

Composition modulo powers of polynomials

Joris van der Hoeven, Grégoire Lecerf

▶ **To cite this version:**

Joris van der Hoeven, Grégoire Lecerf. Composition modulo powers of polynomials. 2017. hal-01455722v2

HAL Id: hal-01455722
https://hal.science/hal-01455722v2

Preprint submitted on 27 Apr 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Composition modulo powers of polynomials

Joris van der Hoeven and Grégoire Lecerf
Laboratoire d'informatique de l'École polytechnique, CNRS UMR 7161
École polytechnique
91128 Palaiseau Cedex, France
{vdhoeven,lecerf}@lix.polytechnique.fr

ABSTRACT

Modular composition is the problem to compose two univariate polynomials modulo a third one. For polynomials with coefficients in a finite field, Kedlaya and Umans proved in 2008 that the theoretical bit complexity for performing this task could be made arbitrarily close to linear. Unfortunately, beyond its major theoretical impact, this result has not led to practically faster implementations yet. In this paper, we study the more specific case of composition modulo the power of a polynomial. First we extend previously known algorithms for power series composition to this context. We next present a fast direct reduction of our problem to power series composition.

Keywords

Modular composition; Series Composition; Algorithm; Complexity

1. INTRODUCTION

Let $\mathbb{K}$ be an effective field and let f, g, h be polynomials in $\mathbb{K}[x]$ of degrees $\leqslant n$. The problem of *modular composition* is to compute $f \circ g$ modulo h. This is a fundamental problem in complexity theory because of its applications to polynomial factorization [16, 17, 18].

From a theoretical point of view, Kedlaya and Umans achieved a major breakthrough by showing that modular compositions over a finite field $\mathbb{F}_q$ can be computed in time $(n \log q)^{1+o(1)}$. Unfortunately, the practical impact of these results has been limited so far. The best current implementations still admit nearly quadratic complexities in n. More precisely, over a field $\mathbb{K}$, denoting by $\mathsf{M}_{\mathbb{K}}(n)$ the cost to multiply two polynomials in $\mathbb{K}[x]$ of degrees $\leqslant n$, Brent and Kung [3] gave an algorithm with cost $O(\sqrt{n}\,\mathsf{M}_{\mathbb{K}}(n) + n^2)$. It uses the *baby-step giant-step* technique due to Paterson–Stockmeyer [21] and even yields a sub-quadratic cost when using fast linear algebra (see [15, p. 185] and our Theorem 2 below).

In the special case when $h = x^n$, composition modulo h corresponds to the composition of truncated power series at order n. Better practical algorithms are known for this particular situation. Brent and Kung gave an algorithm [3] with complexity $O(\sqrt{n}\ \mathsf{M}_{\mathbb{K}}(n)\ \log^{1/2} n)$ under the conditions that $g'(0)$ is invertible and that the characteristic is at least $\lceil \sqrt{n \log n} \rceil$. A variant proposed by van der Hoeven [11, Section 3.4.3] removes the condition on $g'(0)$. For fields of small positive characteristic, Bernstein [1] proposed an algorithm that is softly linear in the precision n but linear in the characteristic. Series with integer, rational or floating point coefficients can often be composed in quasi-linear time as well in suitable bit complexity models, as shown by Ritzmann [22]; see also [12].

In this paper, we focus on the intermediate case between power series and general modular compositions. More precisely, we assume h to be a power $\hbar^m$ of some polynomial $\hbar$. Composition modulo $\hbar^m$ can also be regarded as a special "base" case of composition modulo a polynomial h that can be factored into $h = h_1 \cdots h_m$. Using Chinese remaindering, one may reduce composition modulo such h to compositions modulo powers of irreducible polynomials. In a separate paper [13], we exploit this observation for the design of fast practical (although not fully general) algorithms for modular composition over finite fields.

We develop two main approaches for fast composition modulo $h = \hbar^m$. In section 3, we generalize Brent and Kung's and Bernstein's algorithms. We achieve a complexity exponent close to $3/2$, but only under the condition that n is much larger than $\deg \hbar$. In section 4, we directly reduce composition modulo h to truncated power series composition. Our main result is Theorem 11: one composition modulo h is essentially reduced to $m+1$ compositions modulo $\hbar$ and one power series composition in $\mathbb{K}[z]/(\hbar(z))[y]/(y^m)$. An important ingredient of this reduction is a softly optimal algorithm for converting between $\mathbb{K}[x]/(h(x))$ and $\mathbb{K}[z]/(\hbar(z))[y]/(y^m)$ where x is sent to $y + z$. Potential applications of these conversions might also concern algebraic algorithms for plane curves such as integral basis computations and Puiseux expansions, which usually require several Taylor expansions at roots of discriminants.

Our paper is organized as follows. In section 2, we introduce basic notations and recall several well known complexity results. We also specify the complexity model that we work with. Further technical complexity results are gathered in the appendix. In sections 3 and 4, we present our main algorithms. For testing purposes, our algorithms are implemented in the MATHEMAGIX language within the FACTORIX library (www.mathemagix.org, revision 10538).

Preprint version, April 26, 2017

2. PRELIMINARIES

2.1 Costs of elementary operations

Recall that an *effective ring* is a ring $\mathbb{A}$ with unity whose elements can be represented on a computer and such that we have algorithms for performing the ring operations. Effective fields $\mathbb{K}$ and effective algebras over an effective ring are defined similarly.

Given an effective ring $\mathbb{A}$, *algebraic complexity models* express running times in terms of the number of ring operations in $\mathbb{A}$. Unless otherwise stated, we will analyze the costs of the algorithms in this paper in this way. More precisely, our results both apply for the straight-line program and computation tree models [4, Chapter 4].

Let $\mathbb{A}$ be an effective ring with unity, let $n \in \mathbb{N}$, and denote

$$\mathbb{A}[x]_{<n} \;=\; \{P \in \mathbb{A}[x] : \deg P < n\}.$$

We write $\mathsf{M}_{\mathbb{A}} : \mathbb{N} \to \mathbb{R}^{\geqslant}$ for a cost function such that two polynomials in $\mathbb{A}[x]_{<n}$ can be multiplied using $\mathsf{M}_{\mathbb{A}}(n)$ operations in $\mathbb{A}$. It will be convenient to assume that $\mathsf{M}_{\mathbb{A}}(n)/n$ is a nondecreasing function in n. Notice that this assumption implies the *super-additivity* of $\mathsf{M}_{\mathbb{A}}$, namely $\mathsf{M}_{\mathbb{A}}(n_1) + \mathsf{M}_{\mathbb{A}}(n_2) \leqslant \mathsf{M}_{\mathbb{A}}(n_1 + n_2)$ for all $n_1 \geqslant 0$ and $n_2 \geqslant 0$.

The schoolbook algorithm allows us to take $\mathsf{M}_{\mathbb{A}}(n) = O(n^2)$. The fastest currently known algorithm [5] yields $\mathsf{M}_{\mathbb{A}}(n) = O(n \log n \log\log n)$. If $\mathbb{K}$ is a field of finite characteristic, then we even have $\mathsf{M}_{\mathbb{K}}(n) = O(n \log n\, 8^{\log^* n})$, where $\log^* n = \min\{i \in \mathbb{N} : (\log \circ \overset{i\times}{\ldots} \circ \log)(n) \leqslant 1\}$ [10].

The constant $\omega > 2$ represents a *feasible exponent* for the multiplication cost of matrices: two square matrices of size $n \times n$ can be multiplied using $O(n^\omega)$ operations in their coefficient ring. The constant $\varpi > 1.5$ is defined similarly but for multiplying a $\sqrt{n} \times \sqrt{n}$ matrix by a $\sqrt{n} \times n$ rectangular one. The best known bound $\omega < 2.3729$ is due to Le Gall [19]. This naturally yields $\varpi \leqslant (\omega + 1)/2 < 1.6845$. However the latter bound does not improve upon the earlier bound $\varpi < 1.6667$ due to Huang and Pan [14, Theorem 10.1].

2.2 Problems related to modular composition

If f and g are polynomials in $\mathbb{A}[x]$, then $f \operatorname{rem} g$ represents the remainder in the division of f by g. Let $\mathbb{A}$ be an effective ring and $\mathbb{B}$ be an effective $\mathbb{A}$-algebra. We introduce the following cost functions:

- $\mathsf{S}_{\mathbb{A}}(n)$: the cost of computing the *power series composition* $f \circ g \operatorname{rem} x^n$, where $f, g \in \mathbb{A}[x]_{<n}$.
- $\mathsf{S}_{\mathbb{B}/\mathbb{A}}(n)$: the cost of computing the *power series composition* $f \circ g \operatorname{rem} x^n$ in terms of operations in $\mathbb{A}$, where $f, g \in \mathbb{B}[x]_{<n}$.
- $\mathsf{C}_{\mathbb{A}}(n)$: the cost of computing the *modular composition* $f \circ g \operatorname{rem} h$, where $h \in \mathbb{A}[x]$ is a monic polynomial of degree n and $f, g \in \mathbb{A}[x]_{<n}$.
- $\mathsf{Q}_{\mathbb{A}}(n)$: the cost of computing the *characteristic polynomial* χ of $g \in \mathbb{A}[x]_{<n}$ modulo a monic polynomial $h \in \mathbb{A}[x]$ of degree n. This is the characteristic polynomial $\chi \in \mathbb{A}[x]$ of the multiplication endomorphism by $g \bmod h$ in $\mathbb{A}[x]/(h(x))$.
- $\mathsf{P}_{\mathbb{A}}(n)$: the cost of *modular power projections*, i.e. the cost to compute $\varphi(1)$, $\varphi(g)$, ..., $\varphi(g^{n-1} \operatorname{rem} h)$, where $h \in \mathbb{A}[x]$ is monic of degree n and φ is a linear form on $\mathbb{A}[x]_{<n}$.

Let us recall a few known results about these functions. For $a \in \mathbb{R}$, we denote $\lfloor a \rfloor = \max\{b \in \mathbb{Z} : b \leqslant a\}$ and $\lceil a \rceil = \min\{b \in \mathbb{Z} : b \geqslant a\}$.

THEOREM 1. *Let $\mathbb{K}$ be a field of characteristic p.*

a) [3] If $p = 0$ or $p > \lceil \sqrt{n \log n} \rceil$, then we may take $\mathsf{S}_{\mathbb{K}}(n) = O(\mathsf{M}_{\mathbb{K}}(n) \sqrt{n \log n})$.

b) [1] If $p > 0$, then $\mathsf{S}_{\mathbb{K}}(n) = O\Big(p\, n + \frac{p}{\log p}\, \mathsf{M}_{\mathbb{K}}(n) \log n\Big)$.

THEOREM 2. *Let h be a monic polynomial of degree n over a ring $\mathbb{A}$ and let $f, g \in \mathbb{A}[x]_{<n}$. The composed polynomial $f \circ g \operatorname{rem} h$ may be computed with*

a) $O(n\, \mathsf{M}_{\mathbb{A}}(n))$ operations in $\mathbb{A}$, or

b) $O(n^\varpi + n^{1/2}\, \mathsf{M}_{\mathbb{A}}(n))$ or $O(n^\varpi)$ operations in $\mathbb{A}$.

PROOF. The first bound is immediate. The proof of the second bound is detailed in [8, Section 12.2]. The bound $O(n^\varpi)$ follows from $\varpi > 1.5$ by taking $\mathsf{M}_{\mathbb{A}}(n) = n^{1+o(1)}$. □

For a fixed monic polynomial h in $\mathbb{A}[x]$ of degree n, the modular composition $f \circ g \operatorname{rem} h$ is a linear operation in f, for f and g in $\mathbb{A}[x]_{<n}$. The corresponding transposed application is precisely the operation of *modular power projections*: it corresponds to computing $\varphi(1)$, $\varphi(g)$, ..., $\varphi(g^{n-1} \operatorname{rem} h)$, where φ is a linear form on $\mathbb{A}[x]_{<n}$. If a modular composition algorithm with cost $\mathsf{C}_{\mathbb{A}}(n)$ can be transposed in the sense of [2], then this leads to a power projection algorithm with cost $\mathsf{P}_{\mathbb{A}}(n) = \mathsf{C}_{\mathbb{A}}(n) + O(n)$.

THEOREM 3. *Let h be a monic polynomial of degree n over a ring $\mathbb{A}$ and let $g \in \mathbb{A}[x]_{<n}$. The characteristic polynomial χ of g modulo h can be computed using*

a) $O(\mathsf{M}_{\mathbb{A}}(n^2) \log n + n\, \mathsf{M}_{\mathbb{A}}(n) \log^2 n)$ operations in $\mathbb{A}$, including divisions in $\mathbb{A}$ (the partial division in $\mathbb{A}$ is supposed to be implemented), or

b) $O(\mathsf{M}_{\mathbb{A}}(n^2) \log n)$ operations in $\mathbb{A}$, if $\mathbb{A}$ is a field, or

c) $O(n\, \mathsf{M}_{\mathbb{A}}(n) \log n)$ operations in $\mathbb{A}$, if $\mathbb{A}$ is a field with $>n$ elements, or

d) $\mathsf{P}_{\mathbb{A}}(n) + \mathsf{M}_{\mathbb{A}}(n)$ operations in $\mathbb{A}$, if there exist given inverses of $2, 3, ..., n$ in $\mathbb{A}$.

PROOF. The first two bounds directly rely on the formula $\chi(x) = \operatorname{Res}_z(g(z) - x, h(z))$ by using [20, Corollary 29]. The third bound is obtained by computing $\operatorname{Res}_z(g(z) - a, h(z))$ for $n + 1$ values of a in $\mathbb{A}$, from which χ is interpolated using $O(\mathsf{M}_{\mathbb{A}}(n) \log n)$ additional operations in $\mathbb{A}$. We refer to Appendix A.1 for the fourth bound. □

For the particular kind of moduli $h = \hbar^m$ that interest us in here, the problem of computing characteristic polynomials quite easily reduces to the case when $m = 1$:

PROPOSITION 4. *Let $\hbar$ be of degree d in $\mathbb{K}[x]$, let $h = \hbar^m$ with $n = m\,d$, and let g be in $\mathbb{K}[x]_{<n}$. Then the characteristic polynomial χ of g modulo h can be computed using $\mathsf{Q}_{\mathbb{K}}(d) + O(\mathsf{M}_{\mathbb{K}}(n))$ operations in $\mathbb{K}$.*

PROOF. The characteristic polynomial χ is simply the m-th power of the characteristic polynomial of g modulo $\hbar$. □

3. EXTENDING KNOWN ALGORITHMS

From now on, we assume the modulus h to be the m-th power of a polynomial $\hbar$ of degree d. In this section we extend the two fast known algorithms for power series composition from Theorem 1 to composition modulo h. We directly state and prove the generalized algorithms. For more explanations we refer to the original papers [1, 3] and [11, Section 3.4.3].

3.1 Extension of Bernstein's algorithm

If $\mathbb{K}$ has positive characteristic $p > 0$, then Bernstein's algorithm [1] exploits the Frobenius map in order to recursively reduce the degrees of the polynomials to be composed by a factor p.

Algorithm 1

Input. $\hbar, h \in \mathbb{K}[x]$, $f, g \in \mathbb{K}[x]_{<n}$, where $n = \deg h$.

Output. $f \circ g \operatorname{rem} h$.

Assumptions. $\operatorname{char} \mathbb{K} = p$, $h = \hbar^m$, $\deg \hbar = d = n/m$.

1. If $m = 1$, then return $f \circ g \operatorname{rem} h$, computed by a general modular composition algorithm of cost $\mathsf{C}_{\mathbb{K}}(d)$.
2. Split f into polynomials $\tilde{f}_0, ..., \tilde{f}_{p-1}$ such that $f(x) = \sum_{i=0}^{p-1} \tilde{f}_i(x^p)\, x^i$. Let $\tilde{m} = \lceil m/p \rceil$ and $\tilde{g}, \tilde{h}$ be such that $\tilde{h}(x^p) = \hbar(x)^p$ and $\tilde{g}(x^p) = g(x)^p \operatorname{rem} \tilde{h}(x^p)^{\tilde{m}}$.
3. Recursively compute $\tilde{u}_i = \tilde{f}_i \circ \tilde{g} \operatorname{rem} \tilde{h}^{\tilde{m}}$ for all $0 \leqslant i \leqslant p-1$.
4. Return $\sum_{i=0}^{p-1} \tilde{u}_i(x^p)\, g(x)^i \operatorname{rem} h(x)$.

PROPOSITION 5. *Algorithm 1 is correct and takes* $p\,m\,\mathsf{C}_{\mathbb{K}}(d) + O\Big(\frac{p}{\log p}\mathsf{M}_{\mathbb{K}}(n)\log m\Big)$ *operations in* $\mathbb{K}$.

PROOF. Since $(d\,m-1)/p < d\,\tilde{m}$, the degrees of $\tilde{g}$ and the $\tilde{f}_i$ are at most $d\,\tilde{m}-1$. Now $\tilde{u}_i = \tilde{f}_i \circ \tilde{g} \operatorname{rem} \tilde{h}^{\tilde{m}}$ implies

$$\tilde{u}_i(x^p) = \tilde{f}_i(\tilde{g}(x^p)) \operatorname{rem} \tilde{h}(x^p)^{\tilde{m}} = \tilde{f}_i(g(x)^p) \operatorname{rem} \hbar(x)^{p\tilde{m}},$$

which ensures the correctness of the algorithm.

In step 2 we need: $O(n \log p)$ operations in $\mathbb{K}$ to compute all the necessary p-th powers, $O(\mathsf{M}_{\mathbb{K}}(d\ \tilde{m}))$ for $\tilde{h}^{\tilde{m}}$, and $O(\mathsf{M}_{\mathbb{K}}(n))$ for the division to obtain $\tilde{g}$. In step 4, by [8, Exercise 9.16], the computation of $\tilde{u}_i(x^p) \operatorname{rem} h(x)$ for $i = 0, ..., p-1$ may be done in time

$$O\Big(p\,d\,\tilde{m} + \Big\lceil\frac{p\,\tilde{m}}{m}\Big\rceil \mathsf{M}_{\mathbb{K}}(n)\Big) = O\Big(d\,(m+p) + \Big(1+\frac{p}{m}\Big)\mathsf{M}_{\mathbb{K}}(n)\Big) = O(p\,\mathsf{M}_{\mathbb{K}}(n)).$$

The remaining calculations of step 4 amount to $O(p\,\mathsf{M}_{\mathbb{K}}(n))$ operations in $\mathbb{K}$.

Let $\mathsf{T}(m)$ be the time complexity for precision m and let $e \in \mathbb{N}$ be such that $p^{e-1} < m \leqslant p^e$. By what precedes, we have

$$\mathsf{T}(m) \leqslant p\,\mathsf{T}(\lceil m/p \rceil) + c\,p\,\mathsf{M}_{\mathbb{K}}(d\,m),$$

for all $m > 1$ and a sufficiently large constant c. Unrolling this inequality $e - 1 = O(\log_p m)$ times, we obtain

$$\mathsf{T}(m) \leqslant p^e\,\mathsf{T}(\lceil m/p^e \rceil) + c\,p \sum_{i=0}^{e-1} p^i\,\mathsf{M}_{\mathbb{K}}(d\,\lceil m/p^i \rceil).$$

Using $p^i\,\lceil m/p^i \rceil \leqslant m + p^i - 1$ and the super-additivity of $\mathsf{M}_{\mathbb{K}}$, we conclude that

$$\mathsf{T}(m) = p\,m\,\mathsf{C}_{\mathbb{K}}(d) + O(p\,\mathsf{M}_{\mathbb{K}}(n)\log_p m). \qquad \square$$

3.2 Extension of Brent and Kung's algorithm

We now extend Brent and Kung's series composition algorithm to composition modulo $h = \hbar^m$. This method uses Taylor series expansion in order to reduce the degrees of the polynomials to be composed. For a given polynomial u, we write $u^{(l)}$ for its l-th derivative. Algorithm 2 is a sub-algorithm that is used when the arguments of the composition both have small degree.

Algorithm 2

Input. $u, g, h \in \mathbb{K}[x]$, where u has degree t.

Output. $u \circ g \operatorname{rem} h$.

1. If $t \leqslant 0$ then return $u(0)$.
2. Split $u(x) = u_0(x) + x^{\lceil t/2 \rceil}\, u_1(x)$ with $\deg u_0 < \lceil t/2 \rceil$ and $\deg u_1 = t - \lceil t/2 \rceil$.
3. Recursively compute $v_i = u_i \circ g \operatorname{rem} h$ for $i \in \{0, 1\}$,
4. Return $(v_1 + g^{\lceil t/2 \rceil}\, v_2) \operatorname{rem} h$.

Algorithm 3

Input. $\hbar, h \in \mathbb{K}[x]$, $f, g \in \mathbb{K}[x]_{<n}$, where $n = \deg h$.

Output. $f \circ g \operatorname{rem} h$.

Assumption. $\mathbb{K}$ has characteristic $p \geqslant k$, where $l = \big\lceil \sqrt{m/(d \log n)} \big\rceil$ and $k = \lceil m/l \rceil$; $h = \hbar^m$, $\hbar$ is separable and irreducible, $\deg \hbar = d = n/m$.

1. Compute $g_1 = g \operatorname{rem} \hbar^l$ and $g_2 = g - g_1$.
2. If $g_1 \neq 0$ then compute the valuation v of g_1' in $\hbar$ and the modular inverse $u = (g_1'/\hbar^v)^{-1} \bmod \hbar^{m-l}$.
3. Compute $r_0 = f \circ g_1 \operatorname{rem} h$ using Algorithm 2.
4. If $g_1 = 0$ then compute $r_i = f^{(i)}(0)$ for all i from 1 to $k-1$. Otherwise, for i from 0 to $k-2$, compute $r_{i+1} = u\,(r_i' \operatorname{quo} \hbar^v) \operatorname{rem} \hbar^{m-(i+1)l}$.
5. Return $\sum_{i=0}^{k-1} \frac{r_i}{i!}\, g_2^i \operatorname{rem} \hbar^m$.

PROPOSITION 6. *Algorithm 3 is correct and takes* $O(\sqrt{n}\,\mathsf{M}_{\mathbb{K}}(n)\log^{1/2} n)$ *operations in* $\mathbb{K}$ *if* $d = O(m/\log n)$.

PROOF. The main ingredient of the proof is the following Taylor expansion:

$$f(g_1(x) + g_2(x)) = \sum_{i=0}^{k-1} \frac{f^{(i)}(g_1(x))}{i!}\, g_2^i \mod \hbar^m,$$

which makes use of $\operatorname{val}_{\hbar}\, g_2 \geqslant l$ and $k\,l \geqslant m$. If $g_1 = 0$ then $r_i = f^{(i)}(g_1(x))$ holds for all i, so the algorithm is correct. Otherwise we may write $g_1 = a(x)\ \hbar^w \bmod \hbar^{w+1}$ with $0 \leqslant w < l$ and $a \neq 0$ of degree $< d$, which implies $g_1'(x) = w\ a(x)\ \hbar'(x)\ \hbar^{w-1} \bmod \hbar^w$. Since $p > w$ and $\hbar$ separable, it follows that $0 \leqslant v \leqslant l-2$, whence u is well defined.

Let us show that $r_i = f^{(i)}(g_1(x)) \bmod \hbar^{m-il}$, by induction on i. This is clear for $i = 0$. Assume that the identity holds for some $i \geqslant 0$. From $f^{(i+1)} \circ g_1 = (f^{(i)} \circ g_1)' / g_1'$ we see that the valuation of $(f^{(i)} \circ g_1)'$ in $\hbar$ is necessarily $\geqslant v$ and that $u\ (r_i' \operatorname{quo} \hbar^v) = (f^{(i)} \circ g_1)' \bmod \hbar^{m-il-v-1}$. The induction hypothesis is thus satisfied for $i+1$ because $i\,l + v + 1 \leqslant (i+1)\,l$. We are done with the correctness.

The $\hbar$-adic expansion of g takes $O(\mathsf{M}_{\mathbb{K}}(n)\log n)$ operations and dominates the cost of steps 1 and 2. Steps 4 and 5 amount to $O\big(\frac{m}{l}\,\mathsf{M}_{\mathbb{K}}(n)\big) = O(\sqrt{n}\,\mathsf{M}_{\mathbb{K}}(n)\log^{1/2} n)$ operations. As to step 3, we enter Algorithm 2 with $u = f$.

Let $\mathsf{T}(t)$ be the cost function of Algorithm 2. The degrees of $u_1 \circ g_1$, $u_2 \circ g_1$ and $g_1^{\lceil t/2\rceil}$ are $\leqslant d\,l\,\lceil t/2\rceil$, so that

$$\mathsf{T}(t) \leqslant 2\,\mathsf{T}\Big(\Big\lceil \frac{t}{2}\Big\rceil\Big) + c\,\mathsf{M}_{\mathbb{K}}\Big(\min\Big(d\,l\Big\lceil\frac{t}{2}\Big\rceil, n\Big)\Big),$$

for some constant c. Let e be the largest integer such that $d\,l\,\lceil n/2^e\rceil > n$. Unrolling $e-1$ times the inequality for $\mathsf{T}(t)$ and $t = n$, we obtain

$$\begin{aligned}\mathsf{T}(n) &\leqslant 2^e\,\mathsf{T}(\lceil n/2^e\rceil) + c\sum_{i=0}^{e-1} 2^i\,\mathsf{M}_{\mathbb{K}}(n)\\ &= 2^e\,\mathsf{T}(\lceil n/2^e\rceil) + O(2^e\,\mathsf{M}_{\mathbb{K}}(n)).\end{aligned}$$

In a similar way we obtain $\mathsf{T}(\lceil n/2^e\rceil) = O(\mathsf{M}_{\mathbb{K}}(n)\log n)$. Consequently, $\mathsf{T}(n) = O(2^e\,\mathsf{M}_{\mathbb{K}}(n)\log n)$. The conclusion follows from $2^e = O(d\,l) = O\big(\sqrt{n/\log n}\big)$. □

Corollary 7. *Let $\hbar$ be of degree d in $\mathbb{K}[x]$, let $h = \hbar^m$ be of degree $n = m\,d$, and let f, g be in $\mathbb{K}[x]_{<n}$. If $d = O(m/\log n)$, then the modular composition $f \circ g \bmod h$ may be computed using $m\sqrt{n}\,\mathsf{C}_{\mathbb{K}}(d) + O(\sqrt{n}\,\mathsf{M}_{\mathbb{K}}(n)\log n)$ operations in $\mathbb{K}$.*

Proof. If $p = O(\sqrt{n})$, then we may use Algorithm 1, which requires $m\sqrt{n}\,\mathsf{C}_{\mathbb{K}}(d) + O(\sqrt{n}\,\mathsf{M}_{\mathbb{K}}(n)\log n)$ operations in $\mathbb{K}$. Otherwise, we use Algorithm 3. □

Notice that for fixed d, this corollary gives a cost $n^{1.5+o(1)}$, independently of the characteristic.

4. REDUCTION TO SERIES COMPOSITION

Recall that the modulus h is the m-th power of a polynomial $\hbar$ of degree d. From now on, we assume that $\hbar$ is separable. So far, our algorithms for composition modulo h relied on extensions of known algorithms dedicated to power series. In this section, we investigate the other natural approach: we design algorithms that reduce one composition modulo $\hbar^m$ to one series composition of order m over a suitable extension of the ground field. We set $\mathbb{L} = \mathbb{K}[z]/(\hbar(z))$ and write α for the residue class of z in $\mathbb{L}$. The associated trace function is denoted by $\mathrm{Tr}_{\mathbb{L}/\mathbb{K}}$. This function naturally extends to a map $\mathrm{Tr}_{\mathbb{L}/\mathbb{K}}\colon \mathbb{L}[x] \to \mathbb{K}[x]$ in a coefficientwise fashion.

4.1 Case of small moduli

The first method we study is rather elementary. It works well for any characteristic, but it is only efficient when d is much smaller than n. We compute $(f\circ g)(y) \bmod (y-\zeta)^m$ for all roots ζ of $\hbar$ and recover $f \circ g \bmod \hbar^m$ by Chinese remaindering. The simultaneous computation with all roots of $\hbar$ is emulated *via* standard algebraic techniques.

Since $\hbar$ is not assumed to be irreducible, $\mathbb{L}$ is not necessarily a field. We will compute inverses in $\mathbb{L}$ using the technique of *dynamic evaluation*, which is briefly recalled in Appendix A.2. For convenience of the reader, we use the auxiliary variable y for power series and keep the variable x for polynomials.

Algorithm 4

Input. $\hbar, h \in \mathbb{K}[x]$, $f, g \in \mathbb{K}[x]_{<n}$, where $n = \deg h$.
Output. $f \circ g \bmod h$.
Assumptions. $h = \hbar^m$, $\hbar$ is separable, $\deg\hbar = d = n/m$.

1. Compute $G(y) = g(y+\alpha) - g(\alpha)$ in $\mathbb{L}[y]$.
2. Compute $F(y) = f(y + g(\alpha))$ in $\mathbb{L}[y]$.
3. Perform the power series composition $R(y) = (F \circ G)(y)$ in $\mathbb{L}[y]/(y^m)$.
4. Compute $H(x) = h(x)/(x-\alpha)^m$ in $\mathbb{L}[x]$.
5. With the algorithm underlying Corollary 20, compute the modular inverse $I(x) = H^{-1}(x) \bmod (x-\alpha)^m$.
6. Compute $Q(x) = H(x)\,I(x)$ and then $P(x) = R(x-\alpha)\,Q(x)$.
7. Return $\mathrm{Tr}_{\mathbb{L}/\mathbb{K}}(P(x)) \bmod h(x)$.

Proposition 8. *Algorithm 4 is correct and takes*

$$O(\mathsf{S}_{\mathbb{L}/\mathbb{K}}(m) + \mathsf{M}_{\mathbb{K}}(d)\,\mathsf{M}_{\mathbb{K}}(n)\log d\log n)$$

operations in $\mathbb{K}$.

Proof. For a root ζ of $\hbar$ in some algebraic closure $\bar{\mathbb{K}}$ of $\mathbb{K}$, we write σ_ζ for the homomorphism from $\mathbb{L}$ to $\bar{\mathbb{K}}$ that sends α to ζ. This homomorphism extends coefficientwise into a map $\sigma_\zeta\colon \mathbb{L}[x] \to \bar{\mathbb{K}}[x]$. One verifies that $\mathrm{Tr}_{\mathbb{L}/\mathbb{K}}(P(x))$ equals

$$\begin{aligned}&\sum_{\hbar(\zeta)=0} \sigma_\zeta(R(x-\alpha))\,\sigma_\zeta(Q(x))\\ &= \sum_{\hbar(\zeta)=0} \Bigg\{((f\circ g)(x) \bmod (x-\zeta)^m)\,\frac{h(x)}{(x-\zeta)^m}\\ &\qquad\qquad \times\bigg(\Big(\frac{h(x)}{(x-\zeta)^m}\Big)^{-1} \bmod (x-\zeta)^m\bigg)\Bigg\},\end{aligned}$$

so $\mathrm{Tr}_{\mathbb{L}/\mathbb{K}}(P(x)) \bmod (x-\zeta)^m$ equals $(f\circ g)(x) \bmod (x-\zeta)^m$ for all roots ζ of $\hbar$. This proves the correctness of the algorithm thanks to the Chinese remainder theorem.

For the complexity analysis, step 3 amounts to $\mathsf{S}_{\mathbb{L}/\mathbb{K}}(m)$ operations in $\mathbb{K}$. In step 5, we use Corollary 20 to compute $I(x)$ with $O(\mathsf{M}_{\mathbb{K}}(d)\,\mathsf{M}_{\mathbb{K}}(n)\log d\log n)$ operations in $\mathbb{K}$. Steps 1, 2, 4, and 6 require $O(\mathsf{M}_{\mathbb{L}}(n)\log n)$ additional ring operations in $\mathbb{L}$. In step 7, the row matrix of $\mathrm{Tr}_{\mathbb{L}/\mathbb{K}}$ may be computed with $O(\mathsf{M}_{\mathbb{K}}(d))$ operations in $\mathbb{K}$ by formula (20), so each trace takes $O(d)$ operations in $\mathbb{K}$. Therefore step 7 amounts to $O(\mathsf{M}_{\mathbb{K}}(d) + \mathsf{M}_{\mathbb{K}}(n) + d\,n)$ operations in $\mathbb{K}$, which is negligible. □

When d is much smaller than m, we may directly benefit from fast power series composition. But overall this does not improve much on Corollary 7. In the rest of this section, we aim at decreasing the dependency in d in complexity bounds.

4.2 Fast reduction to series composition

In order to improve on Algorithm 4, we shall use a better way to expand $g(y+\alpha)$ and $f(y+g(\alpha))$ modulo y^m. For this purpose we introduce the $\mathbb{K}$-algebra homomorphism

$$\begin{aligned}\pi_{\hbar,m}\colon\ \mathbb{K}[x]/(\hbar(x)^m) &\to \mathbb{L}[y]/(y^m)\\ u(x) &\mapsto u(y+\alpha).\end{aligned}$$

We regard $\pi_{\hbar,m}$ and $\pi_{\hbar,m}^{-1}$ as conversions that we respectively call the *untangling* and *tangling* mappings.

LEMMA 9. *For all separable polynomial $\hbar$ and all $m \geqslant 1$, the map $\pi_{\hbar,m}$ is a $\mathbb{K}$-algebra isomorphism.*

PROOF. $\pi_{\hbar,m}$ is clearly a homomorphism between $\mathbb{K}$-algebras of equal dimensions, so we need to prove its injectivity when $\hbar$ is separable. We consider the $\hbar$-adic expansion $u_0(x) + u_1(x)\,\hbar(x) + \cdots + u_{m-1}(x)\,\hbar(x)^{m-1}$ of $u(x)$ modulo $\hbar(x)^m$, where all the u_i have degrees $<\deg \hbar$, and we assume that $\pi_{\hbar,m}(u) = 0$. It is immediate that $u_0 = 0$. We may thus suppose, by induction on $i \geqslant 1$, that $u_j = 0$ for all $0 \leqslant j \leqslant i-1$. From $\hbar(y+\alpha) = \hbar'(\alpha)\,y + O(y^2)$ we obtain $u(y+\alpha) = u_i(\alpha)\,\hbar'(\alpha)^i\,y^i + O(y^{i+1})$. Since $\hbar'(\alpha)$ is invertible, it follows that $u_i(\alpha) = 0$, whence $u_i = 0$. This completes the induction. □

Remark. The separability assumption on $\hbar$ is necessary: if $\hbar(x) = x^2$ and $m = 2$, then we have $\pi_{\hbar,m}(x^3) = 0$.

Algorithm 5

Input. $\hbar, h \in \mathbb{K}[x]$, $f, g \in \mathbb{K}[x]_{<n}$, where $n = \deg h$.

Output. $f \circ g \operatorname{rem} h$.

Assumptions. $h = \hbar^m$, $\hbar$ is separable, $\deg \hbar = d = n/m$.

1. Compute $G = \pi_{\hbar,m}(g) - g(\alpha)$.
2. Let $\bar{g} = g \operatorname{rem} \hbar$ and compute the characteristic polynomial $\bar{\chi}$ of $\bar{g}$ modulo $\hbar$.
3. Compute $\bar{F} = \pi_{\bar{\chi},m}(f)$.
4. Write $\bar{F}(y) = \bar{F}_0(\beta) + \bar{F}_1(\beta)\,y + \cdots + \bar{F}_{m-1}(\beta)\,y^{m-1}$, where β is the class of z in $\mathbb{K}[z]/(\bar{\chi}(z))$, and the $\bar{F}_i$ have degrees $<d$. For each $i \in \{0, ..., m-1\}$, compute $F_i = \bar{F}_i \circ \bar{g} \operatorname{rem} \hbar$ and let $F(y) = F_0(\alpha) + F_1(\alpha)\,y + \cdots + F_{m-1}(\alpha)\,y^{m-1} \in \mathbb{L}[y]$.
5. Perform the power series composition $R(y) = (F \circ G)(y)$ in $\mathbb{L}[y]/(y^m)$.
6. Return $\pi_{\hbar,m}^{-1}(R)$.

From now on, in cost analyses, the expression $\Pi_{\mathbb{K}}(d, m)$ (resp. $\Pi^*_{\mathbb{K}}(d, m)$) represents a cost function for computing the *direct image* (resp. the *preimage*) of $\pi_{\hbar,m}$. If good algorithms are known for $\pi_{\hbar,m}$ and $\pi_{\hbar,m}^{-1}$, for characteristic polynomials and for compositions modulo $\hbar$, then we observe that Algorithm 5 boils down to one power series composition at precision m over $\mathbb{L}$.

PROPOSITION 10. *Algorithm 5 is correct and it requires*

$$\begin{aligned}\mathsf{H}_{\mathbb{K}}(d\,m) \;=\; & 2\,\Pi_{\mathbb{K}}(d, m) + \Pi^*_{\mathbb{K}}(d, m) + \mathsf{Q}_{\mathbb{K}}(d) + m\,\mathsf{C}_{\mathbb{K}}(d) + \\ & \mathsf{S}_{\mathbb{L}/\mathbb{K}}(m) + O(d\,m)\end{aligned}$$

operations in $\mathbb{K}$.

PROOF. The homomorphism $\mathbb{K}[z]\,/\,(\bar{\chi}(z)) \to \mathbb{L}$ that sends β to $g(\alpha)$ is well defined, so applying it coefficientwise to $\bar{F}(y) = f(y+\beta) \in \mathbb{K}[z]/(\bar{\chi}(z))[y]$ we obtain $f(y + g(\alpha)) \bmod y^m = F(y)$. Therefore we have $(F \circ G)(y) = (f \circ g)(y + \alpha) \bmod y^m = \pi_{\hbar,m}(f \circ g \operatorname{rem} h)$, which ensures the correctness of the algorithm. The cost directly follows from the definitions. □

The rest of this section is devoted to algorithms with softly linear costs for computing $\pi_{\hbar,m}$ and its inverse. More precisely, we will prove that we may take $\Pi_{\mathbb{K}}(d,\ m)$ and $\Pi^*_{\mathbb{K}}(d,\ m)$ to be $O(\mathsf{M}_{\mathbb{K}}(d\ m)\ \log^2\ m\ +\ \mathsf{M}_{\mathbb{K}}(d)\ \log\ d)$ in the latter proposition. This leads to our main theorem:

THEOREM 11. *Let $\hbar$, $h = \hbar^m \in \mathbb{K}[x]$ and f, $g \in \mathbb{K}[x]_{<n}$, where $n = \deg h$, such that $\hbar$ is separable of degree $d = n/m$. Then, we can compute $f \circ g \operatorname{rem} h$ using*

$$\begin{aligned}\mathsf{H}_{\mathbb{K}}(d, m) \;=\; & \mathsf{S}_{\mathbb{L}/\mathbb{K}}(m) + m\,\mathsf{C}_{\mathbb{K}}(d) + \mathsf{Q}_{\mathbb{K}}(d) + \\ & O(\mathsf{M}_{\mathbb{K}}(d\,m)\log^2 m + \mathsf{M}_{\mathbb{K}}(d)\log d)\end{aligned}$$

operations in $\mathbb{K}$.

4.3 Untangling in characteristic zero

The following lemma is elementary, but we shall refer to it several times.

LEMMA 12. *Given integers $0 \leqslant k < l$ and a polynomial $v \in \mathbb{K}[x]$, we have $(v\,\hbar^l)^{(k)}(\alpha) = 0$.*

PROOF. We have $(v\ \hbar^l)^{(k)} = \sum_{i=0}^{k} \binom{k}{i}\ v^{(i)}\ (\hbar^l)^{(k-i)}$ by Leibnitz formula, showing that $(v\,\hbar^l)^{(k)}$ is divisible by $\hbar$. □

We begin with the easiest situation when the characteristic p of $\mathbb{K}$ is zero or sufficiently large and achieve softly linear time for $\Pi(d, m)$.

Algorithm 6

Input. $\hbar, h \in \mathbb{K}[x]$, $u \in \mathbb{K}[x]_{<n}$.

Output. $u(\alpha), u'(\alpha), ..., u^{(m-1)}(\alpha)$.

Assumptions. $h = \hbar^m$, $\hbar$ is separable, $\deg \hbar = d = n/m$.

1. If $m = 1$ then return $u(\alpha)$, otherwise let $l = \lfloor m/2 \rfloor$.
2. Recursively apply the algorithm to u rem $\hbar^l$, giving $u(\alpha), u'(\alpha), ..., u^{(l-1)}(\alpha)$.
3. Recursively apply the algorithm to $u^{(l)}$ rem $\hbar^{m-l}$, giving $u^{(l)}(\alpha), u^{(l+1)}(\alpha), ..., u^{(m-1)}(\alpha)$.
4. Return $u(\alpha), u'(\alpha), ..., u^{(m-1)}(\alpha)$.

PROPOSITION 13. *Algorithm 6 is correct and takes $O(\mathsf{M}_{\mathbb{K}}(n)\ \log\ m)$ operations in $\mathbb{K}$. If $p = 0$ or $p \geqslant m$, then, given $u \in \mathbb{K}[x]_{<n}$, one may compute $\pi_{\hbar,m}(u)$ using $O(\mathsf{M}_{\mathbb{K}}(n)\log m)$ operations in $\mathbb{K}$.*

PROOF. The correctness follows from Lemma 12 which ensures that $u^{(k)}(\alpha) = (u \text{ rem } \hbar^l)^{(k)}(\alpha)$ for all $0 \leqslant k < l$ and $u^{(l+k)}(\alpha) = (u^{(l)} \text{ rem } \hbar^{m-l})^{(k)}(\alpha)$ for all $0 \leqslant k < m-l$. The cost analysis is straightforward. Finally, to deduce $\pi_{\hbar,m}(u)$, thanks to the assumption on the characteristic, we may simply use the Taylor expansion $u(y+\alpha) = \sum_{i=0}^{i=m-1} u^{(i)}(\alpha)\,\frac{y^i}{i!} + O(y^m)$. □

4.4 Untangling in positive characteristic

In order to handle the case $p < m$, we need to pay attention to vanishing coefficients when computing Taylor expansions using higher order derivatives. For this, we rely on the following adaptation of the Taylor expansion of u:

$$u(y+\alpha) = \sum_{i \geqslant 0} D^{(i)}(u)(\alpha)\,y^i.$$

The polynomial $D^{(l)}(u)$ is called the l-th order *Hasse derivative* of u. The operator $D^{(l)}$ is linear and we may compute it as follows: if $u = x^i$ is a monomial and $l > i$, then $D^{(l)}(x^i) = 0$; otherwise

$$D^{(l)}(x^i) = \binom{i}{l} x^{i-l}.$$

LEMMA 14. *For all integers k and l, we have $D^{(k)} \circ D^{(l)} = \binom{k+l}{k} D^{(k+l)}$.*

PROOF. Let $i \geqslant k+l$. A straightforward calculation in characteristic zero yields

$$\begin{aligned} D^{(k)} \circ D^{(l)}(x^i) &= \binom{i-l}{k}\binom{i}{l} x^{i-l-k} \\ &= \binom{i}{k+l}^{-1}\binom{i-l}{k}\binom{i}{l} D^{(k+l)}(x^i) \\ &= \binom{k+l}{k} D^{(k+l)}(x^i). \end{aligned}$$

Now if the characteristic $p > 0$ divides the integer $\binom{i}{k+l}$, then it also divides $\binom{i-l}{k}\binom{i}{l}$, and the lemma remains correct. □

It is convenient to also introduce the *Pochhammer symbol*

$$(i)_j = i\,(i-1)\cdots(i-j+1).$$

The next lemma expresses conditions to compute Hasse derivatives from ones of lower orders, in positive characteristic.

LEMMA 15. *Let $l \geqslant p$ be an integer power of $p > 0$, let s be an integer such that $\mathrm{val}_p\, s > \mathrm{val}_p\, l$, and let $a < b$ be two integers in $\{0, ..., p-1\}$. Then, for any polynomial $u \in \mathbb{K}[x]$, we may compute $D^{(s+bl)}(u)$ from $D^{(s+al)}(u) = \sum_{i \geqslant 0} v_i\, x^i$ using the formula*

$$D^{(s+bl)}(u) = \sum_{c \geqslant b-a} \frac{(c)_{b-a}}{(b)_{b-a}} \sum_{cl \leqslant i < (c+1)l} v_i\, x^{i-(b-a)l}. \qquad (1)$$

PROOF. The previous lemma implies that $D^{(s+bl)} = \binom{s+bl}{(b-a)l}^{-1} D^{((b-a)l)} \circ D^{(s+al)}$. Now in characteristic zero, we have

$$\begin{aligned} &\binom{s+bl}{(b-a)l}^{-1} D^{((b-a)l)}(x^i) \\ &= \binom{s+bl}{(b-a)l}^{-1}\binom{i}{(b-a)l} x^{i-(b-a)l} \\ &= \frac{(i)_{(b-a)l}}{(s+bl)_{(b-a)l}}\, x^{i-(b-a)l} \end{aligned}$$

for all $i \geqslant (b-a)\,l$. In positive characteristic, we claim that the ratios $\frac{(i)_{(b-a)l}}{(s+bl)_{(b-a)l}}$ have non-negative valuations in p.

Let us first consider the case when $b = a+1$. If $i = c\,l$, with $c \geqslant 1$, then $\mathrm{val}_p\, i \geqslant \mathrm{val}_p\, l = \mathrm{val}_p(s+bl)$. If $0 < j < l$, then $\mathrm{val}_p(i-j) = \mathrm{val}_p\, j = \mathrm{val}_p(s+b\,l-j)$. This shows that $(i)_l/(s+bl)_l = c/b \bmod p$ is well defined and may be computed in $\mathbb{K}$ whenever i is a multiple of l.

At a second stage, we make the induction hypothesis that $(i)_l/(s+bl)_l$ has been computed in $\mathbb{K}$ for $i \geqslant cl$ such that $i+1 < (c+1)\,l$. The value for $i+1$ may then be obtained *via*

$$\frac{(i+1)_l}{(s+bl)_l} = \frac{i+1}{i+1-l}\,\frac{(i)_l}{(s+bl)_l} = \frac{c}{b} \bmod p,$$

since $\mathrm{val}_p(i+1) \geqslant \mathrm{val}_p(i+1-l)$. This deals with the case when $b = a+1$.

Finally, for any $b > a$, the ratio $(i)_{(b-a)l}/(s+bl)_{(b-a)l}$ is also well defined since it rewrites into

$$\begin{aligned} &\frac{(i)_{(b-a)l}}{(s+bl)_{(b-a)l}} \\ &= \frac{(i)_l}{(s+bl)_l}\,\frac{(i-l)_l}{(s+(b-1)\,l)_l}\cdots\frac{(i-(b-a-1)\,l)_l}{(s+(a+1)\,l)_l} \\ &= \frac{c\,(c-1)\cdots(c-(b-a-1))}{b\,(b-1)\cdots(a+1)} \bmod p, \end{aligned}$$

which concludes the proof. □

We are now in a position to adapt the "divide and conquer" Algorithm 6 to the case when $\mathbb{K}$ has small positive characteristic p.

Algorithm 7

Input. $\hbar \in \mathbb{K}[x]$ of degree d; integers s, l, a, b; $D^{(s+al)}(u) \operatorname{rem} \hbar^{(b-a)l}$.

Output. $D^{(s+kl)}(u) \operatorname{rem} \hbar^l$, for all $k \in \{a, ..., b-1\}$.

Assumption. $\mathbb{K}$ has positive characteristic p; l is a power of p, $1 \leqslant \mathrm{val}_p\, l < \mathrm{val}_p\, s$; $0 \leqslant a < b \leqslant p$; $u \in \mathbb{K}[x]_{<n}$.

1. If $b = a+1$, then return $D^{(s+al)}(u) \operatorname{rem} \hbar^l$.
2. Let $c = \lceil (a+b)/2 \rceil$.
3. Compute $D^{(s+al)}(u) \operatorname{rem} \hbar^{(c-a)l}$.
4. By using formula (1), compute $D^{(s+cl)}(u) \operatorname{rem} \hbar^{(b-c)l}$ from $D^{(s+al)}(u) \operatorname{rem} \hbar^{(c-a)l}$.
5. Recursively apply the algorithm to $u, \hbar, s, l, a, c$, and $D^{(s+al)}(u) \operatorname{rem} \hbar^{(c-a)l}$. This yields $D^{(s+kl)}(u) \operatorname{rem} \hbar^l$ for all $k \in \{a, ..., c-1\}$.
6. Recursively apply the algorithm to $u, \hbar, s, l, c, b$, and $D^{(s+cl)}(u) \operatorname{rem} \hbar^{(b-c)l}$. This yields $D^{(s+kl)}(u) \operatorname{rem} \hbar^l$, for all $k \in \{c, ..., b-1\}$.
7. Return $D^{(s+kl)}(u) \operatorname{rem} \hbar^l$, for all $k \in \{a, ..., b-1\}$.

PROPOSITION 16. *Algorithm 7 is correct and takes $O(\mathsf{M}_{\mathbb{K}}((b-a)\,d\,l)\log(b-a))$ operations in $\mathbb{K}$.*

PROOF. The correctness follows from Lemmas 12 and 15; the cost analysis is straightforward. □

Algorithm 8

Input. $\hbar, h \in \mathbb{K}[x]$ with $d = \deg \hbar$, $\deg h = n = d\,m$; an integer s; $D^{(s)}(u) \operatorname{rem} \hbar^m$.

Output. $D^{(s)}(u)(\alpha), ..., D^{(s+m-1)}(u)(\alpha)$.

Assumptions. $\mathbb{K}$ has positive characteristic p; $h = \hbar^m$; $p^{\mathrm{val}_p\, s} \geqslant m$; $u \in \mathbb{K}[x]_{<n}$.

1. If $m \leqslant p$, then use Algorithm 6 to compute $\pi_{\hbar,m}(D^{(s)}(u)) = \sum_{i=0}^{m-1} (D^{(i)} \circ D^{(s)})(u)(\alpha)\; y^i + O(y^m)$ and return $D^{(s)}(u)(\alpha), ..., (D^{(m-1)} \circ D^{(s)})(u)(\alpha)$.
2. Let e be minimal with $p^e \geqslant m$, let $l = p^{e-1}$, and $k = \lfloor m/l \rfloor$.
3. Compute $D^{(s+il)}(u) \operatorname{rem} \hbar^l$ for all $i \in \{0, ..., k-1\}$ using Algorithm 7 (called with s, l, $a = 0$ and $b = k$).
4. If $k\,l < m$, then compute $D^{(s+kl)}(u) \operatorname{rem} \hbar^{m-kl}$ from $D^{(s)}(u) \operatorname{rem} \hbar^m$ with formula (1).
5. For all $i \in \{0, ..., k-1\}$, use the algorithm recursively with $D^{(s+il)}(u) \operatorname{rem} \hbar^l$ in order to obtain $D^{(s+il)}(u)(\alpha), ..., D^{(s+(i+1)l-1)}(u)(\alpha)$.
6. If $k\ l < m$ then use the algorithm recursively with $D^{(s+kl)}(u) \operatorname{rem} \hbar^{m-kl}$ in order to obtain $D^{(s+kl)}(u)(\alpha), ..., D^{(s+m-1)}(u)(\alpha)$.
7. Return $D^{(s)}(u)(\alpha), ..., D^{(s+m-1)}(u)(\alpha)$.

PROPOSITION 17. *Algorithm 8 is correct and takes*

$$\Pi_{\mathbb{K}}(d, m) = O(\mathsf{M}_{\mathbb{K}}(d\,m)\log m)$$

operations in $\mathbb{K}$.

PROOF. Let us first examine the case when $m \leqslant p$. We need to show that $(D^{(j)} \circ D^{(s)})(u)(\alpha)$ coincides with $D^{(s+j)}(u)(\alpha)$ for all $j \in \{0, ..., m-1\}$. From Lemma 14, we have $D^{(s+j)} = \binom{s+j}{j}^{-1} D^{(j)} \circ D^{(s)}$. Since $\mathrm{val}_p\ s \geqslant 1$, it follows that $\binom{s+j}{j} = 1 \bmod p$.

Now we suppose that $m > p$. We have $e \geqslant 2$, so l is a power of p, we have $k \leqslant p$, and the assumption $p^{\mathrm{val}_p s} \geqslant m$ is preserved through recursive calls. The conditions of Algorithm 7 are satisfied, whence the correctness of step 3 by Proposition 16. Step 4 is a consequence of Lemmas 12 and 15. We are done with the correctness.

Step 1 costs $O(\mathsf{M}_{\mathbb{K}}(d\,m)\log m)$ by Proposition 13. Step 3 costs $O(\mathsf{M}_{\mathbb{K}}(k\ d\ l)\ \log\ p) = O(\mathsf{M}_{\mathbb{K}}(d\ m)\ \log\ p)$ by Proposition 16 and step 4 takes $O(\mathsf{M}_{\mathbb{K}}(d\,m))$ operations in $\mathbb{K}$. The total cost function $\mathsf{T}(m)$ of the algorithm satisfies

$$\mathsf{T}(m) \leqslant k\,\mathsf{T}(l) + \mathsf{T}(m - k\,l) + c\,\mathsf{M}_{\mathbb{K}}(d\,m)\log p,$$

where c is a sufficiently large constant and $\mathsf{T}(m) \leqslant c\ \mathsf{M}_{\mathbb{K}}(d\ m)\ \log\ m$ when $m \leqslant p$. This leads to $\mathsf{T}(m) = O(\mathsf{M}_{\mathbb{K}}(d\,m)\log m + \mathsf{M}_{\mathbb{K}}(d\,m)\log p \log_p m)$. □

4.5 Tangling

The next algorithm is independent of the characteristic. It relies on the untangling algorithms from the previous subsections, by using the "divide and conquer" strategy. For a polynomial or series f in x and $a \leqslant b$, we denote $f_{a;b} = \sum_{a \leqslant i < b} f_i\, x^{i-a}$ and $f_{;b} = f_{0;b}$. For convenience, we assume that $\Pi_{\mathbb{K}}(d, m)/m$ is a nondecreasing function of m.

Algorithm 9

Input. $\hbar \in \mathbb{K}[x]$ of degree d; an integer $m \in 2^{\mathbb{N}}$; $\sigma \in \mathbb{L}[y]\,/\,(y^m)$, where $\mathbb{L} = \mathbb{K}[z]\,/\,(\hbar(z))$; $(w(y)^{2^i})_{;2^i}$ for $1 \leqslant 2^i \leqslant m/2$ where $w(y) = (y/\hbar(y+\alpha))_{;m}$.

Output. $\pi^{-1}_{\hbar,m}(\sigma)$.

1. If $m = 1$, then let $u(z)$ be the preimage of $\sigma(0)$ in $\mathbb{K}[z]$ and return $u(x)$.
2. Let $l = m/2$ and compute $v_0(x) = \pi^{-1}_{\hbar,l}(\sigma_{;l})$, $v_1(x) = (\sigma - \pi_{\hbar,m}(v_0(x)))_{l;m}$.
3. Return $v_0(x) + \hbar(x)^l\, \pi^{-1}_{\hbar,m-l}((w(y)^l\, v_1(y))_{;m-l})$.

LEMMA 18. *Algorithm 9 is correct and takes time* $\Pi_{\mathbb{K}}(d,m)\ \log\ m + \mathsf{M}_{\mathbb{K}}(d\ m)\ \log\ m + \mathsf{M}_{\mathbb{K}}(d)\ \log\ d$. *In particular, for any* m, *we have*

$$\Pi^*_{\mathbb{K}}(d,m) = O(\mathsf{M}_{\mathbb{K}}(d\,m)\log^2 m + \mathsf{M}_{\mathbb{K}}(d)\log d).$$

PROOF. The computation is clear when $m = 1$. Otherwise we verify that

$$\begin{aligned}
& \pi_{\hbar,m}(v_0(x) + \hbar(x)^l\, \pi^{-1}_{\hbar,m-l}((w(y)^l\, v_1(y))_{;m-l})) \\
=\ & \pi_{\hbar,m}(v_0(x)) + \pi_{\hbar,m}(\hbar(x))^l\, (w(y)^l\, v_1(y))_{;m-l} + O(y^m) \\
=\ & \pi_{\hbar,m}(v_0(x)) + \hbar(y+\alpha)^l\, (w(y)^l\, v_1(y))_{;m-l} + O(y^m) \\
=\ & \pi_{\hbar,m}(v_0(x)) + y^l\, v_1(y) + O(y^m) \\
=\ & \sigma_{0;l}(y) + y^l\, \pi_{\hbar,m}(v_0(x))_{l;m} + y^l\, (\sigma - \pi_{\hbar,m}(v_0(x)))_{l;m} \\
=\ & \sigma(y).
\end{aligned}$$

Kronecker substitution [8, Chapter 8, Section 4] allows us to multiply two series in $\mathbb{L}[[y]]$ at precision $O(y^m)$ using $O(\mathsf{M}_{\mathbb{K}}(d\,m))$ operations in $\mathbb{K}$.

The computation of $w(y)$ requires one inversion in $\mathbb{L}$ that takes $O(\mathsf{M}_{\mathbb{K}}(d)\ \log\ d)$ operations in $\mathbb{K}$, together with $O(\mathsf{M}_{\mathbb{K}}(d\,m))$ operations for the series inversion. The other $(w(y)^{2^i})_{;2^i}$ amount to $O(\mathsf{M}_{\mathbb{K}}(d\ m))$ more operations in $\mathbb{K}$. The cost function $\mathsf{T}(m)$ satisfies

$$\mathsf{T}(m) \leqslant 2\,\mathsf{T}(m/2) + \Pi_{\mathbb{K}}(d,m) + c\,\mathsf{M}_{\mathbb{K}}(d\,m),$$

for some constant c, which leads to $\mathsf{T}(m) = O((\Pi_{\mathbb{K}}(d,m) + \mathsf{M}_{\mathbb{K}}(d\ m))\ \log\ m + \mathsf{M}_{\mathbb{K}}(d)\ \log\ d)$. The final bound of the lemma follows from Proposition 17. □

APPENDIX A

A.1 Trace and Newton–Girard identities

Let h be a monic polynomial in $\mathbb{A}[x]$ of degree n, let φ be a linear form on $\mathbb{A}[x]_{<n}$, and let $g \in \mathbb{A}[x]_{<n}$. We study the *modular power projection* problem, which corresponds to computing $\varphi(1), \varphi(g), ..., \varphi(g^{n-1} \operatorname{rem} h)$.

Let $\mathbb{B}$ represent $\mathbb{A}[x]\ /\ (h(x))$. If we take $\varphi(u) = \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(g\,u)$, where $\mathrm{Tr}_{\mathbb{B}/\mathbb{A}}$ denotes the usual trace function of $\mathbb{B}$ over $\mathbb{A}$, then the latter power projections write as

$$\mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(g), \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(g^2 \operatorname{rem} h), ..., \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(g^n \operatorname{rem} h).$$

It is well known that these traces are related to the characteristic polynomial of g modulo h by the *Newton–Girard identities*:

$$\begin{aligned}
-\frac{\tilde{\chi}'(x)}{\tilde{\chi}(x)} \ =\ & \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(g) + \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(g^2 \operatorname{rem} h)\, x + \cdots \\
& + \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(g^n \operatorname{rem} h)\, x^{n-1} + O(x^n),
\end{aligned}$$

where $\tilde{\chi}(x) = x^n\ \chi(1\,/\,x)$ is the reverse polynomial of χ. If there exist given inverses of 2, 3, ..., n in $\mathbb{A}$, then it is possible to integrate the latter differential equation in χ with $O(\mathsf{M}_{\mathbb{A}}(n))$ operations in $\mathbb{A}$, which directly leads to an algorithm to compute χ from the power sums of the roots of χ (see for instance [9, Section 2] for details, which also cover the case when $\mathbb{A}$ is a finite field).

First, let us explain how to compute φ efficiently. Recall that $\tilde{h}(x) = x^n\, h(1/x)$. In the basis $1, x, ..., x^{n-1}$, the linear form $\mathrm{Tr}_{\mathbb{B}/\mathbb{A}}$ may be computed as

$$\begin{aligned}
-\frac{\tilde{h}'(x)}{\tilde{h}(x)} \ =\ & \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(x) + \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(x^2)\, x + \cdots \\
& + \mathrm{Tr}_{\mathbb{B}/\mathbb{A}}(x^n)\, x^{n-1} + O(x^n),
\end{aligned} \tag{2}$$

which uses $O(\mathsf{M}_{\mathbb{A}}(n))$ operations in $\mathbb{A}$. If γ represents the multiplication endomorphism by g in $\mathbb{B}$, then we can write $\varphi = \mathrm{Tr}_{\mathbb{B}/\mathbb{A}} \circ \gamma$ and therefore $\varphi^\top = \gamma^\top \circ \mathrm{Tr}^\top_{\mathbb{B}/\mathbb{A}}$. By transposing the modular multiplication by g with the techniques from [2, Sections 4 and 5], the computation of $\varphi^\top$ requires $O(\mathsf{M}_{\mathbb{A}}(n))$ operations in $\mathbb{A}$.

A.2 Modular inversions

Let $\mathbb{K}$ be a field. Often in computer algebra, we are led to compute in monogen algebras such as $\mathbb{L} = \mathbb{K}[z]/(\lambda(z))$, where λ is separable of degree d, but not necessarily irreducible. Ring operations in $\mathbb{L}$ require $O(\mathsf{M}_{\mathbb{K}}(d))$ operations in $\mathbb{K}$. Algorithms such as the determinant are often slower over rings than their counterparts over fields. For this reason, we often like to conduct our computations as if $\mathbb{L}$ were a field. From the programming point of view, this is easy to achieve: whenever we need to invert an element $a \in \mathbb{L}$

or test whether $a=0$, we factor $\lambda=\lambda_0\,\lambda_1$, where $a=0 \bmod \lambda_0$ and a is invertible modulo λ_1. We then restart the relevant part of the computations with λ_0 and/or λ_1 in the role of λ. This approach is known as the *dynamic evaluation principle* in computer algebra [6, 7]. We borrow the following complexity result for extended gcds from Dahan *et al.* [6]:

PROPOSITION 19. *Let λ be a separable polynomial of degree d over $\mathbb{K}$ and let A and B be univariate polynomials over $\mathbb{L} = \mathbb{K}[z]\,/\,(\lambda(z))$ of degrees $\leqslant n$. Using $O(\mathsf{M}_{\mathbb{K}}(d)\ \mathsf{M}_{\mathbb{K}}(n)\ \log d\ \log n)$ operations in $\mathbb{K}$, one can compute a factorization $\lambda_1 \cdots \lambda_s$ of λ and triples of polynomials (G_i, U_i, V_i) respectively over $\mathbb{K}[z]\,/\,(\lambda_i(z))$, such that $\deg \lambda_i \geqslant 1$, G_i is monic, G_i generates the extension of the ideal (A,B) to $\mathbb{K}[z]\,/\,(\lambda_i(z))$, and the Bézout relation $G_i = A_i\,U_i + B_i\,V_i$ holds with $\deg U_i < \deg B_i$ and $\deg V_i < \deg A_i$, for all $i \in \{1, ..., s\}$.*

PROOF. This statement rephrases [6, Propositions 2.4 and 4.1], where we replace the assumption "λ is square-free over a perfect field" by "λ is separable". In fact all arguments of [6] apply in this setting *mutatis mutandis*. An alternative point of view is to apply [6, Propositions 2.4 and 4.1] over the algebraic closure of $\mathbb{K}$, while noticing that all actual computations are really done over $\mathbb{K}$. □

COROLLARY 20. *Let λ be a separable polynomial of degree d over $\mathbb{K}$ and let A and B be univariate polynomials over $\mathbb{L}=\mathbb{K}[z]\,/\,(\lambda(z))$ of degrees $\leqslant n$, such that B is monic and A is invertible modulo B. Then the inverse of A modulo B may be computed using $O(\mathsf{M}_{\mathbb{K}}(d)\ \mathsf{M}_{\mathbb{K}}(n)\ \log d\ \log n)$ operations in $\mathbb{K}$.*

PROOF. The triples (G_i, U_i, V_i) of the previous proposition satisfy $G_i = 1 = A\,U_i \bmod B$ over $\mathbb{K}[z]\,/\,(\lambda_i(z))$. We recover the inverse U of A modulo B from the U_i using the Chinese remainder theorem. □

5. REFERENCES

[1] D. J. Bernstein. Composing power series over a finite ring in essentially linear time. *J. Symbolic Comput.*, 26(3):339–341, 1998.

[2] A. Bostan, G. Lecerf, and É. Schost. Tellegen's principle into practice. In Hoon Hong, editor, *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation*, ISSAC '03, pages 37–44, New York, NY, USA, 2003. ACM.

[3] R. P. Brent and H. T. Kung. Fast algorithms for manipulating formal power series. *J. ACM*, 25(4):581–595, 1978.

[4] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic complexity theory*, volume 315 of *Grundlehren der Mathematischen Wissenschaften*. Springer-Verlag, 1997.

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

[6] X. Dahan, M. Moreno Maza, É. Schost, and Yuzhen Xie. On the complexity of the D5 principle. In J.-G. Dumas, editor, *Proceedings of Transgressive Computing 2006: a conference in honor of Jean Della Dora*, pages 149–168. U. J. Fourier, Grenoble, France, 2006.

[7] J. Della Dora, C. Dicrescenzo, and D. Duval. A new method for computing in algebraic number fields. In G. Goos and J. Hartmanis, editors, *Eurocal'85 (2)*, volume 174 of *Lect. Notes in Comp. Science*, pages 321–326. Springer, 1985.

[8] J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, New York, 2 edition, 2003.

[9] B. Grenet, J. van der Hoeven, and G. Lecerf. Deterministic root finding over finite fields using Graeffe transforms. *Appl. Alg. Eng. Comm. Comp.*, 27(3):237–257, 2016.

[10] D. Harvey, J. van der Hoeven, and G. Lecerf. Faster polynomial multiplication over finite fields. *J. ACM*, 63(6), 2017. Article 52.

[11] J. van der Hoeven. Relax, but don't be too lazy. *J. Symbolic Comput.*, 34(6):479–542, 2002.

[12] J. van der Hoeven. Fast composition of numeric power series. Technical Report 2008-09, Université Paris-Sud, Orsay, France, 2008.

[13] J. van der Hoeven and G. Lecerf. Modular composition via factorization. Technical report, CNRS & École polytechnique, 2017. http://hal.archives-ouvertes.fr/hal-01457074.

[14] Xiaohan Huang and V. Y. Pan. Fast rectangular matrix multiplication and applications. *J. Complexity*, 14(2):257–299, 1998.

[15] E. Kaltofen and V. Shoup. Fast polynomial factorization over high algebraic extensions of finite fields. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation*, ISSAC '97, pages 184–188, New York, NY, USA, 1997. ACM.

[16] E. Kaltofen and V. Shoup. Subquadratic-time factoring of polynomials over finite fields. *Math. Comp.*, 67(223):1179–1197, 1998.

[17] K. S. Kedlaya and C. Umans. Fast modular composition in any characteristic. In *FOCS'08: IEEE Conference on Foundations of Computer Science*, pages 146–155, Washington, DC, USA, 2008. IEEE Computer Society.

[18] K. S. Kedlaya and C. Umans. Fast polynomial factorization and modular composition. *SIAM J. Comput.*, 40(6):1767–1802, 2011.

[19] F. Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, ISSAC '14, pages 296–303, New York, NY, USA, 2014. ACM.

[20] G. Lecerf. On the complexity of the Lickteig-Roy subresultant algorithm. Technical report, CNRS & École polytechnique, 2017. https://hal.archives-ouvertes.fr/hal-01450869.

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

[22] P. Ritzmann. A fast numerical algorithm for the composition of power series with complex coefficients. *Theoret. Comput. Sci.*, 44:1–16, 1986.