Improved algorithm for computing separating linear forms for bivariate systems

Yacine Bouzidi, Sylvain Lazard, Guillaume Moroz, Marc Pouget, Fabrice Rouillier

Improved algorithm for computing separating linear forms for bivariate systems

Yacine Bouzidi
INRIA Nancy Grand Est
LORIA, Nancy, France
Yacine.Bouzidi@inria.fr

Sylvain Lazard
INRIA Nancy Grand Est
LORIA, Nancy, France
Sylvain.Lazard@inria.fr

Guillaume Moroz
INRIA Nancy Grand Est
LORIA, Nancy, France
Guillaume.Moroz@inria.fr

Marc Pouget
INRIA Nancy Grand Est
LORIA, Nancy, France
Marc.Pouget@inria.fr

Fabrice Rouillier
INRIA Paris-Rocquencourt
IMJ, Paris, France
Fabrice.Rouillier@inria.fr

ABSTRACT

We address the problem of computing a linear separating form of a system of two bivariate polynomials with integer coefficients, that is a linear combination of the variables that takes different values when evaluated at the distinct solutions of the system. The computation of such linear forms is at the core of most algorithms that solve algebraic systems by computing rational parameterizations of the solutions and this is the bottleneck of these algorithms in terms of worst-case bit complexity. We present for this problem a new algorithm of worst-case bit complexity $\widetilde{O}_B(d^7 + d^6\tau)$ where d and τ denote respectively the maximum degree and bitsize of the input (and where $\widetilde{O}$ refers to the complexity where polylogarithmic factors are omitted and O_B refers to the bit complexity). This algorithm simplifies and decreases by a factor d the worst-case bit complexity presented for this problem by Bouzidi et al. [5]. This algorithm also yields, for this problem, a probabilistic Las-Vegas algorithm of expected bit complexity $\widetilde{O}_B(d^5 + d^4\tau)$.

1. INTRODUCTION

A classical approach for solving a system of polynomials with a finite number of solutions is to compute a rational parameterization of its solutions.

A rational parameterization is a representation of the (complex) solutions by a set of univariate polynomials and associated rational one-to-one mappings that send the roots of the univariate polynomials to the solutions of the system. Such representations enable to reduce computations on the system to computations with univariate polynomials and thus ease, for instance, the isolation of the solutions or the evaluation of other polynomials at the solutions.

At the core of the algorithms that compute such parameterizations (see for example [1, 3, 6, 8, 9, 14] and references therein), is the computation of a so-called *linear separating form* for the solutions, that is a linear combination of the coordinates that takes different values when evaluated at different solutions of the system. Since a random linear form is separating with probability one, probabilist Monte-Carlo algorithms can overlook this issue. However, when it comes to deterministically computing a linear separating form, or even to check that an arbitrary chosen form is separating, this, surprisingly, turns out to be the bottleneck in the computation of rational parameterizations, in particular for bivariate systems as discussed below. This explains why, among the many algorithms that compute rational parameterizations, seldom search deterministically for a separating linear form.

Considering systems of two bivariate polynomials of total degree bounded by d with integer coefficients of bitsize bounded by τ, one approach for computing a separating linear form together with a rational parameterization of the solutions has been presented by Gonzalez-Vega and El Kahoui [9] and its bit complexity analyzed in [6]. The analysis of this approach shows a bit complexity in $\widetilde{O}_B(d^{10} + d^9\tau)$ for computing a separating form and a bit complexity in $\widetilde{O}_B(d^7 + d^6\tau)$ for computing the corresponding rational parameterization. The computation of a separating linear form was thus the bottleneck in the computation of the rational parameterization. This is still true even when considering the additional phase of computing isolating boxes of the solutions (from the rational parameterization), which state-of-the-art complexity is in $\widetilde{O}_B(d^8 + d^7\tau)$ [5, Prop. 35].

More recently, Bouzidi et al. [5] presented a new algorithm for computing a separating linear form that reduces the previous bit complexity to $\widetilde{O}_B(d^8 + d^7\tau)$. The same authors also showed that, given such a separating linear form, an alternative rational parameterization called RUR [14] can be computed using $\widetilde{O}_B(d^7 + d^6\tau)$ bit operations [5, Thm. 22] and that isolating boxes of the solutions can be computed from this RUR in $\widetilde{O}_B(d^6 + d^5\tau)$ [4, Thm. 6.1.2]. Consequently, despite the complexity improvement brought to the separating form computation, this step was still the bottleneck in the computation of a rational parameterization of a bivariate system and more generally in the whole solving process, i.e. including the numerical isolation phase.

In addition, although the problem of searching deterministically for a separating form is interesting from the theoret-

ISSAC'14 July 23–25, 2014, Kobe, Japan.

ical point of view, in practice, a preferable approach would be to design a Las-Vegas algorithm that chooses randomly a linear form and then checks that the latter is separating. However up to now, the problem of checking that an arbitrary linear form is separating has not been shown to be easier (at least in terms of asymptotic bit complexity) than the deterministic computation of a separating linear form.

Main results. Our main contribution is a new deterministic algorithm of worst-case bit complexity $\widetilde{O}_B(d^7 + d^6\tau)$ for computing a separating linear form of a zero-dimensional system of two bivariate polynomials of total degree at most d and integer coefficients of bitsize at most τ (Theorem 13).

This algorithm is simpler than the one presented by Bouzidi et al. [5] and it decreases by a factor d its complexity. This brings the complexity of solving bivariate systems by computing a rational parameterization to $\widetilde{O}_B(d^7 + d^6\tau)$.

A second contribution is a Las-Vegas algorithm for computing a separating linear form with an expected bit complexity in $\widetilde{O}_B(d^5 + d^4\tau)$ (Theorem 19). This Las-Vegas algorithm stems naturally from the previous algorithm replacing the deterministic version of the univariate gcd computation by a Las-Vegas one. Recall that, in Las-Vegas algorithms, the result is always correct and only the running time is probabilistic.

2. OVERVIEW

Our algorithm is based on the one presented by Bouzidi et al. [5] on the same problem. For clarity, we briefly recall the essence of that algorithm. It first computes the number of distinct (complex) solutions of the input system $\{P, Q\}$ as well as a prime number μ such that the input system considered modulo μ has the same number of distinct solutions. This first step has worst-case bit complexity $\widetilde{O}_B(d^8 + d^7\tau)$. All polynomials and computations are thereafter considered modulo μ. The algorithm then considers iteratively a candidate separating element $x + ay$ with an integer a incrementing from 0. The input polynomials are considered through a shearing of the coordinate system $(x, y) \rightsquigarrow (t - ay, y)$, and the degree of the squarefree part of their resultant (with respect to y) is computed; in other words, the algorithm computes the number of distinct solutions after projection along the direction of the line $x + ay = 0$. The algorithm stops when a value a is found such that the number of distinct projected solutions equals that of the system. This step trivially computes a separating element $x + ay$ of the input system considered modulo μ but the proof that this form is also separating of the input system is not straightforward. This second step of the algorithm is presented in [5] with the same worst-case bit complexity as the first step but we show in Section 4 that it is straightforward to slightly modify it so that it has complexity $\widetilde{O}_B(d^7 + d^3\tau)$.

We present in this paper an improvement of the above algorithm using the following two ingredients. First, we show in Section 5 that computing a separating linear form for a system $\{P, Q\}$ is essentially equivalent (in terms of asymptotic bit complexity) to computing a separating linear form for the critical points of a curve. Second, we present in Section 6 a rather simple algorithm of worst-case bit complexity $\widetilde{O}_B(d^7 + d^6\tau)$ for computing the number of critical points of a curve, as well as a prime number μ such that the curve modulo μ has the same number of critical points. In essence, given a curve of equation H, this algorithm first computes a subresultant-based triangular decomposition [9] of the system $\{H, \frac{\partial H}{\partial y}\}$ and the sum of the degrees of the resulting systems; the same computation is done for the system $\{H, (\frac{\partial H}{\partial y})^2\}$ and we show that the difference of these two sums of degrees is equal to the number of critical points of the curve H. We then perform the same computation modulo some prime numbers μ until the same number of critical points is obtained. Finally, given this number of solutions and a corresponding prime μ, we obtain a separating linear form for the input system by applying the variant presented in Section 4 of the algorithm of [5] for computing a separating linear form for the critical points of the curve.

Furthermore, we show in Section 8 how this algorithm naturally extends to a Las-Vegas algorithm of expected bit complexity $\widetilde{O}_B(d^5 + d^4\tau)$.

3. NOTATION AND PRELIMINARIES

We introduce notation and recall some classical material. Most of the material in this section is taken literally from [5].

The bitsize of an integer p is the number of bits needed to represent it, that is $\lfloor \log p \rfloor + 1$ (log refers to the logarithm in base 2). The bitsize of a polynomial with integer coefficients is the *maximum* bitsize of its coefficients. As mentioned earlier, O_B refers to the bit complexity and $\widetilde{O}$ and $\widetilde{O}_B$ refer to complexities where polylogarithmic factors are omitted, see [15, Def. 25.8] for details.

In the following, μ is a prime number and we denote by $\mathbb{Z}_\mu$ the quotient $\mathbb{Z}/\mu\mathbb{Z}$. We denote by $\phi_\mu \colon \mathbb{Z} \to \mathbb{Z}_\mu$ the reduction modulo μ, and extend this definition to the reduction of polynomials with integer coefficients. We denote by $\mathbb{D}$ a unique factorization domain, typically $\mathbb{Z}[x, y]$, $\mathbb{Z}[x]$, $\mathbb{Z}_\mu[x]$, $\mathbb{Z}$ or $\mathbb{Z}_\mu$. We also denote by $\mathbb{F}$ a field, typically $\mathbb{Q}$, $\mathbb{C}$, or $\mathbb{Z}_\mu$ and by $\mathbb{F}_\mathbb{D}$ the fraction field of $\mathbb{D}$.

For any polynomial $P \in \mathbb{D}[x]$, let $Lc_x(P)$ denote its leading coefficient with respect to the variable x and $d_x(P)$ its degree with respect to x. For any curve defined by $H(x, y) \in \mathbb{D}[x, y]$, we call the critical points of H with respect to x or more shortly the critical point of H, the points that are solutions of the system $\{H, \frac{\partial H}{\partial y}\}$. In this paper, the solutions of a system of polynomial are always considered in the algebraic closure of $\mathbb{F}_\mathbb{D}$.

Subresultant sequences. We first recall the concept of *polynomial determinant* of a matrix which is used in the definition of subresultants. Let M be an $m \times n$ matrix with $m \leqslant n$ and M_i be the square submatrix of M consisting of the first $m - 1$ columns and the i-th column of M, for $i = m, \ldots, n$. The *polynomial determinant* of M is the polynomial defined as $\det(M_m)y^{n-m} + \det(M_{m+1})y^{n-(m+1)} + \cdots + \det(M_n)$.

Let $P = \sum_{i=0}^{p} a_i y^i$ and $Q = \sum_{i=0}^{q} b_i y^i$ be two polynomials in $\mathbb{D}[y]$ and assume without loss of generality that $p \geqslant q$. The Sylvester matrix of P and Q, $Sylv(P, Q)$ is the $(p + q)$-square matrix whose rows are $y^{q-1}P, \ldots, P, y^{p-1}Q, \ldots, Q$ considered as vectors in the basis $y^{p+q-1}, \ldots, y, 1$.

DEFINITION 1. *([7, §3]). For $i = 0, \ldots, \min(q, p - 1)$, let $Sylv_i(P, Q)$ be the $(p + q - 2i) \times (p + q - i)$ matrix obtained from $Sylv(P, Q)$ by deleting the i last rows of the coefficients of P, the i last rows of the coefficients of Q, and the i last columns.*

For $i = 0, \ldots, \min(q, p - 1)$, the i-th polynomial subresultant of P and Q, denoted by $Sres_{y,i}(P, Q)$ is the polynomial determinant of $Sylv_i(P, Q)$. When $q = p$, the q-th polynomial subresultant of P and Q is $b_q^{-1}Q$.

$Sres_{y,i}(P,Q)$ has degree at most i in y, and the coefficient of its monomial of degree i in y, denoted by $sres_{y,i}(P,Q)$, is called the i-th *principal subresultant coefficient*. Note that $Sres_{y,0}(P,Q) = sres_{y,0}(P,Q)$ is the *resultant* of P and Q with respect to y, which we also denote by $Res_y(P,Q)$.

We state below a fundamental property of subresultants which is instrumental in the triangular decomposition algorithm used in Section 6.1. For clarity, we state this property for bivariate polynomials $P = \sum_{i=0}^{p} a_i y^i$ and $Q = \sum_{i=0}^{q} b_i y^i$ in $\mathbb{D}[x,y]$, with $p \geqslant q$. Note that this property is often stated with a stronger assumption that is that *none* of the leading terms $a_p(\alpha)$ and $b_q(\alpha)$ vanishes. This property is a direct consequence of the specialization property of subresultants and of the gap structure theorem; see for instance [7, Lemmas 2.3, 3.1 and Cor. 5.1].

LEMMA 2. *For any α such that $a_p(\alpha)$ and $b_q(\alpha)$ do not both vanish, the first $Sres_{y,k}(P,Q)(\alpha,y)$ (for k increasing) that does not identically vanish is of degree k and it is the gcd of $P(\alpha,y)$ and $Q(\alpha,y)$ (up to a nonzero constant in the fraction field of $\mathbb{D}(\alpha)$).*

Complexity. We recall complexity results, using fast algorithms, on subresultants and gcd computations.

LEMMA 3 ([2, PROP. 8.46] [13, §8] [15, COR. 11.15]). *Let P and Q be in $\mathbb{Z}[x_1,\ldots,x_n][y]$ (n fixed) with coefficients of bitsize at most τ such that their degrees in y are bounded by d_y and their degrees in the other variables are bounded by d.*

- *The coefficients of $Sres_{y,i}(P,Q)$ have bitsize in $\widetilde{O}(d_y\tau)$.*
- *The degree in x_j of $Sres_{y,i}(P,Q)$ is at most $2d(d_y-i)$.*
- *Any subresultant $Sres_{y,i}(P,Q)$ as well as the sequence of principal subresultant coefficients $sres_{y,i}(P,Q)$ can be computed in $\widetilde{O}(d^n d_y^{n+1})$ arithmetic operations, and $\widetilde{O}_B(d^n d_y^{n+2}\tau)$ bit operations.*

In the sequel, we often consider the gcd of two univariate polynomials P and Q and the gcd-free part of P with respect to Q, that is, the divisor D of P such that $P = \gcd(P,Q)D$. Note that, when $Q = P'$, the latter is the squarefree part of P, provided that the characteristic of the coefficient ring is zero or sufficiently large (e.g., larger than the degree of P).

LEMMA 4 ([2, REM. 10.19]). *Let P and Q in $\mathbb{F}[x]$ of degree at most d. $\gcd(P,Q)$ or the gcd-free part of P with respect to Q can be computed with $\widetilde{O}(d)$ operations in $\mathbb{F}$.*

4. SEPARATING LINEAR FORM

As mentioned in the overview, our approach for computing a separating form of a zero-dimensional system $\{P,Q\}$ is similar to the one in [5] once we know the number of distinct solutions and a so-called lucky prime μ. Such a lucky prime is, roughly speaking, a prime such that $\{P,Q\}$ has the same number of distinct solutions as its image modulo μ. Before presenting Algorithm 1, which computes a separating linear form in this context, we introduce the following notation and formally define lucky primes.

Given the two input polynomials P and Q, we consider the "generic" change of variables $x = t - sy$, and define the "sheared" polynomials $P(t-sy,y)$, $Q(t-sy,y)$, and their resultant with respect to y,

$$R(t,s) = Res_y(P(t-sy,y), Q(t-sy,y)).$$

Algorithm 1 Separating form for $\{P,Q\}$

Input: P,Q in $\mathbb{Z}[x,y]$ of total degree at most d and defining a zero-dimensional system, its number N of distinct (complex) solutions and a lucky prime μ of bitsize $O(\log d)$

Output: A separating linear form $x+ay$ for $\{P,Q\}$, with $a < 2d^4$

1: Compute $P(t-sy,y)$ and $Q(t-sy,y)$
2: Compute $\Upsilon_\mu(s) = \phi_\mu(L_P(s))\ \phi_\mu(L_Q(s))$
3: Compute $P_\mu = \phi_\mu(P)$ and $Q_\mu = \phi_\mu(Q)$
4: $a := 0$
5: **repeat**
6: Compute $P_\mu(t-ay,y)$, $Q_\mu(t-ay,y)$ and their resultant $R_{\mu,a}(t)$
7: Compute the degree N_a of the squarefree part of $R_{\mu,a}(t)$
8: $a := a+1$
9: **until** $\Upsilon_\mu(a) \neq 0$[2]and $N_a = N$
10: **return** The linear form $x+ay$

We introduce the following notation for the leading coefficients of these polynomials;

$$L_P(s) = Lc_y(P(t-sy,y)) \quad L_Q(s) = Lc_y(Q(t-sy,y)).$$

Note that these polynomials do not depend on t.

DEFINITION 5 ([5, DEF. 8]). *A prime number μ is said to be* **lucky** *for a zero-dimensional system $\{P,Q\}$ if $\{P,Q\}$ and $\{\phi_\mu(P),\phi_\mu(Q)\}$ have the same number of distinct solutions and if $\mu > 2d^4$ and*

$$\phi_\mu(L_P(s))\ \phi_\mu(L_Q(s)) \not\equiv 0.$$

Note that we consider μ in $\Omega(d^4)$ in Definition 5 because, in Algorithm 1, we want to ensure that there exists, for the system $\{P_\mu,Q_\mu\}$ (resp. $\{P,Q\}$), a separating form $X+aY$ with $a \in \mathbb{Z}_\mu$ (resp. $0 \leqslant a < \mu$ in $\mathbb{Z}$). The constant 2 in the bound $2d^4$ is an overestimate, which simplifies some proofs in [5].

Recall that we consider we know the number of distinct (complex) solutions of system $\{P,Q\}$ and a lucky prime μ for that system. Algorithm 4 of [5] computes a separating linear form for $\{P,Q\}$ by considering iteratively linear forms $x+ay$, where a is an integer incrementing from 0 and by computing the degree of the squarefree part of the reduction modulo μ of $R(t,a)$ until this degree is equal to the (known) number of distinct solutions of the system and such that $\phi_\mu(L_P(a))\ \phi_\mu(L_Q(a)) \neq 0$.

Doing so, the algorithm computes a separating form for the system modulo μ, which, under the hypothesis of the luckiness of μ, has been proven to be also separating for the system $\{P,Q\}$. In Algorithm 1, we follow the same approach except that we perform the computations in a slightly different way[1] so that the complexity is in $\widetilde{O}_B(d^7+d^3\tau)$ (instead of $\widetilde{O}_B(d^8+d^7\tau)$ in [5]).

[1]Namely, in Algorithm 1, we first compute the reduction modulo μ of the input polynomials P and Q (Line 3) and then, for every value of a, the resultant of their sheared images through the change of variables $(x,y) \rightsquigarrow (t-ay,y)$ (Line 6), while in [5, Algorithm 4], we first compute the reduction modulo μ of the resultant $R(t,s)$ and then, for every value of a, its specialization at $s=a$.

[2]$\Upsilon_\mu(s) \in \mathbb{Z}_\mu[s]$ and we consider $\Upsilon_\mu(a)$ in $\mathbb{Z}_\mu$.

PROPOSITION 6. *Algorithm 1 computes a separating linear form $x+ay$ for $\{P,Q\}$ with $a < 2d^4$ with a bit complexity $\widetilde{O}_B(d^7 + d^3\tau)$.*

PROOF. We first prove the correctness of Algorithm 1 which essentially follows from [5, Algorithm 4]. The latter algorithm computes the degree of the squarefree part of $\phi_\mu(R)(t,a)$ until the condition of Line 9 is satisfied, and it returns the corresponding form $x + ay$. It is thus sufficient to argue that $\phi_\mu(R)(t,a) = R_{\mu,a}(t)$.

Denoting by ψ_a the morphism that evaluates a polynomial at $s = a$, and Res_y the resultant with respect to y, we have

$$\phi_\mu(R)(t,a) = \psi_a \circ \phi_\mu(\mathrm{Res}_y(P(t-sy,y),Q(t-sy,y)) = \\ \mathrm{Res}_y(\psi_a \circ \phi_\mu(P(t-sy,y)),\psi_a \circ \phi_\mu(Q(t-sy,y)))$$

by the specialization property of the resultants since the leading coefficients of P and Q (with respect to y) do not vanish through $\psi_a \circ \phi_\mu$ when the condition $\Upsilon_\mu(a) \neq 0$ is satisfied in Line 9. Furthermore, $\phi_\mu(P(t-sy,y)) = \phi_\mu(P)(t-sy,y)$ and similarly for Q, which implies that the right-hand side of the equation is equal to $R_{\mu,a}(t)$. This concludes the proof of correctness. Note that this correctness includes the property that the output integer a is less than $2d^4$.

We now prove the complexity of our algorithm. It is straightforward that, in Line 1, the sheared polynomials $P(t-sy,y)$ and $Q(t-sy,y)$ can be computed in bit complexity $\widetilde{O}_B(d^4 + d^3\tau)$ and that their bitsizes are in $\widetilde{O}(d+\tau)$ (see e.g. [5, Lemma 7]). In Lines 2 and 3, the polynomials, in one or two variables, have degree at most d and bitsize $\widetilde{O}(d+\tau)$. The reduction of each of their $O(d^2)$ coefficients modulo μ can be done in a bit complexity that is softly linear in the maximum bitsizes [15, Thm. 9.8], that is in a total bit complexity of $\widetilde{O}_B(d^3 + d^2\tau)$. In Line 6, computing the polynomials $P_\mu(t-ay,y)$ and $Q_\mu(t-ay,y)$ is performed using $\widetilde{O}_B(d^3)$ bit operations (see e.g. the proof [5, Lemma 7]) and similarly for their resultant $R_{\mu,a}(t)$ according to Lemma 3. In Line 7, the squarefree part of $R_{\mu,a}(t)$ can also be computed in $\widetilde{O}_B(d^3)$ bit operations by Lemma 4, since the resultant has degree $O(d^2)$. We have shown that the loop stops with $a < 2d^4$, thus the whole loop has complexity $\widetilde{O}_B(d^7)$, which concludes the proof. □

5. FROM A SYSTEM TO A CURVE

In this section, we consider two polynomials $P, Q \in \mathbb{Z}[x,y]$ of total degree at most d and maximum bitsize τ and show that it is essentially equivalent from an asymptotic worst-case bit complexity point of view to compute a separating linear form for a system $\{P,Q\}$ and to compute a separating linear form for the critical points of a curve. For simplicity, we refer to the latter as a separating linear form for a curve.

By definition, the critical points of a curve of equation H are the solutions of the system $\{H, \frac{\partial H}{\partial y}\}$, thus computing a separating linear form for a curve amounts by definition to computing a separating linear form for a system of two equations. Conversely, a separating linear form for the curve PQ is also separating for the system $\{P,Q\}$ since any solution of $\{P,Q\}$ is also solution of PQ and of $\frac{\partial PQ}{\partial y} = P\frac{\partial Q}{\partial y} + \frac{\partial P}{\partial y}Q$.

However, it may happen that the curve PQ admits no separating linear form even if $\{P,Q\}$ admits one. Indeed, $\{P,Q\}$ can be zero-dimensional while PQ is not squarefree (and such that the infinitely many critical points cannot be separated by a linear form). Nevertheless, if P and Q are coprime and squarefree, then PQ is squarefree and thus it has finitely many singular points. Still the curve $H = PQ$ may contain vertical lines, and thus infinitely many critical points, but this issue can easily be handled by shearing the coordinate system.

LEMMA 7. *Given a zero-dimensional system of two polynomials P and Q in $\mathbb{Z}[x,y]$ of maximum degree d and maximum bitsize τ, we can compute in complexity $\widetilde{O}_B(d^6 + d^5\tau)$ a shearing of the coordinate system $(x,y) \rightsquigarrow (t-\alpha y, y)$ (α integer in $O(d)$) and a polynomial H in $\mathbb{Z}[t,y]$ of degree at most $2d$ and bitsize $\widetilde{O}(d+\tau)$ so that the system $\{H, \frac{\partial H}{\partial y}\}$ is zero-dimensional and any separating linear form for that system is also separating for $\{P,Q\}$ after being sheared back.*

PROOF. As discussed above, we first compute the squarefree part of each polynomial P and Q, which can be done in complexity $\widetilde{O}_B(d^6 + d^5\tau)$ [11, Lemma 13]. Let $H(x,y)$ denote their product, which is squarefree since P and Q are coprime. We then consider a generic shearing of the coordinate system $(x,y) \rightsquigarrow (t-sy,y)$ in order to find a value $s = \alpha$ so that the sheared curve $\widehat{H}(t,y) = H(t-\alpha y, y)$ has no vertical asymptote and thus no vertical line. The leading coefficient of $H(t-sy,y)$ (seen as a polynomial in y) is a polynomial of degree at most d in $\mathbb{Z}[s]$ (t does not appear in the leading term); furthermore an expanded form of $H(t-sy,y)$ can be computed in complexity $\widetilde{O}_B(d^4 + d^3\tau)$ and the coefficients have bitsize $\widetilde{O}(d+\tau)$ (see e.g. [5, Lemma 7]). Finding an integer value $s = \alpha$ where the leading coefficient does not vanish can thus be done in d evaluations of complexity $\widetilde{O}_B(d(d+\tau))$ each [5, Lemma 6] and such α can be found in $[0,d]$. Then, computing $H(t-\alpha y, y)$ can be done by evaluating each of the coefficients of $H(t-sy,y)$ at $s = \alpha$, which can again be done with $O(d)$ evaluations of complexity $\widetilde{O}_B(d(d+\tau))$ each. Thus, we can shear the curve in complexity $\widetilde{O}_B(d^4 + d^3\tau)$ so that the leading coefficient of the resulting polynomial $\widehat{H}(t,y) = H(t-\alpha y, y)$ (seen as a polynomial in y) is a constant.

Modulo the shearing, all solutions of $\{P,Q\}$ are solutions of the system $\{\widehat{H}, \frac{\partial \widehat{H}}{\partial y}\}$. Indeed, a solution (x_0,y_0) of $\{P,Q\}$ is such that $(t_0 = x_0 + \alpha y_0, y_0)$ is solution of $\{\widehat{P},\widehat{Q}\}$ with $\widehat{P}(t,y)$ equal to the squarefree part of $P(t-\alpha y,y)$ and similarly for $\widehat{Q}$; thus (t_0,y_0) is solution of $\widehat{H} = \widehat{P}\widehat{Q}$ and of $\frac{\partial \widehat{H}}{\partial y} = \widehat{P}\frac{\partial \widehat{Q}}{\partial y} + \frac{\partial \widehat{P}}{\partial y}\widehat{Q}$. Thus, any separating linear form for $\{\widehat{H}, \frac{\partial \widehat{H}}{\partial y}\}$ is also separating for $\{P,Q\}$ modulo the shearing. Finally, $\{\widehat{H}, \frac{\partial \widehat{H}}{\partial y}\}$ is zero-dimensional since, by construction, $\widehat{H}$ is squarefree and contains no vertical line. Renaming $\widehat{H}$ by H, this concludes the proof. □

6. THE CASE OF A CURVE

In this section, we consider an arbitrary curve defined by $H \in \mathbb{Z}[x,y]$ of degree d and bitsize τ, with a constant leading coefficient in y, and such that H has a finite number of critical points, i.e., the system $\{H, \frac{\partial H}{\partial y}\}$ is zero-dimensional. We show in the following that (i) computing the number of the critical points of H and (ii) computing a lucky prime for $\{H, \frac{\partial H}{\partial y}\}$ (see Definition 5) can be done in a bit complexity in $\widetilde{O}_B(d^7 + d^6\tau)$. Combined with the results of the previous sections, this will yield that we can compute a separating

Algorithm 2 Triangular decomposition [9, 10]

Input: P, Q in $\mathbb{F}[x, y]$ coprime such that $Lc_y(P)$ and $Lc_y(Q)$ are coprime, $d_y(Q) \leqslant d_y(P)$
Output: Triangular decomp. $\{(A_i(x), B_i(x, y))\}_{i\in\mathcal{I}}$ such that the set of solutions of $\{P, Q\}$) is the disjoint union of the sets of solutions of $\{A_i(x), B_i(x, y)\}_{i\in\mathcal{I}}$
1: Compute the subresultant sequence of P and Q with respect to y: $B_i = Sres_{y,i}(P, Q)$
2: $G_0 =$ squarefree part$(Res_y(P, Q))$ and $\mathcal{T} = \emptyset$
3: **for** $i = 1$ **to** $d_y(Q)$ **do**
4: $\quad G_i = \gcd(G_{i-1}, sres_{y,i}(P, Q))$
5: $\quad A_i = G_{i-1}/G_i$
6: $\quad$ if $d_x(A_i) > 0$, add (A_i, B_i) to $\mathcal{T}$
7: **return** $\mathcal{T} = \{(A_i(x), B_i(x, y))\}_{i\in\mathcal{I}}$

linear form for an arbitrary zero-dimensional system $\{P, Q\}$ in the same complexity.

6.1 Number of critical points

Our algorithm for computing the number of (complex) critical points of a curve is based on a classical algorithm for computing a triangular decomposition of a system of two bivariate polynomials. We first recall this algorithm and then show how it can be slightly modified and used to compute the number of critical points of a curve.

Triangular decomposition. Let P and Q be two polynomials in $\mathbb{F}[x, y]$ of degree at most d. A decomposition of the system $\{P, Q\}$ using the subresultant sequence appears in the theory of triangular sets [10] and for the computation of the topology of curves [9].

The idea is based on Lemma 2 which states that, after specialization at $x = \alpha$, the first (with respect to increasing i) nonzero subresultant $Sres_{y,i}(P, Q)(\alpha, y)$ is of degree i and is equal to the gcd of $P(\alpha, y)$ and $Q(\alpha, y)$. This induces a decomposition into triangular subsystems ($\{A_i(x)$, $Sres_{y,i}(P, Q)(x, y)\}$) where a solution α of $A_i(x) = 0$ is such that the system $\{P(\alpha, y), Q(\alpha, y)\}$ admits exactly i roots (counted with multiplicity), which are exactly those of $Sres_{y,i}(P, Q)(\alpha, y)$. Furthermore, these triangular subsystems are regular chains, i.e., the leading coefficient of the bivariate polynomial (seen in y) is coprime with the univariate polynomial. For clarity and self-containedness, we recall this decomposition in Algorithm 2. Note that this algorithm performs $\widetilde{O}(d^4)$ arithmetic operations in $\mathbb{F}$ (see e.g. [5, Lemma 15]). We also state the following properties which directly follow from the algorithm and Lemma 2.

LEMMA 8 ([9, 10]). *Algorithm 2 computes a triangular decomposition $\{(A_i(x), B_i(x, y))\}_{i\in\mathcal{I}}$ such that*

- *the set of solutions of $\{P, Q\}$ is the disjoint union of the sets of solutions of the $\{A_i(x), B_i(x, y)\}$, $i \in \mathcal{I}$*
- *$\prod_{i\in\mathcal{I}} A_i$ is squarefree,*
- *for any root α of A_i, $B_i(\alpha, y)$ is of degree i and is equal to $\gcd(P(\alpha, y), Q(\alpha, y))$.*

Degree of the triangular decomposition. We call the *degree of the triangular decomposition of $\{P, Q\}$*, the sum of the degrees of the triangular systems computed by Algorithm 2, that is,

$$\sum_{i\in\mathcal{I}} \deg_x(A_i(x))\, \deg_y(B_i(x, y))$$

Algorithm 3 Degree of the triangular decomposition

Input: P, Q in $\mathbb{F}[x, y]$ coprime such that $Lc_y(P)$ and $Lc_y(Q)$ are coprime, $d_y(Q) \leqslant d_y(P)$
Output: The degree of the triangular decomposition of $\{P, Q\}$
1: Compute the principal subresultant sequence of P and Q with respect to y: $sres_{y,i}(P, Q)$
2: $G_0 =$ squarefree part$(Res_y(P, Q))$
3: **for** $i = 1$ **to** $d_y(Q)$ **do**
4: $\quad G_i = \gcd(G_{i-1}, sres_{y,i}(P, Q))$
5: **return** $\sum_{i\in\mathcal{I}}(\deg(G_{i-1}) - \deg(G_i))\, i$

where $\deg_x$ refers to the degree of the polynomial with respect to x and similarly for y. As we will see below, we only need the degree of the triangular decomposition of some systems for computing the number of critical points of H.

We present in Algorithm 3 a slight variation of the triangular decomposition algorithm in which we only compute the degree of the decomposition. Instead of computing the subresultant sequence $Sres_{y,i}(P, Q)$ of P and Q as in Algorithm 2, we only compute the sequence of principal subresultant coefficients of P and Q (that is, the sequence of coefficients of the monomials of degree i in y in $Sres_{y,i}(P, Q)$), which is sufficient for computing the degree of the decomposition. As we will see, this decreases by a factor d the arithmetic complexity in $\mathbb{F}$ of the algorithm, which is critical for our global algorithm.[3]

LEMMA 9. *Algorithm 3 computes the degree of the triangular decomposition of $\{P, Q\}$. If $P, Q \in \mathbb{F}[x, y]$ have degree at most d, the algorithm performs $\widetilde{O}(d^3)$ arithmetic operations in $\mathbb{F}$. If $P, Q \in \mathbb{Z}[x, y]$ ($\subset \mathbb{Q}[x, y]$) have degree at most d and bitsize at most τ, the algorithm performs $\widetilde{O}_B(d^7 + d^6\tau)$ bit operations in $\mathbb{Z}$.*

PROOF. The correctness of Algorithm 3 directly follows from Lemma 8. Concerning the complexity, the resultant and the sequence of the principal subresultant coefficients of P and Q can be computed in $\widetilde{O}(d^3)$ arithmetic operations, and each of these principal subresultants (including the resultant) has degree in $O(d^2)$, by Lemma 3 (note that this lemma is stated for the coefficient ring $\mathbb{Z}$, but the arithmetic complexity is the same for any field $\mathbb{F}$). The algorithm performs at most d gcd computations between these polynomials. The arithmetic complexity of one such gcd computation is softly linear in their degrees, that is $\widetilde{O}(d^2)$ (Lemma 4). Hence the complexity of computing all the gcds is in $\widetilde{O}(d^3)$. The bit complexity over $\mathbb{Z}$ Algorithm 3 is bounded by that of Algorithm 2 which is in $\widetilde{O}_B(d^7 + d^6\tau)$ according to the proof of [6, Thm. 19].[4] □

LEMMA 10. *The degree of the triangular decomposition of $\{P, Q\}$ is equal to the sum, over all distinct solutions (α, β) of $\{P, Q\}$, of the multiplicities of β in $\gcd(P(\alpha, y), Q(\alpha, y))$.*

[3] Note that, while this complexity improvement does not impact the bit complexity of computing the number of critical points of a curve H over $\mathbb{Z}$, it is critical when computing a lucky prime for $\{H, \frac{\partial H}{\partial y}\}$ where the number of critical points is computed for $O(d^4 + d^3\tau)$ systems defined over distinct $\mathbb{Z}_\mu$ (Proposition 12).

[4] Note that this bound is not an obvious overestimate because known bounds yield a complexity of $\widetilde{O}_B(d^7 + d^6\tau)$ for all the gcd computations in Line 4 of Algorithm 2, which is the same for Line 4 of Algorithm 3.

Algorithm 4 Number of critical points of H

Input: H in $\mathbb{F}[x,y]$ squarefree such that $Lc_y(H) \in \mathbb{F}$
Output: The number of critical points of H

1: **return** Algo 3 $(H, (\frac{\partial H}{\partial y})^2)$ - Algo 3 $(H, \frac{\partial H}{\partial y})$

PROOF. By Lemma 8, the sets of solutions of the systems of the triangular decomposition of Algorithm 2 are disjoint and polynomials A_i are squarefree. The degree of the triangular decomposition of $\{P,Q\}$ is thus

$$\sum_{i\in\mathcal{I}} \deg_x(A_i(x))\ \deg_y(B_i(x,y)) = \sum_{(\alpha,\beta)\in V} \mathrm{mult}(\beta, B_i(\alpha,y)),$$

where V is the set of solutions of $\{P,Q\}$ and $\mathrm{mult}(\beta, B_i(\alpha,y))$ denotes the multiplicity of β in $B_i(\alpha,y)$. The result follows since $B_i(\alpha,y) = \gcd(P(\alpha,y), Q(\alpha,y))$ by Lemma 8. □

Number of critical points of H. Algorithm 4 computes the number of critical points of H as the difference between the degree of the triangular decompositions of the systems $\{H, (\frac{\partial H}{\partial y})^2\}$ and $\{H, \frac{\partial H}{\partial y}\}$. We first prove the correctness of this algorithm and then its complexity.

PROPOSITION 11. *Algorithm 4 computes the number of critical points of H. If $H \in \mathbb{F}[x,y]$ has degree d, the algorithm performs $\widetilde{O}(d^3)$ arithmetic operations in $\mathbb{F}$. If $H \in \mathbb{Z}[x,y]$ $(\subset \mathbb{Q}[x,y])$ has degree d and bitsize τ, the algorithm performs $\widetilde{O}_B(d^7 + d^6\tau)$ bit operations in $\mathbb{Z}$.*

PROOF. We first prove that for any critical point (α,β) of H, the multiplicity of β in $\gcd(H(\alpha,y), (\frac{\partial H}{\partial y})^2(\alpha,y))$ is greater by one than the multiplicity of β in $\gcd(H(\alpha,y), \frac{\partial H}{\partial y}(\alpha,y))$. Since (α,β) is a critical point of H, it is solution of both the systems $\{H, \frac{\partial H}{\partial y}\}$ and $\{H, (\frac{\partial H}{\partial y})^2\}$. This implies that β is a root of both $\gcd(H(\alpha,y), \frac{\partial H}{\partial y}(\alpha,y))$ and $\gcd(H(\alpha,y), (\frac{\partial H}{\partial y})^2(\alpha,y))$. If m is the multiplicity of β in $H(\alpha,y)$ then β has multiplicity $m-1$ in $\frac{\partial H}{\partial y}(\alpha,y)$ and thus, that it has multiplicity $2m-2$ in $(\frac{\partial H}{\partial y})^2$. It follows that β has multiplicity $m-1$ in $\gcd(H(\alpha,y), \frac{\partial H}{\partial y}(\alpha,y))$ and m in $\gcd(H(\alpha,y), (\frac{\partial H}{\partial y})^2(\alpha,y))$ because $m \leqslant 2m-2$, that is $m-1 \geqslant 1$, since β is solution of $\frac{\partial H}{\partial y}(\alpha,y)$.

We denote the multiplicity of β in $\gcd(P(\alpha,y), Q(\alpha,y))$ as $\mathrm{mult}(\beta, \gcd(P(\alpha,y), Q(\alpha,y)))$. Summing over all the critical points of H and noticing that the set V_H of distinct solutions of $\{H, \frac{\partial H}{\partial y}\}$ is the same as that of $\{H, (\frac{\partial H}{\partial y})^2\}$, we obtain that the number of critical points is

$$\#V_H = \sum_{(\alpha,\beta)\in V_H} \mathrm{mult}(\beta, \gcd(H(\alpha,y), (\frac{\partial H}{\partial y})^2(\alpha,y))) - \sum_{(\alpha,\beta)\in V_H} \mathrm{mult}(\beta, \gcd(H(\alpha,y), \frac{\partial H}{\partial y}(\alpha,y))),$$

which is equal, by Lemma 10, to the difference of the degrees of the decompositions of $\{H, (\frac{\partial H}{\partial y})^2\}$ and $\{H, \frac{\partial H}{\partial y}\}$. These degrees are computed by Algorithm 3, which concludes the proof of correctness of Algorithm 4.

The complexity analysis of the algorithm directly follows from Lemma 9 noticing that $\frac{\partial H}{\partial y}$ and $(\frac{\partial H}{\partial y})^2$ have degrees at most $2d$ (and bitsizes in $O(d+\tau)$ when defined over $\mathbb{Z}$) and that $(\frac{\partial H}{\partial y})^2$ can be computed from $\frac{\partial H}{\partial y}$ in complexity $\widetilde{O}(d^2)$ (and $\widetilde{O}_B(d^2\tau)$ when defined over $\mathbb{Z}$) [15, Cor. 8.28]. □

Algorithm 5 Lucky prime for $\{H, \frac{\partial H}{\partial y}\}$

Input: H in $\mathbb{Z}[X,Y]$ such that $Lc_y(H) \in \mathbb{Z}$
Output: A lucky prime μ for the system $\{H, \frac{\partial H}{\partial y}\}$

1: N= Algorithm 4 (H)
2: Compute $H(t-sy, y)$ and $\frac{\partial H}{\partial y}(t-sy, y)$
3: $m = 2d^4$
4: **while** true **do**
5: Compute the set B of the first $d^4 + d^3\tau$ primes $> m$
6: **for all** μ in B **do**
7: Compute the reduction mod. μ of H, $\frac{\partial H}{\partial y}$, L_H, $L_{\frac{\partial H}{\partial y}}$
8: **if** $\phi_\mu(L_H(s))\ \phi_\mu(L_{\frac{\partial H}{\partial y}}(s)) \not\equiv 0$ **then**
9: Compute N_μ = Algorithm 4$(\phi_\mu(H), \phi_\mu(\frac{\partial H}{\partial y}))$
10: **if** $N_\mu = N$ **then**
11: **return** μ
12: m = the largest prime in B

6.2 Lucky prime

In Algorithm 5, we compute a lucky prime for $\{H, \frac{\partial H}{\partial y}\}$ (see Definition 5) in a straightforward manner by first computing the number of distinct solutions of the system and then by computing the number of solutions of its image modulo distinct prime numbers μ until the same number of solutions is found (and checking that some leading coefficients do not vanish modulo μ). Note that Algorithm 5 is a simplified variant of [5, Algorithm 3] where we use here the knowledge of the number of critical points of H to avoid computing an explicit bound on the number of unlucky primes.

PROPOSITION 12. *Given $H \in \mathbb{Z}[x,y]$ of degree d and bitsize τ, Algorithm 5 computes a lucky prime for $\{H, \frac{\partial H}{\partial y}\}$ using $\widetilde{O}_B(d^7 + d^6\tau)$ bit operations.*

PROOF. The correctness of Algorithm 5 follows directly from the fact that the number of unlucky primes is finite (see [5, Prop. 13]).

We now analyze the complexity of the algorithm. Computing the number of critical points of H in Line 1 has complexity $\widetilde{O}_B(d^7 + d^6\tau)$ by Proposition 11. It is straightforward that the computations in Line 2 can be done in bit complexity $\widetilde{O}_B(d^4 + d^3\tau)$ (see e.g. [5, Lemma 7]). There are $O(\log d\tau)$ iterations of the loop in Line 4 because there are $\widetilde{O}(d^4 + d^3\tau)$ unlucky primes [5, Prop. 13]. Each iteration of this loop consists in testing, for the $d^4 + d^3\tau$ primes in B, the non-vanishing of the reduction of the two polynomials $L_H(s)$ and $L_{\frac{\partial H}{\partial y}}(s)$ and the equality between the number of solution over $\mathbb{Z}$ and its analogue over $\mathbb{Z}_\mu$.

Polynomials H, $\frac{\partial H}{\partial y}$, L_H and $L_{\frac{\partial H}{\partial y}}$ are of degree at most d in one or two variables and they have bitsize at most $\widetilde{O}(d+\tau)$ (see e.g. [5, Lemma 7]). The reduction of all their $O(d^2)$ coefficients modulo all the primes in B can be computed via a remainder tree in a bit complexity that is soft linear in the total bitsize of the input [12, Thm. 1], which is dominated by the sum of the bitsizes of the $d^4+d^3\tau$ primes in B each being of bitsize $O(\log d\tau)$ (since there are $O(\log d\tau)$ iterations of the loop in Line 4). Hence, the bit complexity of Line 7 is $\widetilde{O}_B(d^4 + d^3\tau)$.

Finally, the arithmetic complexity of Algorithm 4 is in $\widetilde{O}(d^3)$, by Lemma 11, thus its bit complexity is also in $\widetilde{O}_B(d^3)$ since $\mu \in O(\log d\tau)$. Hence, the total bit complexity of Line 9 is $\widetilde{O}_B(d^7+d^6\tau)$, and so is the bit complexity of one iteration of the loop in Line 4. Since at most $O(\log d\tau)$ iterations are performed, this yields an overall bit complexity for Algorithm 5 in $\widetilde{O}_B(d^7+d^6\tau)$. □

7. WRAP UP

The results of the previous sections can easily be combined in the following theorem.

THEOREM 13. *Let $P, Q \in \mathbb{Z}[x,y]$ of total degree at most d and maximum bitsize τ. A separating linear form $x+ay$ for $\{P,Q\}$ with a an integer of bitsize in $O(\log d)$ can be computed using $\widetilde{O}_B(d^7+d^6\tau)$ bit operations.*

PROOF. According to Lemma 7, we can compute in complexity $\widetilde{O}_B(d^6+d^5\tau)$ a shearing $t = x+\alpha y$, $\alpha \in O(d)$, and a polynomial $H \in \mathbb{Z}[t,y]$ of total degree at most $2d$ and bitsize $\widetilde{O}(d+\tau)$ such that $\{H, \frac{\partial H}{\partial y}\}$ is zero-dimensional and such that $x+(\alpha+a)y$ is separating for $\{P,Q\}$ if $t+ay$ is separating for $\{H, \frac{\partial H}{\partial y}\}$. The result follows since by Propositions 6, 11 and 12, since an integer a in $O(d^4)$ such that $t+ay$ separates $\{H, \frac{\partial H}{\partial y}\}$ can be computed using $\widetilde{O}_B(d^7+d^6\tau)$ bit operations. □

8. LAS-VEGAS ALGORITHM

In this section, we present a Las-Vegas version of the algorithm presented in the previous sections, whose expected bit complexity is $\widetilde{O}_B(d^5+d^4\tau)$ (Theorem 19).

The Las-Vegas version of our algorithm is the same as the deterministic one except that we use Las-Vegas algorithms for gcd computations and that we choose randomly candidates for a separating linear form and a lucky prime in Algorithms 1 and 5.

More precisely, in the Las-Vegas version of Algorithm 1, the separating linear form is computed by choosing at random an integer a in $[0, 4d^4]$ until a candidate satisfying the condition of Line 9 is found. There are at most $2d^4$ integers that do not satisfy this condition,[5] thus a good candidate is chosen with probability at least $\frac{1}{2}$, and so at most 2 candidates are chosen on average.

In the Las-Vegas version of Algorithm 5, we first compute a set B of $2m$ prime numbers where m is an upper bound on the number of unlucky primes for $\{H, \frac{\partial H}{\partial y}\}$; such a set B can be computed in bit complexity $\widetilde{O}_B(d^4+d^3\tau)$ (see the proof of [5, Lemma 18]). Then, we iteratively choose at random a prime number μ in B until the conditions of Algorithm 5 are satisfied (Lines 8 and 10). The primes not satisfying these conditions are the unlucky ones, by definition, thus a lucky prime is found with probability at least $\frac{1}{2}$, and so at most 2 candidates are chosen on average.

It remains to prove that the expected bit complexity of Algorithms 1, 4, and 5, as well as the initial shearing of the coordinate systems, are in $\widetilde{O}_B(d^5+d^4\tau)$. Our analysis is based on the following result on the expected complexity of gcd computations.

LEMMA 14 ([15, COR. 11.11]). *Let $f, g \in \mathbb{Z}[x]$ of degree at most d and maximum bitsize τ. The* gcd *of f and g can be computed using an expected number of $\widetilde{O}_B(d^2+d\tau)$ bit operations.*

LEMMA 15. *Given $P, Q \in \mathbb{Z}[x,y]$ of total degree at most d and maximum bitsize τ, the Las-Vegas version of Algorithm 1 computes a separating linear form $x+ay$ for $\{P,Q\}$ with $a < 2d^4$ with an expected bit complexity in $\widetilde{O}_B(d^4+d^3\tau)$.*

PROOF. In the proof of Proposition 6, we proved that the complexity of the algorithm is $\widetilde{O}_B(d^4+d^3\tau)$ plus $\widetilde{O}_B(d^3)$ times the number of considered choices of integer a. As argued above, at most two candidate integers are considered on average, which yields the lemma. □

LEMMA 16. *Given $P, Q \in \mathbb{Z}[x,y]$ of total degree at most d and maximum bitsize τ, Algorithm 3 computes the degree of the triangular decomposition of $\{P,Q\}$ with an expected bit complexity in $\widetilde{O}_B(d^5+d^4\tau)$.*

PROOF. According to Lemma 3, the sequence of the principal subresultant coefficients $sres_{i,y}(P,Q)$, $i = 0, \ldots, d$ can be computed in $\widetilde{O}_B(d^4\tau)$ bit operations, and each of these principal subresultants (including the resultant) has degree $O(d^2)$ and bitsize $\widetilde{O}(d\tau)$. The algorithm then performs at most d gcd computations between these polynomials (including the computation of the squarefree part of the resultant). By Lemma 14 and using Mignotte's bound [2, Cor. 10.12], each of these gcds can be computed in an expected bit complexity $\widetilde{O}_B(d^4+d^3\tau)$. Hence computing d such gcds can be done with expected bit complexity $\widetilde{O}_B(d^5+d^4\tau)$, which concludes the proof. □

LEMMA 17. *Given $P, Q \in \mathbb{Z}[x,y]$ of total degree at most d and maximum bitsize τ, Algorithm 4 computes the number of critical points of H with an expected bit complexity in $\widetilde{O}_B(d^5+d^4\tau)$.*

PROOF. As seen in the proof of Proposition 11, Algorithm 4 computes $\frac{\partial H}{\partial y}$ and $(\frac{\partial H}{\partial y})^2$ in bit complexity $\widetilde{O}_B(d^2\tau)$ and calls Algorithm 3 on two systems of degrees $O(d)$ and bitsizes $O(d+\tau)$. The result follows from Lemma 16. □

LEMMA 18. *Given $H \in \mathbb{Z}[x,y]$ of total degree d and bitsize τ, the Las-Vegas version of Algorithm 5 computes a lucky prime for $\{H, \frac{\partial H}{\partial y}\}$ with an expected bit complexity in $\widetilde{O}_B(d^5+d^4\tau)$.*

PROOF. By Lemma 17, the first call to Algorithm 4 has an expected bit complexity in $\widetilde{O}_B(d^5+d^4\tau)$. As shown in the proof of Proposition 12, the bit complexity of shearing H and $\frac{\partial H}{\partial y}$, as well as the reductions modulo μ have bit complexity $\widetilde{O}_B(d^4+d^3\tau)$. Finally, by Proposition 11, the calls in $\mathbb{Z}_\mu$ to Algorithm 4 have arithmetic complexity $\widetilde{O}(d^3)$ and thus bit complexity $\widetilde{O}_B(d^3)$ (since $\mu \in O(\log d\tau)$). This concludes the proof since, as discussed above, the expected number of such calls is at most 2. □

Combining the above results, we obtain the following theorem.

[5] Indeed, Υ is of degree at most $2d$ and the system $\{P_\mu, Q_\mu\}$ has at most d^2 solutions which define at most $\binom{d^2}{2}$ directions in which two solutions are aligned. Furthermore $2d + \binom{d^2}{2} < 2d^4$ (for $d \geqslant 2$).

THEOREM 19. *Let $P, Q \in \mathbb{Z}[x, y]$ of total degree at most d and maximum bitsize τ. A separating linear form $x + ay$ for $\{P, Q\}$ with a an integer of bitsize in $O(\log d)$ can be computed using an expected number of $\widetilde{O}_B(d^5 + d^4\tau)$ bit operations.*

PROOF. As in the proof of Theorem 13, Lemmas 15, 17 and 18 yield the result once we prove, as in Lemma 7, that we can compute with the right complexity a shearing $t = x + \alpha y$, $\alpha \in O(d)$, and a polynomial $H \in \mathbb{Z}[t, y]$ of total degree at most $2d$ and bitsize $\widetilde{O}(d + \tau)$ such that $\{H, \frac{\partial H}{\partial y}\}$ is zero-dimensional and such that, if $t + ay$ is separating for $\{H, \frac{\partial H}{\partial y}\}$, then $x + (\alpha + a)y$ is separating for $\{P, Q\}$.

According to the proof of Lemma 7, we only need to prove that the computation of the squarefree part of $H = P\,Q$ can be done with an expected bit complexity in $\widetilde{O}_B(d^5 + d^4\tau)$. Replacing in the proof of [11, Lemma 13] the bit complexity of computing a univariate gcd by the one in Lemma 14 yields the result. □

9. CONCLUSION

This paper focuses on the computation of separating linear forms for bivariate systems. First, we proved that the computation of such a separating form can be done with a bit complexity $\widetilde{O}_B(d^7 + d^6\tau)$ in the worst case. As mentioned in the introduction, this result directly yields, within the same worst-case bit complexity, the rational parameterization of Gonzalez-Vega et al. [9, 6] and that of Rouillier [14, 5]. Second, we proved that the computation of a separating linear form can be done in a Las-Vegas setting using an expected number of $\widetilde{O}_B(d^5 + d^4\tau)$ bit operations. As a consequence, the computation in this setting of a separating linear form now becomes non-dominant in the whole process of computing a rational parameterization; indeed, given a separating linear form, computing Gonzalez-Vega et al. and Rouillier's parameterizations both have bit complexity in $\widetilde{O}_B(d^7 + d^6\tau)$ even in the Las-Vegas setting.

It should be mentioned that the best known upper bound for the total bitsize of the parameterization of Gonzalez-Vega et al. is $\widetilde{O}(d^5 + d^4\tau)$.[6] Thus, some progress on this upper bound would be required before any further progress on the computation of a separating linear form in the Las-Vegas setting could impact that of computing this parameterization. However, note that the situation is slightly different for the Rational Univariate Representation (RUR) of Rouillier [14] whose total bitsize is $\widetilde{O}(d^4 + d^3\tau)$ [5, Theorem 22].

Finally, we note that, for computing a separating linear form of an *arbitrary* system $\{P, Q\}$, the algorithm presented here is likely purely theoretical because considering the system $\{PQ, \frac{\partial PQ}{\partial y}\}$ instead $\{P, Q\}$ essentially doubles the degree of the input polynomials, which is likely not efficient in practice. However, for the problem of computing the critical points of a curve, there is some good hope that our algorithm is efficient in practice.

[6] Indeed, the approach of Gonzalez-Vega et al. first applies a linear change of variables to the input polynomials, which increases the bitsize of the polynomials to $\tau' \in \widetilde{O}(d + \tau)$, and then computes rational parameterizations of the solutions of the $O(d)$ systems of the triangular decomposition (Algorithm 2). The rational parameterizations are ratios of coefficients of the polynomial subresultants (seen as polynomials in y) which have degrees $O(d^2)$ and bitsize $\widetilde{O}(d\tau') = \widetilde{O}(d^2 + d\tau)$ (Lemma 3). The total bitsize of the $O(d)$ parameterizations is thus $\widetilde{O}(d^5 + d^4\tau)$.

10. REFERENCES

[1] M.-E. Alonso, E. Becker, M.-F. Roy, and T. Wörmann. Multiplicities and idempotents for zerodimensional systems. In *Algorithms in Algebraic Geometry and Applications*, volume 143 of *Progress in Mathematics*, pages 1–20. Birkhäuser, 1996.

[2] S. Basu, R. Pollack, and M.-F. Roy. *Algorithms in Real Algebraic Geometry*, volume 10 of *Algorithms and Computation in Mathematics*. Springer-Verlag, 2nd edition, 2006.

[3] A. Bostan, B. Salvy, and É. Schost. Fast algorithms for zero-dimensional polynomial systems using duality. *Applicable Algebra in Engineering, Communication and Computing*, 14(4):239–272, 2003.

[4] Y. Bouzidi. *Solving bivariates algebraic systems and topology of plane curves*. PhD thesis, Université de Lorraine, March 2014.

[5] Y. Bouzidi, S. Lazard, M. Pouget, and F. Rouillier. Separating linear forms and rational univariate representations of bivariate systems. *J. Symb. Comput.*, 2014. To appear.

[6] D. I. Diochnos, I. Z. Emiris, and E. P. Tsigaridas. On the asymptotic and practical complexity of solving bivariate systems over the reals. *J. Symb. Comput.*, 44(7):818–835, 2009.

[7] M. El Kahoui. An elementary approach to subresultants theory. *J. Symb. Comput.*, 35(3):281–292, 2003.

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

[9] L. González-Vega and M. El Kahoui. An improved upper complexity bound for the topology computation of a real algebraic plane curve. *J. of Complexity*, 12(4):527–544, 1996.

[10] X. Li, M. Moreno Maza, R. Rasheed, and É. Schost. The modpn library: Bringing fast polynomial arithmetic into maple. *J. Symb. Comput.*, 46(7):841–858, 2011.

[11] K. Mehlhorn, M. Sagraloff, and P. Wang. From approximate factorization to root isolation with application to cylindrical algebraic decomposition. *CoRR*, abs/1301.4870, 2013.

[12] R. Moenck and A. Borodin. Fast modular transforms. *Journal of Computer and System Sciences*, 8, 1974.

[13] D. Reischert. Asymptotically fast computation of subresultants. In *Proceedings of the 10th international Symposium on Symbolic and Algebraic Computation*, ISSAC'97, pages 233–240, 1997.

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

[15] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge Univ. Press, Cambridge, U.K., 2nd edition, 2003.