

Polygero: Proximity Testing From Polynomial Challenges

Matteo Frigo¹ and abhi shelat^{1,2}

¹ Google, Cambridge, MA

² Northeastern University

Abstract. We present a new proximity test for Reed–Solomon codes that admits a higher-agreement regime with soundness approaching the field’s bit length.

All prior work in this area [AHIV17,BSCI⁺20,BSCH⁺25] uses the idea of *scalar* batching in which the verifier provides a scalar challenge for each codeword, and the prover must supply a codeword equivalent to the combination, and the verifier tests it at randomly opened columns.

Our main insight is to instead use a low-degree polynomial challenge for each codeword. We prove that the low-dimensional spaces generated by polynomial challenges become maximum distance separable (MDS) after deleting a few coordinates, and this phenomenon leads to a new kind of proximity gap that we can exploit.

This insight enables a simpler, in fact elementary counting proof for soundness of the proximity tests. We do not need the Polishchuk–Spielman lemma or Guruswami–Sudan decoding.

The same technique proves soundness of Polygero, a Liger-inspired IOP [AHIV17], beyond the unique-decoding regime. Through recursion, Polygero produces proofs with a number of field elements logarithmic in the witness size, independently of the number of linear constraints. It does not require roots of unity in the finite field or out-of-domain sampling.

The soundness analysis, including the extractor and the recursive composition, and every reported parameter schedule are machine-checked in Lean 4.

1 Introduction

A zero-knowledge non-interactive argument of knowledge (zk-NARK) allows a prover to prove knowledge of a witness w such that $C(x, w) = 0$, where C is a public circuit and x is a public input. The proof is often short or *succinct*, and it reveals nothing about w beyond the truth of the statement.³ These systems are

³ Such systems are commonly called SNARKs, but technically that term refers to arguments whose verifier also runs in time sublinear in the statement. Our verifier combines the linear constraints itself, so this stronger property can be achieved with our protocol for special types of statements or with preprocessing; we therefore use the term NARK.

widely used to protect the privacy of individuals in digital identity applications, such as Longfellow [Fs24], which proves statements about identity documents signed with ECDSA without revealing them; to provide transaction privacy in distributed ledgers [BSCG⁺14]; and as components of other cryptographic tools, for tasks such as multi-party computation, threshold signing, and voting.

Modern constructions of these systems rely on Reed–Solomon error-correcting codes, and on a key primitive called a *proximity test*, which allows the prover to show that a committed word is close to a Reed–Solomon codeword. The protocols are usually designed and analyzed as *interactive oracle proofs* (IOPs) [BSCS16,RRR16]. In an IOP, the prover sends *oracles*: abstractly, it commits to a long word that it cannot later change. The verifier sends random challenges, the prover answers with further messages or oracles, and the verifier finally reads a few entries of the oracles, at positions of its choice, and accepts or rejects the statement. Merkle-tree commitments realize the oracles, and the Fiat–Shamir transform removes the interaction, turning the IOP into a NARK [Kil92,Mic00,BSCS16]. In this framework, proving $C(x, w) = 0$ reduces to several tests on a committed encoding of the witness: that the encoding consists of codewords, and that the encoded values satisfy the constraints of C . The most expensive of these is usually the proximity test, whose soundness determines how many entries the verifier must read.

Scalar batching. Prior proximity tests [AHIV17,BSBHR18,BSCI⁺20,BSCH⁺25] check a batch of committed words $w_1, \dots, w_m$ at once. The verifier sends a *scalar* challenge $\beta_r \in \mathbb{F}$ for each word, the prover returns a codeword that should equal the combination $\sum_r \beta_r w_r$, and the verifier compares the two at a few random positions. Soundness rests on a *proximity gap*: if some word is far from the code, then for all but a small fraction of the challenges, the combination is far from the code as well [BSCI⁺20].

The analysis of this family of tests has been hampered by two issues. First, beyond the unique-decoding radius, the proofs rely on list decoding, through the Guruswami–Sudan decoder and the Polishchuk–Spielman lemma, and they are correspondingly intricate. Second, the best list-decoding analyses introduce large error terms. Write k for the dimension of the code, ρ for its rate, and η for the margin between the proximity parameter and the Johnson radius $1 - \sqrt{\rho}$. In this regime, the error of [BSCI⁺20, Theorem 1.2] is

$$\frac{k^2}{(2 \min\{\eta, \sqrt{\rho}/20\})^7 q},$$

so the analysis gives up $2 \log_2 k + 7 \log_2 (1/(2 \min\{\eta, \sqrt{\rho}/20\}))$ bits of the field size, which for realistic code dimensions is a large part of a 128-bit field. Later work reduces the number of exceptional challenges in this regime, but it also shows that, beyond the Johnson radius, some Reed–Solomon codes require many more [BSCH⁺25]. Deployed systems therefore may require working over large extension fields. For example, we are unaware of any previous result that is able to achieve more than 120 bits of security in a 128-bit field, whereas our approach can achieve 127 bits, see §8.

Polynomial challenges. Scalar challenges are applicable to any linear code. We introduce a new kind of challenge, which exploit the polynomial structure of Reed-Solomon codes. Instead of a scalar, the verifier sends a *polynomial challenge*: a polynomial B_r of degree at most s for each committed word w_r . The challenged word has j th coordinate $\sum_r B_r(x_j)w_{r,j}$, where x_j is the j th evaluation point of the code. If each w_r is the evaluation of a polynomial of degree less than k , the challenged word is the evaluation of a polynomial of degree less than $k + s$. The prover answers with such a polynomial p , and the verifier compares p with the challenged word at t random positions. Scalar challenges are the case $s = 0$.

Rank defect and the MDS theorem. Polynomial challenges enlarge the space of challenges in a way compatible with the code: multiplying a polynomial of degree less than k by a challenge of degree at most s gives a response of degree less than $R = k + s$. To see what the larger space buys, fix the committed words and a set T of columns on which the prover hopes to answer. Each column of T imposes one linear equation on the challenge and response coefficients. With the challenge fixed, the first R independent equations determine the response, and every additional independent equation imposes a condition on the challenge. The number δ of these additional equations is the *rank defect* of T : the dimension that the committed words and their *shifts*, obtained by multiplying coordinate j by $x_j, x_j^2, \dots, x_j^s$, add to the response code on T . A response that agrees with the challenged word on T exists for exactly a $q^{-\delta}$ fraction of the challenges (Lemma 7). A union bound over the $\binom{n}{c}$ sets of $c = k + 2s + 1$ columns shows that, except on a fraction $\mu = \binom{n}{c}/q^{s+1}$ of the challenges, every set of at least c columns on which the prover can answer has rank defect at most s (Lemma 8). This union bound takes the place of the list-size bounds of prior analyses.

On such a set, the response code and the shifted words span a space of dimension at most $k + 2s$. The structural fact behind our analysis is that these low-dimensional spaces become maximum distance separable (MDS) after deleting at most s coordinates (Theorem 1), which we prove by elementary linear algebra. The theorem separates the errors that can be localized, which occupy at most s coordinates and are discarded, from a residual error whose effect is controlled by its rank: after the discarded coordinates are deleted, all sets of coordinates of the same size behave alike. In our analysis, this theorem replaces the proximity gap.

The soundness analysis then proceeds by counting, without the Polishchuk–Spielman lemma, Guruswami–Sudan decoding, or a list-size bound. Its error combines a cut-and-choose term, which decays exponentially in t , with the rank term μ , which a moderate s makes negligible in a large field. For instance, over a field with 2^{128} elements, a code of length 25724 and rate about $1/16$, with $s = 74$ and 64 opened columns, has per-level error at most 2^{-120} (Remark 2).

Beyond proximity. For a given committed word, the usual proximity-gap analysis seeks to find an *agreement set* of coordinates such that, except for a small number of unlucky challenges, all challenges admit a response that agrees with

the code on the agreement set. Our analysis redefines what agreement with the code means. We do not insist on exact agreement, but we allow the response on the agreement set to be the sum of a codeword and a vector in some low-dimensional subspace. A verifier fails to detect a cheating prover if it unluckily opens coordinates outside the agreement set (which is rare if the agreement set is large), or if it unluckily picks a challenge that happens to annihilate the error subspace, a rare Schwartz-Zippel-type phenomenon that has a field size term at the denominator, and that we can count directly.

This treatment helps the analysis because we want the provable agreement set to be as large as possible, and we can trade the size of the agreement set for the dimension of the subspace of errors. As an extreme toy example, at $n > k$ distinct points, the word with code points x_j^k has Hamming distance $n - k$ from the code of polynomials of degree less than k . Thus the maximum size of the agreement set is k . On the other hand, the same situation can be described as an agreement set of size n with one residual error vector.

A single error vector, as in the previous example, is not too interesting. The power of this method is that polynomial challenges yield error subspaces of dimension that can be analyzed, and moreover analyzed by elementary (if lengthy) linear algebra arguments. The connection between the agreement set and the space of errors is given by Theorem 1. After discarding at most s coordinates, and outside the rank exception, the residual rank of the error space is at most s .

This perspective reorients the analysis from an attempt to implement a decoder to the implementation of an extractor. An extractor must produce a witness that satisfies the statement, not recover an intended message, and the prover's response can resolve ambiguities that the committed words alone leave open. In our case, and somewhat surprisingly, even though the verifier opens a small number of coordinates that would normally be in the list-decoding regime (or worse), the extractor has the entire large agreement set available, and it can determine this set with the help of the prover's response. Translating dimensions of error spaces into error coordinates within the agreement set Theorem 1 bounds the number of errors to at most s , which a bounded-error decoder such as Berlekamp-Welch can easily handle.

As a result, the error has no proximity parameter and no Johnson margin η (Remark 3). Nevertheless, a Johnson-like square root still appears, from a second-moment count of the overlaps between candidate response systems. This count resembles the proof of the Johnson bound, but it ranges over response systems, including systems with no codeword explanation such as the word x_j^k above, rather than over a decoding list. It keeps the counting manageable but does not fully exploit the structure of the response systems, which we regard as a limitation of the present analysis.

Polygero. We build this test into Polygero, an interactive oracle proof modeled on Liger [AHIV17]. Polygero proves statements consisting of linear constraints $Aw = b$ and multiplication constraints among witness entries, which suffice to express arithmetic circuits. The prover encodes the witness as rows of Reed–

Solomon codewords and commits to the resulting tableau column by column. The verifier sends a linear challenge, which combines the linear constraints into one equation, and a polynomial challenge, which combines the rows. The prover returns two polynomial responses, and the verifier compares their evaluations with a sample of committed columns, checks the multiplication constraints on the opened entries, and checks one global equation. With base degree bound k and polynomial-challenge degree s , the two responses contain $3k + s$ coefficients.

At a nonterminal level of recursive Polygero, the prover instead commits to a child oracle intended to encode those coefficients. This commitment is made before the parent query. Once the queried columns are opened, the parent's response-dependent tests become public linear constraints on the coefficients; these constraints form the child statement. The coefficient placement and child layout were fixed in advance, so the later statement does not change the interpretation of the committed child oracle. Recursion leaves only the terminal responses explicit. For soundness error $2^{-\lambda}$, a witness with N values and M multiplication gates then has proofs of $O(\lambda(1 + \log(N + M + 1)))$ field elements, independently of the number of linear constraints (Corollary 1). Polygero needs no roots of unity or other special structure in the field, and no out-of-domain sampling.

The key issue in the recursive analysis of Polygero is that a malicious child oracle *need not* correspond to a unique low-degree witness. The prover commits to the child oracle before the parent query Q , but the child statement depends on Q . For different queries, different polynomial tuples may satisfy the induced child statements, and each tuple encodes its own parent response. Soundness of the child for false statements is therefore not enough: a child statement can be true even when the response it encodes would fail the parent's tests on most queries. A direct composition would apply the parent's soundness bound to each tuple that the child oracle might explain, and pay a factor for their number, much as list-decoding analyses pay for the list size.

Extraction. We handle this issue by strengthening what we require of the extractor for a single invocation (Theorem 4). The extractor itself is simple. It reads the full committed oracle, the challenge, and the response pair, and finds the *good* columns, those that pass the verifier's column tests. If there are enough of them, it decodes each witness row on the good columns by unique decoding with at most s errors. It returns the decoded polynomial tuple if all errors lie in one common set of at most s columns and the induced witness satisfies the statement, and $\perp$ otherwise. The guarantee we prove for it differs from the usual one in two ways.

- The extractor is deterministic and never reads the verifier's query. Hence each fixed child tuple determines a parent response, and whether the parent extractor fails on it, independently of the parent query.
- The bound is *response-uniform*. For each challenge, a single bound on the fraction of accepted queries applies to every response on which extraction fails, and these bounds average to at most the per-level error ε over the chal-

lenges. The bound extends to a family of statements in which some outputs are declared unwanted, provided that no tuple is unwanted for more than a fraction ρ of the family; the error becomes $\varepsilon + (1 - \varepsilon)\rho$.

These two properties are what the recursion needs. Call a child tuple *unwanted* if the parent extractor fails on the response it encodes, or returns an output that is unwanted higher in the recursion. By the first property, this classification does not depend on the parent query; by the second, each unwanted tuple passes the parent’s tests on only a small fraction of parent queries. Applying the child’s guarantee with these unwanted sets controls all child tuples at once, with no factor for their number, and a backward induction over the levels composes the per-level errors. The proof that our extractor meets this guarantee (Section 7) is where the MDS theorem is used.

The result. For every root oracle, root statement, and prover strategy, the probability that the verifier accepts while the recursive extractor fails is at most $\kappa_0 = 1 - \prod_{j=0}^L (1 - \varepsilon_j) \leq \sum_{j=0}^L \varepsilon_j$, where ε_j is the error of level j (Theorem 5). Every successful output induces a witness that satisfies the root statement. For a false root statement, extraction therefore always fails, and every prover is accepted with probability at most κ_0 . These guarantees are for the interactive oracle proof: oracles are ideal, and the binding of the commitment and the Fiat–Shamir transform are outside the analysis.

Machine-checked proofs. Our knowledge-soundness analysis is formalized in Lean 4. Every parameter and proof size reported in Section 8 has an exact rational certificate, checked in Lean, that connects its parameters to this verifier, its error to the target $2^{-\lambda}$, and its byte count to the communication model. For zero knowledge, Lean proves that the distributions of the root view coincide for a bounded batch of root columns chosen after both responses.

Organization. Section 3 proves the MDS theorem for polynomial challenges. Section 4 specifies the ordinary and recursive protocols and proves completeness. Section 5 proves zero knowledge for the oracle view. Section 6 defines the extractor and derives recursive extraction and soundness from the per-level extraction bound. Section 7 proves the per-level extraction bound using a fixed rank exception, reduced affine systems, and marginal system-exception bounds. Section 8 presents the numerical instantiations and their communication model. Appendix A proves the binomial counting inequalities used in the per-level analysis.

AI Statement: Our work on this result has been ongoing for several months, well before LLMs were capable. Recently, we have extensively used Gemini, Claude, and OpenAI to verify our result by producing Lean proofs of main theorems and to verify and simplify our written proofs. By our measure, the tools have improved the paper quality. The authors take full responsibility for the results in the paper.

2 Notation

A *setup* is a tuple $(\mathbb{F}, n, \mathbf{x}, m, k, s)$ consisting of a finite field $\mathbb{F}$ with $q = |\mathbb{F}| > n$ elements, integers $n \geq 1$, $m \geq 0$, $k \geq 1$, and $s \geq 0$, and distinct evaluation points $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{F}^n$. All vector spaces, spans, and dimensions are finite and over $\mathbb{F}$. We write $[n] = \{1, \dots, n\}$.

Words. A *word* is a vector $u \in \mathbb{F}^n$, with j th coordinate u_j . Its *support* is $\text{supp}(u) = \{j \in [n] : u_j \neq 0\}$. Let e_j be the word that is one at coordinate j and zero elsewhere, let $\mathbf{1} = (1, \dots, 1)$ be the constant word, and write $u \odot v$ for the coordinatewise product of two vectors of the same length, so that $u \odot v = (u_1 v_1, \dots, u_n v_n)$ for words. Honest parties in our protocols commit to codewords, but we analyze malicious parties, who may commit to arbitrary words $\mathbf{w} = (w_1, \dots, w_m) \in \mathbb{F}^{n \times m}$.

Reed–Solomon codes. For a nonnegative integer r , let $\mathcal{P}_{<r} = \{p \in \mathbb{F}[X] : \deg p < r\}$, so that $\mathcal{P}_{<0} = \{0\}$, and let $\mathcal{P}_{\leq r} = \mathcal{P}_{<r+1}$. Define

$$\text{ev}(p) = (p(x_1), \dots, p(x_n)), \quad \mathcal{C}_r = \text{ev}(\mathcal{P}_{<r}),$$

so that $\text{ev}(\mathbf{1}) = \mathbf{1}$. For a set $S \subseteq [n]$, evaluation at the points x_j with $j \in S$ has rank $\min(r, |S|)$ on $\mathcal{P}_{<r}$: it is injective when $r \leq |S|$ by the root bound, and surjective when $r \geq |S|$ by interpolation. Hence, for $0 \leq r \leq n$, $\mathcal{C}_r$ is the Reed–Solomon code $\text{RS}[n, r]$ of length n and dimension r , with *rate* $\rho = r/n$ and, for $r \geq 1$, minimum distance $n - r + 1$.

Polynomial challenges and the shift. A polynomial challenge multiplies a word coordinatewise by the evaluations of a polynomial. Multiplication by X is the linear *shift* operator $\text{shift } v = \text{ev}(X) \odot v$, that is, $(\text{shift } v)_j = x_j v_j$; in particular, $\text{shift } \text{ev}(p) = \text{ev}(Xp)$. Since $\text{shift}^a v = \text{ev}(X^a) \odot v$, linearity gives, for every $B = \sum_a b_a X^a \in \mathbb{F}[X]$,

$$\text{ev}(B) \odot v = \sum_a b_a \text{shift}^a v, \tag{2.1}$$

a linear combination of $v, \text{shift } v, \dots, \text{shift}^{\deg B} v$.

3 Low-dimensional spaces become MDS after puncturing

We prove that low-dimensional spaces generated by polynomial challenges become maximum distance separable (MDS) after deleting a few coordinates: restriction to any set of remaining coordinates whose size equals the remaining dimension has full rank. We will use this property once in the soundness analysis, to establish the MDS structure of a union of overlapping canonical cores (Lemma 12).

3.1 The explosion lemma

For a linear operator L on a space V , a subspace J is *invariant* if $LJ \subseteq J$. A vector c is *cyclic* if $V = \text{span}\{L^a c : a \geq 0\}$.

Lemma 1. *Let $L : V \rightarrow V$ be linear, and let $U \subseteq V$ be a subspace containing a cyclic vector for L . Assume*

$$\dim(U + LU) \leq \dim U + 1.$$

Let $J \subseteq V$ be an invariant subspace. If $U \cap J$ is not invariant, then

$$U + J = V.$$

Proof. Pick $v \in U \cap J$ with $Lv \notin U \cap J$. Since J is invariant, $Lv \in J$, so $Lv \notin U$. Thus Lv supplies a new direction modulo U . The dimension bound says there is room for only this one direction:

$$U + LU = U + \text{span}\{Lv\} \subseteq U + J.$$

It follows that

$$L(U + J) = LU + LJ \subseteq U + J.$$

The enlarged space is invariant and contains a cyclic vector. It therefore contains all its iterates and equals V . $\square$

3.2 Low dimension forces slow growth

Put $R = k + s$, and recall that $\mathcal{C}_R$ consists of evaluations of polynomials of degree less than R . For arbitrary words $w_1, \dots, w_m \in \mathbb{F}^n$, put

$$U = \mathcal{C}_R + \text{span}\{\text{shift}^a w_i : 1 \leq i \leq m, 0 \leq a \leq s\}. \quad (3.1)$$

The next lemma shows that, when U has low dimension, shifting adds at most one new dimension. The key fact is that successive dimension increments decrease.

Lemma 2. *Let U be as in (3.1). If $\dim U \leq k + 2s$ and $\dim U < n$, then $\dim(U + \text{shift } U) \leq \dim U + 1$.*

Proof. Put $U_{-1} = \mathcal{C}_{k-1}$ and, for $0 \leq a \leq s + 1$, put

$$U_a = \mathcal{C}_{k+a} + \text{span}\{\text{shift}^b w_i : 1 \leq i \leq m, 0 \leq b \leq a\}.$$

Then $U_s = U$, $\text{shift } U_{-1} \subseteq U_0$, and $U_{a+1} = U_a + \text{shift } U_a$ for $0 \leq a \leq s$. For these indices, the maps

$$U_a/U_{a-1} \longrightarrow U_{a+1}/U_a, \quad [u] \longmapsto [\text{shift } u],$$

are well-defined: changing u by an element of U_{a-1} changes its shift by an element of U_a , since $\text{shift } U_{a-1} \subseteq U_a$. They are surjective because $U_{a+1} = U_a + \text{shift } U_a$:

every class modulo U_a is represented by the shift of some $u \in U_a$. Thus the integer increments $\Delta_a = \dim U_a - \dim U_{a-1}$ are nonincreasing.

By definition, $\mathcal{C}_R \subseteq U$ and $R = k + s$. By hypothesis, $\dim U < n$, so $\mathcal{C}_R$ is proper and therefore $R < n$. Hence $k - 1 < k + s = R < n$, and $\dim U_{-1} = \dim \mathcal{C}_{k-1} = \min\{k - 1, n\} = k - 1$. There are $s + 1$ increments $\Delta_0, \dots, \Delta_s$ from U_{-1} to $U_s = U$. Telescoping gives $\sum_{a=0}^s \Delta_a = \dim U - (k - 1)$. Since the increments are nonincreasing,

$$(s + 1)\Delta_{s+1} \leq \sum_{a=0}^s \Delta_a = \dim U - (k - 1) \leq 2s + 1 < 2(s + 1).$$

The bound $\dim U - (k - 1) \leq 2s + 1$ uses the hypothesis $\dim U \leq k + 2s$. So Δ_{s+1} is an integer less than 2, and thus $\Delta_{s+1} \leq 1$. Since $U_{s+1} = U + \text{shift } U$ and $U_s = U$, this proves the claim. $\square$

3.3 Puncturing yields an MDS code

A linear code of dimension r is MDS exactly when restriction to any r coordinates has full rank. We prove that removing at most s coordinates from the space in (3.1) yields an MDS code.

Recall that e_j is the word that is 1 at coordinate j and 0 elsewhere.

Theorem 1. *Let $w_1, \dots, w_m \in \mathbb{F}^n$ be arbitrary words, and put*

$$U = \mathcal{C}_R + \text{span}\{\text{shift}^a w_i : 1 \leq i \leq m, 0 \leq a \leq s\},$$

as in (3.1). Assume $\dim U \leq k + 2s$ and $\dim U < n$. Set $E = \{j : e_j \in U\}$ and $S = [n] \setminus E$. Then

$$|E| \leq \dim U - R, \tag{3.2}$$

and, for all $H \subseteq S$,

$$\dim(U|_H) = \min\{|H|, \dim U - |E|\}. \tag{3.3}$$

Thus the punctured code $U|_S$ has dimension $\dim U - |E|$ and is MDS.

Proof. For $A \subseteq [n]$, let P_A be the subspace of $\mathbb{F}^n$ supported on A . By definition of E , we have $P_E \subseteq U$. By definition, $S = [n] \setminus E$, so $|S| = n - |E|$ and the restriction map $U \rightarrow \mathbb{F}^S$, $u \mapsto u|_S$, has kernel P_E . Rank-nullity gives $\dim(U|_S) = \dim U - |E|$. By (3.1), $\mathcal{C}_R|_S \subseteq U|_S$. The evaluation rank formula from Section 2 therefore gives

$$\min\{R, |S|\} = \dim(\mathcal{C}_R|_S) \leq \dim(U|_S) = \dim U - |E| < n - |E| = |S|,$$

where the strict inequality uses the hypothesis $\dim U < n$. The minimum must therefore be R , giving $R \leq \dim U - |E|$ and proving (3.2).

Take any $H \subseteq S$, and let $J = P_{[n] \setminus H}$ be the kernel of restriction to H . The space J is invariant under shift because shifting preserves support. We consider two cases.

If $U \cap J$ is invariant under **shift**, it is spanned by the vectors e_j it contains. Indeed, for any word u in an invariant space and any j with $u_j \neq 0$, interpolation and (2.1) express e_j as a linear combination of shifts of u . For each $j \in E$, we have $e_j \in U$ by definition of E , and $e_j \in J$ because $H \subseteq S = [n] \setminus E$, so e_j vanishes on H . Conversely, $e_j \in U \cap J$ implies $j \in E$ by definition of E . Thus $U \cap J = P_E$, and rank-nullity gives

$$\dim(U|_H) = \dim U - \dim(U \cap J) = \dim U - |E|.$$

If $U \cap J$ is not invariant under **shift**, apply the explosion lemma. The vector $\mathbf{1} \in \mathcal{C}_R \subseteq U$ is cyclic because its shifts are the monomial evaluations, which span $\mathbb{F}^n$ by interpolation. The dimension hypotheses and Lemma 2 give $\dim(U + \text{shift } U) \leq \dim U + 1$, so Lemma 1 gives $U + J = \mathbb{F}^n$. Since every vector in J vanishes on H by definition of J , restriction to H maps U onto $\mathbb{F}^H$, giving $\dim(U|_H) = |H|$.

In either case,

$$\dim(U|_H) \geq \min\{|H|, \dim U - |E|\}.$$

The reverse inequality follows because restriction from S to H cannot increase dimension, and $\dim(U|_H) \leq |H|$. This proves (3.3).

Finally, $U|_S$ has dimension $\dim U - |E|$. For every set $H \subseteq S$ of that size, (3.3) says that restriction from $U|_S$ to H has full rank. This is precisely the MDS property. $\square$

Bibliographical notes The explosion lemma appears in several other works where it is explained as an elementary consequence of submodularity of dimension growth; see Bachoc–Serra–Zémor [BSZ17] and [BSZ18]. Brunovský states the decreasing-increment principle used in Lemma 2 in his treatment of controllability indices [Bru70, Lemma 1(i), p. 175].

4 The recursive Polygero protocol

We are now ready to present a practical IOP protocol that ultimately relies on the polynomial challenge idea and the structural MDS lemma from the previous section. Although we believe the MDS theorem can be used to improve many IOPs, we here choose to introduce Polygero, a natural extension of Ligerio [AHIV17] that uses our polynomial challenges, applies a recursive construction that leads to a poly-logarithmic sized proof, and exploits an analysis that extends close to the Johnson bound. We choose this direction because it is the conceptually simplest way to enforce linear and quadratic constraints on the inputs and to achieve the zero-knowledge property. Section 8 reports certified proof sizes for this construction.

We remark that the original Ligerio paper’s analysis (specifically the analysis of the linear constraints) does not extend past the unique decoding regime; Flock [BRW26] is analyzed only for binary fields under a specific construction, and WHIR [ACFY24] requires a finite field with specific root structures. In contrast, our protocol has none of these restrictions.

4.1 Overview

Polygero, like Ligerito [AHIV17], proves that a public system of linear and quadratic constraints is satisfiable. A statement asserts that there exist witness values $w = (w_1, w_2, \dots)$ such that

- the linear constraints $Aw = b$ hold, for a public matrix A and vector b ; and
- for each multiplication gate, given by a triple of indices (i_1, i_2, i_3) , the quadratic constraint $w_{i_1}w_{i_2} = w_{i_3}$ holds.

These constraints suffice to express the satisfiability of any arithmetic circuit over $\mathbb{F}$: every wire value becomes a witness value, addition gates and wiring become linear constraints, and multiplication gates become quadratic constraints. The statements are therefore universal.

The protocol works roughly as follows. The prover arranges the witness values into a matrix with d rows and $|\mathcal{W}|$ columns; these dimensions, like the code parameters below, are optimized in Section 8. The prover encodes each row r with a Reed–Solomon code in two steps.

1. It interpolates a *row polynomial* f_r of degree less than k that takes the row’s witness values at fixed public points ξ_a , one for each column $a \in \mathcal{W}$. In the zero-knowledge variant, the remaining degrees of freedom are chosen at random.
2. It evaluates f_r at the n code points $x_1, \dots, x_n$, which gives the codeword $\text{ev}(f_r) \in \mathcal{C}_k = \text{RS}[n, k]$ of the Reed–Solomon code with length n , dimension k , and rate $\rho = k/n$.

For multiplication, the prover places the two inputs and the output of each gate at the same position of three rows, which form a *row triple* g . The two input rows are interpolated with degree less than $k/2$, and the output row is their polynomial product, so the identity $f_{i_1(g)}f_{i_2(g)} = f_{i_3(g)}$ holds at every code point. This choice is different from Ligerito and Ligerito [NA25]; as shown by the Longfellow [Fs24] scheme, the most efficient use of an IOP requires only a small number of quadratic constraints, and so we optimize for that case.

In the case of zero-knowledge, the prover appends three random mask rows: two mask the linear response and one masks the proximity response. We include these rows so that the soundness theorem applies in both cases.

Altogether, the encoded rows form a *tableau* with n columns, and the prover commits to it column by column, in practice with a Merkle tree whose leaves are the columns. Our analysis treats this commitment as an oracle (Section 1).

The verifier then sends a challenge $\gamma = (\alpha, B)$. The linear challenge α combines the linear constraints into a single equation, and the polynomial challenge $B = (B_1, \dots, B_m)$ combines the rows with polynomial coefficients of degree at most s , as in Section 3. The prover answers with two polynomials. Although our analysis treats them together, we follow the historical presentation of Ligerito and call them the *proximity response* p , of degree less than $R = k + s$, which should equal the challenged combination of the rows, and the *linear response* Z , of degree less than $2k$, which certifies the combined linear constraint. Finally,

the verifier opens t random columns of the tableau and checks the responses against them, together with the multiplication identities and one global sum; Section 4.4 states these tests precisely. In the recursive protocol, the prover does not send the coefficients of p and Z . It instead commits to them as the witness of a smaller child instance (Sections 4.5 and 4.6).

4.2 Statements, witnesses, and public encoding data

An invocation of Polygero is parameterized by a setup $(\mathbb{F}, n, \mathbf{x}, m, k, s)$ as in Section 2, whose witness rows are codewords of $\mathcal{C}_k = \text{RS}[n, k]$ of rate $\rho = k/n$, and by the number t of columns that the verifier opens, where $0 \leq t \leq n$. The number m of challenged rows is fixed in Section 4.3. Let $R = k + s \leq n$ be the degree bound for the proximity response.

Witness layout. Fix a k -element interpolation set $\Omega \subseteq \mathbb{F}$ disjoint from the code points, which requires $n + k \leq q$, and a finite set $\mathcal{W}$ of *logical positions* with distinct points $\xi_a \in \Omega$ for $a \in \mathcal{W}$. A witness has d_w linear rows and M row triples for multiplication, giving $d = d_w + 3M$ rows. Row triple $g \in \mathcal{M} = [M]$ consists of the rows

$$i_1(g) = d_w + g, \quad i_2(g) = d_w + M + g, \quad \text{and} \quad i_3(g) = d_w + 2M + g,$$

which hold the first inputs, the second inputs, and the outputs of $|\mathcal{W}|$ multiplication gates, one at each logical position. A statement therefore has $M|\mathcal{W}|$ scalar multiplication constraints.

Definition 1 (Statement and witness). *Fix a finite index set I for the linear constraints. A witness is a matrix $w \in \mathbb{F}^{[d] \times \mathcal{W}}$ with rows $w_1, \dots, w_d \in \mathbb{F}^{\mathcal{W}}$; for linear algebra, we also regard w as a vector indexed by $[d] \times \mathcal{W}$. A statement is a pair $\sigma = (A, b)$ with $A \in \mathbb{F}^{I \times ([d] \times \mathcal{W})}$ and $b \in \mathbb{F}^I$. A witness w satisfies σ , written $\text{Sat}(\sigma, w)$, if*

$$Aw = b, \tag{4.1}$$

$$w_{i_1(g)} \odot w_{i_2(g)} = w_{i_3(g)} \quad \text{for all } g \in \mathcal{M}. \tag{4.2}$$

A statement is true if some witness satisfies it, and false otherwise.

Since $\odot$ is the coordinatewise product of rows, (4.2) imposes the gate of row triple g at every logical position. Logical positions that a statement leaves unconstrained, such as the unused slots of a child witness (Section 4.5), correspond to zero columns of A . The row indices, logical positions, multiplication layout, and index set I are fixed before execution; only A and b vary with the statement. Only the top level of the recursion may have row triples; at every positive recursive level, $M = 0$.

Constraint polynomials. Our protocol requires mapping the linear constraints $Aw = b$ into polynomials. For a linear challenge $\alpha \in \mathbb{F}^I$, the row vector $\alpha^\top A$ combines the constraints into the single constraint $\alpha^\top Aw = \alpha^\top b$; like a witness, it is indexed by $[d] \times \mathcal{W}$. For $r \in [d]$, the *constraint polynomial* $C_{\sigma, \alpha, r} \in \mathcal{P}_{<k}$ interpolates row r of $\alpha^\top A$ at the logical points and vanishes at the other points of Ω :

$$C_{\sigma, \alpha, r}(\xi_a) = (\alpha^\top A)_{r, a} \quad (a \in \mathcal{W}), \quad C_{\sigma, \alpha, r}(y) = 0 \quad (y \in \Omega \setminus \{\xi_a : a \in \mathcal{W}\}).$$

Since $|\Omega| = k$, this determines $C_{\sigma, \alpha, r}$ uniquely. The map is designed for the following identity. If row polynomials f_r take the witness values, $f_r(\xi_a) = w_{r, a}$, then

$$\sum_{a \in \mathcal{W}} \sum_{r=1}^d C_{\sigma, \alpha, r}(\xi_a) f_r(\xi_a) = \sum_{r, a} (\alpha^\top A)_{r, a} w_{r, a} = \alpha^\top Aw.$$

Thus the combined constraint holds exactly when the polynomial $\sum_r C_{\sigma, \alpha, r} f_r$ sums to $\alpha^\top b$ over the logical points, which is what the linear response certifies (Section 4.4).

4.3 Honest row polynomials and masks

Here we describe how the honest prover encodes a witness as row polynomials, and how it adds mask rows, which are random rows needed for zero knowledge and may be zero when only soundness is needed.

Row polynomials. An honest encoding realizes the witness by row polynomials $f_1, \dots, f_d \in \mathcal{P}_{<k}$ with $f_r(\xi_a) = w_{r, a}$ for all $r \in [d]$ and $a \in \mathcal{W}$, and forms the output row of each row triple as the polynomial product of its two input rows. Section 4.4 gives concrete encoders.

Masks. Besides the d witness rows, the prover commits to two linear-mask rows f_{d+1}, f_{d+2} , which receive polynomial challenges like the witness rows, and to a proximity mask $f_\star$ with default challenge 1. An opened column therefore contains $d+3$ field elements. The honest mask conditions are

$$\begin{aligned} f_{d+1} \in \mathcal{P}_{<k} \quad \wedge \quad f_{d+2} \in \left\{ f \in \mathcal{P}_{<k} : \sum_{a \in \mathcal{W}} f(\xi_a) = 0 \right\} \\ \wedge \quad f_\star \in \mathcal{P}_{<R}. \end{aligned} \tag{4.3}$$

The linear masks are added to the linear response through two further public polynomials, which hide them from the sum over the logical points. Fix a set $E \subseteq \Omega \setminus \{\xi_a : a \in \mathcal{W}\}$, which may be empty for a non-hiding encoder; let $v(X) = \prod_{y \in \Omega} (X - y)$, and let $z_R \in \mathcal{P}_{<k}$ be zero on E and one on the rest of Ω . Together with the constraint polynomials, the public polynomials satisfy

$$\deg v = k \quad \wedge \quad z_R \in \mathcal{P}_{<k} \quad \wedge \quad C_{\sigma, \alpha, r} \in \mathcal{P}_{<k}, \tag{4.4}$$

$$v(\xi_a) = 0 \quad \wedge \quad z_R(\xi_a) = 1 \quad \wedge \quad C_{\sigma, \alpha, r}(\xi_a) = (\alpha^\top A)_{r, a}. \tag{4.5}$$

By (4.5), at the logical points the term vf_{d+1} vanishes and $z_R f_{d+2}$ agrees with f_{d+2} , whose values there sum to zero by (4.3); so neither mask changes the sum of the linear response over the logical points. The conditions on the row polynomials and masks in this subsection are the *honest-encoding conditions* used for completeness.

The committed oracle. An honest prover commits to the column oracle

$$\forall j \in [n] : \quad W(j) = (f_1(x_j), \dots, f_m(x_j), f_\star(x_j)) \in \mathbb{F}^{m+1}. \quad (4.6)$$

A malicious oracle is simply $W = (W_1, \dots, W_m, W_\star) \in (\mathbb{F}^n)^{m+1}$. We view an oracle as the function $j \mapsto W(j)$ that returns column j , so $W(j) = (W_{1,j}, \dots, W_{m,j}, W_{\star,j})$, where $W_{r,j}$ is the j th entry of the word W_r . The challenged rows $W_1, \dots, W_m$ are the words of the setup, and the proximity mask $W_\star$ supplies the offset in the proximity equation (7.4). The names “witness row” and “mask row” impose no degree or zero-sum condition. The proximity and linear tests both use the two linear-mask rows, with the same opened entries.

4.4 Base Polygero

In this section, we describe a base-case invocation of Polygero.

Challenge space. The verifier samples independent uniform challenges

$$\begin{aligned} \alpha \in \mathcal{A} &:= \mathbb{F}^I, & B = (B_1, \dots, B_m) &\in \mathcal{B} := (\mathcal{P}_{\leq s})^m, \\ \gamma &= (\alpha, B) \in \Gamma := \mathcal{A} \times \mathcal{B}. \end{aligned}$$

The linear challenge α combines the linear constraints into one global equation. The polynomial challenge B defines the challenged word against which the verifier checks the proximity response. The complete challenge space consists of the pairs $\gamma = (\alpha, B)$ and has cardinality

$$C = |\Gamma| = q^{|I|+m(s+1)}. \quad (4.7)$$

Prover’s response. The Prover honest responses are

$$p = f_\star + \sum_{r=1}^m B_r f_r \in \mathcal{P}_{<R}, \quad (4.8)$$

$$Z = \sum_{r=1}^d C_{\sigma, \alpha, r} f_r + v f_{d+1} + z_R f_{d+2} \in \mathcal{P}_{<2k}. \quad (4.9)$$

i.e., the Prover sends one polynomial of degree less than R and another of degree less than $2k$.

Verifier check. The proximity response p is compared with the challenged word. The linear response Z is compared with a linear functional of each opened column and tested by a global witness-sum equation. Their degree bounds account for $R + 2k = 3k + s$ coefficients.

For arbitrary committed words, define

$$(Y_B)_j = W_{*,j} + \sum_{r=1}^m B_r(x_j)W_{r,j},$$

$$L_{\sigma,\alpha,j}(y) = \sum_{r=1}^d C_{\sigma,\alpha,r}(x_j)y_r + v(x_j)y_{d+1} + z_R(x_j)y_{d+2}.$$

After the responses are fixed, the verifier samples uniformly from the query set

$$\mathcal{Q} = \{Q \subseteq [n] : |Q| = t\}, \quad |\mathcal{Q}| = \binom{n}{t}. \quad (4.10)$$

Thus $|\mathcal{Q}|$ counts all possible verifier queries. At every $j \in Q$ it checks

$$p(x_j) = (Y_B)_j, \quad (4.11)$$

$$Z(x_j) = L_{\sigma,\alpha,j}(W_{1,j}, \dots, W_{m,j}), \quad (4.12)$$

$$\forall g \in \mathcal{M} : \quad W_{i_1(g),j}W_{i_2(g),j} = W_{i_3(g),j}, \quad (4.13)$$

and it also checks

$$\sum_{a \in \mathcal{W}} Z(\xi_a) = \alpha^\top b. \quad (4.14)$$

The proximity equation compares p with Y_B . The linear and witness-sum equations test the linear constraints. Multiplication is checked directly on opened witness-row entries.

Protocol 1: One-shot Polygero

Public input: a statement σ and the encoding data of Section 4.2.

Honest prover input: a satisfying witness.

1. *The committed oracle* (Section 4.3). The prover commits to an oracle W . An honest prover encodes its witness by the row polynomials and masks of Section 4.3 and commits to (4.6).
2. *Challenge space.* The verifier samples and sends a uniform challenge $\gamma = (\alpha, B) \in \Gamma$.
3. *Prover's response.* The prover sends the coefficient vectors of $p \in \mathcal{P}_{<R}$ and $Z \in \mathcal{P}_{<2k}$; responses outside these spaces are rejected. An honest prover uses (4.8)–(4.9).
4. *Verifier check.* The verifier samples $Q \in \mathcal{Q}$ uniformly, opens the columns in Q , and accepts exactly when (4.11)–(4.14) all hold.

For bounded responses, define the *good-column set*

$$G_W(\sigma, \gamma, p, Z) = \{j \in [n] : \text{all three column tests (4.11)–(4.13) hold at } j\}. \quad (4.15)$$

Write $\text{Verify}(W, \sigma, \gamma, p, Z, Q)$ for the predicate requiring (4.14) and $Q \subseteq G_W(\sigma, \gamma, p, Z)$. Once the separate witness-sum equation holds, acceptance is exactly containment in the good-column set. This turns the verifier equations into the subset counts used in soundness.

Concrete honest encoders. For a logical row $y \in \mathbb{F}^{\mathcal{W}}$ and $h \geq |\mathcal{W}|$, the affine space

$$\mathcal{F}_h(y) = \{f \in \mathcal{P}_{<h} : \forall a \in \mathcal{W} : f(\xi_a) = y_a\}$$

is nonempty by interpolation. It describes the polynomial extensions of the prescribed logical values. For a purely linear layout, assign zero at every nonlogical point of Ω and interpolate. This gives a fixed linear map from logical rows to row polynomials. Together with zero masks, it satisfies the honest-encoding conditions at all positive recursive levels. These child encoders require no independent hiding property: their inputs encode responses already available to the zero-knowledge simulator of Section 5.

An ordinary root encoder. Independent interpolation of a row triple need not preserve the multiplication tests at the code points. Instead, fix $r_{\text{enc}} \geq 0$, assume $|E| = r_{\text{enc}}$, and use the sufficient degree budget

$$k = 2(r_{\text{enc}} + |\mathcal{W}|).$$

Given a satisfying witness, independently sample

$$\begin{aligned} \forall r \in [d_w] : & \quad f_r \text{ uniformly in } \mathcal{F}_k(w_r), \\ \forall g \in \mathcal{M} : & \quad f_{i_1(g)} \text{ uniformly in } \mathcal{F}_{r_{\text{enc}} + |\mathcal{W}|}(w_{i_1(g)}), \\ \forall g \in \mathcal{M} : & \quad f_{i_2(g)} \text{ uniformly in } \mathcal{F}_{r_{\text{enc}} + |\mathcal{W}|}(w_{i_2(g)}), \end{aligned}$$

and set $f_{i_3(g)} = f_{i_1(g)}f_{i_2(g)} \in \mathcal{P}_{<k}$. The logical multiplication equations give the prescribed output values at every ξ_a , while the polynomial identity holds at every code point. Independently sample the three masks uniformly from the spaces in (4.3). Section 8 also considers unmasked encodings for the numerical comparison.

Lemma 3 (Base Completeness). *Let the witness satisfy the statement and let the encoding satisfy the honest-encoding conditions. Then the honest responses pass the ordinary verifier for all challenges and queries.*

Proof. The public and row degrees give the response degree bounds. Their definitions give both response equations at the code points, and the product rows give the multiplication tests. At the logical points, v vanishes, z_R equals one, and the zero-sum linear mask contributes zero to the witness sum. Equations (4.5) and the linear constraints then give (4.14). $\square$

Soundness. We defer proving soundness until Section 6 where we exhibit an extractor that reads the full committed oracle and the transcript, decodes the witness rows on the columns that pass the verifier’s tests, and outputs a witness after checking that it satisfies the statement.

4.5 Recursive Polygero

The one-shot protocol can be optimized to produce proofs that grow roughly as the *square-root* of the size of the witness $|w|$. We now explain how to apply the construction recursively to attain a proof that only requires $O(\lambda \log(|w|))$ field elements in the IOP model where λ is a security parameter.

The verifier’s proximity, linear, and witness-sum checks (4.11), (4.12), and (4.14) are linear constraints on the coefficients of the response (p, Z) ; only the multiplication checks (4.13) do not involve the response. A natural way to shrink the proof is therefore to apply the same mechanism recursively, without zero knowledge or quadratic constraints, to prove that “there exists a response that satisfies these constraints.” This idea leads to a *recursive* instantiation of Polygero. The intuitive argument for its soundness faces obstacles: the child oracle is committed before its statement is known, and a malicious child oracle need not encode a unique response. Section 6 handles these obstacles; here we describe the protocol.

Packing the response. Write $p(X) = \sum_{i < R} p_i X^i$ and $Z(X) = \sum_{i < 2k} z_i X^i$. Before execution, fix a map from these $3k + s$ coefficient positions into the logical slots of the child witness layout. Let $\text{pack}(p, Z)$ place the coefficients in their slots and fill the remaining slots with zero, and let $\text{unpack}(w)$ read the designated slots of a child witness w . Then

$$\text{unpack}(w) \in \mathcal{P}_{<R} \times \mathcal{P}_{<2k} \quad \wedge \quad \text{unpack}(\text{pack}(p, Z)) = (p, Z), \quad (4.16)$$

where the degree bounds hold for every child witness, satisfying or not, because they come from the fixed placement.

The child statement. After the parent opens the columns in Q , the child statement σ_Q asks whether the packed coefficients pass the parent’s response-dependent tests. Its constraints are, for each $j \in Q$,

$$\sum_{i=0}^{R-1} p_i x_j^i = W_{\star,j} + \sum_{r=1}^m B_r(x_j) W_{r,j}, \quad (4.17)$$

$$\sum_{i=0}^{2k-1} z_i x_j^i = L_{\sigma,\alpha,j}(W_{1,j}, \dots, W_{m,j}), \quad (4.18)$$

together with the global constraint

$$\sum_{i=0}^{2k-1} z_i \left(\sum_{a \in \mathcal{W}} \xi_a^i \right) = \alpha^\top b. \quad (4.19)$$

Everything except the packed coefficients is public after the openings, so these are $2t + 1$ linear constraints on the child witness, and they leave the unused slots unconstrained. The parent performs the multiplication checks itself, so the child statement is purely linear. A fixed ordering of the code columns enumerates Q , so the child's layout and constraint index set are fixed before execution; only its coefficients and targets depend on the parent transcript. The statement σ_Q is defined for every query Q , but the verifier continues only when the parent multiplication checks pass.

4.6 The recursive protocol

A recursive execution has depth $L \geq 0$, with levels $\ell = 0, \dots, L$. Each level has its own setup and layout, indicated by a subscript (for instance, $R_\ell = k_\ell + s_\ell$), and the coefficient placements between successive levels are fixed in advance. Depth zero is the ordinary protocol.

The child oracle is committed before the parent query, while its statement is determined only by the openings. This is possible because the child's witness layout does not depend on its statement, and our soundness analysis is uniform over all coefficients and targets in that fixed layout.

Protocol 2: recursive Polygero, starting with a committed root oracle

Initial state: root statement σ_0 and committed root oracle W_0 .

At level ℓ :

1. The verifier samples and sends fresh independent uniform $\gamma_\ell \in \Gamma_\ell$.
2. If $\ell < L$, the prover commits to a child oracle $W_{\ell+1}$ intended to encode the parent response coefficients. Those coefficients are not sent explicitly.
3. At a nonterminal level, the verifier samples Q_ℓ uniformly and opens W_ℓ there. It performs the parent multiplication checks. If they pass, the openings and γ_ℓ determine $\sigma_{\ell+1}$ by (4.17)–(4.19), and execution continues at the child.
4. At level L , replace steps 2–3 by the ordinary response message and tests: the prover sends p_L, Z_L before Q_L is sampled, and the verifier performs the ordinary invocation.

Accept exactly when all parent multiplication checks pass and the terminal invocation accepts. Earlier response-dependent equations are imposed through child statements.

The essential nonterminal order is

$$\gamma_\ell \longrightarrow W_{\ell+1} \longrightarrow Q_\ell \text{ and parent openings } \longrightarrow \sigma_{\ell+1}. \quad (4.20)$$

Well-formed protocol data. Each level is an invocation of Section 4.4 with its own setup $(\mathbb{F}, n_\ell, \mathbf{x}^{(\ell)}, m_\ell, k_\ell, s_\ell)$, its own layout, and its own committed oracle

W_ℓ . Within a level, the code points are distinct and disjoint from that level's interpolation set, and the public polynomials satisfy (4.4)–(4.5), by the construction of Sections 4.2 and 4.3. Levels may use different code lengths, so their evaluation points need not coincide. Consecutive levels are linked in one way only: the coefficients of the level- ℓ response are placed into logical positions of the level- $(\ell + 1)$ witness (Section 4.5). The protocol data are *well formed* when every layout and this placement are fixed before execution, each child layout has at least $3k_\ell + s_\ell$ logical slots for its parent's response, and messages follow the order (4.20). These conditions constrain only public data, not committed words; completeness also uses the honest-encoding conditions of Section 4.3.

Prover strategies. Our bounds hold against every prover. A *prover strategy* computes each prover message as a deterministic function of the transcript so far; it does not see later verifier choices. A fixed root oracle, statement, strategy, and sequence of verifier choices therefore determine one transcript. A randomized prover is covered by fixing its coins and averaging the bounds. For counting, an execution that rejects early is padded with dummy messages and fresh verifier choices, so that every complete sequence of verifier choices has the same form.

4.7 Completeness and proof size

The next lemma is the bridge between consecutive levels: a child witness satisfies the induced statement exactly when the response it encodes passes the parent's tests. Completeness uses it below, and the soundness analysis uses it to compose extraction across levels (Section 6).

Lemma 4 (Exact substitution). *Fix a parent transcript through its openings and a child witness w . Put $(p, Z) = \text{unpack}(w)$. Then*

$$\begin{aligned} & [\text{parent multiplication checks pass} \wedge \text{Sat}(\sigma_{\ell+1}, w)] \\ & \iff \text{Verify}(W_\ell, \sigma_\ell, \gamma_\ell, p, Z, Q_\ell). \end{aligned}$$

Proof. Evaluation is linear in the coefficient vector. The child equations repeat precisely the proximity, linear, and witness-sum tests, without constraining unused slots. The other parent tests are exactly the multiplication checks. This proves the equivalence for all child witnesses. $\square$

What we count. We measure *proof size* in field elements, in the oracle model. We count every field element that the prover reveals to the verifier: the entries of all opened columns, summed over the oracles of all levels, and the coefficients that the prover sends explicitly. An opened column of the level- ℓ oracle reveals $d_\ell + 3$ field elements: one entry of each witness row, of the two linear-mask rows, and of the proximity mask. We ignore the overhead of realizing the oracles, namely commitments (such as Merkle roots), the authentication paths that accompany opened columns, and salts. We also count only messages from the prover; the verifier's challenges and query indices are listed separately in Theorem 2.

Theorem 2 (Completeness and proof size). *Assume well-formed protocol data of depth L . Let σ_0 be a root statement, let w be a witness with $\text{Sat}(\sigma_0, w)$, and let W_0 be an encoding of w that satisfies the honest-encoding conditions of Section 4.3.*

- (i) *Perfect completeness. Consider the honest strategy: at each level $\ell < L$, after receiving γ_ℓ , the prover computes the honest responses (4.8)–(4.9), packs them into a child witness, and commits to its encoding by the child’s linear encoder with zero masks; at level L , it sends the honest responses. This is a prover strategy in the sense of Section 4.6, and the verifier accepts it for every sequence of verifier choices.*
- (ii) *Proof size. In every execution in which all mask coordinates are transmitted, the verifier reads $(d_\ell + 3)t_\ell$ field elements at each level ℓ it reaches, and the prover sends only the terminal responses, of $3k_L + s_L$ field elements. The proof size is therefore at most*

$$\sum_{\ell=0}^L (d_\ell + 3)t_\ell + 3k_L + s_L, \quad (4.21)$$

with equality for executions that reach level L . The verifier sends $|I_\ell| + (d_\ell + 2)(s_\ell + 1)$ challenge coefficients and t_ℓ query indices at level ℓ , and the level- ℓ oracle has $(d_\ell + 3)n_\ell$ field entries.

Proof. For (i), the child encoder exists because a child has no multiplication constraints, so its encoder need not know the later statement; the zero masks satisfy (4.3). For every parent query, Lemma 3 shows that the honest responses pass the parent’s tests, so Lemma 4 shows that the packed witness satisfies the induced statement. The child encoding again satisfies the honest-encoding conditions, so the same argument applies at every level, and Lemma 3 gives acceptance at level L . Every action uses only the preceding transcript, so this is a prover strategy.

For (ii), an opened column of the level- ℓ oracle holds the d_ℓ witness rows, the two linear-mask rows, and the proximity mask, and the verifier opens t_ℓ columns. Earlier responses are encoded in child oracles rather than sent, and the terminal responses have $R_L + 2k_L = 3k_L + s_L$ coefficients. The challenge γ_ℓ has $|I_\ell|$ linear and $m_\ell(s_\ell + 1) = (d_\ell + 2)(s_\ell + 1)$ polynomial coefficients. $\square$

A mask row that the protocol fixes publicly to zero need not be transmitted, which only lowers (4.21). Section 8 accounts for these omissions and adds commitments, authentication, and salts.

The security parameter must control both the query count and the field size. The following schedule gives a simple explicit bound; the numerical schedules in Section 8 use the sharper per-level formula directly.

Corollary 1 (Logarithmic proof size). *Let $\lambda \geq 4$ be an integer, and let the workload consist of N ordinary witness values and M multiplication gates. Then there exists a schedule over a binary field of size*

$$q = 2^{\lambda + \lceil \log_2 n \rceil}, \quad n 2^\lambda \leq q < 2n 2^\lambda,$$

with perfect completeness, soundness error at most $2^{-\lambda}$, and prover communication at most

$$t(15 + 16L) \text{ field elements,}$$

where

$$\begin{aligned} t &= \max \left\{ 3 \left\lceil \frac{\lambda + 3}{2} \right\rceil, 16 \right\}, & k_0 &= 2 \max \left\{ 1, \left\lceil \frac{N + 6M}{10} \right\rceil \right\}, \\ n &= 8k_0, & L &= \lceil \log_4 \max\{1, k_0/t\} \rceil. \end{aligned}$$

The communication bound is $O(\lambda(1 + \log(N + M + 1)))$ field elements, independently of the number of linear constraints.

The proof, a direct computation, appears in Appendix B. This is a general-purpose analytic bound. We apply numerical optimizations of the schedule to present sharper concrete communication bounds in §8.

Remark 1. This corollary counts prover-supplied field elements from opened column values and terminal response coefficients in the IOP model. It therefore excludes commitments and authentication data. For fixed λ , this count is logarithmic in $N + M + 1$. Each field element takes $\lambda + \lceil \log_2 n \rceil$ bits, so the corresponding bit bound is $O(\log^2(N + M + 1))$. $\square$

5 Zero knowledge

We prove that recursive Polygero with a masked root encoder is *perfect zero knowledge* in the oracle model, against verifiers that read a bounded number of root columns. To do so, we construct, for every such verifier, an efficient simulator that generates the verifier's view of an interaction with the honest prover from the statement and the verifier's auxiliary input, without knowing the witness. For every satisfying witness, the real and simulated views have exactly the same distribution. For a single invocation, the protocol is Ligerio with slightly different masks, so we recall the argument of [AHIV17, Lemma 4.13] briefly and concentrate on the differences: the proximity mask of degree bound R , the two linear masks, and the recursion.

Honest prover. Let $\Xi = \{\xi_a : a \in \mathcal{W}\}$ be the set of witness positions at the root. The honest prover encodes its witness with the ordinary root encoder of Section 4.4, with parameter r_{enc} , a set $E \subseteq \Omega \setminus \Xi$ of $|E| = r_{\text{enc}}$ points, and fresh randomness, and then follows the honest strategy of Theorem 2(i). Below the root, the encoders are the fixed linear encoders with zero masks, so the root encoding is the prover's only source of randomness.

Verifiers and views. A verifier V^* may deviate arbitrarily from the protocol, but its messages must have the prescribed form: a challenge $\gamma_\ell \in \Gamma_\ell$ at each level it reaches, in the message order (4.20), and, at each nonterminal level, a parent query $Q_\ell \in \mathcal{Q}_\ell$, which defines the child statement and consists of columns that

V^* reads. It has a *root query budget* of t_0 : after sending γ_0 , and at any later point of the execution, including after it has seen child data, it may read root columns adaptively, one at a time, but it reads at most t_0 distinct root columns in total. Repeated reads return the same column, and the parent query Q_0 counts against the same budget. Reads of child oracles are unrestricted. The honest verifier reads exactly the t_0 columns of Q_0 , so it satisfies the budget. The *view* of V^* consists of its auxiliary input, its random coins, the prover's explicit messages, and the answers to all of its oracle reads.

Theorem 3 (Perfect zero knowledge). *Assume well-formed protocol data of depth L , let P be the honest prover above, and let $t_0 \leq r_{\text{enc}}$. For every verifier V^* with root query budget t_0 , there is a simulator S that uses V^* as a black box, without rewinding, and runs in time polynomial in the public data, the oracle sizes, and the running time of V^* , with the following property. For every root statement σ_0 , every witness w with $\text{Sat}(\sigma_0, w)$, and every auxiliary input z , the output of $S(\sigma_0, z)$ has exactly the distribution of the view of $V^*(\sigma_0, z)$ in an interaction with $P(\sigma_0, w)$.*

Proof. Reduction to the root. It suffices to simulate the root's response pair (p, Z) jointly with the root columns that V^* reads. Indeed, at a nonterminal root, the child witness is $\text{pack}(p, Z)$, and the child oracle W_1 is its image under the fixed public encoder with zero masks, so the simulator constructs W_1 from (p, Z) before the parent query. The child statement σ_1 is determined by γ_0 , Q_0 , and the openings of Q_0 . Once the simulated openings pass the parent checks, exact substitution (Lemma 4) makes $\text{pack}(p, Z)$ a satisfying witness for σ_1 , as in an honest execution. Every later prover message is a deterministic function of this witness, the child statements, and the verifier's messages, so the simulator runs the honest recursive prover itself, while it continues to answer any later root reads. If the root is terminal, the prover's only message is the pair (p, Z) . Thus recursion needs no additional hiding randomness.

Witness rows. The additional *per-row entropy* hides the witness even though an opening reveals every row at the queried position. A uniform degree-bounded polynomial extending a row's witness entries can be written as $f_0 + \prod_{\xi \in \Xi} (X - \xi) \psi$, where f_0 is one extension and ψ is uniform in the remaining polynomial space. Multiplication inputs have r_{enc} free coefficients; linear rows have at least r_{enc} . At any at most r_{enc} code points, which lie outside Ξ , interpolation makes the queried values independent and uniform, regardless of the witness entries. Since V^* reads at most $t_0 \leq r_{\text{enc}}$ root columns, this applies to all of its root reads. Linear rows and multiplication-input rows use independent randomness. In an honest execution, the opened multiplication inputs are therefore independent uniform field elements, and the outputs are their products. The simulator reproduces this distribution by sampling the inputs and computing the products.

Responses. The *masks* hide the complete responses. The proximity mask $f_\star$ is uniform among polynomials of degree less than $R = k + s$. Since $\sum_r B_r f_r$ has degree less than R , adding $f_\star$ makes p uniform in that space, independently of all other honest polynomials. This is the reason why the mask must have degree bound R , rather than k .

For the linear response, consider the two linear masks f_{d+1} and f_{d+2} of Section 4.3, and recall that z_R vanishes on E . The encoder samples $f_{d+1} \in \mathcal{P}_{<k}$ and $f_{d+2} \in \mathcal{P}_{<k}$ independently and uniformly, subject to $\sum_{\xi \in \Xi} f_{d+2}(\xi) = 0$. The linear response Z includes the masking term $vf_{d+1} + z_R f_{d+2}$. This term is uniform over all polynomials of degree less than $2k$ that vanish on E and sum to zero over the witness positions. Indeed, for any such polynomial T , interpolate $f_{d+2} = T$ on $\Omega \setminus E$, with arbitrary values on E ; then $f_{d+1} = (T - z_R f_{d+2})/v$ has degree less than k . This is a surjective linear map, so uniform masks give a uniform image. The witness contribution vanishes on E and sums over the witness positions to the public value $\alpha^\top b$, where $\sigma_0 = (A, b)$. Consequently,

$$Z \text{ is uniform in } \left\{ Z \in \mathcal{P}_{<2k} : Z|_E = 0, \sum_{\xi \in \Xi} Z(\xi) = \alpha^\top b \right\},$$

independently of the witness rows. This affine space can be sampled by solving its public linear constraints; for a true statement it is nonempty.

Opened mask entries. Fixing the responses must also leave enough randomness for the opened mask entries. Put $\Pi = \prod_{y \in \Omega \setminus E} (X - y)$. For fixed witness rows and Z , the remaining freedom in f_{d+2} is exactly $f_{d+2}^{(0)} + \Pi\chi$, with χ uniform in $\mathcal{P}_{<r_{\text{enc}}}$: adding $\Pi\chi$ changes nothing on $\Omega \setminus E$, and the change in $z_R f_{d+2}$ is absorbed by vf_{d+1} . Since Π is nonzero at code points, the opened f_{d+2} -entries remain independent and uniform at up to r_{enc} positions. Fixing p imposes no further restriction, because the independent proximity mask absorbs it.

The simulator. The simulator runs V^* on (σ_0, z) and, after receiving γ_0 , samples p, Z from these public distributions. For each new root column, it samples independent uniform linear-row entries, multiplication inputs, and the f_{d+2} -entry, and sets multiplication outputs to the corresponding products. It solves the linear equation for the f_{d+1} -entry, dividing by the nonzero value $v(x_j)$, and the proximity equation for the $f_\star$ -entry. These are precisely the honest conditional distributions given p, Z . Caching answers gives the same joint law for adaptive queries within the total budget, including queries made after seeing child data. Continuing with the honest child prover thus reproduces the entire verifier view. All steps run in polynomial time in the public data, constructed oracle sizes, and verifier's running time, without rewinding. $\square$

6 Extraction and soundness

One of the key contributions of this paper is a novel method for analyzing the soundness of such an IOP through the construction and analysis of an explicit, deterministic extractor. In contrast to prior extractor algorithms, ours does not require list-decoding routines or the out-of-domain sampling idea [ACFY24] to support recursion beyond the unique-decoding regime.

The section is organized around the interface between a single invocation of the extractor and the recursion. Section 6.1 describes admissible parameters for our scheme and their induced per-level soundness error ε . Section 6.2 states what

an extractor for a single invocation must guarantee, and asserts that such an extractor exists (Theorem 4). This theorem for the single-shot extractor includes a few unusual clauses, and we explain why the recursion needs each of them.

To motivate its use, Section 6.3 then builds the recursive extractor Ext_ℓ from any extractor satisfying Theorem 4 and proves, by a comparatively short backward induction, the recursive extraction guarantee and soundness (Theorem 5). The construction of the per-level extractor and the proof of Theorem 4 are long and occupy Section 7; they rely on the MDS theorem of Section 3 and on a careful classification of exceptional events.

6.1 Admissible parameters

The next definition lists all of the constraints on the parameters needed to instantiate our system.

Definition 2 (Admissible parameters with error ε). *We say that a tuple of integer parameters (n, k, t, s, q) is admissible with error ε if*

$$\begin{aligned} k &\geq 1, \quad s \geq 0, \quad 1 \leq t \leq n, \quad h \leq n, \\ k + 2s - 1 &< n, \quad n + k \leq q, \quad \mu + 1/q < 1, \end{aligned} \tag{6.1}$$

and

$$\varepsilon = \theta + (1 - \theta)(\mu + 1/q), \tag{6.2}$$

where

$$R = k + s, \quad c = R + s + 1, \tag{6.3}$$

$$h = \max\{k + 2s + 1, 2k + s\}, \quad \mu = \binom{n}{c} / q^{s+1}. \tag{6.4}$$

$$\theta = \max\left\{\delta, \frac{u-1}{2(q-1)} + \frac{q}{2u}\omega\right\}, \tag{6.5}$$

$$\delta = \binom{h-1}{t} / \binom{n}{t}, \quad \omega = \binom{k+2s-1}{t} / \binom{n}{t}, \tag{6.6}$$

and u is the smallest positive integer with $u(u+1) \geq q(q-1)\omega$.

Here a binomial coefficient is zero when its lower index exceeds its upper index. Only ε , h , and s appear in the theorems of this section. The intermediate quantities are used in Section 7: δ bounds acceptance when few columns pass the verifier's tests, ω bounds the fraction of queries shared by the auxiliary counting sets of two distinct systems, μ bounds the rank exception, and θ is the counting threshold that combines δ and ω .

Admissibility implies $\theta < 1$, and hence $\varepsilon < 1$ because $\mu + 1/q < 1$. Indeed, $h \leq n$, $k + 2s - 1 < n$, and $1 \leq t \leq n$ give $\delta < 1$ and $\omega < 1$. The definition of u gives $u \leq q - 1$ and bounds the second term in the maximum defining θ by $u/(q-1)$. This is strictly less than one if $u < q - 1$; when $u = q - 1$, that term equals $(q - 2 + q\omega)/(2(q - 1)) < 1$.

The conditions $0 \leq t \leq n$, $R \leq n$, and $n + k \leq q$ imposed in Section 4.2 follow from admissibility, since $h > R$. The structural requirements on a recursive execution are those of well-formed protocol data (Section 4.6).

Remark 2 (Setting admissible parameters). Admissibility is easy to achieve with realistic parameters, because each term of ε has its own knob. Since $\binom{h-1}{t}/\binom{n}{t} \leq ((h-1)/n)^t$, the term δ decays exponentially in the number t of opened columns once the code length n exceeds h by a constant factor, and the same holds for ω with $k + 2s - 1$ in place of $h - 1$; the second term defining θ is then roughly $\sqrt{\omega}$. The rank term satisfies $\mu \leq 2^{-\lambda}$ as soon as $(s+1)\log_2 q \geq \log_2 \binom{n}{c} + \lambda$, so a large field keeps the challenge degree s small, and the term $1/q$ is negligible in such a field. For instance, the tuple $(n, k, t, s, q) = (25724, 1640, 64, 74, 2^{128})$ is admissible with error $\epsilon \leq 2^{-120}$: roughly, rate $1/16$ with 64 column openings yields 120 bits of soundness. More examples appear in §8. $\square$

Remark 3 (No proximity or Johnson-gap parameter). Analyses based on proximity gaps, such as [BSCI⁺20], state their errors in terms of a proximity parameter: the relative distance that the verifier’s column test is meant to detect, kept below the Johnson radius $1 - \sqrt{\rho}$ by a margin η . The number of queries depends on that distance, and the error terms grow polynomially in n and $1/\eta$ as the distance approaches the Johnson radius, so the margin must be tuned. Definition 2 has no such parameter. The extractor decodes only within the unique-decoding radius s on the columns that pass the tests, and the quantity that plays the role of distance is the agreement threshold h inside δ . A Johnson-like square root still appears, as $\theta \approx \sqrt{\omega}$, but it arises from a second-moment count of the pairwise overlaps between candidate systems in Section 7, the same kind of argument that proves the Johnson bound, rather than from a radius fixed in advance. The trade-off between queries and error is therefore settled by exact counting, with no margin to optimize and no factor of $1/\eta$. $\square$

6.2 The guarantee required of a per-level extractor

We first state what an extractor Ext for a single invocation must achieve. Section 7 constructs such an extractor and proves that it meets this guarantee.

A per-level extractor takes as input the parameters (n, k, t, s, q) and public layout of one invocation, a statement σ , the full committed oracle W , which is a matrix of arbitrary words, the verifier’s challenge γ , and the prover’s response pair (p, Z) . It outputs either a polynomial tuple g , whose induced witness satisfies σ , or $\perp$. Ideally, Ext fails only when the verifier is unlikely to accept: averaged over the challenge, the fraction of queries accepted while Ext fails should be at most the per-level error ε . This must hold for every response the prover might send, since the prover chooses its response after seeing the challenge. The recursion requires a stronger form of this guarantee, for two reasons: the prover commits to each child oracle after seeing the parent challenge, and several child tuples may each encode a response that passes the parent’s checks. We now explain the resulting clauses.

Recall the steps at a nonterminal invocation with parent statement σ . With the parent oracle W already committed, the verifier sends the parent challenge γ . An ordinary invocation would next receive the proximity response p and linear response Z ; these are the *parent response pair*. In the recursive protocol, the parent response pair is encoded in a child tuple of polynomials: the coefficients of p and Z occupy fixed logical positions in the child witness, and the child polynomials take these values at those positions. The prover commits to a child oracle intended to contain evaluations of such polynomials. The verifier then samples the parent query Q , opens those columns of W , and checks their multiplication constraints. If these checks pass, the remaining parent tests become linear constraints on the encoded coefficients. These constraints form the child statement, which the child invocation proves using its own fresh challenge.

The extractor works in the reverse direction. If child extraction returns a polynomial tuple g' , evaluate it at the child logical positions to obtain $w_{g'}$, then read the slots assigned to the coefficients of p and Z . This is the operation $(p, Z) = \text{unpack}(w_{g'})$. The slot assignment is fixed before execution, so each fixed child tuple determines a parent response pair independently of the parent query. The tuple actually returned by child extraction may nevertheless vary with that query. The parent extractor then runs $\text{Ext}(W, \sigma, \gamma, p, Z)$. Even a valid child witness can encode responses that pass every queried column but fewer than h columns in the full parent oracle, causing the parent extractor to fail. Thus we must count certain successful child outputs as well as child extraction failures.

A successful parent output may itself encode responses on which extraction fails higher in the recursion. We therefore track specified outputs as well as outright failure. Fix the parent oracle, statement, and challenge, and a set $\mathcal{U}$ of outputs we wish to exclude. Call a parent response pair *unwanted* if the parent extractor fails on it or returns a tuple in $\mathcal{U}$. Call a child tuple *unwanted* if its encoded parent response pair is unwanted. At the root, take $\mathcal{U} = \emptyset$, so we count only extraction failure. This classification does not depend on the parent query.

For each fixed unwanted child tuple, we count the parent queries on which its encoded responses make the parent verifier accept. The child statement changes with the query, but the tuple's encoded responses and the parent extractor's result do not. The parent's per-level guarantee bounds this fraction separately for each unwanted tuple.

We now analyze recursion using an abstract guarantee at each level. The theorem below bounds acceptance when the extractor fails or returns an output in an unwanted set $\mathcal{U}_i$. In the child application, the indices i are parent queries Q , and we denote the child sets by $\mathcal{U}'_Q$. Membership in $\mathcal{U}'_Q$ requires two conditions: the child tuple is unwanted, and it certifies acceptance for Q . The latter means that the parent's direct checks pass and the tuple satisfies the induced child statement. Thus the sets record which queries an unwanted tuple can explain. They may vary with Q even though each fixed tuple's unwantedness does not. We count queries separately even when they induce the same child statement.

Theorem 4 (A per-level extractor with a response-uniform bound).

Let (n, k, t, s, q) be admissible parameters with error ε (Definition 2), and consider an invocation of Section 4.4 with these parameters. Then there is a single-level extractor Ext with the following properties.

- (i) Determinism and efficiency. Ext is deterministic, does not read the verifier's query, and runs in polynomial time in its input.
- (ii) Valid outputs. Every tuple $g \neq \perp$ returned by $\text{Ext}(W, \sigma, \gamma, p, Z)$ induces a witness w_g that satisfies σ and agrees with the oracle on at least $h-s$ columns, i.e. on at least $h-s$ columns j with $W_{r,j} = g_r(x_j)$ for every witness row r .
- (iii) Response-uniform bound. Let W be any oracle, shared by all indices, and let $(\sigma_i, \mathcal{U}_i)_{i \in \mathcal{I}}$ be any family, where $\mathcal{I}$ is finite and nonempty, each σ_i is a statement for the invocation's layout, and $\mathcal{U}_i$ is a set of polynomial witness tuples, called unwanted for index i . The oracle and family are fixed before the fresh challenge γ . Let $0 \leq \rho \leq 1$. Assume that, for all tuples g , the fraction of indices i for which g belongs to $\mathcal{U}_i$ is at most ρ :

$$\frac{|\{i \in \mathcal{I} : g \in \mathcal{U}_i\}|}{|\mathcal{I}|} \leq \rho. \quad (6.7)$$

Then there are functions $H_i : \Gamma \rightarrow [0, 1]$, independent of the actual responses, with the following properties:

- (a) For all $i \in \mathcal{I}$, challenges γ , and response pairs (p, Z) , if $\text{Ext}(W, \sigma_i, \gamma, p, Z)$ fails or returns a tuple in $\mathcal{U}_i$, the accepted-query fraction is at most $H_i(\gamma)$.
- (b)
$$\frac{\sum_{i \in \mathcal{I}} \sum_{\gamma \in \Gamma} H_i(\gamma)}{|\mathcal{I}| |\Gamma|} \leq \varepsilon + (1 - \varepsilon)\rho.$$

Proof. Section 7.1 constructs Ext and verifies properties (i) and (ii). The rest of Section 7 proves property (iii); the proof is completed in Section 7.9. $\square$

Remark 4 (Interpretation of $H_i(\gamma)$). For an index i and challenge γ , consider any response pair for which the extractor fails or returns a tuple in $\mathcal{U}_i$. The verifier accepts at most a fraction $H_i(\gamma)$ of that level's queries. The same bound holds for every such response pair, and it need not be zero or one. The parameter ρ bounds the fraction of indices whose sets $\mathcal{U}_i$ contain any one tuple. In the child application, it therefore bounds the fraction of parent queries for which an unwanted child tuple certifies acceptance. The queries bounded by $H_i(\gamma)$ are then child queries. $\square$

Taking all $\mathcal{U}_i$ empty gives the per-level extraction-failure bound.

6.3 Recursive extraction and soundness

The recursive extractor Ext_ℓ in Figure 1 works backward through a complete transcript using an Ext extractor that satisfies Theorem 4. It returns $\perp$ immediately if the verifier rejected. Otherwise it starts with the terminal responses

```

Require: Level  $\ell$ ; a complete transcript with committed oracles  $W_\ell, \dots, W_L$ ,
statements  $\sigma_\ell, \dots, \sigma_L$ , challenges  $\gamma_\ell, \dots, \gamma_L$ , and terminal responses  $p_L, Z_L$ .
1: if the verifier rejected then
2:   return  $\perp$ 
3: end if
4:  $(p, Z) \leftarrow (p_L, Z_L)$ 
5: for  $j = L, L-1, \dots, \ell$  do
6:    $g \leftarrow \text{Ext}(W_j, \sigma_j, \gamma_j, p, Z)$ 
7:   if  $g = \perp$  then
8:     return  $\perp$ 
9:   end if
10:  if  $j > \ell$  then
11:     $(p, Z) \leftarrow \text{unpack}(w_g)$   $\triangleright$  responses of level  $j-1$ 
12:  end if
13: end for
14: return  $g$ 

```

Fig. 1. The recursive extractor Ext_ℓ . For $j > \ell$, the statement σ_j is the one induced by the transcript at level $j-1$ (Section 4.5).

(p_L, Z_L) and, at each level j from L down to ℓ , runs $\text{Ext}(W_j, \sigma_j, \gamma_j, p, Z)$, returning $\perp$ if it fails. If $j > \ell$, it evaluates the returned tuple to obtain w_g and unpacks w_g into the responses of the parent level. At level ℓ , it returns the decoded tuple.

Padding rejected executions, as in Section 4.6, is used only to count complete verifier sequences in the proof. Since Ext is deterministic and runs in polynomial time (Theorem 4(i)), so does Ext_ℓ , in the full accessed data, explicit public layouts, and finite-field representation.

Remark 5 (The extracted tuple satisfies the statement). Every successful output of Ext_ℓ induces a witness satisfying the statement at level ℓ , since the final call to Ext returns only satisfying tuples (Theorem 4(ii)). $\square$

We now bound how often the verifier accepts while the entire recursive extractor fails. We work backward from the terminal level. At each step, the parent bound controls how many queries an unwanted child tuple's encoded responses can pass. The child guarantee then bounds acceptance when the child extractor fails or returns such a tuple.

Theorem 5 (Polygero Knowledge-soundness). *Consider a recursive execution with well-formed protocol data, in which the parameters (n_j, k_j, t_j, s_j, q) of each level j are admissible with error ε_j . Let Ext be a single-level extractor satisfying Theorem 4 at every level, and let Ext_ℓ be the recursive extractor of Figure 1 built from it. Then, for every root oracle, root statement, and prover strategy (Section 4.6),*

$$\Pr[\text{Acc} \wedge \text{Ext}_0 = \perp] \leq \kappa_0 = 1 - \prod_{j=0}^L (1 - \varepsilon_j) \leq \sum_{j=0}^L \varepsilon_j, \quad (6.8)$$

where the probability is over the verifier's random choices. Every successful output of Ext_0 induces a witness that satisfies the root statement (Remark 5). In particular, if no satisfying polynomial tuple agrees with the root oracle on at least $h_0 - s_0$ common columns, which holds for every false root statement, then every prover strategy is accepted with probability at most κ_0 .

Proof. Put $\kappa_\ell = 1 - \prod_{j=\ell}^L (1 - \varepsilon_j)$. The bound (6.8) cannot be proved by induction on its own, because a successful child extraction can encode a parent response on which the parent extractor fails. We therefore prove a stronger claim, by backward induction on ℓ . Let W_ℓ be any oracle, shared by all indices, and fix a nonempty finite family of statements σ_i for level ℓ 's layout, prover strategies for the remaining levels, and unwanted sets $\mathcal{U}_i$, indexed by $\mathcal{I}$, before the remaining verifier choices. Suppose that, for every tuple g , at most a fraction ρ of the indices satisfy $g \in \mathcal{U}_i$. Write Acc_i for acceptance and $\text{Ext}_{\ell,i}$ for the output of Ext_ℓ on index i . The claim is that

$$\frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \Pr[\text{Acc}_i \wedge (\text{Ext}_{\ell,i} = \perp \vee \text{Ext}_{\ell,i} \in \mathcal{U}_i)] \leq \kappa_\ell + (1 - \kappa_\ell)\rho. \quad (6.9)$$

With $\ell = 0$, a single index, $\mathcal{U} = \emptyset$, and $\rho = 0$, this is (6.8); iterating $\kappa_\ell \leq \varepsilon_\ell + \kappa_{\ell+1}$ gives the sum bound.

For the base case, apply Theorem 4 to the current family to obtain the functions H_i . At the terminal level, for each i and challenge γ , the response is fixed before the query. Its accepted-query fraction is at most $H_i(\gamma)$ whenever extraction fails or returns a tuple in $\mathcal{U}_i$. Averaging over i and γ proves the base case with $\kappa_L = \varepsilon_L$.

For the induction step, assume the claim for every child oracle and indexed family, and apply Theorem 4 to the parent family to obtain the functions H_i . Write $\kappa' = \kappa_{\ell+1}$ and suppress parent level subscripts. Fix an index i and the parent challenge γ . The child oracle V is now fixed before the parent query Q . For a child tuple g' , write $\pi(g') = \text{unpack}(w_{g'})$. Call g' unwanted if $\text{Ext}(W, \sigma_i, \gamma, \pi(g'))$ fails or returns a tuple in $\mathcal{U}_i$. Its encoded responses and this classification are fixed before Q : coefficient placement is fixed, and the parent extractor does not use Q (Theorem 4(i)).

Define the child's unwanted sets by

$$\mathcal{U}'_Q = \{g' : g' \text{ is unwanted} \wedge \text{Verify}(W, \sigma_i, \gamma, \pi(g'), Q)\}.$$

By exact substitution (Lemma 4), membership means that the tuple is unwanted, satisfies the induced child statement, and passes the parent's direct checks. In particular, the set is empty when a direct check fails. Only the root may have multiplication constraints; at every other level these checks pass automatically.

If the child extractor returns a tuple in $\mathcal{U}'_Q$, it succeeds for the child statement but supplies responses on which the parent extractor fails or returns an unwanted output. We must therefore count these outputs even when child extraction succeeds.

For a fixed unwanted tuple g' , membership in $\mathcal{U}'_Q$ is exactly acceptance of $\pi(g')$ on query Q . Hence

$$\frac{|\{Q \in \binom{[n]}{t} : g' \in \mathcal{U}'_Q\}|}{\binom{n}{t}} = \text{acc}_W(\sigma_i, \gamma, \pi(g')) \leq H_i(\gamma),$$

where the inequality is the per-level bound applied to $\pi(g')$. Any other tuple belongs to no $\mathcal{U}'_Q$, so its query fraction is zero. Thus the frequency bound holds for all child tuples, without a union over tuples.

If an execution accepts but the parent extractor fails or returns a tuple in $\mathcal{U}_i$, then the child extractor either fails or returns a tuple in $\mathcal{U}'_Q$. Indeed, a successful child output is unwanted and satisfies the child statement. Acceptance ensures that the parent's direct checks pass, so exact substitution gives the parent verification condition in the definition of $\mathcal{U}'_Q$.

Apply the child induction hypothesis to the family indexed by parent queries Q , using the common oracle V and unwanted sets $\mathcal{U}'_Q$. The child statements, strategies, and sets are fixed before the child challenge, and the frequency bound $H_i(\gamma)$ lies in $[0, 1]$. After averaging over Q and all subsequent child randomness, the probability of acceptance with parent extraction failure or an output in $\mathcal{U}_i$, with i and γ still fixed, is at most

$$\kappa' + (1 - \kappa')H_i(\gamma).$$

Since $1 - \kappa' \geq 0$, we can now average over i and the parent challenge γ and apply the average bound on H_i . This gives

$$\begin{aligned} \kappa' + (1 - \kappa')(\varepsilon + (1 - \varepsilon)\rho) &= \kappa + (1 - \kappa)\rho, \\ \kappa &= \varepsilon + (1 - \varepsilon)\kappa'. \end{aligned}$$

This recurrence gives the product formula for κ_ℓ and completes the induction. Randomized strategies are covered by fixing their internal randomness, applying the bound, and then averaging.

For the last assertion, a successful root extraction returns a satisfying tuple that agrees with the root oracle on at least $h_0 - s_0$ common columns, by Theorem 4(ii). Under the hypothesis no such tuple exists, so every accepting execution is an extraction failure, and (6.8) bounds acceptance by κ_0 . $\square$

Remark 6 (From the theorem to a concrete soundness error). The last assertion of Theorem 5 is the soundness bound: on a false statement, every prover strategy is accepted with probability at most κ_0 . The bound is explicit: each ε_j is computed from (n_j, k_j, t_j, s_j, q) by Definition 2, and $\kappa_0 \leq \sum_j \varepsilon_j$. It is not vacuous, because every term of ε can be made small. Since $\binom{h-1}{t}/\binom{n}{t} \leq ((h-1)/n)^t$, the term δ decays exponentially in the number t of opened columns when the rate h/n is bounded below one, and similarly for ω ; the second term in the maximum defining θ is roughly $\sqrt{\omega}$. The rank term μ is small once q^{s+1} exceeds $\binom{n}{c}$, and $1/q$ is small in a large field. Corollary 1 exhibits schedules with soundness error at most $2^{-\lambda}$, and Section 8 reports schedules whose errors are certified exactly. $\square$

The product bound assumes no independence between extraction failures. Theorem 5 has no frequency hypothesis; it appears only in the stronger claim proved inside its proof. The extractor and the interactive, full-oracle scope remain those of Section 7.1.

7 Polygero base extractor

This section proves Theorem 4 by constructing and analyzing the one-level extractor that Section 6 uses at every level of the recursion.

The core argument. Most of this section bounds exceptional events; ignoring them, the extractor argument is short (Figure 2). Fix a challenge $\gamma = (\alpha, B)$ and a response that passes the degree and global checks, and let G be its good-column set. If $|G| < h$, the accepted fraction is at most δ (Lemma 6). Suppose instead that $|G| \geq h$, that the challenge avoids the rank exception (Definition 5), and that no system exception occurs (Definition 10). Then the proximity equations on G have rank defect at most s (Lemma 8). Removing at most s columns leaves a canonical core, whose closure D belongs to a family of systems fixed from W alone, independently of the response (Lemma 9). Since no system exception occurs, D has rank defect zero, so its equations determine fixed row polynomials g_D . With at least h good columns, unique decoding recovers g_D ; the many good columns force the multiplication constraints, and the global linear check forces the linear constraints. Thus Ext returns the fixed tuple g_D (Lemma 19).

The rest of the section bounds what the exceptions cost, which produces the terms of $\varepsilon = \theta + (1 - \theta)(\mu + 1/q)$. The rank exception has probability at most μ (Lemma 8), and each fixed system is exceptional, that is, can pass the checks without yielding a satisfying witness, with probability at most $1/q$ (Lemma 16). The difficult part is adaptivity: the prover chooses its response after seeing the challenge, so it can steer its good set onto whichever system is exceptional for that challenge and has the largest query mass. A union bound over systems would pay for their number, which can be very large. Instead, we prove that actual consistent sets assigned to different systems have pairwise overlap at most $k + 2s - 1$ (Lemma 13). Its auxiliary rank argument (Lemma 12) is the sole use of Theorem 1 in the analysis. The query fractions above the threshold θ have small total excess, and the average largest exceptional mass is at most $\theta + (1 - \theta)/q$ (Lemma 20). A system whose fixed tuple is unwanted is charged in the same way, which produces the term with ρ in Theorem 4. The systems, both kinds of exceptions, and the unwanted sets are fixed from the committed oracle and statement before the prover selects its response, so the bound holds simultaneously for all response pairs.

7.1 Preliminary definitions

Fix the invocation data and encoding layout of Section 4.2. For arbitrary words W , a polynomial tuple $g = (g_1, \dots, g_d) \in (\mathcal{P}_{<k})^d$ has common agreement set

$$D_g(W) = \{j \in [n] : \forall r \in [d] : W_{r,j} = g_r(x_j)\} \quad (7.1)$$

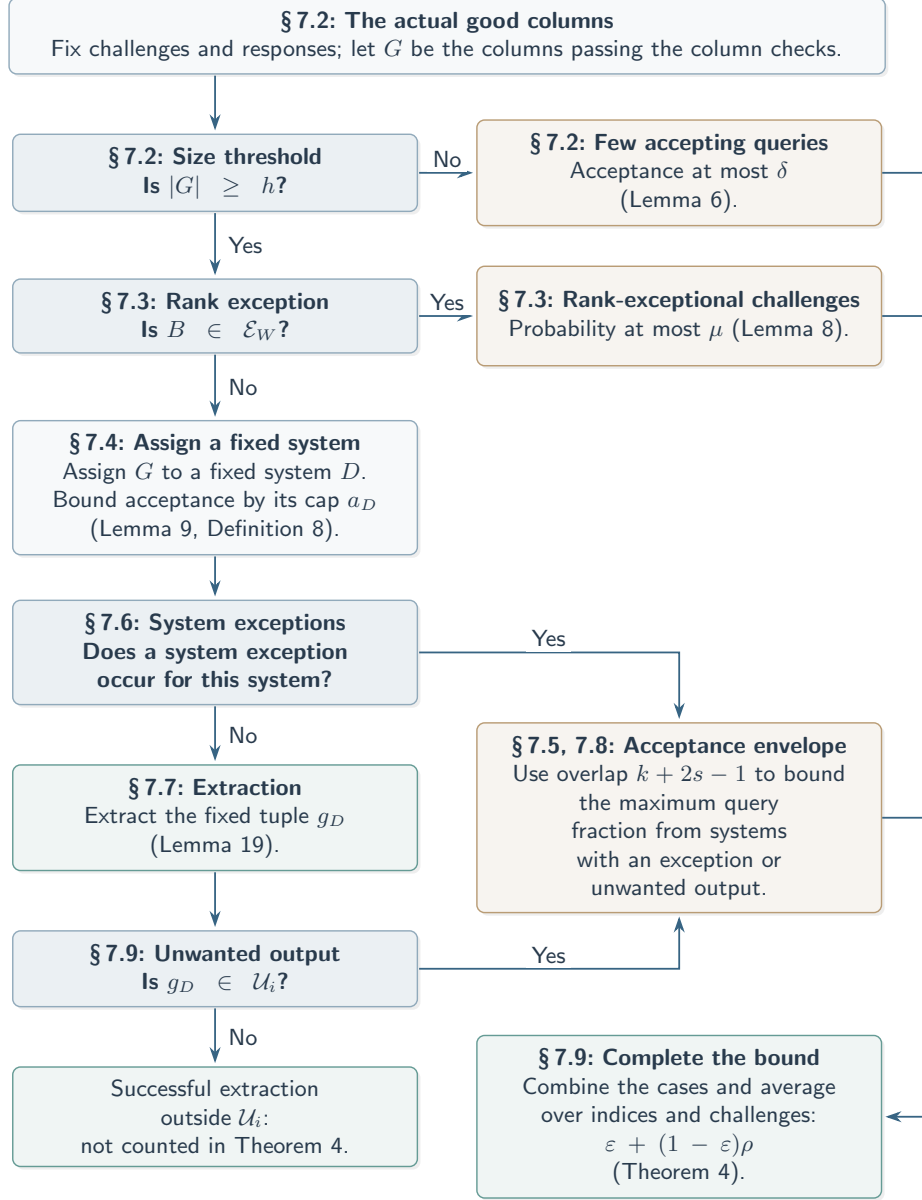


Fig. 2. The analysis of accepting executions after the response degree and global checks pass. Fix an index i , its statement, oracle, and unwanted set $\mathcal{U}_i$. A large good set outside the rank exception is assigned to a fixed system. Without a system exception, extraction returns g_D ; this output still contributes to the bound if it belongs to $\mathcal{U}_i$. The right-hand branches feed the final bound, which combines the cases in a response-uniform envelope and averages it. The counted cases may also contain successful extractions.

and induces the witness

$$w_g = (g_r(\xi_b))_{r \in [d], b \in \mathcal{W}}. \quad (7.2)$$

Only witness rows enter $D_g(W)$; mask rows are excluded. Recall the threshold h from (6.4). Its first term ensures that a large good set has strictly more columns than its rank outside the rank exception, so canonical reduction applies. Its second term leaves at least $2k$ common agreement columns after discarding s errors, enough to identify the linear response. Unique decoding alone would require only $k + 2s$ columns.

Even though the number of queries is much smaller than required for unique decoding, the extractor needs only a unique-decoding-radius decoder that supports puncturing, for instance Berlekamp–Welch. It decodes on all good columns of the committed oracle and does not require a list decoder.

Definition 3 (Decoder for a punctured code). *Let $\text{Enc} : \mathcal{M} \rightarrow \mathbb{F}^n$ be a code with message set $\mathcal{M}$. On a coordinate set $S \subseteq [n]$, a received word $w \in \mathbb{F}^S$, and radius e , a bounded-distance decoder returns a message $f \in \mathcal{M}$ or $\perp$. Every returned message satisfies*

$$|\{j \in S : w_j \neq \text{Enc}(f)_j\}| \leq e.$$

Whenever the encodings of distinct messages differ on more than $2e$ coordinates of S , the decoder must return the unique such message if one exists, and $\perp$ otherwise.

For the degree- $< k$ evaluation code, distinct polynomials disagree on at least $|S| - k + 1$ coordinates, so $|S| \geq k + 2e$ permits unique decoding on the punctured code. The classical Berlekamp–Welch algorithm achieves this efficiently.

Lemma 5 (Berlekamp–Welch decoder [WB86]). *There is a deterministic algorithm Decode_k that, on a coordinate set $S \subseteq [n]$ with $|S| \geq k + 2e$, a received word $w \in \mathbb{F}^S$, and a radius e , is a bounded-distance decoder for the degree- $< k$ evaluation code punctured to S , in the sense of Definition 3, and uses a number of field operations polynomial in $|S|$.*

The extractor uses Decode_k with $e = s$; its output is a polynomial of degree less than k . This radius is small. The full code $\mathcal{C}_k = \text{RS}[n, k]$ has minimum distance $n - k + 1$, so its unique-decoding radius is $\lfloor (n - k)/2 \rfloor$, but the analysis never needs to correct that many errors. The extractor decodes only on the good columns, and only when $|G| \geq h$. The punctured code on G has minimum distance $|G| - k + 1$, and the first term of h in (6.4) gives $|G| \geq h \geq k + 2s + 1$. Hence

$$s \leq \frac{|G| - k - 1}{2} \leq \frac{n - k - 1}{2},$$

using $|G| \leq n$. The radius s therefore lies within the unique-decoding radius of the punctured code, and hence of the full code, so Lemma 5 applies with $S = G$ and $e = s$. The number of errors to tolerate is set by the challenge degree

```

Require: Parameters  $(n, k, t, s, q)$  and public layout; statement  $\sigma$ ; full oracle  $W$ ,
a matrix of arbitrary words; challenge  $\gamma$ ; responses  $p, Z$ .
1: if  $p \notin \mathcal{P}_{<R}$  or  $Z \notin \mathcal{P}_{<2k}$  or (4.14) fails then
2:   return  $\perp$ 
3: end if
4: Scan all columns and compute  $G = G_W(\sigma, \gamma, p, Z)$ .
5: if  $|G| < h$  then
6:   return  $\perp$ 
7: end if
8: Compute  $g_r = \text{Decode}_k(G, W_r|_G, s)$  for each witness row  $r$ .
9: Return  $\perp$  if any row fails to decode; otherwise let  $g$  be the decoded tuple.
10: if  $|G \setminus D_g(W)| > s$  or  $\text{Sat}(\sigma, w_g)$  fails then
11:   return  $\perp$ 
12: end if
13: return  $g$ .

```

Fig. 3. The per-level extractor $\text{Ext}(W, \sigma, \gamma, p, Z)$, which performs deterministic extraction at one invocation.

s , which also bounds the rank defect, rather than by the distance of the code; admissibility, through $h \leq n$, guarantees that a good set of size h fits.

Figure 3 gives the extractor. After decoding each witness row on the good columns G of (4.15), it checks that all row errors lie in *one common set of at most s columns*, and that the induced witness satisfies the statement. The common-error check is necessary, because separate row decoders could disagree with the oracle on different columns.

Decoding can fail even when a satisfying polynomial tuple agrees with many oracle columns. The verifier checks the queried columns, whereas the extractor decodes on all good columns; agreement on the former need not imply proximity on the latter. The per-level analysis in Section 7 controls this discrepancy. The mask rows are used to identify good columns but are not decoded, and Ext does not use the query. Every successful output of Ext satisfies the statement, since its final step checks $\text{Sat}(\sigma, w_g)$. Properties (i) and (ii) of Theorem 4 are immediate from Figure 3. The extractor is deterministic, never reads the query, and runs in polynomial time, since it calls the decoder of Lemma 5 only when $|G| \geq h > k + 2s$. It returns g only after checking $|G| \geq h$, $|G \setminus D_g(W)| \leq s$, and $\text{Sat}(\sigma, w_g)$, so $|D_g(W)| \geq h - s$. The rest of this section proves property (iii).

7.2 Counting accepting queries

Our goal is to partition accepting executions into cases: one guarantees successful extraction, and the others account for all possible extraction failures. We begin by counting the queries accepted by a response pair (p, Z) . Use the verifier of Section 4.4, and fix the oracle W , statement σ , verifier challenge $\gamma = (\alpha, B)$.

Definition 4 (Good columns and query space). Let $G = G_W(\sigma, \gamma, p, Z)$ be the columns on which its proximity, linear, and multiplication checks pass, as in (4.15).

The query is uniform in

$$\mathcal{Q} = \{Q \subseteq [n] : |Q| = t\}, \quad |\mathcal{Q}| = \binom{n}{t}.$$

We use the convention $\binom{v}{u} = 0$ for $u < 0$ or $u > v$, when v is a nonnegative integer.

Lemma 6 (Accepting-query count). If the degree and global checks pass, then the accepting-query fraction is exactly

$$\text{acc}_W(\sigma, \gamma, p, Z) = \frac{|\{Q \in \mathcal{Q} : Q \subseteq G\}|}{|\mathcal{Q}|} = \frac{\binom{|G|}{t}}{|\mathcal{Q}|}. \quad (7.3)$$

Otherwise it is zero. In particular, if $|G| < h$, the accepting-query fraction is at most δ .

Proof. The response pair is fixed before the query, and there are $|\mathcal{Q}|$ possible queries, exactly $\binom{|G|}{t}$ of which lie in G . A failed degree or global check rejects every query. If $|G| \leq h-1$, then $\binom{|G|}{t} \leq \binom{h-1}{t}$, since binomial coefficients increase in their upper index, so the fraction is at most $\binom{h-1}{t}/|\mathcal{Q}| = \delta$. $\square$

This handles responses with a small good-column set. It remains to analyze responses that pass on many columns: either those columns yield a satisfying witness, or we bound their contribution to acceptance.

7.3 Count challenges that can support too many independent equations

Each good column gives a proximity equation: the response value equals the mask value plus the challenged linear combination of oracle-row values. The response has only finitely many degrees of freedom with which to satisfy these equations. Once these are exhausted, every further independent equation constrains the challenge. We count how often these constraints can hold, already allowing the prover to send any bounded-degree response.

Recall that $R = k + s$ and $m = d + 2$. At each good column j , the proximity check requires

$$p(x_j) = (Y_B)_j = W_{\star,j} + \sum_{r=1}^m B_r(x_j)W_{r,j}. \quad (7.4)$$

With W held fixed, view it as an affine linear equation in the coefficients of both B and p :

$$v_j(B, p) := p(x_j) - \sum_{r=1}^m B_r(x_j)W_{r,j} - W_{\star,j} = 0. \quad (7.5)$$

For a column set T , let $\text{rk}(T)$ be the rank of the coefficient matrix in the coefficients of B and p . Call T consistent if the equations admit a solution (B, p) . For a consistent system, this rank also equals the dimension of the span of its affine forms v_j . If T is consistent and $|T| \geq R$, define its rank defect by

$$\delta_T = \text{rk}(T) - R.$$

The subtraction of R has a concrete meaning:

With B fixed, the first R independent equations determine the R response coefficients. Every additional independent equation imposes a condition on the challenge.

Lemma 7 (Rank defect makes consistent challenges rare). *If T is consistent and $|T| \geq R$, then $\delta_T \geq 0$ and*

$$\Pr_{B \in \mathcal{B}} [\exists p \in \mathcal{P}_{<R} : p \text{ satisfies the equations on } T] = q^{-\delta_T}. \quad (7.6)$$

The probability is over the uniform challenge B , with W and T fixed. The response p may depend on B .

Proof. The challenge has $m(s+1)$ coefficients, and the response has R . There are therefore $m(s+1) + R$ variables. Because the system is consistent, translation by a solution removes the constant terms without changing its rank. The response block has rank R , which uniquely determines p once B is fixed: a polynomial of degree less than R is determined by its values at R distinct columns. Hence $\delta_T \geq 0$, and the number of solution pairs is

$$q^{m(s+1)+R-(R+\delta_T)} = q^{m(s+1)-\delta_T}.$$

Each challenge B for which the equations have a solution therefore corresponds to exactly one pair (B, p) . Dividing by the $q^{m(s+1)}$ possible challenges proves the claim. $\square$

We now make the exceptional event independent of the response. We allow rank defect at most s and put $c = R + s + 1$. A consistent set of c independent equations has rank defect $s + 1$, so it is satisfied by some response on only a $q^{-(s+1)}$ fraction of challenges. We collect all such sets before the response is selected.

Definition 5 (Fixed rank exception). *Let $\mathcal{E}_W$ be the set of challenges B for which some response $p \in \mathcal{P}_{<R}$ satisfies the equations on a c -element column set T whose equation system has full rank, $\text{rk}(T) = c$.*

There are $\binom{n}{c}$ possible c -column sets, which yields the following result.

Lemma 8 (Low rank defect outside a rare exception). *Let B be uniform in $\mathcal{B}$. Then*

$$\Pr[B \in \mathcal{E}_W] \leq \mu := \frac{\binom{n}{c}}{q^{s+1}}. \quad (7.7)$$

If $B \notin \mathcal{E}_W$ and some response satisfies the equations on T for this B , with $|T| \geq c$, then $\delta_T \leq s$.

Proof. Every column set in Definition 5 has rank defect $c - R = s + 1$. Lemma 7 gives probability $q^{-(s+1)}$ for each consistency event. There are at most $\binom{n}{c}$ sets, so the union bound gives the stated probability.

If a response satisfies the equations on T and $\text{rk}(T) \geq c$, pick c independent equations from T . The actual challenge and response satisfy all of them, so $B \in \mathcal{E}_W$. Consequently, outside $\mathcal{E}_W$, every T with $|T| \geq c$ admitting a response for the actual challenge B has rank less than $c = R + s + 1$, and therefore $0 \leq \delta_T \leq s$. $\square$

The good-column set depends on the response, but $\mathcal{E}_W$ does not. If $B \notin \mathcal{E}_W$, every response with at least c good columns gives a system of rank defect at most s .

7.4 Canonical cores and fixed systems

Outside $\mathcal{E}_W$, a large good-column set has low rank defect. We first remove equations independent of all the other equations on that set, then close the remaining system under implication. These elementary steps assign every large good set to a system fixed from the oracle alone. The counting argument requires that good sets assigned to different systems cannot overlap too much.

For a column set $T \subseteq [n]$, let $C_R(T)$ be the evaluations of degree- $< R$ polynomials on T , and consider the received-word space

$$U_T = C_R(T) + \text{span}_{\mathbb{F}}\{(x_j^a W_{r,j})_{j \in T} : r \in [m], 0 \leq a \leq s\}. \quad (7.8)$$

The coefficient matrix of the primary equations has one row for each $j \in T$ and one column for each coefficient of B and p . Its columns span U_T . Restricting its rows to $H \subseteq T$ gives the coefficient matrix on H , whose columns span $U_T|_H$. Equality of row rank and column rank therefore gives

$$\text{rk}(H) = \dim(U_T|_H) \quad (H \subseteq T). \quad (7.9)$$

For a consistent T , the constant column $-W_\star|_T$ is a linear combination of the coefficient columns. Appending it preserves all row relations, so the same rank is the dimension of the span of the affine forms v_j .

Let

$$E_T = \{j \in T : \text{the coordinate unit vector at } j \text{ belongs to } U_T\}.$$

This is the exceptional set specified in Theorem 1. The following reduction does not require that theorem.

Lemma 9 (Canonical reduction). *Let T be consistent, with $0 \leq \delta_T \leq s$ and $|T| > R + \delta_T$. Define its canonical core by $S_T = T \setminus E_T$. Then*

$$|E_T| \leq \delta_T, \quad \delta_{S_T} = \delta_T - |E_T|, \quad |S_T| > R + \delta_{S_T}, \quad E_{S_T} = \emptyset. \quad (7.10)$$

Proof. The coordinate unit vectors indexed by E_T belong to U_T , so U_T contains every word supported on E_T . These words are exactly the kernel of restriction from U_T to S_T . Rank-nullity and (7.9) give

$$\text{rk}(S_T) = \text{rk}(T) - |E_T|, \quad |S_T| = |T| - |E_T| > \text{rk}(S_T).$$

The response code $C_R(S_T)$ is contained in U_{S_T} , so

$$\min\{R, |S_T|\} \leq \text{rk}(S_T) < |S_T|.$$

Thus $|S_T| > R$ and $\text{rk}(S_T) \geq R$. The restricted system is consistent, so its defect is defined; the rank identity now gives $\delta_{S_T} = \delta_T - |E_T| \geq 0$ and $|E_T| \leq \delta_T$.

Finally, if a coordinate unit vector at $j \in S_T$ belonged to U_{S_T} , lift it to U_T . Subtract its values on E_T , using the coordinate unit vectors already in U_T . The result is the coordinate unit vector at j on T , so $j \in E_T$ by definition. Since $S_T = T \setminus E_T$, no such j exists. Hence $E_{S_T} = \emptyset$. $\square$

Remark 7. Removing the coordinates in E_T decreases the rank defect by exactly their number:

$$|E_T| = \delta_T - \delta_{S_T} \leq s - \delta_{S_T}.$$

It preserves the strict inequality between the number of equations and their rank. $\square$

The actual core S_G depends on the response through G . We now fix all systems that a core could generate, before any challenge or response is selected. Write $A_T = \text{span}_{\mathbb{F}}\{v_j : j \in T\}$.

Definition 6 (Closure of a column set). Let $T \subseteq [n]$ be consistent. Its closure is

$$\text{cl}(T) = \{j \in [n] : v_j \in A_T\}.$$

These are all columns whose primary equations are implied by the equations on T .

Closure preserves the solution set, consistency, and rank: $T \subseteq \text{cl}(T)$ and $A_{\text{cl}(T)} = A_T$. Taking the closure again adds no columns.

Lemma 10 (Closure preserves canonical cores). Let S be consistent, with $0 \leq \delta_S \leq s$, $|S| > R + \delta_S$, and $E_S = \emptyset$. Put $D = \text{cl}(S)$. Then

$$\delta_D = \delta_S, \quad E_D = \emptyset, \quad |D| > \text{rk}(D).$$

Proof. Closure preserves consistency and rank, so $\delta_D = \delta_S$ and $|D| \geq |S| > \text{rk}(S) = \text{rk}(D)$. Restriction $U_D \rightarrow U_S$ is surjective and its domain and image have the same dimension by (7.9); it is therefore injective. A coordinate unit vector at $j \in S$ in U_D would restrict to one in U_S , contrary to $E_S = \emptyset$. A coordinate unit vector at $j \in D \setminus S$ would restrict to zero, contrary to injectivity. Thus $E_D = \emptyset$. $\square$

Definition 7 (Fixed reduced systems). Consider every consistent $T \subseteq [n]$ with $|T| \geq c$ and $\delta_T \leq s$. Form its canonical core S_T using Lemma 9, which applies because $c = R + s + 1 > R + \delta_T$. Let $\mathcal{J}$ be the family of distinct closures $\text{cl}(S_T)$. The family is finite and may be empty. It is fixed from W and the public parameters, independently of the statement and all challenges.

By Lemmas 9 and 10, each $D \in \mathcal{J}$ satisfies

$$E_D = \emptyset, \quad R \leq \text{rk}(D) \leq R + s, \quad |D| > \text{rk}(D). \quad (7.11)$$

Lemma 11 (Closing a canonical core adds no exceptional columns). Let $T \subseteq [n]$ be consistent, with $|T| \geq c$ and $\delta_T \leq s$, and put $D = \text{cl}(S_T)$. Then

$$T \cap D = S_T, \quad T \setminus D = E_T,$$

and

$$|T \setminus D| = |E_T| = \delta_T - \delta_D \leq s - \delta_D. \quad (7.12)$$

Proof. By definition of closure, $S_T \subseteq D$. For $j \in E_T$, the coordinate unit vector at j belongs to U_T . Expressing it in the coefficient columns gives a change to (B, p) that leaves every equation on S_T unchanged but changes the equation at j by 1. Making this change to a solution on T gives a solution on S_T that violates the equation at j . Thus $j \notin \text{cl}(S_T)$. This proves $T \cap D = S_T$ and $T \setminus D = E_T$. Closure preserves rank, so $\delta_D = \delta_{S_T}$, and Lemma 9 proves (7.12). $\square$

To bound acceptance for every response assigned to D , we use the largest actual consistent set assigned to it.

Definition 8 (System query cap). For $D \in \mathcal{J}$, let

$$\mathcal{T}_D = \{T \subseteq [n] : T \text{ is consistent, } |T| \geq c, \delta_T \leq s, \text{cl}(S_T) = D\}.$$

Define

$$a_D = \max_{T \in \mathcal{T}_D} \binom{|T|}{t} / \binom{n}{t}. \quad (7.13)$$

The family $\mathcal{T}_D$ is finite and nonempty by Definition 7, so the maximum is attained by an existing set and $0 \leq a_D \leq 1$. If a good set G is consistent, has $|G| \geq c$ and $\delta_G \leq s$, and is assigned to D , then $G \in \mathcal{T}_D$ and $\binom{|G|}{t} / \binom{n}{t} \leq a_D$. The cap bounds each response's accepting-query fraction; it does not count the union of their accepted queries.

7.5 Large overlap determines the system

We now prove the structural fact needed for counting: actual consistent sets assigned to different systems have overlap at most $k + 2s - 1$. We first establish an auxiliary lemma about the ranks of two canonical cores and their union. This is the only step in the analysis that invokes Theorem 1.

Lemma 12 (The union of overlapping canonical cores is MDS). *Let $S, S' \subseteq [n]$ have ranks $r = \text{rk}(S)$ and $r' = \text{rk}(S')$, with*

$$E_S = E_{S'} = \emptyset, \quad R \leq r, r' \leq k + 2s, \quad |S| > r, \quad |S'| > r'.$$

If $|S \cap S'| \geq r + r' - (k + 2s)$, then

$$\text{rk}(S \cup S') = r = r', \quad (7.14)$$

and, for all $H \subseteq S \cup S'$,

$$\text{rk}(H) = \min\{|H|, r\}. \quad (7.15)$$

Proof. Pick an inclusion-maximal set M with

$$S \subseteq M \subseteq S \cup S', \quad \text{rk}(M) \leq k + 2s, \quad E_M = \emptyset.$$

Such a set exists: S satisfies these conditions by hypothesis, and there are only finitely many candidates. Also U_M is proper in $\mathbb{F}^M$. By definition of the spaces in (7.8), its restriction to S is U_S . Equation (7.9) gives $\dim U_S = r < |S|$ by hypothesis; a full space on M would restrict to the full space on S . By (7.8), U_M has the form required by Theorem 1. Its evaluation points are distinct by the protocol hypotheses, and (7.9) and the defining conditions for M give $\dim U_M = \text{rk}(M) \leq k + 2s$. Applying that theorem, and using $E_M = \emptyset$, gives

$$\text{rk}(H) = \min\{|H|, \text{rk}(M)\} \quad (H \subseteq M). \quad (7.16)$$

For $H = S$, this says $r = \min\{|S|, \text{rk}(M)\}$. Since $|S| > r$, it forces $\text{rk}(M) = r$. In particular, $\text{rk}(S \cap S') = \min\{|S \cap S'|, r\}$.

We now use rank submodularity,

$$\text{rk}(A \cup B) + \text{rk}(A \cap B) \leq \text{rk}(A) + \text{rk}(B),$$

with $A = S$ and $B = S'$. Together with (7.16), this gives

$$\begin{aligned} \text{rk}(S \cup S') + \text{rk}(S \cap S') &\leq r + r', \\ \text{rk}(S \cup S') &\leq r + r' - \min\{|S \cap S'|, r\} \leq k + 2s. \end{aligned}$$

For the last inequality, if $|S \cap S'| \leq r$, use the overlap hypothesis; otherwise the right-hand side is $r' \leq k + 2s$ by hypothesis.

We also need to show $E_{S \cup S'} = \emptyset$. A coordinate unit vector survives restriction to any set containing its coordinate. By (7.8), $U_{S \cup S'}|_S = U_S$ and $U_{S \cup S'}|_{S'} = U_{S'}$. Thus

$$E_{S \cup S'} \subseteq E_S \cup E_{S'} = \emptyset,$$

where the two sets on the right are empty by hypothesis. Together with the rank bound, this shows that $S \cup S'$ satisfies all the defining conditions for M . Since $M \subseteq S \cup S'$ by construction, inclusion-maximality gives $M = S \cup S'$. Applying (7.16) to S' , and using $|S'| > r'$, now gives

$$\text{rk}(S \cup S') = r = r'.$$

Since $M = S \cup S'$ and $\text{rk}(M) = r$, (7.16) also gives the claimed rank formula for every $H \subseteq S \cup S'$. $\square$

We now use Lemma 12 to show that consistent column sets with large overlap determine the same fixed system. The rank equalities identify the coefficient-row spans of the two cores. The common core equations then identify the affine-form spans, including the fixed mask terms, and hence the closures.

Lemma 13 (Large overlap determines the system). *Let $T, T' \subseteq [n]$ be consistent, with $0 \leq \delta_T, \delta_{T'} \leq s$, $|T| > R + \delta_T$, and $|T'| > R + \delta_{T'}$. If $|T \cap T'| \geq k + 2s$, then $\text{cl}(S_T) = \text{cl}(S_{T'})$.*

Proof. Put $S = S_T$, $S' = S_{T'}$, $E = E_T$, and $E' = E_{T'}$, and write $r = \text{rk}(S)$, $r' = \text{rk}(S')$. Recall that cl is defined using the affine primary equations, including the fixed mask terms, but rk is defined as the rank of the coefficient matrix without the mask column. For $H \subseteq [n]$, write A_H for this coefficient matrix. We first prove

$$\text{rk}(S \cap S') = \text{rk}(S) = \text{rk}(S') = \text{rk}(S \cup S'),$$

then use consistency to obtain $A_S = A_{S'}$, which implies equality of closures by Definition 6.

By hypothesis, Lemma 9 applies to T, T' . Using $R = k + s$ and the definition of rank defect gives

$$\begin{aligned} E_S = E_{S'} = \emptyset, \quad R \leq r, r' \leq k + 2s, \quad |S| > r, \quad |S'| > r', \\ r + |E| = \text{rk}(T) = R + \delta_T \leq k + 2s, \\ r' + |E'| = \text{rk}(T') = R + \delta_{T'} \leq k + 2s. \end{aligned} \tag{7.17}$$

Since $S = T \setminus E$ and $S' = T' \setminus E'$, the overlap hypothesis and (7.17) give

$$\begin{aligned} |S \cap S'| &\geq |T \cap T'| - |E| - |E'| \\ &\geq k + 2s - |E| - |E'| \geq r + r' - (k + 2s). \end{aligned} \tag{7.18}$$

Equations (7.17) and (7.18) verify the hypotheses of Lemma 12. Its conclusion (7.14) gives

$$\text{rk}(S) = \text{rk}(S') = \text{rk}(S \cup S') = r.$$

Selecting the rows indexed by S or S' from $A_{S \cup S'}$ gives

$$\text{rowspan}(A_S) \subseteq \text{rowspan}(A_{S \cup S'}) \supseteq \text{rowspan}(A_{S'}).$$

By equality of row and column rank,

$$\dim \text{rowspan}(A_S) = \dim \text{rowspan}(A_{S'}) = \dim \text{rowspan}(A_{S \cup S'}) = r.$$

Each core's row span is therefore a subspace of $\text{rowspan}(A_{S \cup S'})$ of the same dimension, so both inclusions are equalities:

$$\text{rowspan}(A_S) = \text{rowspan}(A_{S'}) = \text{rowspan}(A_{S \cup S'}).$$

It remains to show that $\text{rk}(S \cap S') = r$. Adjoining E gives

$$\begin{aligned} \text{rowspan}(A_{S \cup E}) &= \text{rowspan}(A_S) + \text{rowspan}(A_E) \\ &= \text{rowspan}(A_{S'}) + \text{rowspan}(A_E) = \text{rowspan}(A_{S' \cup E}). \end{aligned}$$

Using (7.17), rank subadditivity, and $\text{rk}(E \setminus S') \leq |E \setminus S'|$,

$$\begin{aligned} r + |E| &= \text{rk}(S \cup E) = \text{rk}(S' \cup E) \\ &= \text{rk}(S' \cup (E \setminus S')) \\ &\leq \text{rk}(S') + \text{rk}(E \setminus S') \\ &\leq r + |E \setminus S'| = r + |E| - |E \cap S'|. \end{aligned}$$

Thus $|E \cap S'| = 0$, hence $E \cap S' = \emptyset$; symmetrically, $E' \cap S = \emptyset$. Since $T = S \cup E$ and $T' = S' \cup E'$,

$$T \cap T' = (S \cap S') \cup (E \cap E').$$

By the overlap hypothesis and (7.17),

$$|S \cap S'| \geq |T \cap T'| - |E \cap E'| \geq k + 2s - |E \cap E'| \geq k + 2s - |E| \geq r.$$

Equation (7.15), with $H = S \cap S'$, gives

$$\text{rk}(S \cap S') = \min\{|S \cap S'|, r\} = r = \text{rk}(S) = \text{rk}(S') = \text{rk}(S \cup S').$$

We now extend to the affine case. By hypothesis, T, T' are consistent, hence so are $S, S', S \cap S'$. Appending the mask column preserves rank for a consistent system, as explained after (7.9). Therefore

$$\Lambda_S \supseteq \Lambda_{S \cap S'} \subseteq \Lambda_{S'}, \quad \dim \Lambda_S = \dim \Lambda_{S \cap S'} = \dim \Lambda_{S'} = r.$$

Equality of dimensions and Definition 6 give

$$\Lambda_S = \Lambda_{S \cap S'} = \Lambda_{S'} \implies \text{cl}(S) = \text{cl}(S').$$

□

7.6 System exceptions

Section 7.4 fixes the possible systems. We now identify when a system can prevent extraction: decoding may fail, or the decoded witness may violate the statement. For a sufficiently large good set G outside the fixed rank exception, we distinguish these possibilities through its system's rank defect. With positive rank defect, the proximity equations can be satisfied only through accidental cancellations in the random polynomial challenge, which occur with probability at most $1/q$. We charge this case without needing to establish decoding. With zero rank defect, the system equations can be solved for every polynomial challenge, and their solutions determine fixed low-degree row polynomials. Section 7.7 will show that the many good columns guarantee both recovery of the witness-row polynomials by the decoder and satisfaction of the multiplication constraints. A violated linear constraint can then escape detection only through a nontrivial arithmetic cancellation, also of probability at most $1/q$. We define these two rare events as system exceptions and prove that each fixed system is exceptional with probability at most $1/q$.

We first handle positive rank defect. Let $D \in \mathcal{J}$ satisfy $\delta_D > 0$. Lemma 7 gives

$$\Pr_{B \in \mathcal{B}} [\exists p \in \mathcal{P}_{<R} : p \text{ satisfies the equations on } D] = q^{-\delta_D} \leq \frac{1}{q}. \quad (7.19)$$

If the actual good set has canonical core S_G with closure D , its proximity response satisfies the equations on D . Thus this fixed event covers every such response, whether or not decoding succeeds.

We now turn to zero rank defect. The system equations can be solved for every polynomial challenge, so consistency is no longer rare. Solving the system gives row polynomials of degree less than $R = k + s$. The next lemma strengthens this bound to degree less than k for every row except the proximity mask.

Lemma 14 (Zero rank defect gives row polynomials). *Let S be consistent, with $|S| > R$ and $\delta_S = 0$. Then there are unique polynomials*

$$f_{S,1}, \dots, f_{S,m} \in \mathcal{P}_{<k}, \quad f_{S,\star} \in \mathcal{P}_{<R},$$

agreeing with the corresponding rows of W on S . For all $B \in \mathcal{B}$ and $p \in \mathcal{P}_{<R}$, the proximity equations on S hold if and only if

$$p = P_S(B) := f_{S,\star} + \sum_{r=1}^m B_r f_{S,r}. \quad (7.20)$$

Proof. The response exists for every polynomial challenge, by Lemma 7. We use selected challenges to expose each row and its polynomial multiples. At $B = 0$, the unique response gives $f_{S,\star} \in \mathcal{P}_{<R}$. For each row r and $a \in \{0, \dots, s\}$, take $B_r = X^a$ and all other challenges zero. Subtracting $f_{S,\star}$ from the response gives $h_{r,a} \in \mathcal{P}_{<R}$ with

$$h_{r,a}(x_j) = x_j^a W_{r,j} \quad (j \in S).$$

The successive responses must describe successive multiples of the same row. For $a < s$, the polynomial $Xh_{r,a} - h_{r,a+1}$ has degree at most R and vanishes at more than R points. Thus $h_{r,a+1} = Xh_{r,a}$. Set $f_{S,r} = h_{r,0}$. Because $X^s f_{S,r} = h_{r,s}$ has degree less than $R = k + s$, we have $\deg f_{S,r} < k$. This also covers $s = 0$.

These row polynomials now determine every proximity response. The polynomial in (7.20) has degree less than R and agrees with Y_B on S . Uniqueness of the response proves the identity. Uniqueness of all the row polynomials follows from the root bound. $\square$

The row polynomials give a candidate witness, but even successful decoding does not ensure that it satisfies the statement. If the decoded witness violates a linear constraint, the verifier's randomized linear checks may nevertheless pass through accidental cancellation. The extractor checks the original constraints directly, so it rejects that witness and returns $\perp$. Decoding succeeds, but extraction fails; we must count these accepting executions in the error bound.

We now bound the probability of this cancellation for each fixed system. For each linear constraint in (4.1), its *residual* is the left-hand side minus the required value. The constraint holds exactly when this residual is zero.

Definition 9 (Linear residuals and canonical linear response). Let $D \in \mathcal{J}$ have $\delta_D = 0$. Write its row polynomials from Lemma 14 as $f_{D,r}, f_{D,*}$, and put $g_D = (f_{D,1}, \dots, f_{D,d})$. Write $\sigma = (A, b)$, and let $w_D = (f_{D,r}(\xi_a))_{r \in [d], a \in \mathcal{W}}$ be the logical values of these row polynomials. Define the linear residuals and the linear-mask offset by

$$\begin{aligned} r_D &= (r_{D,e})_{e \in I} = Aw_D - b, \\ z_D &= \sum_{a \in \mathcal{W}} f_{D,d+2}(\xi_a). \end{aligned} \tag{7.21}$$

An honest prover using these fixed row polynomials would send the canonical linear response:

$$Z_{D,\sigma}(\alpha) = \sum_{r=1}^d C_{\sigma,\alpha,r} f_{D,r} + v f_{D,d+1} + z_R f_{D,d+2} \in \mathcal{P}_{<2k}. \tag{7.22}$$

The residuals and mask offset are fixed before the challenge. We do not assume that the malicious mask has zero sum, so z_D may be nonzero. The next lemma shows that this offset cannot conceal a nonzero residual vector except through a rare cancellation.

Lemma 15 (A violated linear constraint requires arithmetic cancellation). Let $D \in \mathcal{J}$ have $\delta_D = 0$. For all linear challenges α ,

$$\sum_{a \in \mathcal{W}} Z_{D,\sigma}(\alpha)(\xi_a) - \alpha^\top b = \sum_{e \in I} \alpha_e r_{D,e} + z_D. \tag{7.23}$$

If $(r_{D,e})_{e \in I} \neq 0$, then for uniform α ,

$$\Pr_{\alpha} \left[\sum_{e \in I} \alpha_e r_{D,e} + z_D = 0 \right] \leq \frac{1}{q}. \tag{7.24}$$

Equivalently, if the residual vector is nonzero, the canonical response passes the witness-sum check (4.14) with probability at most $1/q$.

Proof. Evaluate $Z_{D,\sigma}(\alpha)$ at the logical positions and use (4.5). The first linear mask vanishes there; the second contributes z_D . Summing the remaining terms gives (7.23).

If the residual vector is nonzero, pick e with $r_{D,e} \neq 0$. For all values of the other challenge coordinates, at most one value of α_e makes the sum zero. Since α_e is uniform and independent of those coordinates, the probability is at most $1/q$. $\square$

The zero-defect probability bound applies to the canonical linear response, whereas the verifier uses the actual response. Section 7.7 will show that the two responses coincide when the good set is sufficiently large. We now combine the positive- and zero-defect events.

Definition 10 (System exceptions). Let $D \in \mathcal{J}$ and let $\gamma = (\alpha, B) \in \Gamma$ be the verifier challenge. The system D is exceptional for γ when the applicable condition holds:

$$\begin{cases} \exists p \in \mathcal{P}_{<R} : p \text{ satisfies the equations on } D \text{ for } B, & \delta_D > 0, \\ (r_{D,e})_{e \in I} \neq 0 \wedge \sum_{e \in I} \alpha_e r_{D,e} + z_D = 0, & \delta_D = 0. \end{cases}$$

Write $\mathcal{J}(\gamma) = \{D \in \mathcal{J} : D \text{ is exceptional for } \gamma\}$.

Lemma 16 (System exceptions are rare). Let $D \in \mathcal{J}$. Then

$$\Pr_{\gamma \in \Gamma}[D \in \mathcal{J}(\gamma)] \leq \frac{1}{q}. \quad (7.25)$$

Proof. For $\delta_D > 0$, apply (7.19). For $\delta_D = 0$, the event is empty if the residual vector is zero; otherwise apply Lemma 15. $\square$

The counting argument uses only these individual probability bounds and does not require independence between system exceptions.

7.7 Extraction

Assume that the good set is sufficiently large and neither the rank exception of Section 7.3 nor the associated system exception of Section 7.6 occurs. The extractor is defined in Figure 3. We prove that, under these hypotheses, its row decoders and final checks succeed. We first show that the witness-row polynomials can be uniquely decoded, with at least $h - s$ common agreement columns, and that the actual linear response equals the canonical response. To prove extraction, it remains to show that the decoded witness satisfies the statement. The many good columns force the multiplication constraints. The global linear check, together with the absence of a system exception, forces the linear constraints.

We first establish that the fixed witness-row polynomials can be decoded and have enough common agreement columns. This step uses their agreement with the oracle on the fixed system, without assuming that the witness satisfies the statement.

Lemma 17 (Unique decoding of the fixed witness rows). Let $D \in \mathcal{J}$ have $\delta_D = 0$, and let $G \subseteq [n]$. If $|G| \geq h$ and $|G \setminus D| \leq s$, then the row decoders on G , with degree bound k and radius s , return g_D . This tuple agrees with the oracle on $H = G \cap D$, where

$$|H| \geq |G| - s \geq h - s = \max\{2k, k + s + 1\}. \quad (7.26)$$

Proof. Each witness row agrees with its fixed polynomial throughout D , so it has at most s errors on G . Since

$$|G| \geq h \geq c = k + 2s + 1,$$

each fixed witness-row polynomial is the unique message within radius s . Definition 3 therefore guarantees that the decoder returns it. Also, $|G \cap D| = |G| - |G \setminus D| \geq |G| - s$. The definition of h proves (7.26). $\square$

The verifier checks the actual linear response, while the system-exception bound concerns the canonical response. Agreement on sufficiently many good columns identifies these two polynomials.

Lemma 18 (Equality of the actual and canonical linear responses). *Let $D \in \mathcal{J}$ have $\delta_D = 0$. Let $\gamma = (\alpha, B)$ be a verifier challenge, let (p, Z) be a response pair of the prescribed degrees, and put $G = G_W(\sigma, \gamma, p, Z)$. If $|G \cap D| \geq 2k$, then*

$$Z = Z_{D, \sigma}(\alpha).$$

Proof. At each column in $G \cap D$, the linear column check passes and all challenged row polynomials agree with the oracle, including the two linear masks. Thus Z and $Z_{D, \sigma}(\alpha)$ agree at these columns. Both have degree less than $2k$, so agreement at at least $2k$ distinct points makes them equal. $\square$

It remains to prove that the decoded witness satisfies the statement. The same agreement set will force the multiplication identities. Equality of the linear responses will then let us apply the system-exception condition to the actual global check.

Recall that $\text{acc}_W(\sigma, \gamma, p, Z)$ is the fraction of queries accepted for the fixed oracle, statement, challenge, and response pair. When the degree and global checks pass, this fraction is $\binom{|G|}{t} / \binom{n}{t}$; otherwise it is zero (Lemma 6). Only the query is random in this quantity.

Lemma 19 (A large good set yields extraction or a system exception). *Let $\gamma = (\alpha, B)$ be a verifier challenge outside the rank exception, so $B \notin \mathcal{E}_W$, and let (p, Z) be a response pair passing the degree and global checks. If $|G| \geq h$, then there exists a system $D \in \mathcal{J}$ such that*

$$\text{acc}_W(\sigma, \gamma, p, Z) \leq a_D, \tag{7.27}$$

and either $D \in \mathcal{J}(\gamma)$, or $\delta_D = 0$ and Ext returns the fixed tuple g_D .

Proof. First locate the fixed system explaining the good columns. The actual pair (B, p) makes the equations on G consistent. Lemma 8 gives $\delta_G \leq s$. Let D be the closure of its canonical core S_G . Since $|G| \geq h \geq c$, Definition 7 gives $D \in \mathcal{J}$. Since $G \in \mathcal{T}_D$, Definition 8 and Lemma 6 give $\text{acc}_W(\sigma, \gamma, p, Z) \leq a_D$. The actual pair satisfies all equations on S_G , hence all equations on D by definition of closure. Lemma 11, with $T = G$, gives

$$|G \setminus D| = \delta_G - \delta_D \leq s - \delta_D \leq s. \tag{7.28}$$

We now prove the disjunction. If a system exception occurs for D at the actual challenge γ , then $D \in \mathcal{J}(\gamma)$, which is the first alternative.

Suppose instead that $D \notin \mathcal{J}(\gamma)$: no system exception occurs for D at the actual challenge γ . Since the proximity equations on D hold, Definition 10 forces $\delta_D = 0$. By (7.28), we have $|G \setminus D| \leq s$. Lemma 17 therefore shows that the row

decoders return g_D and $|G \cap D| \geq 2k$, so Lemma 18 identifies the actual linear response with $Z_{D,\sigma}(\alpha)$.

For each multiplication constraint, the polynomial

$$f_{D,i_1(g)}f_{D,i_2(g)} - f_{D,i_3(g)} \quad (g \in \mathcal{M})$$

has degree less than $2k$ and vanishes on $G \cap D$, because the multiplication check passes there and the row polynomials have the oracle values. It is therefore the zero polynomial. Evaluating these identities at the logical positions proves all multiplication constraints.

The global check, equality of the linear responses, and (7.23) give

$$\sum_{e \in I} \alpha_e r_{D,e} + z_D = 0.$$

Since $D \notin \mathcal{J}(\gamma)$, this equality forces every linear residual to be zero. Thus w_{g_D} satisfies the statement.

Finally, $D \subseteq D_{g_D}(W)$ gives $|G \setminus D_{g_D}(W)| \leq |G \setminus D| \leq s$, so the joint-error check passes. Thus Ext returns g_D , with the output guarantees of Section 7.1. $\square$

This concludes the extraction argument and the structural part of the proof. The remainder of this section assembles the final bound.

7.8 Bounding the acceptance envelope

Outside the rank exception, a response with many good columns either extracts the fixed tuple of its assigned system or incurs a system exception. In either case its query fraction is bounded by a_D . Since the prover selects its response after seeing the challenge, we must bound the average of the maximum a_D over systems whose exception occurs or whose fixed tuple is unwanted. Recall that $\mathcal{J}$ is the fixed family of systems from Definition 7, and a_D is the query cap for system D defined in (7.13).

The difficulty is that the prover selects its response after seeing the challenge, so the system with the largest query cap can change from one challenge to another. A bound for each fixed system therefore does not directly bound the average maximum. A union bound over systems is too loose: it adds their individual bounds without accounting for the fact that their query caps cannot all be large. We develop a bound that uses the small intersections between actual compatible sets to limit the total excess of their query caps above the threshold. This lets us bound the average maximum without paying separately for every system.

The actual sets attaining the system caps supply the second-moment argument: we count single and pairwise query incidences, using the linear binomial bound of Lemma A.1. Lemma A.2 turns that count into a bound on the total excess of any at most q query caps above θ .

This excess bound is the bridge to the maximum. Caps at or below θ cost only the baseline θ ; the larger caps have limited total excess. Lemma 20 uses

this restriction and the count for each contributing system to bound the sum of maxima, without a factor for the number of systems.

We first establish the excess bound. Write $x_+ = \max\{0, x\}$ for the positive part of x .

For each $D \in \mathcal{J}$, pick an actual set $T_D \in \mathcal{T}_D$ attaining the cap in (7.13). Such a set exists because $\mathcal{T}_D$ is finite and nonempty. Then

$$a_D = \binom{|T_D|}{t} \bigg/ \binom{n}{t}, \quad |T_D \cap T_{D'}| \leq k + 2s - 1 \quad (D \neq D'),$$

where the intersection bound follows from the contrapositive of Lemma 13. Apply Lemma A.2 to these existing sets, with $\ell = k + 2s - 1$ and $U = |\mathcal{Q}|\theta$. Its counts are $M_D = |\mathcal{Q}|a_D$. The definition of θ in (6.5) gives its threshold inequality, and Definition 2 implies $1 \leq u \leq q - 1$ and $0 \leq \theta < 1$, as shown in Section 6.1. Dividing the conclusion by $|\mathcal{Q}|$ gives, for all $\mathcal{I} \subseteq \mathcal{J}$ with $|\mathcal{I}| \leq q$,

$$\sum_{D \in \mathcal{I}} (a_D - \theta)_+ \leq 1 - \theta. \quad (7.29)$$

The next lemma bounds a sum of maxima using only the number of times each entry is available. It keeps the size of the exceptional set exact; this will preserve the factor multiplying the fraction of indices whose unwanted sets contain a given tuple. No restrictions on intersections among the availability sets are needed.

Lemma 20 (Sum of maxima outside an exceptional set). *Let $(a_i)_{i \in J}$ be a finite family in $[0, 1]$, let $q \geq 2$ be an integer, and let $0 \leq \theta \leq 1$. Assume $\sum_{i \in I} (a_i - \theta)_+ \leq 1 - \theta$ for all $I \subseteq J$ with $|I| \leq q$. Let X be a finite set, let $\mathcal{B}, A_i \subseteq X$, and let $L \geq 0$ satisfy*

$$|A_i \setminus \mathcal{B}| \leq L \quad (i \in J), \quad qL \geq |X \setminus \mathcal{B}|.$$

Define $H(x) = 1$ for $x \in \mathcal{B}$, and $H(x) = \max(\{\theta\} \cup \{a_i : x \in A_i\})$ otherwise. Then

$$\sum_{x \in X} H(x) \leq |\mathcal{B}| + |X \setminus \mathcal{B}|\theta + L(1 - \theta). \quad (7.30)$$

Proof. Put $e_i = (a_i - \theta)_+$. For each $x \notin \mathcal{B}$ with $H(x) > \theta$, assign its excess to one index attaining the maximum, breaking ties by a fixed ordering. Let w_i count the assignments to index i . An assignment to i requires $x \in A_i \setminus \mathcal{B}$, so

$$0 \leq w_i \leq L, \quad \sum_i w_i \leq |X \setminus \mathcal{B}| \leq qL.$$

We claim that these weights contribute at most L times the sum of the q largest excesses. If $|J| \leq q$, this follows directly from $w_i \leq L$. Otherwise, let T contain the q largest excesses. The total weight outside T is at most the unused capacity inside it:

$$\sum_{i \notin T} w_i \leq \sum_{i \in T} (L - w_i).$$

Every excess outside T is at most every excess inside T . Hence

$$\sum_{i \notin T} w_i e_i \leq \left(\min_{i \in T} e_i \right) \sum_{i \notin T} w_i \leq \sum_{i \in T} (L - w_i) e_i.$$

Adding the contribution inside T proves the claim. In both cases, the excess budget therefore gives $\sum_i w_i e_i \leq L(1 - \theta)$. Finally,

$$\sum_{x \in X} H(x) = |\mathcal{B}| + |X \setminus \mathcal{B}| \theta + \sum_i w_i e_i \leq |\mathcal{B}| + |X \setminus \mathcal{B}| \theta + L(1 - \theta).$$

□

For example, take $X = \Gamma$, $\mathcal{B} = \emptyset$, and let A_D contain the challenges causing the system exception for D . Lemma 16 gives $|A_D| \leq |\Gamma|/q$. Apply the lemma with $L = |\Gamma|/q$ and divide by $|\Gamma|$ to obtain

$$\frac{1}{|\Gamma|} \sum_{\gamma \in \Gamma} \max(\{\theta\} \cup \{a_D : D \in \mathcal{J}(\gamma)\}) \leq \theta + \frac{1 - \theta}{q}. \quad (7.31)$$

A single fraction equal to one, available on exactly $|\Gamma|/q$ challenges, attains this abstract bound. The overlap budget prevents multiple systems from giving a larger worst-case value.

7.9 Complete the per-level bound

We now prove property (iii) of Theorem 4. Every index uses the same oracle, so its systems, query caps, and zero-defect witness tuples are the same throughout the family. The statements and unwanted sets $\mathcal{U}_i$ may vary with the index, as may the system exceptions. The rank event is the same for every index. For each fixed witness tuple, we count the indices whose sets contain it. In the recursive application, this counts the parent queries for which an unwanted child tuple certifies acceptance.

Recall the threshold θ and error ε from (6.5)–(6.2). As shown after Definition 2, we have $1 \leq u \leq q - 1 < q$ and $\delta \leq \theta < 1$. The query caps satisfy the excess bound (7.29).

Proof of Theorem 4. We construct the functions H_i required by Theorem 4. For each zero-defect system D , use its fixed tuple g_D from Definition 9. Let $A_D(i, \gamma)$ mean that

$$\text{the system exception for } D \text{ occurs at } (\sigma_i, \gamma) \quad \vee \quad (\delta_D = 0 \wedge g_D \in \mathcal{U}_i).$$

Put

$$H_i(\alpha, B) = \begin{cases} 1, & B \in \mathcal{E}_W, \\ \max(\{\theta\} \cup \{a_D : A_D(i, (\alpha, B))\}), & B \notin \mathcal{E}_W. \end{cases} \quad (7.32)$$

The envelopes lie in $[0, 1]$ and do not depend on the actual response. Fix an index i , a challenge γ , and a response for which extraction fails or returns a tuple in $\mathcal{U}_i$. If a degree or global check fails, the accepted-query fraction is zero. On the rank event it is at most one. Outside the rank event, a small good set gives a fraction at most $\delta \leq \theta$. Otherwise Lemma 19 assigns a system D with acceptance at most a_D . Either its exception occurs, or extraction returns its fixed tuple g_D . In the latter case g_D must belong to $\mathcal{U}_i$. Thus A_D holds and acceptance is at most $H_i(\gamma)$. This proves the pointwise bound for all responses.

It remains to average. Write $\alpha_0 = \Pr[B \in \mathcal{E}_W] \leq \mu$. For a fixed system D , put

$$\rho_D = \begin{cases} |\{i \in \mathcal{I} : g_D \in \mathcal{U}_i\}|/|\mathcal{I}|, & \delta_D = 0, \\ 0, & \delta_D > 0. \end{cases}$$

In both cases $\rho_D \leq \rho$. For positive defect, A_D is just the system exception, whose probability is at most $1/q$. For zero defect, a fraction ρ_D of the indices satisfy $g_D \in \mathcal{U}_i$. For these indices, the event $B \notin \mathcal{E}_W \wedge A_D$ has probability $1 - \alpha_0$. For the other indices, it requires the system exception and has probability at most $1/q$. Hence

$$\begin{aligned} \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \Pr_{\gamma} [B \notin \mathcal{E}_W \wedge A_D(i, \gamma)] &\leq (1 - \alpha_0)\rho_D + (1 - \rho_D)/q \\ &\leq (1 - \alpha_0)\rho + (1 - \rho)/q =: \lambda. \end{aligned}$$

The first inequality therefore holds for both defects. The last inequality uses $\alpha_0 + 1/q \leq \mu + 1/q < 1$. Also $q\lambda \geq 1 - \alpha_0$ and $\lambda \geq 0$. Apply Lemma 20 on $X = \mathcal{I} \times \Gamma$, using the excess budget (7.29). Let $\mathcal{B}$ contain the pairs whose challenge lies in the rank event. Then $|\mathcal{B}| = \alpha_0|X|$, and the preceding bound gives $|A_D \setminus \mathcal{B}| \leq \lambda|X|$. Take $L = \lambda|X|$; the condition $qL \geq |X \setminus \mathcal{B}|$ follows from $q\lambda \geq 1 - \alpha_0$. Dividing the conclusion by $|X|$ gives

$$\begin{aligned} \frac{\sum_{i \in \mathcal{I}} \sum_{\gamma \in \Gamma} H_i(\gamma)}{|\mathcal{I}| |\Gamma|} &\leq \alpha_0 + (1 - \alpha_0)\theta + ((1 - \alpha_0)\rho + (1 - \rho)/q)(1 - \theta) \\ &= \rho + (1 - \rho)(\theta + (1 - \theta)(\alpha_0 + 1/q)) \\ &\leq \rho + (1 - \rho)\varepsilon = \varepsilon + (1 - \varepsilon)\rho. \end{aligned}$$

We used that the rank event is the same for every index; no independence between rank and system exceptions is required. $\square$

This completes the per-level bound. The recursive proof in Section 6.3 applies it to the query frequency of each fixed child tuple. No separate counting bound on child tuples is required.

8 Numerical instantiations

We minimize prover-to-verifier communication for workloads with N_{lin} linear witness entries and $N_{\text{lin}}/128$ multiplication gates. Gate inputs and outputs are

additional to the linear entries. We use binary units, a field of size 2^{128} , and

$$N_{\text{lin}} \in \{16\text{K}, 64\text{K}, 256\text{K}, 1\text{M}, 4\text{M}, 16\text{M}\}.$$

The target soundness error is $2^{-\lambda}$, with the values of λ listed in the tables.

We optimize two models separately. The *ZK-root model* uses the root masking construction of Section 4.4: $r_{\text{enc}} = t_0$ random padding positions, two arithmetic-mask rows, an independent primary-mask row, and independent 32-byte Merkle-leaf salts. Its root degree bound k_0 is even, with $k_0/2 - t_0 > 0$ witness slots per row. Fresh encoding randomness, masks, and salts are sampled for each proof. All children omit random padding, masks, and salts. Section 5 proves privacy for the oracle view. The *non-ZK model* omits them at every level; its root linear rows hold k_0 entries and multiplication-input rows hold at most $\lfloor k_0/2 \rfloor$. Omitted masks are implicit zero rows, so the ordinary soundness theorem still applies.

Repacking preserves the relation by assigning the original entries to fixed slots and giving unused slots zero linear coefficients. A satisfying witness extends by interpolation, with output polynomials equal to the products of their input polynomials; the degree budgets above ensure an honest encoding. In both models, children pack the parent’s $3k_{j-1} + s_{j-1}$ response coefficients into rows of width k_j , with no further hiding overhead.

For each model, we compare *recursive* executions with a *single invocation* that sends the response polynomials explicitly. The search varies row widths, domains, queries, challenge degrees, and the number of invocations, up to eight, and it requires $k_j/n_j \geq 1/16$ at every level. Root padding and all transmitted salts are included in the optimization objective. The accompanying export also certifies schedules with no rate floor. They are 33–44% smaller than the corresponding entries of Table 1, but their code lengths reach 5×10^8 , so we do not tabulate them.

Proof sizes include 16-byte field entries, 32-byte commitment roots and authentication hashes, shared-path Merkle multiproofs, opened root salts where applicable, and terminal response coefficients. Verifier messages and transport framing are excluded.

The tables use the $k + 2s - 1$ overlap bound proved in Section 7.5.

Exact integer comparisons determine u and hence the counting threshold in (6.5), with this overlap size. We evaluate the per-level error in (6.2) and the product bound in (6.8). The challenge degree s also serves as the rank allowance, and the same per-level conditions apply at every level.

Every reported schedule has an exact rational Lean certificate for all structural conditions, the per-level error inequalities, communication accounting, and the complete product-error target. The certificates are connected to both the concrete wire verifier and the full-oracle extractor. The grid and its coverage are checked in Lean, and the tables are generated from the Lean export.

These are best-found certified upper bounds, not proved global optima. The soundness targets concern the interactive oracle protocol; cryptographic binding and hiding errors are outside the targets. Complete schedules and reproduction instructions accompany the source.

Table 1. Prover-to-verifier proof size in bytes over a field of size 2^{128} , with rate at least $1/16$ at every level. Columns count linear witness entries; each workload has an additional $N_{\text{lin}}/128$ multiplication gates. Recursive entries list the number of invocations in parentheses. ZK rows use the masked root encoder of Section 4.4; non-ZK rows omit padding, masks, and salts. Every entry is exported from an exact Lean certificate.

λ	protocol	root	16K	64K	256K	1M	4M	16M
104	recursive	ZK	82 272 (2)	111 392 (2)	143 120 (3)	173 760 (4)	214 320 (4)	256 560 (4)
		non-ZK	63 872 (2)	90 272 (2)	119 472 (3)	151 920 (3)	182 560 (4)	223 120 (4)
	single	ZK	100 944	176 784	329 552	630 960	1 234 640	2 460 704
		non-ZK	66 880	121 872	231 296	448 400	880 912	1 742 528
112	recursive	ZK	86 592 (2)	118 304 (3)	150 592 (3)	185 024 (4)	227 760 (4)	274 624 (4)
		non-ZK	67 456 (2)	96 544 (2)	126 304 (3)	160 944 (3)	194 704 (4)	237 200 (4)
	single	ZK	105 424	184 240	342 640	655 168	1 279 584	2 527 776
		non-ZK	70 544	126 928	240 944	466 624	913 952	1 807 136
120	recursive	ZK	92 000 (2)	124 320 (3)	159 424 (3)	195 520 (4)	240 928 (4)	282 288 (5)
		non-ZK	71 040 (2)	101 296 (2)	135 200 (3)	169 504 (3)	207 056 (4)	252 304 (4)
	single	ZK	110 000	191 312	355 344	678 624	1 334 096	2 613 536
		non-ZK	72 896	132 160	250 272	484 064	945 328	1 868 656
126	recursive	ZK	95 616 (2)	131 328 (2)	166 000 (3)	211 872 (3)	277 472 (3)	357 856 (3)
		non-ZK	74 192 (2)	105 248 (2)	144 912 (3)	178 512 (3)	232 240 (3)	303 568 (3)
	single	ZK	113 040	196 544	364 608	701 248	1 366 656	2 698 288
		non-ZK	75 152	136 752	257 520	496 368	976 400	1 929 536
127	recursive	ZK	114 208 (1)	179 776 (2)	249 936 (2)	359 312 (2)	531 968 (2)	802 864 (2)
		non-ZK	75 872 (1)	136 976 (1)	198 864 (2)	291 584 (2)	427 568 (2)	643 744 (2)
	single	ZK	114 208	198 160	367 696	701 296	1 366 752	2 698 448
		non-ZK	75 872	136 976	259 632	500 336	976 464	1 929 680

The Lean development also proves equality of the root view distributions for a bounded batch of columns selected after both responses. The efficient sequential simulator and recursive privacy argument in Section 5 are proved in the paper; these extensions are not currently formalized in Lean.

Table 1 reports the certified proof sizes. Within each target λ , the first two rows are recursive executions, with the number of invocations in parentheses, and the last two are single invocations; within each pair, the first row is the ZK-root model and the second the non-ZK model. Compare entries with the same workload, target, and model.

References

- ACFY24. Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: Reed–Solomon proximity testing with super-fast verification. *Cryptology ePrint Archive*, Paper 2024/1586, 2024.
- AHIV17. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Liger: Lightweight sublinear arguments without a trusted setup. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2087–2104, 2017.
- Bru70. Pavol Brunovský. A classification of linear controllable systems. *Kybernetika*, 6(3):173–188, 1970.

- BRW26. Benedikt Bünz, Ron Rothblum, and William Wang. Flock: Fast proving for batch boolean computations. Cryptology ePrint Archive, Paper 2026/1329, 2026.
- BSBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast Reed–Solomon interactive oracle proofs of proximity. In *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*, volume 107 of *LIPICs*, pages 14:1–14:17, 2018.
- BSCG⁺14. Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from Bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474, 2014.
- BSCH⁺25. Eli Ben-Sasson, Dan Carmon, Ulrich Haböck, Swastik Kopparty, and Shubhangi Saraf. On proximity gaps for Reed–Solomon codes. Technical Report TR25-169, Electronic Colloquium on Computational Complexity, 2025.
- BSCI⁺20. Eli Ben-Sasson, Dan Carmon, Yuval Ishai, Swastik Kopparty, and Shubhangi Saraf. Proximity gaps for Reed–Solomon codes. In *61st IEEE Annual Symposium on Foundations of Computer Science (FOCS 2020)*, pages 900–909, 2020. Full version: ECC Report TR20-083, revision 3, 2021.
- BSCS16. Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In *Theory of Cryptography (TCC 2016-B)*, volume 9986 of *Lecture Notes in Computer Science*, pages 31–60, 2016.
- BSZ17. Christine Bachoc, Oriol Serra, and Gilles Zémor. An analogue of Vosper’s theorem for extension fields. *Mathematical Proceedings of the Cambridge Philosophical Society*, 163(3):423–452, 2017. Numbering follows arXiv:1501.00602v2.
- BSZ18. Christine Bachoc, Oriol Serra, and Gilles Zémor. Revisiting Kneser’s theorem for field extensions. *Combinatorica*, 38(4):759–777, 2018. Numbering follows arXiv:1510.01354v2.
- Fs24. Matteo Frigo and abhi shelat. Anonymous credentials from ECDSA. Cryptology ePrint Archive, Paper 2024/2010, 2024.
- Kil92. Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing (STOC 1992)*, pages 723–732, 1992.
- Mic00. Silvio Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000.
- NA25. Andrija Novakovic and Guillermo Angeris. Ligerito: A small and concretely fast polynomial commitment scheme. Cryptology ePrint Archive, Paper 2025/1187, 2025.
- RRR16. Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC 2016)*, pages 49–62, 2016.
- WB86. Lloyd R. Welch and Elwyn R. Berlekamp. Error correction for algebraic block codes. U.S. Patent 4,633,470, 1986.

A Direct binomial counting

The following counting inequality controls the system query caps. It counts query incidences directly in the given sets.

Lemma A.1 (Linear binomial bound). *Let $D_i \subseteq A$, for i in a finite index set I , and let t be a nonnegative integer. Write $n = |A|$, and put*

$$N = \binom{n}{t}, \quad M_i = \binom{|D_i|}{t},$$

$$P_{ij} = \binom{|D_i \cap D_j|}{t}.$$

Then for all nonnegative integers u ,

$$u \sum_i M_i \leq \binom{u+1}{2} N + \sum_{\{i,j\} \in \binom{I}{2}} P_{ij}. \quad (\text{A.1})$$

In particular, if $|D_i \cap D_j| \leq \ell$ for distinct indices and $m = |I|$, then

$$2u \sum_i M_i \leq u(u+1)N + m(m-1) \binom{\ell}{t}.$$

Proof. For each t -element query $Q \subseteq A$, let $\nu_Q = |\{i : Q \subseteq D_i\}|$. For nonnegative integers u, ν ,

$$u\nu \leq \binom{u+1}{2} + \binom{\nu}{2},$$

because the difference, doubled, is $(\nu - u)(\nu - u - 1) \geq 0$. Sum this inequality over all queries. Counting incidences by sets gives $\sum_Q \nu_Q = \sum_i M_i$; counting shared incidences by pairs gives $\sum_Q \binom{\nu_Q}{2} = \sum_{\{i,j\}} \binom{|D_i \cap D_j|}{t}$. Consequently,

$$u \sum_i M_i \leq \binom{u+1}{2} N + \sum_{\{i,j\}} P_{ij}.$$

The final claim follows by bounding each P_{ij} by $\binom{\ell}{t}$. □

Lemma A.2 (Binomial excess bound). *Let $(D_i)_{i \in J}$ be a finite family of subsets of a set A with $|A| = n$. Let t, ℓ be nonnegative integers, and assume*

$$|D_i \cap D_j| \leq \ell \quad (i \neq j).$$

Put

$$N = \binom{n}{t} > 0, \quad M_i = \binom{|D_i|}{t}.$$

Let $q \geq 2$ be an integer and $0 \leq U \leq N$. Assume that some integer $1 \leq v \leq q$ satisfies

$$\frac{(v-1)N}{2(q-1)} + \frac{q\binom{\ell}{t}}{2v} \leq U.$$

Then every $I \subseteq J$ with $|I| \leq q$ satisfies

$$\sum_{i \in I} (M_i - U)_+ \leq N - U.$$

Proof. We first bound sums of counts over every subfamily of size at most q . We will then apply that bound only to the indices whose counts exceed U , since these are the indices contributing to the positive excess.

Put $p = \binom{\ell}{t}/N$ and $\theta = U/N$, so $0 \leq \theta \leq 1$. Our intermediate claim is that every $I \subseteq J$ of size $m \leq q$ satisfies

$$\sum_{i \in I} M_i/N \leq 1 + (m-1)\theta. \quad (\text{A.2})$$

For $m = 0$, the claim is $0 \leq 1 - \theta$. For $m = 1$, it is $M_i \leq N$, which follows from $D_i \subseteq A$. Assume $2 \leq m \leq q$. Lemma A.1, applied to this subfamily and divided by N , gives

$$2z \sum_{i \in I} M_i/N \leq z(z+1) + m(m-1)p \quad (1 \leq z \leq m).$$

For integers $j \geq 2$ and $1 \leq z \leq j$, define

$$b_j(z) = \frac{z-1}{2(j-1)} + \frac{jp}{2z}.$$

Dividing the counting inequality by $2z$ and rearranging gives

$$\sum_{i \in I} M_i/N \leq 1 + (m-1)b_m(z).$$

Thus it suffices to find one z with $b_m(z) \leq \theta$. The hypothesis supplies such a bound at q , namely $b_q(v) \leq \theta$. The next step shows that a bound at q also supplies a bound at every smaller family size m .

Take $j \geq 2$ and a candidate $b_{j+1}(z)$, where $1 \leq z \leq j+1$. We show that either keeping z or decreasing it by one gives a valid candidate at j with no larger value. At the first endpoint, $z = 1$, keep $z = 1$: since $p \geq 0$,

$$b_j(1) = jp/2 \leq (j+1)p/2 = b_{j+1}(1).$$

At the other endpoint, $z = j+1$, decrease it to j ; both $b_j(j)$ and $b_{j+1}(j+1)$ equal $(1+p)/2$.

For $2 \leq z \leq j$, both z and $z - 1$ are valid at j . Direct subtraction gives

$$b_j(z) - b_{j+1}(z) = \frac{z(z-1) - j(j-1)p}{2j(j-1)z},$$

$$b_j(z-1) - b_{j+1}(z) = \frac{(j+1-z)[j(j-1)p - z(z-1)]}{2j(j-1)z(z-1)}.$$

All denominators and the factor $j+1-z$ are positive. If $z(z-1) \leq j(j-1)p$, the first difference is nonpositive, so keep z . Otherwise the second difference is negative, so use $z-1$. In either case the candidate value does not increase.

Starting with $b_q(v)$, repeat this step until the family size is m . It produces a valid candidate $b_m(z) \leq b_q(v) \leq \theta$. Substituting it into the counting bound proves (A.2) for every subfamily of size at most q .

Finally, for the given I , put $I_+ = \{i \in I : M_i > U\}$ and $m = |I_+| \leq q$. Apply (A.2) to I_+ and multiply by N . Since $N\theta = U$, we obtain $\sum_{i \in I_+} M_i \leq N + (m-1)U$. Therefore

$$\sum_{i \in I} (M_i - U)_+ = \sum_{i \in I_+} M_i - mU \leq N + (m-1)U - mU = N - U.$$

This also covers $I_+ = \emptyset$, because the intermediate claim included the empty subfamily. $\square$

B Proof of Corollary 1

We prove Corollary 1. We first construct the schedule and verify its field size, completeness, and communication, and then bound its soundness error. The query budget depends only on λ . The decreasing code rate makes the sampling contribution decay geometrically with depth, so its sum does not introduce a depth factor.

Proof (Proof of Corollary 1). Schedule, completeness, and communication. Since $n \leq 2^{\lceil \log_2 n \rceil} < 2n$, the stated field-size inequalities hold. Put $b = k_0/2$. Pack the root into $\lceil N/k_0 \rceil$ linear rows and $\lceil M/b \rceil$ multiplication row triples, and set $s_0 = k_0$. Each linear row has k_0 slots, and each multiplication triple has b active positions. Interpolate the two input rows of each triple on those positions and use their product as the output row. Since $2b = k_0$, all three polynomials have degree less than k_0 . Fill unused positions of the triple by evaluation. The total witness-row count satisfies

$$d_0 = \left\lceil \frac{N}{2b} \right\rceil + 3 \left\lceil \frac{M}{b} \right\rceil < \frac{N + 6M}{2b} + 4 \leq 9,$$

so $d_0 \leq 8$. Reindex the linear constraints under this packing. Restricting a satisfying packed witness to the active positions recovers a satisfying original witness, and the interpolation construction preserves every satisfying original witness.

At each child, use 16 linear rows of width $k_{\ell+1} = \lceil k_\ell/4 \rceil$, put $|\mathcal{W}_{\ell+1}| = k_{\ell+1}$, and set $s_{\ell+1} = k_{\ell+1}$. At every level, use

$$n_\ell = 8 \cdot 2^{\lfloor \ell/3 \rfloor} k_\ell, \quad t_\ell = \min\{t, n_\ell\}.$$

Thus the ratio of code length to row degree doubles every three levels, while the query budget remains fixed. A level with fewer than t columns opens all its columns.

Each response pair has $4k_\ell$ coefficients, and the child has $16\lceil k_\ell/4 \rceil \geq 4k_\ell$ slots. The recurrence gives $k_\ell = \lceil k_0/4^\ell \rceil$, so the first width at most t occurs after exactly L substitutions. Before stopping, $k_\ell > t \geq 16$, so $2\lceil k_\ell/4 \rceil \leq k_\ell$. Hence both row degrees and code lengths decrease, and the root field has enough distinct points for every child. These are well-formed layouts, and Theorem 2 gives perfect completeness.

The root openings cost at most $11t$, the L child openings cost at most $16tL$ after omitting the publicly zero child masks, and the terminal responses cost $4k_L \leq 4t$. Thus the prover sends at most

$$t(15 + 16L) = O(\lambda(1 + \log(N + M + 1)))$$

field elements.

Soundness. Write $B = 2^{\lambda+3}$. At level $0 \leq \ell \leq L$, put $a = 2^{\lfloor \ell/3 \rfloor}$ and $k = k_\ell$. Then $s_\ell = k$, $n_\ell = 8ak$, $c = 3k + 1$, $h = 3k + 1$, and the overlap size is $3k - 1$. We first bound the rank-exception term, then the sampling term.

For integers $a, k \geq 1$, comparison of the factors in the two binomial products gives

$$\binom{8ak}{3k+1} \leq (8a/5)^{3k+1} \binom{8k}{3k+1} \leq (8a/5)^{4k} 2^{8k} \leq (2048a^4)^k.$$

Indeed, the denominator of each factor ratio is at least $5k$, while its numerator is at most $8ak$, so every ratio is at most $8a/5$.

When $a = 1$, the simpler bound $\binom{8k}{3k+1} \leq 256^k$ suffices: the field-size choice gives $q \geq 8k_0 2^\lambda \geq 256$, because $k_0 \geq 2$ and $\lambda \geq 4$. Otherwise $L \geq 3$. Minimality of the stopping depth gives $t4^{L-1} < k_0$, and

$$a^4 \leq 2^{4\lfloor L/3 \rfloor} \leq 4^{L-1}.$$

Since $t \geq 16$ and $\lambda \geq 4$, we have $2048 \leq 8t2^\lambda$. Consequently

$$2048a^4 \leq 2048 4^{L-1} \leq 8t2^\lambda 4^{L-1} < 8k_0 2^\lambda \leq q.$$

In both cases,

$$\mu_\ell = \frac{\binom{8ak}{3k+1}}{q^{k+1}} \leq \frac{1}{q}.$$

Both $h-1$ and the overlap size are at most $3k$, a fraction $3/(8a)$ of the columns. If the level opens all its columns, then $\delta_\ell = \omega_\ell = 0$. Otherwise it opens t columns, and sampling without replacement gives

$$\delta_\ell, \omega_\ell \leq \left(\frac{3}{8a} \right)^t \leq (Ba)^{-2}.$$

To verify the last inequality, put $g = \lceil (\lambda + 3)/2 \rceil$. Then $t \geq 3g$, $B^2 \leq 16^g$, and $(8/3)^3 \geq 16$, so $(8/3)^t \geq B^2$. The remaining factor follows from $a^t \geq a^2$.

We can bound the integer counting threshold without adding a rounding error. For a positive integer v , write

$$f(v) = \frac{v-1}{2(q-1)} + \frac{q\omega_\ell}{2v}.$$

The defining inequality for u_ℓ makes $f(u_\ell)$ the minimum over positive integers: $f(v+1) - f(v)$ has the sign of $v(v+1) - q(q-1)\omega_\ell$. Put $D = Ba$ and $v = \lceil (q-1)/D \rceil$. Writing $x = q-1$ and $z = Dv - x$ gives $0 \leq z < D$ and

$$f(u_\ell) \leq f(v) \leq \frac{1}{D} + \frac{z(z-D) - (D-1)x}{2xvD^2} \leq \frac{1}{D},$$

where the middle inequality uses $\omega_\ell \leq D^{-2}$. Together with $\delta_\ell \leq D^{-2} \leq D^{-1}$, this proves $\theta_\ell \leq 1/(Ba)$.

The column conditions hold because $k+s = 2k \leq n_\ell$, $h = 3k+1 \leq n_\ell$, and $3k-1 < n_\ell$. The query count is positive, and $\mu_\ell + 1/q \leq 2/q < 1$ supplies the field-size condition. Hence

$$\varepsilon_\ell = \theta_\ell + (1 - \theta_\ell)(\mu_\ell + 1/q) \leq \frac{1}{B2^{\lfloor \ell/3 \rfloor}} + \frac{2}{q}.$$

Each value of a occurs at most three times, so the first contribution sums to at most $3 \sum_{j \geq 0} 2^{-j}/B = 6/B$. The positive integer widths decrease strictly before stopping, so $L+1 \leq k_0$. The field-size bound $q \geq 8k_0 2^\lambda$ therefore controls the sum of the $2/q$ terms. Theorem 5 therefore bounds the recursive soundness error by

$$\sum_{\ell=0}^L \varepsilon_\ell \leq \frac{6}{B} + \frac{2(L+1)}{q} \leq \frac{3}{4 \cdot 2^\lambda} + \frac{1}{4 \cdot 2^\lambda} = 2^{-\lambda}.$$

□