On the complexity of computing integral bases of function fields

Simon Abelard

On the complexity of computing integral bases of function fields

Simon Abelard

Laboratoire d'informatique de l'École polytechnique (LIX, UMR 7161)
CNRS, Institut Polytechnique de Paris
abelard@lix.polytechnique.fr

Abstract. Let $\mathcal{C}$ be a plane curve given by an equation $f(x, y) = 0$ with $f \in K[x][y]$ a monic irreducible polynomial. We study the problem of computing an integral basis of the algebraic function field $K(\mathcal{C})$ and give new complexity bounds for three known algorithms dealing with this problem. For each algorithm, we study its subroutines and, when it is possible, we modify or replace them so as to take advantage of faster primitives. Then, we combine complexity results to derive an overall complexity estimate for each algorithm. In particular, we modify an algorithm due to Böhm et al. and achieve a quasi-optimal runtime.

Keywords: Puiseux series · Linear algebra · Polynomial matrices

Acknowledgements. Part of this work was completed while the author was at the Symbolic Computation Group of the University of Waterloo. This paper is part of a project that has received funding from the French Agence de l'Innovation de Défense. The author is grateful to Grégoire Lecerf, Adrien Poteaux and Éric Schost for helpful discussions and to Grégoire Lecerf for feedback on a preliminary version of this paper. The author also wishes to thank the anonymous reviewers for their comments.

1 Introduction

When handling algebraic function fields, it is often helpful –if not necessary– to know an integral basis. Computing such bases has a wide range of applications from symbolic integration to algorithmic number theory and applied algebraic geometry. It is the function field analogue of a well-known and difficult problem: computing rings of integers in number fields. As often, the function field version is easier: the algorithm of Zassenhaus [25] described for number fields in the late 60's can indeed be turned into a polynomial-time algorithm for function fields which was later precisely described by Trager [23].

However, there are very few complexity results going further than just stating a polynomial runtime. Consequently, most of the existing algorithms in the literature are compared based on their runtimes on a few examples and this yields no consensus on which algorithm to use given an instance of the problem. In this

paper, we provide complexity bounds for three of the best-known algorithms to compute integral bases and provide complexity bounds based on state-of-the art results for the underlying primitives.

In this paper, we focus on the case of plane curves given by equations of the form $f(x, y) = 0$ with $f \in K[x, y]$ irreducible. Without loss of generality, we also assume that f is monic in y. We set the notation $n = \deg_y f$ and $d_x = \deg_x f$. The associated function field is $K(\mathcal{C}) = \operatorname{Frac}(K(x)[y]/f(x, y))$, it is an algebraic extension of degree n of $K(x)$. An element $h(x, y)$ of $K(\mathcal{C})$ is integral (over $K[x]$) if there exists a monic bivariate polynomial $P(x, y)$ such that $P(x, h(x, y))$ equals 0 in $K(\mathcal{C})$. The set of such elements forms a free $K[x]$-module of rank n and a basis of this module is called an integral basis of $K(\mathcal{C})$.

The irreducibility of f is required to make sure that the function field $K(\mathcal{C})$ is indeed a field and not a product of fields. If this hypothesis fails it will be detected during the factorization process in Algorithm 3 and integral bases for each factor will be computed, while Algorithm 1 will return a basis for the integral closure of $K[x]$ in $K(x)[y]/\langle f\rangle$ (see the beginning of [12, Section 6]). However, these algorithms will both fail if f is not squarefree because it means that $\operatorname{Disc}(f) = \operatorname{Res}_y\left(f, \frac{\partial f}{\partial y}\right) = 0$.

Computing integral bases of algebraic function fields has applications in symbolic integration [23] but more generally an integral basis can be useful to handle function fields. For instance, the algorithm of van Hoeij and Novocin [13] uses such a basis to "reduce" the equation of function fields and thus makes them easier to handle. The algorithm of Hess [11] to compute Riemann-Roch spaces is based on the assumption that integral closures have been precomputed. This assumption is sufficient to establish a polynomial runtime, but a more precise complexity estimate for Hess' approach requires to assess the cost of computing integral closures as well.

Our contribution. We provide complexity estimates for three algorithms dedicated to computing integral bases of algebraic function fields in characteristic 0 or greater than n. This assumption serves two purposes: it ensures the existence of Puiseux expansions used in Algorithms 1 and 3 and is also used in technical hypothesis in Algorithm 2 to compute the radical of an ideal. To the best of our knowledge, no previous bounds were given for these algorithms. Another approach which has received a lot of attention is the use of Montes' algorithm. We do not tackle this approach in the present paper, a complexity estimate has been given by Bauch in [2, Lemma 3.10] in the case of number fields. Using the Montes algorithm, a local integral basis of a Dedeking domain A at a prime ideal $\mathfrak{p}$ is computed in $O\left(n^{1+\varepsilon}\delta \log q + n^{1+\varepsilon}\delta^{2+\varepsilon} + n^{2+\varepsilon}\delta^{1+\varepsilon}\right)$ $\mathfrak{p}$-small operations, with δ the $\mathfrak{p}$-valuation of $\operatorname{Disc}(f)$, q the cardinal of $A/\mathfrak{p}$ and ε any positive real number.

Our contribution is actually not limited to a complexity analysis: the algorithms that we present have been modified so that we could establish better complexity results. We also discuss possible improvements to van Hoeij's algorithm in a particular case which is not uncommon in the literature. Our main complexity results are Theorems 1, 2 and 3. Note that we count field operations

and do not take into account the coefficient growth in case of infinite fields nor the field extensions incurred by the use of Puiseux series. We also made the choice not to delve into probabilistic aspects: all the algorithms presented here are "at worst" Las Vegas due to the use of Poteaux and Weimann's algorithm, see for instance [21, Remark 3].

We decided to give worst-case bounds and to only involve n and $\mathrm{Disc}(f)$ in our theorems so as to give ready-to-use results. Our proofs, however, are meant to allow the interested reader to derive sharper bounds involving more precise parameters such as the regularity and ramification indices of Puiseux series.

We summarize these complexity estimates in Table 1 in a simpler context: we ignore the cost of factorizations and bound both n and $d_x = \deg_x f$ by D. In this case, the input size is in $O(D^2)$ and output size in $O(D^4)$. The constant $2 \leq \omega \leq 3$ refers to a feasible exponent for matrix multiplication, see [18] for the smallest value currently known. Translating the above bound, the complexity of the Montes approach is at best in $\widetilde{O}(D^5)$ but only for computing a local integral basis at one singularity, while the algorithm detailed in Section 4 computes a global integral basis for a quasi-optimal arithmetic complexity (i.e. in $\widetilde{O}(D^4)$).

Although we hope that this will change in a near future, our contribution remains of purely theoretical nature because we crucially rely on primitives (fast computation of Popov and Hermite forms, Puiseux series and factorization over $K[[x]][y]$) which have not been implemented yet. This is the reason why we do not provide timings in the present paper and redirect to [3, Section 8] for runtime comparisons of state-of-the-art implementations.

Organization of the paper. We sequentially analyze the three algorithms: Section 2 is dedicated to van Hoeij's algorithm [12], Section 3 to Trager's algorithm [23] and Section 4 to an algorithm by Böhm et al. introduced in [3]. In each section, we first give an overview of the corresponding algorithm and insist on the parts where we perform some modifications. The algorithms we describe are variations of the original algorithms so we give no detailed proof of exactness and refer to the original papers in which they were introduced. Then, we establish complexity bounds for each algorithm by putting together results from various fields of computer algebra. We were especially careful about how to handle linear algebra, Puiseux series and factorization over $K\left[[x]\right][y]$.

Table 1. Simplified complexity estimates for computing integral bases.

Algorithm	Worst-case complexity
Trager's algorithm [23]	$\widetilde{O}(D^7)$
Van Hoeij's algorithm [12]	$\widetilde{O}(D^{\omega+4})$
Böhm et al.'s algorithm [3]	$\widetilde{O}(D^4)$

2 Van Hoeij's algorithm

2.1 Puiseux series

We recall some basic concepts about Puiseux series and refer to [24] for more details. Assuming that the characteristic of K is either 0 or $> n$, the Puiseux theorem states that $f \in K[x][y]$ has n roots in the field of Puiseux series $\bigcup_{e\geq 1} \overline{K}\left((x^{1/e})\right)$.

Following Duval [10], we group these roots into irreducible factors of f. First, one can write $f = \prod_{i=1}^{r} f_i$ with each f_i irreducible in $K[[x]][y]$. Then, for $1 \leq i \leq r$ we write $f_i = \prod_{j=1}^{\varphi_i} f_{ij}$, where each f_{ij} is irreducible in $\overline{K}[[x]][y]$. Finally, for any $(i,j) \in \{1,\dots,r\} \times \{1,\dots,\varphi_i\}$ we write

$$f_{ij} = \prod_{k=0}^{e_i-1} \left(y - S_{ij}(x^{1/e_i}\zeta_{e_i}^k)\right),$$

where $S_{ij} \in \overline{K}\left((x)\right)$ and ζ_{e_i} is a primitive e_i-th root of unity.

Definition 1. *The n fractional Laurent series $S_{ijk}(x) = S_{ij}(x^{1/e_i}\zeta_{e_i}^k)$ are called the classical Puiseux series of f above 0. The integer e_i is called the ramification index of S_{ijk}.*

Proposition 1. *For a fixed i, the f_{ij}'s all have coefficients in K_i, a degree φ_i extension of K and they are conjugated by the associated Galois action. We have $\sum_{i=1}^{r} e_i\varphi_i = n$.*

Definition 2. *[21, Definition 2] A system of rational Puiseux expansions over K (K-RPE) of f above 0 is a set $\{R_i\}_{1\leq i\leq r}$ such that*

- $R_i(T) = (X_i(T), Y_i(T)) \in K_i((T))^2$,
- $R_i(T) = (\gamma_i T^{e_i}, \sum_{j=n_i}^{\infty} \beta_{ij} T^j)$, *where* $n_i \in \mathbb{Z}$, $\gamma_i \neq 0$ *and* $\beta_{in_i} \neq 0$,
- $f_i(X_i(T), Y_i(T)) = 0$,
- *the integer e_i is minimal.*

In the above setting, we say that R_i is centered at $(X_i(0), Y_i(0))$. We may have $Y_i(0) = \infty$ if $n_i < 0$ but this cannot happen if f is monic.

Definition 3. *[21, Definition 3] The regularity index of a Puiseux series S of f with ramification index e is the smallest $N \geq \min(0, ev_x(S))$ such that no other Puiseux series S' have the same truncation up to exponent N/e. The truncation of S up to its regularity index is called the singular part of S.*

It can be shown that two Puiseux series associated to the same RPE share the same regularity index so we can extend this notion (and the notion of singular part) to RPE's.

2.2 Description of van Hoeij's algorithm

We will be looking for an integral basis of the form $p_i(x,y)/d_i(x)$, where the p_i are degree i monic polynomials in y. It is known that the irreducible factors of the denominators d_i are among the irreducible factors of the discriminant with multiplicity at least 2. We can treat these factors one by one by first looking for local integral bases at each of these factors, i.e. bases whose denominators can only be powers of such an irreducible factor. A global integral basis is then recovered from these local bases by CRT.

To compute a local integral basis at a fixed factor ϕ, van Hoeij [12] follows the following strategy. Start from $(1, y, \ldots, y^{n-1})$ and update it so that it generates a larger module, until this module is integrally closed. This basis is modified by multiplying it by an appropriate triangular matrix in the following way. Let us fix a d, then b_d must be a linear combination of the b_0, …, b_{d-1} such that $(yb_{d-1} + \sum_{i=0}^{d-1} a_i b_i)/\phi^j$ is integral with j as large as possible.

To this end, the coefficients of the linear combination are first set to be variables and we write equations enforcing the fact that the linear combination divided by ϕ has to be integral. If a solution of this system is found, the value of b_d is updated and we repeat the process so as to divide by the largest possible power of ϕ. Note that a solution is necessary unique otherwise the difference of two solutions would be an integral element with numerator of degree $d-1$, which means that the j computed in the previous step was not maximal. When there is no solution, we have reached the maximal exponent and move on to computing b_{d+1}.

For the sake of completeness, we give a description of van Hoeij's algorithm but we refer to van Hoeij's original paper [12] for a proof that this algorithm is correct. This algorithm is originally described for fields of characteristic 0 but also works in the case of positive characteristic provided that we avoid wild ramification (see [12, Section 6.2.]). To deal with this issue, we make the assumption that we are either considering characteristic zero or greater than n.

2.3 Complexity analysis

In this section, we prove the following theorem.

Theorem 1. *Let $f(x,y)$ be a degree n monic irreducible polynomial in y. Algorithm 1 returns an integral basis for the corresponding function field and costs the factorization of* $\mathrm{Disc}(f)$ *and $\widetilde{O}(n^{\omega+2} \deg \mathrm{Disc}(f))$ field operations, where $2 \leq \omega \leq 3$ is a feasible exponent for linear algebra.*

Proof. First, we need to compute the discriminant and recover its square factors, which costs a factorization of a univariate polynomial of degree $\leq nd_x$.

Then, we need to compute the Puiseux expansions η_i of f at one root of each factor in S_{fac}, up to precision $N = \max_i \sum_{i \neq j} v(\eta_i - \eta_j)$. Using the algorithm of Poteaux and Weimann [21], the Puiseux expansions are computed up to precision N in $\widetilde{O}(n(\delta + N))$ field operations, where δ stands for the valuation of

Input : A monic irreducible polynomial $f(y)$ over $K[x]$
Output : An integral basis for $K[x,y]/\langle f\rangle$
$n \leftarrow \deg_y f$;
$S_{fac} \leftarrow$ set of factors P such that $P^2 | \operatorname{Disc}(f)$;
for ϕ *in* S_{fac} **do**
  Compute α a root of ϕ (possibly in an extension);
  Compute η_i the singular parts of the n Puiseux expansions of f at α ;
  $N \leftarrow \max_i \sum_{i\neq j} v(\eta_i - \eta_j)$;
  Extend the precision of the η_i's up to N;
  $b_0 \leftarrow 1$;
  for $d \leftarrow 1$ *to* $n-1$ **do**
    $b_d \leftarrow y b_{d-1}$;
    solutionfound $\leftarrow$ true;
    Let $a_0, \dots a_{d-1}$ be variables;
    $a \leftarrow (b_d + \sum_{i=0}^{d-1} a_i b_i)/(x-\alpha)$;
    while *solutionfound* **do**
      Write the equations, i.e. the coefficients of $a(x, \eta_i(x))$ with negative power of $(x-\alpha)$ for any i;
      Solve this linear system in the a_i's;
      if *no solution* **then**
        solutionfound $\leftarrow$ false;
        else
          There is a unique solution (a_i) in $K(\alpha)^d$;
          Substitute α by x in each a_i;
          $b_d \leftarrow (b_d + \sum_{i=0}^{d-1} a_i b_i)/\phi$;
        end
      end
    end
  end
end
From all the local bases perform CRT to deduce B an integral basis;
return B;

Algorithm 1: Van Hoeij's algorithm [12]

$\operatorname{Disc}(f)$. Indeed, these expansions are computed throughout their factorization algorithm, which runs in $\widetilde{O}(n(\delta+N))$ field operations as stated in [21, Theorem 3]. Therefore, in theory, we will see that computing the Puiseux expansions has a negligible cost compared to other parts of the algorithm since $N \leq n^2$.

Another problem coming from the use of Puiseux expansions is that we have to evaluate bivariate polynomials (the b_i's) at the Puiseux expansions of f. However this matter can be dealt with by keeping them in memory and updating them along the computations. This way, for a fixed d we first initialize $b_d = yb_{d-1}$ so we just have to perform a product of Puiseux expansions at precision $O(n^2)$ and then each time b_d is updated it will amount to performing a linear combination of Puiseux expansions. Since we fix precision at $N \leq n^2$, taking into account the denominator in the exponents of the Puiseux series this amounts to

handling polynomials of degrees $\leq n^3$. Thus, in our case, arithmetic operations on Puiseux series can be performed in $\widetilde{O}(n^3)$ field operations.

The main task in this algorithm is to solve a linear system of c equations in d variables over the extension $K(\alpha)$, where c is the total number of terms of degrees < 1 in the n Puiseux expansions. In the worst case, each Puiseux series has n terms of degree < 1 and so c can be bounded above by n^2. More precisely, we can bound it by ne, where e is the maximum of the ramification indices of the classical Puiseux expansions of f.

In most cases, this system will be rectangular of size $c \times d$ so we solve it in time $\widetilde{O}(cd^{\omega-1})$ using [5, Theorem 8.6]. This step is actually the bottleneck for each iteration and using the bounds on d and c it runs in $\widetilde{O}(n^{\omega+1} \deg \phi)$ field operations, since the extension $K(\alpha)$ of K has degree $\leq \deg \phi$.

This process is iterated over the irreducible factors of the discriminant appearing with multiplicity at least 2, and for ϕ such a factor we have to solve at most $n + M(\phi)/2$ systems, where $M(\phi)$ is the multiplicity of ϕ in $\mathrm{Disc}(f)$. Indeed, each time a solution to a system is found the discriminant is divided by ϕ^2 so that cannot happen more than $M(\phi)/2$ times, but since we need to make sure that we have no solution before incrementing d we will have to handle n additional systems. Thus, for a fixed factor ϕ the cost of solving the systems is bounded by $O(n \cdot n^{\omega+1} \deg \phi + n^{\omega+1} \deg \phi M(\phi))$. Thus, the complexity is in $\widetilde{O}\left(\sum_{\phi \in S_{fac}} n^{\omega+1} M(\phi) \deg \phi + n^{\omega+2} \sum_{\phi \in S_{fac}} \deg \phi\right)$.

Remark 1. If the base field is a finite field $\mathbb{F}_q$, factoring the discriminant is done in $\widetilde{O}((nd_x)^{1.5} \log q + nd_x (\log q)^2)$ bit operations [16].

Remark 2. The above formula shows how the size of the input is unsufficient to give an accurate estimate of the runtime of van Hoeij's algorithm. Indeed, in the best possible case $\#S_{fac}$, $\deg \phi$ and $M(\phi)$ might be constant, and all the $c_{\phi,i}$'s might be equal to d, leading to an overall complexity in $O(n^{\omega+2})$. In the worst possible case however, the sum $\sum_{\phi \in S_{fac}} \deg \phi$ is equal to the degree of the discriminant, leading to an overall complexity in $\widetilde{O}(n^{\omega+2} \deg \mathrm{Disc}(f))$.

2.4 An improvement in the case of low-degree singularities

Instead of incrementally computing the b_i's, it is possible to compute one b_k by solving the exact same systems, except that this time the previous b_i's may not have been computed (and are thus set to their initial values y^i). The apparent drawback of this strategy is that it computes b_k without exploiting previous knowledge of smaller b_i's and therefore leads to solving more systems. More precisely, if we already know b_{k-1} then we have to solve $e_k - e_{k-1} + 1$ systems otherwise we may have to solve up to $e_k + 1$ systems. Using the complexity analysis above, we can bound the complexity of finding a given b_k without knowing any other b_i by $\widetilde{O}(n^2 k^{\omega-1}(e_k + 1) \deg \phi)$.

However, we know that for a fixed ϕ, the b_i's can be taken of the form $p_i(x, y)/\phi^{e_i}$ where the exponents e_i's are non-decreasing and bounded by $M(\phi)$.

Therefore, when $M(\phi)$ is small enough compared to n, it makes sense to pick a number k and compute b_k. If $b_k = y^k$ then we know that $b_i = y^i$ for any i smaller than k. If $b_k = p_k(x,y)/\phi^{M(\phi)}$ then we know that we can take $b_i = y^{i-k}b_k$ for i greater than k. In most cases neither of this will happen but then we can repeat the process recursively and pick one number between 1 and $k-1$ and another one between $k+1$ and n and repeat.

In the extreme case where we treat $M(\phi)$ as a constant (but $\deg\phi$ is still allowed to be as large as $\deg \mathrm{Disc}(f)/2$) this approach saves a factor $\widetilde{O}(n)$ compared to the iterative approach computing the b_i's one after another. This is summarized by the following proposition.

Proposition 2. *Let $f(x,y)$ be a degree n monic irreducible polynomial in y such that irreducible factors of $\mathrm{Disc}(f)$ only appear with exponent bounded by an absolute constant. The above modification of van Hoeij's algorithm returns an integral basis for the corresponding function field and costs a univariate factorization of degree $\leq nd_x$ and $\widetilde{O}(n^{\omega+1}\deg \mathrm{Disc}(f))$ field operations, where ω is a feasible exponent for linear algebra.*

Proof. Let us first assume that $M(\phi) = 1$, then the problem is just to find the smallest k such that $e_k = 1$. Since the e_i's are non-decreasing, we can use binary search and find this k after computing $O(\log n)$ basis elements b_i's, for a total cost in $\widetilde{O}(n^{\omega+1}\deg\phi)$ and we indeed gain a quasi-linear factor compared to the previous approach. As long as $M(\phi)$ is constant, a naive way to get the same result is to repeat binary searches to find the smallest k such that $e_k = 1$, then the smallest k such that $e_k = 2$ and so on.

Remark 3. Such extreme cases are not uncommon among the examples presented in the literature and we believe that beyond this extreme, there will be a trade-off between this strategy and the classical one for non-constant but small multiplicities. We do not investigate this trade-off further because finding proper turning points should be addressed in practice as it depends both on theory and implementation.

3 Trager's algorithm

3.1 A description of Trager's algorithm

We first need to introduce the notion of discriminant of a module as this will give us a measure of the "size" of $K[x]$-modules as well as a stopping criterion in the following algorithm.

Consider n elements $v_1, \dots, v_n$ in $K(x)[y]$, since this is a degree n separable extension of $K(x)$, we can define n distinct embeddings σ_i into an algebraic closure. The matrix $(\sigma_i(v_j)_{i,j})$ is called the conjugate matrix of $(v_1, \dots, v_n)$.

Definition 4. *The discriminant of $(v_1, \dots, v_n)$ is the square of the determinant of the conjugate matrix of $(v_1, \dots, v_n)$.*

Definition 5. *Let V be a $K[x]$-module of rank n and $(v_1, \ldots, v_n)$ a $K[x]$-basis for V. The discriminant of V, denoted by* $\mathrm{Disc}(V)$*, is the ideal generated by the discriminant of $(v_1, \ldots, v_n)$ defined above.*

Computing an integral basis amounts to computing the integral closure of the $K[x]$-module generated by the powers of y. Trager's algorithm [23] computes such an integral closure iteratively using the following integrality criterion to decide when to stop. Note that there exists many similar algorithms like Round 2 and Round 4 using various criteria for integrality. A more precise account on these algorithms and their history is given in the final paragraphs of [9, Section 2.7].

Proposition 3. *[23, Theorem 1] Let R be a principal domain ($K[X]$ in our case) and V a domain that is a finite integral extension of R. Then V is integrally closed if and only if the idealizer of every prime ideal containing the discriminant equals V.*

Proof. See [23].

More precisely, Trager's algorithm uses the following corollary to the above proposition:

Proposition 4. *[23, Corollary 2] The module V is integrally closed if and only if the idealizer of the radical of the discriminant equals V.*

Starting from any basis of integral elements generating a module V the idea is to compute $\hat{V}$ the idealizer of the radical of the product of all such ideals in V. Either $\hat{V}$ is equal to V and we have found an integral basis, or $\hat{V}$ is strictly larger and we can repeat the operation. We therefore build a chain of modules whose length has to be finite. Indeed, the discriminant of each V_i has to be a strict divisor of that of V_{i-1}.

Computing the radical. Following Trager, we avoid computing the radical of the ideal generated by $\mathrm{Disc}(f)$ directly. First, we note that this radical is the intersection of the radical of the prime ideals generated by the irreducible factors of $\mathrm{Disc}(f)$. Let P be such a factor, we then use the fact that in characteristic zero or greater than n, the radical of $\langle P \rangle$ is exactly the so-called P-trace radical of V (see [23]) i.e. the set $J_P(V) = \{u \in V | \forall w \in V, P | \operatorname{tr}(uw)\}$, where the trace $\operatorname{tr}(w)$ is the sum of the conjugates of a $w \in K(x)[y]$ viewed as a degree n algebraic extension of $K(x)$.

The reason we consider this set is that it is much easier to compute than the radical. Note that Ford and Zassenhaus' Round 2 algorithm is designed to handle the case where this assumption fails but we do not consider this possibility because if it should happen it would be more suitable to use another algorithm designed by van Hoeij for the case of small characteristic [14]. This latter algorithm is different from the one we detailed in Section 2 but follows the same principle, replacing Puiseux series by a criterion for integrality based on the Frobenius endomorphism.

Input : A degree n monic irreducible polynomial $f(y)$ over $K[x]$
Output : An integral basis for $K[x,y]/\langle f\rangle$
$D \leftarrow \mathrm{Disc}(f)$;
$B \leftarrow (1, y, \dots, y^{n-1})$;
while *true* **do**
  Set V the $K[x]$-module generated by B;
  $Q \leftarrow \prod P_i$, where $P_i^2 | D$;
  If Q is a unit then return B;
  Compute $J_Q(V)$ the Q-trace radical of V;
  Compute $\hat{V}$ the idealizer of $J_Q(V)$;
  Compute M the change of basis matrix from $\hat{V}$ to V;
  Compute $\det M$, if it is a unit then return V;
  Update B by applying the change of basis;
  $D \leftarrow D/(\det M)^2$;
  $V \leftarrow \hat{V}$;
end

Algorithm 2: A bird's eye view of Trager's algorithm [23]

Finally, for $Q = \prod P_i$ we define $J_Q(V)$, the Q-trace radical of V, to be the intersection of all the $J_{P_i}(V)$. Here, we further restricted the P_i's to be the irreducible factors of $\mathrm{Disc}(f)$ whose square still divide $\mathrm{Disc}(f)$. In what follows, we summarize how $J_Q(V)$ is computed in Trager's algorithm. Once again, we refer to [23] for further details and proofs.

Let M be the trace matrix of the module V, i.e. the matrix whose entries are the $(\mathrm{tr}(w_i w_j))_{i,j}$, where the w_i's form a basis of V. An element u is in the Q-trace radical if and only if Mu is in $Q \cdot K[x]^n$. In Trager's original algorithm, the Q-trace radical is computed via a $2n \times n$ row reduction and one $n \times n$ polynomial matrix inversion.

We replace this step and compute a $K[x]$-module basis of the Q-trace radical by using an algorithm of Neiger [20] instead. Indeed, given a basis w_i of the $K[x]$-module v, the Q-trace radical can be identified to the set

$$\left\{ f_1, \dots, f_n \in K[x]^n \;\middle|\; \forall 1 \le j \le n,\ \sum_{i=1}^{n} f_i \,\mathrm{tr}(w_i w_j) = 0 \bmod Q(x) \right\}.$$

Using [20, Theorem 1.4] with $n = m$ and the shift $s = 0$, there is a deterministic algorithm which returns a basis of the Q-trace radical in Popov form for a cost of $\widetilde{O}(n^\omega \deg Q)$ field operations.

Computing the idealizer. The idealizer of an ideal $\mathfrak{m}$ of V is the set of $u \in \mathrm{Frac}(V)$ such that $u\mathfrak{m} \subset \mathfrak{m}$. Let M_i represent the multiplication matrix by m_i with input basis $(v_1, \dots, v_n)$ and output basis $(m_1, \dots, m_n)$. We define M to be the concatenation of such matrices, namely $M = (M_1^t, \dots, M_n^t)^t$. Then to find the elements $\sum_{i=1}^n u_i v_i$ in the idealizer we have to find all $u = (u_1, \dots, u_n)^t \in K(x)^n$ such that $Mu \in K[x]^{n^2}$. Note that building these multiplication matrices has negligible cost (in $O(n^2)$ field operations) using the technique of [22].

Following Trager, we row-reduce the matrix M and consider $\hat{M}$ the top left $n \times n$ submatrix and the elements of the idealizer are now exactly the u such that $\hat{M}u \in K[x]^n$. Thus, the columns of $\hat{M}^{-1}$ form a basis of the idealizer. Furthermore, the transpose of $\hat{M}^{-1}$ is the change of basis matrix from V_i to V_{i+1}.

3.2 Complexity analysis

The purpose of this section is to prove the following theorem.

Theorem 2. *Consider f a degrees n monic irreducible polynomial in $K[x][y]$, then Algorithm 2 returns an integral basis for the cost of factoring* $\mathrm{Disc}(f)$ *and* $\widetilde{O}(n^5 \deg \mathrm{Disc}(f))$ *operations in K.*

Proof. The dominant parts in this algorithm are the computations of radicals and idealizers, which have been reduced to linear algebra operations on polynomial matrices. First, we have already seen how to compute the Q-trace radical $J_Q(V)$ in $\widetilde{O}(n^\omega \deg Q)$ field operations using the algorithm presented in [20].

To compute the idealizer of $J_Q(V)$, we row-reduce a $n^2 \times n$ matrix with entries in $K[x]$ using naive Gaussian elimination. This costs a total of $O(n^4)$ operations in $K(x)$.

Then we extract the top $n \times n$ square submatrix $\hat{M}$ from this row-reduced $n^2 \times n$ matrix and invert it for $\widetilde{O}(n^\omega)$ operations in $K(x)$. The inverse $\hat{M}^{-1}$ is a basis of a module $\hat{V}$ such that $V \subset \hat{V} \subset \overline{V}$.

To translate operations in $K(x)$ into operations in K, one can bound the degrees of all the rational fractions encountered, however it is quite fastidious to track degree growth while performing the operations described above. In fact, we exploit the nature of the problem we are dealing with.

Our first task is to row-reduce a matrix M built such that a $u = \sum_{i=1}^{n} \rho_i v_i$ is in $\hat{V}$ if and only if $M(\rho_1, \ldots, \rho_n)^t \in K[x]^n$. The ρ_i's are rational fractions but their denominators divide Q. Therefore, we fall back to finding solutions of $M(\tilde{u}_1, \ldots, \tilde{u}_n)^t \in (Q(x) \cdot K[x])^n$, where the $\tilde{u}_i$'s are polynomials. In this case, it does no harm to reduce the entries of the matrix M modulo Q, however performing Gaussian elimination will induce a degree growth that may cause us to handle polynomials of degree up to $n \deg Q$ instead of $\deg Q$. With this bound, the naive Gaussian elemination costs a total of $O(n^5 \deg Q)$ operations in K.

After elimination, we retrieve a $n \times n$ matrix $\hat{M}$ whose entries have degrees bounded by $n \deg Q$. Inverting it will cause another degree increase by a factor at most n. Thus, the inversion step has cost in $\widetilde{O}(n^{\omega+2} \deg Q)$. Since $\omega \leq 3$, each iteration of Trager's algorithm has cost bounded by $O(n^5 \deg Q)$.

Now, let us assess how many iterations are necessary. Let us assume that we are exiting step i and have just computed V_{i+1} from V_i. Let us consider P a square factor of $\mathrm{Disc}(V_i)$. Let $\mathfrak{m}$ be a prime ideal of V_i containing P. Let us consider $u \in V_{i+1}$, then by definition $uP \in \mathfrak{m}$ because $P \in \mathfrak{m}$ and therefore $u \in \frac{1}{P}\mathfrak{m} \subset \frac{1}{P}V_i$. Thus, $V_{i+1} \subset \frac{1}{P}V_i$. This means that at each step i we have

$\mathrm{Disc}(V_{i+1}) = \mathrm{Disc}(V_i)/Q_i^2$, where Q_i is the product of square factors of $\mathrm{Disc}(V_i)$. Thus, the total number of iterations is at most half the multiplicity of the largest factor of $\mathrm{Disc}(f)$.

More precisely, if we assume that the irreducible factors of $\mathrm{Disc}(f)$ are r polynomials of respective degrees d_i and multiplicity ν_i, then the overall complexity of Trager's algorithm is in

$$\widetilde{O}\left(\sum_{i=1}^{\nu} n^5 \sum_{j\leq r,\, \nu_j \geq 2i} d_j\right),$$

where $\nu = \lfloor \max \nu_i/2 \rfloor$.

Since $\sum_{i=1}^{r} \nu_i d_i \leq \deg \mathrm{Disc}(f)$, the above bound is in $\widetilde{O}(n^5 \deg \mathrm{Disc}(f))$, which ranges between $\widetilde{O}(n^6 d_x)$ and $\widetilde{O}(n^5)$ depending on the input f.

Remark 4. In the above proof, our consideration of degree growth seems quite pessimistic given that the change of basis matrix has prescribed determinant. It would be appealing to perform all the computations modulo Q but it is unclear to us whether the algorithm remains valid. Another possibility of improvement would be to apply a more sophisticated technique than Gaussian elimination. However, Trager's algorithm manipulates $n^2 \times n$ polynomial matrices of degrees up to $\deg \mathrm{Disc}(f)$ so it runs in time $\Omega(n^3 \deg \mathrm{Disc}(f))$, which is no better than the bound we give in next section.

4 Integral bases through Weierstrass factorization and truncations of Puiseux series

Like van Hoeij's algorithm, this algorithm due to Böhm et al. [3] relies on computing local integral bases at each "problematic" singularity and then recovering a global integral basis. But this algorithm splits the problem further into computing a local contributions to an integral basis at each branch of each singularity.

More precisely, given a reduced Noetherian ring A we denote by $\overline{A}$ its normalization i.e. the integral closure of A in its fraction field $\mathrm{Frac}(A)$. In order to compute the normalization of $A = K[x,y]/\langle f(x,y)\rangle$ we use the following result to perform the task locally at each singularity.

Proposition 5. *[3, Proposition 3.1] Let A be a reduced Noetherian ring with a finite singular locus $\{P_1, \dots, P_s\}$. For $1 \leq i \leq s$, let an intermediate ring $A \subset A^{(i)} \subset \overline{A}$ be given such that $A^{(i)}_{P_i} = \overline{A_{P_i}}$. Then $\sum_{i=1}^{s} A^{(i)} = \overline{A}$.*

Proof. See the proof of [4, Proposition 3.2].

Each of these intermediate rings is respectively called a *local contribution* to $\overline{A}$ at P_i. In the case where $A^{(i)}_{P_j} = A_{P_j}$ for any $j \neq i$, we say that $A^{(i)}$ is a *minimal local contribution* to $\overline{A}$ at P_i. Here, we consider the case $A = K[x,y]/\langle f(x,y)\rangle$

Input : A monic irreducible polynomial $f(y)$ over $K[x]$
Output : An integral basis for $K[x,y]/\langle f\rangle$
$n \leftarrow \deg_y f$;
$S_{fac} \leftarrow$ set of factors ϕ such that $\phi^2 | \operatorname{Disc}(f)$;
for ϕ *in* S_{fac} **do**
  Compute α a root of ϕ (possibly in an extension);
  Apply a linear transform to fall back to the case of a singularity at $x = 0$;
  Compute the maximal integrality exponent $E(f)$;
  Using Proposition 9, factor f over $K[[x]][y]$;
  Compute the Bézout relations of Proposition 7;
  Compute integral bases for each factor as in Section 4.1;
  As in Section 4.2, recover the local contribution corresponding to ϕ;
  (For this, use Proposition 7 and Proposition 11)
end
From all the local contributions, use CRT to deduce an integral basis B;
return B;

Algorithm 3: Adaptation of the algorithm by Böhm et al. [3]

and will compute minimal local contributions at each singularity of f. This is summarized in Algorithm 3.

In this section, we revisit the algorithm presented by Böhm et al. in [3] and replace some of its subroutines in order to derive a complexity bound stated in Theorem 3. Note that these modifications are performed solely for the sake of complexity and rely on algorithms for which implementations may not be available. Our new description makes this algorithm both simpler and more efficient because Hensel lifting remains "hidden" within Poteaux and Weimann's factorization algorithm over $K[[x]][y]$. Some useful quantities are also shown to come as byproducts of the factorization so we avoid recomputing them.

Theorem 3. *Let $f(x,y)$ be a degree n irreducible monic polynomial in y. Then Algorithm 3 returns an integral basis of $K[x,y]/\langle f\rangle$ and costs a factorization of $\operatorname{Disc}(f)$ over K, at most n factorizations of degree n polynomials over an extension of K of degree $\leq \deg \operatorname{Disc}(f)$ and $\widetilde{O}(n^2 \deg \operatorname{Disc}(f))$ operations in K.*

Remark 5. The factorizations incurred by the use of Poteaux and Weimann's algorithm are only necessary to ensure that quotient rings are actually fields, this cost can be avoided by using the D5 principle [8] at the price of a potential complexity overhead. However, using directed evaluation [15] yields the same result without hurting our complexity bounds.

4.1 Computing normalization at one branch

Let us first address the particular case when $f(x,y)$ is an irreducible Weierstrass polynomial. This way, we will be able to compute integral bases for each branches at a given singularity. The next section will then show how to glue this information first into a local integral basis and then a global integral basis can

be computed using CRT as in van Hoeij's algorithm. The main result of this section is the following proposition.

Proposition 6. *Let g be an irreducible Weierstrass polynomial of degree m whose Puiseux expansions have already been computed up to sufficiently large precision ρ. An integral basis for the normalization of $K[[x]][y]/\langle g\rangle$ can be computed in $\widetilde{O}(\rho m^2)$ operations in K.*

As in van Hoeij's algorithm, the idea is to compute for any $1 \leq d < m$ a polynomial $p_d \in K[x][y]$ and an integer e_d such that $p_d(x,y)/x^{e_d}$ is integral and e_d is maximal. We clarify this notion of maximality in the following definition.

Definition 6. *Let $P \in K[x][y]$ be a degree d monic polynomial (in y). We say that P is d-maximal if there exists an exponent e_d such that $P(x,y)/x^{e_d}$ is integral and there is no degree d monic polynomial Q such that $Q(x,y)/x^{e_d+1}$ is integral.*

Remark 6. To the best of our knowledge this notion has not received a standard name in the literature and was often referred to using only the word maximal.

Let us consider the m Puiseux expansions γ_i of g. Since g is irreducible, these expansions are conjugated but let us first make a stronger assumption: there exists a $t \in \mathbb{Q}$ such that all the terms of degree lower than t of the expansions γ_i are equal and the terms of degree t are conjugate. We truncate all these series by ignoring all terms of degree greater or equal to t. This way, all the expansions share the same truncation $\overline{\gamma}$.

Lemma 1. *[3, Lemma 7.5] Using the notation and hypotheses of previous paragraph, for any $1 \leq d < m$ the polynomial $p_d = (y - \overline{\gamma})^d$ is d-maximal.*

Proof. See [3].

In a more general setting, more truncations are iteratively performed so as to fall back in the previous case. We recall below the strategy followed in [3] for the sake of completeness.

Initially we have $g_0 = g = \prod_{i=1}^{m}(y - \gamma_i)$. We compute the smallest exponent t such that the expansions γ_i are pairwise different. We truncate the expansions to retain only the exponents smaller than t and denote these truncations $\gamma_j^{(1)}$. Among these m expansions, we extract a set of r mutually distinct expansions which we denote by η_i. Note that by local irreducibility, each of these expansions correspond to exactly m/r identical $\gamma_j^{(1)}$'s. We further denote $\overline{g_0} = \prod_{i=1}^{m}(y - \gamma_i^{(1)})$ and $g_1 = \prod_{i=1}^{r}(y - \eta_i)$ and $u_1 = m/r$. We actually have $\overline{g_0} = g_1^{u_1}$.

We recursively repeat the operation: starting from a polynomial $g_{j-1} = \prod_{i=1}^{r_{j-1}}(y - \eta_i)$, we look for the first exponent such that all the truncations of the η_i are pairwise different. Truncating these expansions up to exponent strictly smaller, we compute $\overline{g_{j-1}} = \prod_{i=1}^{m_j}(y - \gamma_i^{(j)})$. Once again we retain only one expansion per set of identical truncations and we define a $g_j = \prod_{i=1}^{r_j}(y - \eta_i)$ and $u_j = m_j/r_j$.

The numerators of the integral basis that the algorithm shall return are products of these g_i's. Loosely speaking, the g_i have decreasing degrees in y and decreasing valuations so for a fixed d the denominator p_d is chosen of the form $\prod g_i^{\nu_i}$ where the ν_i's are incrementally built as follows: ν_1 is the largest integer such that $\deg_y(g_1^{\nu_1}) \leq d$ and $\nu_1 \leq u_1$, then ν_2 is the largest integer such that $\deg_y(g_1^{\nu_1} g_2^{\nu_2}) \leq d$ and $\nu_2 \leq u_2$, and so on. This is Algorithm 6 of [3], we refer to the proof of [3, Lemma 7.8] for exactness.

Since we assumed that we are treating a singularity at 0, the denominators are powers of x. The proper exponents are deduced in the following way: for each g_i we keep in memory the set of expansions that appear, we denote this set by N_{g_i}. Then for any γ in the set Γ of all Puiseux expansions of g we compute $\sigma_i = \sum_{\eta \in N_{g_i}} v(\gamma - \eta)$ which does not depend on the choice of $\gamma \in \Gamma$. For any j, if $p_j = \prod_k g_k^{\nu_k}$ then the exponent e_j of the denominator is given by $\lfloor \sum_k \nu_k \sigma_k \rfloor$. Detailed justifications of this are given in [3].

Complexity analysis. Let us now give a proof of Proposition 6. To do so, remark that the g_k's are polynomials whose Puiseux series are precisely the truncation η_i's of the above $\gamma_j^{(i)}$. Equivalently, one can say that the g_k's are the norms of the Puiseux expansions η_i's.

To compute them, we can appeal to Algorithm NormRPE of Poteaux and Weimann [21, Section 4.1.]. Suppose we know all the expansions involved up to precision ρ sufficiently large. These expansions are not centered at $(0, \infty)$ because g is monic. Therefore, the hypotheses of [21, Lemma 8] are satisfied and Algorithm NormRPE compute each of the g_i's above in time $\widetilde{O}(\rho \deg_y(g_i)^2)$.

Then we remark that the total number of such g_i's is in $O(\log m)$. Indeed, at each step the number of expansions to consider is at least halved (Puiseux expansions are grouped according to their truncations being the same, at least two series having the same truncation). Since the degree of each g_i is no greater than $m-1$, all these polynomials can be computed in $\widetilde{O}(m^2\rho)$ operations in K.

Once the g_i's are known we can deduce the numerators p_i's as explained above. Building them incrementally starting from p_1 each p_i is either equal to a g_j or can be expressed as one product of quantities that were already computed (either a g_j or a p_k for $k < i$). Therefore, computing all the numerators amounts to computing at most m products of polynomials whose degrees are bounded by m over $K[x]/\langle x^\rho \rangle$. Using Schönhage-Strassen's algorithm for these products the total cost is in $\widetilde{O}(\rho m^2)$ operations in K. The cost of computing denominators is negligible so this concludes the proof.

4.2 Branch-wise splitting for integral bases

Once again, let us assume that we are treating the local contribution at the singularity $x = 0$. In the setting of van Hoeij's algorithm, this corresponds to dealing with a single irreducible factor of the discriminant. We further divide the problem by considering the factorization $f = f_0 \prod_{i=1}^r f_i$, where f_0 is a unit in $K[[x]][y]$ and the other f_i's are irreducible Weierstrass polynomials in $K[[x]][y]$.

We can apply the results from the previous section to each f_i for $i > 0$ in order to compute an integral basis of $K[[x]][y]/\langle f_i\rangle$. In this section, we deal with two problems: we explain how to compute the factorization of f and how to efficiently perform an analogue of the Chinese Remainder Theorem to compute an integral basis of $K[[x]][y]/\langle f_1 \cdots f_r\rangle$ from the integral bases at each branch. For the sake of completeness, we recall in Section 4.3, how Böhm et al. take f_0 into account and deduce a minimal local contribution at any given singularity.

Proposition 7. *[3, Proposition 5.9] Let $f_1,\dots,f_r$ be the irreducible Weierstrass polynomials in $K[[x]][y]$ appearing in the factorization of f into branches. Let us set $h_i = \prod_{j=1,j\neq i} f_j$. Then f_i and h_i are coprime in $K((x))[y]$ so that there exist polynomials a_i, b_i in $K[[x]][y]$ and positive integers c_i such that $a_i f_i + b_i h_i = x^{c_i}$ for any $1 \leq i \leq r$.*

Furthermore, the normalization of $K[[x]][y]/(f_1 \cdots f_r)$ splits as

$$\overline{K[[x]][y]/\langle f_1 \cdots f_r\rangle} \cong \bigoplus_{i=1}^{r} \overline{K[[x]][y]/\langle f_i\rangle}$$

and the splitting is given explicitly by

$$(t_1 \bmod f_1, \dots, t_r \bmod f_r) \mapsto \sum_{i=1}^{r} \frac{b_i h_i t_i}{x^{c_i}} \bmod f_1 \cdots f_r.$$

Proof. See [7, Theorem 1.5.20].

The following corollary is used recover an integral basis for $\overline{K[[x]][y]/\langle f_1 \cdots f_r\rangle}$.

Proposition 8. *[3, Corollary 5.10] With the same notation, let*

$$\left(1, \frac{p_1^{(i)}(x,y)}{x^{e_1^{(i)}}}, \dots, \frac{p_{m_i-1}^{(i)}(x,y)}{x^{e_{m_i-1}^{(i)}}}\right)$$

represent an integral basis for f_i, where each $p_j^{(i)} \in K[x][y]$ is a monic degree j polynomial in y. For $1 \leq i \leq r$, set

$$\mathcal{B}^{(i)} = \left(\frac{b_i h_i}{x^{c_i}}, \frac{b_i h_i p_1^{(i)}}{x^{c_i+e_1^{(i)}}}, \dots, \frac{b_i h_i p_{m_i-1}^{(i)}}{x^{c_i+e_{m_i-1}^{(i)}}}\right).$$

Then $\mathcal{B}^{(1)} \cup \cdots \cup \mathcal{B}^{(r)}$ is an integral basis for $f_1 \cdots f_r$.

In [3], these results are not used straightforwardly because the authors remarked that it was time-consuming in practice. Instead, the c_i's are computed from the singular parts of the Puiseux expansions of f and polynomials β_i replace the b_i's, playing a similar role but being easier to compute.

Indeed, these β_i's are computed in [3, Algorithm 8] and they are actually products of the polynomials g_i's already computed by [3, Algorithm 7], which is the algorithm that we detailed above to describe the computation of an integral

basis for each branch. The only new data compute in order to deduce the β_i's are the suitable exponents of the g_i's. This is achieved through solving linear congruence equations. This step can be fast on examples considered in practice and we also note that the β_i's seem more convenient to handle because they are in $K[x][y]$ and they contain less monomials than the b_i's. However the complexity of this problem (often denoted LCON in the literature) has been widely studied, see for example [1,6] but, to the best of our knowledge, none of the results obtained provide bounds that we could use here.

For the sake of complexity, we therefore suggest another way which is based on computing the b_i's of Proposition 7. We also compute the factorization of f into branches in a different way: instead of following the algorithms of [3, Section 7.3 & 7.4] we make direct use of the factorization algorithm of Poteaux and Weimann [21] so we also invoke their complexity result [21, Theorem 3] which is recalled below. Another advantage to this is that we will see that the b_i's can actually be computed using a subroutine involved in the factorization algorithm, which simplifies even further the complexity analysis.

Proposition 9. *[21, Theorem 3] There exists an algorithm that computes the irreducible factors of f in $K[[x]][y]$ with precision N in an expected $\widetilde{O}(\deg_y(f)(\delta+N))$ operations in K plus the cost of one univariate factorization of degree at most $\deg_y(f)$, where δ stands for the valuation of $\mathrm{Disc}(f)$.*

Proof. See [21, Section 7].

Let us now get back to the first steps of Algorithm 3: we have to compute $E(f)$ to assess up to what precision we should compute the Puiseux series and then compute the factorization of f, the integers c_i and the polynomials b_i.

In each section, we tried to keep the notation of the original papers as much as we could which is why we introduced $E(f)$ but the definition given in [3, Section 4.8] is exactly the same as the N in van Hoeij's paper [12]. This bound can be directly computed from the singular part of the Puiseux expansions of f. We recall its definition: $E(f) = \max_i \sum_{i\neq j} v(\gamma_i - \gamma_j)$, where the γ_i's are the Puiseux expansions of f.

Following [3], we need to compute the factorization of f into branches up to precision $E(f)+c_i$. Using Poteaux and Weimann's factorization algorithm from Proposition 9, we can compute the factors f_i up to the desired precision.

Furthermore, using a subroutine contained within this algorithm, we can compute the Bézout relation $a_i f_i + b_i h_i = x^{c_i}$ up to precision $E(f)+c_i$. This is detailed in [21, Section 4.2], where our c_i is the lifting order κ and our f_i and h_i are respectively the H and G of Poteaux and Weimann. The algorithm used to compute the Bézout relations is due to Moroz and Schost [19].

Complexity analysis. We analyze the cost of the computations performed in this section and summarize them by the following proposition.

Proposition 10. *Let $f(x,y)$ be a degree n monic irreducible polynomial in y and let δ be the x-valuation of $\mathrm{Disc}(f)$. Then the integers c_i's and $E(f)$, a factorization in branches $f = f_0 \prod_{i=1}^{r} f_i$ as well as the polynomials a_i's and b_i's*

of Proposition 7 can be computed up to precision $E(f) + c_i$ for a univariate factorization degree n over K and a total of $\widetilde{O}(n^2\delta)$ field operations.

Proof. First, the singular parts of the Puiseux series of f above 0 are computed for $\widetilde{O}(n\delta)$ field operations by [21, Theorem 1]. This allows us to compute $E(f)$.

Then we compute the factorization in branches up to a sufficient precision to compute the c_i's. We then extend the precision further so as to compute the factorization and the Bézout relations $a_i f_i + b_i h_i = x^{c_i}$ up to precision $E(f) + c_i$.

Invoking [19, Corollary 1], computing a single Bézout relation up to precision $E(f) + c_i$ costs $\widetilde{O}(n(E(f) + c_i))$ field operations. Computing the factorization of f in branches up to the same precision with Proposition 9 accounts for $\widetilde{O}(n(\delta + c_i + E(f))$ operations in K and one univariate factorisation of degree n over K.

Using [3, Definition 4.14], we note that $E(f)$ can also be seen as e_{n-1}, which is bounded by the valuation δ of the discriminant because we assumed that we were handling a singularity at $x = 0$. Thanks to [21, Proposition 8] we can bound c_i by the valuation of $\frac{\partial f}{\partial y}$ which is itself bounded by δ.

Putting these bounds together, the overall cost is one univariate factorization of degree n over K and $\widetilde{O}(n\delta)$ operations in K for the factorization step while the n Bézout relations requires $\widetilde{O}(n^2\delta)$ operations in K. This concludes the proof.

4.3 Contribution of the invertible factor f_0

To deal with this problem, we reuse the following result without modification.

Proposition 11. *[3, Proposition 6.1] Let $f = f_0 g$ be a factorization of f with f_0 and g in $K[[x]][y]$, f_0 a unit and g a Weierstrass polynomial of y-degree m. Let $\left(p_0 = 1, \frac{p_1}{x^{e_1}}, \dots, \frac{p_{m-1}}{x^{e_{m-1}}}\right)$ be an integral basis for $K[[x]][y]/\langle g \rangle$ such that the p_i's are degree i monic polynomials in $K[x][y]$ and let $\overline{f_0}$ be a monic polynomial in $K[x][y]$ such that $\overline{f_0} = f_0 \bmod x^{e_{m-1}}$. Let us denote $d_0 = \deg_y(\overline{f_0})$.*

Then

$$\left(1, y, \dots, y^{d_0-1}, \overline{f_0}p_0, \frac{\overline{f_0}p_1}{x^{e_1}}, \dots, \frac{\overline{f_0}p_{m-1}}{x^{e_{m-1}}}\right)$$

is an integral basis for the normalization of $K[[x]][y]/\langle f \rangle$.

Proof. See [3]

Since we handle a single singularity at 0, the previous basis is also a $K[x]$-module basis of the minimal local contribution at this singularity by [3, Corollary 6.4].

Complexity analysis. This step involves a truncation of f_0 modulo $x^{e_{m-1}}$ and m products of polynomials in $K[[x]][y]/\langle x^{e_{m-1}} \rangle$ whose y-degrees are bounded by $n = \deg_y(f)$. This incurs $\widetilde{O}(mne_{m-1})$ field operations. Since we are treating a singularity at $x = 0$, we have $e_{m-1} = O(\delta)$ with δ the valuation of $\mathrm{Disc}(f)$ so that we can simplify the above bound as $\widetilde{O}(n^2\delta)$ field operations.

4.4 Proof of Theorem 3

In this section, we put all the previous bounds together and prove Theorem 3.

Proof. As in van Hoeij's algorithm, we first compute $\mathrm{Disc}(f)$ and factor it in order to recover its irreducible square factors. For each irreducible factor ϕ such that $\phi^2 | \mathrm{Disc}(f)$, we compute the corresponding minimal local contribution. For each of them, we first perform a translation so as to handle a singularity at $x = 0$. If there are several conjugated singularities we can handle them like in van Hoeij's algorithm, at the price of a degree $\deg \phi$ extension of K which we denote by K' in this proof. Also note that through this transform the multiplicity $M(\phi)$ corresponds to the valuation δ of the discriminant.

First, we split f into branches using Proposition 10 for a cost in $\widetilde{O}(n^2 M(\phi))$ operations in K' and one univariate factorization of degree $\leq n$ over K'.

Then, at each branch f_i, we apply Proposition 6 with precision $\rho = E(f) + c_i$. Therefore, the cost of computing an integral basis at each branch f_i is in $\widetilde{O}(M(\phi) \deg_y(f_i)^2)$ operations in K'. Since $\sum_i \deg f_i \leq n$, computing the integral bases at all the branches costs $\widetilde{O}(n^2 M(\phi))$ operations in K'.

At the end of this step, we have integral bases $\mathcal{B}_i$ of the form

$$\left(1, \frac{p_1(x,y)}{x^{e_1}}, \ldots, \frac{p_{m_i-1}(x,y)}{x^{e_{m_i-1}}}\right)$$

with $m_i = \deg_y f_i$ but the p_i's are in $K'[[x]][y]$.

At first glance, this is a problem because Proposition 8 requires the p_i's to be in $K'[x][y]$. However, the power of x in the denominators is bounded a priori by $E := E(f) + \max_{1 \leq i \leq r} c_i$ so we can truncate all series beyond this exponent. Indeed, forgetting the higher order terms amounts to subtracting each element of the basis by a polynomial in $K'[x]$. Such polynomials are obviously integral elements so they change nothing concerning integrality.

We can thus apply Proposition 8 to get an integral basis for $f_1 \cdots f_r$. This costs $O(n)$ operations in $K'[[x]][y]/\langle x^E, f(x,y)\rangle$. Each such operation amounts to nE operations in K'. We have previously seen that E is in $O(M(\phi))$ so the overall cost of applying Proposition 8 is in $O(n^2 M(\phi))$ operations in K'.

After this process, the basis that we obtained must be put in "triangular form" (i.e. each numerator p_i should have degree i in y in order for us to apply Proposition 11. To do this, we first reduce every power of y greater or equal to n using the equation $f(x,y) = 0$. For a fixed i, by the Bézout relations, h_i has y-degree $\leq n - m_i$ and b_i has y-degree $< m_i$, so we have to reduce a total of $O(n)$ bivariate polynomials whose degrees in y are in $O(n)$. Using a fast Euclidean algorithm, this amounts to $\widetilde{O}(n^2)$ operations in $K'[x]/\langle x^E\rangle$, hence a cost in $O(n^2 M(\phi))$ operations in K'.

Once done, every element in the basis can be represented by a vector of polynomials in $K'[x]$ whose degrees are bounded by E. To put the above integral basis in triangular form, it suffices to compute a Hermite Normal Form of a full rank $n \times n$ polynomial matrix. Using [17, Theorem 1.2] an algorithm by Labahn, Neiger and Zhou performs this task in $\widetilde{O}(n^{\omega-1} M(\phi))$ operations in K'.

We can finally apply Proposition 11 and deduce the minimal local contribution for the factor ϕ in $\widetilde{O}(n^2 M(\phi))$ operations in K'.

Overall, given a factor ϕ, computing the corresponding minimal local contribution to the normalization of $K[\mathcal{C}]$ costs the factorization of $\mathrm{Disc}(f)$, one univariate factorization of degree $\leq n$ over K and $\widetilde{O}(n^2 M(\phi))$ operations in K'. Computing all the local contributions can therefore be done for the factorization of $\mathrm{Disc}(f)$, $\#S_{fac}$ univariate factorization of degree $\leq n$ over extensions of K of degree $\leq \max_{\phi \in S_{fac}} \deg \phi$ and $\widetilde{O}(n^2 \deg \mathrm{Disc}(f))$ operations in K.

In the case of conjugate singularities, we follow the idea of van Hoeij rather than [3, Remark 7.17] and simply replace α by x in the numerators and $(x - \alpha)$ by ϕ in the denominators because it does not harm our complexity bound. In this process, some coefficients of the numerators are multiplied by polynomials in x, which clearly preserves integrality. Since the numerators are monic in y, no simplification can occur and the basis property is also preserved.

Finally, a global integral basis for $K[x, y]/\langle f \rangle$ is deduced by a Chinese remainder theorem. This can be achieved in quasi-linear time in the size of the local bases. Each of them being in $O(n^2 \deg \mathrm{Disc}(f))$, this last CRT does not increase our complexity bound. This concludes the proof.

5 Conclusion

In the setting of Table 1, the best bound given in this paper is in $\widetilde{O}(D^4)$ which is quasi-quadratic in the input size, but quasi-linear in the output size. It is surprising that we are able to reach optimality without even treating the local factors f_i through a divide-and-conquer approach like in [21]. This would allow us to work at precision δ/n instead of δ most of the time, but this does not affect the worst-case complexity of the whole algorithm. From an implementation point of view, however, this approach will probably make a significant difference.

Note that we are still relatively far from having implementations of algorithms actually reaching these complexity bounds because we lack implementations for primitives involved in computing Popov/Hermite forms, Puiseux series and factorizations over $K\left[[x]\right][y]$. In some experiments we performed, Puiseux series were actually the most time-consuming part, which is why Trager's algorithm may remain a competitive choice despite our complexity results. We refer to [3, Section 8] for more detailed timings and experiments.

References

1. Vikraman Arvind and TC Vijayaraghavan. The complexity of solving linear equations over a finite ring. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 472–484. Springer, 2005.
2. Jens-Dietrich Bauch. Computation of integral bases. *Journal of Number Theory*, 165:382–407, 2016.
3. Janko Böhm, Wolfram Decker, Santiago Laplagne, and Gerhard Pfister. Computing integral bases via localization and Hensel lifting. *arXiv preprint arXiv:1505.05054*, 2015.

4. Janko Böhm, Wolfram Decker, Santiago Laplagne, Gerhard Pfister, Andreas Steenpaß, and Stefan Steidel. Parallel algorithms for normalization. *Journal of Symbolic Computation*, 51:99–114, 2013.
5. Alin Bostan, Frédéric Chyzak, Marc Giusti, Romain Lebreton, Grégoire Lecerf, Bruno Salvy, and Éric Schost. *Algorithmes efficaces en calcul formel*. 2017.
6. Niel de Beaudrap. On the complexity of solving linear congruences and computing nullspaces modulo a constant. *arXiv preprint arXiv:1202.3949*, 2012.
7. Theo De Jong and Gerhard Pfister. *Local analytic geometry: Basic theory and applications*. Springer Science & Business Media, 2013.
8. Jean Della Dora, Claire Dicrescenzo, and Dominique Duval. About a new method for computing in algebraic number fields. In *European Conference on Computer Algebra*, pages 289–290. Springer, 1985.
9. Claus Diem. *On arithmetic and the discrete logarithm problem in class groups of curves*. Habilitation, Universität Leipzig, 2009.
10. Dominique Duval. Rational Puiseux expansions. *Compositio mathematica*, 70(2):119–154, 1989.
11. Florian Hess. Computing Riemann–Roch spaces in algebraic function fields and related topics. *Journal of Symbolic Computation*, 33(4):425–445, 2002.
12. Mark van Hoeij. An algorithm for computing an integral basis in an algebraic function field. *Journal of Symbolic Computation*, 18(4):353–363, 1994.
13. Mark van Hoeij and Andrew Novocin. A reduction algorithm for algebraic function fields. 2008.
14. Mark van Hoeij and Michael Stillman. Computing an integral basis for an algebraic function field, 2015.
15. Joris van der Hoeven and Grégoire Lecerf. Directed evaluation. working paper or preprint, December 2018.
16. Kiran S Kedlaya and Christopher Umans. Fast polynomial factorization and modular composition. *SIAM Journal on Computing*, 40(6):1767–1802, 2011.
17. George Labahn, Vincent Neiger, and Wei Zhou. Fast, deterministic computation of the Hermite normal form and determinant of a polynomial matrix. *Journal of Complexity*, 42:44–71, 2017.
18. François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303, 2014.
19. Guillaume Moroz and Éric Schost. A fast algorithm for computing the truncated resultant. In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, pages 341–348, 2016.
20. Vincent Neiger. Fast computation of shifted Popov forms of polynomial matrices via systems of modular polynomial equations. In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, pages 365–372, 2016.
21. Adrien Poteaux and Martin Weimann. Computing Puiseux series: a fast divide and conquer algorithm. *arXiv preprint arXiv:1708.09067*, 2017.
22. Barry Marshall Trager. Algorithms for manipulating algebraic functions. *SM thesis MIT*, 1976.
23. Barry Marshall Trager. *Integration of algebraic functions*. PhD thesis, Massachusetts Institute of Technology, 1984.
24. Robert J Walker. *Algebraic curves*. 1950.
25. Hans Zassenhaus. Ein algorithmus zur berechnung einer minimalbasis über gegebener ordnung. In *Funktionalanalysis Approximationstheorie Numerische Mathematik*, pages 90–103. Springer, 1967.