An Algebraic Polynomial Evaluation Datastructure for Fields of Large Characteristic

Hiroki Okada* Rachel Player† Simon Pohmann‡

October 28, 2026

Abstract

In this work, we study (multivariate) polynomial evaluation datastructures, i.e. algorithms that can evaluate a multivariate polynomial at any input point, using random access to some precomputed encoding of the polynomial. Multiple constructions have been proposed in the literature, with the work by Bhargava, Ghosh, Guo, Kumar and Umans (2024) achieving optimal datastructure size and query complexity, up to polynomial factors. For $n \in d^{o(1)}$ this is equivalent to having datastructure size $d^{n+o(n)}(dn \log q)^{O(1)}$ and query complexity $(dn \log q)^{O(1)}$ for an n-variate polynomial of total degree d over $\mathbb{F}_q$. However, the algorithm of Bhargava et al. is not algebraic, since it requires lifting prime field elements to $\mathbb{Z}$. An algebraic algorithm would be desirable, as an algebraic model of computation is natural for studying polynomial evaluation and is the standard model in algebraic complexity theory.

One approach to finding algebraic polynomial evaluation datastructures is to use univariate polynomial interpolation to reconstruct the polynomial along algebraic curves. This approach was first made explicit by Bhargava, Ghosh, Kumar and Mohapatra (2023), who used it to build an essentially optimal algebraic polynomial evaluation datastructure for fields of small characteristic $p = \mathrm{char}(\mathbb{F}_q) \in d^{o(1)}$. Over these fields, their construction refutes a previous conjecture by Miltersen. For fields of larger characteristic, it remains an open problem to find an algebraic algorithm with optimal datastructure size and query complexity.

In this work, we propose a new way to construct algebraic curves over prime fields that allow polynomial interpolation: by deriving curves from the base-β representation of (lifts of) prime field elements, we are able to recursively reduce the evaluation problem to points in boxes of decreasing size. This leads to a datastructure whose performance scales much more favourably with p. Concretely, we improve the dependency of the query complexity on p from $p^{O(1)}$ to $d^{2 \log_2 \ln \ln p} \ln p$, and completely remove the dependency on p from the datastructure size. Its query complexity is still slightly super-polynomial in $d \log p$. Nevertheless, a variant of our results is enough to disprove Miltersen's conjecture for all finite fields, including prime fields.

We further show that for a certain class of datastructures, which captures natural generalizations of our approach, the query complexity is lower-bounded by $d^{(1-o(1)) \log_3 \ln \ln p}$. Consequently, within this class, our construction is optimal up to a constant in the exponent.

Keywords. polynomial evaluation, computational complexity, datastructure, polynomial interpolation, Nikodym set, finite field

*KDDI Research, Fujimino, JP (ir-okada@kddi.com) and The University of Tokyo, Bunkyo, JP
†Royal Holloway, University of London, Egham, UK (rachel.player@rhul.ac.uk)
‡Royal Holloway, University of London, Egham, UK (simon@pohmann.de)

1 Introduction

In this work, we investigate the problem of multivariate polynomial evaluation, when given random access to a datastructure encoding the polynomial. This problem was previously studied in [KU11] as a tool for faster univariate polynomial factorization, in [BKW19] for faster computation of fermionants, and in [BGKM23] to prove upper bounds on the rigidity of the Vandermonde determinant. Before that, polynomial evaluation datastructures were studied for univariate polynomials, e.g. in [Mil95; GM07]. Such polynomial evaluation datastructures have recently been applied in cryptography as a tool to construct so-called "Doubly-Efficient Private Information Retrieval" [LMW23] and for cryptanalytic purposes [Udo26].

For an n-variate polynomial of total degree d over a finite field $\mathbb{F}_q$, the algorithm of [KU11] achieves a datastructure of size $O(d \log d)^n (dn \log q)^{O(1)}$ and query complexity $(dn \log q)^{O(1)}$. In contrast to that, for large q, the algorithm of [BKW19] has a query complexity linear in q, but achieves a datastructure of bit size only $(d+1)^{n+2} \log_2 q$. The best currently known algorithm [BGGKU24] combines both ideas, which results in a datastructure size of $d^{n+o(n)}(dn \log q)^{O(1)}$ and a query complexity of $(dn \log q)^{O(1)}$.

While achieving excellent asymptotic complexities, the algorithms of [KU11] and [BGGKU24] have the downside that they cannot be realized in an algebraic model of computation, which can be defined as an algebraic variant of the cell-probe model [Yao81]. Intuitively, this would require that the result of the evaluation is an algebraic expression in elements of the datastructure.

We consider this form of algebraicity a desirable property. Polynomial evaluation is an inherently algebraic problem, so it is natural to study it in an algebraic model of computation, which is also the standard model in areas such as algebraic complexity theory. An algebraic polynomial evaluation datastructure can therefore serve as a primitive for constructions in this setting. For example, if the evaluation points are fixed in advance, it yields an arithmetic circuit for multipoint evaluation, which is efficient whenever the datastructure itself can be computed in quasi-linear time (as is the case for most known constructions, including the ones we propose in this paper). Restricting to algebraic computation is common in algebraic complexity theory, since it means that results extend naturally to infinite fields, and lower bounds become more tractable. This also applies to polynomial evaluation datastructures: constructions are possible even when the polynomial coefficients lie in an infinite field, and additional algebraic proof techniques may help with showing lower bounds (cf. Section 5). It is even conceivable that algebraic datastructures are inherently weaker than non-algebraic ones, which would be a highly interesting separation result.

Indeed, among existing constructions, only [BKW19] and [BGKM23] are algebraic. However, both have a query complexity that is exponential in the size of field elements: it grows linearly with the field size q in [BKW19], and with the characteristic p in [BGKM23]. Finding an algebraic multivariate polynomial evaluation datastructure that has a query complexity $(dn \log q)^{O(1)}$ for all finite fields, including large-characteristic prime fields, has been emphasized as an important open problem in several prior works [KU11; BGKM23; BGGKU24].

1.1 Main results

Our main contribution is a new construction of an algebraic polynomial evaluation datastructure (cf. Thm. 1.2), whose query complexity has a much more favorable dependency on the characteristic p. Concretely, we achieve query complexity $O(d)^{2 \log_2 \ln \ln p + \log_2 \ln d + O(1)} \ln p$ over a prime field $\mathbb{F}_p$. This is not polynomial in the input size, as achievable with non-algebraic datastructures, but a significant improvement over previous algebraic proposals. We further show that if we allow a datastructure size of $d^{O(n)}$, instead of $d^{n+o(n)}$, we can reduce the query complexity

	Algebraic?	Datastructure size	Query complexity	Restriction
[KU11]	×	$O(d \log d)^n (dn \log q)^{O(1)}$	$(dn \log q)^{O(1)}$	
[BGGKU24]	×	$d^{n+o(n)} (dn \log q)^{O(1)}$	$(dn \log q)^{O(1)}$	
[BKW19]	✓	$(d+1)^{n+2}$	$qd^{O(1)}$	$d \mid q-1$
[BGKM23]	✓	$(dp)^n$	$(dnp \log_p q)^{O(1)}$	$q \leq \exp(...(\exp(d)))$ *
Thm. 1.1	✓	$d^{O(n)}$	$O(d)^{2 \log_2 \ln \ln p + O(1)} \ln p$	$q = p$ **
Thm. 1.2	✓	$d^{n+o(n)}$	$O(d)^{2 \log_2 \ln \ln p + \log_2 \ln d + O(1)} \ln p$	$q = p$ **

Table 1: Comparison of the polynomial evaluation datastructures for an n-variate polynomial of total degree $\leq d$ over the finite field $\mathbb{F}_q$ of characteristic p in prior work and Thm. 1.1 and Thm. 1.2 of this work.
* The height of the tower of exponentials is constant (i.e. big-O constants may depend on it).
** The result can be extended to arbitrary q using techniques from [BGKM23].

to $d^{2 \log_2 \ln \ln p + O(1)} \ln p$ (cf. Thm. 1.1). A comparison between our results and prior polynomial evaluation datastructures is given in Table 1.

Our starting point is the same as in [BKW19; BGKM23]: to evaluate a multivariate polynomial at a point, choose a line (or parameterized curve) through this point, and reconstruct the polynomial along this line (or curve). Let the datastructure be the family of evaluations $(f(x))_{x \in N}$, where $f \in \mathbb{F}_q[X_1, ..., X_n]$ is the degree-d input polynomial and N is a carefully chosen subset of $\mathbb{F}_q^n$. To evaluate f at a point $a \in \mathbb{F}_q^n$, pick a line L through a which meets N in at least $d+1$ points. Restricted to this line, $f|_L$ is a univariate polynomial, and can be reconstructed by univariate polynomial interpolation of the points $(x, f(x))$, $x \in L \cap N$. We now find $f(a) = f|_L(a)$, since $a \in L$. This idea already yields a polynomial evaluation datastructure (cf. Prop. 3.1), but to obtain Thm. 1.1 and the results of [BKW19; BGKM23], it is necessary to replace lines by algebraic curves with low-degree parameterizations.

This idea depends crucially on the choice of N, which must admit a suitable line or curve for any point $a \in \mathbb{F}_q^n$. In [BGKM23], N was chosen to be $\mathbb{F}_{q'}^n$ for a subfield $\mathbb{F}_{q'} \subseteq \mathbb{F}_q$. Obviously, this approach cannot be applied when the underlying field is a prime field $\mathbb{F}_q = \mathbb{F}_p$. Instead, we construct N by considering the base-β representation of elements of $\mathbb{F}_p$. Given $a \in \mathbb{F}_p^n$, we can write

$$a = \sum_{i=0}^{r-1} a_i \beta^i, \quad a_i \in \{0, ..., \beta - 1\}^n ,$$

assuming that $\beta^r \geq p$. This directly induces a degree-$(r-1)$ algebraic curve $g_a(\mathbb{F}_p)$, parameterized by

$$g_a : \mathbb{F}_p \to \mathbb{F}_p^n, \quad \tau \mapsto \sum_{i=0}^{r-1} a_i \tau^i .$$

This curve has the nice property that it has many points with "small" coordinates: for $k \in \mathbb{Z}_{\geq 0}$ we find that $g_a(k) \in \{0, ..., \beta r k^{r-1}\}$ (resp. $g_a(k) \in \{0, ..., \beta\}$ for $k = 0$). Hence, we can choose our set N as the box $\{0, ..., \beta(rd)^r\}^n \subseteq \mathbb{F}_p^n$, which ensures that every $g_a(\mathbb{F}_p)$ intersects N in at least $rd+1$ points.

Moreover, this construction can be iterated: points in N have "smaller" coordinates than points in $\mathbb{F}_p^n$, and thus admit a base-β' representation with $\beta' < \beta$ and $r' < r$. This results in a smaller set $N' \subsetneq N$. Iterating this construction $\log_2 \ln \ln p + \log_2 \ln \ln d$ times yields our first main result.

Theorem 1.1. *Let $\mathbb{F} \supseteq \mathbb{F}_p$ with $p > 2$ and let n, d be positive integers. For sufficiently large d, there exists an algebraic polynomial evaluation datastructure for input points in $\mathbb{F}_p^n$ with encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\{0,...,d^{60}\}^n}, \quad f \mapsto (f(a))_a$$

and query complexity

$$C = O(d)^{\log_2 \ln \ln p + \log_2 \ln \ln d + O(1)} \ln p \leq O(d)^{2 \log_2 \ln \ln p + O(1)} \ln p\,.$$

In particular, the datastructure size is $d^{O(n)}$.

The size of this datastructure is polynomial in d^n, but with a very high exponent. By using a base-β decomposition of length $r = 2$ (and so using lines instead of curves), we can decrease the datastructure size to $(4 + 4\epsilon)^n d^n$. This gives our next main result.

Theorem 1.2. *Let $\mathbb{F} \supseteq \mathbb{F}_p$ with $p > 2$, let n, d be positive integers and let $\epsilon > 0$ with $\epsilon \leq \frac{1}{2}$. For sufficiently large d, there exists an algebraic polynomial evaluation datastructure for input points in $\mathbb{F}_p^n$ with encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\{0,...,\lfloor (4+4\epsilon)d \rfloor\}^n}, \quad f \mapsto (f(a))_a$$

and query complexity

$$C = O(d)^{\log_2 \ln \ln p + \log_2 \ln d + \log_2 \ln \ln d - \log_2 \epsilon + O(1)} \ln p\,.$$

In particular, the datastructure size is $d^{n+o(n)}$.

For parameter regimes with very small $n \in o(\log d)$, the construction of Thm. 1.2 even outperforms the non-algebraic algorithm of [BGGKU24] in terms of datastructure size.

1.2 Miltersen's conjecture over prime fields [Mil95]

In [Mil95], Miltersen investigated univariate polynomial evaluation in the cell-probe model [Yao81]. It was shown that if the datastructure size is s, then the query complexity must be at least

$$\min\Big\{d + 1,\ \frac{\ln q - \ln d}{\ln s}\Big\}\,. \tag{1.1}$$

This implies that for $q \geq \exp(d \ln d)$ and $s = d^{O(1)}$, the trivial algorithm that always reads all coefficients of f is optimal. It was conjectured in [Mil95] that the trivial algorithm remains optimal for values of q that are smaller than $\exp(d)$.

When the characteristic of $\mathbb{F}_q$ is $d^{o(1)}$, the construction of [BGKM23] serves as an (algebraic) counterexample to this conjecture. Using our construction from Thm. 1.1, we show that the counterexample can be extended to all finite fields $\mathbb{F}_q$ with $q \leq \exp(d^{1-\epsilon})$. In other words, the lower bound Eq. (1.1) is essentially tight, even when $\mathbb{F}_q = \mathbb{F}_p$ is a prime field.

Corollary 1.3. *Let $\epsilon > 0$ and let q be a prime power with $q \leq \exp(d^{1-\epsilon})$. For $d \geq d_\epsilon$ (where d_ϵ depends only on ϵ), there is an algebraic polynomial evaluation datastructure for input points in $\mathbb{F}_q$ and univariate polynomials of degree d over $\mathbb{F}_q$ with query complexity $\leq \frac{1}{2}d$ and datastructure size $d^{O(1)}$.*

The main observation here is that we can reduce univariate polynomial evaluation in degree d to n-variate polynomial evaluation in degree $d^{1/n}$, by replacing $X^{d^{k/n}}$ with X_{k+1}. At this point, it is left to choose a suitable value of n, and apply Thm. 1.1 to get the result.

1.3 Limitations of the interpolation approach

We conclude by investigating the remaining gap between the query complexity $d^{2\log_2 \ln\ln p + O(1)} \ln p$ of our construction and $(dn \log p)^{O(1)}$ of the non-algebraic datastructure from [KU11]. Both our construction and the construction of [KU11] are built from iterating "reduction steps", but they decrease the set of potential evaluation points at different rates. The reduction step of [KU11] reduces the evaluation of f at $a \in \mathbb{F}_p^n$ to multiple evaluations in $\mathbb{F}_{p'}$ with $p' \approx d \ln p$, while our reduction step (cf. Prop. 3.2) reduces the evaluation of f at $a \in \{0, ..., B\}^n \subseteq \mathbb{F}_p^n$ to multiple evaluations in $\{0, ..., B'\}^n$ with $B' \approx \exp(\sqrt{\ln B \ln\ln B})$. As a result, our construction requires up to $2\log_2 \ln\ln p$ reduction steps, which causes the factor of $d^{2\log_2 \ln\ln p}$ in the query complexity, while [KU11] only requires a constant number of reduction steps.

One might hope that our approach could be improved to also achieve a reduction step of the form $B^n \rightsquigarrow (d \ln B)^n$. However, we show that a certain class of "reduction step" datastructures cannot achieve an exponential decrease in the set of potential evaluation points. We call such datastructures "separable curve datastructures" (formalized in Def. 5.1). The definition is designed to capture the two known constructions of algebraic polynomial datastructures that work over large finite fields: our constructions Thm. 1.1 and Thm. 1.2 over prime fields, and the construction of [BGKM23] over nontrivial finite extension fields.

Proposition 1.4 (Prop. 5.2, Informal)**.** *Any polynomial evaluation datastructure in the "separable curve model" (cf. Def. 5.1) for input points in $P^n \subseteq \mathbb{F}_p^n$ and n-variate polynomials over $\mathbb{F}$ of total degree d has datastructure size*

$$\exp\Big(\Omega\Big(\frac{n \ln(|P|)^{1/3}}{\ln\ln(|P|)^{3/2}}\Big)\Big).$$

When starting from input points in $\mathbb{F}_p^n$ and iterating separable curve reduction steps until the set of potential evaluation points is of size $d^{O(n)}$, as we did in Thm. 1.1, Prop. 5.2 implies that the number of steps is at least $(1 - o(1)) \log_3 \ln\ln p$. Hence, the query complexity of any such iterated construction is lower-bounded by $d^{(1-o(1))\log_3 \ln\ln p}$, and we therefore cannot hope to achieve a reduction step of the form $B^n \rightsquigarrow (d \ln B)^n$ in the separable curve model. This result also shows that in the separable curve model, our construction is optimal up to a constant in the exponent.

While we are not able to give bounds on the performance of general polynomial evaluation datastructures in the algebraic model (this seems to be a very difficult problem), we believe that the restricted lower bound of Prop. 5.2 gives valuable insight into the properties that an efficient algebraic datastructure must have.

2 Preliminaries

In this work, p always denotes a prime, even when this is not explicitly mentioned. Similarly, q always denotes a power of p.

We denote by ln the natural logarithm and by $\log_b$ the logarithm to base b. For a finite set S, we denote its cardinality by $|S|$. For sets $A, B \subseteq \mathbb{F}_q$, we write $A + B$, $A - B$ and AB for the set of sums, differences or products of pairs in $A \times B$, respectively. If $a \in \mathbb{F}_q$ and $B \subseteq \mathbb{F}_q$, we write aB and $a + B$ for the sets $\{a\} \cdot B$ and $\{a\} + B$. For $a \in \mathbb{F}_p$, we consider the shortest lift (usually denoted $\hat{a} \in \mathbb{Z}$) to be the preimage of a under $\mathbb{Z} \hookrightarrow \mathbb{F}_p$ with smallest absolute value (or one of the two preimages with minimal absolute value in case of a tie).

The total degree of a monomial $m(X_1, ..., X_n)$ is the degree of $m(X, ..., X)$, and the total degree of a polynomial is the maximal total degree among its monomials. For any field $\mathbb{F}$ and $d \geq 0$, we

write $\mathbb{F}[X]_{\leq d}$ for the set of univariate polynomials over $\mathbb{F}$ of degree at most d, and $\mathbb{F}[X_1, ..., X_n]_{\leq d}$ for the set of multivariate polynomials in n variables of total degree $\leq d$ over $\mathbb{F}$.

2.1 Problem statement

This work is concerned with datastructures for multivariate polynomial evaluation. When formalizing what we mean by polynomial evaluation datastructure, we must fix a computational model. The most general computational model we consider in this work is polynomial-time algorithms with random access to a precomputed datastructure.

Definition 2.1. *A polynomial evaluation datastructure for n-variate polynomials of total degree d over $\mathbb{F}_q$ consists of an encoding map $\psi : \mathbb{F}_q[X_1, ..., X_n]_{\leq d} \to \{0,1\}^s$, and a polynomial-time algorithm that, given $a \in \mathbb{F}_q^n$, computes $f(a)$ given random access to $\psi(f)$.*

We call s the size of the datastructure, and the maximum number of bits read by the evaluation algorithm as the query complexity of the datastructure.

Most of the algorithms in this work actually work in a more restricted model, which is algebraic in the following sense.

Definition 2.2. *An algebraic polynomial evaluation datastructure for input points in $S \subseteq \mathbb{F}_q^n$ and n-variate polynomials of total degree d over a field $\mathbb{F} \supseteq \mathbb{F}_q$ consists of a size-$s^{O(1)}$ arithmetic circuit ψ that maps the coefficients of a polynomial in $\mathbb{F}[X_1, ..., X_n]_{\leq d}$ to a vector in $\mathbb{F}^s$, and a polynomial-time algorithm that, given $a \in S$, outputs a list of indices $i_1, ..., i_l$ and an arithmetic circuit e such that $e(\psi(f)_{i_1}, ..., \psi(f)_{i_l}) = f(a)$. The circuits are considered to be division-free and may contain constants from $\mathbb{F}$.*

We call s the size of the datastructure, and l the query complexity of the datastructure.

This model is essentially a uniform and algebraic version of the cell-probe model [Yao81]. Intuitively, the cell probe model allows random access to constant memory, and measures performance solely in terms of the number of memory accesses. However, we don't explicitly use the cell-probe model in this work, and refer the reader to [Yao81] for the details.

Note that representing an element of $\mathbb{F}$ takes $\lceil \log_2 |\mathbb{F}| \rceil$ bits, and so the query complexity and datastructure size differ by a factor of $\log_2 |\mathbb{F}|$[1] between Def. 2.1 (which counts bits) and Def. 2.2 (which counts field elements). This should be kept in mind when comparing datastructures in the standard and the algebraic model.

2.2 Existing constructions

The prior algebraic constructions of [BKW19] and [BGKM23] are conceptually the most similar to our work. Both follow the approach of reconstructing the polynomial along a parameterized curve, but with a very different choice of curve.

In [BKW19], the authors generalize a construction for so-called Kakeya sets [MT04; SS08; KMW13], obtaining a set K such that for any $a \in \mathbb{F}_q^n$, there exists a parameterized curve $C_a : \tau \mapsto a\tau^r + c_{r-1}^{(a)}\tau^{r-1} + ... + c_1^{(a)}\tau + c_0^{(a)}$ that is contained in K. It is assumed that the input polynomial f is homogeneous. Storing the evaluation of f at every point $k \in K$ makes it possible to compute $f(a)$ for any input point a, using the values $k \in C_a$.

[1] This assumes that $\mathbb{F}$ is finite, which is required for Def. 2.1, but not for Def. 2.2.

Theorem 2.3 (Lemma 2 of [BKW19]). *Let $\mathbb{F} \supseteq \mathbb{F}_q$ be a field, let n, d be positive integers with $d \mid q-1$. Then there exists a set $K \subseteq \mathbb{F}_q^n$ of size $|K| \leq (d+1)^{n+1}$ and an algebraic polynomial evaluation datastructure for input points in $\mathbb{F}_q^n$ with encoding map*[2]

$$\mathbb{F}[X_1, ..., X_n]^{\text{hom}}_{=d} \to \mathbb{F}^K, \quad f \mapsto (f(a))_a$$

and query complexity $q-1$.

To see that this is an instantiation of the "reconstruction along a curve" idea, embed into projective n-space, and project from the origin $\pi : \mathbb{P}^n_{\mathbb{F}_q} \setminus \{o\} \to H_\infty$. This transforms K into a set $N := \pi(K \setminus \{o\})$ and each parameterized curve C_a with $o \notin C_a$ into a (projective) degree-r parameterized curve[3] $\tilde{C}_a$ with $[0:a] \in \tilde{C}_a$ and $\tilde{C}_a \setminus \{[0:a]\} \subseteq N$.

The construction of [BGKM23] is based on a field extension $\mathbb{F}_{q^r}/\mathbb{F}_q$ of degree $r \geq 1$. Let $\theta \in \mathbb{F}_{q^r}$ be a generator, i.e. $\mathbb{F}_{q^r} = \mathbb{F}_q[\theta]$. Then every element $a \in \mathbb{F}_{q^r}$ can be written as an $\mathbb{F}_q$-linear combination of $1, \theta, ..., \theta^{r-1}$; thus, for a point $a \in \mathbb{F}_{q^r}^n$, we find a curve $C_a : \tau \mapsto c_r^{(a)}\tau^r + ... + c_1^{(a)}\tau + c_0^{(a)}$ that contains a. This curve also contains q points in $\mathbb{F}_q^n$, namely the points $C_a(\mathbb{F}_q)$. Therefore, if $q \geq rd+1$, it is possible to derive the evaluation $f(a)$ of a polynomial $f \in \mathbb{F}_{q^r}[X_1, ..., X_n]_{\leq d}$ from the evaluations $f(x)$, for $x \in N := \mathbb{F}_q^n$. This construction can be iterated until $q = \Theta(d)$. Using this idea, the authors of [BGKM23] obtain the following datastructure.

Theorem 2.4 (Follows from Theorem 7.1 of [BGKM23]). *Let $\mathbb{F} \supseteq \mathbb{F}_q$ be a field, let n, d be positive integers and assume that q is bounded by $\exp(\exp(...(\exp(d))))$, where the height of this tower is constant. Then there exists an algebraic polynomial evaluation datastructure for input points in $\mathbb{F}_q^n$ and n-variate polynomials of total degree d over $\mathbb{F}$ with datastructure size $(dp)^{n+o(n)}(dnp\log_p q)^{O(1)}$ and query complexity $(dnp\log_p q)^{O(1)}$.*

In this work, we will also require the following variant of their result. While using the same ideas as [BGKM23], it requires slightly different parameters and a new analysis, so we include a proof in Section A.

Corollary 2.5. *Let n, d be positive integers, let $\tilde{q} > d$ be a power of p and let q be a power of p with $q \leq \exp(\exp(...(\exp(d))))$, where the height of this tower is constant. Assume $\mathbb{F}$ is a field containing $\mathbb{F}_q$ and $\mathbb{F}_{\tilde{q}}$. Then there exists an algebraic polynomial evaluation datastructure for points in $\mathbb{F}_q^n$ with encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\mathbb{F}_{\tilde{q}}^n}, \quad f \mapsto (f(a))_a$$

and query complexity $(dn\ln\ln q)^{O(1)}\log_p q$.

3 Our construction

In this section, we present our construction of a polynomial evaluation datastructure that efficiently supports fields of large characteristic. We follow the "reconstruct along curves" paradigm that was previously used by [BKW19; BGKM23], but with a new choice of curves and set N.

We assume the input points are defined over a prime field $\mathbb{F}_q = \mathbb{F}_p$, since finite extensions thereof are already handled by Cor. 2.5. Our first result is based on interpolating along lines.

[2] Here $\mathbb{F}[X_1, ..., X_n]^{\text{hom}}_{=d}$ denotes the set of homogeneous degree-d polynomials.

[3] To make this precise, define an (affine, degree-r, parameterized) curve to be a map $C : \mathbb{F}_q \to \mathbb{F}_q^m$ defined by degree-r polynomials, and a projective (degree-r, parameterized) curve to be a map $C : \mathbb{P}^1(\mathbb{F}_q) \to \mathbb{P}^m(\mathbb{F}_q)$ induced by $m+1$ bivariate homogeneous polynomials of degree r without a common zero in $\mathbb{F}_q$. Relations like $a \in C$ or $C \subseteq N$ should be understood in terms of the image of C.

Proposition 3.1. *Let $\mathbb{F} \supseteq \mathbb{F}_p$ be a field, let n, d, and B be positive integers and assume $p > d$. There exists an algebraic polynomial evaluation datastructure for points in $\{0, ..., B\}^n \subseteq \mathbb{F}_p^n$ with encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\{0,...,\lfloor 2\sqrt{dB}\rfloor\}^n}, \quad f \mapsto (f(a))_a$$

and query complexity $d + 1$.

Proof. Define $\beta := \lceil \sqrt{dB} \rceil$. Consider the algorithm that proceeds as follows. On input $a \in \{0, ..., B\}^n$, compute $m := \lfloor a/\beta \rfloor$ and $x := a - \beta m$ (where the rounding $\lfloor a/\beta \rfloor$ is component-wise). Next, compute the points $x_k := x + km$ for $k \in \{0, ..., d\}$. The value $f(a)$ is then given by univariate polynomial interpolation of the points $(k, f(x_k))$, i.e.

$$f(a) = \sum_{k=0}^{d} f(x_k) \prod_{l \neq k} \frac{\beta - l}{k - l} .$$

In particular, this is a polynomial expression in the $f(x_k)$. Furthermore, all the points $x_0, ..., x_d$ satisfy

$$\|x_k\|_\infty \leq \|x\|_\infty + d\|m\|_\infty \leq (\beta - 1) + dB/\beta \leq \beta - 1 + \sqrt{dB} \leq 2\sqrt{dB}$$

and are thus within $\{0, ..., \lfloor 2\sqrt{dB} \rfloor\}^n$. □

This method gives a reduction $B^n \rightsquigarrow (2\sqrt{dB})^n$, meaning that we reduce the evaluation of f on a set of size B^n to the evaluation of f on a set of size $(2\sqrt{dB})^n$. The map $B \mapsto 2\sqrt{dB}$ has a fixed point $4d$, thus we can use this result recursively to construct a datastructure of size $\approx (4d)^n$, as we will do later in Thm. 1.2. However, when used on its own, this technique will require a large number of recursive steps, and will thus incur a high query complexity. We can improve on this by interpolating along curves instead of lines.

Proposition 3.2. *Let $\mathbb{F} \supseteq \mathbb{F}_p$ be a field, let n, d, B, and r be positive integers with $p > rd \geq 2$. Define*

$$B' := \lfloor \exp\left(\tfrac{1}{r} \ln B + r \ln d + r \ln r\right) \rfloor .$$

Then there exists an algebraic polynomial evaluation datastructure for points in $\{0, ..., B\}^n \subseteq \mathbb{F}_p^n$ with encoding map

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\{0,...,B'\}^n}, \quad f \mapsto (f(a))_a$$

and query complexity $rd + 1$. For sufficiently large d and $B \geq d^4$, we have the special cases

$$\ln B \leq d, \ r = \left\lceil \sqrt{\frac{\ln B}{\ln d}} \right\rceil \ \Rightarrow \quad B' \leq \exp\left(3\sqrt{\ln B \ln d}\right) ;$$

$$\ln B \geq d, \ r = \left\lceil \sqrt{\frac{\ln B}{\ln \ln B}} \right\rceil \ \Rightarrow \quad B' \leq \exp\left(3\sqrt{\ln B \ln \ln B}\right) .$$

Proof. Define $\beta := \lfloor B^{1/r} \rfloor + 1$. Consider the algorithm that proceeds as follows. For a given $a \in \{0, ..., B\}^n$, define $a_0, ..., a_{r-1}$ as the base-β representation of a, i.e.

$$a_0, ..., a_{r-1} \in \{0, ..., \beta - 1\}^n \subseteq \mathbb{F}_p^n \quad \text{such that } a = \sum_{i=0}^{r-1} a_i \beta^i .$$

This works, since $\beta > B^{1/r}$, thus $\beta^r > B$. Next, consider the curve C_a parameterized by

$$g_a : \mathbb{F}_p \to \mathbb{F}_p^n, \quad \tau \mapsto \sum_{i=0}^{r-1} a_i \tau^i .$$

With this definition, we find that $f(g_a(X)) \in \mathbb{F}[X]$ is the unique univariate polynomial interpolating the $(\tau, f(g_a(\tau)))$, hence we can compute $f(g_a(\beta)) = f(a)$ using polynomial interpolation.

It is left to show that all queried points are indeed within $\{0, ..., B'\}^n$. For any $\tau \in \{0, ..., rd\}$, we have

$$\begin{aligned} \|g_a(\tau)\|_\infty &\leq \sum_{i=0}^{r-1} \|a_i\|_\infty \tau^i \leq \sum_{i=0}^{r-1} (\beta - 1)(rd)^i \leq (\beta - 1)(rd)^r \\ &= \exp\left(\ln(\beta - 1) + r \ln d + r \ln r\right) \\ &\leq \exp\left(\tfrac{1}{r} \ln B + r \ln d + r \ln r\right) . \end{aligned}$$

The special cases hold as follows.

- If $\ln B \leq d$ and $B \geq d^9$, then $r \leq \sqrt{d}$ for large d, so $\ln r \leq \frac{1}{2} \ln d$ and $\ln B' \leq \sqrt{\ln B \ln d} + \frac{3}{2}(\sqrt{\ln B / \ln d} + 1) \ln d \leq 3\sqrt{\ln B \ln d}$, since $\sqrt{\ln B \ln d} \geq 3 \ln d$.
- If $d^4 \leq B < d^9$, then $r \leq 3$ and $\ln B' \leq \sqrt{\ln B \ln d} + 3 \ln d + 3 \ln 3 \leq 3\sqrt{\ln B \ln d}$ for large d, since $\sqrt{\ln B \ln d} \geq 2 \ln d$.
- If $\ln B \geq d$, then $\ln d \leq \ln \ln B$ and $\ln r \leq \frac{1}{2} \ln \ln B$, so $\ln B' \leq \sqrt{\ln B \ln \ln B} + \frac{3}{2}(\sqrt{\ln B / \ln \ln B} + 1) \ln \ln B \leq 3\sqrt{\ln B \ln \ln B}$, since $\sqrt{\ln B \ln \ln B} \geq 3 \ln \ln B$ for large d. □

This method achieves a reduction $B^n \rightsquigarrow \exp(3\sqrt{\ln B \ln d})^n$, which is much better than the previous reduction $B^n \rightsquigarrow (2\sqrt{dB})^n$. We now show how to construct a polynomial evaluation datastructure by using this method recursively.

Theorem 1.1. *Let $\mathbb{F} \supseteq \mathbb{F}_p$ with $p > 2$ and let n, d be positive integers. For sufficiently large d, there exists an algebraic polynomial evaluation datastructure for input points in $\mathbb{F}_p^n$ with encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\{0,...,d^{60}\}^n}, \quad f \mapsto (f(a))_a$$

and query complexity

$$C = O(d)^{\log_2 \ln \ln p + \log_2 \ln \ln d + O(1)} \ln p \leq O(d)^{2 \log_2 \ln \ln p + O(1)} \ln p .$$

In particular, the datastructure size is $d^{O(n)}$.

Proof. We show that the algorithm that recursively applies Prop. 3.2 until the points are in $\{0, ..., d^{60}\}^n$ satisfies the claim. Assume wlog that $p > d^{60}$ and define

$$B_0 := p, \quad B_{i+1} = \begin{cases} \exp(3\sqrt{\ln B_i \ln d}) & \text{if } \ln B_i \leq d \\ \exp(3\sqrt{\ln B_i \ln \ln B_i}) & \text{otherwise} \end{cases}$$

In detail, the algorithm proceeds as follows. On input $a \in \{0, ..., \lfloor B_i \rfloor\}^n$, use Prop. 3.2 to write $f(a)$ as a polynomial expression in multiple $f(a_j)$ for $a_j \in \{0, ..., \lfloor B_{i+1} \rfloor\}^n$. Indeed, the prerequisites are satisfied, since $d^{60} < \lfloor B_i \rfloor \leq p$. If $\lfloor B_{i+1} \rfloor \leq d^{60}$, we are done. Otherwise, for each a_j, use

the algorithm recursively to write each $f(a_j)$ as a polynomial expression in multiple $f(a'_k)$ with $a'_k \in \{0, ..., d^{60}\}^n$. Composing the polynomial expressions gives the result.

First, we show that $\tilde{k} := \log_2 \ln \ln p + \log_2 \ln \ln d - \log_2 \ln d$ steps suffice to reach $\ln B_i \le d$. Assume $\tilde{k} \ge 0$ and let k be the smallest integer with $\ln B_k \le d$, so we have to show $k \le \lceil \tilde{k} \rceil$. To this end, let $x_i := \ln(\ln B_i / \ln \ln B_i)$ for $i \le k$, and observe that

$$x_{i+1} \le \ln\Big(\frac{3\sqrt{\ln B_i \ln \ln B_i}}{\ln(3\sqrt{\ln B_i \ln \ln B_i})}\Big) \le \ln(6\sqrt{\ln B_i / \ln \ln B_i}) = \ln(6e^{x_i/2}) = \tfrac{1}{2}x_i + \ln 6\,.$$

Thus

$$x_i \le 2^{-i}x_0 + \ln 6 \sum_{j=0}^{i-1} 2^{-j} \le 2^{-i}x_0 + 2\ln 6\,.$$

Hence $k \ge \tilde{k}$ implies

$$x_{\lceil \tilde{k} \rceil} \le 2^{-\lceil \tilde{k} \rceil}x_0 + 2\ln 6 \le \frac{x_0 \ln d}{\ln \ln p \ln \ln d} + 4 \le \frac{\ln d \ln \ln p - \ln d \ln \ln \ln p}{\ln \ln p \ln \ln d} + 4 \le \frac{\ln d}{\ln \ln d} + 4\,.$$

For sufficiently large d, we conclude that $k \le \lceil \tilde{k} \rceil$.

Next, we show that $\tilde{l} = \log_2 \ln(\ln B_k / \ln d)$ further recursive steps suffice to reach d^{60}. Let l be the smallest integer such that $B_{k+l} \le d^{60}$, so we have to show $l \le \lceil \tilde{l} \rceil$. To this end, let $x'_i := \ln(\ln B_{k+i} / \ln d)$ for $i \le l$, and observe that

$$x'_{i+1} \le \ln(3\sqrt{\ln B_{i+k} \ln d} / \ln d) = \ln(3\sqrt{\ln d \cdot e^{x'_i} / \ln d}) = \ln(3e^{x'_i/2}) = \tfrac{1}{2}x'_i + \ln 3\,.$$

Thus

$$x'_i \le 2^{-i}x'_0 + \ln 3 \sum_{j=0}^{i-1} 2^{-j} \le 2^{-i}x'_0 + 2\ln 3\,.$$

Hence $l \ge \tilde{l}$ implies

$$x'_{\lceil \tilde{l} \rceil} \le 2^{-\lceil \tilde{l} \rceil}x'_0 + 2\ln 3 \le \frac{x'_0}{\ln(\ln B_k / \ln d)} + 2\ln 3 = 1 + 2\ln 3 \le 4\,.$$

Now $\exp(e^4 \ln d) \le d^{60}$, and we conclude $l \le \lceil \tilde{l} \rceil$.

Finally, we can show that the total number of points is bounded by C:

$$\begin{aligned}
&\prod_{i=0}^{k-1} \left(d \left\lceil \sqrt{\ln B_i / \ln \ln B_i} \right\rceil + 1\right) \prod_{i=0}^{l-1} \left(d \left\lceil \sqrt{\ln B_{k+i} / \ln d} \right\rceil + 1\right) \\
&\qquad \le (3d)^{k+l} \exp\Big(\tfrac{1}{2}\sum_{i=0}^{k-1} \ln\Big(\frac{\ln B_i}{\ln \ln B_i}\Big) + \tfrac{1}{2}\sum_{i=0}^{l-1} \ln\Big(\frac{\ln B_{i+k}}{\ln d}\Big)\Big) \\
&\qquad \le (3d)^{k+l} \exp\Big(\tfrac{1}{2}\sum_{i=0}^{k-1} x_i + \tfrac{1}{2}\sum_{i=0}^{l-1} x'_i\Big) \\
&\qquad \le (3d)^{k+l} \exp\Big((k+l)\ln 6 + \tfrac{1}{2}\sum_{i=0}^{k-1} 2^{-i}x_0 + \tfrac{1}{2}\sum_{i=0}^{l-1} 2^{-i}x'_0\Big) \\
&\qquad \le (3d)^{k+l} \exp\Big((k+l)\ln 6 + x_0 + x'_0\Big) \\
&\qquad \le (18d)^{k+l} d \ln p = (18d)^{k+l+O(1)} \ln p\,.
\end{aligned}$$

This is bounded by C, since $k + l \le \lceil \tilde{k} \rceil + \lceil \tilde{l} \rceil \le \log_2 \ln \ln p + \log_2 \ln \ln d + O(1)$. □

While Prop. 3.2 gives a larger decrease in the size of B per recursive step, it stabilizes at $d^{O(1)}$. If we want to have a datastructure of size $O(d)^n$, we also have to use Prop. 3.1.

Theorem 1.2. *Let $\mathbb{F} \supseteq \mathbb{F}_p$ with $p > 2$, let n, d be positive integers and let $\epsilon > 0$ with $\epsilon \leq \frac{1}{2}$. For sufficiently large d, there exists an algebraic polynomial evaluation datastructure for input points in $\mathbb{F}_p^n$ with encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\{0,...,\lfloor(4+4\epsilon)d\rfloor\}^n}, \quad f \mapsto (f(a))_a$$

and query complexity

$$C = O(d)^{\log_2 \ln \ln p + \log_2 \ln d + \log_2 \ln \ln d - \log_2 \epsilon + O(1)} \ln p\,.$$

In particular, the datastructure size is $d^{n+o(n)}$.

Proof. Assume wlog $p > 4d$. Using the previous algorithm Thm. 1.1, it suffices to provide an algorithm that writes $f(a)$ for $a \in \{0, ..., d^{60}\}^n$ as a polynomial in $f(a_i)$ for $(d+1)^{\lceil \tilde{k} \rceil}$ points $a_i \in \{0, ..., \lfloor(4+4\epsilon)d\rfloor\}^n$, where $\tilde{k} = \log_2 \ln d - \log_2(\epsilon/2) + \log_2 60$.

As before, we construct this algorithm by recursively applying Prop. 3.1. Define

$$B_0 := d^{60},\ B_{i+1} = 2\sqrt{dB_i} \quad \text{and} \quad x_i = B_i/(4d)\,.$$

In detail, the algorithm proceeds as follows. On input $a \in \{0, ..., \lfloor B_i \rfloor\}^n$, use Prop. 3.1 to write $f(a)$ as a polynomial expression in multiple $f(a_j)$ for $a_j \in \{0, ..., \lfloor B_{i+1} \rfloor\}^n$. If $B_{i+1} \leq (4+4\epsilon)d$, we are done. Otherwise, for each a_j, use the algorithm recursively to write each $f(a_j)$ as a polynomial expression in multiple $f(a'_k)$ with $a'_k \in \{0, ..., \lfloor(4+4\epsilon)d\rfloor\}^n$. Composing the polynomial expressions gives the result.

First, we show that the number of performed recursive steps is at most $\lceil \tilde{k} \rceil$. Let k be the smallest index where $B_k \leq (4+4\epsilon)d$, so we have to show $k \leq \lceil \tilde{k} \rceil$. Observe that for $i < k$, we have

$$x_{i+1} = 2\sqrt{dB_i}/(4d) = 2\sqrt{4d^2 x_i}/(4d) = \sqrt{x_i}\,.$$

Thus $x_i \leq x_0^{2^{-i}}$. Hence $k \geq \tilde{k}$ implies

$$\begin{aligned} B_{\lceil \tilde{k} \rceil} =& 4dx_{\lceil \tilde{k} \rceil} \leq 4dx_0^{2^{-\lceil \tilde{k} \rceil}} \leq 4d \exp\Big(\frac{\ln x_0}{60 \ln d} 2^{\log_2(\epsilon/2)}\Big) \\ \leq& 4d \exp\Big(\frac{\ln B_0 - \ln(4d)}{60 \ln d}(\epsilon/2)\Big) \leq 4d \exp(\epsilon/2) \leq 4d(1+\epsilon) \end{aligned}$$

and we conclude $k \leq \lceil \tilde{k} \rceil$. Finally, the query complexity increases by a factor of $(d+1)^{\lceil \tilde{k} \rceil}$ compared to Thm. 1.1, thus is bounded as claimed. □

4 Miltersen's conjecture over prime fields [Mil95]

It was conjectured in [Mil95] that an evaluation datastructure for input points in $\mathbb{F}_q$ and univariate polynomials of degree d over $\mathbb{F}_q$ with datastructure size $d^{O(1)}$ must have query complexity d, unless $\ln q \ll d$. An algebraic counterexample to this conjecture was given by [BGKM23], but only for fields $\mathbb{F}_q$ of characteristic $d^{o(1)}$. By using Thm. 1.1, we extend this to all finite fields of size at most $\exp(d^{1-\epsilon})$.

Corollary 1.3. *Let $\epsilon > 0$ and let q be a prime power with $q \le \exp(d^{1-\epsilon})$. For $d \ge d_\epsilon$ (where d_ϵ depends only on ϵ), there is an algebraic polynomial evaluation datastructure for input points in $\mathbb{F}_q$ and univariate polynomials of degree d over $\mathbb{F}_q$ with query complexity $\le \frac{1}{2}d$ and datastructure size $d^{O(1)}$.*

Proof. We show the claim for prime fields $\mathbb{F}_q = \mathbb{F}_p$. Assume wlog $p > 2$. The general case then follows by combining this counterexample with Cor. 2.5.

Define $n := \lceil \ln d / \ln \ln d \rceil$ and $d' := \lceil \ln d \rceil$. In particular, $(d')^n \ge d$ and so there is a multivariate polynomial $f'(X_1, ..., X_n)$ of degree $\le d'$ in every variable such that

$$f'(X, X^{d'}, X^{(d')^2}, ..., X^{(d')^{n-1}}) = f(X)\,.$$

Clearly, f' has total degree at most nd'. Furthermore, to evaluate f at a given point $a \in \mathbb{F}_p$, it is sufficient to evaluate f' at $a' = (a, a^{d'}, ..., a^{(d')^{n-1}})$.

To evaluate f' at a', use Thm. 1.1. The required number of cells is bounded as

$$(nd')^{60n} \le n^{60n}(d')^{60n} \le \exp\Big(60\tfrac{\ln d}{\ln\ln d}(\ln\ln d - \ln\ln\ln d) + 60\tfrac{\ln d}{\ln\ln d}\ln\ln d + O(\ln d)\Big) \le d^{O(1)}\,.$$

Furthermore, the query complexity can be bounded as

$$\begin{aligned} O(nd')^{2\log_2 \ln\ln p + O(1)} \ln p &\le \exp\left(2(\ln n + \ln d' + O(1))(\log_2 \ln\ln p + O(1))\right) d^{1-\epsilon} \\ &\le \exp\Big(O(\ln\ln d)\log_2 \ln d^{1-\epsilon}\Big)\, d^{1-\epsilon} \\ &\le \exp\Big(O(\ln\ln d)^2\Big)\, d^{1-\epsilon}\,. \end{aligned}$$

This is sublinear, and thus $\le \frac{1}{2}d$ for sufficiently large d. □

5 Limitations of the interpolation approach

In light of [KU11], one might hope that the methods we developed in Section 3 could be improved to achieve a reduction step $B^n \rightsquigarrow (d \ln B)^n$. In this section we show that, over prime fields, this is not possible for any "reduction step" datastructures falling in a particular algorithmic framework, which we now formalize.

Definition 5.1. *Let $r \ge 1$, $\theta \in \mathbb{F}_q$ and $\mathcal{A}, \mathcal{B} \subseteq \mathbb{F}_q$ with $|\mathcal{B}| \ge rd+1$. We call $(\mathcal{A}, \mathcal{B}, \theta, r)$ a* separable curve datastructure *for input points in P^n, where $P \subseteq \mathbb{F}_q$, if every $a \in P$ can be written as $a = \tilde{g}(\theta)$ for some $\tilde{g} \in \mathcal{A}[X]_{\le r}$. Its* evaluation set *is*

$$N_{\mathcal{A},\mathcal{B},r} := \{\tilde{g}(b) \;\mid\; \tilde{g} \in \mathcal{A}[X]_{\le r},\ b \in \mathcal{B}\} \subseteq \mathbb{F}_q\,.$$

For $n \ge 2$ and a field $\mathbb{F} \supseteq \mathbb{F}_q$, it induces a polynomial evaluation datastructure for input points in P^n with query complexity $rd+1$ and encoding map

$$\mathbb{F}[X_1, ..., X_n]_{\le d} \to \mathbb{F}^{N^n_{\mathcal{A},\mathcal{B},r}}, \quad f \mapsto (f(a))_a\,.$$

We also use the term separable curve datastructure *to refer to the induced polynomial evaluation datastructure.*

This framework captures all known constructions of algebraic polynomial datastructures that work over large finite fields. Indeed, both our constructions Prop. 3.1 and Prop. 3.2 are separable curve datastructures, for example with $\mathcal{A} = \{0, ..., \beta-1\}$ and $\mathcal{B} = \{0, ..., rd\}$ in the case of Prop. 3.2. Moreover, the construction of [BGKM23] is also of this form, with $\mathcal{A} = \mathcal{B} = \mathbb{F}_{q'}$ for a subfield $\mathbb{F}_{q'} \subseteq \mathbb{F}_q$.

For a separable curve datastructure, we present the following lower bound on its datastructure size, which is proved in Section 5.1.

Proposition 5.2. *Let p be a prime and n, d positive integers. Consider a separable curve datastructure $(\mathcal{A}, \mathcal{B}, \theta, r)$ for input points in $P^n \subseteq \mathbb{F}_p^n$. Then it has datastructure size*

$$|N^n_{\mathcal{A},\mathcal{B},r}| \geq \exp\Big(\Omega\Big(\frac{n \ln(|P|)^{1/3}}{\ln\ln(|P|)^{3/2}}\Big)\Big)\,.$$

While Def. 5.1 permits arbitrary finite base fields, we emphasize that this proposition only applies if the underlying field $\mathbb{F}_q$ is a prime field $\mathbb{F}_q = \mathbb{F}_p$. In particular, the construction of [BGKM23] is a separable curve datastructure, but is not subject to Prop. 5.2, since it requires $\mathbb{F}_q$ to have a nontrivial subfield $\mathbb{F}_{q'} \subsetneq \mathbb{F}_q$. Indeed, its datastructure size is below the lower bound of Prop. 5.2.

When starting from input points in $\mathbb{F}_p^n$ and iterating separable curve reduction steps until the set of potential evaluation points is of size $d^{O(n)}$, as we did in Thm. 1.1, Prop. 5.2 implies that the number of steps is at least $(1 - o(1)) \log_3 \ln\ln p$. Hence, the query complexity of any such iterated construction is lower-bounded by $d^{(1-o(1))\log_3 \ln\ln p}$.

Definition 5.3. *An* iterated separable curve datastructure *is a sequence of separable curve datastructures $(\mathcal{A}_0, \mathcal{B}_0, \theta_0, r_0), ..., (\mathcal{A}_k, \mathcal{B}_k, \theta_k, r_k)$, where $(\mathcal{A}_0, \mathcal{B}_0, \theta_0, r_0)$ is a datastructure for input points in $\mathbb{F}_q^n$ and $(\mathcal{A}_i, \mathcal{B}_i, \theta_i, r_i)$ is a datastructure for input points in $N^n_{\mathcal{A}_{i-1},\mathcal{B}_{i-1},r_{i-1}}$ for all $i \geq 1$. For $n \geq 2$ and a field $\mathbb{F} \supseteq \mathbb{F}_q$, it induces a polynomial evaluation datastructure for input points in $\mathbb{F}_q^n$ with query complexity $\prod_i (r_i d + 1)$ and encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{N^n_{\mathcal{A}_k,\mathcal{B}_k,r_k}}, \quad f \mapsto (f(a))_a\,.$$

We also use the term iterated separable curve datastructure *to refer to the induced polynomial evaluation datastructure.*

Corollary 5.4. *Let $\mathbb{F}_q = \mathbb{F}_p$ be a prime field. An iterated separable curve datastructure for input points in $\mathbb{F}_p^n$ with datastructure size $d^{O(n)}$ has query complexity at least $d^{\log_3 \ln\ln p - \log_3 \ln\ln d - O(1)}$. In particular, if $\ln\ln p \in \ln\ln(d)^{\omega(1)}$, then the query complexity is lower-bounded by $d^{(1-o(1))\log_3 \ln\ln p}$.*

Proof. Assume the iterated separable curve datastructure is $(\mathcal{A}_i, \mathcal{B}_i, \theta_i, r_i)_{0 \leq i \leq k}$ and define $B_i := |N_{\mathcal{A}_i,\mathcal{B}_i,r_i}|$. Its datastructure size is $B_k^n \in d^{O(n)}$.

Set $B_{-1} := p$ and observe that Prop. 5.2 implies $B_i \geq \exp(C \ln(B_{i-1})^{1/3} / \ln\ln(B_{i-1})^{3/2})$ for a constant $0 < C < 1$ and all $0 \leq i \leq k$. By induction, we obtain

$$B_i \geq \exp(C^{3(1-1/3^i)/2} \ln(p)^{1/3^i} \ln\ln(p)^{-9(1-1/3^i)/4})\,.$$

Together with the assumption $B_k^n \in d^{O(n)}$, we conclude $C^{3/2} \ln(p)^{1/3^k} / \ln\ln(p)^{9/4} \in O(\ln d)$, and thus $k \geq \log_3(\ln\ln p / \ln\ln d) - O(1)$. Hence, the query complexity is lower-bounded as $\prod_i (r_i d + 1) \geq d^{\log_3 \ln\ln p - \log_3 \ln\ln d - O(1)}$. □

5.1 Proof of Prop. 5.2

Before we present the proof of Prop. 5.2, we need a lower bound on the size of the set of evaluations for a given class of polynomials, which is established in the following result.

Theorem 5.5 ([Web25]). *If $(d+1)^{3d/2} < p$ and $b \in \mathbb{F}_p^*$ is not a root of unity of order $\leq \gamma$, $\gamma \geq 7$, then*

$$\left|\left\{\sum_{i=0}^{d} a_i b^i \;\middle|\; a_i \in \{0,1\}\right\}\right| \geq \min\left\{2^{\frac{\gamma}{4\ln\ln\gamma}}, 2^{\sqrt{\frac{2d}{\ln 3 \ln(3d)^3}}}\right\}.$$

We include the proof of [Web25] here with permission from the author. We start with two lemmas.

Lemma 5.6. *Let $f, g \in \mathbb{Z}[X]$, $f \neq 0$ be polynomials with coefficients in $\{-1,0,1\}$ of degree at most d, and let $h \in \mathbb{Z}[X]$ be a polynomial dividing f. Then $|\mathrm{Res}(g,h)| \leq (d+1)^{3d/2}$.*

Proof. Since h divides f, the leading coefficient of h must be ± 1. Thus

$$\begin{aligned}|\mathrm{Res}(g,h)| = \Big|\prod_{\substack{z\in\mathbb{C}\\ h(z)=0}} g(z)\Big| &\leq \prod_{\substack{z\in\mathbb{C}\\ h(z)=0}} (d+1)\max\{1,|z|\}^d \\ &\leq ((d+1)M(h))^d \leq ((d+1)M(f))^d \leq (d+1)^{3d/2},\end{aligned}$$

where $M(h)$ denotes the Mahler measure of h and the last inequality is by Landau's inequality. □

Lemma 5.7. *Let $f, g \in \mathbb{Z}[X]$, $f \neq 0$ be polynomials with coefficients in $\{-1,0,1\}$ of degree at most d, and let $h \in \mathbb{Z}[X]$ be a polynomial dividing f. If $h(b) \equiv g(b) \equiv 0 \bmod p$ for some $b \in \mathbb{F}_p$ and $p > (d+1)^{3d/2}$, then $\gcd_{\mathbb{Z}[X]}(h,g)(b) \equiv 0 \bmod p$.*

Proof. By repeatedly dividing h by $\gcd_{\mathbb{Z}[X]}(h,g)$ (in $\mathbb{Z}[X]$), we can factor $h = h'h_0h_1\cdots h_i$ with $h_i \mid \ldots \mid h_1 \mid h_0 = \gcd_{\mathbb{Z}[X]}(h,g)$ and $\gcd_{\mathbb{Z}[X]}(h',g) = 1$. By Lemma 5.6, we find $|\mathrm{Res}(h',g)| \leq (d+1)^{3d/2} < p$. In particular, $\mathrm{Res}(h',g) \not\equiv 0 \bmod p$, and thus $\gcd_{\mathbb{F}_p[X]}(h',g) = 1$. However, this implies that $X - b$ must be a factor of some $h_j \bmod p$, thus a factor of $\gcd_{\mathbb{Z}[X]}(h,g) \bmod p$. □

This shows that the $\mathbb{Z}[X]$-gcd of all polynomials $g \in \mathbb{Z}[X]$ of degree at most d with coefficients in $\{-1,0,1\}$ and $g(b) \equiv 0 \bmod p$ still has a root at b. In particular, we get the following corollary.

Corollary 5.8. *If $p > (d+1)^{3d/2}$ is a prime and $b \in \mathbb{F}_p$, then there exists a monic polynomial $h \in \mathbb{Z}[X]$ such that $h(b) \equiv 0 \bmod p$, which divides (over $\mathbb{Z}[X]$) all polynomials $g \in \mathbb{Z}[X]$ of degree at most d with coefficients in $\{-1,0,1\}$ and $g(b) \equiv 0 \bmod p$.*

Finally, we also require the following result from [Vou96].

Theorem 5.9 ([Vou96, Corollary 2]). *Let $\alpha \in \mathbb{C}$ be a non-zero algebraic number of degree $k \geq 2$ that is not a root of unity. Then*

$$\ln M(\alpha) > \frac{2}{\ln(3k)^3},$$

where $M(\alpha)$ denotes the Mahler measure of (the minimal polynomial of) α.

Together with Cor. 5.8, we can now give the main proof.

Proof of Thm. 5.5. Let $h \in \mathbb{Z}[X]$ be the polynomial given by Cor. 5.8. Then

$$\left|\left\{ \sum_{i=0}^{d} a_i b^i \bmod p \;\middle|\; a_i \in \{0,1\}\right\}\right| \geq \left|\left\{ \sum_{i=0}^{d} a_i X^i \bmod h \;\middle|\; a_i \in \{0,1\}\right\}\right|$$

since $\sum_i a_i b^i = \sum_i a_i' b^i$ implies, by Cor. 5.8, that h divides $\sum_i (a_i - a_i')X^i$. Let $d' := \deg(h)$. All polynomials $\sum_{i=0}^{d'-1} a_i X^i$ with $a_i \in \{0,1\}$ are distinct, so if $\sqrt{2d/\ln 3/\ln(3d)^3} \leq d'$, we are done. Otherwise, h must have an irreducible factor $h' \in \mathbb{Z}[X]$, which is not a cyclotomic polynomial; indeed, if $\Phi_k \mid h$ with $\Phi_k(b) \equiv 0 \bmod p$, then $d' \geq \phi(k) \geq \frac{1}{4}k/\ln\ln k \geq \frac{1}{4}\gamma/\ln\ln\gamma$, so $2^{d'} \geq 2^{\gamma/(4\ln\ln\gamma)}$ and we are done. Let $h' \in \mathbb{Z}[X]$ be that factor, and let $d'' := \deg(h') \leq d'$.

Since h' is monic, Thm. 5.9 now implies that for the root z of h' with maximum absolute value, we have $\ln(|z|^{d''}) > 2/\ln(3d'')^3$, thus $d''\ln(3d'')^3 \ln|z| \geq 2$. Therefore, we find $|z|^m \geq 3$ for $m = \lceil \frac{1}{2}\ln 3 \; d''\ln(3d'')^3 \rceil$. Then all the sums

$$\sum_{i=0}^{\lfloor d/m \rfloor} a_i z^{mi}, \quad a_i \in \{0,1\}$$

are distinct (as complex numbers), so

$$\begin{aligned}\left|\left\{ \sum_{i=0}^{d} a_i b^i \bmod p \;\middle|\; a_i \in \{0,1\}\right\}\right| &\geq \left|\left\{ \sum_{i=0}^{d} a_i X^i \bmod h \;\middle|\; a_i \in \{0,1\}\right\}\right| \\ &\geq 2^{\lfloor d/m \rfloor + 1} \geq 2^{d/m} \geq 2^{d/\sqrt{\frac{\ln 3 \ln(3d)^3 d}{2}}} \\ &= 2^{\sqrt{\frac{2d}{\ln 3 \ln(3d)^3}}},\end{aligned}$$

where we used that $d'' \leq d' \leq \sqrt{2d/\ln 3/\ln(3d)^3}$, thus for $d \geq 2$, we have $m \leq \sqrt{d \ln 3 \ln(3d)^3/2}$ (the claim is trivially satisfied for $d = 1$). □

Proof of Prop. 5.2. First, find that

$$|P|^n \leq \left|\left\{g(\theta) \;\middle|\; g \in \mathcal{A}[X]^n_{\leq r}\right\}\right| \leq \left|\mathcal{A}[X]^n_{\leq r}\right| \leq |\mathcal{A}|^{n(r+1)}$$

and thus $|\mathcal{A}| \geq \exp(\ln|P|/(2r))$.

We now make a case distinction on the size of r. If $r \leq \ln(|P|)^{2/3}$, then

$$|N_{\mathcal{A},\mathcal{B},r}| \geq |\mathcal{A}| \geq \exp\left(\tfrac{\ln|P|}{2r}\right) \geq \exp\left(\Omega\left(\ln(|P|)^{1/3}\right)\right).$$

Next, consider the case $r \geq \ln(|P|)^{2/3}$. Clearly we require $|\mathcal{A}| \geq 2$, thus there are two distinct elements $a_1 \neq a_2$ in $\mathcal{A}$. Furthermore, $|\mathcal{B}| \geq rd + 1$ implies that there is $b \in \mathcal{B} \setminus \{0\}$ which is not a root of unity of order $\leq \gamma := \sqrt{r}$. Let $r' := \lceil \ln(|P|)^{2/3} \rceil \leq r$ and observe that for a_1, a_2, b we have

$$N_{\mathcal{A},\mathcal{B},r} \supseteq (a_1 - a_2)S + \sum_{i=0}^{r} a_2 b^i, \quad S := \left\{\sum_{i=0}^{r'} c_i b^i \;\middle|\; c_i \in \{0,1\}\right\}.$$

By choice of r', we have $(r'+1)^{3r'/2} \leq \exp(O(\ln(|P|)^{2/3}\ln\ln|P|)) \leq \exp(O(\ln(p)^{2/3}\ln\ln p)) < p$ for sufficiently large p. Hence, by Thm. 5.5, we obtain

$$|S| \geq \exp\Big(\min\Big\{\frac{\sqrt{r}\ln 2}{4\ln\ln\sqrt{r}}, \sqrt{\frac{2r'\ln(2)^2}{\ln 3\ln(3r')^3}}\Big\}\Big) \geq \exp\Big(\Omega\Big(\sqrt{\frac{\ln(|P|)^{2/3}}{\ln\ln(|P|)^3}}\Big)\Big) = \exp\Big(\Omega\left(\frac{\ln(|P|)^{1/3}}{\ln\ln(|P|)^{3/2}}\right)\Big).$$

Since $(a_1 - a_2) \in \mathbb{F}_p$ is a unit this implies the claim. □

5.2 Discussion

Ignoring factors of $\ln d$ and $\ln \ln p$, the lower bound in Prop. 5.2 is of the form $\exp(\ln(|P|)^{1/3})$, while Prop. 3.2 reduces B only to $\exp(\ln(B)^{1/2})$. We believe that the $\frac{1}{3}$ is an artifact of the proof and arises as a result of the square root in Thm. 5.5. We conjecture that the

$$2^{\sqrt{\frac{2d}{\ln 3 \ln(3d)^3}}}$$

term of Thm. 5.5 can be strengthened to $\exp(d/\ln(d)^{O(1)})$. In this case, our construction Prop. 3.2 would be optimal except for a polynomial factor in $\ln d$ and $\ln \ln p$.

The framework of Def. 5.1 is deliberately designed to capture the constructions of this work and of [BGKM23]. It is not a general lower bound for algebraic polynomial evaluation datastructures. Nevertheless, we consider it a natural framework for polynomial evaluation datastructures over large fields. Indeed, the only known algebraic construction not captured by Def. 5.1 is the construction of [BKW19], which seems to be inherently restricted to small fields. Whether different techniques can yield algebraic polynomial evaluation datastructures over large fields that circumvent our lower bound remains a question for future research.

Acknowledgements

We thank Simon Blackburn and Alan Lauder for providing feedback on our ideas. We also thank Simon Blackburn for giving feedback on an earlier draft of this paper.

Simon Pohmann was supported by the EPSRC and the UK Government as part of the Centre for Doctoral Training in Cyber Security at Royal Holloway, University of London (EP/S021817/1). Parts of this research were conducted while Simon Pohmann was an invited researcher at KDDI Research, supported by the International Exchange Program of the National Institute of Information and Communication Technologies (NICT).

References

[BGGKU24] Vishwas Bhargava, Sumanta Ghosh, Zeyu Guo, Mrinal Kumar, and Christopher Umans. "Fast multivariate multipoint evaluation over all finite fields". In: *Journal of the ACM* 71.3 (2024), pp. 1–32. DOI: 10.1145/3652025.

[BGKM23] Vishwas Bhargava, Sumanta Ghosh, Mrinal Kumar, and Chandra Kanta Mohapatra. "Fast, algebraic multivariate multipoint evaluation in small characteristic and applications". In: *Journal of the ACM* 70.6 (2023), pp. 1–46. DOI: 10.1145/3625226.

[BKW19] Andreas Björklund, Petteri Kaski, and Ryan Williams. "Generalized Kakeya sets for polynomial evaluation and faster computation of fermionants". In: *Algorithmica* 81.10 (2019), pp. 4010–4028. DOI: 10.1007/s00453-018-0513-7.

[GM07] Anna Gál and Peter Bro Miltersen. "The cell probe complexity of succinct data structures". In: *Theoretical Computer Science* 379.3 (2007), pp. 405–417. DOI: 10.1016/j.tcs.2007.02.047.

[KMW13] Gohar Kyureghyan, Peter Müller, and Qi Wang. "On the size of Kakeya sets in finite vector spaces". In: *The Electronic Journal of Combinatorics* 20.3, P36 (2013). DOI: 10.37236/3190.

[KU11] Kiran S. Kedlaya and Christopher Umans. "Fast polynomial factorization and modular composition". In: *SIAM Journal on Computing* 40.6 (2011), pp. 1767–1802. DOI: 10.1137/08073408X.

[LMW23] Wei-Kai Lin, Ethan Mook, and Daniel Wichs. "Doubly efficient private information retrieval and fully homomorphic RAM computation from ring LWE". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC)*. 2023, pp. 595–608. DOI: 10.1145/3564246.3585175.

[Mil95] Peter Bro Miltersen. "On the cell probe complexity of polynomial evaluation". In: *Theoretical Computer Science* 143.1 (1995), pp. 167–174. DOI: 10.1016/0304-3975(95)80032-5.

[MT04] Gerd Mockenhaupt and Terence Tao. "Restriction and Kakeya phenomena for finite fields". In: *Duke Mathematical Journal* 121.1 (2004), pp. 35–74. DOI: 10.1215/S0012-7094-04-12112-8.

[SS08] Shubhangi Saraf and Madhu Sudan. "An improved lower bound on the size of Kakeya sets over finite fields". In: *Analysis & PDE* 1.3 (2008), pp. 375–379. DOI: 10.2140/apde.2008.1.375.

[Udo26] Aleksei Udovenko. *Solving the supersingular isogeny problem in time $p^{2/5+o(1)}$ using bivariate multipoint evaluation*. Cryptology ePrint Archive, Paper 2026/1575. 2026. URL: https://eprint.iacr.org/2026/1575.

[Vou96] Paul Voutier. "An effective lower bound for the height of algebraic numbers". In: *Acta Arithmetica* 74.1 (1996), pp. 81–95. DOI: 10.4064/aa-74-1-81-95.

[Web25] Daniel Weber. *"Polynomial evaluation estimates" in the spirit of sum-product estimates*. Version 2025-06-21. Answer by user Daniel Weber (https://mathoverflow.net/users/498835/daniel-weber). MathOverflow. 2025. URL: https://mathoverflow.net/q/496472.

[Yao81] Andrew Chi-Chih Yao. "Should tables be sorted?" In: *Journal of the ACM* 28.3 (1981), pp. 615–628. DOI: 10.1145/322261.322274.

A Additional Proofs

In this appendix, we prove the following statement, which is a variant of Theorem 7.1 of [BGKM23], and can be proved using the same ideas.

Corollary 2.5. *Let n, d be positive integers, let $\tilde{q} > d$ be a power of p and let q be a power of p with $q \leq \exp(\exp(...(\exp(d))))$, where the height of this tower is constant. Assume $\mathbb{F}$ is a field containing $\mathbb{F}_q$ and $\mathbb{F}_{\tilde{q}}$. Then there exists an algebraic polynomial evaluation datastructure for points in $\mathbb{F}_q^n$ with encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\mathbb{F}_{\tilde{q}}^n}, \quad f \mapsto (f(a))_a$$

and query complexity $(dn \ln \ln q)^{O(1)} \log_p q$.

To prove this statement, we require the following reduction step.

Lemma A.1. *Let n, d be positive integers and let $\tilde{q}$ be a power of p. Let $q = \tilde{q}^r$ and $q' = \tilde{q}^k$ be powers of $\tilde{q}$ with $q' > d(r-1)$ and $\mathbb{F}_q, \mathbb{F}_{q'} \subseteq \mathbb{F}$. There exists an algebraic polynomial evaluation datastructure for points in $\mathbb{F}_q^n$ with encoding map*

$$\mathbb{F}[X_1, ..., X_n]_{\leq d} \to \mathbb{F}^{\mathbb{F}_{q'}^n}, \quad f \mapsto (f(a))_a$$

and query complexity $d(r-1)+1$.

In particular, if $\tilde{q} > d$ *and* $r \geq 2$*, we have the special case*

$$k = \left\lfloor \log_{\tilde{q}}(rd-d) \right\rfloor + 1 \quad \Rightarrow \quad q' \leq \min\{\tilde{q}dr, q/\tilde{q}\}\,.$$

Proof. Consider the compositum of $\mathbb{F}_q$ and $\mathbb{F}_{q'}$; it is generated by an element θ over $\mathbb{F}_{q'}$, thus denote it by $\mathbb{F}_{q'}[\theta]$. Its degree is at most r, so for any $a \in \mathbb{F}_{q'}[\theta]$, there are $a_0, ..., a_{r-1} \in \mathbb{F}_{q'}$ with

$$a = \sum_{i=0}^{r-1} a_i \theta^i\,.$$

Thus, for any $a \in \mathbb{F}_{q'}[\theta]^n$, we can consider the curve C_a parameterized by

$$g_a : \mathbb{F}_{q'}[\theta] \to \mathbb{F}_{q'}[\theta]^n\,, \quad \tau \mapsto \sum_{i=0}^{r-1} a_i \tau^i\,, \quad a_i \in \mathbb{F}_{q'}^n\,.$$

Since $|\mathbb{F}_{q'}| = q' > d(r-1)$, we find that $f(g_a(X)) \in \mathbb{F}[X]$ is the unique univariate polynomial of degree $\leq d(r-1)$ interpolating the points $(\tau, f(g_a(\tau)))$ where τ runs through $d(r-1)+1$ points of $\mathbb{F}_{q'}$. Hence we can derive $f(g_a(\theta)) = f(a)$ in an algebraic manner from $d(r-1)+1$ points from the datastructure.

Finally, the special case satisfies the requirements of the lemma, since

$$q' = \tilde{q}^k > \tilde{q}^{\log_{\tilde{q}}(rd-d)} = d(r-1)\,.$$

On the other hand, we find

$$q' = \tilde{q}^k \leq \tilde{q} \cdot \tilde{q}^{\log_{\tilde{q}}(rd-d)} = \tilde{q}d(r-1)\,,$$

which implies both $q' \leq \tilde{q}dr$ and, due to $\tilde{q} > d$ and $\tilde{q} \geq 2$, also $\tilde{q}^{k-2} < r-1 \leq \tilde{q}^{r-2}$, thus $k < r$. We conclude $q' \leq \min\{\tilde{q}dr, q/\tilde{q}\}$. □

Proof of Cor. 2.5. We use Lemma A.1 recursively until points are in $\mathbb{F}_{\tilde{q}}^n$. Let $q_0 = q$, let $q_1 = \tilde{q}^{k_1}$ for $k_1 = \lfloor \log_{\tilde{q}}(d \log_p q) \rfloor + 1$ and, for $i \geq 2$, let $q_i = \tilde{q}^{k_i}$ where $k_i = \lfloor \log_{\tilde{q}}(k_{i-1}d - d) \rfloor + 1$. In detail, the algorithm proceeds as follows. On input $a \in \mathbb{F}_{q_i}^n$, use Lemma A.1 (with $\tilde{q} = p$ in the first step and $\tilde{q} = \tilde{q}$ in all later steps, and the compositum $\mathbb{F}\mathbb{F}_{q_i}\mathbb{F}_{q_{i+1}}$ as underlying field) to write $f(a)$ as a polynomial expression in multiple $f(a_j)$ for $a_j \in \mathbb{F}_{q_{i+1}}^n$. Indeed, the prerequisites are satisfied: we have $q_1 > d \log_p q$ by construction, and all later steps are covered by the special case of Lemma A.1. If $q_{i+1} = \tilde{q}$ we are done. Otherwise, for each a_j, use the algorithm recursively to write $f(a_j)$ as a polynomial expression in multiple $f(a'_k)$ for $a'_k \in \mathbb{F}_{\tilde{q}}^n$. We now compose these algebraic circuits: this may require emulating arithmetic in the compositum $\mathbb{F}\mathbb{F}_{q_1}$ resp. $\mathbb{F}\mathbb{F}_{q_i}\mathbb{F}_{q_{i+1}}$, $i \geq 1$, using multiple elements in $\mathbb{F}$, but this only increases circuit size by a polynomial factor, namely $k_i^{O(1)} \leq k_1^{O(1)}$. The result of this composition is an algebraic circuit that evaluates to $f(a)$ on the chosen datastructure elements as claimed.

First, we show that $2h + 3 = O(1)$ reduction steps suffice to reach $q_i = \tilde{q}$, where h is the height of the tower of exponentials bounding q, i.e. $q \leq \exp^{\circ h}(d)$. To this end, note that $\tilde{q} > d$ and $\tilde{q} \geq 2$ imply

$$k_1 \leq 1 + \log_{\tilde{q}}(d) + \log_{\tilde{q}}(\log_p q) < 2 + \log_2 \log_p q$$

and similarly, for $i \geq 1$ with $k_i \geq 2$, that $k_{i+1} \leq 1 + \log_{\tilde{q}}(d) + \log_{\tilde{q}}(k_i - 1) < 2 + \log_2 k_i$. Now define $x_0 = \log_p q$ and $x_{j+1} = 2 + \log_2 x_j$. As $y \mapsto 2 + \log_2 y$ is monotonic, we find inductively that $k_i \leq x_i$,

unless $q_{i'} = \tilde{q}$ for some $i' < i$ already. Since $2+\log_2(2+\log_2 \exp(y)) \le y$ for $y \ge 6$, we find inductively $x_{2j} \le \exp^{\circ(h-j)}(d+6)$. Hence $k_{2h} \le x_{2h} \le d+6$, so $d(k_{2h}-1) \le d(d+5) < (d+1)^3 \le \tilde{q}^3$ and thus $k_{2h+1} \le 3$. Finally, the special case of Lemma A.1 gives $q_{i+1} \le q_i/\tilde{q}$, i.e. $k_{i+1} < k_i$, whenever $k_i \ge 2$, hence $k_{2h+3} = 1$, i.e. $q_{2h+3} = \tilde{q}$.

The total query complexity is

$$\begin{aligned}(d \log_p q) \prod_{i=1}^{2h+2} (dk_i) &\le d^{O(1)} \log_p(q) \prod_{i=1}^{2h+2} k_1 \\ &\le d^{O(1)} \log_p(q)(2+\log_2 \log_p q)^{O(1)} \\ &\le d^{O(1)} \log_p(q) \ln\ln(q)^{O(1)} \le (d \ln\ln q)^{O(1)} \log_p q\,. \end{aligned}$$ □

Disclosure of AI use

We used Claude Code (Opus 5.5) for minor editing of phrasing and to review the manuscript for errors and clarity. All content, including proofs and references, was written and verified by the authors, and all issues raised by the AI review were manually addressed.