

A Simple Algorithm for Trimmed Multipoint Evaluation

Nick Fischer ✉ 

INSAIT, Sofia University “St. Kliment Ohridski”, Bulgaria

Melvin Kallmayer ✉ 

Goethe University Frankfurt, Germany

Leo Wennmann ✉ 

University of Southern Denmark, Odense, Denmark

Abstract

Evaluating a polynomial on a set of points is a fundamental task in computer algebra. In this work, we revisit a particular variant called *trimmed* multipoint evaluation: given an n -variate polynomial with bounded individual degree d and total degree D , the goal is to evaluate it on a natural class of input points. This problem arises as a key subroutine in recent algorithmic results [Dinur; SODA ’21], [Dell, Haak, Kallmayer, Wennmann; SODA ’25]. It is known that trimmed multipoint evaluation can be solved in near-linear time [van der Hoeven, Schost; AAECC ’13] by a clever yet somewhat involved algorithm. We give a *simple* recursive algorithm that avoids heavy computer-algebraic machinery, and can be readily understood by researchers without specialized background.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Algebraic Algorithms, Multipoint Evaluation, Interpolation, LU Decomposition

Digital Object Identifier 10.4230/LIPIcs.ESA.2025.89

Related Version *Full Version*: <https://arxiv.org/abs/2507.00196>

Funding *Nick Fischer*: Partially funded by the Ministry of Education and Science of Bulgaria’s support for INSAIT as part of the Bulgarian National Roadmap for Research Infrastructure. Part of this work was done while the author was affiliated with the Weizmann Institute of Science.

Leo Wennmann: Supported by Dutch Research Council (NWO) project “The Twilight Zone of Efficiency: Optimality of Quasi-Polynomial Time Algorithms” [grant number OCEN.W.21.268].

1 Introduction

One of the most fundamental problems in computer algebra is to efficiently evaluate a polynomial P on some set of points, known as the *multipoint evaluation* problem. Besides its importance as one of the most basic algebraic primitives, this problem finds many further applications in computer algebra (such as modular composition and polynomial factorization) and in algorithm design beyond (in diverse fields such as computational geometry, coding theory and cryptography). The inverse task, to *interpolate* a polynomial from a given set of evaluations, is an equally important primitive.

For univariate polynomials, a textbook algorithm [10, 26] solves the multipoint evaluation problem in near-linear time. This algorithm generalizes to multivariate polynomials [20] (via a simple divide-and-conquer method sometimes referred to as *Yates’ algorithm* [28]), however, only in the restricted setting where the evaluation points form a cartesian *grid*. Specifically, Yates’ algorithm evaluates an n -variate polynomials with individual degree d on all points of a grid

$$Z = \{z_{1,0}, \dots, z_{1,d}\} \times \dots \times \{z_{n,0}, \dots, z_{n,d}\}$$



© Nick Fischer, Melvin Kallmayer, and Leo Wennmann;
licensed under Creative Commons License CC-BY 4.0

33rd Annual European Symposium on Algorithms (ESA 2025).

Editors: Anne Benoit, Haim Kaplan, Sebastian Wild, and Grzegorz Herman; Article No. 89; pp. 89:1–89:11

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

in near-optimal time¹ $O^*((d+1)^n)$. Lifting this restriction to grid points has been the focus of a long and active line of research [18, 23, 15, 25, 6, 24, 2, 1] which, following breakthroughs by Umans [23] and Kedlaya and Umans [15], only recently culminated in an algorithm with almost-optimal running time $(d+1)^{(1+o(1))n} \text{poly}(n, d, \log |\mathbb{F}|)$, for all finite fields $\mathbb{F}$ and for $(d+1)^n$ arbitrary evaluation points, due to Bhargava, Ghosh, Guo, Kumar and Umans [1]. This fully settles the multipoint evaluation problem for *dense* polynomials (over finite fields), but leaves open whether almost-linear time can also be achieved for (some classes of) *sparse* polynomials and evaluation points.

In this paper we focus on one natural such class, called *trimmed* multipoint evaluation, with important applications in the design of exact and parameterized algorithms. Trimmed multipoint evaluation can be solved in near-linear time by an algorithm due to van der Hoeven and Schost [25]. Our contribution is that we make this result accessible to modern algorithm design (beyond computer algebra) by distilling a particularly *simple* recursive algorithm.

Trimmed Multipoint Evaluation

In the trimmed multipoint evaluation problem we focus on the class of n -variate polynomials P with individual degree d and total degree² D . This is a very natural class of polynomials which, for $D \ll nd$, is exponentially sparser than polynomials with just an individual degree bound. As evaluation points we consider triangular subsets of grids Z defined by

$$\{(z_{1,\ell_1}, \dots, z_{n,\ell_n}) : \ell \in \{0, \dots, d\}^n, \ell_1 + \dots + \ell_n \leq D\} \subseteq Z,$$

to which we will informally refer as *trimmed* grids. This is arguably the most naturally matching class of evaluation points. To see this, first observe that the number of relevant (i.e., possibly nonzero) coefficients of P equals exactly the number of grid points. We denote this number by $\binom{n}{\leq D}_d$ (which can be seen as an appropriate generalization of a binomial coefficient called an *extended* binomial coefficient). More importantly, it turns out that the polynomial P is *uniquely* determined by the evaluations on an appropriate trimmed grid, i.e., we can interpolate P given only these evaluations. Trimmed grids are, in a sense, the only sets of evaluation points satisfying this property; see [22].

Applications

Besides being a natural problem in its own right, our interest in trimmed multipoint evaluation stems mainly from its applications in the context of exponential-time algorithms, most notably in a line of research on solving systems of polynomial equations [17, 7, 13, 12]. In this problem the input consists of n -variate polynomials $P_1, \dots, P_m$ over some finite field $\mathbb{F}_q$ with (total) degree Δ , and the task is to test if all polynomials simultaneously vanish at some point $x \in \mathbb{F}_q^n$, i.e., $P_1(x) = \dots = P_m(x) = 0$. The initial breakthrough due to Lokshtanov, Paturi, Tamaki, Williams and Yu [17] established that this problem can be solved exponentially faster than brute-force, in time $(q - \epsilon)^n$ for some $\epsilon > 0$, whenever q and Δ are constant. This inspired several follow-up papers aiming to optimize the precise exponential running time [7, 13, 12]. Trimmed multipoint evaluation shows up as a critical subroutine in the algorithms due to

¹ Here and throughout, we write $O^*(\cdot)$ to omit polynomial factors in n and d .

² Recall that the *individual* degree d of a polynomial is the largest exponent of a variable in a monomial, whereas the *total* degree D is the largest sum of exponents in a monomial. E.g., $X_1^2 X_2$ has individual degree $d = 2$ and total degree $D = 3$.

Dinur [13] (for multilinear polynomials, $d = q - 1 = 1$) and due to Dell, Haak, Kallmayer and Wennmann [12] (for general $d = q - 1 \geq 1$). Solving systems of polynomial equations in turn has many more applications on both the theoretical side – e.g., parity-counting directed Hamiltonian cycles [4] – and the practical side – e.g., the security of several so-called multivariate cryptosystems is based on the hardness of solving quadratic equations [13, 21, 16]. Especially for the latter it could be interesting to achieve simple, implementable algorithms.

In another closely related work, Björklund, Husfeldt, Kaski and Koivisto [5] considered trimmed³ variants of the Zeta and Möbius transforms to develop fast algorithms for various exponential-time graph problems, such as computing the chromatic number for constant-degree-bounded graphs. The Zeta and Möbius transforms can be regarded as special cases of polynomial multipoint evaluation.⁴

State of the Art

The state of the art for trimmed multipoint evaluation, as mentioned before, is a clever algorithm with near-linear running time $O^*\left(\binom{n}{\leq D}_d\right)$ due to van der Hoeven and Schost [25]. Their algorithm is based on the classical concept of Newton interpolation (see e.g. [3]), suitably tailored to the trimmed problem (see also [27, 9, 22, 11, 14] for some more references with a more mathematical point of view). The algorithm also offers two additional benefits: (1) It even solves a strictly more general multipoint evaluation problem on arbitrary downward-closed sets of relevant coefficients and grid points. (2) van der Hoeven and Schost have optimized the lower-order factors achieving an algebraic algorithm with $O(nN \log^2 N \log \log N)$ field operations where $N = \binom{n}{\leq D}_d$. The potentially remaining lower-order improvements even persist in the univariate setting. Thus, all in all, the trimmed multipoint evaluation problem has already been satisfyingly resolved.

The only downside is that van der Hoeven and Schost’s algorithm is arguably somewhat intricate – both in the sense that it can be technically demanding to understand, particularly for researchers outside the computer algebra community, and in that it relies on several textbook algebraic primitives, such as efficient conversions between polynomial bases [3] and the use of truncated Fourier transforms as an implementation detail to achieve further improvements.

Our Contribution

Our focus here is *not* to optimize the running time or generality of this state of the art. Instead, given the many exciting algorithmic applications our contribution is to make van der Hoeven and Schost’s result for trimmed multipoint evaluation accessible to the algorithms community. We design a *simple* recursive algorithm that is teachable to researchers without any background in computer algebra. Moreover, our algorithm does not rely on any black-box algebraic primitives other than Gaussian elimination. As in [25], we also obtain an equally simple algorithm for the interpolation problem.

Our emphasis on simplicity clashes, however, with the additional benefits (1) and (2): (1) It seems hard to obtain a recursive algorithm for the more general problem, and (2) optimizing lower-order factors would involve dealing with more details. Besides, for exponential-time

³ In fact, they consider a more general definition of “trimmed” allowing arbitrary downward-closed sets.

⁴ Indeed, recall that the zeta transform of a function $f : \{0, 1\}^n \rightarrow \mathbb{F}$ is defined as $(f\zeta)(X) = \sum_{Y \subseteq X} f(Y)$, where we identify sets $X \subseteq [n]$ with their indicator vectors $X \in \{0, 1\}^n$. Consider the n -variate polynomial $P(x_1, \dots, x_n) = \sum_{Y \subseteq [n]} f(Y) \prod_{i \in Y} x_i$, and observe that $(f\zeta)(X) = P(X)$. Thus, the zeta transform $f\zeta$ can be read off the evaluations of P on the grid $\{0, 1\}^n$.

algorithms we are typically anyway not bothered with keeping track of polynomial factors. For these reasons we have decided to stick to the simplest version, resulting in a pleasingly simple 8-line algorithm.

Remark on Terminology

The term “trimmed” multipoint evaluation is inspired by the related algorithms for Zeta and Möbius transforms in [5]. It is not standard in the mathematical literature, where one more often encounters terms like “triangular subsets of tensor product grids”. We adopt the “trimmed” terminology here as this paper is primarily intended for the algorithms community.

2 Preliminaries

We write $[n] = \{1, \dots, n\}$. Throughout, let $\mathbb{F}$ be a field and assume that we can evaluate field operations in unit time. For integers n, k, d with $n \geq 0$ and $d \geq 1$ we define the *extended binomial coefficient*

$$\binom{n}{k}_d = |\{\ell \in \{0, \dots, d\}^n : \ell_1 + \dots + \ell_n = k\}|.$$

That is, $\binom{n}{k}_d$ counts the number of multisubsets of $[n]$ with size k and multiplicity at most d , or equivalently, the number of monomials with total degree k and individual degree at most d in an n -variate polynomial. In the same spirit we define the set

$$\binom{[n]}{k}_d = \{\ell \in \{0, \dots, d\}^n : \ell_1 + \dots + \ell_n = k\}.$$

As a shorthand, we write $\binom{n}{\leq k}_d = \sum_{i=0}^k \binom{n}{i}_d$, and similarly define $\binom{[n]}{\leq k}_d = \bigcup_{i=0}^k \binom{[n]}{i}_d$. Further, we rely on the following generalization of Pascal’s triangle; see [8].

► **Lemma 1** (Extended Pascal Triangle). *For $n, d \geq 1$ it holds that*

$$\binom{n}{k}_d = \sum_{j=0}^d \binom{n-1}{k-j}_d.$$

Throughout we refer to a set $Z = \{z_{1,0}, \dots, z_{1,d}\} \times \dots \times \{z_{n,0}, \dots, z_{n,d}\}$ of field elements $z_{i,j}$ as a *grid*. To conveniently refer to the grid points we regularly write $Z_\ell = (z_{1,\ell_1}, \dots, z_{n,\ell_n})$ for $\ell \in \{0, \dots, d\}^n$. In particular, the subset of grid points Z_ℓ where ℓ ranges over $\binom{[n]}{\leq D}_d$ constitutes exactly a trimmed grid as introduced before.

3 Trimmed Multipoint Evaluation and Interpolation

In this section, we present a simple, algebraic algorithm for trimmed multipoint evaluation. As it is usually the case, the same algorithmic approach yields a simple algorithm for trimmed interpolation as well.

3.1 Key Ideas

The problem of evaluating univariate polynomials can be defined via the Vandermonde matrix. For $z_0, \dots, z_d \in \mathbb{F}$, define the (square) Vandermonde matrix $V \in \mathbb{F}^{(d+1) \times (d+1)}$ as

$$V = V(z_0, \dots, z_d) = \begin{pmatrix} 1 & z_0 & z_0^2 & \cdots & z_0^d \\ 1 & z_1 & z_1^2 & \cdots & z_1^d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & z_d & z_d^2 & \cdots & z_d^d \end{pmatrix}.$$

Let $a \in \mathbb{F}^{d+1}$ be the coefficient vector of the univariate polynomial $P(X) = \sum_{i=0}^d a_i X^i$ of degree d . The evaluation of P at all points z_j with $j \in \{0, \dots, d\}$ can be expressed as the matrix-vector product $V \cdot a = y$ (indeed, each entry is of the form $y_j = \sum_{i=0}^d a_i z_j^i = P(z_j)$). Similarly, we can interpolate P from its evaluations $P(z_j)$ using the matrix-vector product $V^{-1} \cdot y = V^{-1} \cdot V \cdot a = a$. The generalization to multivariate polynomials is simple: Let $a \in \mathbb{F}^{(d+1)^n}$ be the coefficient vector of an n -variate polynomial P with individual degree d that we want to evaluate on all grid points in Z . Then the Kronecker product $V(z_{1,0}, \dots, z_{1,d}) \otimes \cdots \otimes V(z_{n,0}, \dots, z_{n,d}) \cdot a = y$ yields the vector $y \in \mathbb{F}^{(d+1)^n}$ of all grid point evaluations. The classical algorithm of Yates [28] allows to compute these evaluations recursively. Notably, this computation does *not* account for the total degree D and automatically results in a running time of $(d+1)^n$ for both multipoint evaluation and interpolation – even in our setting where the total degree D is much smaller than its maximal value nd . Consequently, we need more insights to tailor this approach to the trimmed requirements.

Idea 1: Recursion Scheme. Our goal is to achieve a running time that is linear in $\binom{n}{\leq D}_d$, i.e., in the number of grid points on which we want to evaluate a multivariate polynomial P . As a baseline, we start with a short explanation of Yates' algorithm: write $P(X_1, \dots, X_n) = \sum_{i=0}^d P_i(X_1, \dots, X_{n-1}) \cdot X_n^i$ to obtain $d+1$ many $(n-1)$ -variate polynomials P_i of degree $D-i$, where each P_i can be seen as a coefficient of a univariate polynomial in X_n . By making $d+1$ many recursive calls of size $\binom{n-1}{\leq D}_d$, we compute the evaluations $P_i(Z_{\ell'})$ for all $\ell' \in \binom{[n-1]}{\leq D}_d$. We obtain all $P(Z_{\ell})$ for $\ell \in \binom{[n]}{\leq D}_d$ by evaluating the univariate polynomials $\sum_{i=0}^d P_i(Z_{\ell'}) \cdot X_n^i$ at all grid points $z_{n,i}$. However, this does *not* exploit the degrees $D-i$ of the polynomials P_i in the recursive calls, and hence results in the running time of $(d+1)^n$.

Consider the following identity of the extended binomial coefficient that can be derived from Lemma 1

$$\binom{n}{\leq D}_d = \binom{n-1}{\leq D}_d + \binom{n-1}{\leq D-1}_d + \cdots + \binom{n-1}{\leq D-d}_d.$$

In light of this identity, in order to achieve a running time of $O^*(\binom{n}{\leq D}_d)$ we aim to design an algorithm with $d+1$ recursive calls on $n-1$ variables with total degrees $D, D-1, \dots, D-d$.

Idea 2: LU Decomposition. Our goal is an interleaving of the P_i 's into polynomials Q_j satisfying two properties: (1) All evaluations $Q_j(Z_{\ell'})$ can be recursively computed in a call of size $\binom{n-1}{\leq D-j}_d$; in particular, Q_j must have degree $D-j$. (2) Simultaneously, we need to recover all evaluations $P(Z_{\ell})$ from the recursively computed evaluations $Q_j(Z_{\ell'})$ for $\ell' \in \binom{[n-1]}{\leq D-j}_d$. Due to the degree restriction for the polynomials Q_j in (1) and evaluations $Q_j(Z_{\ell'})$ in (2), this coincides with a matrix factorization of $V = L \cdot U$ where U is upper-triangular and L lower-triangular, i.e., a LU decomposition.

■ **Algorithm 1** TrimmedEvaluation(P).

Input : n -variate polynomial P of individual degree d and total degree D

Output : Evaluations $P(Z_\ell)$ for all $\ell \in \binom{[n]}{\leq D}_d$

- 1 **if** $n = 0$ **then return** the constant P .
- 2 Write $P(X_1, \dots, X_n) = \sum_{i=0}^d P_i(X_1, \dots, X_{n-1}) \cdot X_n^i$.
- 3 Compute the LU decomposition $V(z_{n,0}, \dots, z_{n,d}) = L \cdot U$.
- 4 Compute the vector of polynomials

$$\begin{pmatrix} Q_0 \\ \vdots \\ Q_d \end{pmatrix} = \begin{pmatrix} U_{0,0} & \cdots & U_{0,d} \\ & \ddots & \vdots \\ & & U_{d,d} \end{pmatrix} \cdot \begin{pmatrix} P_0 \\ \vdots \\ P_d \end{pmatrix}.$$

- 5 **for** $j = 0, \dots, d$ **do**
- 6 Recursively call TrimmedEvaluation(Q_j) to evaluate $Q_j(Z_{\ell'})$ for $\ell' \in \binom{[n-1]}{\leq D-j}_d$.
- 7 **for** $\ell' \in \binom{[n-1]}{\leq D}_d$ **do**
- 8 Let $k = \min\{d, D - \sum_{j=1}^{n-1} \ell'_j\}$ and compute all evaluations

$$\begin{pmatrix} P(Z_{(\ell',0)}) \\ \vdots \\ P(Z_{(\ell',k)}) \end{pmatrix} = \begin{pmatrix} L_{0,0} & & \\ \vdots & \ddots & \\ L_{k,0} & \cdots & L_{k,k} \end{pmatrix} \cdot \begin{pmatrix} Q_0(Z_{\ell'}) \\ \vdots \\ Q_k(Z_{\ell'}) \end{pmatrix}.$$

In more detail: Why is the LU decomposition useful in keeping the degrees of the polynomials Q_j “small”? Let $p = (P_0, \dots, P_d)^T$ be the vector of polynomials P_i and consider the product $U \cdot p = (Q_1, \dots, Q_d)^T = q$. Here, the upper triangular structure of U guarantees that each $Q_j = \sum_{i=j}^d U_{j,i} \cdot P_i$ has at most degree $D - j$, since the first non-zero entry in the j -th row of U corresponds to polynomial P_j of degree $D - j$ while all other P_i ’s with (possible) non-zero coefficients have a smaller degree. Additionally, we can also recover $P(Z_\ell)$ from only the recursively computed $Q_j(Z_{\ell'})$ as L has lower triangular structure – appropriately exploiting the fact that $L \cdot q = L \cdot U \cdot p = V \cdot p$.⁵ With these ideas in mind, we are now in the position to present the algorithm in detail.

3.2 Trimmed Multipoint Evaluation

Throughout this section, we will prove the following theorem.

► **Theorem 2** (Trimmed Multipoint Evaluation). *Let P be an n -variate polynomial over $\mathbb{F}$ with individual degree d and total degree D , and let Z be a grid. The evaluations $P(Z_\ell)$ can be computed for all $\ell \in \binom{[n]}{\leq D}_d$ in time $O^*\left(\binom{n}{\leq D}_d\right)$.*

⁵ The analogy to van der Hoeven and Schost’s algorithm is as follows: Broadly speaking, their algorithm is based on efficient transformations between not only the coefficient and evaluation-based representations of polynomials, but also involving a third representation based on so-called *Newton polynomials*. In this terminology, the matrix U performs a basis change from the monomial basis to the Newton basis, and the matrix L performs a basic change from the Newton basis to the evaluation basis.

For the proof of Theorem 2, consider the algorithm `TrimmedEvaluation`. Throughout we assume that the algorithm has access to the grid Z and for simplicity, we omit Z in the recursive calls.

► **Lemma 3** (Correctness of `TrimmedEvaluation`). *Given an n -variate polynomial P over $\mathbb{F}$ with individual degree d and total degree D , `TrimmedEvaluation` correctly computes the evaluations $P(Z_\ell)$ for all $\ell \in \binom{[n]}{\leq D}_d$.*

Proof. Given a polynomial P and a grid Z , we show that `TrimmedEvaluation` correctly computes all evaluations $P(Z_\ell)$ by induction on n .

For $n = 0$, the polynomial P does not depend on any variable. Thus, we correctly return the constant P . For $n \geq 1$, we write $P(X_1, \dots, X_n) = \sum_{i=0}^d P_i(X_1, \dots, X_{n-1}) \cdot X_n^i$ to obtain $d+1$ many $(n-1)$ -variate polynomials $P_0, \dots, P_d$ of degrees $D, \dots, D-d$, respectively. Formally, each polynomial P_i is the coefficient of P viewed as a univariate polynomial in X_n . Let $V := V(z_{n,0}, \dots, z_{n,d})$ be a Vandermonde matrix. It is well-known that V is invertible if and only if all grid points $z_{n,i}$ are distinct, and in this case the LU decomposition $V = L \cdot U$ in Line 3 always exists (see e.g. [19]). Further, let $Q_0, \dots, Q_d$ as computed in Line 4 of `TrimmedEvaluation`. Note that each polynomial $Q_j(X_1, \dots, X_{n-1}) = \sum_{i=j}^d U_{j,i} \cdot P_i(X_1, \dots, X_{n-1})$ has degree $D-j$. Indeed, the upper triangular structure of U guarantees that Q_j has at most degree $D-j$, because the first non-zero entry in the j -th row of U corresponds to the polynomial P_j of degree $D-j$ while all other P_i 's with (possible) non-zero coefficients have a smaller degree.

Calling `TrimmedEvaluation`(Q_j), we recursively compute the evaluations $Q_j(Z_{\ell'})$ for all $\ell' \in \binom{[n-1]}{\leq D-j}_d$ in Lines 5 and 6. Next, we focus on any iteration ℓ' of the loop in Line 7. Let $k = \min\{d, D - \sum_{j=1}^{n-1} \ell'_j\}$, then we show that we correctly compute the evaluations $P(Z_\ell)$. For each $j \in \{0, \dots, k\}$, we have

$$\begin{aligned} \sum_{i=0}^k L_{j,i} \cdot Q_i(Z_{\ell'}) &= \sum_{i=0}^d L_{j,i} \cdot Q_i(Z_{\ell'}) \\ &= \sum_{i=0}^d L_{j,i} \cdot \sum_{m=0}^d U_{i,m} \cdot P_m(Z_{\ell'}) \\ &= \sum_{i=0}^d z_{n,j}^i \cdot P_i(Z_{\ell'}) \\ &= P(Z_{(\ell',j)}), \end{aligned}$$

where the first equality follows from the fact that $L_{j,i} = 0$ whenever $i > k \geq j$. Notably, restricting the computation to k is crucial to ensure that we only use the evaluations $Q_j(Z_{\ell'})$ that we actually computed. This proves that `TrimmedEvaluation` correctly computes the evaluations $P(Z_{(\ell',0)}), \dots, P(Z_{(\ell',k)})$ in Line 8. Since each Z_ℓ can be expressed as some $Z_{(\ell',j)}$ as above, the loop in Line 7 recovers all evaluations $P(Z_\ell)$. As a result, the algorithm `TrimmedEvaluation` correctly computes all evaluations $P(Z_\ell)$. ◀

Lastly, we prove the running time of `TrimmedEvaluation`.

► **Lemma 4** (Running Time of `TrimmedEvaluation`). *The algorithm `TrimmedEvaluation` runs in time $O^*\left(\binom{n}{\leq D}_d\right)$.*

Proof. Let $T(n, d, D)$ be the running time of **TrimmedEvaluation**. The time to compute the LU decomposition in Line 4 and the matrix-vector products in Lines 5 and 9 can be bounded by $\text{poly}(n, d)$. Consequently, the running time of the algorithm *without* the recursive calls in Line 7 is $\binom{n}{\leq D}_d \cdot M(n, d)$ for some function⁶ $M(n, d) = \text{poly}(n, d)$. Thus, the algorithm admits the following recurrence

$$T(n, d, D) \leq \binom{n}{\leq D}_d \cdot M(n, d) + \sum_{i=0}^d T(n-1, d, D-i).$$

By induction on n , we show that $T(n, d, D) \leq \binom{n}{\leq D}_d \cdot M(n, d) \cdot n$. Indeed, it holds that

$$\begin{aligned} T(n, d, D) &\leq \binom{n}{\leq D}_d \cdot M(n, d) + \sum_{i=0}^d T(n-1, d, D-i) \\ &\leq \binom{n}{\leq D}_d \cdot M(n, d) + \sum_{i=0}^d \binom{n}{\leq D-i}_d \cdot M(n, d) \cdot (n-1) \\ &= \binom{n}{\leq D}_d \cdot M(n, d) + \sum_{i=0}^d \sum_{j=0}^{D-i} \binom{n-1}{j}_d \cdot M(n, d) \cdot (n-1) \\ &\leq \binom{n}{\leq D}_d \cdot M(n, d) + \sum_{j=0}^D \binom{n}{j}_d \cdot M(n, d) \cdot (n-1) \\ &= \binom{n}{\leq D}_d \cdot M(n, d) + \binom{n}{\leq D}_d \cdot M(n, d) \cdot (n-1) \\ &= \binom{n}{\leq D}_d \cdot M(n, d) \cdot n \end{aligned}$$

Therefore, **TrimmedEvaluation** runs in time $O^*\left(\binom{n}{\leq D}_d\right)$. ◀

Combining Lemma 3 and Lemma 4 concludes the proof of Theorem 2.

3.3 Trimmed Interpolation

Throughout this section, we show that a polynomial can be (uniquely) interpolated from its evaluations on the trimmed grid points – using essentially the same algorithmic approach as in Section 3.2.

► **Theorem 5 (Trimmed Interpolation).** *Let $\alpha_\ell \in \mathbb{F}$ for $\ell \in \binom{[n]}{\leq D}_d$, and let Z be a grid. The unique n -variate polynomial P with individual degree d and total degree D that satisfies $P(Z_\ell) = \alpha_\ell$ for all $\ell \in \binom{[n]}{\leq D}_d$ can be computed in time $O^*\left(\binom{n}{\leq D}_d\right)$.*

Consider the algorithm **TrimmedInterpolation** for the proof of Theorem 5. As before, we assume that the algorithm has access to the grid Z and for simplicity, we omit Z in the recursive calls.

⁶ In our context, we are content with bounding $M(n, d) = \text{poly}(n, d)$; however, let us remark that following van der Hoeven and Schost [25] we could achieve a dependence on d that is only polylogarithmic. The main insight is that the L- and U-factors of a Vandermonde matrix are sufficiently structured to support matrix-vector operations in time $O(d(\log d)^{O(1)})$.

Algorithm 2 TrimmedInterpolation($((\alpha_\ell)_\ell)$).

Input : Evaluations α_ℓ for all $\ell \in \binom{[n]}{\leq D}_d$

Output : Unique n -variate polynomial P of individual degree d and total degree D such that $P(Z_\ell) = \alpha_\ell$ for all $\ell \in \binom{[n]}{\leq D}_d$

- 1 **if** $n = 0$ **then return** the constant P .
- 2 Compute the LU decomposition $V(z_{n,0}, \dots, z_{n,d})^{-1} = L \cdot U$.
- 3 **for** $\ell' \in \binom{[n-1]}{\leq D-j}_d$ **do**
- 4 Let $k = \min\{d, D - \sum_{j=1}^{n-1} \ell'_j\}$ and compute all evaluations

$$\begin{pmatrix} \beta_{(\ell',0)} \\ \vdots \\ \beta_{(\ell',k)} \end{pmatrix} = \begin{pmatrix} U_{0,0} & \cdots & U_{0,k} \\ & \ddots & \vdots \\ & & U_{k,k} \end{pmatrix} \cdot \begin{pmatrix} \alpha_{(\ell',0)} \\ \vdots \\ \alpha_{(\ell',k)} \end{pmatrix}.$$

- 5 **for** $j = 0, \dots, d$ **do**
- 6 Recursively call **TrimmedInterpolation** $((\beta_{(\ell',j)})_{\ell'})$ for $\ell' \in \binom{[n-1]}{\leq D-j}_d$ to interpolate the $(n-1)$ -variate polynomial Q_j .
- 7 Compute the vector of polynomials

$$\begin{pmatrix} P_0 \\ \vdots \\ P_d \end{pmatrix} = \begin{pmatrix} L_{0,0} & & \\ \vdots & \ddots & \\ L_{d,0} & \cdots & L_{d,d} \end{pmatrix} \cdot \begin{pmatrix} Q_0 \\ \vdots \\ Q_d \end{pmatrix}.$$

- 8 Compute $P(X_1, \dots, X_n) = \sum_{i=0}^d P_i(X_1, \dots, X_{n-1}) \cdot X_n^i$.
-

► **Lemma 6** (Correctness of TrimmedInterpolation). *Given evaluations α_ℓ for $\ell \in \binom{[n]}{\leq D}_d$, TrimmedInterpolation correctly interpolates the unique n -variate polynomial P with individual degree d and total degree D such that $P(Z_\ell) = \alpha_\ell$.*

Proof. Given the evaluations α_ℓ for all $\ell \in \binom{[n]}{\leq D}_d$, we show that **TrimmedInterpolation** correctly interpolates the unique n -variate polynomial P by induction on n . If $n = 0$, simply return the constant P as it does not depend on any variable.

For the remainder of the proof we assume $n \geq 1$. Let $V := V(z_{n,0}, \dots, z_{n,d})$. It is well-known that V is invertible if and only if the grid points $z_{n,0}, \dots, z_{n,d}$ are distinct. Additionally, it implies that the LU decomposition in Line 2, namely $V^{-1} = L \cdot U$, always exists (see e.g. [19]).

In the following, we show that $P(Z_\ell) = \alpha_\ell$ for all $\ell \in \binom{[n]}{\leq D}_d$. Note that each Z_ℓ can be expressed as some $Z_{(\ell',j)}$ with $\ell' \in \binom{[n-1]}{\leq D-j}_d$ and $\sum_{i=1}^{n-1} \ell'_i + j \leq D$. Fix such a pair ℓ' and j , and let $k = \min\{d, D - \sum_{i=1}^{n-1} \ell'_i\}$. It holds that

$$\begin{aligned} P(Z_{(\ell',j)}) &= \sum_{i=0}^d z_{n,j}^i \cdot P_i(Z_{\ell'}) \\ &= \sum_{i=0}^d V_{j,i} \cdot \sum_{m=0}^d L_{i,m} \cdot Q_m(Z_{\ell'}) \end{aligned}$$

$$\begin{aligned}
&= \sum_{i=0}^d V_{j,i} \cdot \sum_{m=0}^d L_{i,m} \cdot \beta_{(\ell',m)} \\
&= \sum_{i=0}^d V_{j,i} \cdot \sum_{m=0}^d L_{i,m} \cdot \sum_{h=0}^k U_{m,h} \cdot \alpha_{(\ell',h)} \\
&= \sum_{i=0}^d V_{j,i} \cdot \sum_{h=0}^k (V^{-1})_{i,h} \cdot \alpha_{(\ell',h)} \\
&= \sum_{h=0}^k I_{j,h} \cdot \alpha_{(\ell',h)} \\
&= \alpha_{(\ell',j)},
\end{aligned}$$

where $I \in \mathbb{F}^{(d+1) \times (d+1)}$ is the identity matrix. The last equality uses the fact that $j \leq k$. Consequently, **TrimmedInterpolation** correctly interpolates the unique n -variate polynomial P with individual degree d and total degree D such that $P(Z_\ell) = \alpha_\ell$. ◀

As the recursion scheme of **TrimmedInterpolation** and **TrimmedEvaluation** are exactly the same, we refer to Lemma 4 for a proof of the running time of **TrimmedInterpolation**. This concludes the proof of Theorem 5.

References

- 1 Vishwas Bhargava, Sumanta Ghosh, Zeyu Guo, Mrinal Kumar, and Chris Umans. Fast multivariate multipoint evaluation over all finite fields. In *63rd IEEE Annual Symposium on Foundations of Computer Science (FOCS 2022)*, pages 221–232. IEEE, 2022. doi:10.1109/FOCS54457.2022.00028.
- 2 Vishwas Bhargava, Sumanta Ghosh, Mrinal Kumar, and Chandra Kanta Mohapatra. Fast, algebraic multivariate multipoint evaluation in small characteristic and applications. In Stefano Leonardi and Anupam Gupta, editors, *54th Annual ACM Symposium on Theory of Computing (STOC 2022)*, pages 403–415. ACM, 2022. doi:10.1145/3519935.3519968.
- 3 Dario Bini and Victor Y. Pan. *Polynomial and matrix computations, 1st Edition*, volume 12 of *Progress in theoretical computer science*. Birkhäuser, 1994. URL: <https://www.worldcat.org/oclc/312012822>.
- 4 Andreas Björklund and Thore Husfeldt. The parity of directed hamiltonian cycles. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 727–735. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.83.
- 5 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Trimmed moebius inversion and graphs of bounded degree. *Theory Comput. Syst.*, 47(3):637–654, 2010. doi:10.1007/S00224-009-9185-7.
- 6 Andreas Björklund, Petteri Kaski, and Ryan Williams. Generalized takeya sets for polynomial evaluation and faster computation of fermionants. *Algorithmica*, 81(10):4010–4028, 2019. doi:10.1007/S00453-018-0513-7.
- 7 Andreas Björklund, Petteri Kaski, and Ryan Williams. Solving systems of polynomial equations over GF(2) by a parity-counting self-reduction. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *LIPIcs*, pages 26:1–26:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICS.ICALP.2019.26.
- 8 Richard C. Bollinger. Extended pascal triangles. *Mathematics Magazine*, 66(2):87–94, 1993. doi:10.1080/0025570x.1993.11996088.

- 9 Carl De Boer and Amos Ron. Computational aspects of polynomial interpolation in several variables. *Mathematics of Computation*, 58(198):705, 1992. doi:10.2307/2153210.
- 10 Allan Borodin and R. Moenck. Fast modular transforms. *J. Comput. Syst. Sci.*, 8(3):366–386, 1974. doi:10.1016/S0022-0000(74)80029-2.
- 11 Abdellah Chkifa, Albert Cohen, and Christoph Schwab. High-dimensional adaptive sparse polynomial interpolation and applications to parametric pdes. *Found. Comput. Math.*, 14(4):601–633, 2014. doi:10.1007/S10208-013-9154-Z.
- 12 Holger Dell, Anselm Haak, Melvin Kallmayer, and Leo Wennmann. Solving polynomial equations over finite fields. In Yossi Azar and Debmalya Panigrahi, editors, *36th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2025)*, pages 2779–2803. SIAM, 2025. doi:10.1137/1.9781611978322.90.
- 13 Itai Dinur. Improved algorithms for solving polynomial systems over $\text{GF}(2)$ by multiple parity-counting. In Dániel Marx, editor, *32nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2021)*, pages 2550–2564. SIAM, 2021. doi:10.1137/1.9781611976465.151.
- 14 Nira Dyn and Michael S. Floater. Multivariate polynomial interpolation on lower sets. *J. Approx. Theory*, 177:34–42, 2014. doi:10.1016/J.JAT.2013.09.008.
- 15 Kiran S. Kedlaya and Christopher Umans. Fast polynomial factorization and modular composition. *SIAM J. Comput.*, 40(6):1767–1802, 2011. doi:10.1137/08073408X.
- 16 Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In Jacques Stern, editor, *International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT 1999)*, volume 1592 of *Lecture Notes in Computer Science*, pages 206–222. Springer, 1999. doi:10.1007/3-540-48910-X_15.
- 17 Daniel Lokshtanov, Ramamohan Paturi, Suguru Tamaki, R. Ryan Williams, and Huacheng Yu. Beating brute force for systems of polynomial equations over finite fields. In Philip N. Klein, editor, *28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2017)*, pages 2190–2202. SIAM, 2017. doi:10.1137/1.9781611974782.143.
- 18 Michael Nüsken and Martin Ziegler. Fast multipoint evaluation of bivariate polynomials. In Susanne Albers and Tomasz Radzik, editors, *12th Annual European Symposium on Algorithms (ESA 2004)*, volume 3221 of *Lecture Notes in Computer Science*, pages 544–555. Springer, 2004. doi:10.1007/978-3-540-30140-0_49.
- 19 Halil Oruç and George M. Phillips. Explicit factorization of the vandermonde matrix. *Linear Algebra and its Applications*, 315(1–3):113–123, 2000. doi:10.1016/S0024-3795(00)00124-5.
- 20 Victor Y. Pan. Simple multivariate polynomial multiplication. *J. Symb. Comput.*, 18(3):183–186, 1994. doi:10.1006/JSC0.1994.1042.
- 21 Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In Ueli M. Maurer, editor, *International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT 1996)*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 1996. doi:10.1007/3-540-68339-9_4.
- 22 Tomas Sauer. Lagrange interpolation on subgrids of tensor product grids. *Math. Comput.*, 73(245):181–190, 2004. doi:10.1090/S0025-5718-03-01557-6.
- 23 Christopher Umans. Fast polynomial factorization and modular composition in small characteristic. In Cynthia Dwork, editor, *40th Annual ACM Symposium on Theory of Computing (STOC 2008)*, pages 481–490. ACM, 2008. doi:10.1145/1374376.1374445.
- 24 Joris van der Hoeven and Grégoire Lecerf. Fast amortized multi-point evaluation. *J. Complex.*, 67:101574, 2021. doi:10.1016/J.JCO.2021.101574.
- 25 Joris van der Hoeven and Éric Schost. Multi-point evaluation in higher dimensions. *Appl. Algebra Eng. Commun. Comput.*, 24(1):37–52, 2013. doi:10.1007/S00200-012-0179-3.
- 26 Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 3rd edition, 2013. doi:10.1017/CB09781139856065.
- 27 Helmut Werner. Remarks on newton type multivariate interpolation for subsets of grids. *Computing*, 25(2):181–191, 1980. doi:10.1007/BF02259644.
- 28 Frank Yates. The design and analysis of factorial experiments, 1937.