LIFTING AND RECOMBINATION TECHNIQUES FOR ABSOLUTE FACTORIZATION

G. CHÈZE AND G. LECERF

Abstract. In the vein of recent algorithmic advances in polynomial factorization based on lifting and recombination techniques, we present new faster algorithms for computing the absolute factorization of a bivariate polynomial. The running time of our probabilistic algorithm is less than quadratic in the dense size of the polynomial to be factored.

Introduction

Throughout this article, F denotes the polynomial we want to factor: it is a squarefree polynomial in two variables x and y over a commutative field $\mathbb{K}$; its total degree is denoted by d and is assumed to be positive. Under

$$\text{Hypothesis (C)} \quad \mathbb{K} \text{ has characteristic 0 or at least } d(d-1)+1,$$

we present new faster probabilistic and deterministic algorithms for computing the *absolute factorization* of F, that is the irreducible factorization over the algebraic closure $\bar{\mathbb{K}}$ of $\mathbb{K}$. In order to avoid confusion we say *rational factorization* for the factorization in $\mathbb{K}[x,y]$.

Absolute factorization is a classical problem which intervenes in several areas. In the beginning of the eighties, the first polynomial time algorithms originated in the field of effective algebraic geometry and were motivated by the problem of computing irreducible decompositions of algebraic closed sets [GC84]. Such computations still remain challenging problems from theoretical and practical points of view. Yet current algorithms already allow one to deal with real world applications. For instance, when $\mathbb{K}=\mathbb{Q}$, the absolute factorization of F is equivalent to the decomposition of the smooth locus of the curve defined by $F=0$ into path-connected components. This decomposition is an important ingredient when solving certain problems arising from kinematics, as exemplified in [SVW04]. The first absolute factorization algorithms were also motivated by symbolic integration [Tra84]. Nowadays absolute factorization is even used in the resolution of linear differential equations [SU97, HRUW99, Bro01].

We start with some prerequisites. Then we present our main results and give an overview of the main steps of our algorithms. We conclude this introduction with discussing the related works.

Notation. The algebra of the polynomials in two variables over $\mathbb{K}$ is denoted by $\mathbb{K}[x,y]$. The vector space of the polynomials of total degree at most m is represented by $\mathbb{K}[x,y]_m$. The field of fractions of $\mathbb{K}[y]$ is written $\mathbb{K}(y)$ and the power series algebra over $\mathbb{K}$ is written $\mathbb{K}[[x]]$. For any polynomial $G\in\mathbb{K}[x,y]$, $\deg(G)$ represents the total degree of G, and $\deg_x(G)$ represents the degree of G in the variable x.

Date: Preliminary version of April 14, 2005.

2000 *Mathematics Subject Classification.* Primary 12Y05, 68W30; Secondary 11Y16, 12D05, 13P05.

Key words and phrases. Absolute factorization, absolute primality, polynomial factorization.

When defined, the greatest common divisor of f and g is denoted by $\gcd(f,g)$. The remainder of f divided by g is written $\mathrm{rem}(f,g)$. The resultant of f and g in $\mathbb{K}[y]$ is written $\mathrm{Res}(f,g)$. For multivariate polynomials, say G and H in $\mathbb{K}[x,y,z]$, the resultant of G and H seen in $\mathbb{K}[x,z][y]$ is written $\mathrm{Res}_y(G,H)$.

We use the notation $\langle \mu_1,\ldots,\mu_r\rangle$ to represent the vector space generated by the vectors $\mu_1,\ldots,\mu_r$. For a real number a, the smallest integer larger than or equal to (resp. the greatest integer less than or equal to) a is denoted by $\lceil a\rceil$ (resp. $\lfloor a\rfloor$).

In the pseudo-code, we use the function coeff in various contexts. For any ring R, if $G\in R[[x,y]]$ then $\mathrm{coeff}(G,x^iy^j)$ represents the coefficient of the monomial x^iy^j in G. For a univariate polynomial $f\in\mathbb{K}[y]$ of degree d, if $\mathbb{A}$ represents the $\mathbb{K}$-algebra $\mathbb{K}[y]/(f(y))$, and if φ denotes the residue class of y in $\mathbb{A}$, then any $b\in\mathbb{A}$ can be uniquely written as $b=b_0+b_1\varphi+\cdots+b_{d-1}\varphi^{d-1}$; we define $\mathrm{coeff}(b,\varphi^i):=b_i$. For readability, we write $\mathrm{coeff}(G,\varphi^ix^jy^k)$ instead of $\mathrm{coeff}(\mathrm{coeff}(G,x^jy^k),\varphi^i)$ for any $G\in\mathbb{A}[[x,y]]$.

Complexity Model. For our complexity analysis, we use the *computation tree* model [BCS97, Chapter 4] with the *total complexity* point of view. This means that complexity estimates charge a constant cost for each arithmetic operation ($+$, $-$, $\times$, $\div$) and the equality test. All the constants in the base fields (or rings) of the trees are thought to be freely at our disposal.

We use the classical $\mathcal{O}$ and $\tilde{\mathcal{O}}$ ("soft Oh") notation in the neighborhood of infinity as defined in [GG03, Chapter 25.7]. Informally speaking, "soft Oh"s are used for readability in order to hide logarithmic factors in complexity estimates.

Polynomials and power series are represented by dense vectors of their coefficients in the usual monomial basis. For each integer d, we assume that we are given a computation tree that computes the product of two univariate polynomials of degree at most d with at most $\mathsf{M}(d)$ operations, independently of the base ring. As in [GG03, Chapter 8.3], for any positive integers d_1 and d_2, we assume that M satisfies:

$$\mathsf{M}(d_1d_2)\le d_1^2\mathsf{M}(d_2) \tag{1}$$

and

$$\mathsf{M}(d_1)/d_1\le \mathsf{M}(d_2)/d_2 \quad\text{if}\quad d_1\le d_2. \tag{2}$$

In particular, the latter assumption implies the *super-additivity* of M, namely:

$$\mathsf{M}(d_1)+\mathsf{M}(d_2)\le\mathsf{M}(d_1+d_2). \tag{3}$$

This way we can design algorithms that do not depend on the subroutine chosen for polynomial multiplication (Karatsuba or fast Fourier transform, for instances). The best function M known so far belongs to $\mathcal{O}(d\log(d)\log\log(d))\subseteq\tilde{\mathcal{O}}(d)$ [GG03, Theorem 8.23].

We recall that the computations of the *resultant* and the *extended greatest common divisor* of two univariate polynomials of degree at most d over $\mathbb{K}$ take $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$ [GG03, Chapter 11]. In particular, if $\mathbb{E}$ is an algebraic extension of $\mathbb{K}$ of degree d then each field operation in $\mathbb{E}$ takes $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$.

We also recall that a polynomial in $\mathbb{K}[z]$ of degree at most d can be interpolated from its values at $d+1$ pairwise distinct points with $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$. A polynomial of degree at most d can also be evaluated at $d+1$ points with the same cost: this operation is often called *multi-point evaluation*. We refer the reader to [GG03, Chapter 10] for these fast algorithms. Recent advances can be found in [BLS03, BS04].

Lastly we use the constant ω to denote a *feasible matrix multiplication* exponent as defined in [GG03, Chapter 12]: two $n\times n$ matrices over $\mathbb{K}$ can be multiplied

with $\mathcal{O}(n^\omega)$ field operations. As in [Sto00] we require that $2 < \omega \leq 3$. In contrast to polynomials, we shall only deal with matrices over $\mathbb{K}$.

Representation of the Absolute Factorization. The absolutely irreducible factors of F are written $F_1, \ldots, F_r$. In our algorithms, these factors are represented by a set of pairs of polynomials $\{(q_1, \mathsf{F}_1), \ldots, (q_s, \mathsf{F}_s)\}$ which satisfy the following properties:

- For all $i \in \{1, \ldots, s\}$, the polynomial q_i belongs to $\mathbb{K}[z]$, is monic, squarefree and $\deg(q_i) \geq 1$.
- For all $i \in \{1, \ldots, s\}$, the polynomial F_i belongs to $\mathbb{K}[x, y, z]$, and $\deg_z(\mathsf{F}_i) \leq \deg(q_i) - 1$. The total degree of $\mathsf{F}_i(x, y, \alpha)$ is constant when α runs over the roots of q_i.
- $\sum_{i=1}^{s} \deg(q_i) = r$ and to each absolutely irreducible factor F_j there corresponds a unique pair $(i, \alpha) \in \{1, \ldots, s\} \times \bar{\mathbb{K}}$ such that $q_i(\alpha) = 0$ and F_j is proportional to $\mathsf{F}_i(x, y, \alpha)$.

This representation is not redundant. In particular, for each i, the polynomials $\mathsf{F}_i(x, y, \alpha)$ are pairwise distinct when α is taken over all the roots of q_i. Of course, this representation is not unique.

Example 1. If F depends on a single variable, say y, then we can take $s := 1$, $q_1(z)$ as the monic part of $F(0, z)$ and $\mathsf{F}_1(x, y, z) := y - z$. Here the absolute factorization is the decomposition of F into linear factors.

Example 2. If $\mathbb{K} := \mathbb{Q}$ and $F := y^2 - 2x^2$ then we can take $s := 1$, $q_1(z) := z^2 - 2$, $\mathsf{F}_1(x, y, z) := y - zx$. Observe that F and q_1 are irreducible over $\mathbb{Q}$.

For any $i \in \{1, \ldots, s\}$, the polynomial $P_i := \mathrm{Res}_z(q_i(z), \mathsf{F}_i(x, y, z)) \in \mathbb{K}[x, y]$ is a factor of F, and its absolute factorization can be represented by (q_i, F_i). In addition, it is easy to see that P_i is irreducible if and only if q_i is irreducible. The rational factorization of F can thus be computed from the irreducible factors of $q_1, \ldots, q_s$ by arithmetic operations in $\mathbb{K}$ alone.

Main Results. In this article we present new algorithms to test the absolute irreducibility of F, and to compute the absolute factorization of F. Since we use the dense representation for F, the size of F is of the order of magnitude of d^2. Our first result implies that the absolute factorization of F can be computed in softly quadratic time by a deterministic algorithm:

Theorem 1. *Under Hypothesis (C), the absolute factorization of a squarefree bivariate polynomial over $\mathbb{K}$ of total degree d can be computed with $\mathcal{O}(d^3\mathsf{M}(d)\log(d))$ arithmetic operations in $\mathbb{K}$.*

The absolute irreducibility test can be performed a bit faster:

Theorem 2. *Under Hypothesis (C), the absolute irreducibility of a squarefree bivariate polynomial over $\mathbb{K}$ of total degree d can be tested with*

$$\mathcal{O}(d^{\omega+1} + d^2\mathsf{M}(d)(\mathsf{M}(d)/d + \log(d)))$$

arithmetic operations in $\mathbb{K}$.

When using fast polynomial multiplication, that is $\mathsf{M}(d) \in \tilde{\mathcal{O}}(d)$, the cost of the test drops to $\mathcal{O}(d^{\omega+1})$.

Although we will use no probabilistic model of computation, we will informally say *probabilistic algorithm* when speaking about the computation trees occurring in the next theorem. For the sake of precision, we prefer to express the probabilistic aspects in terms of families of computation trees. Almost all the trees of a family are expected to be executable on a given input (if the cardinality of $\mathbb{K}$ is infinite). For any polynomial $P \in \bar{\mathbb{K}}[x_1, \ldots, x_n]$, we introduce $\mathcal{U}(P) := \{a \in \mathbb{K}^n \mid P(a) \neq 0\}$.

Theorem 3. *For any positive integer d satisfying Hypothesis (C), there exists a family of computation trees over $\mathbb{K}$ parametrized by*

$$(u, v, a, c_1, \ldots, c_d) \in \mathbb{K}^{d+3}$$

such that, for any input squarefree polynomial $F \in \mathbb{K}[x, y]$ of total degree d, we have:

- *Any executable tree of the family returns the absolute factorization of F.*
- *There exists a nonzero polynomial $P \in \mathbb{K}[U]$ of degree at most d such that, for any $u \in \mathcal{U}(P)$, there exists a nonzero polynomial $Q_u \in \mathbb{K}[V]$ of degree at most $d(d-1)$ such that, for any $v \in \mathcal{U}(Q_u)$, there exists a nonzero polynomial $R_{u,v} \in \mathbb{K}[A]$ of degree at most $d(d-1)$ such that, for any $a \in \mathcal{U}(R_{u,v})$, there exists a nonzero polynomial $S_{u,v,a} \in \bar{\mathbb{K}}[C_1, \ldots, C_d]$ of total degree at most $d(d-1)/2$ such that, for any $(c_1, \ldots, c_d) \in \mathcal{U}(S_{u,v,a})$, the tree corresponding to $(u, v, a, c_1, \ldots, c_d)$ is executable on F.*

The maximum of the costs of the trees of the family belongs to

$$\mathcal{O}(d^{(\omega+3)/2} + d^{3/2}\mathsf{M}(d)(\mathsf{M}(d)^2/d^2 + \log(d))).$$

When using fast polynomial multiplication, that is $\mathsf{M}(d) \in \tilde{\mathcal{O}}(d)$, the preceding cost drops to $\mathcal{O}(d^{(\omega+3)/2})$.

At first sight, the cost estimates of these three theorems only make sense in characteristic 0: for a fixed field $\mathbb{K}$ of positive characteristic, Hypothesis (C) implies that the possible values for d are bounded. However, the constants hidden behind the $\mathcal{O}$ can be made independent of $\mathbb{K}$ if the costs of the linear algebra sub-routines are themselves independent of $\mathbb{K}$. For instance, this is possible with $\omega = 3$. We leave out these details in the sequel.

By the classical *Zippel-Schwartz lemma* [Zip93, Sch80], for any finite subset Z of $\mathbb{K}$, a nonzero polynomial P in n variables has at most $\deg(P)|Z|^{n-1}$ roots in Z^n, where $|Z|$ denotes the cardinality of Z. Therefore, Hypothesis (C) and the degree bounds given in Theorem 3 guarantee the existence of at least one tree of the family that is executable on F. This immediately leads to a deterministic algorithm but with a higher worst case cost than in Theorem 1. However, in practice, we recommend the probabilistic approach: it can be programmed in order to always return a correct answer and the average cost is expected to be much smaller than in Theorem 1.

Overview of the Algorithms. The deterministic and probabilistic algorithms share the same main ideas that are adapted from the rational factorization algorithms of [Lec04]. However this adaptation is not as direct as expected. Our algorithms combine advantages of Gao's algorithm [Gao03] and the classical *(Hensel) lifting and recombination* scheme. This scheme goes back to Zassenhaus [Zas69, Zas78] and is a cornerstone of the fastest rational factorization algorithms [Hoe02, BHKS04, Lec04, Lec05].

We now sketch out the main stages of the algorithms. At the beginning, the coordinates (x, y) are changed to sufficiently generic ones. The coordinates are changed back in each absolutely irreducible factor at the end of the computations. These operations are presented in Section 1. With suitable coordinates, the lifting and recombination scheme proceeds as follows:

1. *Lifting.* We compute a certain power series $\phi(x)$ solution of $F(x, \phi) = 0$ to a precision (x^σ), where σ depends linearly on d. These operations are described in Section 2.
2. *Recombination.* This stage divides into two main steps, namely:
   a. *Linear System Solving.* From the previous series ϕ, we construct a linear system whose basis of solutions has rank r and contains sufficient

information to deduce the absolutely irreducible factors. This step is presented in Section 3.

b. *Absolute Partial Fraction Decomposition.* From a basis of solutions of the previous system, we construct a polynomial $G \in \mathbb{K}[x,y]$ such that the absolutely irreducible factors of F can be easily deduced by computing the partial fraction decomposition of G/F in $\overline{\mathbb{K}(x)}(y)$. This step is detailed in Section 4.

This presentation privileges Zassenhaus' point of view but we shall see later that our algorithms are strongly related to Gao's algorithm [Gao03]. The factorization algorithms are completed in Section 5. In Section 6 we report on our implementation in Magma [Mag]. Before entering details, we briefly describe each stage so that the skeleton of the algorithms becomes clear.

Change of Coordinates. The algorithms start with changing the coordinates (x,y) in order to ensure the following Hypothesis (H):

$$\text{Hypothesis (H)} \quad \begin{cases} (i)\ F \text{ is monic in } y \text{ and } \deg_y(F) = \deg(F) = d, \\ (ii)\ \delta := \operatorname{Res}\left(F(0,y), \frac{\partial F}{\partial y}(0,y)\right) \neq 0. \end{cases}$$

Lifting. We introduce $f(y) := F(0,y)$ and $\mathbb{A} := \mathbb{K}[y]/(f(y))$. Let φ denote the residue class of y in $\mathbb{A}$. Under Hypothesis (H), there exists a unique series $\phi \in \mathbb{A}[[x]]$ such that $\phi - \varphi \in (x)$ and $F(x,\phi) = 0$. It is classical that ϕ can be approximated to any precision (x^σ) by means of Newton's operator.

Linear System Solving. From ϕ computed to the precision (x^σ), we construct a linear system from the coefficients of $\hat{\mathfrak{F}} := F/\mathfrak{F} \in \mathbb{A}[[x]][y]$, where $\mathfrak{F} := y - \phi$. The following definition constitutes the cornerstone of our algorithms:

$$\begin{aligned} L_\sigma := \Big\{ &((\ell_1,\dots,\ell_d), G, H) \in \mathbb{K}^d \times \mathbb{K}[x,y]_{d-1} \times \mathbb{K}[x,y]_{d-1} \mid \\ &G - \sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}\frac{\partial \mathfrak{F}}{\partial y}, \varphi^{i-1}\right) \in (x,y)^\sigma, \qquad (4) \\ &H - \sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}\frac{\partial \mathfrak{F}}{\partial x}, \varphi^{i-1}\right) \in (x,y)^\sigma + (x^{\sigma-1}) \Big\}. \qquad (5) \end{aligned}$$

Here $\operatorname{coeff}(B,\varphi^i)$ abusively represents $B_i \in \mathbb{K}[[x]][y]$ uniquely defined by $B = B_0 + B_1\varphi + \cdots + B_{d-1}\varphi^{d-1}$. The ideal $(x,y)^\sigma$ denotes the σth power of the ideal generated by x and y.

Although this construction does not look intuitive at first sight, we will see that it is directly related to the recombination technique introduced in [Lec04]. In addition, the notation is designed to be consistent with [Lec04] but observe that $\frac{\partial \mathfrak{F}}{\partial y} = 1$ and $\frac{\partial \mathfrak{F}}{\partial x} = -\phi'$.

For any $i \in \{1,\dots,r\}$, we introduce the partial product $\hat{F}_i := \prod_{j=1, j\neq i}^{r} F_j$ and $\operatorname{Tr}_j(f_i) := \sum_{f_i(\psi)=0} \psi^j$ that denotes the sum of the jth powers of the roots of $f_i := F_i(0,y)$. Lastly, π (resp. π_G) denotes the projection that maps $((\ell_1,\dots,\ell_d),G,H)$ to $(\ell_1,\dots,\ell_d)$ (resp. G). We naturally identify the extension $\bar{\mathbb{K}}\otimes L_\sigma$ with the space of solutions $((\ell_1,\dots,\ell_d),G,H) \in \bar{\mathbb{K}}^d \times \bar{\mathbb{K}}[x,y]_{d-1} \times \bar{\mathbb{K}}[x,y]_{d-1}$ of equations (4) and (5). For sufficiently large precisions σ, the vector spaces L_σ contain all the information about the absolute factorization of F. More precisely we have:

Theorem 4. *Under Hypotheses (C) and (H), for any $\sigma \geq 2d$, we have:*

$$\bar{\mathbb{K}}\otimes L_\sigma = \left\langle \left(\mu_i, \hat{F}_i\frac{\partial F_i}{\partial y}, \hat{F}_i\frac{\partial F_i}{\partial x}\right) \mid i \in \{1,\dots,r\} \right\rangle,$$

where $\mu_i := (\mathrm{Tr}_0(f_i), \ldots, \mathrm{Tr}_{d-1}(f_i))$.

For the sake of convenience, we introduce $L_\infty := L_\sigma$, for $\sigma \geq 2d$.

Absolute Partial Fraction Decomposition. We shall see how the previous theorem leads to a fast algorithm for computing a basis $G_1, \ldots, G_r$ of $\pi_G(L_\infty)$. Then we will prove that, for all G in a Zariski dense subset of $\pi_G(L_\infty)$, the monic squarefree part q of

$$Q(z) := \delta^{-1} \operatorname{Res}_y \left(F(0,y), z\frac{\partial F}{\partial y}(0,y) - G(0,y) \right)$$

has degree r (recall that $\delta = \operatorname{Res}_y(F(0,y), \frac{\partial F}{\partial y}(0,y))$). The roots of Q are the residues of G/F and the absolutely irreducible factors of F will be obtained by computing the following partial fraction decomposition in $\overline{\mathbb{K}(x)}(y)$:

$$\frac{G}{F} = \sum_{i=1}^{s} \sum_{q_i(\alpha)=0} \alpha \frac{\frac{\partial \mathsf{F}_i}{\partial y}(x,y,\alpha)}{\mathsf{F}_i(x,y,\alpha)},$$

where $q_1, \ldots, q_s$ denote the squarefree factors of Q. Finally, the absolute factorization returned by our algorithms is $(q_1, \mathsf{F}_1), \ldots, (q_s, \mathsf{F}_s)$.

Example 3. Before going further, we illustrate the computation of the absolute factorization of a small example: let $\mathbb{K} := \mathbb{Q}$ and $F := y^4 + (2x+14)y^2 - 7x^2 + 6x + 47$. Hypothesis (H) is satisfied, one has $f := y^4 + 14y^2 + 47$ and, with $\sigma := 2\deg(F) = 8$, we obtain:

$$\begin{aligned}
\phi = \varphi &- \left(\frac{13}{94}\varphi^3 + \frac{44}{47}\varphi\right) x + \left(\frac{39}{8836}\varphi^3 + \frac{199}{17672}\varphi\right) x^2 \\
&- \left(\frac{4745}{1661168}\varphi^3 + \frac{15073}{830584}\varphi\right) x^3 + \left(\frac{67665}{156149792}\varphi^3 + \frac{1231735}{624599168}\varphi\right) x^4 \\
&- \left(\frac{13201279}{58712321792}\varphi^3 + \frac{19943203}{14678080448}\varphi\right) x^5 \\
&+ \left(\frac{305810505}{5518958248448}\varphi^3 + \frac{3137922039}{11037916496896}\varphi\right) x^6 \\
&- \left(\frac{26241896109}{1037564150708224}\varphi^3 + \frac{76656876747}{518782075354112}\varphi\right) x^7 + \mathcal{O}(x^8).
\end{aligned}$$

A possible basis of $\pi(L_\infty)$ is $(1,0,0,0)$, $(0,0,1,0)$. The corresponding basis of $\pi_G(L_\infty)$ is given by $G_1 := y^3 + (15x+14)y$ and $G_2 := (2x+1)y$. When taking $G := G_2$, we obtain $s = 1$ and $Q(z) = q_1^2$, where $q_1 := z^2 - 1/32$. The absolute partial fraction decomposition of G/F yields the absolute factorization (q_1, F_1), where $\mathsf{F}_1(x,y,z) := y^2 + (1-16z)x - 8z + 7$. Remark that q_1 is irreducible, so is F.

Example 4. Let $\mathbb{K} := \mathbb{Q}$ and $F := y^6 + (-2x^2 + 2x + 14)y^4 + (-4x^3 - 35x^2 + 6x + 47)y^2 + 14x^4 - 12x^3 - 94x^2$. A possible basis of $\pi(L_\infty)$ is $(1,0,0,0,0,0)$, $(0,1,0,2,0,4)$ $(0,0,1,0,0,0)$, $(0,0,0,0,1,0)$. The corresponding basis of $\pi_G(L_\infty)$ is given by $G_1, \ldots, G_4$, where:

$$\begin{aligned}
G_1 &:= y^5 + \left(-\frac{36}{41}x^2 + \frac{172}{123}x + 14\right) y^3 + \left(-\frac{500}{123}x^3 - \frac{2935}{123}x^2 - \frac{3956}{123}x + 14\right) y, \\
G_2 &:= (x+1)y^4 + (2x^2 + 18x + 16)y^2 - 7x^3 - 15x^2 + 38x + 46, \\
G_3 &:= \left(\frac{16}{41}x^2 + \frac{14}{41}x + 1\right) y^3 + \left(\frac{68}{41}x^3 + \frac{286}{41}x^2 + \frac{416}{41}x + 14\right) y, \\
G_4 &:= \left(\frac{1}{41}x^2 - \frac{5}{246}x\right) y^3 + \left(\frac{23}{123}x^3 + \frac{179}{246}x^2 + \frac{119}{123}x + 1\right) y.
\end{aligned}$$

None of the absolute partial fraction decompositions of G_i/F, for $i \in \{1,\ldots,4\}$, yields the absolute factorization of F. With $G := G_1 + G_2$, we obtain $Q := q_1 q_2^2$, where $q_1 := z^2 - 23/41z - 623/13448$ and $q_2 := z^2 - 9/41z - 23/2952$. The rest of the computations yields $\mathsf{F}_1 := (-4z + 46/41)x + y$ and $\mathsf{F}_2 := (-984/49z + 157/49)x + y^2 - 492/49z + 397/49$. Remark that q_1 and q_2 are irreducible. We deduce that the rational irreducible factors of F are $y^2 - 2x^2$ and $y^4 + 2xy^2 + 14y^2 - 7x^2 + 6x + 47$. The latter factor corresponds to the previous example. Notice that its absolute factorization is represented differently here.

Related Work. Polynomial factorization is a central topic in computer algebra. Classical results can be found in [Zip93, GG03]. For comprehensive surveys and recent advances, we refer the reader to [Kal90, Kal92, Kal95, Kal03, Gao03, AGL04, Chè04b, Lec04, GKL04, Poh04, BLS+04, CG05, Lec05]. It is worth recalling that the multivariate factorization problem can be reduced to the bivariate case: this probabilistic reduction originated in computer algebra in [HS81], and is based on a quantitative version of Bertini's irreducibility theorem. We refer the reader to [Kal95, Gao03, CG05, Lec05] on this topic.

In the following paragraphs we discuss how our methods compare to the other absolute factorization algorithms. We start with the most related ones.

Trager's Reduction to Factoring over Algebraic Extensions. Rational factorization algorithms can be directly applied to the absolute factorization problem as soon as computations in $\bar{\mathbb{K}}$ or in a suitable splitting field are possible. For instance, when $\mathbb{K} = \mathbb{Q}$, numerical computations in $\mathbb{C}$ are possible and specific algorithms have been designed, including [SS93, Chè04a]. In general dynamic evaluation [Duv95, Del01, Ste02] can be used to simulate computations in $\bar{\mathbb{K}}$ but this solution is very expensive. On the other hand several algorithms exist for computing splitting fields but their cost intrinsically overwhelms the one of the absolute factorization.

In [Tra84] Trager suggested the following efficient strategy: the absolute factorization can be reduced to the rational factorization over a suitable algebraic extension that contains all the coefficients of a single absolutely irreducible factor. Such an extension can be constructed as the minimal algebraic extension $\mathbb{E}$ that contains the coordinates of a smooth point $(\alpha, \beta) \in \bar{\mathbb{K}}$ of the curve defined by $F(x, y) = 0$. Then, Trager's algorithm [Tra76] can be used to compute the factorization of F in $\mathbb{E}[x, y]$. Further developments of Trager's strategy can be found in [Tra85, DT89]. Such an extension $\mathbb{E}$ is also used in [Kal85] for testing the absolute irreducibility.

In various situations this strategy can be optimized. For example, Kaltofen's algorithm [Kal95, Section 2] computes the minimal polynomial over $\mathbb{E}$ of the power series expansion of the branch of the curve $F(x, y) = 0$ at (α, β). This way the rational factorization in $\mathbb{E}[x, y]$ via [Tra76] is avoided.

Our probabilistic algorithm is faster than the direct use of Trager's strategy with [Lec04]. In fact an optimized version of Trager's strategy adapted to the rational factorization algorithm of [Lec04] would proceed as follows. Let $e(y)$ be an irreducible factor of $F(0, y)$, let $\mathbb{E} := \mathbb{K}[y]/(e(y))$, $\alpha := 0$, and let β denote the residue class of y in $\mathbb{E}$. Thanks to Hypothesis (H), the point (α, β) is a smooth point of the curve $F(x, y) = 0$. First we factor $F(0, y)$ in $\mathbb{E}[y]$, and lift the resulting factors in $\mathbb{E}[[x]][y]$ to a certain precision linear in d. Secondly the irreducible factors over $\mathbb{E}$ are recombined from the lifted factors. The unique factor that vanishes at (α, β) is an absolutely irreducible factor of F. All the absolutely irreducible factors of F can be obtained this way.

For each factor e, the lifting stage takes $\tilde{\mathcal{O}}(d^2)$ operations in $\mathbb{E}$ [BLS+04, Theorem 2], hence the total cost of the lifting stages amounts to $\tilde{\mathcal{O}}(d^3)$ operations in

$\mathbb{K}$, which is greater than the complexity bound presented in Theorem 3. These theoretical estimates are confirmed by experiments in Section 6.

Notice that the conjugates of the absolutely irreducible factor found in $\mathbb{E}[x,y]$ via Trager's strategy are not necessarily pairwise distinct: a post-treatment is necessary in order to remove the casual redundancies. On the contrary, our algorithms directly compute a representation of the absolute factorization that is not redundant. Furthermore they do neither need to factor $F(0,y)$ in $\mathbb{K}[y]$ nor in $\mathbb{E}[y]$. Finally, the costs of our algorithms are very close to the ones given in [Lec04] for rational factorization: the cost $\tilde{\mathcal{O}}(d^4)$ (resp. $\mathcal{O}(d^{(\omega+3)/2})$) of Theorem 1 (resp. Theorem 3) is to be compared to $\mathcal{O}(d^{\omega+1})$ (resp. $\mathcal{O}(d^{\omega})$) of [Lec04, Proposition 1 (resp. Proposition 2)].

Duval's and Ragot's Algorithms. In Duval's algorithm [Duv91], one first computes a $\mathbb{K}$-basis $D_1,\dots,D_r$ of the algebraic closure of $\mathbb{K}$ in $\mathbb{K}(x)[y]/(F(x,y))$. Secondly, from a smooth point (α,β) of the curve $F(x,y)=0$, and by means of elementary linear algebra operations, one computes a $\mathbb{K}$-basis $\tilde{D}_1,\dots,\tilde{D}_{r-1}$ of the elements of $\langle D_1,\dots,D_r\rangle$ that vanish at (α,β). Lastly the greatest common divisor of $\tilde{D}_1,\dots,\tilde{D}_{r-1}$ and F is the unique absolutely irreducible factor of F that vanishes at (α,β). This comes from the fact that a $\bar{\mathbb{K}}$-basis of $\langle D_1,\dots,D_r\rangle$ is given by $\hat{F}_1 I_1,\dots,\hat{F}_r I_r$, where I_i denotes the inverse of $\hat{F}_i$ modulo F_i [Duv91, last remark of Section 3]. As with Trager's strategy, a post-treatment is necessary to remove the casual redundancies [Duv91, end of Section 1].

Ragot's algorithm [Rag97] computes the same basis $D_1,\dots,D_r$. Then the absolutely irreducible factors are recovered more efficiently by means of a variant of the Rothstein-Trager absolute partial fraction decomposition algorithm (see Appendix A). Let D be a polynomial in $\langle D_1,\dots,D_r\rangle$ with coordinates $\rho_1,\dots,\rho_r$ in the basis $\hat{F}_1 I_1,\dots,\hat{F}_r I_r$, so that we have $D=\rho_1\hat{F}_1 I_1+\cdots+\rho_r\hat{F}_r I_r$. Then the resultant $\mathrm{Res}_y(F(x,y),z-D(x,y))$ belongs to $\mathbb{K}[z]$ and its set of roots equals $\{\rho_1,\dots,\rho_r\}$. If the ρ_i are pairwise distinct then the absolute factorization of F can be easily deduced from the formulas $F_i=\gcd(F,\rho_i-D(x,y))$ for $i\in\{1,\dots,r\}$. Therefore Ragot's algorithm directly produces the same representation as ours. The basis $D_1,\dots,D_r$ plays exactly the same role as our basis $G_1,\dots,G_r$.

Ragot's method is probabilistic. It requires F to be irreducible and $\mathbb{K}$ perfect. Instead of the Rothstein-Trager algorithm we use the Lazard-Rioboo-Trager algorithm that avoids factorization in $\mathbb{K}[z]$. This way we do not require F to be irreducible and we only perform arithmetic operations in $\mathbb{K}$. Although Duval's and Ragot's algorithms have polynomial costs, the computation of $D_1,\dots,D_r$ is very expensive since it requires to calculate the ring of integers of $\mathbb{K}(x)[y]/(F(x,y))$.

Ruppert's and Gao's Algorithms. In characteristic zero, the absolute factorization can be obtained via the first algebraic De Rham cohomology group of the complementary of the curve $F(x,y)=0$. The first algorithm based on this idea was proposed by Ruppert for testing the absolute irreducibility [Rup86, Rup99]. In our context this group can be computed as the space of the closed differential forms

$$\frac{H}{F}dx+\frac{G}{F}dy,$$

where G and H belong to $\mathbb{K}[x,y]_{d-1}$ [Rup86, Satz 2]. In addition a $\bar{\mathbb{K}}$-basis of this group is given by

$$\left(\frac{\hat{F}_i\frac{\partial F_i}{\partial x}}{F}dx+\frac{\hat{F}_i\frac{\partial F_i}{\partial y}}{F}dy\right)_{i\in\{1,\dots,r\}}.$$

Ruppert's absolute irreducibility test consists in computing the rank r of the linear system

$$\frac{\partial}{\partial x}\left(\frac{G}{F}\right) = \frac{\partial}{\partial y}\left(\frac{H}{F}\right), \tag{6}$$

where the unknowns are the coefficients of G and H. This system has about d^2 unknown and about d^2 equations. Therefore a direct cost analysis of Ruppert's test yields $\mathcal{O}(d^{2\omega})$. From this test, Ruppert deduced degree bounds on the Noether irreducibility forms, and bounds on the height of the Ostrowski integers. A detailed presentation of these results can be found in Schinzel's book [Sch00, Chapter V].

In [Gao03], Gao proposed a complete probabilistic factorization algorithm based on these ideas, that even works for positive characteristics. Gao's conditions on the characteristic are mostly the same as in Hypothesis (C) (Gao deals with the bi-degree instead of the total degree). His algorithm computes both rational and absolute factorizations together with $\tilde{\mathcal{O}}(d^5)$ operations in $\mathbb{K}$ plus a few factorizations in $\mathbb{K}[z]$.

In our context Gao's algorithm can be presented as follows. It divides into two main stages:

1. *Linear System Solving.* One first computes a basis $(G_1, H_1), \ldots, (G_r, H_r)$ of solutions of system (6). Gao showed that this system can be solved by the black box approach with $\tilde{\mathcal{O}}(rd^4)$ operations in $\mathbb{K}$.
2. *Absolute Partial Fraction Decomposition.* If (G, H) is a solution of (6) then there exist $\rho_1, \ldots, \rho_r$ in $\bar{\mathbb{K}}$ such that

$$G = \rho_1 \hat{F}_1 \frac{\partial F_1}{\partial y} + \cdots + \rho_r \hat{F}_r \frac{\partial F_r}{\partial y}.$$

   If the ρ_i are pairwise distinct then the absolute factorization of F can be obtained from the absolute partial fraction decomposition of

$$\frac{G}{F} = \rho_1 \frac{\frac{\partial F_1}{\partial y}}{F_1} + \cdots + \rho_r \frac{\frac{\partial F_r}{\partial y}}{F_r}.$$

   Gao essentially follows the Rothstein-Trager algorithm (see [Gao03, Theorem 2.8]).

The representation of the absolute factorization is the same as ours. Thus our algorithms can be seen as an improvement of Gao's algorithm. In particular the lifting and recombination technique accelerates the resolution of (6).

The use of the absolute partial fraction decomposition was suggested in [Gao03, Section 4, Additional Remarks]. One of our contributions here is a deterministic algorithm for computing a suitable polynomial G in time $\tilde{\mathcal{O}}(d^4)$ (namely Algorithm 6 in Section 4). Another contribution is a Hensel lifting device to compute the absolute partial fraction decomposition from the one obtained with $x = 0$ (namely Algorithm 5 in Section 4). Finally let us mention that a numerical version of Gao's algorithm has recently been designed in [GKM+04].

Duval's, Ragot's, Gao's and our algorithms have the following point in common to Berlekamp's and Niederreiter's algorithms [Nie93, GG94, MŞ99] (for factoring univariate polynomials over finite fields): one first computes a basis of a certain vector space whose dimension equals the number of factors, then the factors are obtained by means of gcd or sub-resultants. In the next paragraphs, we mention other factorization algorithms that are less related to our methods.

Other Algorithms. In [CSTU02, Section 4.2] an improvement of Duval's and Ragot's algorithms is proposed: the expensive computation of the basis of $D_1, \ldots, D_r$ is replaced by the resolution of a system of linear differential equations. In [CSTU02,

Section 4.1] another factorization algorithm is investigated: the factorization is computed from the minimal differential operator associated to F. Improvements of these techniques are presented in [BT03]. The costs of these algorithms have not been analyzed yet.

Several algorithms have been designed for the special case $\mathbb{K} = \mathbb{Q}$. The use of the connectedness property of the irreducible components of the curve $F(x, y) = 0$ outside the singular locus is explored in [BCGW93]. Other strategies make use of the monodromy theory: the algorithms of Galligo and his collaborators [GR02, CGKW02, Rup04] perform mixed symbolic and numerical computations but the final result is always exact. In [Chè04a], these algorithms are improved thanks to the lattice reduction algorithm. In [SVW02, SVW04], these ideas are turned into a purely numerical algorithm that is well suited to homotopy continuation. These numerical methods are rather efficient in practice. Furthermore the exact factorization can always be recovered from a sufficiently accurate numerical one [CG04].

Lastly and less connected to our present concerns, we mention recent absolute irreducibility tests based on properties of Newton polytopes associated to F: [Gao01, GL01, Rag02, GR03, GL04].

1. Change of Coordinates

In this section, we show that, under Hypothesis (C), Hypothesis (H) is not restrictive. The results presented here are classical, we recall them briefly for completeness.

Let $F \in \mathbb{K}[x, y]$ be a squarefree polynomial of total degree d. We want to characterize the values u and v in $\mathbb{K}$ such that the monic part in y of $F(x+uy+v, y)$ satisfies Hypothesis (H). Let $F^{\#}$ denote the homogeneous component of $F \in \mathbb{K}[x, y]$ of highest degree d.

Lemma 1. *Under Hypothesis (C), for any $u \in \mathbb{K}$ such that $F^{\#}(u, 1) \neq 0$, the polynomial $F_u := F(x + uy, y)/F^{\#}(u, 1)$ is monic in y and*

$$\Delta_u(x) := \mathrm{Res}_y\left(F_u(x, y), \frac{\partial F_u}{\partial y}(x, y)\right) \neq 0.$$

For any $v \in \mathcal{U}(\Delta_u)$, the monic part in y of $F(x+uy+v, y)$ satisfies Hypothesis (H).

Proof. It is straightforward to check that $F^{\#}(u, 1)$ is the coefficient of y^d in F_u. Therefore G_u is monic in y. If we had $\Delta_u(x) = 0$ then F_u and $\frac{\partial F_u}{\partial y}$ would share a common irreducible factor $H \in \mathbb{K}[x, y]$ monic in y. Necessarily, one would have $\frac{\partial H}{\partial y} = 0$, which would contradict Hypothesis (C). □

The cost of the substitution of $x+uy+v$ for x is estimated in the following lemma, which will be used twice: at the beginning and at the end of the factorization algorithms. This is why we use a different notation.

Lemma 2. *Let $\mathbb{E}$ be a commutative unit ring, let H be in $\mathbb{E}[x, y]$ of total degree n and let u, v be in $\mathbb{E}$. If $n!$ is invertible in $\mathbb{E}$, and if we are given its inverse, then $H(x + uy + v, y)$ can be computed with $\mathcal{O}(n\mathsf{M}(n))$ operations in $\mathbb{E}$.*

Proof. First we prove that $H(x + uy, y)$ can be computed with the claimed cost. If H is homogeneous then $H(x + uy, y)$ is also homogeneous, thus it suffices to compute $H(x + u, 1)$ and to homogenize the result. The cost of this operation is dominated by the *shift* operation of the variable of a univariate polynomial, which is in $\mathcal{O}(\mathsf{M}(n))$, according to [BP94, Chapter 1, Section 2] (here we need the inverse of $n!$). If H is not homogeneous then we apply this process on its homogeneous

components, which yields a total cost in $\mathcal{O}(\mathsf{M}(1)+\mathsf{M}(2)+\cdots+\mathsf{M}(n))$. The super-additivity (3) of M implies $\mathsf{M}(i) \leq M(n)$, for any $i \leq n$, which concludes the case $v = 0$.

If $v \neq 0$ then we first compute $H(x+uy, y)$ and secondly $H(x+uy+v, y)$. Thus it remains to examine the case $u = 0$. This task corresponds to shifting the variable x in each coefficient of H seen in $\mathbb{E}[x][y]$. The total cost of these shifts is again in $\mathcal{O}(n\mathsf{M}(n))$. □

By Lemma 1, the number of values for u (resp. v) in $\mathbb{K}$ such that $F^{\#}(u,1) = 0$ (resp. $\Delta_u(v) = 0$) is at most d (resp. $d(d-1)$). Therefore, the existence of suitable values for u and v is guaranteed by Hypothesis (C). In practice, it is interesting to test values for u (resp. v) in increasing order in the range $[0, \ldots, d]$ (resp. $[0, \ldots, d(d-1)]$). Using fast multi-point evaluation, these tests can be performed efficiently, as explained in the proof of the following proposition:

Proposition 1. *For any squarefree polynomial $F \in \mathbb{K}[x, y]$ of total degree d such that Hypothesis (C) holds, one can compute u and v in $\mathbb{K}$ such that the monic part in y of $F(x+uy+v, y)$ satisfies Hypothesis (H) with $\mathcal{O}(d^2\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$.*

Proof. First we compute u such that $F^{\#}(u,1) \neq 0$: using a fast multi-point evaluation algorithm, one can compute all the $F^{\#}(i,1)$, for $i \in \{0, \ldots, d\}$, with $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$. Necessarily, one of these values is nonzero, which determines a suitable value for u.

In order to find a suitable value for v, we partition the set $Z := \{0, \ldots, d(d-1)\}$ into the subsets $Z_j := \{j(d+1), \ldots, \min((j+1)(d+1)-1, d(d-1))\}$, for $j \in \{0, \ldots, \lceil (d(d-1)+1)/(d+1) \rceil - 1\}$. This partition contains at most d subsets. For each Z_j, one can compute $\{F_u(i,y) \mid i \in Z_j\}$ with $\mathcal{O}(d\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$. Then one can deduce $\{\Delta_u(i) \mid i \in Z_j\}$ with $\mathcal{O}(d\mathsf{M}(d)\log(d))$. Therefore, the computation of $\{\Delta_u(i) \mid i \in Z\}$ costs $\mathcal{O}(d^2\mathsf{M}(d)\log(d))$. By Lemma 1, one of these values must be nonzero, which leads to a suitable value for v. □

2. Lifting

From now on and until the absolute factorization of F is computed, we assume that F satisfies Hypothesis (H). This section is devoted to the computation of an approximation of the series ϕ to a given precision (x^σ).

It is classical that this computation can be handled by means of Newton's operator [GG03, Algorithm 9.22]. Since the inverse of $\frac{\partial F}{\partial y}(0, \varphi)$ can be computed with $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$, and since each ring operation in $\mathbb{A}$ involves $\mathcal{O}(\mathsf{M}(d))$ operations in $\mathbb{K}$, we deduce from [GG03, Theorem 9.25] that the computation of ϕ to precision (x^σ) takes $\mathcal{O}(d\mathsf{M}(\sigma)\mathsf{M}(d))$ operations in $\mathbb{K}$. However we will show that [GG03, Algorithm 9.22] can be accelerated if we replace Horner's rule by Paterson and Stockmeyer's evaluation scheme [PS73].

2.1. Polynomial Evaluation.

Let $\mathbb{R}$ denote a commutative unit ring, and let $\mathbb{E}$ be a ring extension of $\mathbb{R}$ which is a free $\mathbb{R}$-module of dimension d. We assume that we know a basis $E_1, \ldots, E_d$ of $\mathbb{E}$ with $E_1 = 1$, and we denote by $E_1^*, \ldots, E_d^*$ the dual basis. From a computational point of view, we assume that the elements of $\mathbb{E}$ are represented by their coordinate vectors in the basis $E_1, \ldots, E_d$.

In this situation, Paterson and Stockmeyer's evaluation scheme is summarized in the following algorithm. For the only computation of ϕ, we could have directly used the version described in [GG03, Chapter 12.2] but for proving Corollary 2 below (that is used in Section 4.4), we need the following slightly stronger version:

Algorithm 1. *Paterson and Stockmeyer's evaluation scheme.*

Input: $P \in \mathbb{R}[y]$ with $\deg(P) \leq d$, and $e \in \mathbb{E}$.
Output: $P(e) \in \mathbb{E}$.

1. Let $k := \lfloor \sqrt{d+1} \rfloor$ and $k' := \lceil (d+1)/k \rceil$.
2. Compute $1, e, \ldots, e^{k-1}$.
3. Build the $d \times k$ matrix M with entries in $\mathbb{R}$ defined by $M_{i,j} := E_i^*(e^{j-1})$.
4. Build the $k \times k'$ matrix N with entries in $\mathbb{R}$ defined by
$$N_{i,j} := \operatorname{coeff}\left(P, y^{k(j-1)+i-1}\right).$$
5. Compute the $d \times k'$ matrix $C := MN$.
6. Let $D_i := \sum_{j=1}^{d} C_{j,i} E_j$, for $i \in \{1, \ldots, k'\}$.
7. Compute $1, e^k, \ldots, e^{k(k'-1)}$.
8. Return $\sum_{i=1}^{k'} D_i e^{k(i-1)}$.

In the following proposition we consider operations in $\mathbb{E}$ and matrix multiplication over $\mathbb{R}$ as black box subroutines that will be specified later in each case of use.

Proposition 2. *Algorithm 1 is correct and takes $\mathcal{O}(\sqrt{d})$ arithmetic operations in $\mathbb{E}$ and one matrix multiplication in size $d \times k$ times $k \times k'$ over $\mathbb{R}$.*

Proof. The correctness of the algorithm is a consequence of the following identities:
$$P(e) = \sum_{i=1}^{k'} D_i e^{k(i-1)} \text{ and } D_i = \sum_{j=1}^{k} \operatorname{coeff}(P, y^{k(i-1)+j-1}) e^{j-1} \text{ for } i \in \{1, \ldots, k'\}.$$
Since k and k' are in $\mathcal{O}(\sqrt{d})$, steps 2, 7 and 8 take $\mathcal{O}(\sqrt{d})$ operations in $\mathbb{E}$. The matrix multiplication of the proposition is the one of step 5 and the other computations are negligible. □

The following corollary is to be used in the next subsection. We carry on using the notation of Algorithm 1.

Corollary 1. *Let $\sigma \in \{1, \ldots, d(d-1)/2+1\}$, $\mathbb{R} := \mathbb{K}[[x]]/(x^\sigma)$, and $\mathbb{E} := \mathbb{R}[y]/(f(y)) = \mathbb{A}[[x]]/(x^\sigma)$. Then, under Hypothesis (C), Algorithm 1 takes*
$$\mathcal{O}\left(\sigma d^{(\omega+1)/2} + d^{3/2}\mathsf{M}(\sigma)(\mathsf{M}(d)/d + \log(d))\right)$$
operations in $\mathbb{K}$.

Proof. Since each ring operation in $\mathbb{E}$ takes $\mathcal{O}(\mathsf{M}(\sigma)\mathsf{M}(d))$ operations in $\mathbb{K}$, the conclusion follows from the previous proposition and Lemma 3 below. □

When using $\sigma \in \mathcal{O}(d)$ and a fast polynomial multiplication, that is $\mathsf{M}(d) \in \tilde{\mathcal{O}}(d)$, the cost of Algorithm 1 drops to $\mathcal{O}(d^{(\omega+3)/2})$ (recall that $\omega > 2$). Therefore, even with $\omega = 3$, Algorithm 1 is faster than Horner's rule by logarithmic factors. On the other hand, when using slow polynomial multiplication, that is $\mathsf{M}(d) \in \mathcal{O}(d^2)$, Algorithm 1 costs $\mathcal{O}(d^{4.5})$, whereas Horner's rule costs $\mathcal{O}(d^5)$.

The second corollary is used in Section 5.2, in order to test whether a candidate absolutely irreducible factor actually divides F or not.

Corollary 2. *Let $\sigma \in \{1, \ldots, d(d-1)/2+1\}$, $\mathbb{R} := \mathbb{K}[[x]]/(x^\sigma)$, $q \in \mathbb{K}[z]$, $r := \deg(q)$, $\mathsf{F} \in \mathbb{K}[x, y, z]$ and assume: r divides d, $\deg_z(\mathsf{F}) \leq r-1$, $\deg_x(\mathsf{F}) \leq \sigma - 1$, $\deg_y(\mathsf{F}) = d/r$, and F is monic in y. Let $\mathbb{E} := \mathbb{R}[y, z]/(q(z), \mathsf{F}(x, y, z))$ and let e denote the residue class of y in $\mathbb{E}$. Then, under Hypothesis (C), Algorithm 1 takes*
$$\mathcal{O}\left(\sigma d^{(\omega+1)/2} + d^{3/2}\mathsf{M}(\sigma)(\mathsf{M}(r)\mathsf{M}(d/r)/d + \log(d))\right)$$

operations in $\mathbb{K}$.

Proof. The basis $E_1, \ldots, E_d$ we consider for $\mathbb{E}$ is composed of the monomials $y^i z^j$ with $0 \leq i \leq d/r - 1$ and $0 \leq j \leq r - 1$. Each ring operation in $\mathbb{E}$ takes $\mathcal{O}(\mathsf{M}(\sigma)\mathsf{M}(r)\mathsf{M}(d/r))$ operations in $\mathbb{K}$. Again, the conclusion follows from the previous proposition and Lemma 3 below. □

The following lemma, which is used in the two previous corollaries, relies on classical techniques.

Lemma 3. *Under Hypothesis (C), for any* $\sigma \in \{1, \ldots, d(d-1)/2+1\}$*, the product of a* $d \times k$ *matrix by a* $k \times k'$ *matrix with entries in* $\mathbb{K}[[x]]/(x^\sigma)$ *can be computed with* $\mathcal{O}(\sigma d^{(\omega+1)/2} + d^{3/2}\mathsf{M}(\sigma)\log(d))$ *operations in* $\mathbb{K}$.

Proof. According to the definitions of k and k', this matrix multiplication reduces to multiplying $\mathcal{O}(\sqrt{d})$ matrices in sizes $\mathcal{O}(\sqrt{d} \times \sqrt{d})$ with entries in $\mathbb{K}[[x]]/(x^\sigma)$. Using fast multi-point evaluation and interpolation algorithms, each of these matrix products can be done with $\mathcal{O}(\sigma d^{\omega/2} + d\mathsf{M}(\sigma)\log(\sigma))$ operations in $\mathbb{K}$ this way: first we evaluate the entries of the matrices on $2\sigma - 1$ points taken in $\mathbb{K}$, then we perform the multiplications of the evaluated matrices, and lastly we interpolate the result. Thanks to Hypothesis (C) and since $\sigma \leq d(d-1)/2 + 1$, one can use the set $\{0, \ldots, 2(\sigma-1)\}$ for evaluation and interpolation. The claimed cost thus follows by replacing $\log(\sigma)$ by $\log(d)$. □

In the factorization algorithms, we shall take $\sigma \leq 2d + 1$, hence the algorithm used in this proof applies as soon as $d \geq 5$. The cost in Lemma 3 can be slightly improved as explained in [GG03, Note 12.2].

2.2. **Computation of** ϕ**.** We are now ready to study the complexity of Newton's operator combined with Paterson and Stockmeyer's evaluation scheme. Recall that our goal is the computation of ϕ to a given precision (x^σ). We closely follow [GG03, Section 9.4].

Algorithm 2. *Computation of* ϕ.

Input: $F \in \mathbb{K}[x, y]$ satisfying Hypothesis (H), and $\sigma \geq 1$.
Output: ϕ to precision (x^σ).

1. Compute the inverse I of $\frac{\partial F}{\partial y}(0, \varphi)$ in $\mathbb{A}$.
2. Let $\psi := \varphi$ and $\kappa := 1$.
3. While $\kappa < \sigma/2$ do
   a. Compute $\psi := \psi - IF(x, \psi)$ to precision (x^κ) (use Algorithm 1 to evaluate F at (x, ψ)).
   b. Compute $I := I + I(1 - I\frac{\partial F}{\partial y}(x, \psi))$ to precision (x^κ) (use Algorithm 1 to evaluate $\frac{\partial F}{\partial y}$ at (x, ψ)).
   c. $\kappa := 2\kappa$.
4. Let $\kappa := \sigma$ and compute $\psi := \psi - IF(x, \psi)$ to precision (x^κ) (use Algorithm 1 to evaluate F at (x, ψ)).
5. Return ψ.

In the calls to Algorithm 1, the polynomials F and $\frac{\partial F}{\partial y}$ are seen in $\mathbb{R}[y]$ where $\mathbb{R} := \mathbb{K}[[x]]/(x^\kappa)$, and $\mathbb{E}$ corresponds to $\mathbb{R}[y]/(f(y)) = \mathbb{A}[[x]]/(x^\kappa)$.

Proposition 3. *Under Hypotheses (C) and (H), for any* $\sigma \in \{1, \ldots, d(d-1)/2+1\}$*, Algorithm 2 is correct and takes*

$$\mathcal{O}\left(\sigma d^{(\omega+1)/2} + d^{3/2}\mathsf{M}(\sigma)(\mathsf{M}(d)/d + \log(d))\right)$$

operations in $\mathbb{K}$.

Proof. The correctness directly follows from [GG03, Theorem 9.23]. Step 1 costs $\mathcal{O}(\mathsf{M}(d)\log(d))$. At each step of the loop, the calls to Algorithm 1 dominate the complexity and cost $\mathcal{O}(\kappa d^{(\omega+1)/2}+d^{3/2}\mathsf{M}(\kappa)(\mathsf{M}(d)/d+\log(d)))$ operations in $\mathbb{K}$, by Corollary 1. By property (2) of M, we have $\mathsf{M}(\kappa)\leq \mathsf{M}(\sigma)\kappa/\sigma$. The conclusion thus follows by adding these costs over the successive values taken by κ. □

3. Linear System Solving

We still follow the notation of the introduction and we still assume that F satisfies Hypothesis (H). In this section, we prove Theorem 4 and present algorithms for computing bases of L_σ. The techniques presented here are adapted from [Lec04]: we will show that, up to a $\bar{\mathbb{K}}$-linear change of the variables $\ell_1,\dots,\ell_d$, the system defining L_σ coincides with the one introduced in [Lec04].

3.1. **Proof of Theorem 4.** Let $\phi_1,\dots,\phi_d$ represent the roots of F in $\bar{\mathbb{K}}[[x]]$, so that $F=\prod_{i=1}^d(y-\phi_i)$. In order to stay close to the notation of [Lec04], for $i\in\{1,\dots,d\}$, we introduce $\mathfrak{F}_i := y-\phi_i$ and the partial product

$$\hat{\mathfrak{F}}_i := \prod_{j=1,j\neq i}^{d} \mathfrak{F}_j.$$

To each $i\in\{1,\dots,r\}$, we associate the vector $\bar{\mu}_i\in\{0,1\}^d$ defined by

$$F_i = \prod_{j=1}^{d} \mathfrak{F}_j^{\bar{\mu}_{i,j}}. \tag{7}$$

If one knows all the ϕ_i to a sufficient precision, then the factorization of F reduces to computing the $\bar{\mu}_i$. This problem is efficiently solved in [Lec04] by means of the following vector space:

$$\begin{aligned}\bar{L}_\sigma :=\Big\{&((\bar{\ell}_1,\dots,\bar{\ell}_d),\bar{G},\bar{H})\in\bar{\mathbb{K}}^d\times\bar{\mathbb{K}}[x,y]_{d-1}\times\bar{\mathbb{K}}[x,y]_{d-1}\mid\\ &\bar{G}-\sum_{i=1}^{d}\bar{\ell}_i\hat{\mathfrak{F}}_i\frac{\partial\mathfrak{F}_i}{\partial y}\in(x,y)^\sigma,\\ &\bar{H}-\sum_{i=1}^{d}\bar{\ell}_i\hat{\mathfrak{F}}_i\frac{\partial\mathfrak{F}_i}{\partial x}\in(x,y)^\sigma+(x^{\sigma-1})\Big\}.\end{aligned}$$

Differentiating (7) with respect to x and y respectively gives:

$$\hat{F}_i\frac{\partial F_i}{\partial x}=\sum_{j=1}^{d}\bar{\mu}_{i,j}\hat{\mathfrak{F}}_j\frac{\partial\mathfrak{F}_j}{\partial x}\quad\text{and}\quad\hat{F}_i\frac{\partial F_i}{\partial y}=\sum_{j=1}^{d}\bar{\mu}_{i,j}\hat{\mathfrak{F}}_j\frac{\partial\mathfrak{F}_j}{\partial y},$$

whence the inclusion $\langle\bar{\mu}_1,\dots,\bar{\mu}_r\rangle\subseteq\pi(\bar{L}_\sigma)$. If σ is sufficiently large then this inclusion becomes an equality, as stated in:

Theorem 5. [Lec04, Theorem 1] *Under Hypotheses (C) and (H), for any $\sigma\geq 2d$ we have:*

$$\bar{L}_\sigma=\left\langle\left(\bar{\mu}_i,\hat{F}_i\frac{\partial F_i}{\partial y},\hat{F}_i\frac{\partial F_i}{\partial x}\right)\mid i\in\{1,\dots,r\}\right\rangle.$$

For shortness, we write $\varphi_i:=\phi_i(0)$ and we introduce the following isomorphism that sends φ to $(\varphi_1,\dots,\varphi_d)$:

$$\begin{aligned}\bar{\mathbb{K}}\otimes\mathbb{A}&\to\bar{\mathbb{K}}^d\\ b&\mapsto(b(\varphi_1),\dots,b(\varphi_d)).\end{aligned}$$

In the usual bases of $\bar{\mathbb{K}}[y]/(f(y)) = \bar{\mathbb{K}}\otimes\mathbb{A}$ and $\bar{\mathbb{K}}^d$, the matrix of this map is the Vandermonde matrix V of $(\varphi_1, \ldots, \varphi_d)$.

Proposition 4. *Under Hypothesis (H), for any $\sigma \geq 1$, the map*

$$\begin{array}{rccc} \Sigma : & \bar{L}_\sigma & \to & \bar{\mathbb{K}}\otimes L_\sigma \\ & ((\bar{\ell}_1, \ldots, \bar{\ell}_d), \bar{G}, \bar{H}) & \mapsto & (V^t(\bar{\ell}_1, \ldots, \bar{\ell}_d), \bar{G}, \bar{H}) \end{array} \tag{8}$$

is an isomorphism.

Proof. For any $b \in \bar{\mathbb{K}}\otimes\mathbb{A}$, $(\ell_1, \ldots, \ell_d) \in \bar{\mathbb{K}}^d$ and $(\bar{\ell}_1, \ldots, \bar{\ell}_d) \in \bar{\mathbb{K}}^d$ such that $(\ell_1, \ldots, \ell_d) = V^t(\bar{\ell}_1, \ldots, \bar{\ell}_d)$, one has:

$$\begin{aligned} \sum_{i=1}^{d} \ell_i \operatorname{coeff}(b, \varphi^{i-1}) &= \sum_{i=1}^{d} \operatorname{coeff}(b, \varphi^{i-1}) \sum_{j=1}^{d} \bar{\ell}_j \varphi_j^{i-1} \\ &= \sum_{j=1}^{d} \bar{\ell}_j \sum_{i=1}^{d} \operatorname{coeff}(b, \varphi^{i-1}) \varphi_j^{i-1} = \sum_{j=1}^{d} \bar{\ell}_j b(\varphi_j). \end{aligned}$$

On the other hand, it is straightforward to verify that substituting φ_i for φ in $\mathfrak{F}$ gives $\mathfrak{F}_i$. Therefore the map Σ is well-defined and is clearly an isomorphism. □

Since $\mu_i = V^t\bar{\mu}_i$, the proof of Theorem 4 directly follows from combining this proposition with Theorem 5.

In order to compute a basis of L_σ it suffices to compute a basis of $\pi(L_\sigma)$, which leads to consider a linear system in d unknowns. The rest of this section is devoted to the complexity analysis of this linear system solving. We first detail the natural deterministic method, and then we adapt the probabilistic speedup presented in [BLS+04, Lec04], which gains in reducing the number of equations.

3.2. **Deterministic Linear Solving.** From the approximation of ϕ to precision (x^σ), it is straightforward to compute a basis of $\pi(L_\sigma)$. For this purpose, we introduce the following linear system D_σ:

$$D_\sigma \begin{cases} \displaystyle\sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}\frac{\partial\mathfrak{F}}{\partial y}, \varphi^{i-1}x^j y^k\right) = 0, & k \leq d-1,\ d \leq j+k \leq \sigma-1, \\ \displaystyle\sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}\frac{\partial\mathfrak{F}}{\partial x}, \varphi^{i-1}x^j y^k\right) = 0, & k \leq d-1,\ d \leq j+k \leq \sigma-1, \\ & j \leq \sigma-2. \end{cases}$$

Lemma 4. *For all $\sigma \geq 1$, we have $\pi(L_\sigma) = \{(\ell_1, \ldots, \ell_d) \in \mathbb{K}^d \mid D_\sigma\}$.*

Proof. The proof is straightforward from the definition of $\pi(L_\sigma)$. □

The deterministic algorithm for computing a basis of $\pi(L_\sigma)$ proceeds as follows:

Algorithm 3. *Deterministic computation of a basis of $\pi(L_\sigma)$.*

Input: $F \in \mathbb{K}[x, y]$ satisfying Hypothesis (H), and ϕ to precision (x^σ).
Output: a basis of $\pi(L_\sigma)$.

1. Compute $\hat{\mathfrak{F}} = F/(y - \phi)$ to precision (x^σ) and build the linear system D_σ.
2. Compute and return a basis of solutions of D_σ.

Let us recall here that the computation of a solution basis of a linear system with m equations and $d \leq m$ unknowns over $\mathbb{K}$ takes

$$\mathcal{O}(md^{\omega-1}) \tag{9}$$

operations in $\mathbb{K}$ [BP94, Chapter 2] (see also [Sto00, Theorem 2.10]). We deduce the following complexity estimate:

Proposition 5. *For any integer $\sigma \geq 1$, Algorithm 3 is correct and takes $\mathcal{O}(\sigma d^\omega + d\mathsf{M}(\sigma)\mathsf{M}(d))$ operations in $\mathbb{K}$.*

Proof. The computation of $\hat{\mathfrak{F}}$ can be handled by means of the schoolbook division algorithm [GG03, Algorithm 2.5], and takes $\mathcal{O}(d)$ ring operations in $\mathbb{A}[[x]]/(x^\sigma)$, hence $\mathcal{O}(d\mathsf{M}(\sigma)\mathsf{M}(d))$ operations in $\mathbb{K}$. The cost of the construction of D_σ is negligible. The system D_σ has d unknowns and $\mathcal{O}(\sigma d)$ equations. It can thus be solved with $\mathcal{O}(\sigma d^\omega)$ operations in $\mathbb{K}$. □

3.3. **Probabilistic Linear Solving.** By Theorem 4, we shall take $\sigma = 2d$ in the deterministic factorization algorithm. To this precision, D_σ involves about d^2 equations. A classical trick for reducing the cost of the resolution of such an overdetermined system consists in replacing the original set of equations by fewer random linear combinations of them. In this subsection we adapt the fast probabilistic strategy of [Lec04, Section 3]. This strategy requires a slightly larger precision $\tau := \sigma + 1$ in the computation of ϕ.

For any $u \in \mathbb{K}$, we introduce the following linear system P_τ^u:

$$P_\tau^u \begin{cases} \displaystyle\sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}(x,ux)\frac{\partial \mathfrak{F}}{\partial x}(x,ux), \varphi^{i-1}x^j\right) = 0, \ d \leq j \leq \tau - 2, \\ \displaystyle\sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}(x,ux)\frac{\partial \mathfrak{F}}{\partial y}(x,ux), \varphi^{i-1}x^j\right) = 0, \ d \leq j \leq \tau - 2. \end{cases}$$

Of course, one has $\frac{\partial \mathfrak{F}}{\partial x}(x,ux) = -\phi'(x)$ and $\frac{\partial \mathfrak{F}}{\partial y}(x,ux) = 1$. For any $(a,b) \in \mathbb{K}^2$, $P_\tau^{a,b}$ represents the union of the two systems P_τ^a and P_τ^b. The following lemma shows how the solution set of $P_\tau^{a,b}$ relates to $\pi(L_\tau)$:

Lemma 5. *For any $(a,b) \in \mathbb{K}^2$, one has $\pi(L_\tau) \subseteq \{(\ell_1,\ldots,\ell_d) \in \mathbb{K}^d \mid P_\tau^{a,b}\}$. In addition, for any $b \in \mathbb{K}$, there exists a nonzero polynomial $R_b \in \mathbb{K}[A]$ of degree at most $d(d-1)$ such that the inclusion $\{(\ell_1,\ldots,\ell_d) \in \mathbb{K}^d \mid P_\tau^{a,b}\} \subseteq \pi(L_{\tau-1})$ holds for all $a \in \mathcal{U}(R_b)$.*

Proof. The proof closely follows the one of [Lec04, Lemma 5]. We introduce:

$$\begin{aligned} \Lambda_\tau := \Big\{ & (\ell_1,\ldots,\ell_d) \in \mathbb{K}^d \mid \\ & \sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}\frac{\partial \mathfrak{F}}{\partial x}, \varphi^{i-1}x^j y^k\right) = 0, \ k \leq d-1, \ d \leq j+k \leq \tau - 2, \\ & \sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}\frac{\partial \mathfrak{F}}{\partial y}, \varphi^{i-1}x^j y^k\right) = 0, \ k \leq d-1, \ d \leq j+k \leq \tau - 2\Big\}. \end{aligned}$$

Let A denote a new variable. By substituting Ax for y in the definition of Λ_τ, we get:

$$\begin{aligned} \Lambda_\tau = \Big\{ & (\ell_1,\ldots,\ell_d) \in \mathbb{K}^d \mid \\ & \sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}(x,Ax)\frac{\partial \mathfrak{F}}{\partial x}(x,Ax), \varphi^{i-1}x^j A^k\right) = 0, \ k \leq d-1, \ d \leq j \leq \tau - 2, \\ & \sum_{i=1}^{d} \ell_i \operatorname{coeff}\left(\hat{\mathfrak{F}}(x,Ax)\frac{\partial \mathfrak{F}}{\partial y}(x,Ax), \varphi^{i-1}x^j A^k\right) = 0, \ k \leq d-1, \ d \leq j \leq \tau - 2\Big\}. \end{aligned}$$

For any $u \in \mathbb{K}$, we define $\Lambda_\tau^u := \{(\ell_1,\ldots,\ell_d) \in \mathbb{K}^d \mid P_\tau^u\}$. Obviously, we have $\Lambda_\tau \subseteq \Lambda_\tau^a \cap \Lambda_\tau^b = \{(\ell_1,\ldots,\ell_d) \in \mathbb{K}^d \mid P_\tau^{a,b}\}$, for any a and b. Let $\lambda_1,\ldots,\lambda_{\dim(\Lambda_\tau^b)}$

be a basis of Λ_τ^b such that the $\dim(\Lambda_\tau)$ first vectors form a basis of Λ_τ. For any $\lambda_k \notin \Lambda_\tau$ there exists $j \in \{d, \dots, \tau-2\}$ such that one polynomial among

$$\sum_{i=1}^{d} \lambda_{k,i} \operatorname{coeff}\left(\hat{\mathfrak{F}}(x, Ax)\frac{\partial \mathfrak{F}}{\partial y}(x, Ax), \varphi^{i-1}x^j\right) \text{ and}$$
$$\sum_{i=1}^{d} \lambda_{k,i} \operatorname{coeff}\left(\hat{\mathfrak{F}}(x, Ax)\frac{\partial \mathfrak{F}}{\partial x}(x, Ax), \varphi^{i-1}x^j\right)$$

is nonzero: let $r_k(A)$ represent one of them which is nonzero. We take $R_b := \prod_{k=\dim(\Lambda_\tau)+1}^{\dim(\Lambda_\tau^b)} r_k$. Since each r_k has degree at most $d-1$, the polynomial R_b has degree at most $d(d-1)$. Finally, for any $a \in \mathbb{K}$ such that $R_b(a) \neq 0$, we have $\lambda_k \notin \Lambda_\tau^a$, whence $\Lambda_\tau = \Lambda_\tau^a \cap \Lambda_\tau^b$. The conclusion follows from the inclusions

$$\pi(L_\tau) \subseteq \Lambda_\tau \subseteq \pi(L_{\tau-1}),$$

which directly come from the definitions. □

The difficulty now relies in constructing the linear system $P_\tau^{a,b}$ efficiently. We start with the same idea as in [Lec04]: we attempt to compute $\hat{\mathfrak{F}}(x, ux)$ for $u \in \{a, b\}$ without performing the division of $F(x, y)$ by $y - \phi(x)$. We try to invert $ux - \phi(x)$: if it is not invertible then we split the computation. For this purpose, we introduce $f_y := \gcd(f, y)$, $\hat{f}_y := f/f_y$, $\mathbb{A}_y := \mathbb{K}[y]/(f_y)$, $\hat{\mathbb{A}}_y := \mathbb{K}[y]/(\hat{f}_y)$. Let $\mathfrak{A}$ denote the usual isomorphism $\mathbb{A} \to \mathbb{A}_y \times \hat{\mathbb{A}}_y$. If f_y is constant then $\mathbb{A}_y = \{0\}$ and $\mathbb{A} = \hat{\mathbb{A}}_y$. This corresponds to the case when φ is invertible in $\mathbb{A}$ or, equivalently ϕ invertible in $\mathbb{A}[[x]]$. Otherwise $f_y = y$ and one has $\mathbb{A}_y = \mathbb{K}$.

For any $\bar{\beta} \in \mathbb{A}$, the computation of $\mathfrak{A}(\bar{\beta})$ takes $\mathcal{O}(\mathsf{M}(d))$ operations in $\mathbb{K}$. For any $(\bar{\beta}_1, \bar{\beta}_2) \in \mathbb{A}_y \times \hat{\mathbb{A}}_y$, the computation of $\mathfrak{A}^{-1}(\bar{\beta}_1, \bar{\beta}_2)$ takes $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$, by [GG03, Corollary 10.23]. This cost can be slightly improved when taking into account the specificity of the situation. Let β_1 and β_2 denote the respective canonical preimages of $\bar{\beta}_1$ and $\bar{\beta}_2$ in $\mathbb{K}[y]$. The preimage $\beta \in \mathbb{K}[y]$ of $\mathfrak{A}^{-1}(\bar{\beta}_1, \bar{\beta}_2)$ is given by the following formulas: $\beta = \beta_1 + y\beta_3$, where β_3 is the preimage of $(\beta_2 - \beta_1)/y$ computed in $\hat{\mathbb{A}}_y$. The computation of the inverse of y in $\hat{\mathbb{A}}_y$ only takes $\mathcal{O}(d)$ operations in $\mathbb{K}$. Finally, the computation of $\mathfrak{A}^{-1}(\bar{\beta}_1, \bar{\beta}_2)$ only costs $\mathcal{O}(\mathsf{M}(d))$.

With a slight abuse of notation, we still write $\mathfrak{A}$ for the natural extension of $\mathfrak{A}$ to $\mathbb{A}[[x]] \to \mathbb{A}_y[[x]] \times \hat{\mathbb{A}}_y[[x]]$ that maps $\mathfrak{A}$ coefficient by coefficient. Lastly, we write $\mathfrak{A}_y$ (resp. $(\hat{\mathfrak{A}}_y)$) for the first (resp. second) projection of $\mathfrak{A}$, so that $\mathfrak{A} = (\mathfrak{A}_y, \hat{\mathfrak{A}}_y)$. We are now ready to present the probabilistic algorithm to compute a basis of $\pi(L_\sigma)$.

Algorithm 4. *Probabilistic computation of a basis of* $\pi(L_\sigma)$.

Input: $F \in \mathbb{K}[x, y]$ satisfying Hypothesis (H), ϕ to precision (x^τ), and $(a, b) \in \mathbb{K}^2$.

Output: a basis of $\{(\ell_1, \dots, \ell_d) \in \mathbb{K}^d \mid P_\tau^{a,b}\}$.

1. For each $u \in \{a, b\}$ do
   a. In $\hat{\mathbb{A}}_y[[x]]$ compute $\hat{\mathfrak{f}}_y := F(x, ux)/\hat{\mathfrak{A}}_y(ux - \phi)$ to precision $(x^{\tau-1})$.
   b. In $\mathbb{A}_y[[x]][y]$ compute the quotient T of $F(x, y)$ by $y - \mathfrak{A}_y(\phi)$ to precision $(x^{\tau-1})$ and let $\mathfrak{f}_y := T(x, ux)$.
   c. Compute $\hat{\mathfrak{F}}(x, ux)$ as $\mathfrak{A}^{-1}(\mathfrak{f}_y, \hat{\mathfrak{f}}_y)$ to precision $(x^{\tau-1})$.
   d. Compute $\hat{\mathfrak{F}}(x, ux)\phi'(x)$ to precision $(x^{\tau-1})$.
2. Compute and return a basis of $\{(\ell_1, \dots, \ell_d) \in \mathbb{K}^d \mid P_\tau^{a,b}\}$.

Proposition 6. *For any* $\tau \geq d$, *Algorithm 4 is correct and takes* $\mathcal{O}(\tau d^{\omega-1} + \mathsf{M}(\tau)\mathsf{M}(d))$ *operations in* $\mathbb{K}$.

Proof. Since $\mathfrak{F}$ exactly divides F and since $\hat{\mathfrak{A}}_y(ux-\phi)$ is invertible, $\hat{\mathfrak{f}}_y$ equals $\hat{\mathfrak{A}}_y(\hat{\mathfrak{F}}(x,ux))$ to precision $(x^{\tau-1})$. On the other hand, $y-\mathfrak{A}_y(\phi)$ divides F hence $\mathfrak{f}_y$ equals $\mathfrak{A}_y(\hat{\mathfrak{F}}(x,ux))$ to precision $(x^{\tau-1})$. Thus the algorithm works correctly.

The computation of $\mathfrak{A}(\phi)$ take $\mathcal{O}(\tau\mathsf{M}(d))$ operations in $\mathbb{K}$. Thanks to Newton's iteration [GG03, Chapter 9.1], the series inversion in step 1a takes $\mathcal{O}(\mathsf{M}(d)(\log(d)+\mathsf{M}(\tau)))$ operations in $\mathbb{K}$. The cost of step 1b belongs to $\mathcal{O}(d\mathsf{M}(\tau))$. Step 1c costs $\mathcal{O}(\tau\mathsf{M}(d))$ and step 1d costs $\mathcal{O}(\mathsf{M}(\tau)\mathsf{M}(d))$. Since the linear system $P_\tau^{a,b}$ contains $\mathcal{O}(\tau)$ equations, its resolution in step 2 requires $\mathcal{O}(\tau d^{\omega-1})$ operations in $\mathbb{K}$, by bound (9). □

4. Absolute Partial Fraction Decomposition

In this section, we assume that we are given a candidate basis of $\pi(L_\infty)$. We explain how the absolutely irreducible factors can be constructed via a suitable partial fraction decomposition. For the sake of completeness and because we deal with positive characteristic, the classical absolute partial fraction decomposition algorithms of Lazard, Rioboo, Rothstein and Trager are revisited in Appendix A.

4.1. Existence of the Representation of the Absolute Factorization.

Let $\nu_1,\dots,\nu_r$ be a basis of $\pi(L_\infty)$. For each ν_i, there exist unique polynomials $G_i \in \mathbb{K}[x,y]_{d-1}$ and $H_i \in \mathbb{K}[x,y]_{d-1}$ such that $(\nu_i,G_i,H_i)\in L_\infty$. These polynomials can be computed by means of the following formulas, in which the series can be truncated to precision (x^{d+1}):

$$G_i = \sum_{j=1}^{d} \nu_{i,j}\,\mathrm{coeff}\left(\hat{\mathfrak{F}}\frac{\partial\mathfrak{F}}{\partial y},\varphi^{j-1}\right), \tag{10}$$
$$H_i = \sum_{j=1}^{d} \nu_{i,j}\,\mathrm{coeff}\left(\hat{\mathfrak{F}}\frac{\partial\mathfrak{F}}{\partial x},\varphi^{j-1}\right).$$

Since $(\nu_1,G_1,H_1),\dots,(\nu_r,G_r,H_r)$ is a basis of L_∞, we deduce from Theorem 4 that their exists an invertible $r\times r$ matrix $(\rho_{i,j})_{(i,j)}$ with entries in $\bar{\mathbb{K}}$ such that:

$$(\nu_i,G_i,H_i) = \sum_{j=1}^{r}\rho_{j,i}\left(\mu_j,\hat{F}_j\frac{\partial F_j}{\partial y},\hat{F}_j\frac{\partial F_j}{\partial x}\right), \quad \text{for all } i\in\{1,\dots,r\}. \tag{11}$$

In particular, the set of row vectors $\{\rho_1,\dots,\rho_r\}$ has cardinality r. We say that a point $(c_1,\dots,c_r)$ in $\mathbb{K}^r$ *separates* $\rho_1,\dots,\rho_r$ when the dot products $(c_1,\dots,c_r)\cdot\rho_1,\dots,(c_1,\dots,c_r)\cdot\rho_r$ are pairwise distinct. The following lemma is going to lead to an estimate on the density of such points.

Lemma 6. *There exists a nonzero polynomial $S\in\bar{\mathbb{K}}[C_1,\dots,C_r]$ of total degree $r(r-1)/2$ such that any $(c_1,\dots,c_r)\in\mathcal{U}(S)$ separates $\rho_1,\dots,\rho_r$.*

Proof. The following polynomial clearly suits us:

$$S := \prod_{1\le i<j\le r}\left(\sum_{k=1}^{r}(\rho_{i,k}-\rho_{j,k})C_k\right). \qquad \square$$

Under Hypothesis (C), the subset $Z:=\{0,\dots,d(d-1)\}\subseteq\mathbb{K}$ has cardinality $d(d-1)+1$. By the Zippel-Schwartz lemma [Zip93, Sch80], the cardinality of $\mathcal{U}(S)\cap Z^r$ is at most $|Z|^{r-1}r(r-1)/2 < |Z|^r/2$ (since $r\le d$). In other words, the proportion of points in Z^r that separate $\rho_1,\dots,\rho_r$ is greater than $1/2$.

Let $(c_1, \ldots, c_r) \in \mathbb{K}^r$ and let $G := c_1G_1 + \cdots + c_rG_r$. The absolute partial fraction decomposition of G/F seen in $\mathbb{K}(x)(y)$ can be written in the following form:

$$\frac{G}{F} = \sum_{i=1}^{r} ((c_1, \ldots, c_r) \cdot \rho_i) \frac{\frac{\partial F_i}{\partial y}(x, y)}{F_i(x, y)}.$$

If $(c_1, \ldots, c_r)$ separate $\rho_1, \ldots, \rho_r$ then we deduce that the Lazard-Rioboo-Trager absolute partial fraction decomposition algorithm (Algorithm 12 of Appendix A) called with input G/F returns $(q_1, \mathsf{F}_1), \ldots, (q_s, \mathsf{F}_s)$ such that:

$$\frac{G}{F} = \sum_{i=1}^{s} \sum_{q_i(\alpha)=0} \alpha \frac{\frac{\partial \mathsf{F}_i}{\partial y}(x, y, \alpha)}{\mathsf{F}_i(x, y, \alpha)}, \tag{12}$$

and

- q_i is a monic squarefree polynomial in $\mathbb{K}[z]$ of degree $r_i \geq 1$;
- F_i belongs to $\mathbb{K}(x)[y, z]$, is monic in y, and satisfies $\deg_z(\mathsf{F}_i) \leq r_i - 1$;
- $r_1 + \cdots + r_s = r$ and $\{F_1, \ldots, F_r\} = \bigcup_{i=1}^{s} \{\mathsf{F}_i(x, y, \alpha) \mid q_i(\alpha) = 0\}$.

We deduce that F_i belongs to $\mathbb{K}[x, y, z]$. Finally $(q_1, \mathsf{F}_1), \ldots, (q_s, \mathsf{F}_s)$ represent the absolute factorization of F.

For efficiency, we do not call the Lazard-Rioboo-Trager algorithm in $\mathbb{K}(x)[y]$. Instead we compute the decomposition with $x = 0$ first and then we lift it. The lifting device is presented in the next subsection.

4.2. **Absolute Multi-factor Hensel Lifting.** In this subsection we assume that we are given a factorization (not necessarily irreducible) of f in $\bar{\mathbb{K}}[y]$ and we wish to lift it as a factorization of F in $\bar{\mathbb{K}}[[x]][y]$ to a given precision. The input factorization is assumed to be given in the following form:

$$f = \prod_{i=1}^{s} \prod_{q_i(\alpha)=0} \mathsf{f}_i(y, \alpha),$$

where

- q_i is a monic squarefree polynomial of $\mathbb{K}[z]$ of degree $r_i \geq 1$;
- f_i belongs to $\mathbb{K}[y, z]$, is monic in y, and satisfies $\deg_z(\mathsf{f}_i) \leq r_i - 1$;
- $r_1 + \cdots + r_s = r$.

Because we are not explicitly given a common field for the coefficients of all the factors $\mathsf{f}_i(y, \alpha)$, we can not directly apply the multi-factor Hensel lifting algorithm given in [GG03, Algorithm 15.17]. For each $i \in \{1, \ldots, s\}$, we introduce

$$p_i(y) := \mathrm{Res}_z(q_i(z), \mathsf{f}_i(y, z)) = \prod_{q_i(\alpha)=0} \mathsf{f}_i(y, \alpha) \in \mathbb{K}[y]$$

and $d_i := \deg(p_i) = r_i \deg_y(\mathsf{f}_i)$. Observe that $d_1 + \cdots + d_s = d$. Let $\mathbb{E}_i := \mathbb{K}[z]/(q_i(z))$ and let α_i denote the residue class of z in $\mathbb{E}_i$. The following lemma serves us to define the objects we are to lift:

Lemma 7. *Let η be an integer such that $d+1 \geq \eta \geq \max(d_i/r_i \mid i \in \{1, \ldots, s\}) + 1$. Assume that Hypothesis (H) holds.*

a. *There exist unique polynomials $\tilde{P}_1, \ldots, \tilde{P}_s$ in $\mathbb{K}[x, y]$ such that:*
   - *$\tilde{P}_i$ is monic in y, $\deg_x(\tilde{P}_i) \leq \eta - 1$, $\deg_y(\tilde{P}_i) = d_i$ and $\tilde{P}_i - p_i \in (x)$, for each $i \in \{1, \ldots, s\}$;*
   - *$F - \tilde{P}_1 \cdots \tilde{P}_s \in (x^\eta)$.*

b. *There exist unique polynomials $\tilde{\mathsf{F}}_1, \ldots, \tilde{\mathsf{F}}_s$ in $\mathbb{K}[x, y, z]$ such that, for all $i \in \{1, \ldots, s\}$:*

– $\tilde{\mathsf{F}}_i$ *is monic in* y, $\deg_x(\tilde{\mathsf{F}}_i) \le d_i/r_i$, $\deg_y(\tilde{\mathsf{F}}_i) = d_i/r_i$, $\deg_z(\tilde{\mathsf{F}}_i) \le r_i - 1$ *and* $\tilde{\mathsf{F}}_i - \mathsf{f}_i \in (x)$;
– $\tilde{\mathsf{F}}_i(x, y, \alpha_i)$ *divides* $\tilde{P}_i$ *in* $\mathbb{E}_i[[x]]/(x^{d_i/r_i+1})[y]$.

Proof. The proof relies on classical arguments (use [GG03, Theorem 15.14] for instance). □

The lifting algorithm starts with lifting all the p_i before lifting each f_i separately with the help of $\tilde{P}_i$. The computation of $\tilde{P}_1, \dots, \tilde{P}_s$ is classical: it can be directly handled by the multi-factor Hensel lifting algorithm [GG03, Algorithm 15.17]. However, the computation of $\tilde{\mathsf{F}}_i$ requires more effort in order to avoid factoring q_i: in general, $\mathbb{E}_i$ is not a field.

Since $\mathsf{f}_i(y, \alpha_i)$ is monic in y, the quotient $p_i(y)/\mathsf{f}_i(y, \alpha_i)$ is well-defined in $\mathbb{E}_i[y]$, and we denote by $\hat{\mathsf{f}}_i(y, z)$ its canonical preimage in $\mathbb{K}[y, z]$, so that we have $\deg_z(\hat{\mathsf{f}}_i) \le r_i - 1$ and $p_i(y) = \mathsf{f}_i(y, \alpha_i)\hat{\mathsf{f}}_i(y, \alpha_i)$. Since $\mathsf{f}_i(y, \alpha)$ and $\hat{\mathsf{f}}_i(y, \alpha)$ are coprime for each root $\alpha \in \bar{\mathbb{K}}$ of q_i, there exist unique polynomials $v_i(y, z)$ and $w_i(y, z)$ in $\mathbb{K}[y, z]$ such that $\deg_z(v_i) \le r_i - 1$, $\deg_z(w_i) \le r_i - 1$, $\deg_y(v_i) \le \deg_y(\hat{\mathsf{f}}_i) - 1$, $\deg_y(w_i) \le \deg_y(\mathsf{f}_i) - 1$, and

$$v_i(y, \alpha_i)\mathsf{f}_i(y, \alpha_i) + w_i(y, \alpha_i)\hat{\mathsf{f}}_i(y, \alpha_i) = 1.$$

The polynomials v_i and w_i can be deduced from the Bézout identity between f_i and $\hat{\mathsf{f}}_i$ in $\mathbb{K}(z)[y]$ but it is faster to compute the Bézout identity between $\mathsf{f}_i(y, \alpha_i)$ and $\mathrm{rem}(\hat{\mathsf{f}}_i(y, \alpha_i), \mathsf{f}_i(y, \alpha_i))$. More precisely, we introduce the polynomials g_i and h_i in $\mathbb{K}[y, z]$ respectively defined as the preimages of the quotient and the remainder of $\hat{\mathsf{f}}_i(y, \alpha_i)$ divided by $\mathsf{f}_i(y, \alpha_i)$, so that we have:

- $\deg_z(\mathsf{g}_i) \le r_i - 1$, $\deg_z(\mathsf{h}_i) \le r_i - 1$;
- $\deg_y(\mathsf{h}_i) \le d_i/r_i - 1$;
- $\hat{\mathsf{f}}_i(y, \alpha_i) = \mathsf{g}_i(y, \alpha_i)\mathsf{f}_i(y, \alpha_i) + \mathsf{h}_i(y, \alpha_i)$.

Since $\mathsf{f}_i(y, \alpha)$ and $\mathsf{h}_i(y, \alpha)$ are coprime for all root α of q_i, the polynomials f_i and h_i are coprime in $\mathbb{K}(z)[y]$. Therefore there exist two polynomials $\tilde{v}_i(y, z)$ and $\tilde{w}_i(y, z)$ in $\mathbb{K}(z)[y]$ such that $\tilde{v}_i\mathsf{f}_i + \tilde{w}_i\mathsf{h}_i = 1$, $\deg_y(\tilde{v}_i) \le \deg_y(\mathsf{h}_i) - 1$ and $\deg_y(\tilde{w}_i) \le \deg_y(\mathsf{f}_i) - 1$.

By [GG03, Theorem 6.55], the denominators of $\tilde{v}_i$ and $\tilde{w}_i$ do not vanish at any root α of q_i. Therefore we obtain:

$$\tilde{v}_i(y, \alpha_i)\mathsf{f}_i(y, \alpha_i) + \tilde{w}_i(y, \alpha_i)\mathsf{h}_i(y, \alpha_i) = 1.$$

We deduce:

$$(\tilde{v}_i(y, \alpha_i) - \tilde{w}_i(y, \alpha_i)\mathsf{g}_i(y, \alpha_i))\mathsf{f}_i(y, \alpha_i) + \tilde{w}_i(y, \alpha_i)\hat{\mathsf{f}}_i(y, \alpha_i) = 1,$$

hence

$$v_i(y, \alpha_i) = \tilde{v}_i(y, \alpha_i) - \tilde{w}_i(y, \alpha_i)\mathsf{g}_i(y, \alpha_i) \quad \text{and} \quad w_i(y, \alpha_i) = \tilde{w}_i(y, \alpha_i). \tag{13}$$

These formulas lead to the following lifting algorithm:

Algorithm 5. *Absolute multi-factor Hensel lifting.*

Input: $F \in \mathbb{K}[x, y]$ such that Hypotheses (C) and (H) hold, and $(q_1, \mathsf{f}_1), \dots, (q_s, \mathsf{f}_s)$, η as in Lemma 7.

Output: $\tilde{P}_1, \dots, \tilde{P}_s$ and $\tilde{\mathsf{F}}_1, \dots, \tilde{\mathsf{F}}_s$ as defined in Lemma 7.

1. For each $i \in \{1, \dots, s\}$ compute $p_i(y) := \mathrm{Res}_z(q_i(z), \mathsf{f}_i(y, z))$.
2. Use the multi-factor Hensel lifting algorithm [GG03, Algorithm 15.17] with input $p_1, \dots, p_s$, F, and required precision (x^η) in order to obtain the polynomials $\tilde{P}_1, \dots, \tilde{P}_s$.
3. For each $i \in \{1, \dots, s\}$ do

a. Compute $\hat{\mathsf{f}}_i(y,\alpha_i) := p_i(y)/\mathsf{f}_i(y,\alpha_i)$.
b. Compute the quotient $\mathsf{g}_i(y,\alpha_i)$ and the remainder $\mathsf{h}_i(y,\alpha_i)$ in the division of $\hat{\mathsf{f}}_i(y,\alpha_i)$ by $\mathsf{f}_i(y,\alpha_i)$.
c. Compute $\tilde{v}_i(y,z)$ and $\tilde{w}_i(y,z)$ by means of the modular Euclidean algorithm [GG03, Corollary 11.9] with input $\mathsf{f}_i(y,z)$ and $\mathsf{h}_i(y,z)$ seen in $\mathbb{K}(z)[y]$.
d. Compute v_i and w_i by means of formulas (13).
e. Use the multi-factor Hensel lifting algorithm [GG03, Algorithm 15.17] with input $\mathsf{f}_i(y,\alpha_i)$, $\hat{\mathsf{f}}_i(y,\alpha_i)$, $\tilde{P}_i$, and required precision (x^{d_i/r_i+1}) in order to obtain the polynomial $\tilde{\mathsf{F}}_i$. By using $v_i(y,\alpha_i)$ and $w_i(y,\alpha_i)$, we can skip the computation of the Bézout identity in step 4 of [GG03, Algorithm 15.17].

4. Return $\tilde{P}_1,\dots,\tilde{P}_s$ and $\tilde{\mathsf{F}}_1,\dots,\tilde{\mathsf{F}}_s$.

Proposition 7. *Algorithm 5 is correct and takes* $\mathcal{O}(\mathsf{M}(d)^2(\mathsf{M}(d)/d+\log(d)))$ *operations in* $\mathbb{K}$.

Proof. The correctness of steps 2 and 3e follows from [GG03, Theorem 15.18]. It remains to analyze the cost.

Let $i\in\{1,\dots,s\}$. We start with the cost of the calculation of p_i. Thanks to Hypothesis (C), it suffices to interpolate p_i from its values on $Z_i := \{0,\dots,d_i\}$. Using fast evaluation, all the values $\mathsf{f}_i(j,z)$, for $j\in Z_i$, can be computed with $\mathcal{O}(r_i^2\mathsf{M}(d_i/r_i)\log(d_i/r_i))$ operations in $\mathbb{K}$. Then, each value $\mathrm{Res}_z(q_i(z),\mathsf{f}_i(j,z))$ can be computed with $\mathcal{O}(\mathsf{M}(r_i)\log(r_i))$ operations. The interpolation of p_i costs $\mathcal{O}(\mathsf{M}(d_i)\log(d_i))$. Finally the cost of step 1 belongs to

$$\begin{aligned}
&\mathcal{O}\left(\sum_{i=1}^{s} r_i^2\mathsf{M}(d_i/r_i)\log(d_i/r_i)+\sum_{i=1}^{s} d_i\mathsf{M}(r_i)\log(r_i)+\sum_{i=1}^{s}\mathsf{M}(d_i)\log(d_i)\right)\\
&\quad\subseteq \mathcal{O}\left(\sum_{i=1}^{s} r_i^2\mathsf{M}(d_i/r_i)\log(d)+d\mathsf{M}(d)\log(d)\right)\\
&\quad\subseteq \mathcal{O}\left(\sum_{i=1}^{s} r_i\mathsf{M}(d_i)\log(d)+d\mathsf{M}(d)\log(d)\right)\quad\text{(by assumption (2) on }\mathsf{M})\\
&\quad\subseteq \mathcal{O}\left(d\mathsf{M}(d)\log(d)\right).
\end{aligned}$$

By [GG03, Theorem 15.18], step 2 takes $\mathcal{O}(\mathsf{M}(d)^2\log(s))$ operations in $\mathbb{K}$ (recall that $\eta\le d+1$).

Steps 3a and 3b take $\mathcal{O}(\mathsf{M}(r_i)\mathsf{M}(d_i))$ operations in $\mathbb{K}$. Except for a finite number of values for d, Hypotheses (C) ensures that we can apply the fast modular Euclidean algorithm [GG03, Corollary 11.9] in step 3c, whence $\mathcal{O}(d_i/r_i\mathsf{M}(d_i)\log(d_i))$ operations in $\mathbb{K}$.

By [GG03, Theorem 6.54], the coefficients of $\tilde{v}_i$ and $\tilde{w}_i$ have numerators and denominators of degree in z at most $2d_i$. Therefore the substitution of α_i for z in $\tilde{v}_i$ and $\tilde{w}_i$ costs $\mathcal{O}(d_i/r_i\mathsf{M}(d_i)+d_i/r_i\mathsf{M}(r_i)\log(r_i))$ operations in $\mathbb{K}$. By assumption (2) on M, this cost drops to $\mathcal{O}(d_i/r_i\mathsf{M}(d_i)+\mathsf{M}(d_i)\log(r_i))\subseteq\mathcal{O}(d_i\mathsf{M}(d_i))$. In order to deduce v_i, it then remains to multiply two polynomials in $\mathbb{E}_i[y]$ of degree bounded by d_i, which takes $\mathcal{O}(\mathsf{M}(r_i)\mathsf{M}(d_i))$ operations in $\mathbb{K}$. We deduce that the cost of step 3d belongs to $\mathcal{O}(\mathsf{M}(d_i)^2)$.

In step 3e, the direct use of [GG03, Theorem 15.18] would yield a cost in $\mathcal{O}(\mathsf{M}(d_i/r_i)\mathsf{M}(d_i)+\mathsf{M}(d_i)\log(d_i))$ in terms of operations in $\mathbb{E}_i$. But, since we skip step 4 of [GG03, Algorithm 15.17], we can discard the term $\mathsf{M}(d_i)\log(d_i)$. Therefore step 3e takes $\mathcal{O}(\mathsf{M}(r_i)\mathsf{M}(d_i/r_i)\mathsf{M}(d_i))\subseteq\mathcal{O}(\mathsf{M}(d_i)^2\mathsf{M}(d)/d)$ operations in $\mathbb{K}$, by assumption (2) on M.

We deduce that the total cost of step 3 belongs to

$$\mathcal{O}\left(\sum_{i=1}^{s} d_i/r_i \mathsf{M}(d_i)\log(d_i) + \sum_{i=1}^{s} \mathsf{M}(d_i)^2\mathsf{M}(d)/d\right) \subseteq \mathcal{O}(d\mathsf{M}(d)\log(d) + \mathsf{M}(d)^3/d),$$

by the super-additivity (3) of M. The total cost of the algorithm easily follows. □

In replacement of [GG03, Algorithm 15.17], a slightly faster multi-factor Hensel lifting can be found in [BLS03].

4.3. **Deterministic Decomposition.** For each $i \in \{1,\dots,r\}$, we introduce $g_i := G_i(0,y)$ and $\hat{f}_i := \hat{F}_i(0,y)$. Recall that $f_i := F_i(0,y)$ has already been defined. Substituting 0 for x in equation (11), we obtain that

$$g_i = \sum_{j=1}^{r} \rho_{j,i} \hat{f}_j f'_j.$$

Under Hypothesis (H), $(\hat{f}_i f'_i)_{i\in\{1,\dots,r\}}$ is a free family, so is $(g_i)_{i\in\{1,\dots,r\}}$. In order to complete the deterministic factorization algorithm we need a last device to compute a point $(c_1,\dots,c_r) \in \mathbb{K}^r$ that separates $\rho_1,\dots,\rho_r$. This is the aim of the following procedure.

Algorithm 6. *Separation of the residues.*

Input: $f \in \mathbb{K}[y]$ and $g_1,\dots,g_r$ as defined above.
Output: $(c_1,\dots,c_r) \in \mathbb{K}^r$ that separates $\rho_1,\dots,\rho_r$.

1. Let $c_1 := 1$ and $g := g_1$.
2. For i from 2 to r do
   a. Compute $Q(w,z) := \mathrm{Res}_y(f(y), zf'(y) - g(y) - wg_i(y))$.
   b. For each $j \in \{0,\dots,d(d-1)\}$ compute the squarefree part $q_j(z)$ of $Q(z,j)$. Take $c_i \in \{0,\dots,d(d-1)\}$ such that $q_{c_i}(z)$ has maximum degree.
   c. Let $g := g + c_i g_i$.
3. Return $(c_1,\dots,c_r)$.

Proposition 8. *Under Hypotheses (C) and (H), Algorithm 6 is correct and takes* $\mathcal{O}(rd^2\mathsf{M}(d)\log(d))$ *operations in* $\mathbb{K}$.

Proof. When entering the main loop in step 2, assume that $(c_1,\dots,c_{i-1})$ separates the elements of $\{(\rho_{1,1},\dots,\rho_{1,i-1}),\dots,(\rho_{r,1},\dots,\rho_{r,i-1})\}$, and that $g = c_1g_1 + \cdots + c_{i-1}g_{i-1}$. These assumptions clearly hold when $i = 2$. Let $q(w,z)$ denote the squarefree part of Q. From Proposition 13 of Appendix A, we have that

$$Q(w,z) = \delta \prod_{j=1}^{r} (z - c_1\rho_{j,1} - \cdots - c_{i-1}\rho_{j,i-1} - w\rho_{j,i})^{d_i},$$

hence $q(j,z)$ is squarefree if and only if $(c_1,\dots,c_{i-1},j)$ separates the elements of $\{(\rho_{1,1},\dots,\rho_{1,i}),\dots,(\rho_{r,1},\dots,\rho_{r,i})\}$.

The discriminant of q seen in $\mathbb{K}[w][z]$ has a degree at most $d(d-1)$ in w. If j annihilates this discriminant then $\deg(q_j) < \deg_z(q)$. Otherwise we have $\deg(q_j) = \deg_z(q)$, hence $q_j = q(j,z)$. Thanks to Hypothesis (C) at least one value for j does not annihilate this discriminant, hence $q(c_i,z)$ is squarefree. Of course $g = c_1g_1 + \cdots + c_ig_i$ at the end of step 2c. The correctness of the algorithm thus follows by induction.

Since Q has total degree d, it can be interpolated from $\mathcal{O}(d^2)$ points with $\mathcal{O}(d\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$. Therefore step 2a costs $\mathcal{O}(d^2\mathsf{M}(d)\log(d))$. By using multi-point evaluation, all the values $Q(z,0),\dots,Q(z,d(d-1))$ can be computed with $\mathcal{O}(d^2\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$. Thanks to Hypothesis (C) again, each squarefree part computation takes $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$ by means

of [GG03, Algorithm 14.19] (when replacing the characteristic zero hypothesis by Hypothesis (C), [GG03, Theorem 14.20] still holds). Finally the cost of step 2b amounts to $\mathcal{O}(d^2\mathsf{M}(d)\log(d))$ operations. □

The computation of the absolute factorization of F from a basis of $\pi(L_\infty)$ proceeds as follows.

Algorithm 7. *Deterministic computation of the absolute factorization of F from a basis of $\pi(L_\infty)$.*

Input: $F \in \mathbb{K}[x,y]$ such that Hypotheses (C) and (H) hold, and a basis $\nu_1, \ldots, \nu_r$ of $\pi(L_\infty)$.
Output: the absolute factorization of F.

1. Let $\mathfrak{f}(y) := y - \varphi$. Compute $\hat{\mathfrak{f}}(y)$ as the quotient of $f(y)$ by $\mathfrak{f}(y)$ in $\mathbb{A}[y]$.
2. For each $i \in \{1, \ldots, r\}$ compute

$$g_i = \sum_{j=1}^{d} \nu_{i,j}\, \mathrm{coeff}(\hat{\mathfrak{f}}\mathfrak{f}', \varphi^{j-1}).$$

3. Call Algorithm 6 with input f, $g_1, \ldots, g_r$ to compute a point $(c_1, \ldots, c_r)$ that separates $\rho_1, \ldots, \rho_r$.
4. Let $g := c_1 g_1 + \cdots + c_r g_r$ and compute the absolute partial fraction decomposition $(q_1, \mathsf{f}_1), \ldots, (q_s, \mathsf{f}_s)$ of g/f by means of the Lazard-Rioboo-Trager algorithm (Algorithm 12 in Appendix A).
5. Let $\tilde{P}_1, \ldots, \tilde{P}_s$, $\tilde{\mathsf{F}}_1, \ldots, \tilde{\mathsf{F}}_s$ be the output of Algorithm 5 called with input F, $(q_1, \mathsf{f}_1), \ldots, (q_s, \mathsf{f}_s)$ and $\eta := \max(\deg_y(\mathsf{f}_i) \mid i \in \{1, \ldots, s\}) + 1$.
6. Return $(q_1, \tilde{\mathsf{F}}_1), \ldots, (q_s, \tilde{\mathsf{F}}_s)$.

Proposition 9. *Algorithm 7 is correct and takes $\mathcal{O}(d\mathsf{M}(d)(rd\log(d) + \mathsf{M}(d)^2/d^2))$ operations in $\mathbb{K}$.*

Proof. The correctness of step 2 follows from substituting 0 for x in (10). The correctness of steps 3, 4, and 5 respectively follows from Propositions 8, 16 and 7. Finally, by Lemma 7 and since $(c_1, \ldots, c_r)$ separates $\rho_1, \ldots, \rho_r$, the output of the algorithm is actually correct: we recover the absolute partial fraction decomposition (12) with $\tilde{\mathsf{F}}_i = \mathsf{F}_i$ for all $i \in \{1, \ldots, s\}$.

Step 1 takes $\mathcal{O}(d\mathsf{M}(d))$ operations in $\mathbb{K}$. Step 2 costs $\mathcal{O}(rd^2)$. By Proposition 8, step 3 costs $\mathcal{O}(rd^2\mathsf{M}(d)\log(d))$. By Proposition 16, except for a finite number of values for d, step 4 costs $\mathcal{O}(d\mathsf{M}(d)\log(d))$. Lastly, by Proposition 7, step 5 costs $\mathcal{O}(\mathsf{M}(d)^2(\mathsf{M}(d)/d + \log(d)))$. Since M is assumed to be at most quadratic (from assumption (1)), the total cost of the algorithm drops to $\mathcal{O}(rd^2\mathsf{M}(d)\log(d) + \mathsf{M}(d)^3/d)$. □

4.4. **Probabilistic Decomposition.** In this subsection, we assume that we are given a free family of vectors $\nu_1, \ldots, \nu_{\tilde{r}}$ of $\mathbb{K}^d$ such that $\pi(L_\infty) \subseteq \langle \nu_1, \ldots, \nu_{\tilde{r}} \rangle$, hence $\tilde{r} \geq r$. When using the probabilistic algorithm of Section 3.3 to compute a basis of $\pi(L_\infty)$, the strict inequality $\tilde{r} > r$ may hold. In contrast to Algorithm 7, the lifted factorization may not equal the absolute factorization of F. A trial division easily raises the doubt.

The following Algorithm, is parametrized by a candidate $(c_1, \ldots, c_{\tilde{r}}) \in \mathbb{K}^{\tilde{r}}$ for the separation of the residues.

Algorithm 8. *Probabilistic computation of the absolute factorization of F from a basis of $\pi(L_\infty)$.*

Input: $F \in \mathbb{K}[x,y]$ such that Hypotheses (C) and (H) hold, a free family $\nu_1,\dots,\nu_{\tilde r}$ of vectors of $\mathbb{K}^d$ such that $\pi(L_\infty) \subseteq \langle \nu_1,\dots,\nu_{\tilde r}\rangle$, and $(c_1,\dots,c_{\tilde r}) \in \mathbb{K}^{\tilde r}$.

Output: the absolute factorization of F.

1. Let $\mathfrak{f}(y) := y - \varphi$. Compute $\hat{\mathfrak{f}}(y)$ as the quotient of $f(y)$ by $\mathfrak{f}(y)$ in $\mathbb{A}[y]$.
2. Compute
$$g := \sum_{i=1}^{d} \left(\sum_{j=1}^{\tilde r} c_j \nu_{j,i} \right) \operatorname{coeff}(\hat{\mathfrak{f}}\mathfrak{f}', \varphi^{i-1}).$$
3. Compute the absolute partial fraction decomposition $(q_1,\mathsf{f}_1),\dots,(q_s,\mathsf{f}_s)$ of g/f by means of the Lazard-Rioboo-Trager algorithm (Algorithm 12 in Appendix A). If $\sum_{i=1}^{s} \deg(q_i) \neq \tilde r$ then stop the execution. Let $\mathbb{E}_i := \mathbb{K}[z]/(q_i(z))$ and let α_i denote the residue class of z in $\mathbb{E}_i$.
4. For each $i \in \{1,\dots,s\}$, let $r_i := \deg(q_i)$ and $d_i := r_i \deg_y(\mathsf{f}_i)$. Let $\tilde P_1,\dots,\tilde P_r$, $\tilde{\mathsf{F}}_1,\dots,\tilde{\mathsf{F}}_s$ be the result of Algorithm 5 called with input F, $(q_1,\mathsf{f}_1),\dots,(q_s,\mathsf{f}_s)$ and $\eta := \max(d_i \mid i \in \{1,\dots,s\}) + 1$.
5. For each $i \in \{1,\dots,s\}$ do: if $\tilde P_i \notin \mathbb{K}[x,y]_{d_i}$ then stop the execution. If $\prod_{i=1}^{s} \tilde P_i \neq F$ then stop the execution.
6. For each $i \in \{1,\dots,s\}$ do
   - If $\tilde{\mathsf{F}}_i \notin \mathbb{K}[z][x,y]_{d_i/r_i}$ then stop the execution.
   - Let ψ_i denote the residue class of y in $\mathbb{E}_i[[x]]/(x^{d_i+1})[y]/(\tilde{\mathsf{F}}_i(x,y,\alpha_i))$ and call Algorithm 1 to compute $\tilde P_i(x,\psi_i)$. Stop the execution if $\tilde P_i(x,\psi_i)$ is nonzero.
7. Return $(q_1,\tilde{\mathsf{F}}_1),\dots,(q_s,\tilde{\mathsf{F}}_s)$.

Proposition 10. *Algorithm 8 either stops prematurely or returns a correct answer. In both cases it takes $\mathcal{O}(d^{(\omega+3)/2} + d^{3/2}\mathsf{M}(d)(\mathsf{M}(d)^2/d^2 + \log(d)))$ operations in $\mathbb{K}$. In addition, if $\tilde r = r$ then, for any valid input F and $\nu_1,\dots,\nu_{\tilde r}$, there exists a nonzero polynomial $S \in \mathbb{K}[C_1,\dots,C_{\tilde r}]$ of total degree at most $d(d-1)/2$ such that Algorithm 8 returns a correct answer when called with input F, $\nu_1,\dots,\nu_{\tilde r}$ and $(c_1,\dots,c_{\tilde r}) \in \mathcal{U}(S)$.*

Proof. In this paragraph we assume that $\tilde r = r$. We are exactly in the situation of Algorithm 7. Let S be the polynomial of Lemma 6, so that, if $(c_1,\dots,c_{\tilde r}) \in \mathcal{U}(S)$, then $(c_1,\dots,c_{\tilde r})$ actually separates the residues. By applying the same arguments as in the proof of Proposition 9, we deduce that $(q_1,\tilde{\mathsf{F}}_1),\dots,(q_s,\tilde{\mathsf{F}}_s)$ computed in step 4 actually represents the absolute factorization of F. Therefore we have $\tilde P_i(x,y) = \operatorname{Res}_z(q_i(z),\tilde{\mathsf{F}}_i(x,y,z))$, hence $\prod_{i=1}^{s} \tilde P_i = F$ holds in step 5. The computations done in step 6 correspond to testing if $\tilde{\mathsf{F}}_i(x,y,\alpha_i)$ divides $\tilde P_i(x,y)$ in $\mathbb{E}_i[[x]]/(x^{d_i+1})[y]$. In this case this division always holds, hence the algorithm returns a correct result.

We do not assume now that $\tilde r = r$. We wish to prove that the algorithm always returns a correct output whenever it finishes normally. When entering step 6, we are sure that $\tilde P_i$ divides F. If $\tilde P_i(x,\psi_i) = 0$ then $\tilde{\mathsf{F}}_i(x,y,\alpha_i)$ divides $\tilde P_i(x,y)$ in $\mathbb{E}_i[[x]]/(x^{d_i+1})[y]$. Since the remainder of $\tilde P_i(x,y)$ divided by $\tilde{\mathsf{F}}_i(x,y,\alpha_i)$ in $\mathbb{E}_i[x,y]$ has degree at most d_i in x, we deduce that $\tilde{\mathsf{F}}_i(x,y,\alpha_i)$ actually divides $\tilde P_i(x,y)$ in $\mathbb{E}_i[x,y]$. Therefore, if the algorithm reaches step 7 then we are sure that the following factorization holds:
$$F = \prod_{i=1}^{\tilde r} \prod_{q_i(\alpha)=0} \tilde{\mathsf{F}}_i(x,y,\alpha).$$

Since $\tilde{r} \geq r$, it follows that $\tilde{r} = r$ and that the output is correct.

The cost analysis of the first steps is straightforward: step 1 costs $\mathcal{O}(d\mathsf{M}(d))$, step 2 costs $\mathcal{O}(d^2)$. Step 3 costs $\mathcal{O}(d\mathsf{M}(d)\log(d))$ by Proposition 16. By Proposition 7, step 4 costs $\mathcal{O}(\mathsf{M}(d)^2(\mathsf{M}(d)/d + \log(d)))$.

Using the sub-product tree algorithm [GG03, Algorithm 10.3], the cost of step 5 amounts to $\mathcal{O}(\mathsf{M}(d)^2\log(s))$ operations by [GG03, Lemma 10.4].

By Corollary 2, for each $i \in \{1, \ldots, s\}$, the computation of $\tilde{P}_i(x, \psi_i)$ in step 6 costs $\mathcal{O}(d_i^{(\omega+3)/2} + d_i^{3/2}\mathsf{M}(d_i)(\mathsf{M}(r_i)\mathsf{M}(d_i/r_i)/d_i + \log(d_i)))$. From assumption (2), we deduce that $\mathsf{M}(r_i)\mathsf{M}(d_i/r_i)/d_i \leq \mathsf{M}(r_i)/r_i\mathsf{M}(d_i)/d_i \leq \mathsf{M}(d)^2/d^2$. The total cost of this step thus belongs to $\mathcal{O}(d^{(\omega+3)/2} + d^{3/2}\mathsf{M}(d)(\mathsf{M}(d)^2/d^2 + \log(d)))$.

From $\log(d) \in \mathcal{O}(\mathsf{M}(d)/\sqrt{d})$, we deduce that $\mathsf{M}(d)^2\log(d) \in \mathcal{O}(d^{3/2}\mathsf{M}(d)^3/d^2)$, which concludes the proof. □

Remark that the direct computation of the remainder of $\tilde{P}_i(x, y)$ divided by $\tilde{\mathsf{F}}_i(x, y, \alpha_i)$ in $\mathbb{E}_i[x][y]$ in step 6 takes $\mathcal{O}(\mathsf{M}(r_i)\mathsf{M}(d_i)^2)$ operations in $\mathbb{K}$, which is slightly slower than the method used in step 6.

5. Main Algorithms

We are now ready to present the main algorithms. In this section we do not assume that F satisfies Hypothesis (H). We only require F to be squarefree.

5.1. Deterministic Absolute Factorization Algorithm.

By putting together the deterministic sub-algorithms presented in the previous sections, we obtain the following top-level factorization procedure:

Algorithm 9. *Deterministic absolute factorization.*

Input: a squarefree polynomial $F \in \mathbb{K}[x, y]$ of total degree d satisfying Hypothesis (C).

Output: the absolute factorization of F.

1. Find $(u, v) \in \mathbb{K}^2$ such that the monic part $F_{u,v}$ in y of $F(x + uy + v, y)$ satisfies Hypothesis (H). Replace F by $F_{u,v}$.
2. Let $\sigma := 2d$ and compute ϕ to precision (x^σ) by means of Algorithm 2.
3. Call Algorithm 3 with input F and ϕ in order to obtain a basis $\nu_1, \ldots, \nu_r$ of $\pi(L_\infty)$.
4. Let $(q_1, \mathsf{F}_1), \ldots, (q_s, \mathsf{F}_s)$ be the absolute factorization of F returned by Algorithm 7 called with input F and $\nu_1, \ldots, \nu_r$.
5. Return $(q_1, \mathsf{F}_1(x - uy - v, y, z)), \ldots, (q_s, \mathsf{F}_s(x - uy - v, y, z))$.

Proposition 11. *Algorithm 9 is correct and takes* $\mathcal{O}(d^{\omega+1} + d\mathsf{M}(d)(rd\log(d) + \mathsf{M}(d))) \subseteq \mathcal{O}(d^3\mathsf{M}(d)\log(d))$ *operations in* $\mathbb{K}$.

Proof. The first step makes sense thanks to Proposition 1. When entering step 2, F satisfies Hypothesis (H). Therefore steps 2, 3 and 4 work correctly by Propositions 3, 5 and 9 respectively.

Let $r_i := \deg(q_i)$ and $d_i := r_i \deg_y(\mathsf{F}_i)$. In the last step, $\mathsf{F}_i(x - uy - v, y, z)$ can be computed in $\mathbb{K}[z]/(q_i(z))$, hence it takes $\mathcal{O}(d_i/r_i\mathsf{M}(d_i/r_i)\mathsf{M}(r_i))$ operations in $\mathbb{K}$ by Lemma 2. Therefore the cost of the last step belongs to

$$\mathcal{O}\left(\sum_{i=1}^{s} d_i/r_i\mathsf{M}(d_i/r_i)\mathsf{M}(r_i)\right) \subseteq \mathcal{O}\left(\sum_{i=1}^{s} d_i/r_i\mathsf{M}(d_i)\mathsf{M}(r_i)/r_i\right)$$

$$\subseteq \mathcal{O}\left(\sum_{i=1}^{s} d_i/r_i \mathsf{M}(d_i)^2/d_i\right) \quad \text{(by assumption (2) on } \mathsf{M})$$

$$\subseteq \mathcal{O}\left(\sum_{i=1}^{s} \mathsf{M}(d_i)^2\right) \subseteq \mathcal{O}(\mathsf{M}(d)^2) \quad \text{(by the super-additivity (3) of } \mathsf{M}).$$

By adding this cost to the ones of Propositions 1, 3, 5 and 9, we directly obtain the following total cost for the whole algorithm:

$$\mathcal{O}\left(d^{\omega+1} + d\mathsf{M}(d)^2 + d\mathsf{M}(d)(rd\log(d) + \mathsf{M}(d)^2/d^2)\right).$$

Since M is assumed to be at most quadratic from (1), we have $\mathsf{M}(d)^2/d^2 \in \mathcal{O}(\mathsf{M}(d))$, which concludes the proof. □

Theorem 1 straightforwardly follows from this proposition. As a consequence of the Lazard-Rioboo-Trager algorithm, it is worth noting that the degrees in y of $\mathsf{F}_1, \ldots, \mathsf{F}_s$ are pairwise distinct.

In order to only test the absolute irreducibility of F, only steps 1 to 3 are necessary. We can complete:

Proof of Theorem 2. By Propositions 1, 3, 5, the total cost of steps 1 to 3 in Algorithm 9 amounts to $\mathcal{O}(d^{\omega+1} + d\mathsf{M}(d)(\mathsf{M}(d) + d\log(d)))$ operations in $\mathbb{K}$. □

5.2. **Probabilistic Absolute Factorization Algorithm.** The probabilistic factorization algorithm is very similar to the deterministic one.

Algorithm 10. *Probabilistic absolute factorization.*

Input: a squarefree polynomial $F \in \mathbb{K}[x, y]$ of total degree d satisfying Hypothesis (C), and $(u, v, a, c_1, \ldots, c_d) \in \mathbb{K}^{d+3}$.

Output: the absolute factorization of F.

1. Replace F by the monic part in y of $F(x + uy + v, y)$. If this new F does not satisfy Hypothesis (H) then stop the execution.
2. Let $\tau := 2d + 1$ and compute ϕ to precision (x^τ) by means of Algorithm 2.
3. Call Algorithm 4 with input F, ϕ and $(a, 0)$ in order to obtain a basis $\nu_1, \ldots, \nu_{\tilde{r}}$ of $\{(\ell_1, \ldots, \ell_d) \in \mathbb{K}^d \mid P_\tau^{a,0}\}$.
4. Let $(q_1, \mathsf{F}_1), \ldots, (q_s, \mathsf{F}_s)$ be the factorization of F returned by Algorithm 8 called with input F, $\nu_1, \ldots, \nu_{\tilde{r}}$ and $(c_1, \ldots, c_{\tilde{r}})$.
5. Return $(q_1, \mathsf{F}_1(x - uy - v, y, z)), \ldots, (q_s, \mathsf{F}_s(x - uy - v, y, z))$.

Proposition 12. *Algorithm 10 either stops prematurely or returns a correct answer. In both cases its takes $\mathcal{O}(d^{(\omega+3)/2} + d^{3/2}\mathsf{M}(d)(\mathsf{M}(d)^2/d^2 + \log(d)))$ operations in $\mathbb{K}$. In addition, for any input polynomial F, there exists a nonzero polynomial $P \in \mathbb{K}[U]$ of degree at most d such that, for any $u \in \mathcal{U}(P)$, there exists a nonzero polynomial $Q_u \in \mathbb{K}[V]$ of degree at most $d(d-1)$ such that, for any $v \in \mathcal{U}(Q_u)$, there exists a nonzero polynomial $R_{u,v} \in \mathbb{K}[A]$ of degree at most $d(d-1)$ such that, for any $a \in \mathcal{U}(R_{u,v})$, there exists a nonzero polynomial $S_{u,v,a} \in \bar{\mathbb{K}}[C_1, \ldots, C_d]$ of total degree at most $d(d-1)/2$ such that, for any $(c_1, \ldots, c_d) \in \mathcal{U}(S_{u,v,a})$, Algorithm 10 called with F and $(u, v, a, c_1, \ldots, c_d)$ returns a correct answer.*

Proof. By Lemma 1, there exists a nonzero polynomial $P \in \mathbb{K}[U]$ of degree at most d such that, for any $u \in \mathcal{U}(P)$, there exists a nonzero polynomial $Q_u \in \mathbb{K}[V]$ of degree at most $d(d-1)$ such that, for any $v \in \mathcal{U}(Q_u)$, the monic part of $F(x + uy + v, y)$ satisfies Hypothesis (H). The change of the variables in the first step costs $\mathcal{O}(d\mathsf{M}(d))$ by Lemma 2. The test of Hypothesis (H) costs $\mathcal{O}(\mathsf{M}(d)\log(d))$, hence the cost of the first step belongs to $\mathcal{O}(d\mathsf{M}(d))$.

When entering step 2, F satisfies Hypothesis (H). Therefore the correctness of steps 2 and 3 follows from Propositions 3, 6 respectively. By Lemma 5 applied

d	$r=1$	$r=2$	$r=2^{\lfloor \log_2(d)/2 \rfloor}$	$r=d/2$	$r=d$
8	0.08 s	0.03 s	0.03 s	0.03 s	0.02 s
16	0.41 s	0.20 s	0.18 s	0.17 s	0.12 s
32	2.36 s	1.55 s	2.96 s	1.42 s	0.78 s
64	18.8 s	21.0 s	21.8 s	20.5 s	15.8 s
128	147 s	175 s	170 s	179 s	119 s
256	1239 s	1423 s	1419 s	1520 s	973 s

TABLE 1. Probabilistic absolute factorization algorithm

d	Gao/Gaussian	Gao/black box	Lecerf/Trager
8	0.03 s	0.09 s	0.44 s
16	0.35 s	2.38 s	1.43 s
32	8.43 s	45.2 s	12.3 s
64	452 s	512 s	274 s
128	>512 MB	11359 s	2323 s

TABLE 2. Comparisons with other algorithms

with $b=0$, there exists a nonzero polynomial $R_{u,v}$ such that, for any $a \in \mathcal{U}(R_{u,v})$, we have $\pi(L_{2d}) = \langle \nu_1, \ldots, \nu_{\tilde{r}} \rangle$. Therefore $\pi(L_\infty) = \langle \nu_1, \ldots, \nu_{\tilde{r}} \rangle$ by Theorem 4. By Proposition 10, there exists a nonzero polynomial $S_{u,v,a} \in \bar{\mathbb{K}}[C_1, \ldots, C_d]$ such that, for any $(c_1, \ldots, c_d) \in \mathcal{U}(S_{u,v,a})$, step 4 returns a correct answer. On the other hand, by Proposition 10 again we know that step 4 either returns a correct answer or stops prematurely.

We have seen in the proof of the preceding proposition that step 5 takes $\mathcal{O}(\mathsf{M}(d)^2)$ operations in $\mathbb{K}$. The total cost of the algorithm is directly obtained by adding this cost with the ones given in Propositions 3, 6 and 10. □

Theorem 3 straightforwardly follows from this proposition. For the only test of the absolute irreducibility, Algorithm 10 can be simplified. Unfortunately, this does not yield a smaller cost bound. In fact, when using fast polynomial multiplication, that is $\mathsf{M}(d) \in \tilde{\mathcal{O}}(d)$, the cost of Algorithm 10 drops to $\mathcal{O}(d^{(\omega+3)/2})$ (recall that $\omega > 2$). Therefore the computation of ϕ is part of the bottleneck.

6. EXPERIMENTS

In this section we provide timings obtained with our MAGMA [Mag] implementation of the probabilistic absolute factorization algorithm (Algorithm 10). Our package is freely available at `http://www.math.uvsq.fr/~lecerf`. Because no absolute partial fraction decomposition algorithm is implemented in MAGMA, we decided to use the Rothstein-Trager algorithm, which is much easier to implement.

In our experiments we take $\mathbb{K} := \mathbb{Z}/754974721\mathbb{Z}$. In this situation we observe that the rational factorization is faster than the absolute factorization, which agrees with the theoretical cost estimates. For this reason we illustrate the behavior of our program with random irreducible polynomials in $\mathbb{K}[x,y]$ of total degree d with r absolutely irreducible factors (although our program does not require the input polynomial to be irreducible). Timings are measured by means of the command `Cputime()` with a 1.8 GHz Pentium M processor and MAGMA V2.11-14.

In Table 1 we display the running time of our program for various values for d and r. All the computations in this table took 141 MB of memory. One can observe

that the running time does not depend much on r when d is fixed. In fact, most of the time is spent in the computation of ϕ to precision $2d+1$ (which does not depend on r). On the other hand, we can clearly observe that the running time of our implementation is sub-quadratic in the dense size of the polynomial to be factored.

In Table 2 we provide timings with the same family of examples. We arbitrarily take $r = 2^{\lfloor \log_2(d)/2 \rfloor}$. In the column "Gao/Gaussian" we indicate the time needed to solve system (6), with G and H in $\mathbb{K}[x,y]_{d-1}$. Here we use the function `Nullspace` for sparse matrices. This function implements the Gaussian elimination. Since the linear system has $d(d+1)$ unknowns, the running time is cubic in the dense size of F. When $d = 64$, this method took 327 MB of memory so that we were not able to run the test for $d = 128$ with the 512 MB of memory of our computer.

In [Gao03] Gao suggested that system (6) could be solved faster by means of the black box approach *à la Wiedemann*. In order to compute a single random solution of the system this approach performs $d(d+1)$ matrix-vector products. Each product amounts to compute 3 multiplications of bivariate polynomials of total degree at most d. In the column "Gao/black box" of Table 2 we indicate the time needed to compute all the $3d(d+1)$ polynomial multiplications. This way one random solution of (6) can be computed with $\tilde{\mathcal{O}}(d^4)$ operations in $\mathbb{K}$ when using fast polynomial multiplication.

In the subsection "Related Work" of the Introduction, we explained how Trager's reduction to factoring over algebraic extensions could be combined to the rational factorization algorithm of [Lec04]. We first compute an irreducible factor $e(y)$ of $F(0,y)$. Let $\mathbb{E} := \mathbb{K}[y]/(e(y))$ and let β denote the residue class of y in $\mathbb{E}$. We factor $F(0,y)$ in $\mathbb{E}[y]$, and lift the resulting irreducible factors in $\mathbb{E}[[x]][y]$ to the precision (x^{2d+1}). Then we apply the recombination algorithm of [Lec04]. In the column "Lecerf/Trager" of Table 2 we give the total running time for: the factorization of $F(0,y)$ in $\mathbb{K}[y]$ and in $\mathbb{E}[y]$, and the lifting in $\mathbb{E}[[x]][y]$ to precision (x^{2d+1}).

Table 2 shows that the theoretical cost estimates can roughly be observed in high degrees. We can also observe that the black box approach does not gain versus the Gaussian elimination up to degree 64. Although the running times of the two versions of Gao's algorithm and of the "Lecerf/Trager" strategy only represent rough lower bounds for complete implementations, we can observe that our algorithm gains even in small degrees by comparing the two tables. The good theoretical behavior of our algorithm is well reflected in practice.

Conclusion

We have presented new faster algorithms for computing the absolute factorization of a bivariate polynomial. Experiments show that these algorithms are of practical interest for dense polynomials over large finite fields. In near future, we shall design a faster version of our program for the special case when $\mathbb{K} = \mathbb{Q}$: an important intermediate growth of the integers occurs during the computation of ϕ. This phenomenon can be observed in Example 3 given in the introduction. We also plan to extend our methods to small characteristics and to improve them for sparse polynomials.

Appendix A. Univariate Absolute Partial Fraction Decomposition

Throughout this appendix, $\mathbb{K}$ is a field and f denotes a polynomial in $\mathbb{K}[y]$, which satisfies

$$\text{Hypothesis (h)} \quad \begin{cases} (i)\ f \text{ is monic of degree } d \geq 1, \\ (ii)\ \delta := \operatorname{Res}(f, f') \neq 0. \end{cases}$$

Let g be a polynomial in $\mathbb{K}[y]$ of total degree at most $d-1$. Under Hypothesis (h), there exist unique pairwise distinct elements $\rho_1, \ldots, \rho_r$ in $\bar{\mathbb{K}}$ and unique monic polynomials $f_1, \ldots, f_r$ in $\bar{\mathbb{K}}[y]$ such that $f_1 \cdots f_r = f$ and

$$\frac{g}{f} = \sum_{i=1}^{r} \rho_i \frac{f_i'}{f_i}. \tag{14}$$

The right-hand side of this equality is called the *absolute partial fraction decomposition* of g/f. The set of factors $\{f_1, \ldots, f_r\}$ can be represented by a set of pairs of polynomials $\{(q_1, \mathsf{f}_1), \ldots, (q_s, \mathsf{f}_s)\}$ that satisfies the following properties:

- For all $i \in \{1, \ldots, s\}$, the polynomial q_i belongs to $\mathbb{K}[z]$, is monic, squarefree and $\deg(q_i) \geq 1$.
- For all $i \in \{1, \ldots, s\}$, the polynomial f_i belongs to $\mathbb{K}[y, z]$, is monic in y, and $\deg_z(\mathsf{f}_i) \leq \deg(q_i) - 1$.
- $\deg(q_1) + \cdots + \deg(q_s) = r$, the set of roots of $q_1 \cdots q_s$ is $\{\rho_1, \ldots, \rho_r\}$, and $\{f_1, \ldots, f_r\} = \bigcup_{i=1}^{s} \{\mathsf{f}_i(y, \alpha) \mid q_i(\alpha) = 0\}$.

Such a representation is not redundant: to each f_j there corresponds a unique pair (i, α) such that $f_j(y) = \mathsf{f}_i(y, \alpha)$ and $q_i(\alpha) = 0$. Decomposition (14) rewrites to:

$$\frac{g}{f} = \sum_{i=1}^{s} \sum_{q_i(\alpha)=0} \alpha \frac{\frac{\partial \mathsf{f}_i}{\partial y}(y, \alpha)}{\mathsf{f}_i(y, \alpha)}. \tag{15}$$

In this appendix we briefly recall the classical algorithms for computing the decomposition of g/f in form (15). These algorithms were originally designed to compute symbolic integrals of rational functions in characteristic zero. The aim of this appendix is to verify that they still apply in positive characteristic under Hypothesis (h). We closely follow the presentation made in [GG03, Chapter 22]. The reader may also read [Bro05, Chapter 2].

For all $i \in \{1, \ldots, r\}$, we introduce $d_i := \deg(f_i)$ and $\hat{f}_i := f/f_i$. We also define:

$$Q(z) := \prod_{i=1}^{r} (z - \rho_i)^{d_i} \quad \text{and} \quad q(z) := \prod_{i=1}^{r} (z - \rho_i).$$

A.1. **The Rothstein-Trager Algorithm.** The following proposition is adapted from [GG03, Theorem 22.8]. The original idea is due to Rothstein [Rot76, Rot77] and Trager [Tra76] independently.

Proposition 13. *Under Hypothesis (h), the polynomials Q and q belong to $\mathbb{K}[z]$. We have $\delta Q(z) = \mathrm{Res}_y(f(y), zf'(y) - g(y))$, and f_i is proportional to $\gcd(f, \rho_i f' - g)$ for all $i \in \{1, \ldots, r\}$.*

Proof. Multiplying both sides of (14) by f, we obtain

$$g = \sum_{i=1}^{r} \rho_i f_i' \hat{f}_i,$$

and then

$$\mathrm{rem}(g, f_i) = \mathrm{rem}(\rho_i f_i' \hat{f}_i, f_i) = \mathrm{rem}(\rho_i f', f_i).$$

By the multiplicativity of the resultant, we deduce:

$$\begin{aligned}\mathrm{Res}_y(f(y), zf'(y) - g(y)) &= \prod_{i=1}^{r} \mathrm{Res}_y(f_i(y), zf'(y) - g(y)) \\ &= \prod_{i=1}^{r} \mathrm{Res}_y(f_i(y), (z-\rho_i)f'(y)) \\ &= \prod_{i=1}^{r} \mathrm{Res}(f_i, f') \prod_{i=1}^{r} \mathrm{Res}_y(f_i(y), z-\rho_i) \\ &= \mathrm{Res}(f, f') \prod_{i=1}^{r} (z-\rho_i)^{d_i}.\end{aligned}$$

It follows that $Q \in \mathbb{K}[z]$. Thanks to Hypothesis (h), f is separable, so is its splitting field $\mathbb{E}$. Since $\rho_i = g(\beta)/f_i'(\beta)$ for any root β of f_i, the residues $\rho_1, \ldots, \rho_r$ belong to $\mathbb{E}$. Therefore the minimal polynomial of ρ_i over $\mathbb{K}$ is separable. Since it divides Q, it is an irreducible factor of q. All the irreducible factors of q can be obtained this way, whence $q \in \mathbb{K}[z]$.

For any i and j in $\{1, \ldots, r\}$, by taking both sides of the equality

$$\rho_i f' - g = \sum_{j=1}^{r} (\rho_i - \rho_j) f_j' \hat{f}_j$$

modulo f_j, we obtain:

$$\mathrm{rem}(\rho_i f' - g, f_j) = (\rho_i - \rho_j)\,\mathrm{rem}(f_j' \hat{f}_j, f_j).$$

Thanks to Hypothesis (h) again, the polynomial $f_j' \hat{f}_j$ is invertible modulo f_j. We finally deduce that f_j divides $\rho_i f' - g$ if and only if $\rho_i = \rho_j$. □

Lemma 8. *Under Hypothesis (h), if $\mathbb{K}$ has cardinality at least $d+1$ then Q can be computed from f and g with $\mathcal{O}(d\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$.*

Proof. For each $i \in \{0, \ldots, d\}$, the value $Q(i)$ can be computed with $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations. Since Q has degree d, it can be interpolated with $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations. □

Let us mention that the methods of [BFSS02] would yield a slightly better cost for the computation of Q. Let $q_1, \ldots, q_s$ denote the monic irreducible factors of q. For each $i \in \{1, \ldots, s\}$, let $r_i := \deg(q_i)$, $\mathbb{E}_i := \mathbb{K}[z]/(q_i(z))$, and let α_i denote the residue class of z in $\mathbb{E}_i$. There exists a unique polynomial $\mathsf{f}_i(y, z) \in \mathbb{K}[y, z]$ that satisfies the following properties:

- f_i is monic in y and $\deg_z(\mathsf{f}_i) \leq r_i - 1$;
- $\mathsf{f}_i(y, \alpha_i)$ is proportional to $\gcd(f, \alpha_i f' - g)$.

By Proposition 13, we have:

$$\{f_1, \ldots, f_s\} = \bigcup_{i=1}^{s} \{\mathsf{f}_i(y, \alpha) \mid q_i(\alpha) = 0\}.$$

Therefore $(q_1, \mathsf{f}_1), \ldots, (q_s, \mathsf{f}_s)$ represents the absolute partial fraction decomposition of g/f. These formulas lead to the following algorithm:

Algorithm 11. *The Rothstein-Trager algorithm.*

Input: $f \in \mathbb{K}[y]$ satisfying Hypothesis (h), and $g \in \mathbb{K}[y]$ with $\deg(g) \leq d-1$.
Output: the absolute partial fraction decomposition of g/f.

1. Compute $Q(z) := \mathrm{Res}_y(f(y), zf'(y) - g(y))$.
2. Compute the irreducible factors $q_1, \ldots, q_s$ of Q.
3. For each $i \in \{1, \ldots, s\}$, compute $\mathsf{f}_i(y, z)$ as the canonical preimage of the monic part of $\gcd(f, \alpha_i f' - g)$ in $\mathbb{E}_i[y]$.
4. Return $(q_1, \mathsf{f}_1), \ldots, (q_s, \mathsf{f}_s)$.

Proposition 14. *Algorithm 11 is correct. If the $\mathbb{K}$ has cardinality at least $d+1$ then it performs one irreducible factorization of a univariate polynomial of degree d in $\mathbb{K}[y]$ plus $\mathcal{O}(\mathsf{M}(d)(\mathsf{M}(r)\log(d)^2 + d\log(d)))$ operations in $\mathbb{K}$.*

Proof. By Lemma 8, step 1 costs $\mathcal{O}(d\mathsf{M}(d)\log(d))$. In step 3, the computation of each f_i takes $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations in $\mathbb{E}_i$, hence $\mathcal{O}(\mathsf{M}(r_i)\log(r_i)\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$. We use the super-additivity (3) of M to conclude the proof. □

A.2. **The Lazard-Rioboo-Trager Algorithm.** In [LR90] Lazard and Rioboo modified Algorithm 11 in order to avoid the irreducible factorization of Q in step 2. They showed that the squarefree factorization of Q suffices to deduce the decomposition of g/f. The same idea was independently found and implemented by Trager in his SCRATCHPAD II package, but never published.

We define the monic polynomial remainder sequence $p_1(y,z), \ldots p_m(y,z)$ of $f(y)$ and $zf'(y) - g(y)$ in $\mathbb{K}(z)[y]$ recursively as follows:

$$p_0 := f(y)/\operatorname{coeff}(f, y^d) = 1, \; p_1 := (zf'(y) - g(y))/\operatorname{coeff}(zf'(y) - g(y), y^{d-1}),$$

$$p_{i+1} := \operatorname{rem}(p_{i-1}, p_i)/c_{i+1} \text{ for } i \geq 1,$$

where c_{i+1} denotes the leading coefficient in y of $\operatorname{rem}(p_{i-1}, p_i)$, for $i \geq 1$. The integer m is defined as the first integer such that $p_{m+1} = 0$. The following result is adapted from [GG03, Theorem 22.9].

Proposition 15. *Assume that Hypothesis (h) holds and let α be a root of Q of multiplicity e. Then there exists a unique remainder p_i of degree e in y. In addition, the polynomial $p_i(y, \alpha)$ is well-defined and is proportional to $\gcd(f, \alpha f' - g)$.*

Proof. From Proposition 13, we already know that $\gcd(f, \alpha f' - g)$ has degree e. The rest of the proof follows from [GG03, Theorem 6.55], exactly as in the proof of [GG03, Theorem 22.9]. □

From now on $q_1, \ldots, q_s$ represent the squarefree factors of Q, so that we have:

$$Q := \prod_{i=1}^{s} q_i^{e_i},$$

with $r_i := \deg(q_i) \geq 1$ and $1 \leq e_1 < e_2 < \cdots < e_s \leq d$. By the previous proposition, for each $i \in \{1, \ldots, s\}$, there exists a unique remainder $w_i(y,z) \in \mathbb{K}(z)[y]$ of degree e_i. In addition, $w_i(y, \alpha)$ is well-defined for any root α of q_i. Let α_i now denote the residue class of z in $\mathbb{E}_i := \mathbb{K}[z]/(q_i(z))$. We can define the polynomial $\mathsf{f}_i(y,z)$ as the canonical preimage of $w_i(y, \alpha_i)$ in $\mathbb{K}[y,z]$. Finally, $(q_1, \mathsf{f}_1), \ldots, (q_s, \mathsf{f}_s)$ represents the absolute partial fraction decomposition of g/f. These formulas lead to the following algorithm:

Algorithm 12. *The Lazard-Rioboo-Trager algorithm.*

Input: $f \in \mathbb{K}[y]$ satisfying Hypothesis (h), and $g \in \mathbb{K}[y]$ with $\deg(g) \leq d-1$.
Output: the absolute partial fraction decomposition of g/f.

1. Compute $Q(z) := \mathrm{Res}_y(f(y), zf'(y) - g(y))$.
2. Compute the squarefree decomposition $q_1^{e_1} \cdots q_s^{e_s}$ of Q.
3. Compute the remainders $w_1, \ldots, w_s$ of respective degrees $e_1, \ldots, e_s$ in the monic polynomial remainder sequence of $f(y)$ and $zf'(y) - g(y)$ in $\mathbb{K}(z)[y]$.
4. For each $i \in \{1, \ldots, s\}$, construct $\mathsf{f}_i(y, z)$ as the canonical preimage of $w_i(y, \alpha_i)$.
5. Return $(q_1, \mathsf{f}_1), \ldots, (q_s, \mathsf{f}_s)$.

Proposition 16. *Algorithm 12 is correct. If $\mathbb{K}$ has characteristic 0 or at least $d+1$, and if $\mathbb{K}$ has cardinality at least $6d+3$, then Algorithm 12 takes $\mathcal{O}(d\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$.*

Proof. By Lemma 8, step 1 takes $\mathcal{O}(d\mathsf{M}(d)\log(d))$ operations. Thanks to the hypothesis on the characteristic of $\mathbb{K}$, step 2 can be done with $\mathcal{O}(\mathsf{M}(d)\log(d))$ operations by means of Yun's algorithm [GG03, Algorithm 14.21]: we can apply [GG03, Theorem 14.23] *mutatis mutandis*. By [GG03, Part (ii) of Exercise 11.9] and thanks to the hypothesis on the cardinality of $\mathbb{K}$, step 3 can be done with $\mathcal{O}(d\mathsf{M}(d)\log(d))$ operations in $\mathbb{K}$. By [GG03, Theorem 6.54], each coefficient of w_i have numerator and denominator of degree at most $2d$ in z. Therefore the total cost of step 4 amounts to

$$\mathcal{O}\left(\sum_{i=1}^{s}(e_i\mathsf{M}(d) + e_i\mathsf{M}(r_i)\log(r_i))\right)$$

operations in $\mathbb{K}$. From assumption (2), we deduce that this cost belongs to

$$\mathcal{O}\left(d\mathsf{M}(d) + \sum_{i=1}^{s}\mathsf{M}(e_i r_i)\log(d)\right) \subseteq \mathcal{O}\left(d\mathsf{M}(d) + \mathsf{M}(d)\log(d)\right),$$

which concludes the proof. □

It is worth noting that the above hypothesis on the cardinality of $\mathbb{K}$ could be slightly refined, but this would yield us too far from our present concerns. Lastly, we refer the reader to [Mul97] for implementation details, and to [GL03] for a recent comprehensive survey on polynomial remainder sequences.

References

[AGL04] F. Abu Salem, S. Gao, and A. G. B. Lauder. Factoring polynomials via polytopes. In *Proceedings of ISSAC 2004*, pages 4–11. ACM Press, 2004.

[BCGW93] C. Bajaj, J. Canny, T. Garrity, and J. Warren. Factoring rational polynomials over the complex numbers. *SIAM J. Comput.*, 22(2):318–331, 1993.

[BCS97] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic complexity theory.* Springer-Verlag, 1997.

[BFSS02] A. Bostan, Ph. Flajolet, B. Salvy, and É. Schost. Fast computation with two algebraic numbers. Research Report 4579, Institut National de Recherche en Informatique et en Automatique, October 2002.

[BHKS04] K. Belabas, M. van Hoeij, J. Klüners, and A. Steel. Factoring polynomials over global fields. Manuscript, October 2004.

[BLS03] A. Bostan, G. Lecerf, and É. Schost. Tellegen's principle into practice. In *Proceedings of ISSAC 2003*, pages 37–44. ACM Press, 2003.

[BLS+04] A. Bostan, G. Lecerf, B. Salvy, É. Schost, and B. Wiebelt. Complexity issues in bivariate polynomial factorization. In *Proceedings of ISSAC 2004*, pages 42–49. ACM Press, 2004.

[BP94] D. Bini and V. Y. Pan. *Polynomial and matrix computations. Vol. 1. Fundamental algorithms.* Progress in Theoretical Computer Science. Birkhäuser, 1994.

[Bro01] M. Bronstein. Computer algebra algorithms for linear ordinary differential and difference equations. In *European Congress of Mathematics, Vol. II (Barcelona, 2000)*, volume 202 of *Progr. Math.*, pages 105–119. Birkhäuser, 2001.

[Bro05] M. Bronstein. *Symbolic integration. I Transcendental functions*. Springer-Verlag, second edition, 2005.

[BS04] A. Bostan and É. Schost. Polynomial evaluation and interpolation on special sets of points. *J. Complexity (to appear)*, 2004.

[BT03] M. Bronstein and B. M. Trager. A reduction for regular differential systems. Manuscript, 2003.

[CG04] G. Chèze and A. Galligo. From an approximate to an exact factorization. Manuscript, 2004.

[CG05] G. Chèze and A. Galligo. Four lessons on polynomial absolute factorization. In A. Dickenstein and I. Z. Emiris, editors, *Solving Polynomial Equations: Foundations, Algorithms, and Applications*, volume 14 of *Algorithms and Computation in Mathematics*. Springer-Verlag, due to appear in April 2005.

[CGKW02] R. M. Corless, A. Galligo, I. S. Kotsireas, and S. M. Watt. A geometric-numeric algorithm for absolute factorization of multivariate polynomials. In *Proceedings of ISSAC 2002*, pages 37–45. ACM Press, 2002.

[Chè04a] G. Chèze. Absolute polynomial factorization in two variables and the knapsack problem. In *Proceedings of ISSAC 2004*, pages 87–94. ACM Press, 2004.

[Chè04b] G. Chèze. *Des méthodes symboliques-numériques et exactes pour la factorisation absolue des polynômes en deux variables*. PhD thesis, Université de Nice-Sophia Antipolis (France), 2004.

[CSTU02] O. Cormier, M. F. Singer, B. M. Trager, and F. Ulmer. Linear differential operators for polynomial equations. *J. Symbolic Comput.*, 34(5):355–398, 2002.

[Del01] S. Dellière. On the links between triangular sets and dynamic constructible closure. *J. Pure Appl. Algebra*, 163(1):49–68, 2001.

[DT89] R. Dvornicich and C. Traverso. Newton symmetric functions and the arithmetic of algebraically closed fields. In *Applied algebra, algebraic algorithms and error-correcting codes (Menorca, 1987)*, volume 356 of *Lecture Notes in Comput. Sci.*, pages 216–224. Springer-Verlag, 1989.

[Duv91] D. Duval. Absolute factorization of polynomials: a geometric approach. *SIAM J. Comput.*, 20(1):1–21, 1991.

[Duv95] D. Duval. Évaluation dynamique et clôture algébrique en Axiom. *J. Pure Appl. Algebra*, 99:267–295, 1995.

[Gao01] S. Gao. Absolute irreducibility of polynomials via Newton polytopes. *J. Algebra*, 237(2):501–520, 2001.

[Gao03] S. Gao. Factoring multivariate polynomials via partial differential equations. *Math. Comp.*, 72(242):801–822, 2003.

[GC84] D. Yu. Grigoriev and A. L. Chistov. Fast factorization of polynomials into irreducible ones and the solution of systems of algebraic equations. *Dokl. Akad. Nauk SSSR*, 275(6):1302–1306, 1984.

[GG94] S. Gao and J. von zur Gathen. Berlekamp's and Niederreiter's polynomial factorization algorithms. In *Finite fields: theory, applications, and algorithms (Las Vegas, NV, 1993)*, volume 168 of *Contemp. Math.*, pages 101–116. Amer. Math. Soc., 1994.

[GG03] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, second edition, 2003.

[GKL04] S. Gao, E. Kaltofen, and A. Lauder. Deterministic distinct degree factorization for polynomials over finite fields. *J. Symbolic Comput.*, 38(6):1461–1470, 2004.

[GKM+04] S. Gao, E. Kaltofen, J. May, Z. Yang, and L. Zhi. Approximate factorization of multivariate polynomials via differential equations. In *Proceedings of ISSAC 2004*, pages 167–174. ACM Press, 2004.

[GL01] S. Gao and A. G. B. Lauder. Decomposition of polytopes and polynomials. *Discrete Comput. Geom.*, 26(1):89–104, 2001.

[GL03] J. von zur Gathen and T. Lücking. Subresultants revisited. *Theor. Comput. Sci.*, 297(1-3):199–239, 2003.

[GL04] S. Gao and A. G. B. Lauder. Fast absolute irreducibility testing via Newton polytopes. Manuscript, 2004.

[GR02] A. Galligo and D. Rupprecht. Irreducible decomposition of curves. *J. Symbolic Comput.*, 33(5):661–677, 2002.

[GR03] S. Gao and V. M. Rodrigues. Irreducibility of polynomials modulo p via Newton polytopes. *J. Number Theory*, 101(1):32–47, 2003.

[Hoe02] M. van Hoeij. Factoring polynomials and the knapsack problem. *J. Number Theory*, 95(2):167–189, 2002.

[HRUW99] M. van Hoeij, J.-F. Ragot, F. Ulmer, and J.-A. Weil. Liouvillian solutions of linear differential equations of order three and higher. *J. Symbolic Comput.*, 28(4-5):589–609, 1999.

[HS81] J. Heintz and M. Sieveking. Absolute primality of polynomials is decidable in random polynomial time in the number of variables. In *Automata, languages and programming (Akko, 1981)*, volume 115 of *Lecture Notes in Comput. Sci.*, pages 16–28. Springer-Verlag, 1981.

[Kal85] E. Kaltofen. Fast parallel absolute irreducibility testing. *J. Symbolic Comput.*, 1(1):57–67, 1985.

[Kal90] E. Kaltofen. Polynomial factorization 1982–1986. In *Computers in mathematics (Stanford, CA, 1986)*, volume 125 of *Lecture Notes in Pure and Appl. Math.*, pages 285–309. Dekker, 1990.

[Kal92] E. Kaltofen. Polynomial factorization 1987–1991. In *LATIN '92 (São Paulo, 1992)*, volume 583 of *Lecture Notes in Comput. Sci.*, pages 294–313. Springer-Verlag, 1992.

[Kal95] E. Kaltofen. Effective Noether irreducibility forms and applications. *J. Comput. System Sci.*, 50(2):274–295, 1995.

[Kal03] E. Kaltofen. Polynomial factorization: a success story. In *Proceedings of ISSAC 2003*, pages 3–4. ACM Press, 2003.

[Lec04] G. Lecerf. Sharp precision in Hensel lifting for bivariate polynomial factorization. *Math. Comp. (to appear)*, 2004.

[Lec05] G. Lecerf. Improved dense multivariate polynomial factorization algorithms. Manuscript, 2005.

[LR90] D. Lazard and R. Rioboo. Integration of rational functions: rational computation of the logarithmic part. *J. Symbolic Comput.*, 9(2):113–115, 1990.

[Mag] The Magma computational algebra system for algebra, number theory and geometry. http://magma.maths.usyd.edu.au/magma/. Computational Algebra Group, School of Mathematics and Statistics, The University of Sydney, NSW 2006 Australia.

[MŞ99] M. Mignotte and D. Ştefănescu. *Polynomials. An algorithmic approach*. Springer-Verlag, 1999.

[Mul97] T. Mulders. A note on subresultants and the Lazard/Rioboo/Trager formula in rational function integration. *J. Symbolic Comput.*, 24(1):45–50, 1997.

[Nie93] H. Niederreiter. A new efficient factorization algorithm for polynomials over small finite fields. *Appl. Algebra Engrg. Comm. Comput.*, 4(2):81–87, 1993.

[Poh04] M. E. Pohst. Factoring polynomials over global fields I. *J. Symbolic Comput.*, 2004. In Press, Corrected Proof, Available online.

[PS73] M. Paterson and L. Stockmeyer. On the number of nonscalar multiplications necessary to evaluate polynomials. *SIAM J. on Computing*, 2(1):60–66, 1973.

[Rag97] J.-F. Ragot. *Sur la factorisation absolue des polynômes*. PhD thesis, Université de Limoges (France), 1997.

[Rag02] J.-F. Ragot. Probabilistic absolute irreducibility test for polynomials. *J. Pure Appl. Algebra*, 172(1):87–107, 2002.

[Rot76] M. Rothstein. *Aspects of symbolic integration and simplification of exponential and primitive functions*. PhD thesis, University of Wisconsin-Madison (USA), 1976.

[Rot77] M. Rothstein. A new algorithm for the integration of exponential and logarithmic functions. In *Proceedings of the 1977 MACSYMA Users Conference*, pages 263–274. NASA Pub. CP-2012, 1977.

[Rup86] W. M. Ruppert. Reduzibilität ebener Kurven. *J. Reine Angew. Math.*, 369:167–191, 1986.

[Rup99] W. M. Ruppert. Reducibility of polynomials $f(x, y)$ modulo p. *J. Number Theory*, 77(1):62–70, 1999.

[Rup04] D. Rupprecht. Semi-numerical absolute factorization of polynomials with integer coefficients. *J. Symbolic Comput.*, 37:557–574, 2004.

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

[Sch00] A. Schinzel. *Polynomials with special regard to reducibility*, volume 77 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2000.

[SS93] T. Sasaki and M. Sasaki. A unified method for multivariate polynomial factorizations. *Japan J. Indust. Appl. Math.*, 10(1):21–39, 1993.

[Ste02] A. Steel. A new scheme for computing with algebraically closed fields. In *Algorithmic number theory (Sydney, 2002)*, volume 2369 of *Lecture Notes in Comput. Sci.*, pages 491–505. Springer-Verlag, 2002.

[Sto00] A. Storjohann. *Algorithms for matrix canonical forms*. PhD thesis, ETH, Zürich (Switzerland), 2000.

[SU97] M. F. Singer and F. Ulmer. Linear differential equations and products of linear forms. *J. Pure Appl. Algebra*, 117/118:549–563, 1997.

[SVW02] A. J. Sommese, J. Verschelde, and C. W. Wampler. Symmetric functions applied to decomposing solution sets of polynomial systems. *SIAM J. Numer. Anal.*, 40(6):2026–2046, 2002.

[SVW04] A. J. Sommese, J. Verschelde, and C. W. Wampler. Advances in polynomial continuation for solving problems in kinematics. *ASME J. Mech. Design*, 126(2):262–268, 2004.

[Tra76] B. M. Trager. Algebraic factoring and rational function integration. In *Proceedings of the third ACM symposium on Symbolic and algebraic computation*, pages 219–226. ACM Press, 1976.

[Tra84] B. M. Trager. *Integration of algebraic functions*. PhD thesis, M.I.T. (USA), 1984.

[Tra85] C. Traverso. A study on algebraic algorithms: the normalization. *Rend. Sem. Mat. Univ. Politec. Torino*, Special Issue:111–130, 1985.

[Zas69] H. Zassenhaus. On Hensel factorization I. *J. Number Theory*, 1(1):291–311, 1969.

[Zas78] H. Zassenhaus. A remark on the Hensel factorization method. *Math. Comp.*, 32(141):287–292, 1978.

[Zip93] R. Zippel. *Effective Polynomial Computation*. Kluwer Academic Publishers, 1993.

Guillaume Chèze, Laboratoire de Mathématiques J.-A. Dieudonné, Université de Nice Sophia-Antipolis, Parc Valrose, 06108 Nice Cedex 2, France

E-mail address: Guillaume.Cheze@math.unice.fr

Grégoire Lecerf, Laboratoire de Mathématiques, Université de Versailles Saint-Quentin-en-Yvelines, 45 avenue des États-Unis, 78035 Versailles, France

E-mail address: Gregoire.Lecerf@math.uvsq.fr