Change Of Order For Bivariate Triangular Sets

Cyril Pascal
École polytechnique
91128 Palaiseau, France
Cyril.Pascal@m4x.org

Éric Schost
LIX, École polytechnique
91128 Palaiseau, France
Eric.Schost@polytechnique.fr

ABSTRACT

Changing the order of variables in bivariate triangular sets has applications in Trager's factorization algorithm, or in rational function integration. We discuss the complexity of this question, using baby steps / giant steps techniques and trace formulas, obtaining subquadratic estimates.

Categories and Subject Descriptors:
I.1.2 [**Computing Methodologies**]: Symbolic and Algebraic Manipulation – *Algebraic Algorithms*

General Terms:
Algorithms, Experimentation, Theory

Keywords:
Triangular sets, Change of order

1. INTRODUCTION

The purpose of this paper is to discuss the complexity of *change of order* for triangular sets in dimension zero. We restrict the discussion to *bivariate systems*: the relevant ideas already appear in this situation, the notational difficulties are minimized, and many applications are covered.

Let thus k be a field, which we assume to be perfect. A *triangular set* for the order $Y > X$ is a pair of polynomials $(S(X), T(X,Y))$ in $k[X,Y]$, with T *monic* in Y, of degree in X less than that of S, and such that the ideal (S,T) is radical. A triangular set for the order $X > Y$ is defined similarly, exchanging the roles of X and Y.

Given a triangular set (S,T) for the order $Y > X$, we wish to compute triangular sets (U_j, V_j) for the order $X > Y$, that is, polynomials of the form

$$\left| \begin{array}{l} V_1(Y,X) \\ U_1(Y) \end{array} \right. , \left| \begin{array}{l} V_2(Y,X) \\ U_2(Y) \end{array} \right. \cdots \left| \begin{array}{l} V_n(Y,X) \\ U_n(Y) \end{array} \right. \quad (1)$$

which describe the same set of points. Technically, we ask that the product of all ideals (U_j, V_j) equals (S,T). Such triangular sets are not unique, but a canonical choice exists, see Subsection 3.1. Our monicity requirement is the reason why in general, several triangular sets may be needed in the output: the ideal generated by the polynomials $X^2 - X$ and $Y^2 - (2X+1)Y + 2X$ cannot be generated by a single triangular set (U,V) for the order $X > Y$.

ISSAC'06, July 9–12, 2006, Genova, Italy.

Complexity measures. Our goal is to give complexity estimates for computing polynomials (U_j, V_j) as above. To state these results, we denote by $\mathsf{M} : \mathbb{N} \to \mathbb{N}$ a function such that over any ring R, degree d polynomials in $R[X]$ can be multiplied in $\mathsf{M}(d)$ operations in R. Furthermore, we make the super-additivity assumptions of [19, Ch. 8]. Using the results of [9], one can take $\mathsf{M}(d) \in O(d \log(d) \log\log(d))$.

We let $\omega \in [2,3]$ be such that $n \times n$ matrices over k can be multiplied in $O(n^\omega)$ operations in k. Since it is used repeatedly in the sequel, the quantity $\frac{\omega-1}{2} \in [\frac{1}{2}, 1]$ is denoted by η. Using the results of [11], one can take $\omega \leq 2.38$, and thus $\eta \leq 0.69$ and $\eta + 1 \leq 1.69$.

We write $f \in O\tilde{}(g)$ if f is in $O(g \log^\alpha(g))$ for some α. Thus, $\mathsf{M}(d)$ is in $O\tilde{}(d)$.

Main results. In what follows, we write $s = \deg_X S$ and $t = \deg_Y T$ (so that optimal algorithms would have complexity linear in st). We make the following assumption, which is used below to apply some exponentiation techniques:

(H) : $2, \ldots, st$ are units in k, *i.e.*, $\mathrm{char}(k) > st$.

Our results cover several cases, from the particular to the general. In generic situations, the output consists in a single pair of polynomials (U,V); our first statement applies to this case. Our second result assumes that T is linear in Y; in the general case, we rely on probabilistic techniques.

THEOREM 1. *Suppose that the ideal (S,T) can be generated by a* single *triangular set (U,V) for the order $X > Y$. Then (U,V) can be computed in $O(\mathsf{M}(st)(s^\eta + \log(t))) \subset O\tilde{}(s^{\eta+1}t)$ operations in k.*

THEOREM 2. *Suppose that $t = \deg_Y T = 1$. Then polynomials (U_j, V_j) as in Equation* (1) *can be computed using $O(s^\eta \mathsf{M}(s) \log(s)) \subset O\tilde{}(s^{\eta+1})$ operations in k.*

THEOREM 3. *One can compute polynomials (U_j, V_j) as in Equation* (1) *by a probabilistic algorithm with the following characteristics. The algorithm chooses $a, b \in k$; there exists a non-zero polynomial $\Delta \in k[\mathbf{a}, \mathbf{b}]$ of degree at most $st(st-1)$, such that if $\Delta(a,b) \neq 0$, the algorithm succeeds, with a complexity in $O((st)^\eta \mathsf{M}(st) \log(st)) \subset O\tilde{}((st)^{\eta+1})$.*

Using the Zippel-Schwartz lemma [19, Lem. 6.44], one can deduce estimates on the probability of success, if $|k|$ is large enough. Furthermore, failure in the algorithm can be detected, so Las-Vegas type estimates could be deduced.

Previous work. A more general approach of the conversion from any Gröbner basis is described in [25], with cost cubic

in *st*. `Palgie` [6] solves our question as well, under a primality assumption that can likely be lifted in our case, but no complexity statement is given. The FGLM algorithm [18] performs this operation, when the output consists in a single triangular set, with a cubic complexity upper bound.

The problem of changing order actually admits a well-known solution in the bivariate case, essentially described in [38]. The first step consists in computing the resultant U of S and T with respect to X; the squarefree factors of U are the polynomials U_i introduced above. To recover the algebraic dependence of X in terms of Y, one computes the gcd of S and T modulo each U_i.

The best known algorithms for such resultant computations have complexity in $O^{\tilde{}}(s^2t)$, see [19, Ch. 11]. The GCD computation admits similar upper bounds: if all U_i are irreducible, the analysis of [19, Th. 22.10] readily yields this estimate; otherwise, using the analysis done in [14] of the D5 principle [15, 21] leads to the same bound.

In the general case, these estimates have to be compared with the estimates of $O^{\tilde{}}((st)^{\eta+1})$ of Theorem 3, which indicates that a threshold should exist. In the particular cases of Theorem 1 and 2, however, our estimates are better. For instance, in the important case when T is linear in Y, *i.e.* when $t = 1$, the previous results were of *quadratic* complexity in s, whereas that of Theorem 2 becomes *subquadratic*.

The basic contribution of this paper consists in bringing together several ideas already present in the literature.

The bases of this work are baby steps / giant steps techniques for "power projections", which can be traced back to work of Kaltofen [22] and Shoup [34]. More recent articles discuss the specific problem of *bivariate* power projection that we address below: the article [37] makes no use of fast matrix multiplication, the article [23] gives no complexity estimate, and the algorithm of [3] treats only a particular case. Our novelty is the use of polynomial matrix multiplication for power projection; this idea was used before by Nüsken and Ziegler for the similar (dual) problem of bivariate modular composition, in the case where $t = 1$ [27].

We use trace formulas to perform the change of order. This idea appears in [29], in the special case when Y is a "separating element", and in [17] for the general case. Baby steps / giant steps techniques are not used in these references, but the input of these algorithms is more general than ours; those techniques are then used in [5, 30]. Last, the geometric point-of-view (equiprojectable decomposition) used in Section 3 is introduced in [13], following [24, 10].

Applications. In this paper, we discuss only bivariate operations, having in mind the following applications.

Trager's factorization algorithm [38]. Let S be a irreducible polynomial in $k[X]$, and let $P = k[X]/S$. Trager's algorithm addresses the question of factoring a monic polynomial T in $P[Y]$, by reducing this question to univariate factorization over k. We present a variant of this algorithm, all of whose basic steps rely on the material presented above.

The polynomial T can be seen in $k[X, Y]$; after performing a random change of variable $Y \leftarrow Y + \alpha X$ to put Y in general position, one changes the order of X and Y in the system (S, T). Since we are in general position, the output consists in a single triangular set $(U(Y), V(Y, X))$, where additionally V has degree 1 in X. Theorem 1 can be applied to perform this change of order.

The next step consists in factoring U. Then, for each factor U_i of U, define $V_i = V \bmod U_i$. The final step of the algorithm consists in applying the *inverse change of order* to each system $(U_i(Y), V_i(Y, X))$, to restore the original order $Y > X$. Since each V_i has degree 1 in X, we can apply Theorem 2 to perform this task (exchanging the roles of X and Y in the statement of the theorem).

Trager's original algorithm uses resultant and GCD computations. Using fast resultant and GCD techniques, the cost of all conversions performed in [38] is seen to be in $O^{\tilde{}}(s^2t)$, with our usual notation $s = \deg_X S$ and $t = \deg_Y T$. For comparison's sake, applying Theorems 1 and 2 yields a complexity estimate in $O^{\tilde{}}((st)^{\eta+1})$ for our approach.

Rational function integration [28, 38, 26]. A closely related question is rational function integration. We address here *logarithmic integration*: let A, B be in $k[X]$, with A squarefree, and $\deg B < \deg A$; then, the antiderivative of $\frac{B}{A}$ can be expressed as a sum of logarithms.

More precisely, Rothstein and Trager showed the following (see also [19, Ch. 22]). Let C be $Y - (\frac{B}{A'} \bmod A)$ in $k[X, Y]$, and let $(U_j, V_j)_{i \le n}$ be the polynomials obtained after changing the order of X and Y in the system (A, C). Then the following formula holds:

$$\frac{B}{A} = \sum_{j \le n} \sum_{U_j(\beta)=0} \beta \, \frac{V_j'(\beta, X)}{V_j(\beta, X)},$$

which amounts (in characteristic zero) to

$$\int \frac{B}{A} = \sum_{j \le n} \sum_{U_j(\beta)=0} \beta \, \log(V_j(\beta, X)).$$

Since C is linear in Y, Theorem 2 yields the following.

Corollary 1. *The logarithmic part in rational function integration can be performed in* $O(s^\eta \mathsf{M}(s) \log(s)) \subset O^{\tilde{}}(s^{\eta+1})$ *operations in* k.

This subquadratic result improves [19, Th. 22.10], which gives a quadratic complexity estimate of $O(\mathsf{M}(s)^2 \log(s))$ (and requires an additional factorization).

Univariate factorization with symmetries. A final motivation for this work comes from a problem of univariate factorization, originating from the Schoof-Elkies-Atkin point counting algorithm (see [32] and references therein).

For simplicity, we present an analogous, simplified version of this problem: we are to factor the self-reciprocal polynomial $S = X^6 - 5X^5 + 6X^4 - 9X^3 + 6X^2 - 5X + 1$. The set of roots of S is globally invariant under the map $X \mapsto \frac{1}{X}$, and the function $X \mapsto X + \frac{1}{X}$ is invariant for this action. Hence, it is natural to introduce $T = Y - (X + 1/X \bmod S)$, which equals $Y - (X^5 - 5X^4 + 6X^3 - 9X^2 + 5X - 5)$. Now, change the order of X and Y in (S, T). We obtain

$$V = X^2 - YX + 1 \ , \quad U = Y^3 - 5Y^2 + 3Y + 1.$$

We factor U, obtaining for instance $U_1 = Y^2 - 4Y - 1$ as a factor, and we restore the initial order $Y > X$ in the system (V, U_1). This yields

$$\left| \begin{array}{l} T_1 = Y + X^3 - 4X^2 - 4 \\ S_1 = X^4 - 4X^3 + X^2 - 4X + 1, \end{array} \right.$$

from which we deduce the factor S_1 of S. Hence, through change of order, we were able to halve the degree of the polynomial to factor. In our initial question, we were given a polynomial that plays the role of $X + \frac{1}{X}$ here, from which we similarly could make a factorization problem simpler.

Acknowledgments. We wish to thank Erich Kaltofen and Greg Reid for helpful discussions, and the referees for their constructive remarks.

2. PRELIMINARIES

We first discuss basic operations on polynomials, such as multiplication or remaindering. The main result of this section is Proposition 3 of Subsection 2.2, which gives cost estimates for multiple evaluation of a linear form. Our first goal is to establish complexity results; space limits prevent us from providing pseudo-code for the algorithms we describe.

2.1 Polynomial operations

Notation. We introduce notation for polynomials with a given support. If M is a finite subset of $\mathbb{N}^2$, we let

$$k[X,Y]_M = \left\{ \sum_{(i,j)\in M} a_{(i,j)} X^i Y^j \mid a_{(i,j)} \in k \right\}.$$

Given a Laurent series $c = \sum c_{(i,j)} X^i Y^j$, we write

$$\mathsf{Trunc}_M(c) = \sum_{(i,j)\in M} c_{(i,j)} X^i Y^j \in k[X,Y]_M.$$

Our most useful supports will be *rectangular*. Given q, r and s, t in $\mathbb{N}$, we denote by $\mathsf{A}(q, r\,;\, s, t)$ the subset of $\mathbb{N}^2$ given by

$$\mathsf{A}(q, r\,;\, s, t) = \{(i,j) \in \mathbb{N}^2 \mid q \le i \le r,\ s \le j \le t\}.$$

When $q = s = 0$, we will write $\mathsf{A}(r,t) = \mathsf{A}(0, r\,;\, 0, t)$. Then, for $M = \mathsf{A}(r,t)$ and a in $k[X,Y]_M$, we write $\mathrm{rev}_M(a) = X^r Y^t a\,(1/X, 1/Y) \in k[X,Y]_M$. This extends the notation $\mathrm{rev}_d(b) = X^d b(1/X)$ in use for univariate polynomials $b \in k[X]$ (see [19, Ch. 9]), which we use below as well.

In terms of complexity, our basic measure is the cost of multiplication. Using Kronecker's substitution $Y \leftarrow X^{2r+1}$, we get the following easy result, also in Corollary 8.28 of [19].

PROPOSITION 1. *Let $M = \mathsf{A}(r,t)$. Given a, b in $k[X,Y]_M$, one can compute ab in $O(\mathsf{M}(rt)) \subset O^{\sim}(rt)$ operations in k.*

Finally, given $E, F \subset \mathbb{N}^2$, $E + F$ denotes their Minkowski sum, and $2E$ denotes the sum $E + E$.

Computing truncated products. We now discuss the complexity of computing several *truncated products* of bivariate polynomials. In the following subsection, we will need to solve the following technical problem. Let $N = \mathsf{A}(p,q) \subset \mathbb{N}^2$, for some $p, q \in \mathbb{N}$, let s, t, d be in $\mathbb{N}$, and define $F = \mathsf{A}(d, t-1) + \{(s-1, t-1)\}$, that is, $F = \mathsf{A}(s-1,\ s-1+d\,;\ t-1,\ 2t-2)$. Given S and Z in $k[X,Y]_N$, consider the truncated product $(S, Z) \mapsto \mathsf{Trunc}_F(SZ)$. Computing one such product is straightforward, and little can be gained over a naive approach. However, when several instances of this operation are performed, polynomial matrix multiplication can help.

PROPOSITION 2. *Let n be in $\mathbb{N}$, such that $(d+1)n \in O(s)$. Given $S_1, \ldots, S_n$ and $Z_1, \ldots, Z_n$ in $k[X,Y]_N$, one can compute all $\mathsf{Trunc}_F(S_i Z_j)$ using $O(n^{\omega-1}\mathsf{M}(sq)) \subset O^{\sim}(n^{\omega-1}sq)$ operations in k.*

PROOF. It is useful to define $e = d+1$, and to let $E \subset \mathbb{N}^2$ be $\mathsf{A}(d, q)$. Then, the polynomials S_i and Z_i in $k[X,Y]_N$ can be expressed as polynomials in X^e:

$$\begin{aligned} S_i &= S_{i,0} + S_{i,1}X^e + S_{i,2}X^{2e} + \cdots + S_{i,m}X^{me}, \\ Z_i &= Z_{i,0} + Z_{i,1}X^e + Z_{i,2}X^{2e} + \cdots + Z_{i,m}X^{me}, \end{aligned}$$

with all $S_{i,j}$ and $Z_{i,j}$ in $k[X,Y]_E$, and $m = \lfloor p/e \rfloor$. Next, writing $\alpha = \lfloor (s-1)/e \rfloor$, we define for $r \in \{-1, 0, 1\}$ the vectors with entries in $k[X,Y]_E$:

$$\mathbf{s}_{i,r} = [S_{i,0}, \ldots, S_{i,\alpha+r}] \quad \text{and} \quad \mathbf{z}_{i,r} = [Z_{i,\alpha+r}, \ldots, Z_{i,0}].$$

These definitions yield the following lemma.

LEMMA 1. *For all $1 \le i, j \le n$, $\mathsf{Trunc}_F(S_i Z_j)$ equals $\mathsf{Trunc}_F(\sum_{-1\le r\le 1} (\mathbf{s}_{i,r} \cdot \mathbf{z}^t_{j,r}) X^{(\alpha+r)e})$.*

PROOF. For all i, j, $S_i Z_j = \sum_{0\le c\le 2m} \sum_{a+b=c} S_{i,a} Z_{j,b} X^{ce}$. All monomials in a summand $S_{i,a} Z_{j,b} X^{ce}$ have degree in X between ce and $ce + 2d$. Now, the definitions of α and e show that $ce > (s-1) + d$ for $c \ge \alpha + 2$ and $ce + 2d < (s-1)$ for $c \le \alpha - 2$. Due to our truncation pattern, it thus suffices to consider $c \in \{\alpha - 1, \alpha, \alpha + 1\}$. □

We now conclude the proof of the proposition. For $r \in \{-1, 0, 1\}$, let $\mathbf{S}_r$ be the matrix with rows $\mathbf{s}_{i,r}$, $i \le n$, and let $\mathbf{Z}_r$ be the matrix with columns $\mathbf{z}^t_{i,r}$, $i \le n$. Then by the previous lemma, the products $\mathsf{Trunc}_F(S_i Z_j)$ are the entries of the polynomial matrix $\mathsf{Trunc}_F(\sum_{-1\le r\le 1} (\mathbf{S}_r \cdot \mathbf{Z}_r) X^{(\alpha+r)e})$, where truncation is applied entry-wise. The matrix $\mathbf{S}_r$ has n rows and $\alpha + r + 1 \in O(s/e)$ columns, and the matrix $\mathbf{Z}_r$ has $\alpha + r + 1 \in O(s/e)$ rows and n columns.

Since $en \in O(s)$, computing each product $\mathbf{S}_r \cdot \mathbf{Z}_r$ is reduced to perform $O(s/en)$ sums and products of $n \times n$ matrices with entries in $k[X,Y]_E$. Hence, the complexity is that of $O(n^\omega s/en) = O(n^{\omega-1} s/e)$ multiplication of polynomials in $k[X,Y]_E$, plus of some additions which have a negligible cost. By Proposition 1, one multiplication of polynomials in $k[X,Y]_E$ has cost in $O(\mathsf{M}(eq))$. Since $en \in O(s)$, e is in $O(s)$; hence, this cost is in $O(e\mathsf{M}(sq)/s)$, due to the assumptions of [19, Ch. 8] on the function M. Putting the previous estimates together concludes the proof. □

2.2 Modular operations

Let now $(S(X), T(X,Y))$ be a triangular set in $k[X,Y]$. The main purpose of this subsection is to prove the estimates of Proposition 3 below, on the cost of evaluating *linear forms* modulo (S,T) on suitable entries.

Let Q be the quotient $k[X,Y]/(S,T)$. Writing $s = \deg_X S$, $t = \deg_Y T$ and $M = \mathsf{A}(s-1, t-1)$, Q admits as a basis the set of monomials $k^M = \{x^i y^j \mid i \le s-1,\ j \le t-1\}$, where x and y are the images of X and Y modulo (S,T). The set of k-linear forms $Q \to k$ forms the *dual space* $Q^* = \mathrm{Hom}_k(Q, k)$; in what follows, elements of Q^* will be represented through their values on the basis k^M.

PROPOSITION 3. *Let d, e be in $\mathbb{N}$, with $d < s$, $e \ge t$, and $(d+1)e \le 2st$. Given ℓ in Q^*, one can compute all $\ell(x^i y^j)$, for $i \le d$ and $j \le e$, in time $O(s^\eta \mathsf{M}(st)) \subset O^{\sim}(s^{\eta+1}t)$.*

Similar questions for univariate polynomials have been addressed in [34, 37], and the bivariate case is discussed in [37, 23, 3]. Our approach is in the continuation of these previous works: the direct extension of these results would lead to a complexity in $O^{\sim}((st)^{\eta+1})$; using Proposition 2 of the previous subsection will enable us to reduce the cost $O^{\sim}(s^{\eta+1}t)$.

Duality. We start by recalling a basic duality operation. Let $\mathscr{R}$ be a polynomial ring over k (below, $\mathscr{R}$ equals $k[X]$ or $k[X,Y]$), let $\mathscr{I}$ be a zero-dimensional ideal of $\mathscr{R}$ and let $\mathscr{Q} = \mathscr{R}/\mathscr{I}$. Then the dual space $\mathscr{Q}^*$ is endowed with a natural $\mathscr{Q}$-module structure: for $a \in \mathscr{Q}$ and $\ell \in \mathscr{Q}^*$, $a \cdot \ell$ is the linear form such that $(a \cdot \ell)(b) = \ell(ab)$ for all b in $\mathscr{Q}$.

REMARK. The *transposition principle* [8, Th. 13.20] asserts that the cost of applying a linear map is the same as that of applying its dual map, up to small correction terms (see also [34, 37, 23]). The map $\ell \mapsto a \cdot \ell$ is the dual of the "multiplication-by-a" endomorphism of $\mathscr{Q}$. Hence, following [4, 3], we could present an algorithm to realize the map $\ell \mapsto a \cdot \ell$ by "transposing" that for the map $b \mapsto ab$, using tools such as "transposed polynomial multiplication". However, giving the details of this transposition process exceeds our space limits. Thus, we will give an ad-hoc algorithm for the map $\ell \mapsto a \cdot \ell$; an advantage of this approach is that this algorithm can be implemented "as-is" on most platforms.

Basic tasks: univariate case. We continue with known material on operations modulo a univariate polynomial. Let $\mathscr{R}$ be a ring, let $\mathscr{F}$ be a monic polynomial in $\mathscr{R}[\Gamma]$ of degree d, and let $\mathscr{Q} = \mathscr{R}[\Gamma]/\mathscr{F}$ (below, $\mathscr{R}$ is either our initial base field k, or the quotient $k[X]/S$, and Γ is then either X or Y). We discuss the complexity of the following operations:

1. For $\mathscr{A}$ in $\mathscr{R}[\Gamma]$, with $\deg \mathscr{A} \leq 2d-2$, compute $\mathscr{A} \bmod \mathscr{F}$.

2. For $\lambda \in \mathscr{Q}^* = \mathrm{Hom}_{\mathscr{R}}(\mathscr{Q}, \mathscr{R})$, compute $\lambda(\gamma^i)$, for $i \leq 2d-2$, with $\gamma = \Gamma \bmod \mathscr{F}$.

To perform these tasks, we will use the algorithms of respectively [19, Ch. 9.1] and [33, Th. 3.1]. They yield the following complexity results, where we write $\mathrm{rev}_d(\mathscr{F}) = \Gamma^d \mathscr{F}(1/\Gamma)$ for the reciprocal polynomial of $\mathscr{F}$.

LEMMA 2. *Suppose that $1/\mathrm{rev}_d(\mathscr{F}) \bmod \Gamma^{d-1}$ is known. Then operations* **1** *and* **2** *can be done using 2 multiplications in degree at most d in $\mathscr{R}[\Gamma]$, and $O(d)$ additions in $\mathscr{R}$.*

Basic tasks: bivariate case. We now discuss bivariate analogues of the two questions raised above. Given a finite subset $F \subset \mathbb{N}^2$, consider the following tasks:

1. Mod_F : Given a in $k[X,Y]_F$, compute $a \bmod (S,T)$.

2. Eval_F : Given $\ell \in Q^*$, compute $\ell(x^i y^j)$, for $(i,j) \in F$.

Observe that the question of Proposition 3 can then be restated as computing Eval_E, with $E = \mathsf{A}(d,e)$. In this paragraph however, we will consider only a special case of this question, where $F = 2M = \mathsf{A}(2s-2, 2t-2)$.

PROPOSITION 4. *For $F = 2M$, one can solve problems* **1** *and* **2** *using $O(\mathsf{M}(st)) \subset O\tilde{}(st)$ operations in k.*

PROOF. The proof is now split in 4 steps. As a preamble, note that the triangular structure of the input polynomials enables us to define the following tower of extensions:

$$k \to P = k[X]/S \to Q = k[X,Y]/(S,T) \simeq P[Y]/T.$$

Observe the following fact, proved as Lemma 2.2.(i) in [20]:

(F) degree d multiplication in $P[Y]$ has cost $O(\mathsf{M}(sd))$.

STEP 1. Let us first define the auxiliary polynomials $\mathrm{rev}_s(S) = X^s S(1/X)$ and $\mathrm{rev}_t(T) = Y^t T(X, 1/Y)$, whose power series inverses are needed to apply Lemma 2. Both inverses are computed using Newton's iteration. The analysis of [19, Ch. 9.1] shows that $1/\mathrm{rev}_s(S) \bmod X^{s-1} \in k[X]$ can be computed in time $O(\mathsf{M}(s))$; using Fact (F), $1/\mathrm{rev}_t(T) \bmod Y^{t-1} \in P[Y]$ can be computed in time $O(\mathsf{M}(st))$.

STEP 2. To reduce a polynomial $a \in k[X,Y]_{2M}$ modulo (S,T), we first consider it as a polynomial in $k[X][Y]$ and reduce all its coefficients modulo S, obtaining a polynomial b. Applying Lemma 2 with $\mathscr{F} = S \in k[X]$, the cost of this step is $O(\mathsf{M}(s))$ for each coefficient; hence the total cost is in $O(\mathsf{M}(s)t) \subset O(\mathsf{M}(st))$. Now, b can be seen in $P[Y]$; the second step thus consists in reducing it modulo T in $P[Y]$. Applying Lemma 2 with $\mathscr{F} = T \in P[Y]$ and using Fact (F), the cost of this step is in $O(\mathsf{M}(st))$ as well. This proves the first point of the proposition.

STEP 3. From now on, we will denote by $Q^* = \mathrm{Hom}_k(Q,k)$ the dual of Q and by $P^* = \mathrm{Hom}_k(P,k)$ the dual of P. To prove the second point, we first define a P-module map $\phi : P^* \to P$. For any λ in P^*, there exists a unique $\Phi(\lambda) \in k[X]$ of degree less than s such that

$$\textstyle\sum_{i \geq 0} \frac{\lambda(x^i)}{X^{i+1}} = \frac{\Phi(\lambda)}{S},$$

see Proposition 1 in [5]. We let $\phi(\lambda) = \Phi(\lambda) \bmod S \in P$; then the same proposition implies that ϕ is P-linear. Knowing λ (*i.e.*, its values on the basis $1, \ldots, x^{s-1}$ of P), computing $\phi(\lambda)$ requires one polynomial multiplication in degree at most s; recovering λ from $\phi(\lambda)$ is done by a division modulo X^s. Hence both can be done in time $O(\mathsf{M}(s))$.

STEP 4. Let now ℓ be in Q^*; given ℓ, *i.e.*, the values $\ell(x^i y^j)$ for $i \leq s-1$ and $j \leq t-1$, we want to compute the values $\ell(x^i y^j)$, for $i \leq 2s-2$ and $j \leq 2t-2$.

For $j \geq 0$, define a linear form $\ell_j \in P^*$ by $\ell_j(a) = \ell(ay^j)$. Then, our problem can be restated as follows: knowing the values of $\ell_0, \ldots, \ell_{t-1}$ at $1, \ldots, x^{s-1}$, compute $\ell_j(x^i)$ for $i \leq 2s-2$ and $j \leq 2t-2$.

We first compute $\ell_t, \ldots, \ell_{2t-2}$ at $1, \ldots, x^{s-1}$. To this effect, for $j \in \mathbb{N}$, let $m_j = \phi(\ell_j)$. Since we know $\ell_0, \ldots, \ell_{t-1}$ at $1, \ldots, x^{s-1}$, by the remarks above, $m_0, \ldots, m_{t-1}$ can be computed in $O(\mathsf{M}(s)t)$ operations in k. We next deduce $m_t, \ldots, m_{2t-2}$. Define $L \in \mathrm{Hom}_P(Q,P)$ by $L(y^j) = m_j$ for $0 \leq j \leq t-1$. We claim that $L(y^j) = m_j$ for all $j \in \mathbb{N}$. Indeed, for any j, we can rewrite y^j as $\sum_{0 \leq r \leq t-1} \alpha_{j,r} y^r$, for some $\alpha_{j,r}$ in P, whence by application of ϕ,

$$m_j = \textstyle\sum_{0 \leq r \leq t-1} \alpha_{j,r} m_r = \sum_{0 \leq r \leq t-1} \alpha_{j,r} L(y^r) = L(y^j).$$

Applying Lemma 2 to L with $\mathscr{F} = T \in P[Y]$, and using Fact (F), we deduce that $m_t, \ldots, m_{2t-2}$ can be computed in $O(\mathsf{M}(st))$ operations in k. From these polynomials, $\ell_t, \ldots, \ell_{2t-2}$ can be deduced for $O(\mathsf{M}(s)t)$ operations. It then suffices to apply Lemma 2 to each ℓ_j, $j \leq 2t-2$, with $\mathscr{F} = S \in k[X]$ to conclude. The cost is again in $O(\mathsf{M}(s)t)$; putting all costs together finishes the proof. □

Modular multiplication and its dual. We can now give complexity estimates for the maps $(a,b) \mapsto ab \in Q$ and $(a,\ell) \mapsto (a \cdot \ell) \in Q^*$. Recall that M denotes $\mathsf{A}(s-1, t-1)$.

PROPOSITION 5. *Given a, b in Q, one can compute $ab \in Q$ using $O(\mathsf{M}(st)) \subset O\tilde{}(st)$ operations in k.*

PROOF. We compute the product C of the canonical preimages a and b in $k[X,Y]_M$, and reduce it modulo (S,T). Then, C belongs to $k[X,Y]_{2M}$, so the complexity estimate follows from Propositions 1 and 4. □

To discuss the dual product, we use a preparatory result. Given $\ell \in Q^*$, and a finite $E \subset \mathbb{N}^2$, we define the generating series $\mathsf{S}_E(\ell) = \sum_{(i,j) \in E} \ell(x^i y^j) X^i Y^j \in k[X,Y]_E$. Let then a be in Q, and A its canonical preimage in $k[X,Y]_M$. The following lemma shows how deduce $\mathsf{S}_E(a \cdot \ell)$ from the product of suitable polynomials; see [5, Prop. 1] for a proof.

LEMMA 3. *Let F be the Minkowski sum $E + \{(s-1, t-1)\}$. Then $X^{s-1} Y^{t-1} \mathsf{S}_E(a \cdot \ell) = \mathsf{Trunc}_F(\mathrm{rev}_M(A) \cdot \mathsf{S}_{E+M}(\ell))$.*

COROLLARY 2. *Given a in Q and ℓ in Q^*, one can compute $a \cdot \ell$ using $O(\mathsf{M}(st)) \subset O^{\tilde{}}(st)$ operations in k.*

PROOF. We first compute $\mathsf{Eval}_{2M}(\ell)$; the cost is in $O(\mathsf{M}(st))$ by Proposition 4. Then, we compute $\mathrm{rev}_M(A) \cdot \mathsf{S}_{2M}(\ell)$, which costs $O(\mathsf{M}(st))$ as well by Proposition 1. Applying Lemma 3 with $E = M$, we obtain $\mathsf{S}_M(a \cdot \ell)$, which yields the values of $a \cdot \ell$ on the canonical basis of Q. □

Proof of Proposition 3. We conclude by proving Proposition 3. As an intermediate step, we show how *several* dual products of the form $\mathsf{Eval}_E(a_i \cdot \ell_j)$ can be computed fast, using the results of the previous subsection.

LEMMA 4. *Let $d < s$ be in $\mathbb{N}$, and let $E = \mathsf{A}(d, t-1)$. Given $a_1, \ldots, a_n$ in Q, $\ell_1, \ldots, \ell_n$ in Q^*, with $(d+1)n \in O(s)$, one can compute all $\mathsf{Eval}_E(a_i \cdot \ell_j)$ using $O\left(n^{\omega-1}\mathsf{M}(st)\right) \subset O^{\tilde{}}(n^{\omega-1}st)$ operations in k.*

PROOF. We first compute all $\mathsf{Eval}_{E+M}(\ell_i)$, for $i \leq n$. Since $d < s$, E is contained in M, so Proposition 4 shows that this can be done in $O(n\mathsf{M}(st))$ operations in k. For $i \leq n$, let A_i be the canonical preimage of a_i in $k[X,Y]_M$, and let $B_i = \mathrm{rev}_M(A_i)$; define also $C_i = \mathsf{S}_{E+M}(\ell_i)$, which we can deduce for no cost from $\mathsf{Eval}_{E+M}(\ell_i)$. Defining

$$F = E + \{(s-1, t-1)\} = \mathsf{A}(s-1, s-1+d;\ t-1, 2t-2),$$

Lemma 3 shows that computing $\mathsf{Trunc}_F(B_iC_j)$ for $1 \leq i, j \leq n$ yields all $\mathsf{S}_E(a_i \cdot \ell_j)$, which gives the wanted output. Let $N = E + M = \mathsf{A}(s+d-1, 2t-2)$; then all B_i and C_j have support in N. Since $(d+1)n \in O(s)$, the result follows from Proposition 2, with $p = s+d-1$ and $q = 2t-2$. □

We can now conclude the proof of Proposition 3, using baby steps / giant steps techniques. Recall the problem: given $\ell \in Q^*$, we want to compute the values $\ell(x^iy^j)$ for $i \leq d$ and $j \leq e$, with $d < s, e \geq t$, and $(d+1)e \leq 2st$. Writing $E = \mathsf{A}(d, e)$, our problem thus amounts to compute $\mathsf{Eval}_E(\ell)$.

Define $\delta = \lceil e/t \rceil$. The set E can be rewritten as the union $E = E_0 \cup \cdots \cup E_{\delta-1}$, with $E_m = \mathsf{A}(0, d\ ;\ mt, mt+(t-1))$, and in particular $E_0 = \mathsf{A}(d, t-1)$. Then, to compute $\mathsf{Eval}_E(\ell)$, it suffices to compute all $\mathsf{Eval}_{E_m}(\ell)$, for $m \leq \delta - 1$. Writing $z = y^t$, observe further that $\mathsf{Eval}_{E_m}(\ell) = \mathsf{Eval}_{E_0}(y^{mt} \cdot \ell) = \mathsf{Eval}_{E_0}(z^m \cdot \ell)$. Let $n = \lceil \sqrt{\delta} \rceil$, and let $w = z^n$. Any $m \leq \delta - 1$ can be written as $m = i + nj$, with $i, j \leq n$. Then, z^m equals z^iw^j, so the linear form $z^m \cdot \ell$ equals $z^i \cdot \ell_j$, with $\ell_j = w^j \cdot \ell$. Thus, our problem is reduced to compute all $\mathsf{Eval}_{E_0}(z^i \cdot \ell_j)$, which was dealt with in Lemma 4.

Computing z requires no operation, since it can be read off the polynomial T. Computing all required powers z^i, including w, takes $O(n\mathsf{M}(st))$ operations using Proposition 5. Using Corollary 2, computing all linear forms ℓ_j takes time $O(n\mathsf{M}(st))$ as well. Finally, our assumptions $d < s$, $e \geq t$ and $(d+1)e \leq 2st$ easily imply that $(d+1)n \leq 4s$ and that n is in $O(\sqrt{s})$; hence, Lemma 4 gives a complexity estimate in $O(s^{\frac{\omega-1}{2}}\mathsf{M}(st)) = O(s^\eta \mathsf{M}(st))$, as requested. □

3. PROOF OF THE MAIN RESULTS

3.1 Geometric description

We first discuss basic properties of zero-dimensional varieties in the affine plane $\mathbb{A}^2(\overline{k})$, where $\overline{k}$ is an algebraic closure of k. Let π be the map $\pi : \ (\alpha, \beta) \in \mathbb{A}^2(\overline{k}) \mapsto \beta \in \mathbb{A}^1(\overline{k})$, let Z be a zero-dimensional variety in $\mathbb{A}^2(\overline{k})$, and let π_Z be the restriction of π to Z. Then, we define

$$N_Z : \ \beta \in \mathbb{A}^1(\overline{k}) \mapsto |\pi_Z^{-1}(\beta)|.$$

Thus, $N_Z(\beta)$ is the number of points of the form (α, β) in Z. We write $\deg(\pi, Z) = \max N_Z(\beta)$ for β in $\mathbb{A}^1(\overline{k})$. Following [1], we say that Z is *equiprojectable* (for the projection π) if all fibers of π_Z have the same cardinality, *i.e.*, if $N_Z(\beta)$ equals $\deg(\pi, Z)$ for all $\beta \in \pi(Z)$.

LEMMA 5. [1] *Z is equiprojectable if and only if it can be defined by a triangular set $(U(Y),\ V(Y,X))$ for the order $X > Y$. Then, $\deg_Y U = |\pi(Z)|$ and $\deg_X V = \deg(\pi, Z)$.*

When Z is not equiprojectable, we can decompose it into a union of equiprojectable varieties [13]. For $i \in \mathbb{N}_{>0}$, define

$$Z_i = \{(\alpha, \beta) \in Z \mid N_Z(\beta) = i\},$$

that is, Z_i is the subset of all points in Z having exactly i points in their π-fiber. Thus, if Z_i is not empty, it is equiprojectable, and the degree $\deg(\pi, Z_i)$ equals i.

There exist only finitely many indices i for which Z_i is not empty; we denote them by $d_1 < \cdots < d_n$. We call $Z_{d_1}, \ldots, Z_{d_n}$ the *equiprojectable components* of Z; they form a partition of Z. By the remark above, $\deg(\pi, Z_{d_j}) = d_j$; writing $e_j = |\pi(Z_{d_j})|$, we have $|Z_{d_j}| = d_je_j$. Defining $e = |\pi(Z)|$, remark in particular the equality

$$d_1e_1 + \cdots + d_ne_n = |Z_{d_1}| + \cdots + |Z_{d_n}| = |Z|. \qquad (2)$$

By Lemma 5, each Z_{d_j} can be defined by a triangular set $(U_{d_j}(Y),\ V_{d_j}(Y,X))$ for the order $X > Y$, with $\deg_Y U_{d_j} = e_j$ and $\deg_X V_{d_j} = d_j$. Note the factorizations over $\overline{k}$:

$$U_{d_j}(Y) = \textstyle\prod_{\beta \in \pi(Z_{d_j})}(Y - \beta) \qquad (3)$$

$$V_{d_j}(\beta, X) = \textstyle\prod_{(\alpha,\beta) \in \pi_Z^{-1}(\beta)}(X - \alpha) \text{ for } \beta \in \pi(Z_{d_j}). \qquad (4)$$

3.2 Algebraic tools

We now additionally suppose that Z is defined by a triangular set $(S(X),\ T(X,Y))$ for the order $Y > X$; we want to compute all triangular sets $(U_{d_j}(Y), V_{d_j}(Y,X))$, for the order $X > Y$ (observe that we have slightly modified the notation of Equation (1), since it makes indexation easier later on). As in Section 2, we write $s = \deg_X S$ and $t = \deg_Y T$, so that $|Z| = st$; note the inequalities $d_n \leq s$ and $e \geq t$. Finally, we define $Q = k[X,Y]/(S,T)$, and let x and y be the images of X and Y in Q.

Preliminaries: fast exponentiation. Let $\mathscr{R}$ be a ring, $d \in \mathbb{N}$, and $F \in \mathscr{R}[\Gamma]$, with $F(0) = 0$. Assuming that $2, \ldots, d$ are units in $\mathscr{R}$, we define the *truncated exponential* of F by

$$\exp_d(F) = \textstyle\sum_{i=0}^{d} \frac{F^i}{i!} \bmod \Gamma^{d+1}.$$

LEMMA 6. *The following holds:*

1. *If $\mathscr{R} = k$, then $\exp_d(F)$ can be computed in $O(\mathsf{M}(d))$ operations in k.*
2. *If $\mathscr{R} = k[Y]/U$, with U in $k[Y]$ of degree e, then $\exp_d(F)$ can be computed in $O(\mathsf{M}(ed))$ operations in k.*

See [7, 31] for a proof in the first case; the second case follows similarly, using Kronecker's substitution.

Minimal and characteristic polynomials. We return to our bivariate setting. Given $a \in Q$, we denote by Mul_a the

endomorphism of multiplication by a in Q. The *characteristic* and *minimal* polynomials of a are then defined as the characteristic and minimal polynomials of Mul_a; we will denote them by χ_a and m_a. Then, χ_a satisfies the following fundamental property [12, Prop. 4.2.8], originating from [2].

PROPOSITION 6. *For all a in Q, the polynomial $\chi_a \in k[Y]$ equals $\prod_{(\alpha,\beta)\in Z}(Y - a(\alpha,\beta))$.*

PROPOSITION 7. *The squarefree factorization of χ_y is $\chi_y = U_{d_1}^{d_1}\cdots U_{d_n}^{d_n}$. The minimal polynomial m_y factors as $m_y = U_{d_1}\cdots U_{d_n} = \prod_{\beta\in\pi(Z)}(Y-\beta)$.*

PROOF. Consequence of Proposition 6 and Equation (3). □

Trace formulas. The *trace* of an element $a \in Q$ is the trace of Mul_a; hence, the trace is a linear map, which we will write $\mathrm{tr} \in Q^*$. Following [29, 17], the bases of all our algorithms will be suitable trace formulas, relying on the following easy consequence of Proposition 6: for all $a \in Q$, we have

$$\mathrm{tr}(a) = \textstyle\sum_{(\alpha,\beta)\in Z} a(\alpha,\beta). \tag{5}$$

PROPOSITION 8. *All $\mathrm{tr}(x^i y^j)$, with $i \le s-1$ and $j \le t-1$, can be computed using $O(\mathsf{M}(st)) \subset O^{\tilde{}}(st)$ operations in k.*

PROOF. Define the *residue* $\mathrm{res} \in Q^*$ as the linear form that maps $a \in Q$ to its coefficient $\mathsf{coeff}(a, x^{s-1}y^{t-1})$, when a is written on the canonical basis of Q. Then the residue formula [16] shows that $\mathrm{tr} = \partial S/\partial X \cdot (\partial T/\partial Y \cdot \mathrm{res})$, and Corollary 2 concludes the proof. □

It is well-known that given the traces of the powers of y, one can deduce its characteristic polynomial.

PROPOSITION 9. *The polynomial $\mathrm{rev}_{st}(\chi_y) \in k[Y]$ equals $\exp_{st}(-\sum_{i=1}^{st}\mathrm{tr}(y^i)\frac{Y^i}{i})$.*

PROOF. Proposition 6 and Equation (5) imply that the Newton sums of χ_y are the traces of the powers of y. The result is now a generating series restatement of Newton's relations, which can be applied due to our assumption (H). □

Proposition 9 will be used to eliminate X. The next step will consist in recovering the dependency of X in terms of Y: this is done in Proposition 11 below, for which we need some preliminary results. We first define a "trace-like" map tr_b, by the following formula: for all $\beta \in \mathbb{A}^1(\overline{k})$, and all $b \in Q$,

$$\mathrm{tr}_b(\beta) = \sum\nolimits_{(\alpha,\beta)\in\pi_Z^{-1}(\beta)} b(\alpha,\beta), \tag{6}$$

where the empty sum equals zero. The definition of tr_b involves algebraic numbers, but the following proposition shows that it can be computed using rational formulas.

PROPOSITION 10. *For b in Q, the product $S_b = \mathrm{rev}_e(m_y)\sum_{i\ge0}\mathrm{tr}(b\,y^i)Y^i$ is in $k[Y]$, and has degree at most $e-1$. Defining $R_b = \mathrm{rev}_{e-1}(S_b)\,(m_y')^{-1} \bmod m_y$, the equality $R_b(\beta) = \mathrm{tr}_b(\beta)$ holds for all $\beta \in \pi(Z)$.*

PROOF. By Equation (5), $\sum_{i\ge0}\mathrm{tr}(b\,y^i)Y^i$ equals

$$\textstyle\sum_{i\ge0}\sum_{(\alpha,\beta)\in Z} b(\alpha,\beta)\beta^i Y^i = \sum_{(\alpha,\beta)\in Z}\frac{b(\alpha,\beta)}{1-\beta Y}$$
$$\textstyle= \sum_{\beta\in\pi(Z)}\frac{\sum_{(\alpha,\beta)\in\pi_Z^{-1}(\beta)} b(\alpha,\beta)}{1-\beta Y} = \sum_{\beta\in\pi(Z)}\frac{\mathrm{tr}_b(\beta)}{1-\beta Y}.$$

Reducing the last sum to common denominator yields

$$\frac{\prod_{\beta\in\pi(Z)}\mathrm{tr}_b(\beta)\prod_{\beta'\in\pi(Z),\beta'\neq\beta}(1-\beta' Y)}{\prod_{\beta\in\pi(Z)}(1-\beta Y)};$$

identifying the denominator with $\mathrm{rev}_e(m_y)$, we get

$$S_b = \textstyle\prod_{\beta\in\pi(Z)}\mathrm{tr}_b(\beta)\prod_{\beta'\in\pi(Z),\beta'\neq\beta}(1-\beta' Y),$$

which establishes our first claim. We further deduce that

$$\mathrm{rev}_{e-1}(S_b) = \textstyle\prod_{\beta\in\pi(Z)}\mathrm{tr}_b(\beta)\prod_{\beta'\in\pi(Z),\beta'\neq\beta}(Y-\beta').$$

Hence, for all $\beta \in \pi(Z)$, $\mathrm{rev}_{e-1}(S_b)(\beta) = \mathrm{tr}_b(\beta)m_y'(\beta)$. □

For $i \in \mathbb{N}$ and $j = 1,\dots,n$, we write $R_i = R_{x^i}$ and $R_{i,j} = R_i \bmod U_{d_j}$. Next, we define $P_j = k[Y]/U_{d_j}$, and let v_{d_j} be the image of V_{d_j} in $P_j[X]$. Knowing v_{d_j} enables us to recover V_{d_j}, since V_{d_j} has degree in Y less than that of U_{d_j}. Now, working modulo U_{d_j} yields the following proposition.

PROPOSITION 11. *For all $j \le n$, the equality $\mathrm{rev}_{d_j}(v_{d_j}) = \exp_{d_j}(-\sum_{i=1}^{d_j} R_{i,j}\frac{X^i}{i})$ holds in $P_j[X]$.*

PROOF. By logarithmic derivation, it suffices to prove that

$$\tfrac{\mathrm{rev}_{d_j}(v_{d_j})'}{\mathrm{rev}_{d_j}(v_{d_j})} = -\textstyle\sum_{i\ge0} R_{i+1,j}X^i \text{ in } P_j[[X]].$$

We extend scalars from k to $\overline{k}$, and prove this identity in $\overline{P_j}[X]$, where $\overline{P_j} = \overline{k}[Y]/U_{d_j}$. By Chinese remaindering,

$$\overline{P_j} \simeq \textstyle\prod_{\beta \text{ root of } U_{d_j}} \overline{k}[Y]/(Y-\beta) \simeq \overline{k}^{d_j},$$

the reduction maps being evaluations at the roots of U_{d_j}. Let us thus consider a root β of U_{d_j}. By Equation (4), the image of v_{d_j} through evaluation at β is the polynomial

$$V_{d_j}(\beta, X) = \textstyle\prod_{(\alpha,\beta)\in\pi_Z^{-1}(\beta)}(X-\alpha),$$

which implies that

$$\tfrac{\mathrm{rev}_{d_j}(V_{d_j}(\beta,X))'}{\mathrm{rev}_{d_j}(V_{d_j}(\beta,X))} = -\textstyle\sum_{(\alpha,\beta)\in\pi_Z^{-1}(\beta)}\sum_{i\ge0}\alpha^{i+1}X^i.$$

By Equation (6), this equals $-\sum_{i\ge0}\mathrm{tr}_{x^{i+1}}(\beta)X^i$. By Proposition 10, $\mathrm{tr}_{x^{i+1}}(\beta) = R_{i+1}(\beta) = R_{i+1,j}(\beta)$. Using the isomorphism (7) finishes the proof. □

3.3 Complexity statements

Computing $\mathbf{U_{d_1},\dots,U_{d_n}}$. The first step consists in eliminating the variable X, by computing $U_{d_1},\dots,U_{d_n}$.

PROPOSITION 12. *One can compute $U_{d_1},\dots,U_{d_n}$ and m_y using $O(\mathsf{M}(st)(s^\eta + \log(t))) \subset O^{\tilde{}}(s^{\eta+1}t)$ operations in k.*

PROOF. We first compute the traces of all elements in the monomial basis of Q; by Proposition 8, the cost is $O(\mathsf{M}(st))$. The assumptions of Proposition 3 being satisfied, all $\mathrm{tr}(y^j)$ for $j \le st$ can be computed in time $O(s^\eta\mathsf{M}(st))$. Then, we compute χ_y using Proposition 9; the cost is in $O(\mathsf{M}(st))$ by Lemma 6.**1**. By Proposition 7, its squarefree factorization yields $U_{d_1},\dots,U_{d_n}$ and m_y. Due to assumption (H), the cost is in $O(\mathsf{M}(st)\log(st))$, see [19, Th. 14.23]. □

Computing points of bounded degree. The next step consists in recovering $V_{d_1},\dots,V_{d_n}$. To reach an admissible complexity result, however, we compute a *partial* output first.

Let $\delta \le d_n$ be some fixed threshold. Then, there exists a unique $j(\delta) \le n$ such that $d_{j(\delta)} \le \delta < d_{j(\delta)+1}$, with for consistency $d_0 = 0$ and $d_{n+1} = \infty$. The variety Z can correspondingly by decomposed as the disjoint union

$$Z = Z_{d_1} \cup \cdots \cup Z_{d_{j(\delta)}} \cup Z(\delta),$$

where $Z(\delta)$ is the union of all π-fibers of cardinality more than δ. Observe in particular that the equiprojectable decomposition of $Z(\delta)$ is given by $Z_{d_{j(\delta)+1}}, \ldots, Z_{d_n}$.

Using the previous results, we show how to compute all polynomials $V_{d_1}, \ldots, V_{d_{j(\delta)}}$, assuming that all quantities computed in the proof of Proposition 12 are still available.

PROPOSITION 13. *Suppose that* $(\delta+1)e \leq 2st$. *Then one can compute all polynomials* $V_{d_1}, \ldots, V_{d_{j(\delta)}}$ *using* $O(\mathsf{M}(st)(s^\eta + \log(t))) \subset O^\sim(s^{\eta+1}t)$ *operations in* k.

PROOF. We first compute all traces $\mathrm{tr}(x^i y^j)$, for $i \leq \delta$ and $j \leq e$. Recall that $\delta \leq d_n \leq s$ and $e \geq t$; hence, if $\delta < s$, Proposition 3 gives this output, with a cost of $O(s^\eta \mathsf{M}(st))$. If $\delta = s$, Proposition 3 only gives $\mathrm{tr}(x^i y^j)$ for $i < \delta$; however, the missing values can be recovered for the same cost, for instance by applying the same approach to $(x \cdot \mathrm{tr})$.

We now compute all polynomials $R_{i,j}$, for $i \leq \delta$ and $j \leq n$. For fixed i, knowing $\mathrm{tr}(x^i y^j)$ for $j \leq e$, the polynomial S_{x_i} of Proposition 10 is obtained by a multiplication in degree e, hence for a cost of $\mathsf{M}(e)$; then $R_i = R_{x_i}$ can be computed in time $O(\mathsf{M}(e)\log(e))$ using [19, Cor. 11.6]. Given R_i, all polynomials $R_{i,j}$, for $j \leq n$, can be deduced in $O(\mathsf{M}(e)\log(e))$ operations, using [19, Cor. 10.17]. Hence, all $R_{i,j}$ for $i \leq \delta$ and $j \leq n$ can be computed in $O(\delta \mathsf{M}(e)\log(e))$ operations. Due to our assumption $(\delta+1)e \leq 2st$, this cost is in $O(\mathsf{M}(st)\log(st))$. Finally, we deduce $V_{d_1}, \ldots, V_{d_{j(\delta)}}$ using Proposition 11; by Lemma 6.**2**, each exponential requires time $O(\mathsf{M}(e_i d_i))$; using Equation (2), the total cost is in $O(\mathsf{M}(st))$. Summing all costs yields our bound. □

If Z is equiprojectable, we have $n = 1$ and $d_n e = st$, so that $(d_n+1)e \leq 2st$. Hence, a single application of Proposition 13 with $\delta = d_n$ gives us V_{d_n}; this proves Theorem 1.

Proof of Theorem 2. We now out the details of a special case, when the input polynomial $T(X,Y)$ has the form $Y - \mathfrak{T}(X)$, *i.e.* when $t = 1$. In this case, each subset Z' of Z can be defined by a triangular set $(S_{Z'}(X), Y - \mathfrak{T}_{Z'}(X))$. Two particular cases will be considered:

CASE 1. For $j \leq n$, Z_{d_j} can be defined by a triangular set $(S_{Z_{d_j}}(X), Y - \mathfrak{T}_{Z_{d_j}}(X))$. We will write $S_{Z_{d_j}} = A_{d_j}$ and $\mathfrak{T}_{Z_{d_j}} = B_{d_j}$ for simplicity.

CASE 2. For $\delta \leq d_n$, the set $Z(\delta) = Z - Z_{d_1} \cdots - Z_{d_{j(\delta)}}$ introduced previously can be defined by a triangular set $(S_{Z(\delta)}(X), Y - \mathfrak{T}_{Z(\delta)}(X))$. We will write $S_{Z(\delta)} = C_\delta$ and $\mathfrak{T}_{Z(\delta)} = D_\delta$ for simplicity. Note that $S = C_0$ and $\mathfrak{T} = D_0$. For $\delta < \delta'$, we have $C_\delta = C_{\delta'} \prod_{j(\delta)<j\leq j(\delta')} A_{d_j}$ and $D'_\delta = D_\delta \bmod C_{\delta'}$. This follows from the equality $Z(\delta) = \bigcup_{j(\delta)<j\leq j(\delta')} Z_{d_j} \cup Z(\delta')$, where the union is disjoint.

These remarks suggest the following algorithm: suppose that $(C_\delta, Y - D_\delta)$ are known. Then, we compute (U_{d_j}, V_{d_j}) for all $j(\delta) < j \leq j(\delta')$, then all corresponding (A_{d_j}, B_{d_j}), and finally $(C_{\delta'}, Y - D_{\delta'})$ through the above equations.

For complexity statements, we extend the previous notation by $e(\delta) = |\pi(Z(\delta))|$. In $Z(\delta)$, all π-fibers have cardinality at least $\delta+1$, so that in particular, $(\delta+1)e(\delta) \leq |Z(\delta)|$.

PROPOSITION 14. *Let* $\delta < \delta' \in \mathbb{N}$, *such that* $(\delta'+1)e(\delta) \leq 2|Z(\delta)|$. *Given* $(C_\delta, Y - D_\delta)$, *one can compute all* (U_{d_j}, V_{d_j}) *and* (A_{d_j}, B_{d_j}) *for* $j(\delta) < j \leq j(\delta')$, *as well as* $(C_{\delta'}, Y - D_{\delta'})$, *using* $O(s^\eta \mathsf{M}(s))$ *operations in* k.

PROOF. We apply Propositions 12 and 13 to $(C_\delta, Y - D_\delta)$, since all necessary conditions are satisfied; one checks that these yields the polynomials (U_{d_j}, V_{d_j}) for $j(\delta) < j \leq j(\delta')$, and that the complexity fits into the requested bound. Then, we have to compute all (A_{d_j}, B_{d_j}). This is done by applying Theorem 1 to the (U_{d_j}, V_{d_j}), but *exchanging the roles of* X *and* Y *in the statement of the theorem.* For a given j, the cost is in $O((d_j e_j)^\eta \mathsf{M}(d_j e_j))$; in view of Equation (2), the sum of these costs is in $O(s^\eta \mathsf{M}(s))$ as well.

Computing the product of all A_{d_j} has cost $O(\mathsf{M}(s)\log(s))$ by [19, Lem. 10.4]; $C_{\delta'}$ and $D_{\delta'}$ are obtained in time $O(\mathsf{M}(s))$ [19, Th. 9.6]. Putting all estimates together finishes the proof. □

This enables us to conclude the proof of Theorem 2. We set $\delta_0 = 0$; the corresponding $C_0 = S$ and $D_0 = \mathfrak{T}$ are known. Then, we set $\delta_i = 2^i$, for $1 \leq i \leq \lceil \log_2(s) \rceil$, and successively apply the previous proposition with $\delta = \delta_i$ and $\delta' = \delta_{i+1}$, for $i \geq 0$; in particular, this produces all requested polynomials (U_{d_j}, V_{d_j}). It remains to check that the assumption of the proposition is satisfied: for $i \geq 0$, $(\delta_{i+1}+1)e(\delta_i)$ equals $(2\delta_i+1)e(\delta_i)$. By the above discussion, this is upper-bounded by $2|Z(\delta_i)|$, as requested. The complexity estimate of Theorem 2 follows, concluding the proof.

The general case. Our solution reduces to the previous case, using a general position argument.

STEP 1. Substitute Y by $Y = Y' + aX$ in the input system (S,T), for some random $a \in k$, and perform the change of order in the modified system: for generic a, the output consists in a single set of polynomials $U(Y'), V(Y',X)$, where V has degree 1 in X, and U has degree st.

STEP 2. Substitute X by $X = X' + bY'$ in (U,V), for some random $b \in k$, and restore the initial order: for generic b, the output consists in a single set of polynomials $(S'(X'), T'(X',Y'))$, where T' has degree 1 in Y', and S' has degree st. We are now in general coordinates (X',Y').

STEP 3. Restore the initial Y coordinate, by replacing Y' by its value $(Y - aX')/(ab+1)$ in (S',T'). Perform the change of order on the system, which has degree 1 in Y; this yields a family of polynomials $(U_{d_j}(Y,X'), V_{d_j}(Y,X'))$. Replacing X' by its value $(1+ab)X - bY$ yields the wanted output.

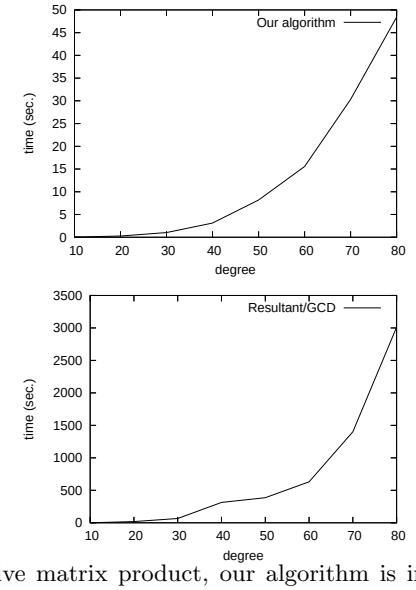
Steps 1 and 2 are done using Theorem 1; Step 3 relies on Theorem 2; the changes of variables have negligible cost [19, Th. 9.15]. The complexity estimate of Theorem 3 follows. Success of the algorithm requires that both $Y' = -aX + Y$ and $X' = (1+ab)X - bY$ separate the points of Z. Let $\Delta_{X'}$ and $\Delta_{Y'} \in k[\mathbf{a},\mathbf{b}]$ be the discriminants of the characteristic polynomials of $-\mathbf{a}X + Y$ and $(1+\mathbf{ab})X - \mathbf{b}Y$ in $k(\mathbf{a},\mathbf{b})[X,Y]/(S,T)$, where $\mathbf{a},\mathbf{b}$ are new variables. If (a,b) does not cancel $\Delta = \Delta_{X'}\Delta_{Y'}$, our separating condition holds. Now, both $\Delta_{X'}$ and $\Delta_{Y'}$ are non-zero polynomials of degree at most $st\frac{st-1}{2}$, and the result follows. □

4. EXPERIMENTS

We implemented the algorithms of the previous section in C++, on top of Shoup's NTL library [35, 36]. Our implementation works over finite fields, for which NTL provides product, division, GCD and resultant. The implementation uses improvements not detailed above (reuse of quantities often needed, transposed operations [4]); finer threshold selection strategies were used, for the determination of the number of baby / giant steps. Polynomial matrix multiplication was done using evaluation and interpolation. We used a cubic matrix multiplication algorithm, but the low

constant in this algorithm made the linear algebra part less than 10% of the time for the largest reported example.

We compare our algorithm with the resultant / GCD approach discussed in the introduction, on input of the form $(S(X), T(X,Y))$ with $s = \deg_X S = \delta^2$ and $t = \deg_Y T = 1$, and output $(U(Y), V(Y,X)))$, with $\deg_Y U = \deg_X V = \delta$. The following figure reports our result; the timings are obtained on a Pentium 3-M, 1GHz machine, with 256 Mb of memory.



Using naive matrix product, our algorithm is in the same complexity class as the resultant / GCD approach; however, as mentioned above, the linear algebra contribution to the total time is far from dominant. This partly explains the ratio we observe; the other reason is that our algorithm uses mostly basic operations (multiplications), where the resultant computation relies on the fast Euclidean algorithm.

5. CONCLUSION

An important practical case is that of polynomials with rational coefficients (particularly for the application to rational integration). A natural approach is to use Hensel lifting: the article [13] gives a probabilistic algorithm to do so, and its performances have to be experimented in our situation. Besides, the question remains of extending these algorithms to the n-variate case. The ideas are the same, but the analysis still has to be done.

6. REFERENCES

[1] P. Aubry and A. Valibouze. Using Galois ideals for computing relative resolvents. *JSC*, 30(6):635–651, 2000.

[2] W. Auzinger and H. J. Stetter. An elimination algorithm for the computation of all zeros of a system of multivariate polynomial equations. In *Proc. Int. Conf. on Num. Math.*, number 86 in Int. Ser. in Num. Math., pp. 11–30. Birkhäuser, 1988.

[3] A. Bostan, P. Flajolet, B. Salvy, and É. Schost. Fast computation of special resultants. *JSC*, 41:1–29, 2006.

[4] A. Bostan, G. Lecerf, and É. Schost. Tellegen's principle into practice. In *ISSAC 03*, pp. 37–44. ACM, 2003.

[5] A. Bostan, B. Salvy, and É. Schost. Fast algorithms for zero-dimensional polynomial systems using duality. *Appl. Alg. in Eng. Comm. Comput.*, 14(4):239–272, 2003.

[6] F. Boulier, F. Lemaire, and M. Moreno Maza. Pardi! In *ISSAC 01*, pp. 38–47. ACM, 2001.

[7] R. P. Brent. Multiple-precision zero-finding methods and the complexity of elementary function evaluation. In *Analytic computational complexity*, pp. 151–176. Academic Press, 1976.

[8] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic complexity theory*. Springer, 1997.

[9] D. G. Cantor and E. Kaltofen. On fast multiplication of polynomials over arbitrary algebras. *Acta Inform.*, 28(7):693–701, 1991.

[10] L. Cerlienco and M. Mureddu. From algebraic sets to monomial linear bases by means of combinatorial algorithms. *Discrete Math.*, 139:73–87, 1995.

[11] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *JSC*, 9(3):251–280, 1990.

[12] D. A. Cox, J. Little, and D. O'Shea. *Using algebraic geometry*. Springer-Verlag, New York, 1998.

[13] X. Dahan, M. Moreno Maza, É. Schost, W. Wu, and Y. Xie. Lifting techniques for triangular decompositions. In *ISSAC 05*, pp. 108–115. ACM, 2005.

[14] X. Dahan, M. Moreno Maza, É Schost, and Y. Xie. On the complexity of the D5 principle, 2005.

[15] J. Della Dora, C. Dicrescenzo, and D. Duval. About a new method method for computing in algebraic number fields. In *EUROCAL 85, Vol. 2*, volume 204 of *LNCS*. Springer, 1985.

[16] M. Demazure. Charles Hermite déjà .... Note informelle du calcul formel, École polytechnique, 1987.

[17] G. M. Díaz-Toca and L. González-Vega. An explicit description for the triangular decomposition of a zero-dimensional ideal through trace computations. In *Symbolic computation*, volume 286 of *Cont. Math.*, pp. 21–35. AMS, 2001.

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

[19] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, 1999.

[20] J. von zur Gathen and V. Shoup. Computing Frobenius maps and factoring polynomials. *C. Compl.*, 2(3):187–224, 1992.

[21] E. Kaltofen. Fast parallel absolute irreducibility testing. *JSC*, 1(1):57–67, 1985.

[22] E. Kaltofen. On computing determinants of matrices without divisions. In *ISSAC 92*, pp. 342–349. ACM, 1992.

[23] E. Kaltofen. Challenges of symbolic computation: my favorite open problems. *JSC*, 29(6):891–919, 2000.

[24] D. Lazard. Ideal bases and polynomial decomposition: Case of two variables. *JSC*, 1:261–270, 1985.

[25] D. Lazard. Solving zero-dimensional algebraic systems. *JSC*, 13:117–133, 1992.

[26] D. Lazard and R. Rioboo. Integration of rational functions: rational computation of the logarithmic part. *JSC*, 9(2):113–115, 1990.

[27] M. Nüsken and M. Ziegler. Fast multipoint evaluation of bivariate polynomials. In *ESA 2004*, number 3222 in LNCS, pp. 544–555. Springer, 2004.

[28] M. Rothstein. A new algorithm for the integration of exponential and logarithmic functions. In *1977 MACSYMA users conference*, pp. 263–274. NASA, 1977.

[29] F. Rouillier. Solving zero-dimensional systems through the Rational Univariate Representation. *Appl. Alg. in Eng. Comm. Comput.*, 9(5):433–461, 1999.

[30] F. Rouillier. On the rational univariate representation. In *ICPSS*, pp. 75–79, 2004.

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

[32] R. Schoof. Counting points on elliptic curves over finite fields. *J. Théor. Nombres Bordeaux*, 7(1):219–254, 1995.

[33] V. Shoup. A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic. In *ISSAC 91*, pp. 14–21. ACM, 1991.

[34] V. Shoup. Fast construction of irreducible polynomials over finite fields. *JSC*, 17(5):371–391, 1994.

[35] V. Shoup. A new polynomial factorization algorithm and its implementation. *JSC*, 20(4):363–397, 1995.

[36] V. Shoup. NTL: A library for doing number theory. `http://www.shoup.net`, 1996–2006.

[37] V. Shoup. Efficient computation of minimal polynomials in algebraic extensions of finite fields. In *ISSAC 99*, pp. 53–58. ACM, 1999.

[38] B. M. Trager. Algebraic factoring and rational function integration. In *SYMSAC 76*, pp. 219–226. ACM, 1976.