Amortized bivariate multi-point evaluation*†

JORIS VAN DER HOEVEN[ab], GRÉGOIRE LECERF[ac]

a. CNRS, École polytechnique, Institut Polytechnique de Paris
Laboratoire d'informatique de l'École polytechnique (LIX, UMR 7161)
1, rue Honoré d'Estienne d'Orves
Bâtiment Alan Turing, CS35003
91120 Palaiseau, France

b. Email: `vdhoeven@lix.polytechnique.fr`
c. Email: `lecerf@lix.polytechnique.fr`

Preliminary version of May 11, 2021

The evaluation of a polynomial at several points is called the problem of multi-point evaluation. Sometimes, the set of evaluation points is fixed and several polynomials need to be evaluated at this set of points. Efficient algorithms for this kind of "amortized" multi-point evaluation were recently developed for the special case when the set of evaluation points is sufficiently generic. In this paper, we design a new algorithm for arbitrary sets of points, while restricting ourselves to bivariate polynomials.

1. INTRODUCTION

Let $\mathbb{K}$ be an effective field, so that we have algorithms for the field operations. Given a polynomial $P \in \mathbb{K}[x_1, \ldots, x_D]$ and a tuple $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n) \in (\mathbb{K}^D)^n$ of points, the computation of $P(\boldsymbol{\alpha}) = (P(\alpha_1), \ldots, P(\alpha_n)) \in \mathbb{K}^n$ is called the problem of *multi-point evaluation*. The converse problem is called *interpolation* and takes a candidate support of P as input.

These problems naturally occur in several areas of applied algebra. When solving a polynomial system, multi-point evaluation can for instance be used to check whether all points in a given set are indeed solutions of the system. In [14], we have shown that fast algorithms for multi-point evaluation actually lead to efficient algorithms for polynomial system solving. The more specific problem of bivariate multi-point evaluation appears for example in the computation of generator matrices of algebraic geometry error correcting codes [18].

The general problem of multivariate multi-point evaluation is notoriously hard. If $\mathbb{K} = \mathbb{Q}$ or $\mathbb{K}$ is a field of finite characteristic, then theoretical algorithms of Kedlaya and Umans [17] achieve a complexity exponent $1 + \epsilon$, where $\epsilon > 0$ represents a constant that can be taken arbitrarily close to zero. Unfortunately, to our best knowledge, these algorithms do not seem suitable for practical purposes [13, Conclusion].

The best known bound for $D = 2$ over general fields is due to Nüsken and Ziegler [22]: the evaluation of P at $n = O(\deg_{x_1} P \deg_{x_2} P)$ points can be done with $O(\deg_{x_1} P\, (\deg_{x_2} P)^{\varpi})$ operations in $\mathbb{K}$. This bound is based on the Paterson–Stockmeyer technique for modular composition [23]. Here, the constant $\varpi > 1.5$ is a real value such that the product of a $m \times \sqrt{m}$ matrix by a $\sqrt{m} \times \sqrt{m}$ matrix takes $O(m^{\varpi})$ operations; one may take $\varpi < 1.667$;

∗. This paper is part of a project that has received funding from the French "Agence de l'innovation de défense".

†. This article has been written using GNU $\mathrm{T_EX_{MACS}}$ [10].

see [16, Theorem 10.1]. We further cite [15] for an efficient algorithm in the case of special sets of points $\boldsymbol{\alpha}$.

Last year, new softly linear algorithms have been proposed for multi-point evaluation and interpolation in the case when $\boldsymbol{\alpha}$ is a fixed generic tuple of points [12, 20]. These algorithms are *amortized* in the sense that potentially expensive precomputations as a function of $\boldsymbol{\alpha}$ are allowed. When the dimension D is arbitrary but fixed, the algorithms from [12] take softly linear time: they generalize the classical univariate "divide and conquer" approach, as presented for instance in [6, chapter 10]. The results in [20] are restricted to the case $D=2$. They take into account the partial degrees of P and are based on changes of polynomial bases that are similar to the ones of [11, section 6.2].

In the present paper, we turn our attention to arbitrary (*i.e.* possibly non-generic) tuples of evaluation points $\boldsymbol{\alpha}$, while restricting ourselves to the bivariate case $D=2$ and $\deg P = O(\sqrt{n})$. Combining ideas from [12] and [20], we present a new softly linear algorithm for amortized multi-point evaluation. For the sake of simplicity, we have not optimized all constant factors involved in the cost analysis of our new algorithm, so our complexity bound is mostly of theoretical interest for the moment. The opposite task of interpolation is more subtle: since interpolants of total degree $O(\sqrt{n})$ do not necessarily exist, the very problem needs to be stated with additional care. For this reason, we do not investigate interpolation in this paper.

2. WEIGHTED BIVARIATE POLYNOMIALS

Our bivariate multi-point evaluation makes use of polynomial arithmetic with respect to weighted graded monomial orderings. This section is devoted to the costs of products and divisions in this context.

2.1. Complexity model

For complexity analyses, we will only consider algebraic complexity models like computation trees [3], for which elements in $\mathbb{K}$ are freely at our disposal. The time complexity simply measures the number of arithmetic operations and zero-tests in $\mathbb{K}$.

We denote by $\mathsf{M}(d)$ the time that is needed to compute a product PQ of two polynomials $P, Q \in \mathbb{K}[x]$ of degree $<d$. We make the usual assumptions that $\mathsf{M}(d)/d$ is non-decreasing as a function of d. Using a variant of the Schönhage–Strassen algorithm [4], it is well known that we may take $\mathsf{M}(d) = O(d \log d \log \log d)$. If we restrict our attention to fields $\mathbb{K}$ of positive characteristic, then we may even take $\mathsf{M}(d) = O(d \log d \, 4^{\log^* d})$ [7].

2.2. Monomial orderings

General monomial orderings, that are suitable for Gröbner basis computations, have been classified in [24]. For the purpose of this paper, we focus on the following specific family of bivariate monomial orderings.

DEFINITION 1. *Let $k \in \mathbb{N} \setminus \{0\}$. We define the* ***k-degree*** *of a monomial $x^a y^b$ with $a, b \in \mathbb{N}$ by*

$$\deg_k(x^a y^b) := a + kb.$$

We define the ***k-ordering*** *to be the monomial ordering $\prec_k$ such that*

$$x^a y^b \prec_k x^u y^v \quad \Leftrightarrow \quad \begin{cases} a+kb < u+kv & \text{or} \\ a+kb = u+kv \text{ and } b<v. \end{cases}$$

Let us mention that the idea of using such kinds of families of monomial orderings in the design of fast algorithms for bivariate polynomials previously appeared in [11].

2.3. Multiplication

Consider the product $C = A B$ of two non-zero bivariate polynomials $A, B \in \mathbb{K}[x, y]$ and the obvious bound

$$s := (\deg_x A + \deg_x B + 1)(\deg_y A + \deg_y B + 1)$$

for the number of terms of C. Then it is well known that Kronecker substitution allows for the computation of the product C using $O(\mathsf{M}(s))$ operations in $\mathbb{K}$; see [6, Corollary 8.27], for instance.

Writing $\operatorname{val}_x A$ for the valuation of A in x, the number of non-zero terms of C is more accurately bounded by

$$\tilde{s} := (\deg_x A + \deg_x B - \operatorname{val}_x A - \operatorname{val}_x B + 1)(\deg_y A + \deg_y B + 1).$$

Via the appropriate multiplications and divisions by powers of x, we observe that C can be computed using $O(\mathsf{M}(\tilde{s}))$ operations in $\mathbb{K}$.

Let us next show that a similar bound holds for slices of polynomials that are dense with respect to the k-ordering. More precisely, let $\operatorname{val}_k A$ denote the minimum of $i + k j$ over the monomials $x^i y^j$ occurring in A. By convention we set $\operatorname{val}_k 0 := +\infty$.

LEMMA 2. *Let $A, B \in \mathbb{K}[x, y]$. The product $C = A B$ can be computed using $O(\mathsf{M}(s_k))$ operations in $\mathbb{K}$, where*

$$s_k := (\deg_k A + \deg_k B - \operatorname{val}_k A - \operatorname{val}_k B + 1)(\deg_y A + \deg_y B + 1).$$

Proof. From the monomial identity

$$x^i (x^k y)^j = x^{i+kj} y^j$$

we observe that the monomials of k-degree d are in one-to-one correspondence with the monomials of degree d in x and degree in y in $\{0, \ldots, \lfloor d/k \rfloor\}$. It also follows that the number of terms of a non-zero polynomial A is bounded by

$$(\deg_k A - \operatorname{val}_k A + 1)(\deg_y A + 1),$$

and that

$$\operatorname{val}_x(A(x, x^k y)) = \operatorname{val}_k A$$
$$\deg_x(A(x, x^k y)) = \deg_k A.$$

In addition, the number of non-zero terms in the product $C = A B$ is bounded by s_k. So it suffices to compute C *via* the formula

$$C(x, x^k y) = A(x, x^k y) B(x, x^k y)$$

in order to obtain the claimed complexity bound. □

2.4. Division

Let B be a polynomial in $\mathbb{K}[x, y]$ of k-degree δ and of leading monomial written $x^\alpha y^\beta$. Without loss of generality we may assume that the coefficient of this monomial is 1. We examine the cost of the division of A of k-degree d by B with respect to $\prec_k$:

$$A = Q B + R,$$

where Q and R are in $\mathbb{K}[x,y]$, and such that no monomial in R is divisible by $x^{\alpha} y^{\beta}$. Such a division does exist: this is a usual fact from the theory of Gröbner bases. In this context, a polynomial A is said to be *reduced* with respect to B when none of its terms is divisible by $x^{\alpha} y^{\beta}$.

If $A = \tilde{Q} B + \tilde{R}$ for polynomials $\tilde{Q}$ and $\tilde{R}$ such that $\tilde{R}$ is reduced with respect to B, then $(Q - \tilde{Q})\, B = \tilde{R} - R$, so $\tilde{Q} = Q$ and $\tilde{R} = R$. In other words, Q and R are unique, so we may write $\mathrm{quo}_k(A, B)$ for *the* quotient Q of A by B with respect to $\prec_k$.

In the remainder of this section, we assume that k has been fixed once and for all. Given $A = \sum_{(i,j) \in \mathbb{N}^2} A_{i,j} x^i y^j \in \mathbb{K}[x,y]$, we define

$$A_{[a]} \;:=\; \sum_{i+kj=a} A_{i,j} x^i y^j, \qquad\qquad A_{(a,b]} \;:=\; \sum_{a<i+kj\leqslant b} A_{i,j} x^i y^j.$$

The naive division algorithm proceeds as follows: if A has a term $A_{i,j} x^i y^j$ that is divisible by $x^{\alpha} y^{\beta}$, then we pick a maximal such term for $\prec_k$ and compute

$$\tilde{A} \;:=\; A - A_{i,j} x^{i-\alpha} y^{j-\beta} B.$$

Then $\deg_k \tilde{A} \leqslant \deg_k A$ and the largest term of $\tilde{A}$ that is divisible by $x^{\alpha} y^{\beta}$ is strictly less than $x^i y^j$ for $\prec_k$. This division step is repeated for $\tilde{A}$ and for its successive reductions, until Q and R are found.

During this naive division process, we note that $Q_{(d-\delta-l, d-\delta]}$ only depends on $A_{(d-l,d]}$ and $B_{(\delta-l,\delta]}$, for $l = 0, \ldots, d - \delta + 1$. When $l = 0$ nothing needs to be computed. Let us now describe a more efficient way to handle the case $l = 1$, when we need to compute the quasi-homogeneous component of Q of maximal k-degree $d - \delta$:

$$A_{[d]} \;=\; Q_{[d-\delta]} B_{[\delta]} + R_{[d]}.$$

LEMMA 3. *We may compute $Q_{[d-\delta]}$ and $R_{[d]}$ using $O(\mathsf{M}(\deg_y A))$ operations in $\mathbb{K}$.*

Proof. We first decompose

$$A_{[d]} = H + T, \qquad H := \sum_{\substack{i+kj=d\\ i\geqslant\alpha}} A_{i,j} x^i y^j, \qquad T := \sum_{\substack{i+kj=d\\ i<\alpha}} A_{i,j} x^i y^j$$

and note that T is reduced with respect to B. In particular, the division of $A_{[d]}$ by $B_{[\delta]}$ yields the same quotient as the division of H by $B_{[\delta]}$, so

$$\begin{aligned} H &= Q_{[d-\delta]} B_{[\delta]} + U \qquad (1)\\ R_{[d]} &= T + U, \end{aligned}$$

for some quasi-homogeneous polynomial U of k-degree d with $\deg_x U < \alpha$. Dehomogenization of the relation (1) yields

$$H(1,y) \;=\; Q_{[d-\delta]}(1,y)\, B_{[\delta]}(1,y) + U(1,y),$$

with $\deg U(1,y) < \beta$. Consequently, the computation of $Q_{[d-\delta]}(1,y)$ and $U(1,y)$ takes $O(\mathsf{M}(\deg_y A))$ operations in $\mathbb{K}$, using a fast algorithm for Euclidean division in $\mathbb{K}[y]$; see [6, chapter 9] or [8], for instance. □

For higher values of l, the following "divide and conquer" division algorithm is more efficient than the naive algorithm:

Algorithm 1

Input. $A, B \in \mathbb{K}[x,y]$ and an integer $l \in \{0,\ldots,d-\delta+1\}$, where $d := \deg_k A$ and $\delta := \deg_k B$.

Output. $\mathrm{quo}_k(A,B)_{(d-\delta-l,d-\delta]}$.

1. If $d<\delta$ or $l=0$ then return 0.
2. If $l=1$ then compute and return $\mathrm{quo}_k(A,B)_{[d-\delta]}$ using the method from Lemma 3.
3. Let $h := \lfloor l/2 \rfloor$.
4. Recursively compute $Q_1 := \mathrm{quo}_k(A,B)_{(d-\delta-h,d-\delta]}$.
5. Let $R_1 := (A_{(d-l,d]} - Q_1 B_{(\delta-l,\delta]})_{(d-l,d-h]}$.
6. Recursively compute $Q_0 := \mathrm{quo}_k(R_1,B)_{(d-\delta-l,d-\delta-h]}$.
7. Return $Q_1 + Q_0$.

PROPOSITION 4. *Algorithm 1 is correct and takes $O(\mathsf{M}(l d/k)\log l)$ operations in $\mathbb{K}$.*

Proof. Let us prove the correctness by induction on l. If $d<\delta$, then $\mathrm{quo}_k(A,B)=0$ and the result of the algorithm is correct. If $l=0$, then $\mathrm{quo}_k(A,B)_{(d-\delta-l,d-\delta]}=0$ and the result is also correct. The case $l=1$ has been treated in Lemma 3.

Now assume that $l\geqslant 2$ and $d\geqslant\delta$, so $l>h\geqslant 1$. The induction hypothesis implies that $(A-Q_1 B)_{(d-h,d]}$ is reduced with respect to B and that $(R_1-Q_0 B)_{(d-l,d-h]}$ is reduced with respect to B. After noting that

$$\begin{aligned} R_1 &= (A_{(d-l,d]} - Q_1 B_{(\delta-l,\delta]})_{(d-l,d-h]} \\ &= (A-Q_1 B)_{(d-l,d-h]}, \end{aligned}$$

we verify that

$$\begin{aligned} (A-(Q_1+Q_0)B)_{(d-l,d]} &= (A-Q_1 B)_{(d-l,d-h]} + (A-Q_1 B)_{(d-h,d]} - (Q_0 B)_{(d-l,d]} \\ &= R_1 - (Q_0 B)_{(d-l,d]} + (A-Q_1 B)_{(d-h,d]} \\ &= R_1 - (Q_0 B)_{(d-l,d-h]} + (A-Q_1 B)_{(d-h,d]} \quad (\text{since } \deg_k(Q_0 B)\leqslant d-h) \\ &= (R_1 - Q_0 B)_{(d-l,d-h]} + (A-Q_1 B)_{(d-h,d]}. \end{aligned}$$

Consequently, $(A-(Q_1+Q_0)B)_{(d-l,d]}$ is reduced with respect to B, whence

$$Q_1+Q_0 = \mathrm{quo}_k(A,B)_{(d-\delta-l,d-\delta]}.$$

This completes the induction and our correctness proof.

Concerning the complexity, step 2 takes $O(\mathsf{M}(\deg_y A)) = O(\mathsf{M}(d/k))$ operations in $\mathbb{K}$, by Lemma 3. In step 5, the computation of R_1 takes $O(\mathsf{M}(l\deg_y A)) = O(\mathsf{M}(l d/k))$ operations in $\mathbb{K}$, by Lemma 2.

Let $\mathsf{T}(\hat{d},\hat{l})$ stand for the maximum of the costs of Algorithm 1 for $d\leqslant\hat{d}$ and $l\leqslant\hat{l}$. We have shown that $\mathsf{T}(d,1)=O(\mathsf{M}(d/k))$ and that

$$\begin{aligned} \mathsf{T}(d,l) &\leqslant \mathsf{T}(d,h)+\mathsf{T}(d-h,l-h)+O(\mathsf{M}(l d/k)) \\ &\leqslant \mathsf{T}(d,h)+\mathsf{T}(d,l-h)+O(\mathsf{M}(l d/k)). \end{aligned}$$

Unrolling this inequality, we obtain the claimed complexity bound. □

COROLLARY 5. *Let $A,B\in\mathbb{K}[x,y]$ be of k-degree $\leqslant d$. The remainder in the division of A by B with respect to $\prec_k$ can be computed using $O(\mathsf{M}(d^2/k)\log d)$ operations in $\mathbb{K}$.*

Proof. By Proposition 4, the computation of $Q := \mathrm{quo}_k(A,B)$ takes $O(\mathsf{M}(d^2/k)\log l)$ operations in $\mathbb{K}$. The computation of the corresponding remainder $A-QB$ takes $O(\mathsf{M}(d^2/k))$ further operations in $\mathbb{K}$ by Lemma 2. □

Remark 6. The complexity bound from Proposition 4 is also a consequence of [9, Theorem 4] by taking $\mathsf{SM}(s) := O(\mathsf{M}(s))$ for the cost of sparse polynomial products of size s. This cost is warranted *mutatis mutandis* by the observation that all sparse bivariate polynomial products occurring within the algorithm underlying [9, Theorem 4] are either univariate products or products of slices of polynomials that are dense with respect to the k-ordering. We have seen in section 2.3 how to compute such products efficiently.

3. GENERAL POSITION

Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n) \in (\mathbb{K}^2)^n$ be a tuple of pairwise distinct points. We define the *vanishing ideal* for $\boldsymbol{\alpha}$ by

$$I_{\boldsymbol{\alpha}} := \{P \in \mathbb{K}[x,y] : P(\boldsymbol{\alpha}) = (0, \ldots, 0)\},$$

where we recall that $P(\boldsymbol{\alpha}) := (P(\alpha_1), \ldots, P(\alpha_n))$.

A monic polynomial $P \in I_{\boldsymbol{\alpha}}$ is said to be *axial* if its leading monomial with respect to $\prec_1$ is of the form y^d. The goal of this section is to prove the existence of such a polynomial modulo a sufficiently generic change of variables of the form

$$x = \tilde{x} + \lambda y. \tag{2}$$

This change of variables transforms $\boldsymbol{\alpha}$ into a new tuple $\tilde{\boldsymbol{\alpha}} \in (\mathbb{K}^2)^n$ with

$$\alpha_i = (x,y) \implies \tilde{\alpha}_i = (x - \lambda y, y) \tag{3}$$

and the ideal $I_{\boldsymbol{\alpha}}$ into

$$I_{\tilde{\boldsymbol{\alpha}}} := \{\tilde{P} \in \mathbb{K}[\tilde{x},y] : \tilde{P}(\tilde{\boldsymbol{\alpha}}) = (0, \ldots, 0)\}.$$

For any degree $d \in \mathbb{N}$, we define

$$\mathbb{K}[x,y]_{\leqslant d} := \{P \in \mathbb{K}[x,y] : \deg_1 P \leqslant d\}.$$

Given a polynomial $P \in \mathbb{K}[x,y]_{\leqslant d}$ such that $\deg_1 P = d$ we may decompose

$$P = D + R,$$

where $D \in \mathbb{K}[x,y]$ is homogeneous of degree d and $R \in \mathbb{K}[x,y]_{\leqslant d-1}$. The change of variables (2) transforms P into

$$\tilde{P}(\tilde{x},y) := \tilde{D}(\tilde{x},y) + \tilde{R}(\tilde{x},y),$$

where

$$\tilde{R}(\tilde{x},y) := R(\tilde{x} + \lambda y, y) \in \mathbb{K}[\tilde{x},y]_{\leqslant d-1},$$
$$\tilde{D}(\tilde{x},y) := D(\tilde{x} + \lambda y, y).$$

The coefficient of the monomial y^d in $\tilde{D}(\tilde{x},y)$ is $\tilde{D}(0,1) = D(\lambda,1)$.

The $\mathbb{K}$-vector space $I_{\boldsymbol{\alpha}} \cap \mathbb{K}[x,y]_{\leqslant d}$ is the solution set of a linear system consisting of n equations and $\binom{d+2}{2}$ unknowns, that are the unknown coefficients of a polynomial in $\mathbb{K}[x,y]_{\leqslant d}$. Such a system admits a non-zero solution whenever $\binom{d+2}{2} > n$. Now assume that P is a non-zero element of $I_{\boldsymbol{\alpha}}$ of minimal total degree d and let $\Lambda_{\boldsymbol{\alpha}}$ denote the set of roots of $D(x,1)$ in $\mathbb{K}$. Since d is minimal we have

$$\binom{d+1}{2} = \frac{d\,(d+1)}{2} \leqslant n.$$

that implies

$$d \leqslant \sqrt{2n}. \tag{4}$$

On the other hand, we have $|\Lambda_{\boldsymbol{\alpha}}| \leqslant d$. And if $\lambda \in \mathbb{K} \setminus \Lambda_{\boldsymbol{\alpha}}$, then y^d is the leading monomial of $\tilde{P}$ for $\prec_1$. Assuming that $|\mathbb{K}| > n$, this proves the existence of an axial polynomial $\tilde{P}$ of degree d after a suitable change of variables of the form (2).

We say that $\boldsymbol{\alpha}$ is in *general position* if there exists a polynomial of minimal degree in $I_{\boldsymbol{\alpha}}$ that is axial.

4. POLYNOMIAL REDUCTION

Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n) \in (\mathbb{K}^2)^n$ be a tuple of points in general position, as defined in the previous section. In this section, we describe a process for reducing a polynomial $P \in \mathbb{K}[x,y]$ modulo $I_{\boldsymbol{\alpha}}$. The reduction of P is a polynomial whose support is controlled in a way that will be made precise.

4.1. Heterogeneous bases

Since $\boldsymbol{\alpha}$ is assumed to be in general position, thanks to (4), we first precompute an axial polynomial $B_0 \in I_{\boldsymbol{\alpha}}$ for $\prec_1$ of degree

$$\delta_0 := \deg_1 B_0 \leqslant \sqrt{2n}.$$

For $i \geqslant 1$, let $N(d, 2^i)$ denote the number of monomials of 2^i-degree $\leqslant d$. We have

$$N(d, 2^i) = d + 1 + (d + 1 - 2^i) + \cdots + \left(d + 1 - \left\lfloor \frac{d}{2^i} \right\rfloor 2^i\right).$$

The $\mathbb{K}$-vector space of the polynomials in $I_{\boldsymbol{\alpha}}$ with 2^i-degree $\leqslant d$ is the solution set of a linear system consisting of n equations and $N(d, 2^i)$ unknowns. Consequently, if $N(d, 2^i) > n$, then there exists a non-zero polynomial in $I_{\boldsymbol{\alpha}}$ of 2^i-degree $\leqslant d$. Let $B_i \in I_{\boldsymbol{\alpha}}$ be a monic polynomial whose leading monomial is minimal for $\prec_{2^i}$ and set

$$\delta_i := \deg_{2^i} B_i.$$

We may precompute B_i, e.g. by extracting it from a Gröbner basis for $I_{\boldsymbol{\alpha}}$ with respect to $\prec_{2^i}$. By the minimality of the 2^i-degree δ_i of B_i, we have

$$N(\delta_i - 1, 2^i) \leqslant n.$$

Now write $\delta_i = q 2^i + r$ with $q \in \mathbb{N}$ and $r \in \{0, \ldots, 2^i - 1\}$. Then

$$\begin{aligned}
N(\delta_i - 1, 2^i) &= (q 2^i + r) + ((q-1) 2^i + r) + \cdots + (2^i + r) + r \\
&= (q + \cdots + 1) 2^i + (q+1) r \\
&= \frac{1}{2} (q+1) q 2^i + (q+1) r \\
&= \frac{1}{2} (q+1) (q 2^i + 2r) \\
&= \frac{1}{2^{i+1}} (q 2^i + 2^i) (q 2^i + 2r) \\
&= \frac{1}{2^{i+1}} (\delta_i + 2^i - r) (\delta_i + r) \\
&\geqslant \frac{1}{2^{i+1}} \delta_i^2.
\end{aligned}$$

Consequently, $\delta_i^2 \leqslant 2^{i+1} n$ and

$$\delta_i \leqslant \sqrt{2^{i+1} n}. \tag{5}$$

We let ℓ be the smallest integer such that $2^\ell > n$, hence

$$2^{\ell - 1} \leqslant n \tag{6}$$

and $\ell := \lceil \log_2(n+1) \rceil$. There exists a monic non-zero polynomial Q in $\mathbb{K}[x] \cap I_{\boldsymbol{\alpha}}$ of minimal degree $\leqslant n$. Since $2^\ell > n$, we may take $B_i := Q$ for $i \geqslant \ell$. We call $(B_i)_{i \geqslant 0}$ a *heterogeneous basis* for $\boldsymbol{\alpha}$. We further define

$$h_i \;:=\; \deg_y B_i \leqslant \left\lfloor \sqrt{2^{1-i} n} \right\rfloor. \tag{7}$$

Note that $i = \ell + 1$ is the first integer such that the upper bound (7) is zero, although $h_i = 0$ holds even for $i = \ell$.

Remark. If the α_i are pairwise distinct and if the cardinality of $\mathbb{K}$ is sufficiently large, then, after a sufficiently generic linear change of coordinates, a Gröbner basis for $I_{\boldsymbol{\alpha}}$ with respect to the lexicographic ordering induced by $y \prec x$ can be computed in softly linear time: see [22, section 6], for instance. Then, a Gröbner basis for $I_{\boldsymbol{\alpha}}$ with respect to $\prec_{2^i}$ can be deduced with $O(n^3)$ operations in $\mathbb{K}$ thanks to the FGLM algorithm [5]. This bound has recently been lowered to $O(n^\omega \log n)$ in [21, Theorem 1.7, Example 1.3], where $\omega > 2$ is a feasible exponent for matrix multiplication.

4.2. Elementary reductions

Given $A = \sum_{\alpha,\beta} A_{\alpha,\beta} x^\alpha y^\beta \in \mathbb{K}[x,y]$ and $i \geqslant 0$, we may use the division procedure from section 2.4 to reduce A with respect to B_i. This yields a relation

$$A \;=\; Q B_i + R, \tag{8}$$

where $\deg_{2^i} R \leqslant \deg_{2^i} A$ and such that none of the monomials in the support of R is divisible by the leading monomial of B_i. We write $\rho_i(A) := R$ and recall that ρ_i is a $\mathbb{K}$-linear map.

We also define the projections π_i and $\bar{\pi}_i$ by

$$\begin{aligned} \pi_i(A) &:= \sum_{\alpha \in \mathbb{N}, \beta \leqslant h_i} A_{\alpha,\beta} x^\alpha y^\beta \\ \bar{\pi}_i(A) &:= \sum_{\alpha \in \mathbb{N}, \beta > h_i} A_{\alpha,\beta} x^\alpha y^\beta. \end{aligned}$$

4.3. Compound reductions

For $d \geqslant 1$, we let $\mathbb{K}[x,y]_d^*$ denote the set of tuples of polynomials $(P_0, \ldots, P_m) \in \mathbb{K}[x,y]^{m+1}$ such that

- m is the first integer such that $2^m > d$,
- $\deg_{2^i} P_i \leqslant \sqrt{2^i d}$, for $i = 0, \ldots, m$.

Intuitively speaking, such a tuple will represent a sum $P = P_0 + \cdots + P_m$ modulo $I_{\boldsymbol{\alpha}}$. Note that $\deg_y P_m \leqslant \lfloor \sqrt{2^{-m} d} \rfloor = 0$, so $P_m \in \mathbb{K}[x]$.

Given $(P_0, \ldots, P_m) \in \mathbb{K}[x,y]^{m+1}$, we define three new sequences of polynomials by

$$\begin{array}{rclrclrcl} R_0 &:=& \rho_0(P_0) & \Pi_0 &:=& \pi_0(R_0) & \bar{\Pi}_0 &:=& \bar{\pi}_0(R_0) \\ R_1 &:=& \rho_1(P_1 + \Pi_0) & \Pi_1 &:=& \pi_1(R_1) & \bar{\Pi}_1 &:=& \bar{\pi}_1(R_1) \\ R_2 &:=& \rho_2(P_2 + \Pi_1) & \Pi_2 &:=& \pi_2(R_2) & \bar{\Pi}_2 &:=& \bar{\pi}_2(R_2) \\ \vdots & & & \vdots & & & \vdots & & \\ R_{m-1} &:=& \rho_{m-1}(P_{m-1} + \Pi_{m-2}) & \Pi_{m-1} &:=& \pi_{m-1}(R_{m-1}) & \bar{\Pi}_{m-1} &:=& \bar{\pi}_{m-1}(R_{m-1}) \\ R_m &:=& \rho_m(P_m + \Pi_{m-1}). & & & & & & \end{array}$$

LEMMA 7. *With the above notations. Let* $\beta \geqslant 16$ *and let* $\eta := \left(1 + \sqrt{\frac{2}{\beta}}\right)^2$. *If* $d \geqslant \beta n$, *then*

$$\begin{aligned} \deg_{2^i} R_i &\leqslant \sqrt{2^i d} \\ \deg_{2^{i+1}} \Pi_i &\leqslant \sqrt{2^i \eta d} \end{aligned}$$

for $i = 0, \ldots, m-1$.

Proof. We first note that $\eta \leqslant 2$ and

$$\sqrt{2^i d} + \sqrt{2^{i+1} n} \leqslant \sqrt{2^i \eta d}. \tag{9}$$

Let us prove the degree bounds by induction on $i = 0, \ldots, m-1$. For $i = 0$, we have

$$\begin{aligned} \deg_1 R_0 &\leqslant \deg_1 P_0 \leqslant \sqrt{d} \\ \deg_2 \Pi_0 &\leqslant \deg_1 R_0 + \deg_y \Pi_0 \leqslant \deg_1 R_0 + h_0 \leqslant \sqrt{d} + \sqrt{2n} \leqslant \sqrt{\eta d}, \end{aligned}$$

by using (7) and (9). Assume now that $0 < i \leqslant m-1$ and that the bounds of the lemma hold for all smaller i. Since $\eta \leqslant 2$, the induction hypothesis yields

$$\deg_{2^i} R_i \leqslant \max(\deg_{2^i} P_i, \deg_{2^i} \Pi_{i-1}) \leqslant \sqrt{2^i d}.$$

Using (7) and (9) again, we deduce

$$\deg_{2^{i+1}} \Pi_i \leqslant \deg_{2^i} R_i + 2^i \deg_y \Pi_i \leqslant \deg_{2^i} R_i + h_i 2^i \leqslant \sqrt{2^i d} + \sqrt{2^{i+1} n} \leqslant \sqrt{2^i \eta d}. \quad \square$$

LEMMA 8. *Under the assumptions of Lemma 7, we further have*

$$\bar{\Pi}_1 + \cdots + \bar{\Pi}_{m-1} + R_m = P_0 + \cdots + P_m \mod I_{\boldsymbol{\alpha}}. \tag{10}$$

Assume that $\theta := \frac{1}{\sqrt{2}} + \frac{2}{\sqrt{\beta}} < 1$, *let* $\tilde{d} := \theta^2 d$, *and let* $\tilde{m}$ *be the first integer such that* $2^{\tilde{m}} > \tilde{d}$. *If* $\tilde{m} = m$, *then*

$$(\bar{\Pi}_1, \ldots, \bar{\Pi}_{m-1}, R_m, 0) \in \mathbb{K}[x,y]^*_{\tilde{d}};$$

otherwise, $\tilde{m} = m-1$ *and*

$$(\bar{\Pi}_1, \ldots, \bar{\Pi}_{m-1}, R_m) \in \mathbb{K}[x,y]^*_{\tilde{d}}.$$

Proof. By construction, we have

$$R_0 + \cdots + R_m = P_0 + \cdots + P_m + \Pi_0 + \cdots + \Pi_{m-1} \mod I_{\boldsymbol{\alpha}},$$

whence

$$\begin{aligned} \bar{\Pi}_0 + \cdots + \bar{\Pi}_{m-1} + R_m &= (R_0 - \Pi_0) + \cdots + (R_{m-1} - \Pi_{m-1}) + R_m \\ &= P_0 + \cdots + P_m \mod I_{\boldsymbol{\alpha}}. \end{aligned}$$

Since B_0 is axial, we have $\bar{\Pi}_0 = 0$, which entails (10). From Lemma 7 we deduce

$$\deg_y \bar{\Pi}_i \leqslant \frac{\deg_{2^i} \bar{\Pi}_i}{2^i} \leqslant \frac{\deg_{2^i} R_i}{2^i} \leqslant \frac{\sqrt{2^i d}}{2^i} = \sqrt{2^{-i} d}. \tag{11}$$

Now R_i contains no monomial that is divisible by the leading monomial of B_i for $\prec_{2^i}$ and $\deg_y \bar{\Pi}_i > \deg_y B_i$. Using (5), it follows that

$$\deg_x \bar{\Pi}_i \leqslant \deg_x B_i \leqslant \delta_i \leqslant \sqrt{2^{i+1} n}. \tag{12}$$

Consequently, for $i=1,\ldots,m-1$, inequalities (11) and (12), combined with $d\geqslant\beta n$, lead to

$$\begin{aligned}\deg_{2^{i-1}}\bar{\Pi}_i &\leqslant \deg_x\bar{\Pi}_i+2^{i-1}\deg_y\bar{\Pi}_i\\ &\leqslant \sqrt{2^{i+1}n}+\sqrt{2^{i-2}d}\\ &\leqslant \theta\sqrt{2^{i-1}d}=\sqrt{2^{i-1}\tilde{d}}.\end{aligned}$$

From $d\geqslant\beta n\geqslant 16n$ and (6), we deduce that $d\geqslant 2^4\times 2^{\ell-1}=2^{\ell+3}$, whence $m\geqslant\ell+4$. If follows that

$$\deg_y\Pi_{m-1}\leqslant h_{m-1}\leqslant h_{\ell+3}=0.$$

From $\deg_y P_m=0$, we thus obtain $\deg_y R_m=0$. Since $m\geqslant\ell+4$, the polynomial B_m belongs to $\mathbb{K}[x]$ and has degree $\leqslant n$. It follows that $\deg_x R_m<\deg_x B_m\leqslant n$.

Since $n\leqslant d/16$ and $d<2^m$, we further deduce

$$\begin{aligned}\deg_{2^{m-1}}R_m &= \deg_x R_m\\ &< \sqrt{2^{-8}d^2}\\ &\leqslant \sqrt{2^{-7}\times 2^{m-1}d}\\ &\leqslant \theta\sqrt{2^{m-1}d}=\sqrt{2^{m-1}\tilde{d}}.\end{aligned}$$

Finally $d>\tilde{d}\geqslant d/2$ and $2^{m-1}\leqslant d<2^m$ imply $2^{m-2}\leqslant\tilde{d}<d$, whence $\tilde{m}\in\{m-1,m\}$. □

We call the tuple $(\bar{\Pi}_1,\ldots,\bar{\Pi}_{m-1},R_m)$ the *reduction* of $(P_0,\ldots,P_m)$ with respect to $(B_i)_{i\geqslant 0}$.

LEMMA 9. *With the notation and assumptions of Lemma 7, the reduction of* $(P_0,\ldots,P_m)$ *with respect to* $(B_i)_{i\geqslant 0}$ *takes* $O(\mathsf{M}(d)\log^2 d)$ *operations in* $\mathbb{K}$.

Proof. First note that $(\deg_1 P_0)^2\leqslant d$ and $(\deg_1 B_0)^2=O(d)$. By Corollary 5, the computation of $\rho_0(P_0)$ therefore takes $O(\mathsf{M}(d)\log d)$ operations in $\mathbb{K}$. For $i=1,\ldots,m$, Lemma 7 and (5) imply

$$\begin{aligned}\deg_{2^i}(P_i+\Pi_{i-1}) &\leqslant \sqrt{2^i d}\\ \deg_{2^i}B_i &\leqslant \sqrt{2^{i+1}n}.\end{aligned}$$

By Corollary 5, we deduce that the computation of $\rho_i(P_i+\Pi_{i-1})$ takes

$$O\Big(\mathsf{M}\big(\big(\sqrt{2^i d}\big)^2/2^i\big)\log d\Big) = O(\mathsf{M}(d)\log d)$$

operations in $\mathbb{K}$. Summing the costs to compute R_i for $i=0,\ldots,m=O(\log d)$, we obtain the claimed bound. □

PROPOSITION 10. *Let* $(P_i)_{i\geqslant 0}\in\mathbb{K}[x,y]^*_{128n}$. *Then a sequence* $(Q_i)_{i\geqslant 0}\in\mathbb{K}[x,y]^*_{64n}$ *with* $\sum_{i\geqslant 0}Q_i=\sum_{i\geqslant 0}P_i \bmod I_{\boldsymbol{\alpha}}$ *can be computed using* $O(\mathsf{M}(n)\log^2 n)$ *operations in* $\mathbb{K}$.

Proof. We set $\beta:=64$, so that $\theta=\frac{1}{\sqrt{2}}+\frac{2}{\sqrt{\beta}}<0.958$. We set $(P_i^{[0]})_{i\geqslant 0}:=(P_i)_{i\geqslant 0}$ and define $(P_i^{[k+1]})_{i\geqslant 0}$ recursively as the reduction of $(P_i^{[k]})_{i\geqslant 0}$ with respect to $(B_i)_{i\geqslant 0}$, for $k\geqslant 0$. The integer $K:=8$ is the first integer such that

$$\theta^{2K}\leqslant\frac{1}{2}.$$

We finally take $(Q_i)_{i\geqslant 0}:=(P_i^{[K]})_{i\geqslant 0}$. Lemma 8 implies that $(Q_i)_{i\geqslant 0}$ belongs to $\mathbb{K}[x,y]^*_{64n}$. The complexity bound follows from Lemma 9. □

5. MULTI-POINT EVALUATION

Let $\boldsymbol{\alpha} \in (\mathbb{K}^2)^n$ be a tuple of pairwise distinct points and consider the problem of fast multi-point evaluation of a polynomial $P \in \mathbb{K}[x,y]$ of total degree $<\sqrt{2n}$ at $\boldsymbol{\alpha}$. For simplicity of the exposition, it is convenient to first consider the case when $n = 2^\nu$ is a power of two and $\boldsymbol{\alpha}$ is in general position. The core of our method is based on the usual "divide and conquer" paradigm.

We say that $\boldsymbol{\alpha}$ is in *recursive general position* if $\boldsymbol{\alpha}$ is in general position and if $\boldsymbol{\alpha}_{1,n/2} := (\alpha_1, \ldots, \alpha_{n/2})$ and $\boldsymbol{\alpha}_{n/2+1,n} := (\alpha_{n/2+1}, \ldots, \alpha_n)$ are in recursive general position. An empty sequence is in recursive general position. With the notation of section 3 a recursive general position is ensured if the cardinality of $\mathbb{K}$ is strictly larger that λ_n that is recursively defined by

$$\lambda_n := n + 2\lambda_{n/2}$$

for $n \geqslant 4$ and with $\lambda_2 := 2$. Consequently $\lambda_n = n\,\nu$. Whenever $|\mathbb{K}| > n\,\nu$, we know from section 3 that we can reduce to the case where $\boldsymbol{\alpha}$ is in recursive general position. In this case, we can compute a *recursive heterogeneous basis*, that is made of a heterogeneous basis for $\boldsymbol{\alpha}$ and recursive heterogeneous bases for $\boldsymbol{\alpha}_{1,n/2}$ and $\boldsymbol{\alpha}_{n/2+1,n}$.

Algorithm 2

Input. $\boldsymbol{\alpha} \in (\mathbb{K}^2)^n$ with $n = 2^\nu$ and $(P_i)_{i \geqslant 0} \in \mathbb{K}[x,y]^*_{128n}$.

Output. $(\sum_{i \geqslant 0} P_i)(\boldsymbol{\alpha})$.

Precomputed. A recursive heterogeneous basis for $\boldsymbol{\alpha}$.

Assumption. $\boldsymbol{\alpha}$ is in recursive general position.

1. If $\nu = 0$, then return $(\sum_{i \geqslant 0} P_i)(\alpha_1)$.
2. Compute the reduction $(Q_i)_{i \geqslant 0} \in \mathbb{K}[x,y]^*_{64n}$ of $(P_i)_{i \geqslant 0}$ with respect to the heterogeneous basis $(B_i)_{i \geqslant 0}$ for $\boldsymbol{\alpha}$, via Proposition 10.
3. Recursively apply the algorithm to $\boldsymbol{\alpha}_{1,n/2}$ and $(Q_i)_{i \geqslant 0}$.
4. Recursively apply the algorithm to $\boldsymbol{\alpha}_{n/2+1,n}$ and $(Q_i)_{i \geqslant 0}$.
5. Return the concatenations of the results of the recursive evaluations.

THEOREM 11. *Algorithm 2 is correct and runs in time $O(\mathsf{M}(n) \log^3 n)$.*

Proof. The algorithm is clearly correct if $\nu = 0$. If $\nu > 0$, then we first observe that both $\boldsymbol{\alpha}_{1,n/2}$ and $\boldsymbol{\alpha}_{n/2+1,n}$ are in recursive general position by definition. Furthermore, Proposition 10 ensures that

$$\left(\sum_{i \geqslant 0} P_i\right)(\boldsymbol{\alpha}) \;=\; \left(\sum_{i \geqslant 0} Q_i\right)(\boldsymbol{\alpha}).$$

The concatenation of the results of the recursive applications of the algorithm therefore yields the correct result.

As to the complexity bound, the cost of step 2 is bounded by $O(\mathsf{M}(n) \log^2 n)$ according to Proposition 10. Hence, the total execution time $\mathsf{T}(n)$ satisfies

$$\mathsf{T}(n) \;\leqslant\; 2\,\mathsf{T}\!\left(\tfrac{n}{2}\right) + O(\mathsf{M}(n) \log^2 n).$$

The desired complexity bound follows by unrolling this recurrence inequality. □

COROLLARY 12. *Consider an arbitrary effective field* $\mathbb{K}$ *and* $\boldsymbol{\alpha} \in (\mathbb{K}^2)^n$, *where n is not necessarily a power of two. Modulo precomputations that only depend on* $\mathbb{K}$ *and* $\boldsymbol{\alpha}$, *we can evaluate any polynomial in* $\mathbb{K}[x,y]_{\leqslant \sqrt{2n}}$ *at* $\boldsymbol{\alpha}$ *in time* $O(\mathsf{M}(n \log n) \log^3 n)$.

Proof. Modulo the repetition of at most $n-1$ more points, we may assume without loss of generality that n is a power of two 2^ν.

If $\mathbb{K}$ is finite then we build an algebraic extension $\mathbb{E}$ of $\mathbb{K}$ of degree $e := O(\log n)$, so that $|\mathbb{E}| > n\nu$. Multiplying two polynomials in $\mathbb{E}[x]_{\leqslant n}$ takes

$$O(\mathsf{M}(e n)) = O(\mathsf{M}(n \log n))$$

operations in $\mathbb{K}$. Consequently, up to introducing an extra $\log n$ factor in our complexity bounds, we may assume that $|\mathbb{K}| > n\nu$.

Modulo a change of variables (2) from section 3, we may then assume that $\tilde{\boldsymbol{\alpha}}$, defined in (3), is in recursive general position, and compute a recursive heterogeneous basis.

Given $P \in \mathbb{K}[x,y]_{\leqslant \sqrt{2n}}$, we claim that we may compute $\tilde{P}(\tilde{x}, y) := P(\tilde{x} + \lambda y, y)$ using $O(\mathsf{M}(n \log n) \log n)$ operations in $\mathbb{K}$. Indeed, we first decompose

$$P(x,y) = p_0(x,y) + \cdots + p_m(x,y),$$

where $m \leqslant \sqrt{2n}$ and each p_i is zero or homogenous of degree i. Computing $p_i(\tilde{x} + \lambda y, y)$ then reduces to computing $p_i(\tilde{x} + \lambda, 1)$. This corresponds in turn to a univariate Taylor shift, which takes $O(\mathsf{M}(i \log n) \log i)$ operations in $\mathbb{K}$; see [1, Lemma 7], for instance. Finally, we apply Theorem 11 to $(\tilde{P}, 0, \ldots, 0) \in \mathbb{K}[\tilde{x}, y]^*_{128n}$ and $\tilde{\boldsymbol{\alpha}}$. □

6. CONCLUSION

We have designed a softly linear time algorithm for bivariate multi-point evaluation, modulo precomputations that only depend on the evaluation points. This result raises several new questions. First, it would be useful to optimize the constant factors involved in the cost analysis, and study the efficiency of practical implementations. Second, one may investigate extensions of Corollary 12 that take into account the partial degrees of the input polynomial, e.g. by applying Theorem 11 to inputs of the form $(0, \ldots, 0, \tilde{P}, 0, \ldots, 0)$. A final challenge concerns the possibility to perform all precomputations in subquadratic time. For this, one might use techniques from [2, 19], as in [12]. The problem of achieving an almost linear cost appears to be even harder.

Acknowledgments. We thank the anonymous referees for their useful comments.

BIBLIOGRAPHY

[1] J. Berthomieu, G. Lecerf, and G. Quintin. Polynomial root finding over local rings and application to error correcting codes. *Appl. Alg. Eng. Comm. Comp.*, 24(6):413–443, 2013.

[2] A. Bostan, C.-P. Jeannerod, and É. Schost. Solving structured linear systems with large displacement rank. *Theor. Comput. Sci.*, 407(1):155–181, 2008.

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

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

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

[6] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2nd edition, 2002.

[7] D. Harvey and J. van der Hoeven. Faster polynomial multiplication over finite fields using cyclotomic coefficient rings. *J. Complexity*, 54:101404, 2019.

[8] J. van der Hoeven. Newton's method and FFT trading. *J. Symbolic Comput.*, 45(8):857–878, 2010.

[9] J. van der Hoeven. On the complexity of multivariate polynomial division. In I. S. Kotsireas and E. Martínez-Moro, editors, *Applications of Computer Algebra. Kalamata, Greece, July 20–23, 2015*, volume 198 of *Springer Proceedings in Mathematics & Statistics*, pages 447–458. Cham, 2017. Springer International Publishing.

[10] J. van der Hoeven. *The Jolly Writer. Your Guide to GNU TeXmacs*. Scypress, 2020.

[11] J. van der Hoeven and R. Larrieu. Fast reduction of bivariate polynomials with respect to sufficiently regular Gröbner bases. In C. Arreche, editor, *Proceedings of the 2018 ACM International Symposium on Symbolic and Algebraic Computation*, ISSAC '18, pages 199–206. New York, NY, USA, 2018. ACM.

[12] J. van der Hoeven and G. Lecerf. Fast amortized multi-point evaluation. Technical Report, HAL, 2020. `https://hal.archives-ouvertes.fr/hal-02508529`.

[13] J. van der Hoeven and G. Lecerf. Fast multivariate multi-point evaluation revisited. *J. Complexity*, 56:101405, 2020.

[14] J. van der Hoeven and G. Lecerf. On the complexity exponent of polynomial system solving. *Found. Comput. Math.*, 21:1–57, 2021.

[15] J. van der Hoeven and É. Schost. Multi-point evaluation in higher dimensions. *Appl. Alg. Eng. Comm. Comp.*, 24(1):37–52, 2013.

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

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

[18] D. Le Brigand and J.-J. Risler. Algorithme de Brill–Noether et codes de Goppa. *Bulletin de la société mathématique de France*, 116(2):231–253, 1988.

[19] V. 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*, ISSAC '16, pages 365–372. New York, NY, USA, 2016. ACM.

[20] V. Neiger, J. Rosenkilde, and G. Solomatov. Generic bivariate multi-point evaluation, interpolation and modular composition with precomputation. In A. Mantzaflaris, editor, *Proceedings of the 45th International Symposium on Symbolic and Algebraic Computation*, ISSAC '20, pages 388–395. New York, NY, USA, 2020. ACM.

[21] V. Neiger and É. Schost. Computing syzygies in finite dimension using fast linear algebra. *J. Complexity*, 60:101502, 2020.

[22] M. Nüsken and M. Ziegler. Fast multipoint evaluation of bivariate polynomials. In S. Albers and T. Radzik, editors, *Algorithms – ESA 2004. 12th Annual European Symposium, Bergen, Norway, September 14-17, 2004*, volume 3221 of *Lect. Notes Comput. Sci.*, pages 544–555. Springer Berlin Heidelberg, 2004.

[23] 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.

[24] L. Robbiano. Term orderings on the polynomial ring. In B. F. Caviness, editor, *EUROCAL '85. European Conference on Computer Algebra. Linz, Austria, April 1-3, 1985. Proceedings. Volume 2: Research Contributions*, volume 204 of *Lect. Notes Comput. Sci.*, pages 513–517. Springer-Verlag Berlin Heidelberg, 1985.