QUASI-OPTIMAL MULTIPLICATION OF LINEAR DIFFERENTIAL OPERATORS (EXTENDED ABSTRACT)

ALEXANDRE BENOIT, ALIN BOSTAN[†], AND JORIS VAN DER HOEVEN[‡]

Abstract. We show that linear differential operators with polynomial coefficients can be multiplied in quasi-optimal time. This answers an open question raised by van der Hoeven.

1. Introduction

The product of *polynomials* and the product of *matrices* are two of the most basic operations in mathematics; the study of their computational complexity is central in computer science. In this paper, we will be interested in the computational complexity of multiplying two *linear differential operators*. These algebraic objects encode linear differential equations, and form a non-commutative ring that shares many properties with the commutative ring of usual polynomials [21, 22]. The structural analogy between polynomials and linear differential equations was discovered long ago by Libri and Brassinne [18, 7, 13]. Yet, the algorithmic study of linear differential operators is currently much less advanced than in the polynomial case: the complexity of multiplication has been addressed only recently [16, 6], but not completely solved. The aim of the current work is to make a step towards filling this gap, and to solve an open question raised in [16].

Let $\mathbb{K}$ be an effective field. That is, we assume data structures for representing the elements of $\mathbb{K}$ and algorithms for performing the field operations. The aim of *algebraic complexity theory* is to study the cost of basic or more complex algebraic operations over $\mathbb{K}$ (such as the cost of computing the greatest common divisor of two polynomials of degrees less than d in $\mathbb{K}[x]$, or the cost of Gaussian elimination on an $r \times r$ matrix in $\mathbb{K}^{r\times r}$) in terms of the number of operations in $\mathbb{K}$. The *algebraic complexity* usually does not coincide with the *bit complexity*, which also takes into account the potential growth of the actual coefficients in $\mathbb{K}$. Nevertheless, understanding the algebraic complexity usually constitutes a first useful step towards understanding the bit complexity. Of course, in the special, very important, case when the field $\mathbb{K}$ is finite, both complexities coincide up to a constant factor.

Date: April 5, 2012.

† Supported in part by the Microsoft Research – INRIA Joint Centre.

‡ Supported by the ANR-09-JCJC-0098-01 MaGiX project, as well as by a Digiteo 2009-36HD grant and Région Ile-de-France.

The complexities of operations in the rings $\mathbb{K}[x]$ and $\mathbb{K}^{r\times r}$ have been intensively studied during the last decades. It is well established that polynomial multiplication is a *commutative complexity yardstick*, while matrix multiplication is a *non-commutative complexity yardstick*, in the sense that the complexity of operations in $\mathbb{K}[x]$ (resp. in $\mathbb{K}^{r\times r}$) can generally be expressed in terms of the cost of multiplication in $\mathbb{K}[x]$ (resp. in $\mathbb{K}^{r\times r}$), and for most of them, in a quasi-linear way [2, 4, 8, 24, 14].

Therefore, understanding the algebraic complexity of multiplication in $\mathbb{K}[x]$ and $\mathbb{K}^{r\times r}$ is a fundamental question. It is well known that two polynomials of degrees $< d$ can be multiplied in time $\mathsf{M}(d) = \mathcal{O}(d \log d \log\log d)$ using algorithms based on the Fast Fourier Transform (FFT) [11, 25, 9], and two $r \times r$ matrices in $\mathbb{K}^{r\times r}$ can be multiplied in time $\mathcal{O}(r^\omega)$, with $2 \leqslant \omega \leqslant 3$ [27, 23, 12]. The current tightest upper bound, due to Vassilevska Williams [28], is $\omega < 2.3727$, following work of Coppersmith and Winograd [12] and Stothers [26]. Finding the best upper bound on ω is one of the most important open problems in algebraic complexity theory.

In a similar vein, our thesis is that understanding the algebraic complexity of multiplication of linear differential operators is a very important question, since the complexity of more involved, higher-level, operations on linear differential operators can be reduced to that of multiplication [17].

Let $\mathbb{K}[x, \partial]$ denote the associative algebra $\mathbb{K}\langle x, \partial; \partial x = x\partial + 1\rangle$ of linear differential operators in $\partial = \frac{d}{dx}$ with polynomial coefficients in x. Any element L of $\mathbb{K}[x, \partial]$ can be written as a finite sum $\sum_i L_i(x)\partial^i$ for uniquely determined polynomials L_i in $\mathbb{K}[x]$. We say that L has bidegree less than (d, r) in (x, ∂) if L has degree less than r in ∂, and if all L_i's have degrees less than d in x. The degree in ∂ of L is usually called the *order* of L.

The main difference with the commutative ring $\mathbb{K}[x, y]$ of bivariate polynomials is the commutation rule $\partial x = x\partial + 1$ that simply encodes, in operator notation, Leibniz's differentiation rule $\frac{d}{dx}(xf) = x\frac{d}{dx}(f) + f$. This slight difference between $\mathbb{K}[x, \partial]$ and $\mathbb{K}[x, y]$ has a considerable impact on the complexity level. On the one hand, it is classical that multiplication in $\mathbb{K}[x, y]$ can be reduced to that of polynomials in $\mathbb{K}[x]$, due to a technique commonly called *Kronecker's trick* [19, 14]. As a consequence, any two polynomials of degrees less than d in x, and less than r in y, can be multiplied in *quasi-optimal time* $\mathcal{O}(\mathsf{M}(dr))$. On the other hand, it was shown by van der Hoeven [16] that, if the base field $\mathbb{K}$ has characteristic zero, then the product of two elements from $\mathbb{K}[x, \partial]$ of bidegree less than (n, n) can be computed in time $\mathcal{O}(n^\omega)$. Moreover, it has been proved in [6] that conversely, multiplication in $\mathbb{K}^{n\times n}$ can be reduced to a constant number of multiplications in $\mathbb{K}[x, \partial]$, in bidegree less than (n, n). In other words, multiplying operators of *well-balanced bidegree* is computationally equivalent to matrix multiplication.

However, contrary to the commutative case, higher-level operations in $\mathbb{K}[x, \partial]$, such as *left common least multiple* (LCLM) and *greatest common right divisor* (GCRD), do not preserve well-balanced bidegrees [15, 5]. For

instance, the LCLM of two operators of bidegrees less than (n, n) is of bidegree less than $(2n(n+1), 2n) = \mathcal{O}(n^2, n)$, and this bound is generically reached. This is a typical phenomenon: operators obtained from computations in $\mathbb{K}[x, \partial]$ tend to have much larger degrees in x than in ∂.

In the general case of operators with possibly unbalanced degrees d in x and r in ∂, the naive algorithm has cost $\mathcal{O}(d^2 r^2 \min(d, r))$; a better algorithm, commonly attributed to Takayama, has complexity $\tilde{\mathcal{O}}(dr \min(d, r))$. We refer to [6, §2] for a review of these algorithms. When $r \leqslant d$ in ∂, the best current upper bound for multiplication is $\mathcal{O}(r^{\omega-2} d^2)$ [16, 17]. It was asked by van der Hoeven [16, §6] whether this complexity could be lowered to $\tilde{\mathcal{O}}(r^{\omega-1} d)$. Here, and hereafter, the soft-O notation $\tilde{\mathcal{O}}(\,)$ indicates that polylogarithmic factors in d and in r are neglected. The purpose of the present work is to provide a positive answer to this open question. Our main result is encapsulated in the following theorem.

Theorem 1. *Let $\mathbb{K}$ be an effective field of characteristic zero. Operators in $\mathbb{K}[x, \partial]$ of bidegree less than (d, r) in (x, ∂) can be multiplied using*

$$\tilde{\mathcal{O}}(dr \min(d, r)^{\omega-2})$$

operations in $\mathbb{K}$.

In the important case $d \geqslant r$, this complexity reads $\tilde{\mathcal{O}}(dr^{\omega-1})$. This is quasi-linear (thus quasi-optimal) with respect to d. Moreover, by the equivalence result from [6, §3], the exponent of r is also the best possible. Besides, under the (plausible, still conjectural) assumption that $\omega = 2$, the complexity in Theorem 1 is almost linear with respect to the output size. For $r = 1$ we retrieve the fact that multiplication in $\mathbb{K}[x]$ in degree $< d$ can be done in quasi-linear time $\tilde{\mathcal{O}}(d)$; from this perspective, the result of Theorem 1 can be seen as a generalization of the fast multiplication for usual polynomials.

In an expanded version [3] of this extended abstract, we will show that analogues of Theorem 1 also hold for other types of *skew polynomials*. More precisely, we will deal with the cases when the skew indeterminate $\partial : f(x) \mapsto f'(x)$ is replaced by the Euler derivative $\delta : f(x) \mapsto xf'(x)$, or a shift operator $\sigma^c : f(x) \mapsto f(x + c)$, or a dilatation $\chi_q : f(x) \mapsto f(qx)$. In [3], we will also prove refined versions of Theorem 1 and complexity bounds for several other interesting operations on skew polynomials.

Main ideas. The fastest known algorithms for multiplication of usual polynomials in $\mathbb{K}[x]$ rely on an evaluation-interpolation strategy at special points in the base field $\mathbb{K}$ [11, 25, 9]. This reduces polynomial multiplication to the "inner product" in $\mathbb{K}$." We adapt this strategy to the case of linear differential operators in $\mathbb{K}[x, \partial]$: the evaluation "points" are *exponential polynomials* of the form $x^n e^{\alpha x}$ on which differential operators act nicely. With this choice, the evaluation and interpolation of operators is encoded by *Hermite evaluation and interpolation* for usual polynomials (generalizing the classical Lagrange interpolation), for which quasi-optimal algorithms exist. For operators of bidegree less than (d, r) in (x, ∂), with $r \geqslant d$, we use $p = \mathcal{O}(r/d)$

evaluation points, and encode the inner multiplication step by p matrix multiplications in size r. All in all, this gives an FFT-type multiplication algorithm for differential operators of complexity $\tilde{\mathcal{O}}(d^{\omega-1}r)$. Finally, we reduce the case $r \leqslant d$ to the case $r \geqslant d$. To do this efficiently, we design a fast algorithm for the computation of the so-called *reflection* of a differential operator, a useful ring morphism that swaps the indeterminates x and ∂, and whose effect is exchanging orders and degrees.

2. Preliminaries

Throughout the paper, $\mathbb{K}[x]_d$ will denote the set of polynomials of degree less than d with coefficients in the field $\mathbb{K}$, and $\mathbb{K}[x,\partial]_{d,r}$ will denote the set of linear differential operators in $\mathbb{K}[x,\partial]$ with degree less than r in ∂, and polynomial coefficients in $\mathbb{K}[x]_d$.

The cost of our algorithms will be measured by the number of field operations in $\mathbb{K}$ they use. We recall that polynomials in $\mathbb{K}[x]_d$ can be multiplied within $\mathsf{M}(d) = \mathcal{O}(d\log(d)\log\log(d)) = \tilde{\mathcal{O}}(d)$ operations in $\mathbb{K}$, using the FFT-based algorithms in [25, 9], and that ω denotes a feasible exponent for matrix multiplication over $\mathbb{K}$, that is, a real constant $2 \leqslant \omega \leqslant 3$, such that two $r \times r$ matrices with coefficients in $\mathbb{K}$ can be multiplied in time $\mathcal{O}(r^\omega)$.

Most basic polynomial operations in $\mathbb{K}[x]_d$ (division, Taylor shift, extended gcd, multipoint evaluation, interpolation, etc.) have cost $\tilde{\mathcal{O}}(d)$ [2, 4, 8, 24, 14]. Our algorithms will make a crucial use of the following result due to Chin [10], see also [20] for a formulation in terms of structured matrices.

Theorem 2 (Fast Hermite evaluation and interpolation)**.** *Let $c_0,\dots,c_{k-1}$ be k integers, $d = \sum_i c_i$, and let $\mathbb{K}$ be an effective field of characteristic zero. Given k mutually distinct points $\alpha_0,\dots,\alpha_{k-1}$ in $\mathbb{K}$ and a polynomial $P \in \mathbb{K}[x]_d$, one can compute the vector of d values*

$$\begin{aligned}\mathcal{H} \;=\; & (P(\alpha_0), P'(\alpha_0), \dots, P^{(c_0-1)}(\alpha_0), \dots\dots,\\ & P(\alpha_{k-1}), P'(\alpha_{k-1}), \dots, P^{(c_{k-1}-1)}(\alpha_{k-1}))\end{aligned}$$

in $\mathcal{O}(\mathsf{M}(d)\log(k)) = \tilde{\mathcal{O}}(d)$ arithmetic operations in $\mathbb{K}$. Conversely, P is uniquely determined by $\mathcal{H}$, and its coefficients can be recovered from $\mathcal{H}$ in $\mathcal{O}(\mathsf{M}(d)\log(k)) = \tilde{\mathcal{O}}(d)$ arithmetic operations in $\mathbb{K}$.

3. The new algorithm in the case $r \geqslant d$

3.1. **Multiplication by evaluation and interpolation.** Most fast algorithms for multiplying two polynomials $P, Q \in \mathbb{K}[x]_d$ are based on the evaluation-interpolation strategy. The idea is to pick $2d$ distinct points $\alpha_0,\dots,\alpha_{2d-1}$ in $\mathbb{K}$, and to perform the following three steps:

Evaluation: Evaluate P and Q at $\alpha_0,\dots,\alpha_{2d-1}$.
Inner multiplication: Compute $(PQ)(\alpha_i) = P(\alpha_i)Q(\alpha_i)$ for $i < 2d$.
Interpolation: Recover PQ from $(PQ)(\alpha_0),\dots,(PQ)(\alpha_{2d-1})$.

The inner multiplication step requires only $\mathcal{O}(d)$ operations. Consequently, if both the evaluation and interpolation steps can be performed fast, then we obtain a fast algorithm for multiplying P and Q. For instance, if $\mathbb{K}$ contains a 2^p-th primitive root of unity with $2^p \leqslant 2d \leqslant 2^{p+1}$, then both evaluation and interpolation can be performed in time $\mathcal{O}(d \log d)$ using the Fast Fourier Transform [11].

For a linear differential operator $L \in \mathbb{K}[x, \partial]_{d,r}$, it is natural to consider evaluations at powers of x instead of roots of unity. It is also natural to represent the evaluation of L at a suitable number of such powers by a matrix. More precisely, given $k \in \mathbb{N}$, we may regard L as an operator from $\mathbb{K}[x]_k$ into $\mathbb{K}[x]_{k+d}$. We may also regard elements of $\mathbb{K}[x]_k$ and $\mathbb{K}[x]_{k+d}$ as column vectors, written in the canonical bases with powers of x. We will denote by

$$\Phi_L^{k+d,k} = \begin{pmatrix} L(1)_0 & \cdots & L(x^{k-1})_0 \\ \vdots & & \vdots \\ L(1)_{k+d-1} & \cdots & L(x^{k-1})_{k+d-1} \end{pmatrix} \in \mathbb{K}^{(k+d)\times k}$$

the matrix of L with respect to these bases. Given two operators K, L in $\mathbb{K}[x, \partial]_{d,r}$, we clearly have

$$\Phi_{KL}^{k+2d,k} = \Phi_K^{k+2d,k+d} \Phi_L^{k+d,k}, \quad \text{for all } k \geqslant 0.$$

For $k = 2r$ (or larger), the operator KL can be recovered from the matrix $\Phi_{KL}^{2r+2d,2r}$, whence the formula

$$\Phi_{KL}^{2r+2d,2r} = \Phi_K^{2r+2d,2r+d} \Phi_L^{2r+d,2r} \tag{1}$$

yields a way to multiply K and L. For the complexity analysis, we thus have to consider the three steps:

Evaluation: Computation of $\Phi_K^{2r+2d,2r+d}$ and $\Phi_L^{2r+d,2r}$ from K and L.
Inner multiplication: Computation of the matrix product (1).
Interpolation: Recovery of KL from $\Phi_{KL}^{2r+2d,2r}$.

In [16, 6], this multiplication method was applied with success to the case when $d = r$. In this "square case", the following result was proved in [6, §4.2].

Lemma 1. *Let $L \in \mathbb{K}[x, \partial]_{d,d}$. Then*

(1) *We may compute $\Phi_L^{2d,d}$ as a function of L in time $\mathcal{O}(d\mathsf{M}(d))$.*
(2) *We may recover L from $\Phi_L^{2d,d}$ in time $\mathcal{O}(d\mathsf{M}(d))$.*

3.2. **Evaluation and interpolation at exponential polynomials.** Assume now that $r > d$. Then a straightforward application of the above evaluation-interpolation strategy yields an algorithm of suboptimal complexity. Indeed, the matrix $\Phi_{KL}^{2r+2d,2r}$ contains a lot of redundant information and since its mere size exceeds r^2, one cannot expect a direct multiplication algorithm of quasi-optimial complexity $\tilde{\mathcal{O}}(rd^{\omega-1})$.

In order to maintain quasi-optimal complexity in this case as well, the idea is to evaluate at so called *exponential polynomials* instead of ordinary polynomials. More specifically, given $L \in \mathbb{K}[x,\partial]_{d,r}$ and $\alpha \in \mathbb{K}$, we will use the fact that L also operates nicely on the vector space $\mathbb{K}[x]\mathrm{e}^{\alpha x}$. Moreover, for any $P \in \mathbb{K}[x]$, we have

$$L(P\mathrm{e}^{\alpha x}) \quad = \quad L_{\ltimes\alpha}(P)\mathrm{e}^{\alpha x},$$

where

$$L_{\ltimes\alpha} \quad = \quad \sum_i L_i(x)(\partial+\alpha)^i$$

is the operator obtained by substituting $\partial+\alpha$ for ∂ in $L = \sum_i L_i(x)\partial^i$. Indeed, this is a consequence of the fact that, by Leibniz's rule:

$$\partial^i(P\mathrm{e}^{\alpha x}) = \left(\sum_{j\leqslant i}\binom{i}{j}\alpha^j\partial^{i-j}P\right)\mathrm{e}^{\alpha x} = (\partial+\alpha)^i(P)\mathrm{e}^{\alpha x}.$$

Now let $p = \lceil r/d\rceil$ and let $\alpha_0,\ldots,\alpha_{p-1}$ be p pairwise distinct points in $\mathbb{K}$. For each k, we define the vector space

$$\mathbb{V}_k \quad = \quad \mathbb{K}[x]_k\mathrm{e}^{\alpha_0 x} \oplus \cdots \oplus \mathbb{K}[x]_k\mathrm{e}^{\alpha_{p-1}x}$$

with canonical basis

$$(\mathrm{e}^{\alpha_0 x},\ldots,x^{k-1}\mathrm{e}^{\alpha_0 x},\ldots\ldots,\mathrm{e}^{\alpha_{p-1}x},\ldots,x^{k-1}\mathrm{e}^{\alpha_{p-1}x}).$$

Then we may regard L as an operator from $\mathbb{V}_k$ into $\mathbb{V}_{k+d}$ and we will denote by $\Phi_L^{[k+d,k]}$ the matrix of this operator with respect to the canonical bases. By what precedes, this matrix is block diagonal, with p blocks of size d:

$$\Phi_L^{[k+d,k]} \quad = \quad \begin{pmatrix} \Phi_{L_{\ltimes\alpha_0}}^{k+d,k} & & \\ & \ddots & \\ & & \Phi_{L_{\ltimes\alpha_{p-1}}}^{k+d,k} \end{pmatrix}.$$

Let us now show that the operator L is uniquely determined by the matrix $\Phi_L^{[2d,d]}$, and that this gives rise to an efficient algorithm for multiplying two operators in $\mathbb{K}[x,\partial]_{d,r}$.

Lemma 2. *Let $L \in \mathbb{K}[x,\partial]_{d,r}$ with $r \geqslant d$ and let $p = \lceil r/d\rceil$. Then*

1. *We may compute $\Phi_L^{[2d,d]}$ as a function of L in time $\mathcal{O}(d\mathsf{M}(r)\log r)$.*
2. *We may recover L from $\Phi_L^{[2d,d]}$ in time $\mathcal{O}(d\mathsf{M}(r)\log r)$.*

Proof. For any operator $L = \sum_{i<d,j<r} L_{i,j}x^i\partial^j$ in $\mathbb{K}[x,\partial]_{d,r}$, we define its truncation L^* at order $\mathcal{O}(\partial^d)$ by

$$L^* \quad = \quad \sum_{i<d,j<d} L_{i,j}x^i\partial^j.$$

Since $L - L^*$ vanishes on $\mathbb{K}[x]_d$, we notice that $\Phi_L^{2d,d} = \Phi_{L^*}^{2d,d}$.

If $L \in \mathbb{K}[\partial]_r$, then L^* can be regarded as the power series expansion of L at $\partial = 0$ and order d. More generally, for any $i \in \{0, \ldots, p-1\}$, the operator $L^*_{\kappa\alpha_i}$ coincides with the Taylor series expansion at $\partial = \alpha_i$ and order d:

$$\begin{aligned} L^*_{\kappa\alpha_i}(\partial) &= L(\partial + \alpha_i)^* \\ &= L(\alpha_i) + L'(\alpha_i)\partial + \cdots + \tfrac{1}{(d-1)!} L^{(d-1)}(\alpha_i)\partial^{d-1}. \end{aligned}$$

In other words, the computation of the truncated operators $L^*_{\kappa\alpha_0}, \ldots, L^*_{\kappa\alpha_{p-1}}$ as a function of L corresponds to a Hermite evaluation at the points α_i, with multiplicity $c_i = d$ at each point α_i. By Theorem 2, this computation can be performed in time $\mathcal{O}(\mathsf{M}(pd)\log(pd)) = \mathcal{O}(\mathsf{M}(r)\log r)$. Furthermore, Hermite interpolation allows us to recover L from $L^*_{\kappa\alpha_0}, \ldots, L^*_{\kappa\alpha_{p-1}}$ with the same time complexity $\mathcal{O}(\mathsf{M}(r)\log r)$.

Now let $L \in \mathbb{K}[x, \partial]_{d,r}$ and consider the expansion of L in x

$$L(x, \partial) = L_0(\partial) + \cdots + x^{d-1} L_{d-1}(\partial).$$

For each i, one Hermite evaluation of L_i allows us to compute the $L^*_{\kappa\alpha_j, i}$ with $j < p$ in time $\mathcal{O}(\mathsf{M}(r)\log r)$. The operators $L^*_{\kappa\alpha_j}$ with $j < p$ can therefore be computed in time $\mathcal{O}(d\mathsf{M}(r)\log r)$. By Lemma 1, we need $\mathcal{O}(r\mathsf{M}(d)) = \mathcal{O}(d\mathsf{M}(r))$ additional operations in order to obtain $\Phi_L^{[2d,d]}$. Similarly, given $\Phi_L^{[2d,d]}$, Lemma 1 allows us to recover the operators $L^*_{\kappa\alpha_j}$ with $j < p$ in time $\mathcal{O}(d\mathsf{M}(r)\log r)$. Using d Hermite interpolations, we also recover the coefficients L_i of L in time $\mathcal{O}(d\mathsf{M}(r)\log r)$. □

Theorem 3. *Let $K, L \in \mathbb{K}[x, \partial]_{d,r}$ with $r \geqslant d$. Then we may compute the product KL in time $\mathcal{O}(d^{\omega-1}r + d\mathsf{M}(r)\log r)$.*

Proof. Considering K and L as operators in $\mathbb{K}[x, \partial]_{3d,3r}$, Lemma 2 implies that the computation of $\Phi_K^{[4d,3d]}$ and $\Phi_L^{[3d,2d]}$ as a function of K and L can be done in time $\mathcal{O}(d\mathsf{M}(r)\log r)$. The multiplication

$$\Phi_{KL}^{[4d,2d]} = \Phi_K^{[4d,3d]} \Phi_L^{[3d,2d]}$$

can be done in time $\mathcal{O}(pd^\omega) = \mathcal{O}(d^{\omega-1}r)$. Lemma 2 finally implies that we may recover KL from $\Phi_{KL}^{[4d,2d]}$ in time $\mathcal{O}(d\mathsf{M}(r)\log r)$. □

4. The new algorithm in the case $d \geqslant r$

Any differential operator $L \in \mathbb{K}[x, \partial]_{d,r}$ can be written in a unique form

$$L = \sum_{i<r, j<d} L_{i,j} x^j \partial^i, \quad \text{for some scalars} \quad L_{i,j} \in \mathbb{K}.$$

This representation, with x on the left and ∂ on the right, is called the *canonical form of L*.

Let $\varphi : \mathbb{K}[x, \partial] \to \mathbb{K}[x, \partial]$ denote the map defined by

$$\varphi\left(\sum_{i<r, j<d} L_{i,j} x^j \partial^i \right) = \sum_{i<r, j<d} L_{i,j} \partial^j (-x)^i.$$

In other words, φ is the unique $\mathbb{K}$-algebra automorphism of $\mathbb{K}[x,\partial]$ that keeps the elements of $\mathbb{K}$ fixed, and is defined on the generators of $\mathbb{K}[x,\partial]$ by $\varphi(x) = \partial$ and $\varphi(\partial) = -x$. We will call φ the *reflection morphism* of $\mathbb{K}[x,\partial]$. The map φ enjoys the nice property that it sends $\mathbb{K}[x,\partial]_{d,r}$ onto $\mathbb{K}[x,\partial]_{r,d}$. In particular, to an operator whose order is higher than its degree, φ associates a "mirror operator" whose degree is higher than its order.

4.1. **Main idea of the algorithm in the case $d \geqslant r$.** If $d \geqslant r$, then the reflection morphism φ is the key to our fast multiplication algorithm of operators in $\mathbb{K}[x,\partial]_{d,r}$, since it allows us to reduce this case to the previous case when $r \geqslant d$. More precisely, given K, L in $\mathbb{K}[x,\partial]_{d,r}$ with $d \geqslant r$, the main steps of the algorithm are:

(S1) compute the canonical forms of $\varphi(K)$ and $\varphi(L)$,
(S2) compute the product $M = \varphi(K)\varphi(L)$ of operators $\varphi(K) \in \mathbb{K}[x,\partial]_{r,d}$ and $\varphi(L) \in \mathbb{K}[x,\partial]_{r,d}$, using the algorithm described in the previous section, and
(S3) return the (canonical form of the) operator $KL = \varphi^{-1}(M)$.

Since $d \geqslant r$, step (S2) can be performed in complexity $\tilde{\mathcal{O}}(r^{\omega-1}d)$ using the results of Section 3. In the next subsection, we will prove that both steps (S1) and (S3) can be performed in $\tilde{\mathcal{O}}(rd)$ operations in $\mathbb{K}$. This will enable us to conclude the proof of Theorem 1.

4.2. **Quasi-optimal computation of reflections.** We now show that the *reflection* and the *inverse reflection* of a differential operator can be computed quasi-optimally. The idea is that performing reflections can be interpreted in terms of Taylor shifts for polynomials, which can be computed in quasi-linear time using the algorithm from [1].

A first observation is that the composition $\varphi \circ \varphi$ is equal to the involution $\psi : \mathbb{K}[x,\partial] \to \mathbb{K}[x,\partial]$ defined by

$$\psi\left(\sum_{i<r,j<d} L_{i,j} x^j \partial^i\right) = \sum_{i<r,j<d} (-1)^{i+j} L_{i,j} x^j \partial^i.$$

As a direct consequence of this fact, it follows that the map φ^{-1} is equal to $\varphi \circ \psi$. Since $\psi(L)$ is already in canonical form, computing $\psi(L)$ only consists of sign changes, which can be done in linear time $\mathcal{O}(dr)$. Therefore, computing the inverse reflection $\varphi^{-1}(L)$ can be performed within the same cost as computing the direct reflection $\varphi(L)$, up to a linear overhead $\mathcal{O}(rd)$.

In the remainder of this section, we focus on the fast computation of direct reflections. The key observation is encapsulated in the next lemma. Here, and in what follows, we use the convention that the entries of a matrix corresponding to indices beyond the matrix sizes are all zero.

Lemma 3. *Assume that $(p_{i,j})$ and $(q_{i,j})$ are two matrices in $\mathbb{K}^{r\times d}$ such that*

$$\sum_{i,j} q_{i,j} x^i \partial^j = \sum_{i,j} p_{i,j} \partial^j x^i.$$

Then

$$i!\,q_{i,j} = \sum_{k\geqslant 0} \binom{j+k}{k} (i+k)!\,p_{i+k,j+k}$$

and where we use the convention that $p_{i,j} = 0$ *as soon as* $i \geqslant r$ *or* $j \geqslant d$.

Proof. Leibniz's differentiation rule implies the commutation rule

$$\partial^j \frac{x^i}{i!} \quad = \quad \sum_{k=0}^{j} \binom{j}{k} \frac{x^{i-k}}{(i-k)!} \partial^{j-k}.$$

Together with the hypothesis, this implies the equality

$$\sum_{i,j} (i!\,q_{i,j}) \frac{x^i}{i!} \partial^j = \sum_{i,j} (i!\,p_{i,j}) \partial^j \frac{x^i}{i!} = \sum_{k\geqslant 0} \left(\sum_{i,j} (i!\,p_{i,j}) \binom{j}{k} \frac{x^{i-k}}{(i-k)!} \partial^{j-k} \right).$$

We conclude by extraction of coefficients. □

Theorem 4. *Let* $L \in \mathbb{K}[x,\partial]_{d,r}$. *Then we may compute* $\varphi(L)$ *and* $\varphi^{-1}(L)$ *using* $\mathcal{O}(\min(d\,\mathsf{M}(r), r\,\mathsf{M}(d))) = \tilde{\mathcal{O}}(rd)$ *operations in* $\mathbb{K}$.

Proof. We first study the case $r \geqslant d$. If $L = \sum_{i<r,j<d} p_{i,j} x^j \partial^i$, then by the first equality of Lemma 3, the reflection $\varphi(L)$ is equal to

$$\varphi(L) = \sum_{i<r,j<d} p_{i,j} \partial^j (-x)^i = \sum_{i<r,j<d} q_{i,j} (-x)^j \partial^i,$$

where

$$i!\,q_{i,j} = \sum_{\ell\geqslant 0} \binom{j+\ell}{j} (i+\ell)!\,p_{i+\ell,j+\ell}. \tag{2}$$

For any fixed k with $1-r \leqslant k \leqslant d-1$, let us introduce $G_k = \sum_i i! q_{i,i+k} x^{i+k}$ and $F_k = \sum_i i! p_{i,i+k} x^{i+k}$. These polynomials belong to $\mathbb{K}[x]_d$, since $p_{i,j} = q_{i,j} = 0$ for $j \geqslant d$. If $k \leqslant 0$, then Equation (2) translates into

$$G_k(x) \quad = \quad F_k(x+1).$$

Indeed, Equation (2) with $j = i+k$ implies that $G_k(x)$ is equal to

$$\sum_{i,\ell} \binom{i+k+\ell}{i+k} (i+\ell)!\,p_{i+\ell,i+k+\ell}\,x^{i+k} = \sum_{j,s} j! p_{j,j+k} \binom{j+k}{s} x^s = F_k(x+1).$$

Similarly, if $k > 0$, then the coefficients of x^i in $G_k(x)$ and $F_k(x+1)$ still coincide for all $i \geqslant k$. In particular, we may compute $G_{1-r}, \ldots, G_{d-1}$ from $F_{1-r}, \ldots, F_{d-1}$ by means of $d+r \leqslant 2r$ Taylor shifts of polynomials in $\mathbb{K}[x]_d$. Using the fast algorithm for Taylor shift in [1], this can be done in time $\mathcal{O}(r\,\mathsf{M}(d))$.

Once the coefficients of the G_k's are available, the computation of the coefficients of $\varphi(L)$ requires $\mathcal{O}(dr)$ additional operations.

If $d \geqslant r$, then we notice that the equality (2) is equivalent to

$$j!\, q_{i,j} = \sum_{\ell \geqslant 0} \binom{i+\ell}{i} (j+\ell)!\, p_{i+\ell,j+\ell},$$

as can be seen by expanding the binomial coefficients. Redefining $G_k := \sum_i q_{i+k,i} x^{i+k}$ and $F_k := \sum_i i! p_{i+k,} x^{i+k}$, similar arguments as above show that $\varphi(P)$ can be computed using $\mathcal{O}(d\,\mathsf{M}(r))$ operations in $\mathbb{K}$.

By what has been said at the beginning of this section, we finally conclude that the inverse reflection $\varphi^{-1}(L) = \varphi(\psi(L))$ can be computed for the same cost as the direct reflection $\varphi(L)$. □

4.3. **Proof of Theorem 1 in the case $d \geqslant r$.** We will prove a slightly better result:

Theorem 5. *Let $K, L \in \mathbb{K}[x,\partial]_{d,r}$ with $d \geqslant r$. Then we may compute the product KL using $\mathcal{O}(r^{\omega-1}d + r\mathsf{M}(d)\log d)$ operations in $\mathbb{K}$.*

Proof. Assume that K and L are two operators in $\mathbb{K}[x,\partial]_{d,r}$ with $d \geqslant r$. Then $\varphi(K)$ and $\varphi(L)$ belong to $\mathbb{K}[x,\partial]_{r,d}$, and their canonical forms can be computed in $\mathcal{O}(d\mathsf{M}(r))$ operations by Theorem 4. Using the algorithm from section 3, we may compute $M = \varphi(L)\varphi(K)$ in $\mathcal{O}(r^{\omega-1}d + r\mathsf{M}(d)\log d)$ operations. Finally, $LK = \varphi^{-1}(M)$ can be computed in $\mathcal{O}(r\mathsf{M}(d))$ operations by Theorem 4. We conclude by adding up the costs of these three steps. □

References

[1] A. V. Aho, K. Steiglitz, and J. D. Ullman. Evaluating polynomials at fixed sets of points. *SIAM J. Comput.*, 4(4):533–539, 1975.

[2] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The design and analysis of computer algorithms*. Addison-Wesley Publishing Co., 1974.

[3] A. Benoit, A. Bostan, and J. van der Hoeven. Fast multiplication of skew polynomials. In preparation.

[4] Dario Bini and Victor Y. Pan. *Polynomial and matrix computations. Vol. 1*. Progress in Theoretical Computer Science. Birkhäuser Boston Inc., Boston, MA, 1994. Fundamental algorithms.

[5] A. Bostan, F. Chyzak, Z. Li, and B. Salvy. Fast computation of common left multiples of linear ordinary differential operators. In *ISSAC'12*, 2012. To appear. Preliminary version available at http://algo.inria.fr/bostan/publications/BoChLiSa12.pdf.

[6] Alin Bostan, Frédéric Chyzak, and Nicolas Le Roux. Products of ordinary differential operators by evaluation and interpolation. In *ISSAC 2008*, pages 23–30. ACM, New York, 2008.

[7] E. Brassinne. Analogie des équations différentielles linéaires à coefficients variables, avec les équations algébriques. In *Note III du Tome 2 du Cours d'analyse de Ch. Sturm, École polytechnique, 2ème édition*, pages 331–347, 1864.

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

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

[10] Francis Y. Chin. A generalized asymptotic upper bound on fast polynomial evaluation and interpolation. *SIAM J. Comput.*, 5(4):682–690, 1976.

[11] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19:297–301, 1965.
[12] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, March 1990.
[13] S. S. Demidov. On the history of the theory of linear differential equations. *Arch. Hist. Exact Sci.*, 28(4):369–387, 1983.
[14] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, Cambridge, second edition, 2003.
[15] D. Yu. Grigor′ev. Complexity of factoring and calculating the GCD of linear ordinary differential operators. *J. Symbolic Comput.*, 10(1):7–37, 1990.
[16] J. van der Hoeven. FFT-like multiplication of linear differential operators. *Journal of Symbolic Computation*, 33(1):123–127, 2002.
[17] J. van der Hoeven. On the complexity of skew arithmetic, 2011. Technical Report, HAL 00557750, http://hal.archives-ouvertes.fr/hal-00557750.
[18] G. Libri. Mémoire sur la résolution des équations algébriques dont les racines ont entre elles un rapport donné, et sur l'intégration des équations différentielles linéaires dont les intégrales particulières peuvent s'exprimer les unes par les autres. *J. Reine Angew. Math.*, 10:167–194, 1833.
[19] Robert T. Moenck. Another polynomial homomorphism. *Acta Informat.*, 6(2):153–169, 1976.
[20] Vadim Olshevsky and Amin Shokrollahi. Matrix-vector product for confluent Cauchy-like matrices with application to confluent rational interpolation. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 573–581 (electronic), New York, 2000. ACM.
[21] Oystein Ore. Formale Theorie der linearen Differentialgleichungen. *J. Reine Angew. Math.*, 167:221–234, 1932.
[22] Oystein Ore. Theory of non-commutative polynomials. *Ann. of Math. (2)*, 34(3):480–508, 1933.
[23] Victor Pan. *How to multiply matrices faster*, volume 179 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1984.
[24] Victor Y. Pan. *Structured matrices and polynomials*. Birkhäuser Boston Inc., Boston, MA, 2001. Unified superfast algorithms.
[25] A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7:281–292, 1971.
[26] A. Stothers. *On the Complexity of Matrix Multiplication*. PhD thesis, University of Edinburgh, 2010.
[27] V. Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13:354–356, 1969.
[28] V. Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In *44th ACM Symp. on Theory of Computing (STOC'12)*, 2012. To appear. Preliminary version available at http://cs.berkeley.edu/~virgi/matrixmult.pdf.

Université Pierre et Marie Curie (France)
E-mail address: Alexandre.Benoit@lip6.fr

INRIA (France)
E-mail address: Alin.Bostan@inria.fr

CNRS & École polytechnique (France)
E-mail address: vdhoeven@lix.polytechnique.fr