Generic bivariate multi-point evaluation, interpolation and modular composition with precomputation

Vincent Neiger, Johan Rosenkilde, Grigory Solomatov

Generic bivariate multi-point evaluation, interpolation and modular composition with precomputation

Vincent Neiger
Univ. Limoges, CNRS, XLIM, UMR 7252
F-87000 Limoges, France

Johan Rosenkilde
Technical University of Denmark
Lyngby, Denmark

Grigory Solomatov
Technical University of Denmark
Lyngby, Denmark

ABSTRACT

If $\mathbb{K}$ is a large enough field and $\mathcal{P} \subset \mathbb{K}^2$ is a fixed, generic set of points, which is available for precomputation, we show how to compute all the evaluations of any dense polynomial f on $\mathcal{P}$ in quasi-linear time. Similarly, in quasi-linear time then given interpolation constraints on $\mathcal{P}$ and a target y-degree, we compute an f having those evaluations on $\mathcal{P}$ and at most that y-degree. Our genericity assumption is explicit and we prove most point sets over a large enough field satisfy it. If $\mathcal{P}$ violates the assumption our algorithms still work and the performance degrades smoothly according to a distance from being generic.

We apply the same technique to modular composition: fix a square-free $G \in \mathbb{K}[x]$ and generic $R \in \mathbb{K}[x]$ both available for precomputation, we then input $f \in \mathbb{K}[x, y]$ and output $f(x, R(x)) \operatorname{rem} G \in \mathbb{K}[x]$ in quasi-linear time in the size of f, G, R.

KEYWORDS

Multi-point evaluation, interpolation, modular composition, bivariate polynomials.

1 INTRODUCTION

Let $\mathbb{K}$ be an effective field. We consider the three classical problems for bivariate polynomials $\mathbb{K}[x, y]$ mentioned in the title. We assume a model where parts of the input are given early as *preinput* which is available for heavier computation, and the primary goal is to keep the complexity of the *online phase*, once the remaining input is given, to a minimum.

Multi-point evaluation (MPE): with preinput a point set $\mathcal{P} = \{(\alpha_i, \beta_i)\}_{i=1}^n \subseteq \mathbb{K}^2$ and input $f \in \mathbb{K}[x, y]$, compute $\big(f(\alpha_i, \beta_i)\big)_{i=1}^n$. We give two algorithms: the first requires that $\mathcal{P}$ has distinct x-coordinates and has online complexity $\tilde{O}(\deg_x f \deg_y f + n)$ as long as $\mathcal{P}$ is *balanced*, a notion we define in the paper[1]; the second accepts repeated x-coordinates with online complexity $\tilde{O}(\deg_x f(\deg_x f + \deg_y f) + n)$ as long as a certain "shearing" of $\mathcal{P}$ is balanced.

Interpolation: with preinput a point set $\mathcal{P}$ as before, and input interpolation constraints $\boldsymbol{\gamma} \in \mathbb{K}^n$, compute $f \in \mathbb{K}[x, y]$ such that $\big(f(\alpha_i, \beta_i)\big)_{i=1}^n = \boldsymbol{\gamma}$, satisfying some constraints on the monomial support. We give an algorithm which preinputs a degree bound $d_y \in \mathbb{Z}_{>0}$ and such that the output polynomial f is dense with $\deg_y f < d_y$ and $\deg_x f \in O(n/d_y)$. The online complexity is $\tilde{O}(n)$ assuming that $\mathcal{P}$ is balanced. d_y should exceed the *x-valency* of $\mathcal{P}$, i.e. the maximal number of y-coordinates for any given x-coordinate.

Modular composition: with preinput $G, R \in \mathbb{K}[x]$, we input $f \in \mathbb{K}[x, y]$ and compute $f(x, R) \operatorname{rem} G$. Our algorithm has online complexity $\tilde{O}(\deg_x f \deg_y f + \deg R + \deg G)$, as long as the ideal $\langle G, y - R\rangle$ is balanced.

We prove that if $\mathcal{P} \subseteq \mathbb{K}^2$ is random of fixed cardinality n, and if $|\mathbb{K}| \gg n^2 \log(n)$ then $\mathcal{P}$ is balanced with high probability[2]. Similarly, if G is square-free and R is uniformly random, then $\langle G, y - R\rangle$ is balanced with high probability. For our second MPE algorithm, we shear the point set, see below. Since our current genericity techniques do not extend to this, we do not make claims on it being generically balanced. Ad-hoc simulations indicate that this is be the case unless the x-valency of $\mathcal{P}$ is very high. The cost of the second MPE algorithm is not symmetric in the x and y-degree, so whenever $\deg_x f < \deg_y f$ one should consider transposing the input, i.e. MPE of $f(Y, X)$ on $\mathcal{P}^\top = \{(\beta_i, \alpha_i)\}_{i=1}^n$. In this case, the balanced assumption is on $\mathcal{P}^\top$.

Our algorithms are deterministic, and once the preinput has been processed, the user knows whether it is balanced and hence whether the algorithms will perform well once the input arrives. Further, the performance of our algorithms deteriorate smoothly with how "unbalanced" the input is. In a toolbox one might therefore apply our algorithms whenever the input turns out to be sufficiently balanced and reverting to other algorithms on very unbalanced input.

Precomputation can be reasonable if we e.g. compute MPE's on the same point set for many different polynomials. This occurs in coding theory, where MPE of bivariate polynomials corresponds to "encoding" of certain algebraic codes such as some Reed–Muller codes [1, Chapter 5] and some algebraic-geometric codes [13]: here $\mathcal{P}$ is fixed and communication commences by a (very long) series of MPE's on bivariate polynomials on $\mathcal{P}$. In these applications, $\mathcal{P}$ will often not be random, but chosen carefully, and so our genericity assumptions might not apply.

Techniques. We introduce a tool we call *reshaping* of polynomials for achieving the following: given an ideal $I \subseteq \mathbb{K}[x, y]$ and $f \in \mathbb{K}[x, y]$, compute $\hat{f} \in f + I$ with smaller y-degree. For instance, in MPE and when $\mathcal{P}$ has distinct x-coordinates, we let $\Gamma \subset \mathbb{K}[x, y]$ be the ideal of polynomials which vanish on all the points $\mathcal{P}$. Then all elements of $f + \Gamma$ have the same evaluations on $\mathcal{P}$, so we compute a $\hat{f} \in f + \Gamma$ of y-degree 0, and then apply fast univariate MPE.

An obvious idea to accomplish this is to choose some $g \in \Gamma$ of lower y-degree and whose leading y-term is monic, and then compute $\tilde{f} = f \operatorname{rem} g$. The problem is to control the x-degree of $\tilde{f}$, which generically grows by $(\deg_y f - \deg_y g) \deg_x g$. Our idea is to look for polynomials g that we call *reshapers*, which have the form

$$g = y^{2d_y/3} - \hat{g}\,,$$

where $\deg_y \hat{g} < d_y/3$, and where $d_y = \deg_y f + 1$ (and divisible by 3 in this example). Writing $f = f_1 y^{2d_y/3} + f_0$ with $\deg_y f_0 < 2d_y/3$,

[1] "*soft-O*" ignores logarithmic terms: $O(f(n)(\log f(n))^c) \subset \tilde{O}(f(n))$ for any $c \in \mathbb{Z}_{>0}$.

[2] If $\mathbb{K}$ is infinite, we consider $\mathcal{P} \subseteq \mathcal{T}^2$ for some finite $\mathcal{T} \subset \mathbb{K}$.

we see that

$$\tilde{f} := f \operatorname{rem} g = f_1 \hat{g} + f_0 \,,$$

which is easy to compute and has y-degree less than $2d_y/3$ and x-degree only $\deg_x f + \deg_x g$. To reduce the y-degree down to 0, as in MPE, we repeat the process logarithmically many times.

For efficiency, we therefore need the x-degrees of all these reshapers g to be small. For MPE, stating that $g \in \Gamma$ specifies n homogeneous linear restrictions on the coefficients of $\hat{g}$, so generically we could expect that $\approx n$ monomials suffice to solve for g; since $\deg_y \hat{g} < d_y/3$ we might expect that $\deg_x g_i \approx 3n/d_y$ suffice. Informally, by $\mathcal{P}$ being "balanced" we mean that all the g-polynomials we need for reshaping will satisfy this "expected" degree constraint.

To handle point sets with repeated x-coordinates, we shear the points by $(\alpha, \beta) \mapsto (\alpha + \theta\beta, \beta)$, where $\theta \in \mathbb{L} \setminus \mathbb{K}$ and $\mathbb{L} : \mathbb{K}$ is a field extension of degree 2. The resulting point set now has distinct x-coordinates. This replaces $f(x, y)$ with $f(x - \theta y, y)$, and whenever $\deg_x f < \deg_y f$ we stay within quasi-linear complexity if the sheared point set is balanced.

Previous work. Quasi-linear complexity has been achieved for multivariate MPE and interpolation on special point sets and monomial support: Pan [17] gave an algorithm on grids, and van der Hoeven and Schost [25] (see also [5, Sec. 2]) generalised this to certain types of subsets of grids, constraining both the points and the monomial support. See [25] for earlier work on interpolation, achieving worse than quasi-linear complexity.

In classical univariate modular composition, we are given f, G, R in $\mathbb{K}[x]$ and seek $f(R) \operatorname{rem} G$. Brent and Kung's baby-step giant-step algorithm [3, 18] performs this operation in $\tilde{O}(n^{(\omega+1)/2})$, where ω is the matrix multiplication exponent with best known bound $\omega < 2.37286$ [12]. Nüsken and Ziegler [16] extended this to a bivariate input $f \in \mathbb{K}[x, y]$: they showed how to compute $f(x, R) \operatorname{rem} G$ in complexity $O(\deg_x f (\deg_y f)^{(\omega+1)/2})$, assuming that both $\deg_x R$ and $\deg_x G$ are at most $\deg_x f \deg_y f$. They applied this result to solve MPE in the same cost; in this paper, we use essentially the same link between these problems. To the best of our knowledge, this is currently the best known cost bound for these problems, in the algebraic complexity model.

In a breakthough result, Kedlaya and Umans [10] achieved "almost linear" time for modular composition and MPE, for specific types of fields $\mathbb{K}$ and in the bit complexity model. For univariate modular composition, the cost is $O(n^{1+\epsilon})$ bit operations for any $\epsilon > 0$, while for MPE it is $O((n + (\deg_x f)^2)^{1+\epsilon})$, assuming $\deg_y f < \deg_x f$ (the algorithm also supports multivariate MPE). Unfortunately, these algorithms have so far resisted attempts at a practical implementation [24].

Our quasi-linear online complexities improve on the above results (including Kedlaya and Umans' ones since quasi-linear compares favorably to almost linear). However we stress that none of the above algorithms have the two constraints of our work: allowing precomputation on $\mathcal{P}$, and genericity of $\mathcal{P}$. For modular composition, allowing precomputation on G was proposed in [23] to leverage the factorisation structure of G. Except for slight benefits of precomputation in the context of Brent and Kung's modular composition algorithm (used e.g. in the Flint and NTL libraries [7, 21]), we are unaware of other work focusing on the use of precomputation for MPE, Interpolation, and Modular Composition.

Genericity has recently been used for such problems by Villard [26], who showed how to efficiently compute the resultant of two generic bivariate polynomials $f, g \in \mathbb{K}[x, y]$; a particular case is the computation of the characteristic polynomial of $G \in \mathbb{K}[x]$ in the quotient $\mathbb{K}[x]/\langle G \rangle$ for given R and G in $\mathbb{K}[x]$, with direct links to the univariate composition $f(R) \operatorname{rem} G$ [26, 27]. This led to an ongoing work on achieving modular composition with exponent $(\omega + 2)/3$ [14], thus improving upon Brent and Kung's result but not reaching quasi-linear complexity (even for $\omega = 2$). In that line of work, the main benefit from genericity is that the ideal $\langle G, y - R \rangle$ admits bases formed by m polynomials of y-degree $< m$ and x-degree at most $\deg(G)/m$, for a given parameter $2 \le m \le \deg(G)$. Such a basis is represented as an $m \times m$ matrix over $\mathbb{K}[x]$ with all entries of degree at most $\deg(G)/m$, and one can then rely on fast univariate polynomial matrix algorithms. One of the main contributions of [26] is to show how to compute such a basis efficiently.

In this paper, genericity serves a purpose similar to that in [14, 26]: it ensures the existence of such bases for several parameters m, and also of the reshapers g mentioned above; besides we make use of these bases to precompute these reshapers. Note that, these objects being only used in the precomputation stage, the speed of computing them is not a main concern in this paper. Once the reshapers are known, our algorithms work without requiring any other genericity property.

Organisation. After some preliminaries in Section 2, we describe the reshaping strategy for an arbitrary ideal in Section 3. Then Sections 4 to 6 give algorithms for each of the three problems. We discuss precomputation in Section 7 and genericity in Section 8.

2 PRELIMINARIES

For complexity estimates, we use the algebraic RAM model and count arithmetic operations in $\mathbb{K}$. By $\mathsf{M}(n)$ we denote the cost of multiplying two univariate polynomials over $\mathbb{K}$ of degree at most n. We may take $\mathsf{M}(n) \in O(n \log n \log\log n) \subset \tilde{O}(n)$ [4], or the slightly faster [8]. Univariate division with remainder, QUO_REM, has similar cost $O(\mathsf{M}(n))$, see e.g. [28, Theorem 9.6]. When degrees of a polynomial, say $f \in \mathbb{K}[x]$, appears in a complexity estimates, we will abuse notation and denote by $\deg_x f$ the quantity $\max(\deg_x f, 1)$. This shorthand makes e.g. the expression $O(\deg_x g \deg_y g)$ denote the time for e.g. scanning all coefficients of some $g \in \mathbb{K}[x, y]$.

The following two results on univariate polynomials are well-known, see e.g. [28, Corollaries 10.8 and 10.12]: firstly, given $f \in \mathbb{K}[x]$ and $\alpha_1, \ldots, \alpha_n \in \mathbb{K}$, we may compute $\big(f(\alpha_i)\big)_{i=1}^n$ in time $O(\mathsf{M}(\deg_x f + n) \log n) \subseteq \tilde{O}(\deg_x f + n)$. Secondly, given $\alpha_1, \ldots, \alpha_n$, $\beta_1, \ldots, \beta_n \in \mathbb{K}$, we may compute the unique $f \in \mathbb{K}[x]$ with $\deg_x f < n$ and $f(\alpha_i) = \beta_i$ for $i = 1, \ldots, n$ in time $O(\mathsf{M}(n) \log n) \subseteq \tilde{O}(n)$. We will also use the fact that given $f, g \in \mathbb{K}[x, y]$, we may compute the product fg in time $O(\mathsf{M}(d_x d_y)) \subset \tilde{O}(d_x d_y)$, where $d_x = \max(\deg_x f, \deg_x g)$ and $d_y = \max(\deg_y f, \deg_y g)$, see e.g. [28, Corollary 8.28].

For a bivariate polynomial $f = \sum_{i=0}^k f_i(x) y^i \in \mathbb{K}[x, y]$, we define its y-leading coefficient as $\mathsf{LC}_y(f) = f_k \in \mathbb{K}[x]$. We say that f is "y-monic" if $\mathsf{LC}_y(f) = 1$.

For our genericity results, we will invoke the following staple:

Lemma 2.1 (DeMillo-Lipton-Schwartz-Zippel [6, 20, 29]). *Let $f \in \mathbb{K}[x_1, \ldots, x_n]$ and $\mathcal{T} \subseteq \mathbb{K}$ Let $\alpha_1, \ldots, \alpha_k \in \mathcal{T}$ be chosen independently and uniformly at random. Then the probability that $f(\alpha_1, \ldots, \alpha_k) = 0$ is at most $d/|\mathcal{T}|$, where d is the total degree of f.*

2.1 Vanishing ideals of point sets

For a point set $\mathcal{P} \subseteq \mathbb{K}^2$, we define its *vanishing ideal* as follows:

$$\Gamma(\mathcal{P}) = \{g \in \mathbb{K}[x, y] \mid g(P) = 0 \text{ for all } P \in \mathcal{P}\} \subseteq \mathbb{K}[x, y] .$$

The x-valency of $\mathcal{P}$, denoted by $v_x(\mathcal{P})$, is the maximal number of y-coordinates for any given x-coordinate, i.e. $\max_{\alpha \in \mathbb{K}} |\{\beta \in \mathbb{K} \mid (\alpha, \beta) \in \mathcal{P}\}|$. When $v_x(\mathcal{P}) = 1$ then all the x-coordinates of $\mathcal{P}$ are different.

The following is an explicit lex-ordered Gröbner basis for $\Gamma(\mathcal{P})$:

Proposition 2.2. *Let $\mathcal{P} \subseteq \mathbb{K}^2$ be a point set and $v_x = v_x(\mathcal{P})$, and let $v_\alpha = |\mathcal{Y}_\alpha| \leq v_x$ for each $\alpha \in \mathbb{K}$, where $\mathcal{Y}_\alpha = \{\beta \in \mathbb{K} \mid (\alpha, \beta) \in \mathcal{P}\}$. Define for each $i = 0, \ldots, v_x$ the set $\mathcal{X}_i = \{\alpha \in \mathbb{K} \mid v_\alpha > i\}$. A Gröbner basis of $\Gamma(\mathcal{P})$ according to the lex-order $\prec$ with $x \prec y$ is given by $G = \{b_0, \ldots, b_{v_x}\} \subset \mathbb{K}[x, y]$ where*

$$b_0 = \prod_{\alpha \in \mathcal{X}_0} (x - \alpha) \in \mathbb{K}[x] \quad \text{and} \quad b_i = (y^i - b_i') \prod_{\alpha \in \mathcal{X}_i} (x - \alpha) ,$$

for $i = 1, \ldots, v_x$, where $b_i' \in \mathbb{K}[x, y]$ is any polynomial with $\deg_y b_i' < i$ satisfying $b_i'(\alpha, \beta) = \beta^i$ for each $\alpha \in \mathcal{X}_0 \setminus \mathcal{X}_i$ and $\beta \in \mathcal{Y}_\alpha$. Such b_i' always exists. A minimal Gröbner basis $G' \subseteq G$ is given by $G' = \{b_j \mid j \in J\}$, where $J = \{0\} \cup \{1 \leq j \leq v_x \mid \mathcal{X}_j \neq \mathcal{X}_{j-1}\}$.

The proof of Proposition 2.2 is in the appendix , but remark that we can take b_i' to be

$$b_i' = \sum_{\alpha \in \mathcal{X}_0 \setminus \mathcal{X}_i} b_{i,\alpha}(y) \prod_{\alpha' \in \mathcal{X}_0 \setminus (\mathcal{X}_i \cup \{\alpha\})} \frac{x - \alpha'}{\alpha - \alpha'} . \tag{1}$$

for $i = 1, \ldots, v_x$, where $b_{i,\alpha} \in \mathbb{K}[y]$ is the interpolation polynomial satisfying $b_{i,\alpha}(\beta) = \beta^i$ for each $\beta \in \mathcal{Y}_\alpha$.

When $v_x(\mathcal{P}) = 1$ then $\Gamma(\mathcal{P}) = \langle b_0, y - r \rangle$, where $b_0 \in \mathbb{K}[x]$ is as in the proposition and $r \in \mathbb{K}[x]$ is the interpolation polynomial satisfying $r(\alpha) = \beta$ for each $(\alpha, \beta) \in \mathcal{P}$.

The following lemma bounds the cost of computing a reduced Gröbner basis of $\Gamma(\mathcal{P})$, which we use only to bound the complexity of our precomputation; it rests on [2] which is currently in review. The interpolation algorithm of that paper is a small generalisation of Pan's interpolation algorithm on grids [17].

Lemma 2.3. *Given a point set $\mathcal{P} \subseteq \mathbb{K}^2$ of size n, we can compute a reduced Gröbner basis $G' \subset \mathbb{K}[x, y]$ for $\Gamma(\mathcal{P})$ according to the lex-order $\prec$ with $x \prec y$ in $\tilde{O}(n{v_x}^3)$ operations in $\mathbb{K}$, where $v_x = v_x(\mathcal{P})$.*

Proof. Use the notation of Proposition 2.2. For each $i = 1, \ldots, v_x$, we need to compute a polynomial $b_i' \in \mathbb{K}[x, y]$ such that $b_i'(\alpha, \beta) = \beta^i$ for each $\alpha \in \mathcal{X}_0 \setminus \mathcal{X}_i$ and $\beta \in \mathcal{Y}_\alpha$. For each such α, then $|\mathcal{Y}_\alpha| < i$. Now [2, Theorem IV.5] shows that for any interpolation constraints on such a point set, we can compute a polynomial with y-degree less than i and x-degree less than $n_i := |\mathcal{X}_0 \setminus X_i| < n$. The complexity of this algorithm is $\tilde{O}(n_i i)$, so to compute all the b_i' costs $\tilde{O}(n{v_x}^2)$. To obtain a reduced Gröbner basis, we then compute $b_i = (\ldots((b_i' \operatorname{rem} b_{i-1}') \operatorname{rem} b_{i-2}') \ldots) \operatorname{rem} b_0'$ for $i = 1, \ldots, v_x$ at a total cost of $\tilde{O}(n{v_x}^3)$. □

We define $\Gamma_m(\mathcal{P}) \subset \Gamma(\mathcal{P})$ to be the subset of polynomials of y-degree less than m. The following shows that $\Gamma_m(\mathcal{P})$ is a $\mathbb{K}[x]$-module of rank m, and we can compute an explicit basis for it using Lemma 2.3. The lemma follows from Lazard's structure theorem on lex-Gröbner bases of bivariate ideals [11], see the appendix .

Corollary 2.4. *Let $I \subseteq \mathbb{K}[x, y]$ be an ideal and $G = \{b_0, \ldots, b_s\} \subset \mathbb{K}[x, y]$ a minimal Gröbner basis according to lex-order $\prec$ with $x \prec y$. For some $m \in \mathbb{Z}_{>0}$ let $I_m = \{f \in I \mid \deg_y f < m\}$. Let $\hat{s} = \max_i\{\deg_y b_i < m\}$, and let $d_i = \deg_y b_i$ for $i = 0, \ldots, \hat{s}$ and $d_{\hat{s}+1} = m$. Then I_m is a $\mathbb{K}[x]$-module of rank $m - \deg_y b_0$ with a basis $B = \{y^j b_i\}_{i,j}$, where $i = 0, \ldots, \hat{s}$ and $j = 0, \ldots, d_{i+1} - d_i - 1$.*

3 RESHAPE

We first describe our "reshape" algorithm which takes $f \in \mathbb{K}[x, y]$ and an ideal I and finds $\hat{f} \in f + I$ whose y-degree is below some target. This will pass through several intermediate elements of $f + I$ of progressively smaller y-degree. This sequence of y-degrees has to be of the following form:

Definition 3.1. We say $\boldsymbol{\eta} = (\eta_0)_{i=0}^k \in \mathbb{Z}_{>0}^{k+1}$ is a (η_0, η_k)-*reshaping sequence* if $\eta_{i-1} > \eta_i \geq \lfloor \frac{2}{3}\eta_{i-1} \rfloor$ for $i = 1, \ldots, k$. Let $I \subseteq \mathbb{K}[x, y]$ be an ideal and $\boldsymbol{\eta} = (\eta_i)_{i=0}^k$ a reshaping sequence. The tuple $(g_i)_{i=1}^k \in I^k$ is an $\boldsymbol{\eta}$-*reshaper for* I if for each $i = 1, \ldots, k$ then $g_i = y^{\eta_i} + \hat{g}_i$ where $\deg_y \hat{g}_i \leq 2\eta_i - \eta_{i-1}$.

Our algorithms are faster with short reshaping sequences, so we should choose $\eta_i \approx \frac{2}{3}\eta_{i-1}$, and hence $2\eta_i - \eta_{i-1} \approx \frac{1}{3}\eta_i$. It is easy to see that for any $a, b \in \mathbb{Z}_{>0}$, there is an (a, b)-reshaping sequence of length at most $\log_{3/2}(a) + 1$. We return in Section 3.1 to when $\boldsymbol{\eta}$-reshapers exist for vanishing ideals of point sets.

Algorithm 1 Reshape$(f, \boldsymbol{\eta}, \boldsymbol{g})$

Input: A reshaping sequence $\boldsymbol{\eta} = (\eta_i)_{i=0}^k \in \mathbb{Z}_{>0}^{k+1}$. $\boldsymbol{\eta}$-reshaper $\boldsymbol{g} = (g_i)_{i=1}^k \in I^k$ for some ideal $I \subseteq \mathbb{K}[x, y]$. $f \in \mathbb{K}[x, y]$ with $\deg_y f < \eta_0$.

Output: $\hat{f} \in f + I$ satisfying $\deg_y \hat{f} < \eta_k$ and $\deg_x \hat{f} \leq \deg_x f + \sum_{i=1}^k \deg_x g_i$.

1: $\hat{f} \leftarrow f$
2: **for** $i = 1, \ldots, k$ **do**
3: Let $\hat{f} = \hat{f}_1 y^{\eta_i} + \hat{f}_0$, where $\deg_y \hat{f}_0 < \eta_i$
4: $\hat{f} \leftarrow \hat{f} - \hat{f}_1 g_i$
5: **return** $\hat{f}$

Theorem 3.2. *Algorithm 1 is correct and has complexity*

$$\tilde{O}(k \deg_y f \deg_x f + k \textstyle\sum_{i=1}^k \eta_i \deg_x g_i) .$$

Proof. $\hat{f} \in f + I$ since each $g_i \in I$. We turn to the degree bounds. Let $\hat{f}_i, \hat{f}_{i,0}, \hat{f}_{i,1}$ be the values of $\hat{f}, \hat{f}_0$ resp. $\hat{f}_1$ at the end of iteration i. We show the following loop-invariants, which imply the bounds on the degrees on the output $\hat{f}$:

- $\deg_x \hat{f}_i \leq \deg_x f + \sum_{j=1}^i \deg_x g_j$; and
- $\deg_y \hat{f}_i < \eta_i$.

Both are true for $i = 0$, (just before the loop). For the x-degree bound, then clearly $\deg_x \hat{f}_i \le \deg_x \hat{f}_{i-1} + \deg_x g_i$, and the loop invariant follows. For the y-degree bound, then write $g_i = y^{\eta_i} + \hat{g}_i$, where $\deg_y \hat{g}_i \le 2\eta_i - \eta_{i-1}$. We get that

$$\hat{f}_i = \hat{f}_{i-1} - \hat{f}_{i,1}(y^{\eta_i} + \hat{g}_i) = \hat{f}_{i,0} + \hat{f}_{i,1}\hat{g}_i$$

Since $\deg_y \hat{f}_{i,1} = \deg_y \hat{f}_{i-1} - \eta_i < \eta_{i-1} - \eta_i$, it follows that $\deg_y(\hat{f}_{i,1}\hat{g}_i) < \eta_i$, and since also $\deg_y \hat{f}_{i,0} < \eta_i$ then $\deg_y \hat{f}_i < \eta_i$.

For complexity, work is only done in Line 4. We can assume $\eta_1 \le \deg_y f$ for otherwise the algorithm could be called with the same input but the first element of both $\boldsymbol{\eta}$ and $\boldsymbol{g}$ removed. Since $\deg_y \hat{f}_{i,1}\hat{g}_i < \eta_i$ then the multiplication can be done in complexity $O(\mathsf{M}(d_x\eta_i))$, where $d_x = \max(\deg_x f_{i,1}, \deg_x \hat{g}_i)$. The addition in the same line is only linear in the same amount since similar degree bounds hold for $\hat{f}_{i,0}$. We get that the i'th iteration has cost

$$\tilde{O}\left((\deg_x \hat{f}_{i,1} + \deg_x \hat{g}_i)\eta_i\right) \subseteq \tilde{O}((\deg_x f_i + \textstyle\sum_{j=1}^{k} \deg_x \hat{g}_j)\eta_i)$$
$$\subseteq \tilde{O}(\deg_x f \deg_y f + \eta_i \textstyle\sum_{j=1}^{k} \deg_x \hat{g}_j)$$

Over all iterations, we get $\tilde{O}(k \deg_x f \deg_y f + \sum_{i=1}^{k} \eta_i \sum_{j=1}^{k} \deg_x \hat{g}_j)$. Transpose the double-sum and use $\eta_1 > \eta_2 > \ldots > \eta_k$ to conclude. □

3.1 Reshapers for point sets

Definition 3.3. Let $\mathcal{P} \subseteq \mathbb{K}^2$ be a point set with $n = |\mathcal{P}|$, and let $\boldsymbol{\eta} = (\eta_i)_{i=0}^{k}$ be a reshaping sequence. Then $\mathcal{P}$ is "$\boldsymbol{\eta}$-balanced" if there exists an $\boldsymbol{\eta}$-reshaper $\boldsymbol{g} = (g_i)_{i=1}^{k} \in \mathbb{K}[x, y]$ for $\Gamma(\mathcal{P})$ such that $\deg_x g_i \le \left\lfloor \frac{n}{2\eta_i - \eta_{i-1}+1} \right\rfloor + 1$ for $i = 1, \ldots, k$.

In Section 8 we prove that balancedness captures the *expected* x-degree of a reshaping sequence. The following is crucial for our complexity estimates:

Lemma 3.4. *Let $\mathcal{P} \subseteq \mathbb{K}^2$ be an $\boldsymbol{\eta}$-balanced point set with $|\mathcal{P}| = n$ for some reshaping sequence $\boldsymbol{\eta} = (\eta_i)_{i=0}^{k}$, and $\boldsymbol{g} = (g_i)_{i=1}^{k}$ a corresponding $\boldsymbol{\eta}$-reshaper. Then $\sum_{i=1}^{k} \deg_x g_i \le \frac{(3n+1)k}{\eta_k} \le (3n+1)k$.*

Proof. Since $\eta_i \ge \lfloor \frac{2}{3}\eta_{i-1} \rfloor$ then $2\eta_i - \eta_{i-1} + 1 \ge \lfloor \frac{\eta_{i-1}}{3} \rfloor + 1 \ge \frac{\eta_i}{3}$. Then $\deg_x g_i \le \left\lfloor \frac{n}{2\eta_i - \eta_{i-1}+1} \right\rfloor + 1 \le \frac{3n}{\eta_i} + 1 \le \frac{3n}{\eta_k} + 1$. □

Lemma 3.5. *Let $\mathcal{P} \subseteq \mathbb{K}^2$ be a point set and $\boldsymbol{\eta} = (\eta_i)_{i=0}^{k}$ a reshaping sequence. As long as $\delta_i := 2\eta_i - \eta_{i-1} + 1 \ge v_x(\mathcal{P})$ for each $i = 1, \ldots, k$, there exists an $\boldsymbol{\eta}$-reshaper $\boldsymbol{g} \in \mathbb{K}[x, y]^k$ for $\Gamma(\boldsymbol{g})$.*

Proof. Let $G = (b_i)_{i=0}^{v_x(\mathcal{P})}$ be as in Proposition 2.2. Since $b_{v_x(\mathcal{P})}$ is y-monic, $\deg_y(y^{\eta} \operatorname{rem} G) < v_x(\mathcal{P})$ for any $\eta \in \mathbb{Z}_{>0}$, Hence we may set $g_i = y^{\eta_i} - (y^{\eta_i} \operatorname{rem} G)$, and this yields an $\boldsymbol{\eta}$-reshaper as long as $v_x(\mathcal{P}) \le \delta_i$. □

Corollary 3.6. *Let $\mathcal{P} \subseteq \mathbb{K}^2$ be a point set of cardinality n and $a, b \in \mathbb{Z}_{>0}$ with $n > a > b \ge v_x := \gamma(P)$. Then there is an (a, b)-reshaping sequence $\boldsymbol{\eta}$ which satisfies the conditions of Lemma 3.5 and has length $k \le \log_{3/2}(a) + 1 \in O(\log(a))$.*

Proof. Let $v = v_x - 1$ and let $\boldsymbol{\eta'} = (\eta'_0, \ldots, \eta'_k)$ be any $(a - v, b - v)$-reshaping sequence with $k \le \log_{3/2}(a - v) + 1$. Now let $\boldsymbol{\eta} = (\eta_0, \ldots, \eta_k)$ be defined by $\eta_i = \eta'_i + v$ for $i = 0, \ldots, k$. It is not hard to see that $\boldsymbol{\eta}$ is an (a, b)-reshaping sequence. Indeed, clearly the endpoints are correct, and $\eta_{i-1} > \eta_i$ for $i = 1, \ldots, k$. Moreover,

$$\eta_i = \eta'_i + v \ge \lfloor \tfrac{2}{3}\eta'_{i-1} \rfloor + v = \lfloor \tfrac{2}{3}(\eta_{i-1} - v) \rfloor + v \ge \lfloor \tfrac{2}{3}\eta_{i-1} \rfloor .$$

Finally, observe that

$$2\eta_i - \eta_{i-1} + 1 = 2(\eta'_i + v) - (\eta'_{i-1} + v) + 1$$
$$= 2\eta'_i - \eta'_{i-1} + v + 1 > v + 1 = v_x ,$$

which concludes the proof. □

4 MULTI-POINT EVALUATION

In this section we use reshaping for MPE with precomputation; i.e. given a point set $\mathcal{P} \subset \mathbb{K}^2$ upon which we are allowed to perform precomputation, and a polynomial $f \in \mathbb{K}[x, y]$ which is assumed to be received at online time, compute $f(P)$ for all $P \in \mathcal{P}$. Algorithm 2 deals with the case $v_x(\mathcal{P}) = 1$, which we can reduce to an instance of univariate MPE using Reshape. The cost of the Algorithm 2 follows directly from Theorem 3.2.

Algorithm 2 BivariateMPE$_{d,\boldsymbol{\eta},\mathcal{P}}(f)$

Preinput: $d \in \mathbb{Z}_{>0}$; a $(d, 1)$-reshaping sequence $\boldsymbol{\eta}$. Point set $\mathcal{P} = \{(\alpha_i, \beta_i)\}_{i=1}^{n} \subset \mathbb{K}^2$ with α_i pairwise distinct.
Precomputation:
a: $\boldsymbol{g} \leftarrow \boldsymbol{\eta}$-reshapers for $\Gamma(\mathcal{P})$.
Input: $f \in \mathbb{K}[x, y]$ with $\deg_y f < d$.
Output: $\big(f(\alpha_1, \beta_1), \ldots, f(\alpha_n, \beta_n)\big) \in \mathbb{K}^n$
1: $\hat{f} \leftarrow$ Reshape$(f, \boldsymbol{\eta}, \boldsymbol{g}) \in \mathbb{K}[x]$
2: **return** $\big(\hat{f}(\alpha_1), \ldots, \hat{f}(\alpha_n)\big) \in \mathbb{K}^n$ ▷ *computed using univariate MPE*

Theorem 4.1. *Algorithm 2 is correct. If $\mathcal{P}$ is $\boldsymbol{\eta}$-balanced and $k \in O(\log(n))$, the complexity is $\tilde{O}(\deg_x f \deg_y f + n)$.*

Algorithm 2 can easily be extended to the case where $v_x(\mathcal{P}) > 1$ by partitioning $\mathcal{P}$ into $v_x(\mathcal{P})$ many subsets, each having x-valency one. This approach also has quasi-linear complexity in the input size as long as $v_x(\mathcal{P}) \ll n$, or more precisely if $nv_x(\mathcal{P}) \in \tilde{O}(n)$.

When $v_x(\mathcal{P})$ is large, this strategy is costly, and we proceed instead by shearing the point set, as proposed by Nüsken and Ziegler [16], so that the resulting point set has distinct x-coordinates: by taking $\theta \in \mathbb{L} \setminus \mathbb{K}$, where $\mathbb{L}$ is an extension field of $\mathbb{K}$ of degree 2, we apply the map $(\alpha, \beta) \mapsto (\alpha + \theta\beta, \beta)$ to each element of $\mathcal{P}$. The problem then reduces to evaluating $\hat{f} = f(x - \theta y, y)$ at the sheared points. To compute $\hat{f}$ [16] provides an algorithm with complexity $O(\mathsf{M}(d_x(d_x + d_y))\log(d_x))$. Algorithm 3 describes a basic algorithm for this task which improves the cost on the logarithmic level.

Theorem 4.2. *Algorithm 3 is correct and has complexity*

$$O\left((d_x + d_y)\mathsf{M}(d_x)\log(d_x)\right) \subset \tilde{O}\left(d_x(d_x + d_y)\right) .$$

Proof. For correctness we write $f = \sum_{t=0}^{d_x+d_y} \hat{h}_t$, where

$$\hat{h}_t = \textstyle\sum_{i=\max(0,\, t-d_y)}^{\min(t,\, d_x)} f_{i,\, t-i} x^i y^{t-i}$$

Algorithm 3 ShearPoly(h, θ_x, θ_y)

Input: $f = \sum_{i=0}^{d_x}\sum_{j=0}^{d_y} f_{i,j}x^iy^j \in \mathbb{L}[x,y]$, where $d_x < |\mathbb{L}|$. $\theta_x, \theta_y \in \mathbb{L}$ with $\theta_x \neq 0$.
Output: $\hat{f} = f(\theta_x x + \theta_y y, y)$ with $\deg_x \hat{f} \leq d_x$ and $\deg_y \hat{f} \leq d_x + d_y$.

1: **for** $t = 0, \ldots, d_x + d_y$ **do**
2: $\quad h_t \leftarrow \sum_{i=\max(0,\,t-d_y)}^{\min(t,\,d_x)} f_{t-i,i}z^i \in \mathbb{L}[z]$
3: $\quad s_t \leftarrow h_t(\theta_x z + \theta_y)$ ▷ *Taylor shift*
4: $\hat{f} \leftarrow \sum_{t=0}^{d_x+d_y} y^t s_t(x/y)$
5: **return** $\hat{f}$

are homogeneous polynomials. Since $\hat{h}_t(x,y) = y^t h_t(x/y)$, we have

$$f(\theta_x x + \theta_y y, y) = \sum_{t=0}^{d_x+d_y} y^t h_t\left(\frac{\theta_x x + \theta_y y}{y}\right)$$
$$= \sum_{t=0}^{d_x+d_y} y^t h_t\left(\theta_x \tfrac{x}{y} + \theta_y\right) = \sum_{t=0}^{d_x+d_y} y^t s_t(x/y).$$

For complexity we observe only computing the Taylor shifts s_t costs operations in $\mathbb{K}$. We use univariate MPE and interpolation: $\hat{s}_t$ is given by interpolating $\hat{s}_t(z_i) = s_t(\theta_x z_i + \theta_y)$ for $d_x + 1$ distinct $z_i \in \mathbb{L}$. This costs $O(\mathsf{M}(d_x)\log(d_x))$ for each s_t. □

Algorithm 4 ValencyMPE$_{d,\boldsymbol{\eta},\mathcal{P}}(f)$

Preinput: $d \in \mathbb{Z}_{>0}$; a $(d,1)$-reshaping sequence $\boldsymbol{\eta}$. Point set $\mathcal{P} = \{(\alpha_i, \beta_i)\}_{i=1}^n \subset \mathbb{K}^2$ with $\nu_x(\mathcal{P}) > 1$.
Precomputation:
a: $\mathbb{L} \leftarrow$ extension field of $\mathbb{K}$ with $[\mathbb{L} : \mathbb{K}] = 2$
b: $\hat{\mathcal{P}} \leftarrow \{(\alpha_i + \theta_y\beta_i, \beta_i)\}_{i=1}^n \subset \mathbb{L}^2$, where $\theta_y \in \mathbb{L} \setminus \mathbb{K}$
c: Do the precomputation of BivariateMPE$_{d,\boldsymbol{\eta},\hat{\mathcal{P}}}$
Input: $f \in \mathbb{K}[x,y]$ with $\deg_y f + \deg_y f < d$.
Output: $\big(f(\alpha_1,\beta_1), \ldots, f(\alpha_n,\beta_n)\big) \in \mathbb{K}^n$

1: $\hat{f} \leftarrow f(x - \theta_y y, y)$ ▷ *as* ShearPoly$(f, 1, -\theta_y)$
2: **return** BivariateMPE$_{d,\boldsymbol{\eta},\hat{\mathcal{P}}}(\hat{f})$

Theorem 4.3. *Algorithm 4 is correct. If $\hat{\mathcal{P}}$ is $\boldsymbol{\eta}$-balanced and the length of $\boldsymbol{\eta}$ is in $O(\log(n))$, then it has complexity $\tilde{O}(\deg_x f(\deg_x f + \deg_y f) + n)$.*

5 INTERPOLATION

In this section we use reshaping for the interpolation problem in a very similar setting: we preinput a point set $\mathcal{P}$ for precomputation, and interpolation values at online time. When $\mathcal{P}$ are appropriately balanced, we are able to solve the interpolation problem in quasi-linear time (see Algorithm 5). The strategy is to first shear the point set to have unique y-coordinates and then compute $u \in \mathbb{L}[y]$ which interpolates the values on the sheared y-coordinates. We then reshape this into $r \in \mathbb{L}[x,y]$ with x- and y-degree roughly $\sqrt{n}$. Shearing back this polynomial to interpolate the original point set is now in quasi-linear time. We then use reshaping again to obtain the target y-degree.

Algorithm 5 Interpolate$_{\mathcal{P},\boldsymbol{\eta}_1,\boldsymbol{\eta}_2,d_y,\theta_x}(\boldsymbol{\gamma})$

Preinput: Point set $\mathcal{P} = \{(\alpha_i,\beta_i)\}_{i=1}^n \subseteq \mathbb{K}^2$ and $d_y \in \mathbb{Z}_{>0}$ with $\lfloor\sqrt{n}\rfloor + 1 \geq d_y \geq \nu_x(\mathcal{P})$. $\theta_x \in \mathbb{L}$ a generator of the degree 2-extension $\mathbb{L} : \mathbb{K}$. An (n, d_y)-reshaping sequence $\boldsymbol{\eta} = (\eta_i)_{i=0}^k$ with $\eta_{k_1} = \lfloor\sqrt{n}\rfloor$ for some k_1 and satisfying the conditions of Lemma 3.5.
Precomputation:
a: $\boldsymbol{\eta}_1 \leftarrow (\eta_i)_{i=0}^{k_1}$ and $\boldsymbol{\eta}_2 \leftarrow (\eta_i)_{i=k_1}^{k}$.
b: $\boldsymbol{g}_1 \leftarrow \boldsymbol{\eta}_1$-reshapers for $\hat{\mathcal{P}} := \{(\alpha_i, \theta_x\alpha_i + \beta_i)\}_{i=1}^n$.
c: $\boldsymbol{g}_2 \leftarrow \boldsymbol{\eta}_2$-reshapers for $\mathcal{P}$.
Input: Interpolation values $\boldsymbol{\gamma} = (\gamma_i)_{i=1}^n \in \mathbb{K}^n$.
Output: $f \in \mathbb{K}[x,y]$ satisfying: $f(\alpha_i,\beta_i) = \gamma_i$ for $i = 1,\ldots,n$, $\deg_y f < d_y$ and $\deg_x f \leq \lfloor\sqrt{n}\rfloor + \sum_{g\in\boldsymbol{g}_1}\deg_x g + \sum_{g\in\boldsymbol{g}_2}\deg_x g$.

1: $u \in \mathbb{L}[y]$ with $\deg u < n$ and $u(\theta_x\alpha_i + \beta_i) = \gamma_i$ for $i = 1,\ldots,n$
2: $r \leftarrow$ Reshape$(u, \boldsymbol{\eta}_1, \boldsymbol{g}_1) \in \mathbb{L}[x,y]$
3: $s \leftarrow r(x, \theta_x x + y) = s'(y,x)$ ▷ $s' =$ ShearPoly$(r(y,x), \theta_x, 1)$
4: Let $s = s_1 + \theta_x s_2$, where $s_1, s_2 \in \mathbb{K}[x,y]$
5: $f \leftarrow$ Reshape$(s_1, \boldsymbol{\eta}_2, \boldsymbol{g}_2) \in \mathbb{K}[x,y]$
6: **return** f

Theorem 5.1. *Algorithm 5 is correct and has complexity*

$$\tilde{O}\left(k_2(\sqrt{n} + \textstyle\sum_{j=1}^{k_1}\deg_x g_{1,j})^2 + \sum_{\ell=1}^{2} k_l \sum_{j=1}^{k_\ell}\eta_{\ell,k}\deg_x g_{\ell,j}\right).$$

If $\hat{\mathcal{P}}$ is $\boldsymbol{\eta}_1$-balanced and $\mathcal{P}$ is $\boldsymbol{\eta}_2$-balanced the complexity is $\tilde{O}(n)$ assuming that $k_1, k_2 \in O(\log n)$.

Proof. First note that it follows from Lemma 3.5 that $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ actually exist since $d_y > \nu_x(\mathcal{P})$. For correctness, observe that $\theta_x\alpha_i + \beta_i$ are pairwise distinct for $i = 1,\ldots,n$, so it makes sense to compute u. Viewing u as an element of $\mathbb{L}[x,y]$ with $\deg_x u = 0$, then $u(\alpha_i, \theta_x\alpha_i + \beta_i) = \gamma_i$. By Theorem 3.2 then has the same evaluations and $\deg_y r < \lfloor\sqrt{n}\rfloor$ and $\deg_x r \leq \sum_{j=1}^{k_1}\deg_x g_{1,j}$.

By Theorem 4.2, then $s(x,y) = r(x, \theta_x x + y)$, and hence $s(\alpha_i,\beta_i) = s_1(\alpha_i,\beta_i) + \theta_x s_2(\alpha_i,\beta_i) = \gamma_i$ for $i = 1,\ldots,n$. Since $s_1, s_2 \in \mathbb{K}[x,y]$ and all entries in $\boldsymbol{\gamma}$ are in $\mathbb{K}$, we must have that $s_2(\alpha_i,\beta_i) = 0$ for $i = 1,\ldots,n$, which implies that $s_1(\alpha_i,\beta_i) = \gamma_i$. We also then have that $\deg_y s_1 \leq \deg_y s < \lfloor\sqrt{n}\rfloor$ and

$$\deg_x s_1 \leq \deg_x s \leq \deg_y r + \deg_x r \leq \lfloor\sqrt{n}\rfloor + \textstyle\sum_{j=1}^{k_1}\deg_x g_{1,j}.$$

Finally, $f(\alpha_i,\beta_i) = \gamma_i$ for $i = 1,\ldots,n$, and $\deg_y f < d_y$, and

$$\deg_x f \leq \lfloor\sqrt{n}\rfloor + \textstyle\sum_{j=1}^{k_1}\deg_x g_{1,j} + \sum_{j=1}^{k_2}\deg_x g_{2,j}.$$

The complexity estimate simply gathers the calls to Algorithm 1 and Algorithm 3. The relaxed cost under the balance assumptions is due to Lemma 3.4. □

6 MODULAR COMPOSITION

We now turn to the following generalisation of the univariate modular composition problem: given $G, R \in \mathbb{K}[x]$ with $n := \deg_x G > \deg_x R$ as well as $f \in \mathbb{K}[x,y]$, compute

$$f(x, R(x)) \operatorname{rem} G(x) \in \mathbb{K}[x]. \tag{2}$$

We will consider the variant of the problem where G and R are available for precomputation. Consider the ideal $I = \langle G, y - R\rangle \subseteq \mathbb{K}[x, y]$. Computing (2) is tantamount to computing the unique element of $(f + I) \cap \mathbb{K}[x]$ of degree less than n. We can therefore consider this a reshaping task: given f of some y-degree, reshape it to one of y-degree 0 while keeping it fixed modulo I: this is formalised as Algorithm 6.

Similar to the terminology for point sets, if $\boldsymbol{\eta} = (\eta_i)_{i=0}^{k}$ is a reshaping sequence, we say that $I = \langle G, y - R\rangle$ is $\boldsymbol{\eta}$-balanced if there exists an $\boldsymbol{\eta}$-reshaper $\boldsymbol{g} = (g_i)_{i=1,\dots,k} \in \mathbb{K}[x, y]$ for I such that $\deg_x g_i \leq \left\lfloor \frac{n}{2\eta_i - \eta_{i-1}} \right\rfloor + 1$.

Algorithm 6 $\textsc{ModComp}_{G,R,d_y,\boldsymbol{\eta}}(f)$

Preinput: $G, R \in \mathbb{K}[x]$ with $n := \deg_x G > \deg_x R$. A degree bound $d_y \in \mathbb{Z}_{>0}$ with $d_y \leq n$. A $(d_y, 1)$-reshaping sequence $\boldsymbol{\eta}$.
Precomputation:
a: $\boldsymbol{g} \leftarrow \boldsymbol{\eta}$-reshapers for I.
Input: $f \in \mathbb{K}[x, y]$ with $\deg_y f < d_y$.
Output: $f(x, R) \operatorname{rem} G \in \mathbb{K}[x]$
1: $\hat{f} \leftarrow \textsc{Reshape}(f, \boldsymbol{\eta}, \boldsymbol{g}) \in \mathbb{K}[x]$.
2: **return** $\hat{f} \operatorname{rem} G$.

Theorem 6.1. *Algorithm 6 is correct. If $\langle G, y - R\rangle$ is balanced it costs $\tilde{O}(\deg_x f \deg_y f + n)$ whenever the length of $\boldsymbol{\eta}$ is in $O(\log(n))$.*

7 PRECOMPUTING RESHAPERS

We now describe algorithms for the precomputation of reshapers. We use a canonical normal form of univariate polynomial matrices called *Popov form* [19]. They exist for singular and rectangular matrices, but we will only need the full-rank square case.

Definition 7.1. For any row vector $\boldsymbol{v} \in \mathbb{K}[x]^{1\times\delta}$ its *the row degree* denoted $\deg \boldsymbol{v}$ is the maximal degree among its entries. The *pivot* of $\boldsymbol{v}$ is the rightmost entry of $\boldsymbol{v}$ with degree equal to $\deg \boldsymbol{v}$. A matrix $P = [p_{ij}] \in \mathbb{K}[x]^{\delta\times\delta}$ having full rank is in *Popov* form if p_{ii} is the pivot of the i'th row, is monic and $\deg p_{ii} > \deg p_{ji}$ for any $j \neq i$.

For a full-rank $\mathbb{K}[x]$-module $\mathcal{M} \in \mathbb{K}[x]^{\delta\times\delta}$, there is a unique $P \in \mathbb{K}[x]^{\delta\times\delta}$ in Popov form with $\operatorname{RowSp}_{\mathbb{K}[x]}(P) = \mathcal{M}$. We say that P is *the Popov basis of* $\mathcal{M}$. P has minimal row degrees in the following sense: If $N \in \mathbb{K}[x]^{\delta\times\delta}$ is another basis of $\mathcal{M}$, there is a bijection ψ from the rows of P to the rows of N such that $\deg \boldsymbol{p} \leq \deg \psi(\boldsymbol{p})$ for any row $\boldsymbol{p} \in \mathbb{K}[x]^{1\times\delta}$ of P. Any basis of $\mathcal{M}$ has the same determinant up to scalar multiplication, so we denote by $\operatorname{degdet}(\mathcal{M})$ the degree of any of these. For the Popov basis P then $|\operatorname{cdeg} P| = \operatorname{degdet} P = \operatorname{degdet}(\mathcal{M})$.

For any $\boldsymbol{t} = (t_i)_{i=1}^{\delta} \in \mathbb{Z}_{\geq 0}$ let $|\boldsymbol{t}| = \sum_{i=1}^{\delta} t_i$, and for any matrix $M \in \mathbb{K}[x]^{\delta\times\delta}$ let $\operatorname{cdeg}(M) = (d_i)_{i=1}^{\delta} \in \mathbb{Z}_{\geq 0}^{\delta}$, where d_i denotes the maximal degree in the i'th column of M (and 0 if the column is 0), for $i = 1, \dots, \delta$. The following result allows us to compute Popov forms efficiently.

Proposition 7.2 ([15]). *There is an algorithm which inputs a nonsingular matrix $M \in \mathbb{K}[x]^{\delta\times\delta}$ and outputs the Popov basis of $\operatorname{RowSp}_{\mathbb{K}[x]}(M)$ using $\tilde{O}(\delta^{\omega-1}|\operatorname{cdeg}(M)|)$ operations in $\mathbb{K}$.*

Popov forms are special cases of "row reduced forms" which enjoy the Predictable Degree Property [9, Thm. 6.3-13], which implies the following: if $P \in \mathbb{K}[x]^{\delta\times\delta}$ is the Popov basis of $\mathcal{M}$, then any row vector $\boldsymbol{v} \in \mathbb{K}[x]^{1\times\delta}$ can be written uniquely as $\boldsymbol{v} = \boldsymbol{v'} + \boldsymbol{u}P$, where $\operatorname{cdeg}(\boldsymbol{v'}) < \operatorname{degdet}(P)$ entrywise. Furthermore, $\boldsymbol{v'}$ has minimal row degree among all vectors of the coset $\boldsymbol{v} + \mathcal{M}$. We will denote $\boldsymbol{v'}$ by $\boldsymbol{v} \operatorname{rem} P$.

Proposition 7.3 ([15]). *There is an algorithm which given $P \in \mathbb{K}[x]^{\delta\times\delta}$ in Popov form, and $\boldsymbol{v} \in \mathbb{K}[x]^{1\times\delta}$ with $\operatorname{cdeg}(\boldsymbol{v}) < \operatorname{cdeg}(P) + |\operatorname{cdeg}(P)|$ entrywise, computes $\boldsymbol{v} \operatorname{rem} P$ using $\tilde{O}(\delta^{\omega-1}|\operatorname{cdeg}(P)|)$ operations in $\mathbb{K}$ assuming that $\delta \in O(|\operatorname{cdeg}(P)|)$.*

In the following, we will convert between bivariate polynomials and $\mathbb{K}[x]$-matrices using the following map: for any $\delta \in \mathbb{Z}_{>0}$ and $f = \sum_{j=0}^{\delta-1} f_j(x)y^j \in \mathbb{K}[x, y]$, we let

$$\phi_\delta(f) := [f_0, \dots, f_{\delta-1}] \in \mathbb{K}[x]^{1\times\delta} .$$

Note that ϕ_δ is a $\mathbb{K}[x]$-module isomorphism which for every $f \in \mathbb{K}[x, y]$ with $\deg_y f < \delta$ satisfies $\deg \phi_\delta(f) = \deg_x(f)$.

Algorithm 7 can be used to compute reshapers for any ideal $I \subseteq \mathbb{K}[x, y]$ given a lex-ordered reduced Gröbner basis $G = [b_0, \dots, b_s]$ of I. For simplicity, we consider only the case where $I \cap \mathbb{K}[x] \neq \emptyset$.

Algorithm 7 $\textsc{ComputeReshaper}(G, \eta, \delta)$

Input: A reduced Gröbner basis $G = \{b_1, \dots, b_s\}$ with respect to the lex-order $\prec$ with $x \prec y$, for an ideal $\mathcal{I} \subseteq \mathbb{K}[x, y]$, such that $0 = \deg_y b_1 < \cdots < \deg_y b_s$. $\eta, \delta \in \mathbb{Z}_{>0}$ with $\delta < \eta$.
Output: If it exists, $g = y^\eta - \hat{g} \in \mathcal{I}$ with $\deg_y \hat{g} < \delta$ and $\deg_x \hat{g}$ minimal. Otherwise "Fail".
1: $R \leftarrow y^\eta \operatorname{rem} G$
2: **if** $\deg_y R \geq \delta$ **then return** "Fail"
3: $B_\delta \leftarrow$ basis of $I_\delta = \{f \in I \mid \deg_y f < \delta\}$ by Corollary 2.4.
4: $M \in \mathbb{K}[x]^{\delta\times\delta} \leftarrow$ row-wise applying ϕ_δ to elements of B_δ.
5: $P \leftarrow$ Popov basis of I_δ from the basis M
6: $\hat{g} \leftarrow -\phi_\delta^{-1}(\phi_\delta(R) \operatorname{rem} P) \in \mathbb{K}[x, y]$
7: $g \leftarrow y^\eta - \hat{g} \in \mathbb{K}[x, y]$
8: **return** g

Theorem 7.4. *Algorithm 7 is correct. It costs $\tilde{O}(\delta^{\omega-1} \operatorname{degdet}(\phi_\delta(I_\delta)) + \eta s \deg_x b_0)$, assuming $\eta \in O(\operatorname{degdet}(\phi_\delta(I_\delta)))$.*

Proof. If a satisfactory $g = y^\eta - \tilde{g} \in I$ exists, then $\deg_y(y^\eta \operatorname{rem} G) \leq \deg_y(\tilde{g})$ since G is a lex-ordered Gröbner basis with $x \prec y$. Hence the algorithm does not fail in Line 2.

For correctness of the output, observe that $y^\eta - R \in I$ so satisfactory $g = y^\eta - \tilde{g}$ all have $\tilde{g} \in R + I_\delta$. Now, $\hat{g}$ of Line 6 is clearly in $R + I_\delta$ since P is the Popov basis of I_δ, but also $\hat{g}$ has minimal x-degree in the coset $R + I_\delta$. Hence among all g of the correct form, the algorithm returns that of minimal x-degree.

For complexity, work is done in Lines 1, 5 and 6. Note that since G is reduced then $\deg_x b_0 > \deg_x b_1 > \ldots > \deg_x b_s$. This implies that the diagonal elements in M are dominant in their columns and hence $|\operatorname{cdeg} M| = \operatorname{degdet}(M) = \operatorname{degdet}(P) = \operatorname{degdet}(I_\delta)$.

For Line 1 we use van der Hoeven [22]: the multivariate division algorithm computes $q_0, \ldots, q_s, R \in \mathbb{K}[x, y]$ such that $y^\eta = q_0 b_0 + \ldots + q_s b_s + R$, and the cost of the algorithm can be bounded as

$$\sum_{i=0}^{s} \deg_x^\circ(q_i b_i) \deg_y^\circ(q_i b_i) + \deg_x^\circ(r) \deg_y^\circ(r)\,,$$

where $\deg_x^\circ \cdot$ denotes an a priori upper bound on the x-degree, and similarly for $\deg_y^\circ \cdot$. Firstly since G is a lex-ordered Gröbner basis, then $\deg_y^\circ(q_i b_i) \leq \deg_y(y^\eta) = \eta$ and $\deg_y^\circ(r) \leq \eta$. For the x-degrees, note that in an iteration of the division algorithm where $b_i, i > 0$ is used, then $\deg_x r < \deg_x b_0$, where r is the current remainder, since otherwise we would have used b_0 as $\deg_y b_0 = 0$. Hence $\deg_x(q_i) \leq \deg_x(q_i \mathsf{LM}_\prec(b_i)) < \deg_x b_0$ and so $\deg_x^\circ(q_i b_i) \leq 2 \deg_x b_0$. Similarly, $\deg_x^\circ(r) < \deg_x b_0$. Left is only $\deg_x^\circ(q_0 b_0)$: since $q_0 b_0 = y^\eta - q_1 b_1 - \ldots - q_s b_s - R$, then $\deg_x(q_0 b_0) \leq \max_i \left(\deg_x(q_i b_i),\ \deg_x(r) \right) \leq 2 \deg_x b_0$. In total, the cost of Line 1 becomes $\tilde{O}(\eta s \deg_x b_0)$.

Line 5 costs $\tilde{O}(\delta^{\omega-1} |\operatorname{cdeg} M|)$ by Proposition 7.2 and Line 6 costs $\tilde{O}(\delta^{\omega-1} \operatorname{degdet}(P))$ since $\deg_x R < \deg_x b_0 < |\operatorname{cdeg} P|$. □

Corollary 7.5. *Given a point set $\mathcal{P} \subseteq \mathbb{K}^2$ of cardinality n and a reshaping sequence $\boldsymbol{\eta} = (\eta_i)_{i=0}^k$ with $n \geq \eta_k$ and satisfying the conditions of Lemma 3.5, then we can determine if $\mathcal{P}$ is $\boldsymbol{\eta}$-balanced and compute an $\boldsymbol{\eta}$-reshaper $\boldsymbol{g} = (g_i)_{i=1}^k$ for $\mathcal{P}$ where each element has minimal possible x-degree in complexity $\tilde{O}(k\eta_0^{\omega-1} n + \eta_0 \nu_x nk)$.*

Proof. Computing a reduced lex-ordered Gröbner basis $G = (b_i)_{i=0}^{\nu_x}$ of $\Gamma(\mathcal{P})$ first costs $\tilde{O}(n{\nu_x}^3)$ by Lemma 2.3. We then run Algorithm 7 on input $\eta = \eta_i$ and $\delta_i = 2\eta_i - \eta_{i-1} + 1 > \nu_x$ for $i = 1, \ldots, k$.

We claim that for any $\delta > \nu_x$ then $\operatorname{degdet}(\phi_\delta(\Gamma_\delta(\mathcal{P}))) = n$: we use the notation $\mathcal{X}_i$ from Proposition 2.2 and the basis $B = [\hat{b}_i]_{i=0}^{\delta-1}$ of $\Gamma_\delta(\mathcal{P})$ given by Corollary 2.4, and let $M \in \mathbb{K}[x]^{\delta \times \delta}$ be the matrix whose rows are given by $\phi(B)$. Then $\operatorname{degdet}(\Gamma_\delta(\mathcal{P})) = \operatorname{degdet}(M)$ since M is a basis for $\Gamma_\delta(\mathcal{P})$, and M is lower-triangular since the $\deg_y \hat{b}_i = i$, and so $\det(M)$ is the product of the $\mathsf{LC}_y(\hat{b}_i)$. For each $\alpha \in \mathbb{K}$ then $x - \alpha$ is going to divide $\mathsf{LC}_y(\hat{b}_i)$ for each $i < \nu_\alpha$, i.e. $(x - \alpha)$ divides $\det(M)$ exactly ν_α times. $\det(M)$ has no other factors, so this gives in total n linear factors and so $\operatorname{degdet}(M) = \operatorname{degdet}(\Gamma_{\delta_i}(\mathcal{P})) = n$.

Thus the cost of each call to Algorithm 7 becomes $\tilde{O}(\eta_0^{\omega-1} n + \eta_0 \nu_x n)$. □

Corollary 7.6. *Given $G, R \in \mathbb{K}[x]$ with $n := \deg G > \deg R$ and a reshaping sequence $\boldsymbol{\eta} = (\eta_i)_{i=0}^k$ with $n \geq \eta_k$, then we can determine if $I := \langle G, y - R \rangle$ is $\boldsymbol{\eta}$-balanced and compute a $\boldsymbol{\eta}$-reshaper $\boldsymbol{g} = (g_i)_{i=1}^k$ for $\mathcal{P}$ where each element has minimal possible x-degree in complexity $\tilde{O}(k\eta_0^{\omega-1} n)$.*

Proof. For any δ, and using the notation of Algorithm 7, then the basis M of I_δ is lower triangular with 1's on the diagonal except on the first row where it has G. Hence $\operatorname{degdet}(M) = \operatorname{degdet}(\phi(I_\delta)) = n$. Using $s = 1$ and $\deg_x b_0 = \deg_x G = n$, the cost follows from Theorem 7.4. □

8 GENERICITY

In this section we prove that on random input, our algorithms will usually display quasi-linear complexity, i.e. that random point sets are usually balanced and that for random R, G then $\langle G, y - R \rangle$ is balanced.

Lemma 8.1. *Let $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n] \in \mathbb{K}^n$ with $\alpha_i \neq \alpha_j$ when $i \neq j$, and let $x, y, y_1, \ldots, y_n$ be $n+2$ distinct variables (transcendental over $\mathbb{K}$). Consider for $s \in \mathbb{Z}_{>0}$ the following matrix A_s:*

$$A_s = \left[V_s \mid DV_s \mid \ldots \mid D^{m-1} V_s \right] \in \mathbb{K}[y_1, \ldots, y_n]^{n \times ms}\,, \quad (3)$$

where D is the diagonal matrix with entries $(y_1, \ldots, y_n)$, and $V_s = [\alpha_i^{j-1}]_{i,j} \in \mathbb{K}^{n \times s}$. Then A_s has full rank $\min(n, ms)$.

Proof. if we specialise y_i to α_i^s for $i = 1, \ldots, n$, we obtain a matrix $\hat{a}_s = [\alpha_i^{j-1}]_{i,j} \in \mathbb{K}^{n \times ms}$ which is the $n \times ms$ Vandermonde matrix over $\boldsymbol{\alpha}$. Since the α_i are distinct, this has full rank $\min(n, ms)$. Hence A_s must also be full rank over $\mathbb{L}$. □

The columns of A_s correspond to monomials $x^i y^j$ for a bivariate polynomial $p \in \mathbb{K}[x, y]$ with x-degree less than s and y-degree less than m. If $p \in \Gamma(\mathcal{P})$ is a bivariate polynomial which vanishes on all points in some point set $\mathcal{P} \subset \mathbb{K}^2$ having distinct x-coordinates, then we can consider $\hat{A}_s = (A_s)_{|y_i \to \beta_i} \in \mathbb{K}^{n \times ms}$ to be the specialisation of the y_i variables to the values β_i. Then the coefficients of p, properly organised as a vector, will be in the right kernel of $\hat{A}_s$.

We now determine the exact row degrees of the Popov basis P_m of $\phi_m(\Gamma_m(\mathcal{P}))$ for a "random" point set. Note that Γ_m has rank m, and so P_m is a full-rank $m \times m$ matrix. The affine transformation λ will be used for modular composition but will be just the identity function for MPE and interpolation.

Lemma 8.2. *Let $\alpha_1, \ldots, \alpha_n \in \mathbb{K}$ be distinct, let $\mathcal{T} \subseteq \mathbb{K}$ a finite subset, and let $\lambda : \mathbb{K}^n \to \mathbb{K}^n$ be an affine transformation. Let $\gamma_1, \ldots, \gamma_n \subseteq \mathcal{T}$ be chosen independently and uniformly at random, set $(\beta_1, \ldots, \beta_n) = \lambda(\gamma_1, \ldots, \gamma_n)$, and set $\mathcal{P} = \{(\alpha_i, \beta_i)\}_{i=1}^n$. Let $m \in \mathbb{Z}_{>0}$ with $\nu_x(\mathcal{P}) < m \leq n$ and let $(d, t) = \text{QUO\_REM}(n, m)$. Let $P_m \in \mathbb{K}[x]^{m \times m}$ be the Popov basis of $\phi_m(\Gamma_m(\mathcal{P}))$. With probability at least $1 - 2nm/|\mathcal{T}|$ then P_m has exactly $m - t$ rows of degree d and t rows of degree $d+1$. In particular $\deg_x P_m \leq d+1$ with probability at least $1 - 2nm/|\mathcal{T}|$.*

Proof. Let $p_1, \ldots, p_m \in \mathbb{K}[x, y]$ be the non-zero bivariate polynomials corresponding to the rows of P_m. As in the proof of Corollary 7.5, then $\operatorname{degdet} P_m = n$ and in particular $\sum_{i=1}^m \deg_x p_i = n$.

Let $A_s \in \mathbb{K}[y_1, \ldots, y_n]^{n \times ms}$ be as in Lemma 8.1, and let $\hat{A}_s = (A_s)_{|y_i \to \beta_i} \in \mathbb{K}^{n \times ms}$. We know $\operatorname{rank}(A_s) = \min(n, ms)$ for any $s \in \mathbb{Z}_{>0}$. Consider first $s = d$: if $\deg_x p_i < d$ for some row i, then the coefficients of p_i properly organised as a vector is in the right kernel of $\hat{A}_d$, and so $\operatorname{rank}(\hat{A}_d) < \operatorname{rank}(A_d) = md$. In particular, if we let the $M \in \mathbb{K}[y_1, \ldots, y_n]$ be one of the non-zero $md \times md$ minors of A_d then $M(\beta_1, \ldots, \beta_n) = M(\lambda(\gamma_1, \ldots, \gamma_n)) = 0$. M has degree at most $m - 1$ in each variable, so the total degree of M is less than nm. We can write $\lambda(z_1, \ldots, z_n) = (\lambda_1, \ldots, \lambda_n)$, where each $\lambda_i \in \mathbb{K}[z_1, \ldots, z_n]$ has total degree 1, and so the composition $M \circ \lambda \in \mathbb{K}[z_1, \ldots, z_n]$ also has total degree less than nm. By Lemma 2.1 then the probability that $M(\lambda(\gamma_1, \ldots, \gamma_n)) = 0$ for independently and uniformly randomly chosen $\gamma_j \in \mathcal{T}$ is at most $nm/|\mathcal{T}|$.

Assume therefore that we are in a case where there are no rows of P_m with degree less than d. If $\deg_x p_i = d$ for some

row i, then the coefficients of p_i as a vector is in the right kernel of $\hat{A}_{d+1} \in \mathbb{K}^{n \times m(d+1)}$. By Lemma 8.1 then A_{d+1} has a right-kernel of dimension exactly $m(d+1) - n = m - t$. Since the rows of P_m are linearly independent over $\mathbb{K}[x]$, and therefore also over $\mathbb{K}$, this gives at most $m - t$ rows of P_m with x-degree exactly d whenever $\operatorname{rank}(\hat{A}_{d+1}) = \operatorname{rank}(A_{d+1})$. We therefore consider $N \in \mathbb{K}[y_1, \ldots, y_n]$ a non-zero $n \times n$ minor of A_{d+1}. Again N has total degree less than nm, and so the probability that $N(\beta_1, \ldots, \beta_n) = N(\lambda(\gamma_1, \ldots, \gamma_n)) = 0$ for independently and uniformly randomly chosen $\gamma_j \in \mathcal{T}$ is at most $nm/|\mathcal{T}|$, and this then bounds the probability that $\operatorname{rank}(\hat{A}_{d+1}) < \operatorname{rank}(A_{d+1})$.

Hence, with probability at least $1 - 2nd/|\mathcal{T}|$ then P_m has no rows of degree less than d and at most $m - t$ rows of degree exactly d. The remaining t rows each have degree at least $d+1$, while their degrees must sum to

$$n - (m-t)d = (md + t) - (m-t)d = t(d+1)\,.$$

Hence they each have degree exactly $d+1$. □

Algorithm 7 for computing reshapers output a $g = y^\eta - \hat{g}$ with $\deg_y \hat{g} < \delta$ satisfying $\deg_x \hat{g} \leq \deg_x P_\delta$, where P_δ is the Popov basis of $\Gamma_\delta(\mathcal{P})$. Lemma 8.2 states that generically we can expect $\deg_x P_\delta = \lfloor \frac{n}{\delta} \rfloor + 1$, and so when $\delta = 2\eta_i - \eta_{i-1} + 1$ in a reshaping sequence, this exactly matches the definition of $\boldsymbol{\eta}$-balanced.

Proposition 8.3. *Let $\alpha_1, \ldots, \alpha_n \in \mathbb{K}$ be distinct, let $\mathcal{T} \subseteq \mathbb{K}$ a finite subset, and let $\lambda : \mathbb{K}^n \to \mathbb{K}^n$ be an affine transformation. Let $\gamma_1, \ldots, \gamma_n \subseteq \mathcal{T}$ be chosen independently and uniformly at random, set $(\beta_1, \ldots, \beta_n) = \lambda(\gamma_1, \ldots, \gamma_n)$, and set $\mathcal{P} = \{(\alpha_i, \beta_i)\}_{i=1}^n$. Let $\boldsymbol{\eta} = (\eta_i)_{i=0}^k$ be a reshaping sequence with $\eta_0 \leq n$ and satisfying the constraints of Lemma 3.5. With probability at least $1 - \frac{n^2 k}{|\mathcal{T}|}$, then $\mathcal{P}$ is $\boldsymbol{\eta}$-balanced.*

The above proposition directly applies to both our MPE and interpolation algorithm on random point sets with unique x-coordinates. There are many formulations depending on the type of randomness one needs over the point sets; the following is a simple example of such over finite fields:

Corollary 8.4. *Let $d_y, n \in \mathbb{Z}_{>0}$ with $d_y \leq n$ and $\mathbb{F}_q$ be a finite field with q elements, and let $\mathcal{P} = \{(\alpha_i, \beta_i)\}_{i=1}^n \subseteq \mathbb{F}_q^2$ be chosen uniformly at random among point sets with cardinality n.*

With probability at least $\left(1 - \frac{n^2}{q}\right)\left(1 - \frac{3n^2(\log_{3/2}(n)+1)}{q}\right)$ over the choice of $\mathcal{P}$, there exists two deterministic algorithms with complexity $\tilde{O}(n)$ with the following behaviour:

1. *Input polynomial $f \in \mathbb{F}_q[x, y]$ with $\deg_y f < d_y$, and output $(f(\alpha_i, \beta_i))_{i=1}^n \in \mathbb{F}_q^n$.*
2. *Input interpolation values $\boldsymbol{\gamma} = (\gamma_i)_{i=1}^n \in \mathbb{F}_q^n$, and output $f \in \mathbb{F}_q[x, y]$ satisfying $f(\alpha_i, \beta_i) = \gamma_i$ for $i = 1, \ldots, n$, as well as $\deg_y f < d_y$ and $\deg_x f \leq cn$ for some constant c which depends only on n and d_y.*

Proof sketch. The probability simply bounds the probability that $\mathcal{P}$ has unique x-coordinates *and* that it is balanced in all the necessary ways. Corollary 3.6 there is an appropriate reshaping sequence of length $\log_{3/2}(n) + 1$ or less. □

We do not make a claim about the genericity of Algorithm 4: indeed, due to the shearing in that algorithm, the arguments of this section do not immediately apply. Lastly, we turn to the genericity of modular composition:

Theorem 8.5. *Let $G \in \mathbb{K}[x]$ be square-free with degree n, let $d_y \in \mathbb{Z}_{>0}$ with $d_y \leq n$ and let $\boldsymbol{\eta}$ be a $(d_y, 1)$-reshaping sequence of length k. Let $\mathcal{T} \subseteq \mathbb{K}$ be a finite subset, and let $R \in \mathbb{K}[x]$ be chosen uniformly at random of degree less than n with coefficients from $\mathcal{T}$. Then $I = \langle G, y - R \rangle$ is $\boldsymbol{\eta}$-balanced with probability at least $1 - \frac{n^2 k}{|\mathcal{T}|}$.*

Proof. Let $\mathbb{L} : \mathbb{K}$ be the splitting field of G, so there exists $\alpha_1, \ldots, \alpha_n \in \mathbb{L}$ such that $G = \prod_{i=1}^n (x - \alpha_i)$, where $n = \deg_x G$. Since G is square-free then the α_i are distinct. Write $R = \sum_{i=0}^{n-1} r_i x^{i-1}$, where the $r_i \in \mathcal{T}$ are chosen independently and uniformly at random, and define the dependent stochastic variables $\beta_i = R(\alpha_i)$ for $i = 1, \ldots, n$. Then the map $\lambda(r_0, \ldots, r_{n-1}) = (\beta_1, \ldots, \beta_n)$ is an $\mathbb{L}$-linear transformation. Consider the point set $\mathcal{P} = \{(\alpha_i, \beta_i)\}_{i=1}^n \subseteq \mathbb{L}^2$. Proposition 8.3 implies that $\mathcal{P}$ is $\boldsymbol{\eta}$-balanced with probability at least $1 - \frac{n^2 k}{|\mathcal{T}|}$. In such a case, for each i there is a polynomial $g_i = y^{\eta_i} + \hat{g}_i \in I_{\mathbb{L}}$ where, $\deg_y \hat{g}_i < 2\eta_i - \eta_{i-1}$ and $\deg_x \hat{g}_i \leq \lfloor \frac{n}{2\eta_i - \eta_{i-1}} \rfloor + 1$, and where $I_{\mathbb{L}} = \langle G, y - R \rangle_{\mathbb{L}[x,y]}$ is the closure of I over $\mathbb{L}[x, y]$. Write g_i over a power basis of $\mathbb{L} : \mathbb{K}$, say $\{1, \zeta, \ldots, \zeta^{s-1}\} \subset \mathbb{L}$, i.e. $g_i = g_{i,0} + \zeta g_{i,1} + \ldots + \zeta^{s-1} g_{i,s-1}$, with $g_{i,j} \in \mathbb{K}[x, y]$. Since $I_{\mathbb{L}}$ is the closure of I over $\mathbb{K}[x, y]$, then $g_i \in I_{\mathbb{L}}$ implies that $g_{i,0} \in I$, and by the shape of g_i then $g_{i,0} = y^{\eta_i} + \hat{g_{i,0}}$ where the x- and y-degree of $g_{i,0}$ satisfy the same bounds as $\hat{g}_i$. Then the tuple $\boldsymbol{g}_0 = (g_{1,0}, \ldots, \hat{g_{k,0}}) \in \mathbb{K}[x, y]^k$ forms a balanced $\boldsymbol{\eta}$-reshaper for I. □

REFERENCES

[1] E. F. Assmus and J. D. Key. 1992. *Designs and Their Codes.* Cambridge University Press.

[2] Peter Beelen, Johan Rosenkilde, and Grigory Solomatov. [n.d.]. Fast Encoding of AG Codes over Cab Curves. http://jsrn.dk/publications.html#2020-ieee-cab-enc Submitted to IEEE Trans. of Information Theory.

[3] R. P. Brent and H. T. Kung. 1978. Fast Algorithms for Manipulating Formal Power Series. *J. ACM* 25, 4 (Oct. 1978), 581–595. https://doi.org/10.1145/322092.322099

[4] David G. Cantor and Erich Kaltofen. 1991. On fast multiplication of polynomials over arbitrary algebras. *Acta Informatica* 28, 7 (July 1991), 693–701. https://doi.org/10.1007/BF01178683

[5] Nicholas Coxon. 2018. Fast systematic encoding of multiplicity codes. *Journal of Symbolic Computation* (Aug. 2018). https://doi.org/10.1016/j.jsc.2018.08.005

[6] R. A. DeMillo and R. J. Lipton. 1978. A Probabilistic Remark on Algebraic Program Testing. 7, 4 (1978), 193–195. https://doi.org/10.1016/0020-0190(78)90067-4

[7] W. Hart, F. Johansson, and S. Pancratz. 2015. FLINT: Fast Library for Number Theory. Version 2.5.2, http://flintlib.org.

[8] David Harvey, Joris Van Der Hoeven, and Grégoire Lecerf. 2017. Faster polynomial multiplication over finite fields. *J. ACM* 63, 6 (Feb. 2017). https://hal.archives-ouvertes.fr/hal-01022757/document

[9] T Kailath. 1980. *Linear Systems.* Prentice-Hall.

[10] K. Kedlaya and C. Umans. 2011. Fast Polynomial Factorization and Modular Composition. *SIAM J. Comput.* 40, 6 (Jan. 2011), 1767–1802. https://doi.org/10.1137/08073408X

[11] Daniel Lazard. 1985. Ideal bases and primary decomposition: case of two variables. *Journal of Symbolic Computation* 1, 3 (1985), 261–270.

[12] François Le Gall. 2014. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th int. symp. on symbolic and algebraic comp.* ACM, 296–303.

[13] Shinji Miura. 1993. Algebraic geometric codes on certain plane curves. *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)* 76, 12 (Jan. 1993), 1–13. https://doi.org/10.1002/ecjc.4430761201

[14] Vincent Neiger, Bruno Salvy, Éric Schost, and Gilles Villard. 2020. Faster modular composition (work in progress).

[15] Vincent Neiger and Thi Xuan Vu. 2017. Computing Canonical Bases of Modules of Univariate Relations. In *International Symposium on Symbolic and Algebraic Computation.* 8. https://hal.inria.fr/hal-01457979/document

[16] Michael Nüsken and Martin Ziegler. 2004. Fast Multipoint Evaluation of Bivariate Polynomials. In *Algorithms – ESA 2004*, Susanne Albers and Tomasz Radzik (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 544–555.
[17] Victor Y. Pan. 1994. Simple Multivariate Polynomial Multiplication. *Journal of Sym. Comp.* 18, 3 (Sept. 1994), 183–186. https://doi.org/10.1006/jsco.1994.1042
[18] Michael S Paterson and Larry J Stockmeyer. 1973. On the number of nonscalar multiplications necessary to evaluate polynomials. *SIAM J. Comput.* 2, 1 (1973), 60–66.
[19] V Popov. 1970. Some properties of the control systems with irreducible matrix-transfer functions. In *Seminar on Diff. Eq. and Dyn. Sys., II.* 169–180.
[20] J. T. Schwartz. 1980. Fast Probabilistic Algorithms for Verification of Polynomial Identities. 27, 4 (1980), 701–717. https://doi.org/10.1145/322217.322225
[21] V. Shoup. 2020. NTL: A Library for doing Number Theory, version 11.4.3. `http://www.shoup.net`.
[22] Joris Van Der Hoeven. 2015. On the complexity of multivariate polynomial division. In *Special Sessions in App. of Computer Algebra*. Springer, 447–458.
[23] Joris van der Hoeven and Grégoire Lecerf. 2018. Modular composition via factorization. *Journal of Complexity* 48 (Oct. 2018), 36–68. https://doi.org/10.1016/j.jco.2018.05.002
[24] Joris van der Hoeven and Grégoire Lecerf. 2019. Fast multivariate multi-point evaluation revisited. *Journal of Complexity* (April 2019). https://doi.org/10.1016/j.jco.2019.04.001
[25] Joris Van Der Hoeven and Éric Schost. 2013. Multi-point evaluation in higher dimensions. *Applicable Algebra in Engineering, Communication and Computing* 24, 1 (2013), 37–52.
[26] Gilles Villard. 2018. On computing the resultant of generic bivariate polynomials. In *Proc. of the 2018 ACM Int. Symp. on Symbolic and Algebraic Comp.* 391–398.
[27] Gilles Villard. 2018. On computing the resultant of generic bivariate polynomials. Presentation at ISSAC 2018. http://www.issac-conference.org/2018/slides/villard-computingresultant.pdf
[28] J. von zur Gathen and J. Gerhard. 2012. *Modern Computer Algebra* (3rd ed.). Cambridge University Press.
[29] R. Zippel. 1979. Probabilistic algorithms for sparse polynomials. In *EUROSAM'79 (LNCS)*, Vol. 72. Springer, 216–226. https://doi.org/10.1007/3-540-09519-5_73

A PROOFS

Proof of Proposition 2.2. Note first that $\mathcal{X}_0$ is the set of all x-coordinates in $\mathcal{P}$, and $\mathcal{X}_0 \supseteq \mathcal{X}_1 \supseteq \ldots \supseteq \mathcal{X}_{v_x-1} \supsetneq \mathcal{X}_{v_x} = \emptyset$.

We may choose b'_i as in (1): For each i and α there are v_α interpolation constraints on $b_{i,\alpha}$, and since $\alpha \notin \mathcal{X}_i$ then $v_\alpha < i$. Hence we can satisfy the interpolation constraints while $\deg_y b_{i,\alpha} < i$. Therefore $\deg_y b_i = i$ for all $i = 0, \ldots, s$, and $\mathsf{LC}_y(b_i) = \prod_{\alpha \in \mathcal{X}_i}(x - \alpha)$.

We claim $b_i \in \Gamma(\mathcal{P})$: indeed, the partial application $b_i(\alpha, y) = 0$ for each $\alpha \in \mathcal{X}_i$, and $b_i(\alpha, \beta) = 0$ for each $\alpha \in \mathcal{X}_0 \setminus \mathcal{X}_i$ and $\beta \in \{\beta \in \mathbb{K} \mid (\alpha, \beta) \in \mathcal{P}\}$. This shows the claim.

Now we should prove that $\Gamma(\mathcal{P})$ is generated by G. We proceed by induction on the $i \in \mathbb{Z}_{\geq 0}$ where $f \in \Gamma(\mathcal{P})$ has y-degree i. We first take $i < v_x$. Consider the partial application $f(\alpha, y) \in \mathbb{K}[y]$ for some $\alpha \in \mathcal{X}_i$. This has y-degree at most i but needs to have at least $i + 1$ roots $\{\beta \in \mathbb{K} \mid (\alpha, \beta) \in \mathcal{P}\}$. Hence $f(\alpha, y) = 0$ for each $\alpha \in \mathcal{X}_i$ and so $\prod_{\alpha \in \mathcal{X}_i}(x - \alpha) \mid f$. Therefore there is a $q \in \mathbb{K}[x]$ such that $\deg_y(f - qb_i) < i$. If $i = 0$ then we must have $f = qb_0$, i.e. $f \in \Gamma(\mathcal{P})$, and so by induction when $0 < i < v_x$, we also get $f \in \Gamma(\mathcal{P})$. For $i \geq v_x$, observe that b_{v_x} is y-monic and hence there is $q \in \mathbb{K}[x, y]$ such that $\deg_y(f - qb_{v_x}) < v_x$. Hence $\langle G \rangle = \Gamma(\mathcal{P})$. In fact, this shows that any element of $\langle G \rangle$ reduces to zero when divided by G with the bivariate division algorithm according to $\prec$, and hence G is a Gröbner basis.

Consider now G'. For each b_i for $i \notin J$ then $\mathsf{LC}_y(b_i) = \mathsf{LC}(b_{i-1})$, and hence removing b_i from G does not change the ideal generated. Hence $\langle G' \rangle = \langle G \rangle = \Gamma(\mathcal{P})$, and G' is also a Gröbner basis. Observe that for $j \in J$ then $\mathsf{LT}_\prec(b_j)$ is not divisible by any $\mathsf{LT}_\prec(b_i)$ for $i \in J \setminus \{j\}$, since $\deg_y b_j < \deg_y b_i$ for $j < i$ and $\deg_x(\mathsf{LC}_y(b_j)) > \deg_x(\mathsf{LC}_y(b_i))$ for $i < j$ since $\mathcal{X}_i \subsetneq \mathcal{X}_j$. Also $\mathsf{LC}_\prec(b_j) = 1$ for all $j = 0, \ldots, v_x$ and hence G' is a minimal Gröbner basis. □

The following is a simplification of Lazard's structure theorem of ideals of bivariate polynomials [11], which we use for the proof of Corollary 2.4:

Proposition A.1. *Let $I \subseteq \mathbb{K}[x, y]$ be an ideal and $G = \{b_1, \ldots, b_s\} \subset \mathbb{K}[x, y]$ a minimal Gröbner basis according to the lex-order $\prec$ with $x \prec y$ with G ordered by increasing $\prec$-order. Then*

(1) $\deg_y b_1 < \deg_y b_2 < \ldots < \deg_y b_s$; *and*
(2) $\mathsf{LC}_y(b_{i+1}) \mid \mathsf{LC}_y(b_i)$ *for* $i = 1, \ldots, s - 1$.

Proof of Corollary 2.4. That $\mathcal{M}$ is an $\mathbb{K}[x]$-module follows simply from I being an ideal of $\mathbb{K}[x, y]$, and in particular closed under addition and multiplication by $\mathbb{K}[x]$-elements, which is therefore inherited by $\mathcal{M}$. Clearly $B \subseteq \mathcal{M}$, and the elements of B all have different y-degree and so are $\mathbb{K}[x]$-linearly independent. Also $|B| = m - \deg_y b_1$, so if B generates $\mathcal{M}$ then it is a basis and hence the rank of $\mathcal{M}$ is $m - \deg_y b_1$. Left is only show that B generates $\mathcal{M}$, so take some $f \in \mathcal{M}$. Since $f \in I$ the multivariate division algorithm using G and the lex-order $\prec$ results in $q_1, \ldots, q_s \in \mathbb{K}[x, y]$ so $f = q_1 b_1 + \ldots + q_s b_s$ with $\deg_y q_i \leq \deg_y f - \deg_y b_i$. Since $\deg_y f < m$ this means $q_{\hat{s}+1} = \ldots = q_s = 0$. Say that in each iteration of the division algorithm, we use the greatest index i for which $\mathsf{LT}_\prec(b_i)$ divides the leading term of the current remainder. This means that no term of $q_i b_i$ is divisible by $\mathsf{LM}_\prec(b_{i+1})$ for any $i < s$. But by Proposition A.1 then $\mathsf{LC}_y(b_{i+1})$ divides $\mathsf{LC}_y(b_i)$, and so if $\deg_y(q_i b_i) \geq \deg_y b_{i+1}$ then $\mathsf{LM}(b_{i+1}) \mid \mathsf{LM}(q_i b_i)$. Consequently $\deg_y q_i < \deg_y b_{i+1} - \deg_y b_i$, and hence f is in the $\mathbb{K}[x]$-span of B. □