Let Y be a new variable, the substitution consists in replacing y by $(Y-T)/\lambda$ in both the parametrization and the polynomials of the ideal $\mathfrak{I}$. So we need some more notations: let $q_Y(Y,T)=q((Y-T)/\lambda,T)$, $p_Y(Y,T)=q'((Y-T)/\lambda,T)$, $w_{Y,i}(Y,T)=w_i((Y-T)/\lambda,T)$, $1\leq i\leq n$ and $\mathfrak{I}_Y=(e((Y-T)/\lambda,T),\ e\in\mathfrak{I})$. In order to apply Proposition 8, we must ensure that the parametrization of $\mathfrak{I}_Y$ we get is still valid. Indeed, this is true for almost all $\lambda\in k$.

Definition 6 *A point λ is said to be a* Liouville point *with respect to the above geometric resolution of $\mathfrak{I}$ when it is not zero, and when q_Y is monic in T, of the same degree as q in T, square-free and relatively prime with p_Y.*

Lemma 8 *With the above notations, if λ is a Liouville point then the variables $Y,x_1,\ldots,x_n$ are in Noether position with respect to $\mathfrak{I}_Y$ and*

$$q_Y(Y,T)=0,\qquad \left\{\begin{array}{rcl} p_Y(Y,T)x_1 & = & w_{Y,1}(Y,T),\\ & \vdots & \\ p_Y(Y,T)x_n & = & w_{Y,n}(Y,T),\end{array}\right. \tag{21}$$

is a geometric resolution of $\mathfrak{I}_Y$ for the primitive element u.

Proof First we prove that Y is free in $\mathfrak{I}_Y$. Let $h\in k[Y]$ such that $h(Y)\in\mathfrak{I}_Y$. This implies that $q(y,T)$ divides $h(\lambda y+T)$ and so q_Y divides $h(Y)$. Since q_Y is monic in T this implies that $h=0$.

Now we prove that the x_i are dependent over Y. Let $\mathfrak{J}=\mathfrak{I}_Y+(T-u)\subset k[Y,x_1,\ldots,x_n,T]$ and h a bivariate polynomial such that $h(y,x_i)\in\mathfrak{I}$ is monic in x_i, of total degree bounded by $\deg_{x_i}(h)$. We have $h((Y-T)/\lambda,x_i)\in\mathfrak{J}$, and since $q_Y(Y,T)\in\mathfrak{J}$ has a total degree bounded by δ, there exists a polynomial H such that $H(Y,x_i)\in\mathfrak{J}$, monic in x_i, of total degree bounded by its partial degree in x_i. We deduce that $Y,x_1,\ldots,x_n$ are in Noether position with respect to $\mathfrak{I}_Y$.

The conditions that q_Y is square-free and relatively prime with p_Y imply that u remains a primitive element and that we can invert p_Y modulo q_Y. The parametrization of (21) is a geometric resolution of $\mathfrak{I}_Y$. $\square$

Lemma 9 *Almost all elements $\lambda\in k$ are Liouville points.*

Proof We write the proof replacing λ by $1/\lambda$ and then Y by λY, thus q_Y becomes $q(Y-\lambda T,T)$. The discriminant of q_Y and the resultant of q_Y with p_Y are now polynomials in λ and Y and do not vanish for $\lambda=0$. Hence almost all choices of λ satisfy the last two conditions of Definition 6. For the first one, let us consider $h(y,T)$, the homogeneous part of q of maximal degree δ, then the coefficient of T^δ in q_Y is $h(-\lambda,1)$, which does not vanish when $\lambda=0$. $\square$

Lemma 10 *Let $\lambda\in k\backslash\{0\}$ and p be a polynomial in $k[y,T]$ of total degree bounded by δ and stored in a two dimensional array of size $\mathcal{O}(\delta^2)$. The polynomial $p_Y(Y,T)=p((Y-T)/\lambda,T)\in k[Y,T]$, can be computed in $\mathcal{O}(\delta\mathtt{M}(\delta))$ arithmetic operations in k.*

Proof We can write $p = p_0 + p_1 + \cdots + p_\delta$, where each p_i is homogeneous of degree i. So we can suppose that p is homogeneous of degree i. To compute $p_Y(Y,T) = p((Y-T)/\lambda, T)$ we first note that since p_Y is homogeneous of degree i we just compute $p_Y(Y,1) = p((Y-1)/\lambda, 1)$. But $p(y,1)$ is a polynomial in $k[y]$ in which we have to perform a linear transformation. We refer to [10, pp. 15–16]: the cost of the linear substitution is $\mathtt{M}(i)$. Thus the sum of complexities for each i is in $\mathcal{O}\big(\sum_{i=0}^{\delta} \mathtt{M}(i)\big) \subset \mathcal{O}(\delta \mathtt{M}(\delta))$. □

6.3 Computing the Parametrization

Combining the two previous sections, we are now able to describe the core of our intersection method, which is summarized in Algorithm 7.

Algorithm 7: Kronecker Intersection Algorithm

procedure KroneckerIntersect(C, λ, f)

\# C is a geometric resolution of $\mathfrak{I}$, 1-equidimensional,
\# with a first order generic parametrization.
\# λ is a Liouville point for C.
\# f is a polynomial.

\# The procedure returns the constant coefficient
\# of the characteristic polynomial of the endomorphism
\# of multiplication by f in $k[y, x_1, \ldots, x_n]/\mathfrak{I}$.

$q \leftarrow C_{MinimalPolynomial}$;
$\mathbf{w} \leftarrow C_{Parametrization}$;
$q_Y \leftarrow q((Y-T)/\lambda, T)$;
$p_Y \leftarrow \frac{\partial q}{\partial T}((Y-T)/\lambda, T)$;
$\mathbf{w}_Y \leftarrow \mathbf{w}((Y-T)/\lambda, T)$;
$\mathbf{v}_Y \leftarrow \mathbf{w}_Y p_Y^{-1} \bmod q_Y$;
$A \leftarrow \mathrm{Resultant}_T(q_Y, f((Y-T)/\lambda, \mathbf{v}_Y))$;
return(A);
end;

Proposition 8 and Lemma 8 lead to:

Proposition 9 *If λ is a Liouville point for the given geometric resolution of $\mathfrak{I}$, then the polynomial A returned by Algorithm 7 applied on $\mathfrak{I}_Y$ and f_Y satisfies*

$$A(\lambda y + u) \in \mathfrak{I}$$

and its set of roots is exactly the set of values of the linear form $\lambda y + u$ on the points of $\mathfrak{I} + (f)$.

Proof From Proposition 8 we have $A(Y) \in \mathfrak{I}_Y + (f_Y)$ and over each root of A lies a zero of $\mathfrak{I}_Y + (f_Y)$. Replacing Y by $\lambda y + u$ leads to $A(\lambda y + u) \in \mathfrak{I}$ and a zero $(z_Y, z_1, \dots, z_n)$ of $\mathfrak{I}_Y$ lying over z_Y, a root of A, induces a zero of $\mathfrak{I}$, namely $((z_Y - u(z_1, \dots, z_n))/\lambda, z_1, \dots, z_n)$. □

This is not sufficient to describe the points of $\mathfrak{I} + (f)$: the parametrization of the coordinates are still missing. Let $t_y, t_1, \dots, t_n$ be new variables and $k_t = k(t_y, t_1, \dots, t_n)$, let $\mathfrak{I}_t$ be the extension of $\mathfrak{I}$ in k_t and $u_t = u + t_1 x_1 + \cdots + t_n x_n$, we assume that we have the geometric resolution of $\mathfrak{I}_t$ with respect to u_t:

$$q_t(y, T) = 0, \quad \left\{ \begin{array}{lcl} x_1 & = & v_{t,1}(y, T), \\ & \vdots & \\ x_n & = & v_{t,n}(y, T). \end{array} \right. \tag{22}$$

If λ is a Liouville point for $\mathfrak{I}$ then $\lambda + t_y$ is a Liouville point for $\mathfrak{I}_t$. So we can apply Algorithm 7 in this situation, we get a polynomial $A_t \in k_t[T]$ such that $A_t((\lambda + t_y)y + u) \in \mathfrak{I}_t$ and we can write

$$A_t = A + t_y A_y + t_1 A_1 + \cdots + t_n A_n + \mathcal{O}((t_y, t_1, \dots, t_n)^2),$$

where A, A_y and the A_i are polynomials over k. We deduce that

$$A(\lambda y + u), \quad A'(\lambda y + u)y + A_y(\lambda y + u), \quad A'(\lambda y + u)x_i + A_i(\lambda y + u),$$

$1 \le i \le n$, belong to $\mathfrak{I}$. The computation has to be handled only at precision $\mathcal{O}((t_y, t_1, \dots, t_n)^2)$, so we are faced with the same problem as in §5.3: if we use a resultant algorithm without division there is no difficulty, but if we want to benefit from the better complexity of an algorithm for an integral ring we have to make some genericity restriction on the choices of u and λ. We will also speak about *lucky* choices for Algorithm 7. We call the parametrization (22) at precision $\mathcal{O}((t_y, t_1, \dots, t_n)^2)$ the *first order generic parametrization* associated to parametrization (18).

Lemma 11 *With lucky u and λ, Algorithm 7 has complexity in*

$$\mathcal{O}\big(n(L + n^2)\mathtt{M}(\delta)\mathtt{M}(d\delta)\big),$$

in terms of number of arithmetic operations in k.

Proof This is a direct consequence of Lemma 7 replacing k by $k[t_y, t_1, \dots, t_n]/(t_y, t_1, \dots, t_n)^2$. The n Liouville's substitutions are insignificant. □

In §6.5 we explain how to deduce a geometric resolution from A, A_y and the A_i.

6.4 Lifting a First Order Genericity

Now, we have to explain how to compute the first order generic parametrization (22) from (18), that we use in the previous section.

The ideal $\mathfrak{I}$ is given by the geometric resolution of equations (18). Let $B_t = k_t \otimes B$, in B_t we have $x_i = v_i(y, u)$ so $u_t = u + t_1 v_1(y, u) + \cdots + t_n v_n(y, u)$. But at the first order in the t_i we have $u_t = u + t_1 v_1(y, u_t) + \cdots + t_n v_n(y, u_t) + \mathcal{O}((t_y, t_1, \dots, t_n)^2)$, we deduce that $u = u_t - (t_1 v_1(y, u_t) + \cdots + t_n v_n(y, u_t)) +$

$\mathcal{O}((t_y, t_1, \ldots, t_n)^2)$. We can replace u in the parametrization:

$$\begin{aligned} q_t(y,T) &= q(y,T) - \big(\frac{\partial q}{\partial T}(y,T)(t_1 v_1(y,T) + \cdots + t_n v_n(y,T)) \bmod q(y,T)\big) \\ &= q(y,T) - \big(t_1 w_1(y,T) + \cdots + t_n w_n(y,T)\big) + \mathcal{O}((t_y, t_1, \ldots, t_n)^2) \end{aligned}$$

$$\begin{aligned} v_{t,i}(y,T) &= v_i(y,T) - \frac{\partial v_i}{\partial T}(y,T)(t_1 v_1(y,T) + \cdots + t_n v_n(y,T)) \\ &\quad \bmod q_t(y,T) + \mathcal{O}((t_y, t_1, \ldots, t_n)^2), \qquad 1 \le i \le n. \end{aligned}$$

Computations are summarized in Algorithm 8. As in the previous subsection we perform the computations in

$$k_{y,t} = k[y, t_y, t_1, \ldots, t_n]/((y-p)^{d\delta+1} + (t_y, t_1, \ldots, t_n)^2),$$

with a lucky choice of p in order to inverse $\frac{\partial q}{\partial T}$ modulo q with an extended GCD algorithm of complexity $\mathtt{M}(\delta)$. In this situation we have the following complexity estimate:

Lemma 12 *Algorithm 8 has complexity in $\mathcal{O}\big(n^2\mathtt{M}(\delta)\mathtt{M}(d\delta)\big)$, in terms of number of arithmetic operations in k.*

Proof The arithmetic operations in $k_y = k[y]/(y-p)^{d\delta+1}$ have cost in $\mathcal{O}(\mathtt{M}(d\delta))$ in terms of arithmetic operations in k. The computation of $\mathbf{v}$ requires $\mathcal{O}(n\mathtt{M}(\delta))$ in k_y. Then the computation of $\mathbf{v}_t$ requires $\mathcal{O}(n)$ operations in $k_{t,y}/(q_t)$, this is in $\mathcal{O}(n^2\mathtt{M}(d\delta)\mathtt{M}(\delta))$. □

6.5 Removing the Multiplicities

The output of Algorithm 7 is not yet a parametrization of the roots of $\mathfrak{I} + (f)$: it may happen that A_0 has multiplicities. We give a simple method to remove them and thus get a geometric resolution of $\sqrt{\mathfrak{I} + (f)}$.

Assume now that $\mathfrak{I}$ is 0-dimensional, that we have a primitive element $u = \lambda_1 x_1 + \cdots + \lambda_n x_n$ of $\mathcal{V} = \mathcal{V}(\mathfrak{I})$ and that at precision $\mathcal{O}((t_1, \ldots, t_n)^2)$ we have an eliminating polynomial A_t of $u_t = u + t_1 x_1 + \cdots + t_n x_n$, coming from Algorithm 7, such that

$$A_t(u_t) \in \mathfrak{I}_t + \mathcal{O}((t_1, \ldots, t_n)^2),$$

and the roots of A_t are the values of u_t over the points of $\mathcal{V}$. Let $Z_1, \ldots, Z_\delta$ be the points of $\mathcal{V}$, then for some integers $m_i > 0$ we have

$$A_t(T) = \prod_{j=1}^{\delta} (T - u_t(Z_j))^{m_j} + \mathcal{O}((t_1, \ldots, t_n)^2).$$

Now if we write $A_t = A_0 + t_1 A_1 + \cdots + t_n A_n + \mathcal{O}((t_1, \ldots, t_n)^2)$, with A_i polynomials in $k[T]$ we have:

$$A_0(T) = \prod_{j=1}^{\delta} (T - u(Z_j))^{m_j},$$

$$A_i(T) = -\sum_{i=1}^{\delta} \Big(x_i(Z_j) m_i (T - u(Z_i))^{m_i - 1} \prod_{j=1, j\neq i}^{\delta} (T - u(Z_j))^{m_j} \Big), \quad 1 \le i \le n.$$

We deduce the following proposition:

Algorithm 8: Lift First Order Genericity

procedure LiftFirstOrderGenericity(C)

\# C is a geometric resolution of $\mathfrak{I}$, one equidimensional.

\# The procedure returns a geometric resolution C' of $\mathfrak{I}_t$
\# for the primitive element $u_t = u + t_1x_1 + \cdots t_nx_n$,
\# at precision $\mathcal{O}((t_1, \ldots, t_n)^2)$.

$C' \leftarrow C$;
$q \leftarrow C_{MinimalPolynomial}$;
$\mathbf{w} \leftarrow C_{Parametrization}$;
$q_t \leftarrow q - (t_1w_1 + \cdots + t_nw_n)$;
$\mathbf{v} \leftarrow (\frac{\partial q}{\partial T})^{-1}\mathbf{w} \bmod q$;
$\mathbf{v_t} \leftarrow \mathbf{v} - \frac{\partial \mathbf{v}}{\partial T}(t_1v_1 + \cdots + t_nv_n) \bmod q_t$;
${C'}_{MinimalPolynomial} \leftarrow q_t$;
${C'}_{Parametrization} \leftarrow \mathbf{v_t}$;
${C'}_{PrimitiveElement} \leftarrow C_{PrimitiveElement} + t_1x_1 + \cdots + t_nx_n$;
return(C');
end;

Proposition 10 *With the above notations, let* $M = \gcd(A_0, A_0')$ *then* M *divides* A_0, A_0' *and the* A_i. *Let* $q = A_0/M$, $p = A_0'/M$ *and* $w_i = -A_i/M$, $1 \le i \le n$, *then*

$$q(u) = 0, \quad \left\{ \begin{array}{lcl} p(u)x_1 & = & w_1(u), \\ & \vdots & \\ p(u)x_n & = & w_n(u), \end{array} \right. \tag{23}$$

is a geometric resolution of $\mathcal{V}$.

This process is summarized in Algorithm 9.

Lemma 13 *Let* δ *be the degree of* A_t *in* T *then the complexity of Algorithm 9 is in*

$$\mathcal{O}(n\mathtt{M}(\delta)),$$

in terms of arithmetic operations in k.

Note that this method to remove the multiplicity does not work for any kind of parametrization. For example, consider $\mathfrak{I} = (x_1^2, x_2^2)$, x_1 is a primitive element and we have $x_1^4 \in \mathfrak{I}$ and $4x_1^3x_2 - x_1^2 \in \mathfrak{I}$, but x_1^3 does not divide x_1^2.

Algorithm 9: Remove Multiplicity

procedure RemoveMultiplicity(A_t)

\# A_t is an annihilating polynomial of a primitive
\# element u_t modulo $\mathfrak{I}$, coming from Algorithm 7.

\# The procedure returns $q, \mathbf{v}$, a parametrization
\# of $\mathcal{V}(\mathfrak{I})$ for the primitive element u.

\# We write $A_t = A_0 + t_1 A_1 + \cdots + t_n A_n + \mathcal{O}((t_1, \ldots, t_n)^2)$.
  $M \leftarrow \gcd(A_0, A_0')$;
  $q \leftarrow A_0/M$;
  $p \leftarrow A_0'/M$;
  $\mathbf{w} \leftarrow [-A_1/M, \ldots, -A_m/M]$;
  $\mathbf{v} \leftarrow \mathbf{w}/p \bmod q$;
  return$(q, \mathbf{v})$;
end;

6.6 Removing the Extraneous Components

Let $\mathcal{V}$ be a 0-dimensional variety given by a geometric resolution:

$$q(u) = 0, \quad \begin{cases} x_1 &= v_1(u), \\ &\vdots \\ x_n &= v_n(u). \end{cases} \tag{24}$$

Let g be a given polynomial in $k[x_1, \ldots, x_n]$, we are interested in computing a geometric resolution of $\mathcal{V} \backslash \mathcal{V}(g)$. The computations are presented in Algorithm 10.

Proposition 11 *The parametrization*

$$Q(u) = 0, \quad \begin{cases} x_1 &= V_1(u), \\ &\vdots \\ x_n &= V_n(u), \end{cases} \tag{25}$$

returned by Algorithm 10 is a geometric resolution of $\mathcal{V} \backslash \mathcal{V}(g)$.

Lemma 14 *Let L be the complexity of evaluation of g, Algorithm 10 has a complexity in*

$$\mathcal{O}((L + n^2)\mathtt{M}(\delta)),$$

in terms of arithmetic operations in k.

In order to apply this method in the situation of a lifting fiber we must ensure that the choice of the lifting point is not too bad.

Algorithm 10: Cleaning Algorithm

procedure Clean(F, g)

\# F is a geometric resolution of dimension 0.
\# g is a polynomial.

\# At the end F contains a geometric resolution for the variety
\# composed of the points outside $g = 0$.

$q \leftarrow F_{MinimalPolynomial}$;
$\mathbf{v} \leftarrow F_{Parametrization}$;
$e \leftarrow \mathrm{Gcd}(q, g(\mathbf{v}))$;
$q \leftarrow q/e$;
$F_{MinimalPolynomial} \leftarrow q$;
$F_{Parametrization} \leftarrow \mathbf{v} \bmod q$;
end;

Example 7 In $k[x_1, x_2]$, $\mathcal{V} = \mathcal{V}(x_2)$ and $g = x_1$, the choice of $x_1 = 0$ as a lifting point is not a proper choice to compute $\mathcal{V} \backslash \mathcal{V}(g)$.

We now show that almost all choices are correct. Let $\mathcal{V}$ be a r-equidimensional variety given by a lifting fiber, it is sufficient to take the lifting point p of the fiber outside $\pi(\overline{\mathcal{V} \backslash \mathcal{V}(g)} \cap \mathcal{V}(g))$, since then

$$\overline{\mathcal{V} \backslash \mathcal{V}(g)} \cap (x_1 - p_1, \ldots, x_r - p_r) = \mathcal{V} \cap (x_1 - p_1, \ldots, x_r - p_r) \backslash \mathcal{V}(g).$$

Definition 7 *A lifting point is said to be a* cleaning point *with respect to the polynomial g when $p \notin \pi(\overline{\mathcal{V} \backslash \mathcal{V}(g)} \cap \mathcal{V}(g))$.*

Lemma 15 *The lifting points that are not cleaning points are enclosed in an algebraic closed set.*

Proof The hypersurface $g = 0$ intersects regularly $\overline{\mathcal{V} \backslash \mathcal{V}(g)}$. This intersection has dimension $r - 1$, the closure of its projection is a strict algebraic subset of k^r. □

6.7 Summary of the Intersection

We are now able to put §6.3, §6.4, §6.5 and §6.6 together in order to compute a geometric resolution of $\sqrt{\mathfrak{I} + (f)}$. The whole process of intersection is summarized in Algorithm 11.

Lemma 16 *Let C be a geometric resolution of a* 1*-equidimensional ideal $\mathfrak{I}$, u its primitive element, and $\lambda \in k$ a Liouville point for C such that $v = \lambda y + u$ is a primitive element of $\sqrt{\mathfrak{I} + (f)}$. If u, λ and p_r are lucky for Algorithm 11, then it returns a geometric resolution of $\sqrt{\mathfrak{I} + (f)}$. Its complexity is in*

$$\mathcal{O}\big(n(L + n^2)\mathtt{M}(\delta)\mathtt{M}(d\delta)\big),$$

in terms of arithmetic operations in k.

Algorithm 11: One Dimensional Intersect

procedure OneDimensionalIntersect(C, f, λ, g)

\# C is a geometric resolution of $\mathfrak{I}$, 1-equidimensional
\# f is a polynomial intersecting C regularly,
\# λ is a Liouville point of C. Let u be the
\# primitive element of C, $\lambda y + u$ is a primitive
\# element of $\sqrt{\mathfrak{I} + (f)}$,
\# g is a polynomial.

\# The procedure returns F, a geometric resolution of
\# $\mathcal{V}(\mathfrak{I} + (f)) \backslash \mathcal{V}(g)$.

$C_t \leftarrow$ LiftFirstOrderGenericity(C);
$A_t \leftarrow$ KroneckerIntersect(C_t, f, λ);
$q, \mathbf{v} \leftarrow$ RemoveMultiplicity(A_t);
$F_{ChangeOfVariables} \leftarrow C_{ChangeOfVariables}$;
$F_{PrimitiveElement} \leftarrow \lambda y + C_{PrimitiveElement}$;
$F_{MinimalPolynomial} \leftarrow q$;
$F_{Parametrization} \leftarrow \mathbf{v}$;
$F_{Equations} \leftarrow C_{Equations}, f$;
Clean(F, g);
return(F);
end;

7 The Resolution Algorithm

In this section we present the whole resolution algorithm. Let $f_1, \ldots, f_n \in k[x_1, \ldots, x_n]$ be a reduced regular sequence of polynomials outside the hypersurface defined by the polynomial g. That is, if we write $\mathcal{V}_i = \overline{\mathcal{V}(f_1, \ldots, f_i) \backslash \mathcal{V}(g)}$ we have the following situation: for $1 \leq i \leq n$, $\mathcal{V}_i$ is $(n-i)$-equidimensional and for $1 \leq i \leq n-1$, the quotient $(k[x_1, \ldots, x_n]/(f_1, \ldots, f_i))_g$ localized at g is reduced, by the Jacobian criterion this means that the Jacobian matrix of $f_1, \ldots, f_i$ has full rank at each generic point of $\mathcal{V}_i$.

The algorithm is incremental in the number of equations: we solve $\mathcal{V}_1, \ldots, \mathcal{V}_n$ in sequence. We encode each resolution by a lifting fiber. So we need to choose at step i a Noether position for $\mathcal{V}_i$, a lifting point and a primitive element. These choices can be done at random with a low probability of failure, since bad choices are enclosed in strict algebraic subsets.

First we explain the incremental step of the algorithm, then we summarize all the conditions of genericity required by the geometry and the luckiness needed when using an algorithm designed for an integral ring in a non integral one. In §7.3 we discuss the special case when k is $\mathbb{Q}$.

7.1 Incremental Step

Let F_i be a lifting fiber of $\mathcal{V}_i$, in this section we present our method to compute F_{i+1} from F_i, if F_i is generic enough. If this is not the case, we use the techniques of §5 to change the fiber.

We assume that we are given a lifting fiber F for an r-equidimensional variety $\mathcal{V}$, a polynomial f intersecting $\mathcal{V}$ regularly and a polynomial g. Let $\mathfrak{I} = \mathfrak{I}(\mathcal{V})$. We want to compute a lifting fiber for the $(r-1)$-equidimensional variety $\overline{\mathcal{V} \cap \mathcal{V}(f) \backslash \mathcal{V}(g)}$. For the sake of simplicity we assume that the variables $x_1, \ldots, x_n$ are in Noether position for $\mathcal{V}$, let $p = (p_1, \ldots, p_r)$ be a lifting point of $\mathcal{V}$, u a primitive element, q its minimal polynomial on the p-fiber, $x_{r+1} = v_{r+1}(T), \ldots, x_n = v_n(T)$ the parametrization of the dependent variables and $f_1, \ldots, f_{n-r}$ the lifting equations.

In order to apply Algorithm 11 we need to show that the lifted curve intersects regularly the hypersurface $\mathcal{V}(f)$. Let C be the lifted curve of F in the direction of x_r. Namely, let D be the line containing p with direction x_r, $\mathfrak{I}_D = \mathfrak{I} + (x_1 - p_1, \ldots, x_{r-1} - p_{r-1})$ can be seen as a 1-equidimensional ideal of $k[x_r, \ldots, x_n]$. Thanks to the techniques of §4.5 we can compute a geometric resolution C of $\mathfrak{I}_D$ from F.

If the variables $x_1, \ldots, x_n$ are in Noether position for $\mathcal{V} \cap \mathcal{V}(f)$ then there exists a polynomial $A \in k[x_1, \ldots, x_r]$ monic in x_r such that $A \in \mathfrak{I} + (f)$. This implies that $A(p_1, \ldots, p_{r-1}, x_r) \in \mathfrak{I}_D + (f(p_1, \ldots, p_{r-1}, x_r, \ldots, x_n))$. Hence $f(p_1, \ldots, p_{r-1}, x_r, \ldots, x_n)$ intersects regularly the lifted curve, Algorithm 11 applies.

Algorithm 11 applied on C, $f(p_1, \ldots, p_{r-1}, x_r, \ldots, x_n)$, $\lambda \in k$ and $g(p_1, \ldots, p_{r-1}, x_r, \ldots, x_n)$ returns a lifting fiber of $\overline{(\mathcal{V} \cap \mathcal{V}(f)) \backslash \mathcal{V}(g)}$ for the lifting point $(p_1, \ldots, p_{r-1})$ and primitive element $\lambda x_r + u$, if the following conditions hold:

- the Noether position of F is also a Noether position of $\mathcal{V} \cap \mathcal{V}(f)$;
- $(p_1, \ldots, p_{r-1})$ is a lifting point for $\mathcal{V} \cap \mathcal{V}(f)$;
- λ is a Liouville point for C;
- $\lambda y_r + u$ is a primitive element of $\mathcal{V} \cap \mathcal{V}(f)$;
- $(p_1, \ldots, p_{r-1})$ is a cleaning point for $\overline{\mathcal{V} \cap \mathcal{V}(f) \backslash \mathcal{V}(g)}$;
- p_r is lucky for Algorithms 8 and 7;
- u and λ_r are lucky for Algorithm 7.

We have seen that each of the above conditions is generic. If one of them were failing the techniques of §5 would recover a good situation.

> **Example 8** Here is an example where we need to change the primitive element: in $k[t, x_1, x_2]$, let $\mathcal{V}$ be given by the union of two lines D_1 and D_2 parametrized as follows: $(x_1 = 1, x_2 = t)$ and $(x_1 = -1, x_2 = -t)$. The variables t, x_1, x_2 are in Noether position, $t = 0$ is a lifting point and x_2 a primitive element for $t = 0$. Intersecting $\mathcal{V}$ by the equation $x_2 = 0$ the two points solution are $(t = 0, x_1 = 1, x_2 = 0)$ and $(t = 0, x_1 = -1, x_2 = 0)$. For any value of $\lambda \in k$ the linear form $\lambda t + x_2$ does not separate these two points.

7.2 Parameters of the Algorithm

We call the choices on which the algorithm depends its *parameters*. These are functions determining the choices of the Noether positions, lifting points and primitive elements of the fibers $F_1, \ldots, F_n$. In order to make the algorithm compute a correct result, they have to satisfy a few requirements. We have discussed them part by part, we now summarize them.

At step i of the algorithm we have a lifting fiber F_i of $\mathcal{V}_i$, we want to compute a lifting fiber for $\mathcal{V}_{i+1}$. For this we need to choose:

- a Noether position of $\mathcal{V}_{i+1}$, it is determined by a point $\mathcal{N}^{i+1}$ in k^{n-i-1} called the $(i+1)$th *Noether point*;
- a lifting point $\mathcal{L}^{i+1}$ for $\mathcal{V}_{i+1}$;
- a primitive element $u = \lambda_{n-i}y_{n-i} + \cdots + \lambda_n y_n$ for the corresponding fiber, the point $\mathcal{C}^{i+1} = (\lambda_{n-i}, \ldots, \lambda_n)$ is called the $(i+1)$th *Cayley point*.

These three functions $\mathcal{N}$, $\mathcal{L}$, $\mathcal{C}$ constitute the parameters of the algorithm. As seen in the previous subsection, the computations require some more restricting conditions. We distinguish three kinds of restrictions: the first ones are concerned with the geometry of the system, the second ones are also related to the geometry but are specific to the algorithm and the third ones are related with the luckiness of some specializations using algorithms designed for integral rings in case of non integral ones. Namely, let $r = n - i$, we gather all the conditions necessary for the execution and correctness of the whole algorithm:

- The pure geometric restrictions of the algorithm are:
  - Assume that $x_1, \ldots, x_n$ are in Noether position for $\mathcal{V}_i$, the change of variables $x_1 = y_1 + \mathcal{N}_1^{i+1}y_r, \ldots, x_{r-1} = y_{r-1} + \mathcal{N}_{r-1}^{i+1}y_r, x_r = y_r, \ldots, x_n = y_n$ brings the new coordinates y_i into Noether position for $\mathcal{V}_i \cap \mathcal{V}(f_{i+1})$;
  - The lifting point $\mathcal{L}^{i+1} = (p_1, \ldots, p_r)$ is chosen in k^r instead of k^{r-1}, the $r-1$ first coordinates are a lifting point of $\mathcal{V}_i \cap \mathcal{V}(f_{i+1})$ and a cleaning point with respect to g;
  - The Cayley point $\mathcal{C}^{i+1} = (\lambda_r, \ldots, \lambda_n)$ is such that the linear form $\lambda_r y_r + \cdots + \lambda_n y_n$ is primitive for $\mathcal{V}_i \cap \mathcal{V}(f_{i+1})$ for the lifting point $\mathcal{L}^{i+1}$.
- The geometric restrictions specific to the algorithm are:
  - $\mathcal{L}^{i+1}$ is a lifting point of $\mathcal{V}_i$ for the new coordinates y;
  - $u = \lambda_{r+1}y_{r+1} + \cdots + \lambda_n y_n$ is a primitive element of $\mathcal{V}_i$ for the lifting point $\mathcal{L}^{i+1}$, and λ_r is a Liouville point for the lifted curve $\mathcal{V}_i \cap (y_1 - p_1, \ldots, y_{r-1} - p_{r-1})$.
- The luckiness restrictions are:
  - u is lucky for Algorithms 6 and 7;
  - p_r is lucky for Algorithm 7 and 8;
  - λ_r is lucky for Algorithm 7.

Algorithm 12: Geometric Solve

procedure GeometricSolve($\mathbf{f}, g$)

\# $\mathbf{f}$ is a reduced regular system of n equations in n variables
\# g is a polynomial

\# The procedure returns a geometric resolution of the roots of
\# $\mathbf{f} = 0, g \neq 0$

$F \leftarrow$ Initialization;
for i **from** 1 **to** n **do**
ChangeFreeVariables($F, \mathcal{N}^i$);
ChangeLiftingPoint($F, \mathcal{L}^i$);
ChangePrimitiveElement($F, \mathcal{C}^i$);
$C \leftarrow$ subs($t = y_r$,LiftCurve($F, \mathcal{L}^i + (0, \ldots, 0, 1)$));
\# Consider C as the geometric resolution of the
\# corresponding lifted curve to perform:
$F \leftarrow$ OneDimensionalIntersect($C, f_i, \mathcal{C}^i, g$);
od;
return(F);
end;

We have seen along the previous sections that all these restrictions are contained in a Zariski open subset of the space they are lying in. This means that any random choice of these parameters leads to a correct computation with a high probability of success.

The complete algorithm is summarized in Algorithm 12. Let

$$\delta = \max(\deg(\mathcal{V}_1), \ldots, \deg(\mathcal{V}_{n-1}))$$

, d be the maximum of the degrees of the f_i, L the complexity of evaluating $f_1, \ldots, f_{n-r+1}$ and g; $\mathtt{M}$ as before. Combining Lemmas 3, 6 and 16 we get Theorem 1.

The *Initialization* step of Algorithm 12 consists in initializing F as a lifting fiber of the whole space. This particular case must be handled by each subfunctions of the algorithm, for the sake of clarity we do not give more details about this.

7.3 Special Case of the Integers

The complexity of our algorithm is measured in terms of number of arithmetic operations in k. When $k = \mathbb{Q}$ this model does not reflect the real behavior of the method. We now give a method which is efficient in practice, leading to a good running time complexity.

Assume that the input polynomial system $f_1, \ldots, f_n$ is reduced over each point of $\mathcal{V}_n$. Choose now at random a prime number p large enough so that the geometric resolution computed in $\mathbb{Z}/p\mathbb{Z}$ by algorithm 12 is the modular trace of the one computed over $\mathbb{Q}$. It is clear that such prime numbers exist. Now we can apply Algorithm 3 to recover the geometric resolution over $\mathbb{Q}$.

In a future work, we plan to prove that p can be chosen small enough.

8 Practical Results

We have implemented our algorithm within the Magma computer algebra system. The package has been called Kronecker [55] and is available with its documentation at
`http://www.gage.polytechnique.fr/~lecerf/software/kronecker`.

Before presenting some data reporting performances of our method compared to some other ones, we discuss the relevance of such comparisons.

8.1 Relevance of the Comparisons

In computer algebra the best softwares for polynomial solving are based on rewriting techniques. These methods are all deterministic algorithms, so we have to keep in mind that we compare these deterministic algorithms to our probabilistic one. There is a special case when the final number of solutions of the system is equal to the Bézout number of the system, namely $\deg(f_1)\cdots\deg(f_n)$, then we get a deterministic result and the comparison is fair.

We can compare our implementation to Gröbner bases computations and algorithms of change of bases. To compute a Gröbner basis we have several possible choices concerning the elimination order and the algorithm of change of bases. We focus our attention to *grevlex* orders (graded reverse lexicographical order) and *plex* (pure lexicographical order). It is important to notice that our result is stronger than a grevlex basis but weaker than a plex one. One interesting comparison is with a RUR (Rational Univariate Representation) computation [66]: the RUR given in output corresponds exactly to a Kronecker parametrization of the solutions. The software we have retained for these comparisons is: Magma, Gb [25, 26] and RealSolving [66]. To the best of our knowledge they are the best among the most commonly available software for polynomial system solving.

8.2 Systems of Polynomials of Degree 2

We begin with systems composed of n equations in n variables of degree $d = 2$ for different heights h, representing the maximum number of decimal digits of the coefficients of the equations. The number of solutions of the systems is the Bézout number $D = 2^n$.

The following table has be realized with a Compaq Alpha EV6, 500 Mhz, 128 Mb of MEDICIS [2]. The column Gb + Realsolving means that the computations have been done using successively Gb for computing a Gröbner basis for grevlex ordering and Real Solving for computing the RUR from the basis. We have used the interface available within the Mupad computer algebra system [80, 27]. Each entry of the column contains the respective times for each part of the computation. The columns Magma grevlex and lex correspond respectively to Gröbner bases computations for grevlex and lex ordering. Note that Magma uses the Gröbner Walk algorithm in the lex case.

The notation $> 128Mb$ means that the computation can not be performed within $128Mb$.

n	h	Kronecker	Gb grevlex	+	Real Solving	Magma grevlex	Magma lex
4	4	5.4 s	0.5s	+	0.5s	0.3s	1.1s
4	8	6 s	1s	+	1.3s	0.4s	2.2s
4	16	7.5s	2.5s	+	3.7s	0.8s	6s
4	32	11.7s	7s	+	9.3s	1.8s	20s
5	4	29.5s	5s	+	18s	2s	44s
5	8	42.2s	17s	+	57s	5s	155s
5	16	78s	65s	+	180s	15s	563s
5	32	196s	244s	+	592s	46s	2064s
6	4	186s	209s	+	$> 128Mb$	58s	3855s
6	8	335s	773s	+	$> 128Mb$	175s	14112s
6	16	875s	2999s	+	$> 128Mb$	552s	54703s
6	32	2312s	5652s	+	$> 128Mb$	1750s	

This first comparison reveals that our method is faster that Gb+Realsolving, but the more striking is that we are able to compute the same output as Gb+Realsolving even faster than the computation of the grevlex Gröbner basis. Moreover, in this case our result is deterministic since the number of solutions found is equal to the Bézout number of the system.

8.3 Camera Calibration (Kruppa)

The original problem comes from [52] and has been introduced in computer vision in [60]. It is composed of 5 equations in 5 variables. Each equation is a difference of two products of two linear forms. The parameter h is the size of the integers of the input system. The systems have 32 solutions. The comparisons are as above, on the same machine.

h	Kronecker	Gb grevlex	+	Realsolving	Magma grevlex	Magma lex
25	43s	18s	+	36s	5s	118s
60	228s	195s	+	716s	56s	2482s

8.4 Products of Linear Forms

The last example we give is not completely generic. We take 7 equations in 7 variables with integers coefficients of size 18, each equation is a product of two linear forms minus a constant coefficient. The system has 128 solutions, the integers of the output have approximately 8064 decimal digits. The computations have been done using a DEC Alpha EV56, 400 Mhz, 1024 Mb of MEDICIS.

Kronecker	Gb grevlex	Magma grevlex
5h	∞	13.6h

It illustrates the good properties of the practical complexity of our approach.

References

[1] Magma. http://www.maths.usyd.edu.au:8000/u/magma/.

[2] Medicis. http://www.medicis.polytechnique.fr/.

[3] J. Abdeljaoued. *Algorithmes rapides pour le Calcul du Polynôme Caractéristique*. PhD thesis, Université de Franche-Comté, Besançon, France, 1997.

[4] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The design and analysis of computer algorithms*. Addison-Wesley, 1974.

[5] M. Almeida, L. D'Alfonso, and P. Solernó. On the degrees of bases of free modules over a polynomial ring. *Math. Zeitschrift*, pages 1–24, 1998.

[6] M. E. Alonso, E. Becker, M.-F. Roy, and T. Wörmann. Zeroes, multiplicities and idempotents for zerodimensional systems. In *Algorithms in Algebraic Geometry and Applications. Proceedings of MEGA'94*, volume 142 of *Progress in Mathematics*, pages 1–15. Birkhäuser, 1996.

[7] I. Armendáriz and P. Solernó. On the computation of the radical of polynomial complete intersection ideals. In G. Cohen, M. Giusti, and T. Mora, editors, *Applied Algebra, Algebraic Algorithms and Error Correcting Codes. Proceedings of AAECC-11*, volume 948 of *LNCS*, pages 106–119. Springer, 1995.

[8] W. Baur and V. Strassen. The complexity of partial derivatives. *Theoret. Comp. Sci.*, 22:317–330, 1983.

[9] S. J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Information Processing Letters*, 18:147–150, 1984.

[10] D. Bini and V. Pan. *Polynomial and matrix computations*. Progress in theoretical computer science. Birkhäuser Boston-Basel-Berlin, 1994.

[11] A. Borodin and J. Munro. *The Computational Complexity of Algebraic and Numeric Problems*. Elsevier, 1972.

[12] W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system I: The user language. *J. Symbolic Computation*, 24, 1997.

[13] J. Bunch and J. Hopcroft. Triangular factorization and inversion by fast matrix multiplication. *Math.Comp*, (28):231–236, 1974.

[14] P. Bürgisser, M. Clausen, and M. A. Shokrolahi. *Algebraic Complexity Theory*. Springer, 1997.

[15] L. Caniglia, A. Galligo, and J. Heintz. Borne simple exponentielle pour les degrés dans le théorème des zéros sur un corps de caractéristique quelconque. *C. R. Acad. Sci. Paris*, 307:255–258, 1988.

[16] J. Cannon and C. Playoust. Magma: A new computer algebra system. *Euromath Bulletin*, 2(1):113–144, 1996.

[17] J. Canny. Some algebraic and geometric problems in PSPACE. In *Proceedings 20 ACM STOC*, pages 460–467, 1988.

[18] D. Castro, M. Giusti, J. Heintz, G. Matera, and L.M. Pardo. Data structures and smooth interpolation procedures in elimination theory. Manuscript of Facultad de Ciencias, Universitad Cantabria, Santander, 1999.

[19] A. L. Chistov and D. Y. Grigoriev. Polynomial-time factoring of multivariable polynomials over a global field. LOMI preprint E-5-82, Steklov Institute, Leningrad, 1982.

[20] S. Collart, M. Kalkbrener, and D. Mall. Converting bases with the Gröbner walk. *Journal of Symbolic Computation*, 24:465–469, 1997.

[21] D. Coppersmith and S. Winograd. Matrix multiplications via arithmetic progression. In *19th ACM STOC*, pages 1–6, 1987.

[22] L. Csanky. Fast parallel matrix inversion algorithms. *SIAM Journal of Computing*, 5(4):618–623, 1976.

[23] J. Davenport, Y. Siret, and E. Tournier. *Calcul formel. Systèmes et algorithmes de manipulations algébriques*. Masson, 1987.

[24] J. Dixon. Exact solution of linear equations using p-adic expansions. *Numer. Math.*, 40:137–141, 1982.

[25] J.-C. Faugère. *GB Reference Manual*. LITP, 1995. http://posso.ibp.fr/GB.html.

[26] J.-C. Faugère. GB: State of GB + tutorial. LITP, 1997.

[27] J.-C. Faugère and F. Rouillier. Mupad interface for gb/realsolving. http://www.loria.fr/ rouillie/RSDoc/mupdoc/mupdoc.html, 1998.

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

[29] C. M. Fiduccia. A rational view of the fast Fourier transform. In *25th Allerton Conf. Comm., Control and Computing*, 1987.

[30] W. Fulton. *Intersection Theory*. Number 3 in Ergebnisse der Mathematik. Springer, second edition, 1984.

[31] K. O. Geddes, S. R. Czapor, and G. Labahn. *Algorithms for computer algebra*. Kluwer Academic Publishers, 1994.

[32] P. Gianni and T. Mora. Algebraic solution of systems of polynomial equations using Gröbner bases. In *Applied Algebra, Algebraic Algorithms and Error Correcting Codes, Proceedings of AAECC-5*, volume 356 of *LNCS*, pages 247–257. Springer, 1989.

[33] M. Giusti, K. Hägele, J. Heintz, J. E. Morais, J. L. Montaña, and L. M. Pardo. Lower bounds for Diophantine approximation. In *Proceedings of MEGA'96*, number 117,118, pages 277–317. Journal of Pure and Applied Algebra, 1997.

[34] M. Giusti and J. Heintz. La détermination des points isolés et de la dimension d'une variété algébrique peut se faire en temps polynomial. In D. Eisenbud and L. Robbiano, editors, *Computational Algebraic Geometry and Commutative Algebra*, volume XXXIV of *Symposia Matematica*, pages 216–256. Cambridge University Press, 1993.

[35] M. Giusti, J. Heintz, J. E. Morais, J. Morgenstern, and L. M. Pardo. Straight-line programs in geometric elimination theory. *J. of Pure and App. Algebra*, 124:101–146, 1998.

[36] M. Giusti, J. Heintz, J. E. Morais, and L. M. Pardo. When polynomial equation systems can be solved fast ? In G. Cohen, M. Giusti, and T. Mora, editors, *Applied Algebra, Algebraic Algorithms and Error Correcting Codes, Proceedings AAECC-11*, volume 948 of *LNCS*, pages 205–231. Springer, 1995.

[37] M. Giusti, J. Heintz, J. E. Morais, and L. M. Pardo. Le rôle des structures de données dans les problèmes d'élimination. *C. R. Acad. Sci. Paris*, 325:1223–1228, 1997.

[38] M. Giusti, J. Heintz, and J. Sabia. On the efficiency of effective Nullstellensätze. *Computational Complexity*, 3:56–95, 1993.

[39] Marc Giusti, Klemens Hägele, Grégoire Lecerf, Joël Marchand, and Bruno Salvy. Computing the dimension of a projective variety: the projective Noether Maple package. *Journal of Symbolic Computation*, 30(3):291–307, September 2000.

[40] K. Hägele. *Intrinsic height estimates for the Nullstellensatz*. PhD thesis, Universidad de Cantabria, Santander, 1998.

[41] K. Hägele and J. L. Montaña. Polynomial random test for the equivalence problem of integers given by arithmetic circuits. Preprint 4/97 Depto. Matemáticas, Universidad de Cantabria, Santander, Spain, January 1997.

[42] K. Hägele, J. E. Morais, L. M. Pardo, and M. Sombra. On the intrinsic complexity of the arithmetic Nullstellensatz. In J. Heintz, G. Matera, and R. Wachenschauzer, editors, *Proceedings of TERA'97*, Córdoba, Argentina, September 1997. Universidad de Córdoba, Fa.M.A.F, http://tera.medicis.polytechnique.fr/.

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

[44] J. Heintz. On the computational complexity of polynomials and bilinear mappings. A survey. In *Applied Algebra, Algebraic Algorithms and Error Correcting Codes, Proceedings of AAECC-5*, volume 356 of *LNCS*, pages 269–300. Springer, 1989.

[45] J. Heintz, G. Matera, and A. Waissbein. On the time-space complexity of geometric elimination procedures. Manuscript of Universidad Favaloro, Buenos Aires, Argentina, 1999.

[46] J. Heintz, M.-F. Roy, and P. Solernó. Sur la complexité du principe de Tarski-Seidenberg. *Bull. Soc. Math. de France*, 118:101–126, 1990.

[47] H. Kobayashi, T. Fujise, and A. Furukawa. Solving systems of algebraic equations by general elimination method. *J. of Symb. Comp.*, 5:303–320, 1988.

[48] J. König. *Einleitung in die allgemeine Theorie der algebraischen Gröszen.* Druck und Verlag von B. G. Teubner, Leipzig, 1903.

[49] T. Krick and L. M. Pardo. Une approche informatique pour l'approximation diophantienne. *C. R. Acad. Sci. Paris*, 318(1):407–412, 1994.

[50] T. Krick and L. M. Pardo. A computational method for Diophantine approximation. In L. González-Vega and T. Recio, editors, *Algorithms in Algebraic Geometry and Applications. Proceedings of MEGA'94*, volume 143 of *Progress in Mathematics*, pages 193–254. Birkhäuser Verlag, 1996.

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

[52] E. Kruppa. Zur Ermittlung eines Objektes aus zwei Perpesktiven mitinnerer Orientierung. *Sitz.-Ber. Akad. Wiss., Wien. Math. Naturw. Kl.*, 1913.

[53] E. Kunz. *Introduction to Commutative Algebra and Algebraic Geometry.* Birkhäuser Verlag, 1985.

[54] Y. N. Lakshman and D. Lazard. On the complexity of zero-dimensional algebraic systems. In *Effective methods in algebraic geometry*, volume 94 of *Progress in Mathematics*, pages 217–225. Birkhäuser, 1991.

[55] G. Lecerf. *Kronecker, a package for Magma for polynomial system solving.* UMS MEDICIS, Laboratoire GAGE, http://www.gage.polytechnique.fr/~lecerf/software/kronecker, 1999.

[56] U. J. J. Leverrier. Sur les variations séculaires des éléments elliptiques des sept planètes principales: Mercure, Vénus, la terre, Mars, Jupiter, Saturne et Uranus. *J. Math. Pures Appli.*, 4:220–254, 1840.

[57] T. Lickteig and M.-F. Roy. Sylvester-Habicht sequences and fast Cauchy index computation. In *Calcolo 33*, pages 337–371, 1996.

[58] F. S. Macaulay. *The Algebraic Theory of Modular Systems.* Cambridge University Press, 1916.

[59] G. Matera. *Sobre la complejidad en espacio y tiempo de la eliminación geométrica.* PhD thesis, Universidad de Buenos Aires, Argentina, 1997.

[60] S. Maybank and O. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, 1992.

[61] R. Moenck and A. B. Borodin. Fast computation of GCD's. In *5th Annual ACM Symposium on Theory of Computing*, pages 142–151, 1973.

[62] J. E. Morais. *Resolución eficaz de sistemas de ecuaciones polinomiales.* PhD thesis, Universidad de Cantabria, Santander, Spain, 1997.

[63] H. J. Nussbaumer. Fast polynomial transform algorithms for digital convolutions. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 28:205–215, 1980.

[64] L. M. Pardo. How lower and upper complexity bounds meet in elimination theory. In G. Cohen, M. Giusti, and T. Mora, editors, *Applied Algebra, Algebraic Algorithms and Error Correcting Codes, Proceedings of AAECC-5*, volume 948 of *Lecture Notes in Computer Science*, pages 33–69. Springer, Berlin, 1995.

[65] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. Part I. *Journal of Symbolic Computation*, 13(3):255–299, March 1992.

[66] F. Rouillier. *Algorithmes efficaces pour l'étude des zéros réels des systèmes polynomiaux*. PhD thesis, Université de Rennes I, may 1996.

[67] F. Rouillier. Solving zero-dimensional systems through the rational univariate representation. *Journal AAECC*, 6:353–376, 1996.

[68] J. Sabia and P. Solernó. Bounds for traces in complete intersections and degrees in the Nullstellensatz. *Journal AAECC*, 6:353–376, 1996.

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

[70] A. Schönhage. Schnelle Multiplikation von Polynomen über Körpern der Charakteristik 2. *Act. Inf*, (7):395–398, 1977.

[71] A. Schönhage and V Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, (7):281–292, 1971.

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

[73] M. Sieveking. An algorithm for division of power series. *Computing*, 10:153–156, 1972.

[74] S. Smale. The fundamental theorem of algebra and complexity theory. *Bulletin of the Amer. Math. Soc.*, (4):1–36, 1981.

[75] S. Smale. Algorithms for solving equations. In *Proceedings of the International Congress of Mathematicians*, pages 172–195, Berkeley, California, USA, 1986.

[76] H.-J. Stoß. On the representation of rational functions of bounded complexity. *Theo. Comp. Sci.*., 64:1–13, 1989.

[77] V. Strassen. Berechnung und Programm. I, II. *Acta Informatica*, 1(4):320–355; *ibid.* 2(1), 64–79 (1973), 1972.

[78] V. Strassen. Die Berechnungskomplexität von elementarysymmetrischen Funktionen un von Interpolationskoeffizienten. *Num. Math.*, (20):238–251, 1973.

[79] TERA Development Group. A (hopefully) efficient polynomial equation system solver. Manuscript 57 pages, `gmatera@mate.dm.uba.ar`, 1998.

[80] The MuPAD Group, Benno Fuchssteiner et al. *MuPAD User's Manual - MuPAD Version 1.2.2*. John Wiley and sons, Chichester, New York, first edition, march 1996.

[81] W. Vogel. *Results on Bézout's Theorem*. Tata Institute of Fundamental Research. Springer, 1984.

[82] J. von zur Gathen. Parallel arithmetic computations: a survey. In B. Rovan J. Gruska and J. Wiedermann, editors, *Proceedings of the 12th Symposium on Mathematical Foundations of Computer Science*, volume 233 of *LNCS*, pages 93–112, Bratislava, Czechoslovakia, August 1986. Springer.

[83] P. Wadler. Deforestation: transforming programs to eliminate trees. *Theoretical Computer Science*, 73:231–248, 1990. Special issue of selected papers from 2nd ESOP.

[84] V. Weispfenning and T. Becker. *Groebner bases: a computational approach to commutative algebra*, volume 141 of *Graduate Texts in Mathematics: readings in mathematics*. Springer, 1993.

[85] R. Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings EUROSAM' 79*, number 72 in LNCS, pages 216–226. Springer, 1979.

[86] R. Zippel. *Effective Polynomial Computation*. ECS 241. Kluwer Academic Publishers, 1993.