CauchyFold: Residue-Optimal High-Arity Lattice Folding via Scaled Cauchy Challenges

Xiang Wang

Guizhou University, Guiyang, China
ee.wangx24@gzu.edu.cn

Abstract. Folding an accumulator with k fresh inputs under a quadratic relation creates $k + \binom{k}{2}$ mixed terms. Storing them separately makes the intermediate state quadratic in k. CauchyFold instead uses a polynomial representation whose size is linear in k. Scaled Cauchy coefficients encode all mixed terms as a degree-$< k$ polynomial with k coefficients in the mixed-output space. We call this polynomial the Cauchy carrier. A residue characterization for scaled Cauchy interpolation [WX26] makes the representation possible.

CauchyFold turns the carrier into a lattice folding protocol. The prover constructs it without enumerating all source pairs, folds the semantic values over an extension field, and then encodes the folded state canonically. The next fold therefore uses the same committed-state layout and honest opening-norm bound. To reduce public randomness, the lattice reduction replaces explicit projection matrices with short seeds expanded by Nisan's generator [Nis92].

We prove an extract-or-kernel theorem for classical provers. From an accepting prover, the extractor obtains valid source openings or a short nonzero kernel for a commitment matrix. For a compiled arity-16 instance, the no-retry fold transcript uses 10.19 MiB of interactive communication, excluding incoming commitments. For this instance we report exact statistical bounds, typed operation counts, and role-specific lattice-attack estimates.

Keywords: Folding schemes · Lattice-based cryptography · Cauchy matrices · Knowledge extraction

1 Introduction

Folding schemes combine several relation instances into one accumulated instance. The result can be folded again, which makes folding a useful component of recursive proof systems [BCMS20,KST22]. Most constructions first reduce the computation to linear or evaluation claims and then fold those claims. CauchyFold studies a different boundary: the folding challenge is applied while the relation is still quadratic.

The difficulty appears in the quadratic expansion itself. Let Q be a homogeneous quadratic map, and let $B(x, z) = Q(x+z)-Q(x)-Q(z)$ be its polarization.

Folding one accumulated state z_0 with k source states gives

$$Q\Big(z_0 + \sum_{i=1}^{k} a_i z_i\Big) = Q(z_0) + \sum_{i=1}^{k} a_i^2 Q(z_i) + \sum_{i=1}^{k} a_i B(z_0, z_i) + \sum_{i<j} a_i a_j B(z_i, z_j),$$

so the last two sums contain $k + \binom{k}{2}$ mixed outputs. Retaining every value would make the intermediate state quadratic in the arity. This count is a fact about the direct expansion. It is not a lower bound for protocols that first change the relation or the folding boundary.

A companion resource theorem characterizes what must remain at this direct quadratic boundary [WX26]. Choose distinct poles ξ_i, nonzero scales λ_i, and coefficients $a_i(c) = \lambda_i/(c - \xi_i)$ for a challenge c outside the pole set. The entire mixed contribution becomes the evaluation of one polynomial $H(T)$ of degree below k. Its coefficients lie in the span of the outputs of B. If that span has dimension r, the polynomial needs kr field coordinates. Under the boundary and support assumptions of the companion result, this representation attains the exact retained width.

CauchyFold begins from this representation. We call H the Cauchy carrier. The prover commits its coefficients before the challenge c is sampled. The protocol then evaluates the semantic fold over an extension field and returns to bounded lattice openings by encoding the folded values canonically. The output uses the same commitment format and honest coefficient-norm bound as the input accumulator.

1.1 From a carrier representation to a committed protocol

The representation does not immediately give an efficient lattice protocol. Its pairwise definition still contains one bilinear term for every pair of sources. The coefficients $a_i(c)$ are field elements and need not have short integer representatives, so applying them directly to lattice openings would not preserve the opening bound. Extraction creates a related problem: division by a challenge difference is valid modulo q, but an integer lift of that inverse can be large.

The construction addresses these points separately. Product-tree and multipoint algorithms compute the carrier without enumerating the pair table. The fold is carried out on semantic field values and is followed by canonical re-encoding. During extraction, modular division recovers a candidate, while a cross-multiplied relation compares two candidates using only bounded differences. This last step follows the compare-before-clearing method [WF26].

The resulting contributions are as follows.

Direct quadratic folding. CauchyFold combines one accumulated quadratic instance with k fresh instances whose residual components are fixed to zero. Public coordinates remain part of the checked relation. The protocol fixes when the source, carrier, output, and auxiliary commitments are sent, and reduces the remaining field equations through a finite chain of lattice relations.

Fast carrier construction. We give two deterministic algorithms for the same carrier. The multipoint algorithm evaluates the quadratic map at k fixed nonpole points and interpolates. The pole-residue algorithm uses k bilinear evaluations and reconstructs the carrier from its residues. With fixed relation dimensions, both have quasi-linear arithmetic dependence on k. Their output is exactly the polynomial committed by the protocol.

Seeded modular projections. The lattice reduction originally describes every entry of each lazy-ternary projection. We instead generate the complete projection from a short public seed using Nisan's generator for space-bounded computation [Nis92]. A fixed-test argument preserves the projection and retry bounds without adding a computational pseudorandomness assumption.

A finite extraction theorem. For a classical prover that can be restored to its full state at specified points, extraction returns valid source openings or a bounded nonzero kernel for a named commitment matrix. The proof accounts for acceptance-dependent sampling, bounded retries, norm propagation, and conditional running costs. Averaging over the joint setup distribution gives one global time-truncation term and a sum of role-specific module short integer solution (Module-SIS) advantages.

A fully specified concrete instance. A deterministic compiler instantiates a sparse quadratic relation at $k = 16$. It fixes the encoding, field constraints, finite reduction schedule, and all independent commitment matrices. The no-retry transcript contains 334.66 KiB from prover to verifier and 9.86 MiB in the other direction. The latter includes the seeded projections and the still-explicit aggregation randomness. We give exact statistical and message-length calculations, operation counts separated by type, and lattice-attack estimates for every independent matrix.

1.2 Related work

Folding and accumulation. Accumulation separates the task of combining claims from that of finally deciding them [BCMS20,BGH19]. Bulletproofs gives an influential inner-product folding argument [BBB$^+$17], while Halo Infinite connects additive polynomial commitments with proof-carrying data [BDFG20]. Nova folds committed relaxed R1CS instances after committing a cross term [KST22]; SuperNova extends this execution model to non-uniform computations [KS22]. HyperNova and Protostar broaden the relation interface and support multifolding [KS24a,BC23]. Customizable constraint systems supply one such interface [STW23]. ProtoGalaxy, Mova, and NeutronNova explore further arity and prover-cost tradeoffs [EG23,DGMV24,KS24b]. Hash-based schemes instead accumulate code-based claims [BMNW24,BCFW25].

Lattice folding. LatticeFold introduces folding for bounded lattice commitments, where controlling the witness norm is essential [BC26]. LatticeFold+ and Lova refine the linear-relation approach [BC25,FKNP25]. Neo, Cyclo, and Symphony combine different relation interfaces with decomposition, range checks, or high-arity aggregation [NS26,GLLO26,Che25]. Table 1 compares the relation presented at the folding challenge.

Table 1. Relation interfaces at the folding challenge. Reduction refers to reducing the computation relation to linear or evaluation claims.

Scheme	Relation at folding boundary	Lattice?	Reduction before fold?
Nova [KST22]	Relaxed R1CS	No	No
HyperNova [KS24a]	CCS evaluation claims	No	Yes
LatticeFold / LatticeFold+ [BC26,BC25]	Linear/evaluation relation	Yes	Yes
Neo / SuperNeo [NS26]	CCS evaluation claims	Yes	Yes
Cyclo [GLLO26]	Principal linear relation	Yes	Yes
SALSAA [KLOT25]	Semi-structured linear relation	Yes	Yes
Symphony [Che25]	Linear evaluation relation	Yes	Yes
PikkuFold [Osa26]	Polynomial-evaluation relation	Yes	Yes
ProtogaLattice [BNP26]	General polynomial relation	Yes	No
CauchyFold	**Quadratic residual relation**	**Yes**	**No**

Reduction before fold asks whether the cited construction reduces its computation relation to linear or evaluation claims before the challenge that forms the accumulated instance. For SALSAA and PikkuFold, this refers to their applications to algebraic intermediate representation (AIR) relations. This classification does not exclude sumcheck or other reductions later in the protocol.

Most lattice constructions in Table 1 fold a linear or evaluation relation obtained through an earlier reduction. ProtogaLattice also folds nonlinear relations directly, using a general polynomial interface [BNP26]. CauchyFold retains a quadratic residual relation through the folding challenge. It fixes the Cauchy carrier before this challenge and performs the field-to-linear reduction after forming the fold.

Lattice arguments and extraction. Succinct lattice arguments based on homomorphic commitments provide alternatives to generic PCP-based constructions [BLNS20]. Product proofs make multiplicative relations between committed values explicit [ALS20]. Leveled bilinear modules give a route to polylogarith-

mic verification [BCS23]. LaBRADOR, Greyhound, Hachi, Jindo, and Grand Danois provide closely related techniques for proving structured relations over lattices [BS22,NS24,NOZ26,HLSS26,KK26]. RPS gives reductions of knowledge for structured linear and inner-product relations, while RnR uses random projections in reductions for structured linear relations [KLNO24,KLNO25]. Vanishing-polynomial commitments provide another route to recursive linear arguments [CLM23]. Short-challenge invertibility and relaxed binding comparisons are particularly relevant here [LS18,BC25].

Adaptive special soundness studies extraction from an accepting prover beyond a single algebraic transcript [AKLY24]. Other work connects special soundness with commitment binding [KLPS26], while RoKoko develops rewind-based extraction for lattice protocols [KLN+26]. Our complete-reset model makes the sampling experiment explicit. The comparison step uses the short-kernel viewpoint of [WF26]. The resource theorem of [WX26] supplies the carrier representation and its exact-width interpretation, rather than a communication lower bound for arbitrary proof systems.

Field checks and polynomial algorithms. The field front end uses multilinear extensions and sumcheck [LFKN92], a common route from algebraic constraints to a small number of evaluation claims [Set19,XZZ+19,GLS+21,DP25]. The two carrier algorithms use product trees, multipoint evaluation, and interpolation [vzGG13,BS05]. Their contribution is the way these polynomial routines evaluate the Cauchy mixed term and produce the committed basis coefficients.

1.3 Scope and organization

The security statement concerns one classical interactive node with a fixed finite internal reduction schedule. It includes the joint randomized setup and identifies the Module-SIS instance for each possible kernel output. The concrete attack estimates and operation counts quantify this instance; they are not wall-clock measurements or a claim of concrete security for an unbounded recursive composition.

Section 2 explains the construction before its formal details. Section 3 fixes the relations and notation, and Section 4 gives the carrier algorithms. The protocol and extraction theorem appear in Sections 5 and 6. Section 7 presents the efficiency bounds and the compiled instance. The appendices provide the algebraic proofs, encoding specification, extraction details, per-role assumption table, and detailed cost calculations.

2 Technical Overview

CauchyFold uses two representations. The fold itself is defined over an extension field, where the quadratic relation is easy to state. The commitments use a lattice ring and require bounded coefficient vectors. Canonical encoding connects the semantic and lattice representations. Figure 1 shows the order in which the carrier, the folded record, and the later linear checks are fixed.

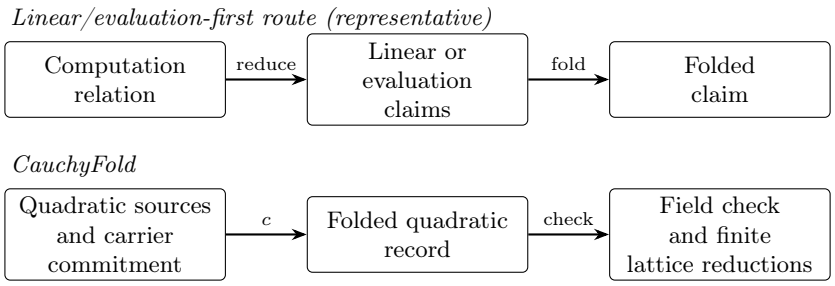

Fig. 1. The order of the semantic fold and the relation reduction. In the lower row, c is the Cauchy challenge. The upper row describes a common route; ProtogaLattice also takes an algebraic route.

2.1 A linear-size record of the mixed terms

Fix one accumulated state z_0 and k source states $z_1, \dots, z_k$. The pairwise expansion from the introduction contains $\binom{k}{2}$ source–source terms. Scaled Cauchy coefficients give those terms more structure. For poles ξ_i, scales λ_i, and challenge c, write $a_i(c) = \lambda_i/(c - \xi_i)$ and $D(T) = \prod_i (T - \xi_i)$. The complete mixed contribution can be written as $H(c)/D(c)$, where

$$H(T) \in Y_B[T]_{<k}.$$

Here Y_B is the span of the outputs of the polarized map B. If $r = \dim Y_B$, the k coefficients of H occupy kr field coordinates. The companion resource theorem establishes this representation and its exact width at the stated direct quadratic boundary [WX26].

The prover commits H before receiving c. This ordering matters. Once the carrier and the source commitments are fixed, a wrong tuple gives a nonzero polynomial after the Cauchy denominators are cleared. Evaluations at the poles isolate the source errors, and the leading coefficient isolates an accumulator error. The remaining coefficients check the carrier itself.

2.2 Constructing the carrier

A linear-size result is useful only if it can be computed without first building the quadratic pair table. CauchyFold provides two constructions.

The multipoint construction evaluates the carrier at k fixed points outside the pole set and interpolates the resulting degree-$< k$ polynomial. A derivative identity evaluates the diagonal correction without summing k squared denominators separately.

The pole-residue construction works directly at the poles. Define

$$v_i = z_0 + \sum_{j \neq i} \frac{\lambda_j z_j}{\xi_i - \xi_j}, \qquad R_i = \lambda_i B(z_i, v_i).$$

The values R_i are the residues of H/D. Product and remainder trees compute all regular parts v_i without a separate loop over every source pair. A second tree reconstructs H from the residues. Both constructions produce exactly the carrier committed by the protocol. Their arithmetic cost is quasi-linear in k when the relation dimensions are fixed.

2.3 Returning to bounded lattice openings

The Cauchy coefficients live in the extension field. Nothing guarantees that their integer representatives are short. CauchyFold therefore does not multiply the lattice openings by these coefficients. It computes the semantic fold

$$z^\star = z_0 + \sum_i a_i(c) z_i, \qquad E^\star = E_0 + \sum_i a_i(c)^2 E_i + D(c)^{-1} H(c)$$

in the field, and then encodes $(z^\star, E^\star)$ canonically. The output uses the same state commitment matrix, layout, and honest coefficient-norm bound as an accumulated input.

The new commitment must still be linked to the source commitments and the carrier. A sparse field circuit checks the fold, the quadratic relation, the public coordinates, and all canonical ranges. Sumcheck reduces these constraints to evaluations of linear forms in the encoded witness. Expanding those evaluations in the fixed field basis gives a scalar linear system over $\mathbb{F}_q$. From this point onward, the backend only needs to prove a bounded opening of an explicit linear relation.

2.4 A finite lattice reduction

One lattice layer partitions its opening into ring blocks. The prover commits radix digits of the block commitments before the verifier chooses a projection. The projection checks the norm of a candidate recovered later by the extractor. Independent field rows then aggregate the public linear equations, and short ring challenges combine the blocks into a smaller response. A radix split of that response, together with the committed auxiliary digits, becomes the next layer's witness. The final layer opens a bounded response directly and links one reconstructed block commitment to a pivot fixed before the challenge.

Sending a full lazy-ternary projection, whose entries lie in $\{-1, 0, 1\}$, would dominate verifier communication. In the concrete profile, the verifier instead sends a complete seed for a space-bounded generator. Both parties expand that seed into exactly the same projection using streaming forward and adjoint routines.

The relevant security question is narrow: for every vector fixed before the seed, does the generated projection preserve the acceptance probability of the complete modular norm test? Section 6 answers this by expressing that test as one finite-state computation. The seed changes its probability by at most 2^{-150}, which fits inside the original statistical margin. Aggregation rows remain independent field elements and are still sent explicitly.

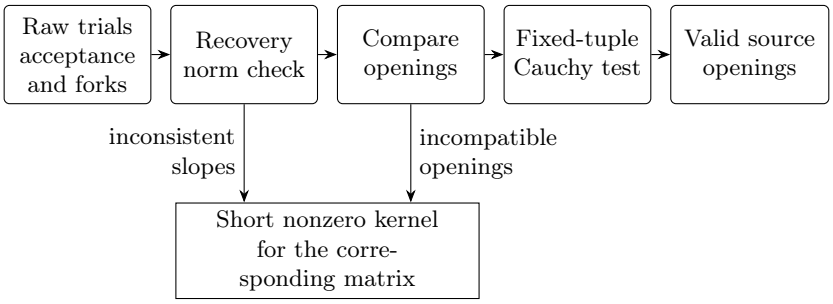

Fig. 2. Extract-or-kernel flow. Acceptance-gated raw trials provide coordinate forks; recovered candidates are checked against the norm bound and compared through the same fixed commitments. Inconsistent slopes or incompatible openings yield a short nonzero kernel for the corresponding matrix. Outside the stated statistical error, a compatible source/carrier tuple passes the Cauchy test only if it gives valid source openings.

2.5 Extraction from bounded differences

The extractor runs the protocol from complete saved states and obtains three bounded responses that differ in one short-challenge coordinate. Let X_1, X_2 be the response differences and Δ_1, Δ_2 the corresponding challenge differences. Modular division gives two candidate slopes. If they agree, the common slope is the recovered source block. If they differ, cross-multiplication gives

$$v = X_1\Delta_2 - X_2\Delta_1.$$

This vector is a nonzero modular kernel of the parent commitment matrix, and its norm is controlled by the response and challenge bounds. The inverses are used only for modular recovery; they are never lifted as short integers.

Projection checks certify the norm of a recovered candidate. A fresh trial is sampled from the checkpoint at which the candidate is already fixed, so the fixed-vector projection bound applies. Differences between two openings of an unchanged commitment give short kernels for the corresponding auxiliary matrix. Repeating this argument through the finite chain yields an opening of the field witness or a named short kernel. The field and Cauchy comparisons then recover the original source relations.

Accepted continuations need not be uniformly distributed over challenges. The extractor therefore analyzes raw trials before conditioning on acceptance and charges every restore, retry, continuation, and final comparison. A single global truncation converts its expected running time into the role-specific Module-SIS reductions stated in Section 6.

3 Preliminaries and Relations

For a positive integer n, write $[n] = \{1, \ldots, n\}$ and $[n]_0 = \{0, \ldots, n-1\}$. The security parameter is κ. We reserve λ_i for the Cauchy scales.

3.1 Fields, rings, and norms

Let $q = q(\kappa)$ be an odd prime and let $\mathbb{K}/\mathbb{F}_q$ be an extension of degree t, with a fixed public basis $1, \theta, \ldots, \theta^{t-1}$. Semantic relations and the field sumcheck are defined over $\mathbb{K}$. Lattice commitments use the rings

$$\mathcal{O} = \mathbb{Z}[X]/(X^d + 1), \qquad R_q = \mathcal{O}/q\mathcal{O},$$

where d is a power of two. The extension-field generator θ and the ring variable X have different roles. In particular, packing field coefficients into R_q does not identify $\mathbb{K}$ with R_q.

For a ring vector v, $\mathrm{coef}(v)$ is its coefficient vector in increasing powers of X, concatenated in vector order. Each residue coefficient has a unique centered representative in $[-(q-1)/2, (q-1)/2]$. The notation $\|v\|_2$ refers to the Euclidean norm of this integer coefficient vector. An integer vector in $\mathcal{O}^b$ is measured before reduction modulo q. For $c \in \mathcal{O}$, $\|c\|_{\mathrm{op}}$ denotes the induced Euclidean norm of negacyclic multiplication by c. Thus

$$\|cv\|_2 \le \|c\|_{\mathrm{op}} \|v\|_2.$$

These norms describe different objects: $\|c\|_{\mathrm{op}}$ controls multiplication, while an extraction radius bounds the resulting integer kernel.

We write $\mathrm{ct}(f)$ for the constant coefficient of $f \in R_q$. Scalar linear forms can be expressed through this operation. For $a = (a_0, \ldots, a_{d-1}) \in \mathbb{F}_q^d$, put

$$\iota(a) = a_0 - \sum_{j=1}^{d-1} a_j X^{d-j}.$$

Then $\mathrm{ct}(\iota(a)v) = \langle a, \mathrm{coef}(v) \rangle$. Applying this identity block by block converts scalar coefficient equations into constant-term equations without changing the underlying scalar relation.

3.2 Quadratic relations and mixed outputs

A relation is specified by an explicit homogeneous quadratic map $Q : \mathbb{K}^n \longrightarrow \mathbb{K}^y$. Its polarization is the symmetric bilinear map

$$B(x, z) = Q(x + z) - Q(x) - Q(z).$$

The description of Q, rather than a value oracle alone, determines its scalar extension and the field circuit used by the protocol. Write

$$Y_B = \mathrm{span}_{\mathbb{K}}\{B(x, z) : x, z \in \mathbb{K}^n\}, \qquad r = \dim_{\mathbb{K}} Y_B.$$

Fix an injective basis map $U : \mathbb{K}^r \to \mathbb{K}^y$ with image Y_B, and its inverse on this image, $J : Y_B \to \mathbb{K}^r$. Thus $UJb = b$ for every mixed output b. We distinguish y, the ambient output dimension, from r, the dimension needed to represent mixed terms.

A semantic state is $z = (u_{\mathrm{hom}}, x_{\mathrm{pub}}, z_{\mathrm{priv}})$, where the first two components are public and z_{priv} is private. The coordinate u_{hom} homogenizes the original relation. An accumulated record also contains a residual $E \in \mathbb{K}^y$ and satisfies $Q(z) = E$. Fresh records satisfy the stricter conditions

$$u_{\mathrm{hom}} = 1, \qquad E = 0, \qquad z \in \mathbb{F}_q^n.$$

The last condition means that all nonconstant extension-field components vanish. It is included when the original language is over $\mathbb{F}_q$, as in our concrete instance. It prevents an extension-field witness from changing the intended language.

3.3 Canonical encodings and commitments

Let $m_q = \lceil \log_2 q \rceil$. A field element $a = \sum_{\nu=0}^{t-1} a_\nu \theta^\nu$ is encoded by the m_q-bit binary representation of each $a_\nu \in \{0, \ldots, q-1\}$, in the declared basis order. The map Enc concatenates these bits and appends any prescribed zero padding. Decoding is $\mathbb{F}_q$-linear on the base-field coordinate vector; groups of t coefficients are then interpreted in the fixed basis of $\mathbb{K}$. The encoding map itself is not linear. If the encoded length is L, then

$$\mathrm{Dec}(\mathrm{Enc}(s)) = s, \qquad \|\mathrm{Enc}(s)\|_2^2 \le L. \tag{1}$$

Canonicality requires both Boolean bits and a represented integer strictly below q. Appendix B gives the comparator constraints and the complete segment layout.

The map Pack groups consecutive d coefficients into ring elements. A state commitment has the form

$$C = A_{\mathrm{st}}\,\mathrm{Pack}(\mathrm{Enc}(z, E)), \qquad A_{\mathrm{st}} \in R_q^{a_{\mathrm{st}} \times b_{\mathrm{st}}}.$$

The same matrix is used for the accumulator, all fresh records, and the folded output. Carrier and field-auxiliary encodings use two further matrices A_H and A_{aux}. All independently sampled matrices are uniform, with their joint distribution given by the product of their individual distributions. Additively homomorphic lattice commitments support algebraic manipulation of committed values [BDL+16]. Here their binding property is used through the explicit short-kernel problem below.

Definition 1 (Accumulated and fresh committed relations). *The public instance is $(u_{\mathrm{hom}}, x_{\mathrm{pub}}, C)$. An opening contains z_{priv}, a residual E, and an encoding e. Put $z = (u_{\mathrm{hom}}, x_{\mathrm{pub}}, z_{\mathrm{priv}})$. The opening belongs to $\mathcal{R}_{\mathrm{acc}}$ if*

$$e = \mathrm{Enc}(z, E), \qquad C = A_{\mathrm{st}}\,\mathrm{Pack}(e), \qquad Q(z) = E.$$

It belongs to $\mathcal{R}_{\mathrm{fresh}}$ if, in addition, $u_{\mathrm{hom}} = 1$, $E = 0$, and $z \in \mathbb{F}_q^n$.

Thus an unspecified residual cannot make an arbitrary fresh witness valid.

3.4 The Cauchy carrier

Fix distinct poles $\xi_1, \dots, \xi_k \in \mathbb{K}$, nonzero scales $\lambda_1, \dots, \lambda_k$, and an exactly samplable challenge support $\mathcal{C} \subseteq \mathbb{K} \setminus \{\xi_1, \dots, \xi_k\}$. Require $|\mathcal{C}| > 2k$ and a public polynomial bound on the expected cost of exact sampling. Put

$$a_i(c) = \frac{\lambda_i}{c - \xi_i}, \quad D(T) = \prod_{i=1}^{k} (T - \xi_i), \quad P_i(T) = \frac{D(T)}{T - \xi_i}.$$

The carrier representation from [WX26] is

$$\begin{aligned} H_{\mathrm{src}}(T) = & \sum_{i=1}^{k} \lambda_i P_i(T) B(z_0, z_i) \\ & + \sum_{1 \le i < j \le k} \lambda_i \lambda_j \frac{D(T)}{(T - \xi_i)(T - \xi_j)} B(z_i, z_j). \end{aligned} \tag{2}$$

It lies in $Y_B[T]_{<k}$. Its basis coordinates JH_{src} therefore consist of kr field elements. We use this representation as an input to the construction. The exact retained-width theorem, its boundary-relative interpretation, and Cauchy attainment belong to the resource analysis of [WX26].

Given $H = U\widehat{H}$ with $\widehat{H} \in \mathbb{K}^r[T]_{<k}$, the semantic fold at $c \in \mathcal{C}$ is

$$\begin{aligned} z^* &= z_0 + \sum_{i=1}^{k} a_i(c) z_i, \\ E^* &= E_0 + \sum_{i=1}^{k} a_i(c)^2 E_i + D(c)^{-1} H(c). \end{aligned} \tag{3}$$

For $H = H_{\mathrm{src}}$ and valid input records, polarization gives $Q(z^*) = E^*$. The output is committed through a fresh canonical encoding $\mathrm{Enc}(z^*, E^*)$. This keeps its shape and honest norm bound equal to those of an accumulator, even though the semantic Cauchy coefficients need not be short integers. No division by u^* is used; the accumulated relation permits $u^* = 0$.

3.5 Multilinear extensions and sumcheck

For a table $f : \{0,1\}^\ell \to \mathbb{K}$, its multilinear extension is the unique polynomial of individual degree at most one agreeing with the table on the Boolean cube. Define

$$\mathrm{eq}(v, b) = \prod_{j=1}^{\ell} \big(v_j b_j + (1 - v_j)(1 - b_j) \big).$$

Then $\widetilde{f}(v) = \sum_{b \in \{0,1\}^\ell} \mathrm{eq}(v, b) f(b)$. The concrete table index uses its first variable as the most significant bit. This convention is separate from the least-significant-bit-first encoding of field coefficients.

We use the usual public-coin sumcheck protocol [LFKN92]. At each round the prover fixes a univariate polynomial before the verifier samples the next field challenge. The analysis uses the total-degree identity bound of Schwartz [Sch80]. If the round polynomial has degree at most D_{sc}, a false claim survives ℓ rounds with probability at most $D_{\mathrm{sc}}\ell/|\mathbb{K}|$, in addition to any preceding random compression error. Our field check has degree three per round. Together with its initial equality-weighted compression, its error is at most $4\ell_F/|\mathbb{K}|$. Section 5 specifies the terminal evaluations and their commitment linkage. Sumcheck arguments over modules give a general framework for connecting such polynomial checks to commitment openings [BCS21]; our handoff is the scalar relation specified in that section.

3.6 Module-SIS and the extraction model

For a matrix role j, let $\vartheta_j = (q, d, a_j, b_j, \beta_j)$. The corresponding Module-SIS problem gives a uniform matrix $A_j \in R_q^{a_j \times b_j}$ and asks for

$$v \in \mathcal{O}^{b_j}, \qquad A_j v = 0 \pmod{q}, \qquad v \bmod q \neq 0, \qquad \|v\|_2 \leq \beta_j. \tag{4}$$

Module lattices and short integer relations provide the underlying commitment assumption [Ajt96,LS15]. When $\beta_j < q$, the usual nonzero-integer-vector formulation agrees with (4) inside the stated ball: a nonzero vector divisible by q has norm at least q. This observation identifies the assumption; it is not a hardness estimate.

We denote by $\mathrm{Adv}^{\mathrm{MSIS}}_{\vartheta_j}(T)$ the maximal success probability within time T in the chosen computational model. Coefficient expansion preserves the negacyclic block structure of A_j; it does not turn the public key into a uniformly random scalar matrix. The security theorem keeps a separate advantage term for every independently sampled matrix.

The prover and extractor are classical. A reset restores the complete prover configuration at a specified checkpoint, including its random tape and mutable state. Verifier coins sampled after that checkpoint are fresh in each raw trial. The extractor may inspect complete continuations and reject some of them, but it cannot reinterpret their accepted challenge distribution as uniform. Section 6 handles this selection explicitly and counts both failed and successful trials in its runtime. Commitment binding is used through comparisons of actual openings. A commitment string is never assumed to determine an information-theoretically unique witness.

4 Cauchy Carrier Algorithms

The carrier has a short coefficient representation, but its definition contains quadratically many pairs of source records. This section gives two algorithms that construct those coefficients without enumerating the pairs. Both operate before the Cauchy challenge. They produce exactly the same object for the commitment in Section 5.

4.1 The carrier representation

Use $Q : S \to Y$, its polarization B, and the basis maps U, J from Section 3. Write $n = \dim_{\mathbb{K}} S$, $y = \dim_{\mathbb{K}} Y$, and $r = \dim_{\mathbb{K}} Y_B$. In this section H denotes the source carrier H_{src} in (2), and

$$P_{ij}(T) = \frac{D(T)}{(T - \xi_i)(T - \xi_j)}.$$

The imported representation places H in $Y_B[T]_{<k}$; its committed coordinates are those of $\widehat{H} = JH \in \mathbb{K}^r[T]_{<k}$.

The same representation gives the rational identity

$$\begin{aligned} \frac{H(T)}{D(T)} = Q\left(z_0 + \sum_i \frac{\lambda_i z_i}{T - \xi_i}\right) - Q(z_0) \\ - \sum_i \frac{\lambda_i^2 Q(z_i)}{(T - \xi_i)^2}. \end{aligned} \tag{5}$$

These representation facts are imported from [WX26]; the algorithms below address the cost of constructing their coefficients. All arithmetic takes place over $\mathbb{K}$. If the original relation is defined over $\mathbb{F}_q$, its scalar extension is defined by the same polynomial equations.

4.2 Multipoint construction

Choose k distinct public points $\tau_1, \ldots, \tau_k$ outside the pole set. These are internal interpolation points. They are fixed before the protocol and introduce no verifier messages or new challenges. Define the two vector-valued numerators

$$N(T) = \sum_i \lambda_i z_i P_i(T), \qquad M_Q(T) = \sum_i \lambda_i^2 Q(z_i) P_i(T).$$

A product tree constructs D, N, and M_Q. At an internal node with children L, R, it combines a denominator and a numerator by

$$D_{L \cup R} = D_L D_R, \qquad N_{L \cup R} = N_L D_R + N_R D_L.$$

The same recurrence applies to M_Q. Thus a vector coordinate is processed by polynomial products along the tree, rather than by a dense Cauchy matrix.

The squared denominators in (5) need no second evaluation routine. Formal differentiation gives

$$\sum_i \frac{\lambda_i^2 Q(z_i)}{(T - \xi_i)^2} = \frac{M_Q(T) D'(T) - M_Q'(T) D(T)}{D(T)^2}.$$

Evaluate D, D', N, M_Q, M_Q' at all τ_j. For each point, set

$$\begin{aligned} s_j &= z_0 + N(\tau_j)/D(\tau_j), \\ h_j &= D(\tau_j) \left(Q(s_j) - Q(z_0) - \frac{M_Q(\tau_j) D'(\tau_j) - M_Q'(\tau_j) D(\tau_j)}{D(\tau_j)^2} \right). \end{aligned}$$

The values h_j are evaluations of H. Interpolating them recovers its k ambient-Y coefficients; applying J gives the committed coordinates. Only the k evaluations $Q(s_j)$ are new if the source Q-values are already available. Otherwise those source values require another $k+1$ evaluations.

This construction uses standard product trees, remainder trees, and interpolation [vzGG13,BS05]. Appendix A specifies the recurrences and the accounting for inversions.

4.3 Pole-residue construction

The second algorithm works at the poles themselves. It avoids the extra interpolation points and constructs the carrier directly in its image basis. Assume $\operatorname{char} \mathbb{K} \neq 2$, as in our instantiation. Define the regular part of the folded source at ξ_i by

$$v_i = z_0 + \sum_{j \neq i} \frac{\lambda_j z_j}{\xi_i - \xi_j}, \qquad R_i = \lambda_i B(z_i, v_i).$$

The useful formulas are

$$\begin{aligned} H(T) &= \sum_i R_i P_i(T), \\ v_i &= z_0 + \frac{N'(\xi_i)}{D'(\xi_i)} - \lambda_i z_i \frac{D''(\xi_i)}{2D'(\xi_i)}. \end{aligned} \tag{6}$$

In particular, the apparent k-by-k sum in the definition of the v_i can be evaluated with polynomial arithmetic.

Construct D, N, differentiate them, and evaluate D', D'', N' at the poles. The distinctness of the poles gives $D'(\xi_i) \neq 0$. Compute their inverses in a batch, form the v_i, and make k calls to B. After applying J to each R_i, another numerator tree reconstructs

$$\widehat{H}(T) = \sum_i J(R_i) P_i(T).$$

Only r output coordinates pass through this last tree. In contrast, the first numerator tree processes the n coordinates of each source vector.

The subtraction in (6) removes the contribution of the pole's own summand. Indeed,

$$N'(\xi_i) = \lambda_i z_i P_i'(\xi_i) + D'(\xi_i) \sum_{j \neq i} \frac{\lambda_j z_j}{\xi_i - \xi_j}, \qquad P_i'(\xi_i) = D''(\xi_i)/2.$$

Without this subtraction the reconstructed carrier would contain diagonal terms absent from (2).

4.4 Correctness and equivalence

Theorem 1 (Fast carrier construction). *For the data above, the multipoint algorithm outputs the carrier $\widehat{H} = JH$ of (2). In odd characteristic, the pole-residue algorithm outputs the same polynomial. Neither algorithm requires a table of the values $B(z_i, z_j)$.*

Table 2. Carrier algorithms. Map-call counts are for $k = 16$ and are reported as (Q, B, J). Multipoint construction uses cached source Q-values; without them, add 17 calls to Q. The streaming reference uses precomputed scalar P_i, P_{ij}.

Algorithm	Polynomial-coordinate work	Map calls, $k = 16$
Streaming pairs	$O(rk^3)$	$(0, 136, 136)$
Multipoint	$O((n+y)\mathsf{P}(k))$	$(16, 0, 16)$
Pole-residue	$O((n+r)\mathsf{P}(k))$	$(0, 16, 16)$

Proof. For multipoint construction, the differentiated numerator identity and (5) give $h_j = H(\tau_j)$. There is a unique polynomial of degree below k with these k values.

For the pole-residue construction, expand $\sum_i R_i P_i$. The terms involving z_0 give the first sum in (2). For a pair $i < j$, symmetry of B combines its two contributions into

$$\lambda_i \lambda_j B(z_i, z_j) \frac{P_i(T) - P_j(T)}{\xi_i - \xi_j} = \lambda_i \lambda_j B(z_i, z_j) P_{ij}(T).$$

The derivative formula computes exactly these R_i. Both algorithms therefore recover H, and the linear map J recovers the same image-basis coefficients.

No validity assumption on the source records is needed for this algorithmic statement. It is an identity for arbitrary source vectors. The protocol uses it to construct the honest carrier, while the security argument treats an adversarially committed carrier separately.

4.5 Arithmetic complexity

Let $M(k)$ be a bound on the cost of multiplying two scalar polynomials of degree $O(k)$ over $\mathbb{K}$. Assume $M(k) \geq k$ and that $M(k)/k$ is nondecreasing. Also assume $M(2k) = O(M(k))$. For brevity, write $\mathsf{P}(k) = M(k)\log(k+1)$. Let C_Q, C_B, C_J be the costs of one Q, B, and J application. The tree operations cost $O(\mathsf{P}(k))$ per vector coordinate. Hence

$$T_{\text{multi}} = O\big((n+y)\mathsf{P}(k) + kC_Q + kC_J\big), \tag{7}$$

$$T_{\text{residue}} = O\big((n+r)\mathsf{P}(k) + k(C_B + C_J)\big). \tag{8}$$

The first bound includes construction in ambient Y and subsequent basis conversion. It does not replace y by r without accounting for that conversion. The public basis maps are part of the relation description; discovering such a basis, when necessary, is separate preprocessing.

Table 2 separates polynomial-coordinate work from map evaluations. The streaming reference evaluates each of the $k + \binom{k}{2}$ bilinear terms, converts it to image coordinates, and accumulates its scalar polynomial coefficient by coefficient. It need not store the pair table, although its arithmetic still visits every

pair. With $M(k) = \widetilde{O}(k)$ and fixed relation dimensions and map costs, both fast algorithms have $\widetilde{O}(k)$ arity dependence. Section 7 accounts for the remaining protocol costs.

5 The CauchyFold Protocol

The protocol first folds the quadratic relation over $\mathbb{K}$ and then reduces the committed statement to scalar linear equations over $\mathbb{F}_q$. A finite lattice reduction chain proves these equations. The concrete profile uses complete public seeds for its lazy-ternary projections. Aggregation vectors remain independent and are sent explicitly.

Figure 3 records the messages that are fixed before each challenge and the two bounded retry loops.

5.1 Public parameters and committed relations

Fix the quadratic map, the public-coordinate map, and the committed relations from Section 3. The input is one accumulator and k strict fresh records. Their state matrix A_{st} is also used for the folded output. Setup independently samples A_H for the carrier and A_{aux} for the field-circuit auxiliary values. Each matrix is uniform in its declared module-matrix space. The encoding layout, segment lengths, public coordinates, and padding are part of the parameters.

Let $U : \mathbb{K}^r \to Y_B$ be the fixed basis embedding for the mixed-output space. The carrier is represented as $H(T) = U(\widehat{H}(T))$, where $\widehat{H}(T) = \sum_{j=0}^{k-1} \widehat{H}_j T^j \in \mathbb{K}^r[T]$. Thus its encoded semantic payload has exactly kr extension-field coordinates. Setup also fixes the poles ξ_i, the nonzero scales λ_i, and the uniformly sampled support $\mathcal{C} \subseteq \mathbb{K} \setminus \{\xi_1, \ldots, \xi_k\}$. We use $D(T) = \prod_i (T - \xi_i)$ and $a_i(c) = \lambda_i/(c - \xi_i)$.

The lattice backend has layers $i = 0, \ldots, L$. At layer i, its input is

$$L_i \operatorname{coef}(w) = y_i, \qquad w = (w_0, \ldots, w_{s_i - 1}), \quad w_j \in R_q^{n_i}. \tag{9}$$

Zero padding makes the coefficient length $ds_i n_i$. The honest squared-norm bound is S_i, while extraction allows $\gamma_i = \sigma_i \sqrt{S_i}$ with $\sigma_i \geq 3$. The public parameters include the main matrix $A_i \in R_q^{a_i \times n_i}$, an even radix $\rho_i < q$, m_i projection rows, τ_i aggregations, and retry bounds $R_{P,i}, R_{C,i}$. The auxiliary matrices used below are independently sampled with their stated dimensions.

The projection family at layer i is a deterministic map $\Pi_i : \{0,1\}^{\ell_i^{\mathrm{seed}}} \to \{-1, 0, 1\}^{m_i \times ds_i n_i}$. Its seed contains one initial binary block and every affine Toeplitz hash used by the recursive generator. The parameters fix the block length, recursion depth, output order, and canonical bit packing. The verifier samples all ℓ_i^{seed} bits independently for each raw projection attempt. Section 6 gives the resulting statistical contract, and Appendix C gives the finite-state calculation.

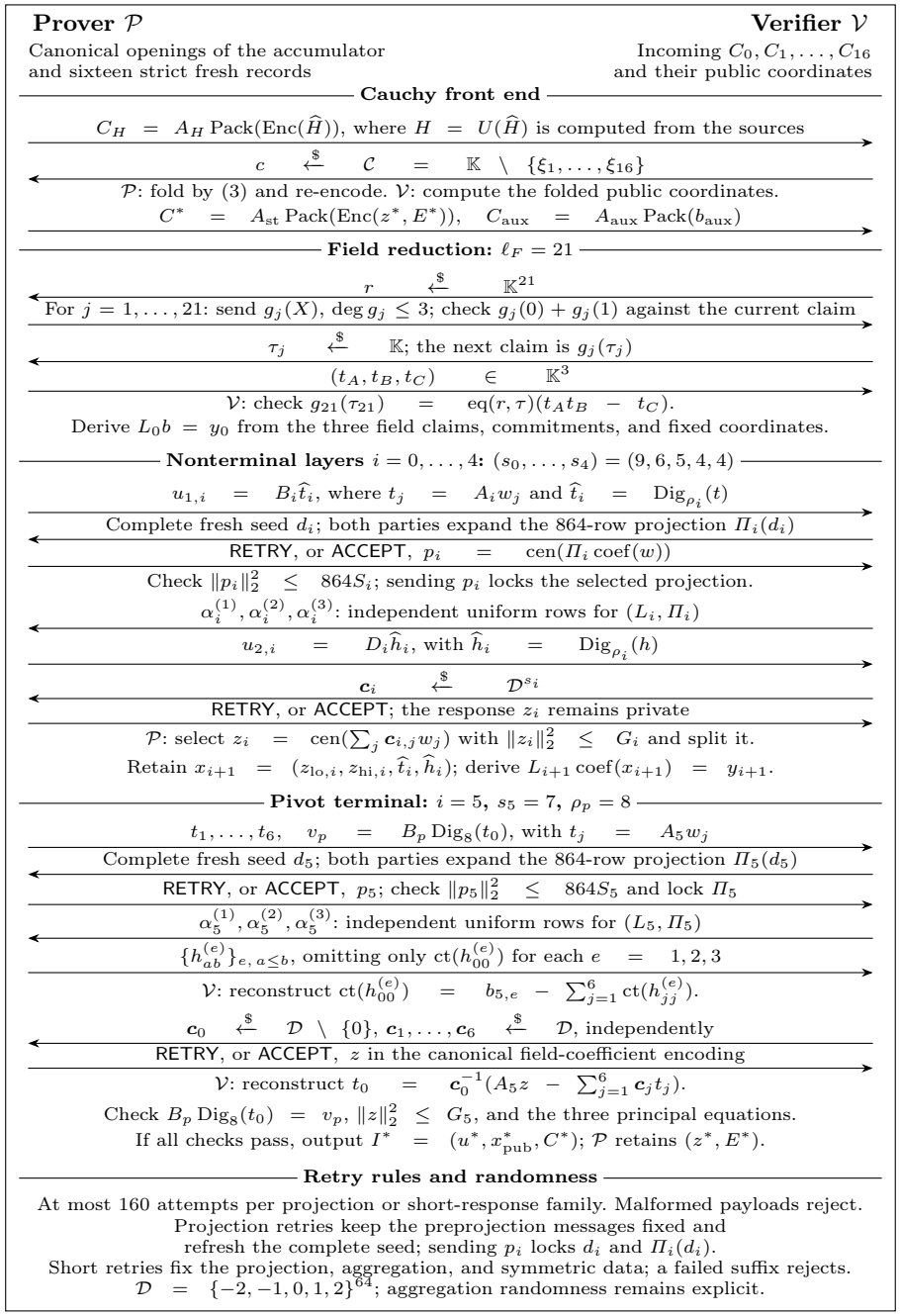

Fig. 3. CauchyFold transcript for the seeded-projection profile at $k = 16$. Arrows denote messages; input commitments are already public. Section 5 defines the seed expansion, scalar lifts, symmetric values, principal equations, and retry checkpoints.

Each short-challenge support $\mathcal{D}_i \subset R_q$ is closed under negation and contains zero. Distinct elements have unit differences. Its uniform distribution satisfies

$$|\mathcal{D}_i| = M_i \geq 4, \qquad \|\boldsymbol{c}\|_{\mathrm{op}} \leq T_i^{\mathrm{op}}, \qquad \mathbb{E}\|\operatorname{cen}(\boldsymbol{c}v)\|_2^2 \leq \mu_i \|v\|_2^2. \tag{10}$$

Here $\boldsymbol{c}$ is a ring element, unlike the Cauchy challenge $c \in \mathbb{K}$. For $q \equiv 5 \pmod 8$ and the parameters of Section 7, we use the full coefficient box $\{-2,-1,0,1,2\}^d$, with $M_i = 5^d$ and $T_i^{\mathrm{op}} = \mu_i = 2d$. The unit condition follows from the short-difference criterion of [BC25, Lemma 2.4]; the moment calculation is given in Appendix C.

5.2 Commit the carrier before the Cauchy challenge

The input commitments $C_0, C_1, \ldots, C_k$ and their public coordinates are already part of the statement. An honest prover computes the carrier H_{src} by either algorithm in Section 4. It sends

$$C_H = A_H \operatorname{Pack}\big(\operatorname{Enc}(\widehat{H}_0, \ldots, \widehat{H}_{k-1})\big).$$

Only then does the verifier sample c uniformly from $\mathcal{C}$. This fixes the checkpoint at which the source commitments and the carrier commitment can be compared across Cauchy replays. The degree bound is supplied by the fixed coefficient layout.

5.3 Form the folded record

The prover computes (z^*, E^*) by (3). The verifier computes the public coordinates of z^* by the same linear combination. The prover sends

$$C^* = A_{\mathrm{st}} \operatorname{Pack}(\operatorname{Enc}(z^*, E^*))$$

and the commitment C_{aux} to the auxiliary values of the field circuit defined next. Both are fixed before the first field challenge.

5.4 Field checks and the scalar linear handoff

The field circuit checks the following relation on the committed objects: all encodings are Boolean and canonical; padding and public coordinates are correct; the fresh records satisfy their strict and base-field flags; equations (3) hold; and $Q(z^*) = E^*$. It does not separately evaluate Q on every input. The Cauchy consistency argument in Section 6 recovers those equations together with $H = H_{\mathrm{src}}$.

Let $m_q = \lceil \log_2 q \rceil$ and $t = [\mathbb{K} : \mathbb{F}_q]$. Each field value is encoded by t canonical m_q-bit coefficients. For every coefficient, the compiler adds the comparison with q from Appendix B; Boolean constraints alone would also admit integers at least q.

The comparison helpers are Boolean variables over $\mathbb{F}_q$ and are not recursively encoded. Nonlinear gate outputs receive canonical $\mathbb{K}$ encodings in C_{aux}, while affine expressions are lowered directly to linear forms. The resulting sparse matrices satisfy

$$\mathcal{A}b \odot \mathcal{B}b - \mathcal{C}b = 0 \quad \text{over } \mathbb{K}, \tag{11}$$

where $b \in \mathbb{F}_q^{N_0}$ concatenates the committed segments, the fixed constant coordinate, and the reserved zero padding. The template and all dimensions are fixed in advance. The value of c changes public coefficients, not the shape of the circuit.

Pad the constraint tables with zero rows to 2^{ℓ_F} entries. Set $v_A = \mathcal{A}b$, $v_B = \mathcal{B}b$, and $v_C = \mathcal{C}b$. After all commitments are fixed, the verifier samples $r \in \mathbb{K}^{\ell_F}$. The parties run sumcheck on

$$\sum_{v \in \{0,1\}^{\ell_F}} \mathrm{eq}(r, v)\big(\widetilde{v}_A(v)\widetilde{v}_B(v) - \widetilde{v}_C(v)\big) = 0. \tag{12}$$

Each round polynomial has degree at most three. The prover sends its four coefficients before the verifier draws the next independent uniform challenge. At the final point $\tau \in \mathbb{K}^{\ell_F}$, the prover sends t_A, t_B, t_C. The verifier checks the final value $\mathrm{eq}(r, \tau)(t_A t_B - t_C)$.

The remaining claims are

$$\langle \mathcal{A}^{\mathsf{T}} \mathrm{eq}_\tau, b\rangle = t_A, \quad \langle \mathcal{B}^{\mathsf{T}} \mathrm{eq}_\tau, b\rangle = t_B, \quad \langle \mathcal{C}^{\mathsf{T}} \mathrm{eq}_\tau, b\rangle = t_C. \tag{13}$$

Because b has coordinates in $\mathbb{F}_q$, expansion in the fixed field basis gives $3t$ scalar equations. Adding all segment commitment equations, the constant-coordinate equation, and the zero-padding equations produces the explicit system $L_0 b = y_0$. Its honest squared-norm bound can be set to $S_0 = N_0$. The following reduction proves this scalar system without requiring a field embedding from $\mathbb{K}$ into R_q.

5.5 The recursive linear reduction

Consider a nonterminal layer and suppress its index where convenient. Put $\ell_i = \lceil \log_{\rho_i} q \rceil$ and $b_i^{\mathrm{dig}} = \rho_i/2$. The deterministic radix decomposition Dig_{ρ_i} produces ℓ_i signed digits per coefficient, each of absolute value at most b_i^{dig}. Its public linear recomposition map is Rec.

Commitment and projection. The prover computes $t_j = A_i w_j$ and forms the digit vector

$$\widehat{t} = \mathrm{Dig}_{\rho_i}(t_0, \ldots, t_{s_i - 1}).$$

It sends

$$u_1 = B_i \widehat{t}, \qquad B_i \in R_q^{a_{t,i} \times (\ell_i a_i s_i)}.$$

The verifier samples a fresh complete seed $d_i \in \{0,1\}^{\ell_i^{\mathrm{seed}}}$ and sends its canonical byte encoding. Both parties expand it to $\Pi = \Pi_i(d_i)$. Consecutive generated bit

pairs (u, v) encode the trit $u-v$, in row-major order. The prover sends a payload-free retry tag or commits to this attempt by sending

$$p = \mathrm{cen}(\Pi\,\mathrm{coef}(w)), \qquad \|p\|_2^2 \le m_i S_i. \tag{14}$$

There are at most $R_{P,i}$ attempts, all with the same u_1. Each retry uses a fresh complete seed. Sending p locks d_i, $\Pi_i(d_i)$, and p. A malformed descriptor, tag, or p, nonzero padding bits, or a violated norm bound causes rejection; it is not another retry.

Aggregation and symmetric data. After p is fixed, form $\widetilde{L} = (L_i^{\mathsf{T}}, \Pi^{\mathsf{T}})^{\mathsf{T}}$ and $\widetilde{y} = (y_i^{\mathsf{T}}, p^{\mathsf{T}})^{\mathsf{T}}$. The verifier samples independent uniform rows $\alpha^{(e)} \in \mathbb{F}_q^{\mathrm{rows}(\widetilde{L})}$ for $e = 1, \dots, \tau_i$, and sets

$$v^{(e)} = \alpha^{(e)}\widetilde{L}, \qquad b_e = \alpha^{(e)}\widetilde{y}.$$

Apply the constant-term lifting ι from Section 3 blockwise to these coefficient rows. This gives public vectors $\phi_j^{(e)} \in R_q^{n_i}$ such that

$$v^{(e)}\,\mathrm{coef}(w) = \mathrm{ct}\sum_j \langle \phi_j^{(e)}, w_j \rangle. \tag{15}$$

The prover forms the upper-triangular symmetric data

$$h_{ab}^{(e)} = \tfrac{1}{2}\big(\langle \phi_a^{(e)}, w_b \rangle + \langle \phi_b^{(e)}, w_a \rangle\big), \qquad 0 \le a \le b < s_i.$$

It commits to their radix digits as $u_2 = D_i\widehat{h}$, where

$$D_i \in R_q^{a_{h,i} \times (\ell_i \tau_i s_i (s_i+1)/2)}.$$

The scalar diagonal identities are $\mathrm{ct}\sum_j h_{jj}^{(e)} = b_e$. These identities concern constant coefficients, rather than equality of the whole ring sum with b_e.

Short response and child. Only after u_2 is fixed does the verifier sample independent $\boldsymbol{c}_j \leftarrow \mathcal{D}_i$. The honest response is $z = \mathrm{cen}(\sum_j \boldsymbol{c}_j w_j)$. The prover may request at most $R_{C,i} - 1$ further short challenges to obtain $\|z\|_2^2 \le G_i$, where $G_i \ge 2\mu_i S_i$. These retries keep u_1, Π, p, α, u_2 unchanged.

Once a response is selected, write $z = z_{\mathrm{lo}} + \rho_i z_{\mathrm{hi}}$ over the integers, with $\|z_{\mathrm{lo}}\|_\infty \le b_i^{\mathrm{dig}}$. The child witness is $x = (z_{\mathrm{lo}}, z_{\mathrm{hi}}, \widehat{t}, \widehat{h})$. Its public scalar linear system is obtained by expanding

$$B_i\widehat{t} = u_1, \qquad\qquad D_i\widehat{h} = u_2,$$

$$A_i(z_{\mathrm{lo}} + \rho_i z_{\mathrm{hi}}) = \sum_j \boldsymbol{c}_j\,\mathrm{Rec}(\widehat{t}_j), \tag{16}$$

$$\left\langle \sum_j \boldsymbol{c}_j \phi_j^{(e)}, z_{\mathrm{lo}} + \rho_i z_{\mathrm{hi}} \right\rangle = \sum_{a,b} \boldsymbol{c}_a \boldsymbol{c}_b\,\mathrm{Rec}(\widehat{h}_{ab}^{(e)}), \tag{17}$$

$$\mathrm{ct}\sum_j \mathrm{Rec}(\widehat{h}_{jj}^{(e)}) = b_e. \tag{18}$$

In an upper-triangular representation, each off-diagonal term in (17) has coefficient $2\boldsymbol{c}_a\boldsymbol{c}_b$. All coefficients are now public, so these equations are linear in x. Any padding added for the next block partition receives an explicit zero equation. The next layer proves this system with its prescribed norm bound. After the child is submitted, a rejecting suffix ends the attempt; it does not reopen the parent's response retry loop.

The raw child coefficient length and a sufficient honest energy bound are

$$N_{i+1}^{\mathrm{raw}} = 2dn_i + d\ell_i \left(a_i s_i + \tau_i \frac{s_i(s_i+1)}{2} \right), \tag{19}$$

$$S_{i+1} \geq \left\lceil dn_i (b_i^{\mathrm{dig}})^2 + \frac{2G_i + 2dn_i (b_i^{\mathrm{dig}})^2}{\rho_i^2} + E_{t,i} + E_{h,i} \right\rceil, \tag{20}$$

where $E_{t,i} = d\ell_i a_i s_i (b_i^{\mathrm{dig}})^2$ and $E_{h,i} = d\ell_i \tau_i s_i (s_i+1)(b_i^{\mathrm{dig}})^2/2$. Zero padding changes the length, but not this energy bound.

5.6 The pivot terminal

The last layer discloses its response and symmetric data directly. One inner commitment is reconstructed from that response. Let its main matrix be $A_L \in R_q^{a_L \times n_L}$ and put $t_j = A_L w_j$. Fix an even pivot radix $\rho_p < q$, let $\ell_p = \lceil \log_{\rho_p} q \rceil$, and sample an independent $B_p \in R_q^{a_p \times (a_L \ell_p)}$.

Before projection, the prover sends $t_1, \ldots, t_{s_L - 1}$ and $v_p = B_p \operatorname{Dig}_{\rho_p}(t_0)$. Projection and aggregation follow the preceding layer definition. The prover then sends the symmetric values $h_{ab}^{(e)}$ before the short challenge. For each e, it omits only $\mathrm{ct}(h_{00}^{(e)})$, which the verifier reconstructs as $b_e - \sum_{j \geq 1} \mathrm{ct}(h_{jj}^{(e)})$. All other coefficients of $h_{00}^{(e)}$ are transmitted.

The verifier samples $\boldsymbol{c}_0 \leftarrow \mathcal{D}_L \setminus \{0\}$ and $\boldsymbol{c}_j \leftarrow \mathcal{D}_L$ for $j \geq 1$. It uses the same fixed-prefix retry rule. On receiving z, it sets

$$t_0(\boldsymbol{c}, z) = \boldsymbol{c}_0^{-1} \left(A_L z - \sum_{j=1}^{s_L - 1} \boldsymbol{c}_j t_j \right). \tag{21}$$

The inverse exists by the short-challenge contract. The verifier checks

$$B_p \operatorname{Dig}_{\rho_p}(t_0(\boldsymbol{c}, z)) = v_p, \qquad \|z\|_2^2 \leq G_L,$$

and every principal equation (17), with z in place of its split representation. The response threshold is

$$G_L \geq 2\mu_L \frac{M_L}{M_L - 1} S_L. \tag{22}$$

The factor accounts for excluding the zero challenge in coordinate zero. This terminal uses A_L and B_p; it does not introduce B_L or D_L.

All field coefficients are parsed canonically before taking centered representatives or testing norms. The selected terminal codec sends fixed-length field coefficients. Retry tags have no payload. Appendix E specifies the resulting message grammar, including the verifier's independent random messages.

6 Security

The extractor works backwards through the reduction chain. At each boundary it either recovers a witness for the preceding relation or produces a short kernel for a specific commitment matrix. Two final comparisons recover the canonical field encoding and then the source quadratic relations.

6.1 A finite extract-or-short-kernel theorem

All probabilities in the statistical statement first refer to one fixed setup and one fixed complete pre-Cauchy checkpoint. A reset restores the prover's entire state at that checkpoint. New trials use fresh verifier and extractor coins. Every child invocation remains after its own checkpoint, so it cannot change a parent's fixed messages or challenges.

The projection analysis starts from the independent lazy-ternary bound in Appendix C.4. The protocol itself samples a complete seed and expands it to the projection matrix. For every vector fixed before that seed, the whole PASS predicate is a finite-state computation. The explicit generator parameters in Appendix C.5 give

$$|\Pr[\mathrm{PASS}_w(\Pi_{\mathrm{seed}})] - \Pr[\mathrm{PASS}_w(\Pi_{\mathrm{iid}})]| \le \eta, \qquad \eta = 2^{-150}. \tag{23}$$

This is a pointwise statistical statement. It neither treats the public seed as hidden nor asserts indistinguishability for an arbitrary interactive computation.

For a layer with $q \ge 128\gamma_i$, define

$$\theta_P = \frac{7}{8} + 2^{-1000}, \qquad \epsilon_{P,i} = \left(\frac{3\sigma_i^2}{3\sigma_i^2 - 1}\theta_P\right)^{m_i}, \tag{24}$$

$$\epsilon_i = R_{P,i}\epsilon_{P,i} + q^{-\tau_i} + 3R_{C,i}\sum_{j=0}^{s_i-1}\frac{1}{M_{ij}}, \tag{25}$$

where $M_{ij} = M_i$ except that $M_{L0} = M_L - 1$ at the pivot terminal. The stated projection bound already absorbs η. Direct substitution gives

$$\left(\frac{97}{95}\left(\frac{7}{8} + 2^{-200}\right)\right)^{864} + 2^{-150} < \left(\frac{48}{47}\cdot\frac{7}{8}\right)^{864} \le \epsilon_{P,i}$$

for every layer in the concrete schedule. The generator error is therefore not added to κ_{alg} a second time. The node's statistical error is

$$\kappa_{\mathrm{alg}} = \frac{4\ell_F}{|\mathbb{K}|} + \frac{2k}{|\mathcal{C}|} + \sum_{i=0}^{L}\epsilon_i. \tag{26}$$

The specified main and auxiliary kernel radii at a nonterminal layer are

$$\beta_{A_i} = \left\lceil 8T_i^{\mathrm{op}}\sqrt{1+\rho_i^2}\,\gamma_{i+1}\right\rceil, \qquad \beta_{B_i} = \beta_{D_i} = \lceil 2\gamma_{i+1}\rceil. \tag{27}$$

At the terminal they are

$$\beta_{A_L} = \lceil 8T_L^{\mathrm{op}}\sqrt{G_L}\rceil, \qquad \beta_{B_p} = \left\lceil \rho_p \sqrt{d a_L \ell_p} \right\rceil. \tag{28}$$

Each of $A_{\mathrm{st}}, A_H, A_{\mathrm{aux}}$ has comparison radius $\lceil 2\gamma_0 \rceil$. These radii apply when openings have only the backend's extracted norm bound. Their values cannot be reduced by anticipating the later Booleanity check.

Theorem 2 (Finite-node extraction). *Use the seeded-projection protocol in Section 5. Assume the field, encoding, and challenge contracts there; $q \geq 128\gamma_i$ at every layer; and the child length and energy conditions* (19)–(20)*. For every fixed pre-Cauchy checkpoint s of a classical prover, let $p(s)$ be the acceptance probability of the actual continuation. There is a resettable extractor that returns one of the following with probability at least $\max\{0, p(s) - \kappa_{\mathrm{alg}}\}$:*

1. *canonical openings of all source commitments, satisfying their specified accumulator or strict fresh relations, together with the carrier consistency equation and an output opening on one actual accepting Cauchy replay;*
2. *a specified matrix A_j and an integer ring vector v such that $A_j v = 0 \bmod q$, $v \bmod q \neq 0$, and $\|v\|_2 \leq \beta_j$.*

For a prover whose continuations have the required uniform conditional cost bounds, the extractor has the expected-time bound below. For fixed internal depth and polynomial-size parameters, these bounds are polynomial. Honest execution fails with probability at most

$$\epsilon_{\mathrm{comp}} \leq \sum_{i=0}^{L} \left(2^{-R_{P,i}} + 2^{-R_{C,i}}\right). \tag{29}$$

The expected-time bound records actual work, including full-seed sampling and parsing, streaming projection expansion, state restoration, arithmetic, and verification. Let H_i uniformly bound the conditional cost of preparing and running one layer-i continuation, excluding the child extractor. Let W_i cover the bounded projection and aggregation preparation, final comparisons, and witness checks. A sufficient recurrence is

$$T_i^{\mathrm{ext}} \leq W_i + 2R_{C,i}(1 + 2s_i)(H_i + T_{i+1}^{\mathrm{ext}}), \qquad i < L. \tag{30}$$

At $i = L$, direct terminal verification replaces the child cost, giving

$$T_L^{\mathrm{ext}} \leq W_L + 2R_{C,L}(1 + 2s_L)H_L.$$

The field and Cauchy wrappers satisfy

$$T_F^{\mathrm{ext}} \leq 2(H_F + T_0^{\mathrm{ext}}) + W_F^{\mathrm{ext}}, \qquad T_{\mathrm{node}}^{\mathrm{ext}} \leq 2(H_C + T_F^{\mathrm{ext}}) + W_C^{\mathrm{ext}}. \tag{31}$$

Here H_F, H_C include their respective prover continuations, and $W_F^{\mathrm{ext}}, W_C^{\mathrm{ext}}$ include the corresponding complete comparisons and semantic checks. A bound on oracle calls alone is not substituted for these quantities.

The proof has three levels. The linear chain first returns a bounded opening of $L_0 b = y_0$. The field wrapper checks that this opening is the intended canonical encoding. The Cauchy wrapper then checks the source relations. We detail these steps next; the sampling and norm proofs are collected in Appendix C.

6.2 Local recovery from three coordinate challenges

We first recover algebraic candidates, following the reduction-of-knowledge viewpoint [KP23], and certify their norms in the next step. Fix a short-challenge checkpoint, including u_1, u_2, the chosen projection, and the aggregation rows. A successful child extraction supplies the child linear equations and an opening with norm at most γ_{i+1}. Recomposition therefore bounds its response by

$$\|z\|_2 \le Z_i^{\mathrm{ext}} := \sqrt{1+\rho_i^2}\,\gamma_{i+1}. \tag{32}$$

This is the bound used in extraction, rather than the honest threshold $\sqrt{G_i}$. The terminal directly enforces $Z_L^{\mathrm{ext}} = \sqrt{G_L}$.

Retain one successful center $(\boldsymbol{c}, z)$. For coordinate j, obtain two further successful continuations that change only $\boldsymbol{c}_j$, using distinct values. If their auxiliary openings disagree modulo q with the center, subtract the two openings of u_1 or u_2 and return a B_i- or D_i-kernel. Otherwise t and h are fixed across the three continuations. Write

$$\Delta_b = \boldsymbol{c}_j^{(b)} - \boldsymbol{c}_j, \qquad X_b = z^{(b)} - z, \qquad w_j^{(b)} = X_b \Delta_b^{-1}, \quad b \in \{1,2\}.$$

The commitment equation gives $A_i X_b = t_j \Delta_b$. If the recovered slopes differ, the integer vector

$$v = X_1 \Delta_2 - X_2 \Delta_1 \tag{33}$$

is a nonzero modular kernel. Its norm is at most $8T_i^{\mathrm{op}} Z_i^{\mathrm{ext}}$. Only short numerators and short challenge differences are lifted to the integers. The modular inverse used to define a slope need not have a short integer lift. This is the compare-before-clearing principle of [WF26] in the present interface.

If the slopes agree, call the result w_j. Replace the jth challenge by a formal variable T in the principal equation, and replace the response by $z + (T - \boldsymbol{c}_j) w_j$. The resulting polynomial has degree at most two and vanishes at all three chosen values. Their pairwise differences are units, so its Vandermonde matrix is invertible over R_q. The quadratic coefficient gives

$$\langle \phi_j^{(e)}, w_j \rangle = h_{jj}^{(e)}, \qquad A_i w_j = t_j.$$

Summing the diagonal equations recovers all actual scalar aggregation equations. This argument needs those diagonal identities; it does not require a separate assertion that every recovered response equals $\sum_j \boldsymbol{c}_j w_j$.

The coordinate sampling procedure keeps the correct original challenge distribution even when continuation success depends on the challenge. It loses at

most $3\sum_j M_{ij}^{-1}$ success mass and makes at most $1+2s_i$ oracle calls in expectation. Accounting for the bounded response attempts gives $3R_{C,i}\sum_j M_{ij}^{-1}$ in (25). At the pivot terminal, disagreement in reconstructed t_0 instead produces a B_p-kernel: its canonical digit openings have length $da_L\ell_p$ and differ coefficientwise by at most ρ_p. If they agree, the same three-point argument applies.

6.3 Certifying a recovered candidate

The slopes just recovered satisfy the sampled aggregation equations. They need not yet satisfy the original linear system or its norm bound. To certify them, return to the complete checkpoint before projection, where u_1 is fixed, and run a fresh recovery trial.

For a fixed candidate w with $L_i\,\mathrm{coef}(w) \neq y_i$, the augmented residual is nonzero for every projection response. Independent aggregation rows chosen after that response all miss it with probability at most $q^{-\tau_i}$. If w satisfies the linear system but $\|w\|_2 > \gamma_i$, an incorrect projection response is covered by the same aggregation bound. A correct projection response must pass the fixed-vector test. Its probability over at most $R_{P,i}$ fresh complete seeds is at most $R_{P,i}\epsilon_{P,i}$. Thus a fresh recovery returns any specified bad w with probability at most

$$\delta_i^{\mathrm{ret}} = q^{-\tau_i} + R_{P,i}\epsilon_{P,i}. \tag{34}$$

This fixed-candidate bound can be used although the first recovered candidate was obtained after seeing projection randomness. Run one raw trial. On a successful candidate, independently repeat the same raw trial until it returns a candidate or kernel. Incompatible candidates yield a kernel. Compatible candidates are accepted only after their original linear relation and norm bound are checked. If the raw success probability is $s > 0$, the second sample's bad compatible probability is at most $\delta_i^{\mathrm{ret}}/s$; the first successful trial supplies the factor at most s. Hence this comparison loses at most δ_i^{ret}, not that quantity divided by an acceptance probability. Its expected trial cost is at most twice the raw cost.

The comparison respects commitment lifetimes. Across new projection trials, u_1 remains fixed. Different $\widehat{t}$ openings therefore give a B_i-kernel. Equal $\widehat{t}$ but different recovered slopes give (33). The commitment u_2 may change after a new projection and aggregation, so its $\widehat{h}$ openings are not compared across that boundary. At the terminal, the fixed pivot commitment replaces the u_1 comparison.

6.4 Finite composition and acceptance conditioning

Put $\Lambda_i = \sum_{h=i}^{L}\epsilon_h$. The terminal supplies the base case by direct verification. Suppose the child extractor succeeds with probability at least $p_t - \Lambda_{i+1}$ at every fixed child checkpoint t. An actual forward execution submits at most one child proposal: once submitted, a suffix rejection cannot trigger another parent retry. Averaging over the chosen checkpoint therefore subtracts Λ_{i+1} only once.

Coordinate recovery then uses the actual success probabilities of this oracle. It does not subtract a new copy of Λ_{i+1} for every fork.

Combining local recovery with (34) gives success at least $p_i - \Lambda_i$ at layer i. The induction preserves the witness relation, its extracted norm, the role and radius of every kernel, and one actual accepting continuation from the center branch. This last record identifies the output commitment and challenge to which the final witness belongs.

The same induction establishes the conditional cost bounds used in (30). If C_n is the cost of call n and $\mathcal{F}_{n-1}$ is the prior history, the required bound is $\mathbb{E}[C_n \mid \mathcal{F}_{n-1}] \le H_i + T_{i+1}^{\mathrm{ext}}$ for every reached checkpoint. Tonelli's theorem then bounds the expected sum by this uniform bound times the expected number of calls. It does not require a call's duration and success to be independent. For a growing depth, the product $\prod_i 2R_{C,i}(1+2s_i)$ and the associated weighted costs must be bounded explicitly. The concrete instance has fixed depth.

6.5 From linear openings to source witnesses

Fix the post-Cauchy commitments. One raw field trial samples fresh field coins, runs sumcheck, and invokes the linear-chain extractor. It returns a kernel or an opening b of $L_0 b = y_0$ with $\|b\|_2 \le \gamma_0$. For any fixed b violating (11), the random equality test misses its nonzero residual with probability at most $\ell_F/|\mathbb{K}|$. Conditional on a nonzero initial claim, the degree-three sumcheck loses at most $3\ell_F/|\mathbb{K}|$ [LFKN92]. The terminal linear equations bind its final three evaluations. The fixed-bad-opening return probability is thus at most $4\ell_F/|\mathbb{K}|$.

Apply the preceding two-trial comparison at this checkpoint. If the openings differ, some committed segment differs: the uncommitted constant and padding coordinates are already fixed by $L_0 b = y_0$. The difference of the two segment openings is a kernel for its corresponding matrix, with radius at most $2\gamma_0$. If they agree, check the full field relation and decoding directly. This gives canonical source, carrier, output, and auxiliary values with the post-Cauchy semantics.

One more comparison moves to the pre-Cauchy checkpoint. Across its independent trials, only source and carrier openings are compared; output and auxiliary commitments are allowed to depend on c. Incompatible source or carrier openings give an A_{st}- or A_H-kernel. For a fixed compatible tuple, put $\delta_i = Q(z_i) - E_i$ and $P_i(T) = D(T)/(T - \xi_i)$. Its accepted fold implies that c is a root of

$$F(T) = D(T)^2\delta_0 + \sum_{i=1}^{k} \lambda_i^2 P_i(T)^2 \delta_i + D(T)(H_{\mathrm{src}}(T) - H(T)). \tag{35}$$

The polynomial is identically zero exactly when every source relation holds and $H = H_{\mathrm{src}}$. Indeed, evaluation at ξ_i isolates δ_i for $i \ge 1$. Once these vanish, the coefficient of T^{2k} isolates δ_0. The remaining identity forces $H = H_{\mathrm{src}}$. Otherwise a nonzero scalar coordinate has degree at most $2k$, giving a return probability at most $2k/|\mathcal{C}|$. The final comparison proves Theorem 2.

Cauchy weights occur only in field equations and public coefficients. The integer kernel bounds use encoded lengths, digit energies, short ring challenges, and bounded responses. No integer norm bound on $(c-\xi_i)^{-1}$ is needed.

6.6 Averaging over setup and reducing each role

Let $A_1, \dots, A_J$ be the complete set of independently sampled matrices. Repeated uses of A_{st} constitute one role. Associate $\vartheta_j = (q, d, a_j, b_j, \beta_j)$ with coefficient ℓ_2 Module-SIS and the nonzero-modulo-q condition of Theorem 2. Let $\bar{p}_N$ be the seeded protocol's actual acceptance probability averaged over setup and adversarial initialization, and let P_N bound the extractor's entire expected running time under the same distribution.

Theorem 3 (Role-specific computational reduction). *Assume the product setup above and the hypotheses of Theorem 2. For a global time budget B_N, truncate the complete extractor once after B_N steps, obtaining E_{B_N}. For each role j, let $S_{j,N}^{\max}$ bound all extra work needed to embed a Module-SIS challenge, generate the other setup coordinates, initialize the prover, and return a checked kernel. In a computation model with exact uniform finite-alphabet sampling,*

$$\begin{aligned}&\Pr[E_{B_N} \textit{ returns source witnesses}]\\ &\geq \max\left\{0, \bar{p}_N - \kappa_{\mathrm{alg}} - \frac{P_N}{B_N} - \sum_{j=1}^{J} \mathrm{Adv}_{\vartheta_j}^{\mathrm{MSIS}}(B_N + S_{j,N}^{\max})\right\}. \end{aligned} \tag{36}$$

For bounded implementations of setup sampling from unbiased bits, subtract additionally $\sum_j \eta_{\mathrm{set},j}$, where $\eta_{\mathrm{set},j}$ is the declared sampling-failure bound for the extra setup in reduction j.

Proof. The statistical theorem holds at every fixed checkpoint with the same error bound. Averaging it over the actual setup and initialization distribution gives $\bar{p}_N - \kappa_{\mathrm{alg}}$. Markov's inequality bounds the loss from the one global truncation by P_N/B_N.

For each fixed j, a separate reduction places its external Module-SIS matrix in position j and samples all other matrices independently. It then generates the remaining public data and adversarial state exactly as in the node experiment. This has the same product-setup distribution. If E_{B_N} outputs a role-j kernel, the reduction returns that vector. Its complete time is bounded by $B_N + S_{j,N}^{\max}$. Summing the resulting bounds on the disjoint labelled kernel events proves the claim. The finite-bit version follows by coupling exact setup with its bounded sampler.

No reduction guesses a random role, so there is no $1/J$ guessing factor. The union cost remains: the sum in (36) can be bounded by $J \max_j \mathrm{Adv}_{\vartheta_j}^{\mathrm{MSIS}}(B_N + S_{j,N}^{\max})$, not by the maximum alone. Appendix D makes the setup and runtime accounting explicit.

This theorem concerns a classical interactive node and a classical resettable prover. The concrete attack estimates in Section 7 evaluate the specified instances under specified lattice-reduction models. They do not replace the role-specific advantages or the reduction's time and probability terms in (36).

7 Efficiency and Concrete Instantiation

A finite schedule trades prover work for communication. We first bound this tradeoff, then give the dimensions, operation counts, and message lengths under the fixed serialization grammar for one compiled $k = 16$ instance.

7.1 Cost model and finite schedules

Let $N = N_0^{\mathrm{raw}}$ be the initial coefficient-witness length. Write W_F for the cost of constructing and applying the sparse field relation, including its public terminal transposes, with $W_F \geq M_c$. Let C_H be the carrier-construction cost from Theorem 1. The operations in W_F and C_H retain their field types. Ring operations below are over R_q, and the dependence on the field and ring parameters is included in the security-parameter factors.

At layer i, pad to $N_i = ds_i n_i$ coefficients. The next raw length is

$$N_{i+1}^{\mathrm{raw}} = \frac{2N_i}{s_i} + d\ell_i \left(a_i s_i + \tau_i \frac{s_i(s_i+1)}{2} \right). \tag{37}$$

The first term contains the two-part response split. The second contains commitment digits and symmetric auxiliary data. It is quadratic in the number of blocks s_i, even when the semantic carrier has only linear arity dependence. The block count s_i is independent of the outer folding arity k.

One nonterminal layer performs

$$O\left(a_i s_i n_i + \tau_i s_i^2 n_i + a_{t,i}\ell_i a_i s_i + a_{h,i}\ell_i \tau_i s_i^2\right)$$

ring work, together with the projection and public aggregation operations. The child relation has structured apply and transpose routines: ring matrix products, radix recomposition, public inner products, and linear combinations of symmetric data. No dense scalar matrix is needed to obtain these bounds.

Theorem 4 (Finite-schedule efficiency). *Fix $0 < \nu \leq 1/3$ and an integer $L \geq 0$, both independent of the security parameter and N. Choose*

$$s_i = \max\{3, \lceil (N_i^{\mathrm{raw}})^{\nu} \rceil\}, \qquad n_i = \left\lceil \frac{N_i^{\mathrm{raw}}}{ds_i} \right\rceil$$

for L nonterminal layers, and a block count of the same order at the terminal layer. Suppose this schedule satisfies the norm, modulus, challenge, and error conditions of Section 6. Assume that the remaining public dimensions and

retry caps are polynomial in the security parameter, and that $\log q = \mathrm{poly}(\lambda) + O(\log N)$*. For the fixed-width terminal codec and* $\gamma = (1-\nu)^{L+1}$*,*

$$T_P = \widetilde{O}_\kappa(W_F + C_H + N^{1+\nu}), \qquad B^{\mathrm{fold}}_{P\to V} = \widetilde{O}_\kappa(N^\gamma + 1), \tag{38}$$

$$T_V = \widetilde{O}_\kappa(W_F + N), \qquad B_{V\to P} = \widetilde{O}_\kappa(N). \tag{39}$$

Here $\widetilde{O}_\kappa$ *hides polynomial security-parameter factors and logarithms, not powers of* k*,* s_i*, or* N*. Incoming fresh commitments contribute a separate* $O(ka_{\mathrm{st}}d\log q)$ *bits.*

Proof. Equation (37) and ceiling padding give

$$N^{\mathrm{raw}}_{i+1} = \widetilde{O}_\kappa\big(N_i^{1-\nu} + N_i^{2\nu} + 1\big) = \widetilde{O}_\kappa(N_i^{1-\nu} + 1).$$

The second equality uses $2\nu \le 1-\nu$. For fixed L, iteration gives $N^{\mathrm{raw}}_i = \widetilde{O}_\kappa(N^{(1-\nu)^i} + 1)$. The largest witness-dependent ring term is proportional to s_iN_i, and therefore has exponent $1+\nu$. The field sumcheck folds tables whose lengths halve each round, so their total work is $O(M_c)$, in addition to W_F.

Each nonterminal layer sends two commitments, one selected projection response, and retry tags. The terminal sends $O(N_L/s_L + \mathrm{poly}(\lambda, \log N)(s_L + s_L^2))$ coefficients. Its leading exponent is $1-\nu$, yielding γ. Each retry is payload-free until a projection or response is selected. The verifier sends a complete projection seed at each attempt and keeps the aggregation randomness explicit. For the schedule above, the latter has total length linear in the source relation. Structured public transposes, seed expansion, and field work give the verifier time bound.

The theorem describes admissible schedule families. The concrete instance below uses an explicitly selected schedule, not a claim that these finite block counts optimize the theorem's asymptotic exponent. Likewise, the fast carrier theorem concerns C_H. It does not remove the symmetric-data work in the lattice layers.

7.2 One compiled quadratic relation

The instance has four public dot-product equations. Let

$$z = (u, x_0, x_1, x_2, x_3, (a_{j,\ell}, b_{j,\ell})_{0\le j<4,\ 0\le \ell<8}) \in \mathbb{K}^{69}$$

and define

$$Q_j(z) = \sum_{\ell=0}^{7} a_{j,\ell}b_{j,\ell} - ux_j \qquad (0 \le j < 4).$$

Fresh records have $u = 1$, zero residual, and base-field coordinates. Thus their public values x_j constrain the witnesses; a freely chosen residual cannot absorb an invalid dot product. The relation serves to instantiate the mathematical interface, rather than to represent an application workload.

Table 3. The single concrete instance. All layers use the common projection, radix, and rank parameters above. The last row of the schedule is terminal.

(a) Global parameters

Parameter	Value
q; d; $[\mathbb{K} : \mathbb{F}_q]$	$2^{48} - 59$; 64; 4
Extension polynomial	$\theta^4 - 4\theta^2 + 2$
Arity; semantic dimensions	$k = 16$; $(n, y, r) = (69, 4, 4)$
Cauchy poles; scales	$0, \ldots, 15$; all $\lambda_i = 1$
State / carrier / auxiliary columns	219 / 192 / 4 350
Front / main / auxiliary / pivot ranks	32 / 24 / 16 / 4
Projection (m, σ); aggregate rows τ	$(864, 4)$; 3
Short alphabet; norm bound	$\{-2, \ldots, 2\}^{64}$; 128
Radix and digits; pivot radix and digits	$(64, 8)$; $(8, 16)$
Projection / response retry caps	160 / 160
Field rows / padded rows / rounds	1 351 938 / 2^{21} / 21
Handoff capacity; occupied coefficients	2^{20}; 542 977
Canonical CRS storage	155 814 912 bytes

(b) Reduction schedule

i	s_i	n_i	N_i	S_i	G_i
0	9	1 821	1 048 896	1 048 576	268 435 456
1	6	1 075	412 800	303 555 488	77 710 204 928
2	5	762	243 840	216 957 652	55 541 158 912
3	4	711	182 016	163 590 043	41 879 051 008
4	4	608	155 648	133 127 892	34 080 740 352
5	7	318	142 464	122 566 619	31 377 054 465

For each j, $B(e_{a_{j,0}}, e_{b_{j,0}}) = e_j$. Hence $n = 69$, $y = r = 4$, and the mixed-output basis maps are identities. The carrier consists of $16 \cdot 4 = 64$ elements of $\mathbb{K}$. Table 3 fixes the field, ring, commitments, and complete six-layer schedule.

Every $\mathbb{K}$-value uses four canonical 48-bit coefficients. The compiler allocates 14,016 bits per state, 12,288 carrier bits, and 278,400 auxiliary bits. The auxiliary segment contains 36 field multiplication values and 271,488 Boolean comparison helpers. Minimal packing into degree-64 rings gives 219, 192, and 4,350 columns for $A_{\mathrm{st}}, A_H, A_{\mathrm{aux}}$. The shared state matrix has eighteen logical uses, but remains one random matrix.

Including the constant coordinate, the field witness occupies 542,977 coefficients of a 2^{20}-coefficient handoff. The other 505,599 coordinates are constrained to zero. The compiler emits 1,351,938 R1CS rows, padded with zero constraints to $M_c = 2^{21}$, so $\ell_F = 21$. The matrix shapes are independent of the Cauchy challenge; only designated public coefficients depend on it. Appendix B gives the exact row categories, bit order, and segment boundaries.

Table 4. Statistical bounds and attack-model summary. The first six rows form the statistical error account; the next two record honest retry failure and the matrix count. The final two rows are generic lattice attack cost exponents, not a bound on the full-node computational advantage.

Quantity	Value
Field check	$84/q^4$
Cauchy consistency	$32/(q^4-16)$
Modular projection	$960[(48/47)(7/8+2^{-1000})]^{864}$
Linear aggregation	$6/q^3$
Coordinate extraction	$480[34/5^{64}+1/(5^{64}-1)]$
Statistical exponent	$-\log_2 \kappa_{\mathrm{alg}} \simeq 130.22218$
Honest retry failure	$12 \cdot 2^{-160} < 2^{-156}$
Independent commitment matrices	20
Weakest classical attack estimate	A_0: 255.49
Weakest quantum attack estimate	A_0: 238.73

7.3 Statistical bounds and Module-SIS estimates

Put $M = 5^{64}$. Substituting the actual field depth and schedule into the statistical theorem gives the exact rational upper bound

$$\begin{aligned}\kappa_{\mathrm{alg}} = \frac{84}{q^4} + \frac{32}{q^4-16} + 960\left[\frac{48}{47}\left(\frac{7}{8}+2^{-1000}\right)\right]^{864} \\ + \frac{6}{q^3} + 480\left(\frac{34}{M}+\frac{1}{M-1}\right).\end{aligned} \tag{40}$$

Thus $-\log_2 \kappa_{\mathrm{alg}} \approx 130.22218$. The projection term dominates this bound. The honest retry-exhaustion probability is at most $12 \cdot 2^{-160}$. Both calculations use exact integers and rational numbers; the displayed logarithms are derived approximations. The seeded projection changes neither displayed bound. Its error is absorbed by the slack already present in the one-row projection estimate, as shown in Appendix C.5.

There are twenty independent commitment matrices: three front-end matrices, three at each of five nonterminal layers, and two at the terminal. Appendix D gives the dimensions and radii. The computational theorem uses precisely this matrix set.

For each role (a, b, β), the attack estimate uses coefficient dimensions $n_{\mathrm{SIS}} = 64a$, $m_{\mathrm{SIS}} = 64b$, modulus q, and Euclidean length bound β. We use the official lattice estimator [A+26] with the configuration in Appendix D.8. We use its SIS lattice attack with classical and quantum cost models. The classical model follows the MATZOV report [MAT22]; the quantum model uses the estimator's near-neighbor option. The weakest role in both models is A_0, with BKZ block size 797 and cost exponents 255.49 and 238.73, respectively.

These are coefficient-lattice estimates; the selected model does not cover additional attacks that exploit module structure. Fifteen roles use block sizes above

Table 5. Concrete costs. Communication uses one successful attempt at each retry point and excludes the separately listed incoming objects. Operation rows count distinct components and retain their units.

(a) Exact syntax-level communication (bytes)

Object / direction	Bytes
Incoming fresh commitments (16)	196 608
Incoming accumulator commitment	12 288
Fold, prover to verifier	342 690
Fold, verifier to prover	10 341 123
Total interactive fold	10 683 813

(b) Selected typed operation counts

Component	Unit	Count
Pole-residue carrier	$\mathbb{K} \times \mathbb{K}$ products	1,152
Pole-residue carrier	$\mathbb{K} \times \mathbb{F}_q$ products	41,995
Pole-residue carrier	$\mathbb{K}$ additions/subtractions	46,720
Sparse field applications	$\mathbb{K} \times \mathbb{F}_q$ products	4 022 047
Field sumcheck	$\mathbb{K} \times \mathbb{K}$ products	29 360 114
Field sumcheck	$\mathbb{K}$ additions/subtractions	35 651 567
One projection scan at each layer	trits expanded / inspected	1 888 413 696

1,024 and are marked as extrapolations in the complete table. The security theorem retains the sum of role-specific advantages and its time loss.

7.4 Communication

The grammar uses six bytes per base-field coefficient, 24 bytes per extension-field element, and 384 bytes per ring element. Projection seed bits are packed least-significant bit first with zero high padding; short-challenge coefficients use three bits each. Messages are separately byte-aligned. Each retry attempt contributes one tag byte. Table 5 reports the no-retry case, with incoming commitments shown separately.

The prover-to-verifier fold comprises 36,864 bytes of new commitments, 2,088 field-message bytes, and 303,738 lattice and terminal bytes including tags. The six complete projection seeds occupy 52,557 bytes in the no-retry transcript. The aggregation randomness remains explicit and occupies 10,286,694 bytes. Table 6 isolates the effect of replacing the projection matrices; all other messages are unchanged.

For an accepted execution reaching all 160-attempt caps, the syntax bounds are 344,598 bytes from prover to verifier and 18,831,246 bytes in the other direction. These are maximum accepted-attempt lengths, not expected lengths. The ordinary no-retry fold totals 10,683,813 bytes, or 10.19 MiB, of interactive communication. Transporting the sixteen fresh commitments adds 196,608 bytes; transporting the accumulator as well adds 12,288 bytes. The public statement

Table 6. Exact communication change from explicit to seeded projections, in bytes. All other protocol messages and the six-layer schedule are fixed.

Quantity	Explicit projections	Seeded projections
Projection descriptions	472 103 424	52 557
Fold $P \to V$	342 690	342 690
Fold $V \to P$	482 391 990	10 341 123
Interactive total	482 734 680	10 683 813
Maximum accepted syntax	75 547 314 564	19 175 844

and setup are prior inputs to this grammar. Appendix E supplies the per-layer formulas.

7.5 Concrete operation counts

For this relation, a Q-evaluation uses 36 extension-field products and 32 additions or subtractions. A B-evaluation uses 72 products and 68 additions or subtractions. The streaming reference evaluates 136 source pairs. The pole-residue algorithm uses sixteen B-evaluations; including its polynomial work, its online count is 1,152 extension-field products, 41,995 extension-by-base-field products, and 46,720 extension-field additions or subtractions. The multipoint alternative uses sixteen Q-evaluations when source Q-values are available. Appendix E compares all three algorithms under the same reference polynomial-arithmetic model.

The field sumcheck uses 29,360,114 extension-field products and 35,651,567 additions or subtractions in the counted table loops. Its sparse matrix applications and public transposes are additional costs. The verifier performs those public transposes and expands the projection seeds. Across the six layers, one no-retry execution uses 9,428,985 prefix-hash evaluations for the prover's projection tests and the two parties' selected projection adjoints. The generator's binary convolution operations are reported separately in Appendix E; they are not converted into field or ring operations.

The operation table separates full ring products, bounded-coefficient products, field operations, projection additions, and coordinate copies.

The compiler, parameters, and supporting code are available in the accompanying repository.[1] The reported counts follow from the relation, matrix layouts, and parameters described above.

These counts apply to the specified relation, seeded-projection profile, and internal schedule. They describe arithmetic and message lengths; elapsed times are not measured here.

[1] `https://github.com/cxcx52/cauchyfold`

8 Conclusion

CauchyFold gives a committed lattice realization of the Cauchy carrier representation. Fast multipoint and pole-residue algorithms compute the carrier without enumerating its full pair table. Canonical encoding preserves a bounded commitment interface while the semantic fold takes place over an extension field. The resulting checks reduce to explicit linear lattice relations.

Extraction follows these interfaces back to the sources. Compatible recoveries yield valid openings; incompatible ones yield a short kernel for a specified commitment matrix. The concrete $k = 16$ instance fixes all twenty such matrices and accounts for statistical error, communication, and typed arithmetic.

The remaining efficiency question is how to carry the carrier's favorable arity dependence through the rest of the construction. Seeded projections remove the largest explicit-randomness message, while aggregation randomness and symmetric lattice work remain substantial. Improving those components without changing their extraction interfaces would bring semantic arity efficiency closer to end-to-end computational efficiency.

References

A$^+$26. M. R. Albrecht et al. Security estimates for lattice problems. Sage software, 2026.

Ajt96. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing (STOC 1996)*, pages 99–108. ACM, 1996.

AKLY24. T. Attema, M. Klooß, R. W. F. Lai, and P. Yatsyna. Adaptive special soundness: A new framework for knowledge extraction with applications to Kilian's protocol. Cryptology ePrint Archive, Paper 2024/2038, 2024.

ALS20. T. Attema, V. Lyubashevsky, and G. Seiler. Practical product proofs for lattice commitments. Cryptology ePrint Archive, Paper 2020/517, 2020.

APS15. M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of Learning with Errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

BBB$^+$17. B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. Cryptology ePrint Archive, Paper 2017/1066, 2017.

BC23. B. Bünz and B. Chen. Protostar: Generic efficient accumulation/folding for special-sound protocols. In *Advances in Cryptology—ASIACRYPT 2023, Part II*, *Lecture Notes in Computer Science* 14439, pages 77–110. Springer, 2023.

BC25. D. Boneh and B. Chen. LatticeFold+: Faster, simpler, shorter lattice-based folding for succinct proof systems. In *Advances in Cryptology—CRYPTO 2025, Part VII*, *Lecture Notes in Computer Science* 16006, pages 327–361. Springer, 2025.

BC26. D. Boneh and B. Chen. LatticeFold: A lattice-based folding scheme and its applications to succinct proof systems. In *Advances in Cryptology—ASIACRYPT 2025, Part III*, *Lecture Notes in Computer Science* 16247, pages 330–362. Springer, 2026.

BCFW25. B. Bünz, A. Chiesa, G. Fenzi, and W. Wang. Linear-time accumulation schemes. Cryptology ePrint Archive, Paper 2025/753, 2025.

BCMS20. B. Bünz, A. Chiesa, P. Mishra, and N. Spooner. Recursive proof composition from accumulation schemes. In *Theory of Cryptography—TCC 2020, Part II*, *Lecture Notes in Computer Science* 12551, pages 1–18. Springer, 2020.

BCS21. J. Bootle, A. Chiesa, and K. Sotiraki. Sumcheck arguments and their applications. Cryptology ePrint Archive, Paper 2021/333, 2021.

BCS23. J. Bootle, A. Chiesa, and K. Sotiraki. Lattice-based succinct arguments for NP with polylogarithmic-time verification. Cryptology ePrint Archive, Paper 2023/930, 2023.

BDFG20. D. Boneh, J. Drake, B. Fisch, and A. Gabizon. Halo infinite: Recursive zk-SNARKs from any additive polynomial commitment scheme. Cryptology ePrint Archive, Paper 2020/1536, 2020.

BDL$^+$16. C. Baum, I. Damgård, V. Lyubashevsky, S. Oechsner, and C. Peikert. More efficient commitments from structured lattice assumptions. Cryptology ePrint Archive, Paper 2016/997, 2016.

BGH19. S. Bowe, J. Grigg, and D. Hopwood. Recursive proof composition without a trusted setup. Cryptology ePrint Archive, Paper 2019/1021, 2019.

BLNS20. J. Bootle, V. Lyubashevsky, N. K. Nguyen, and G. Seiler. A non-PCP approach to succinct quantum-safe zero-knowledge. Cryptology ePrint Archive, Paper 2020/737, 2020.

BLS03. A. Bostan, G. Lecerf, and É. Schost. Tellegen's principle into practice. In *Proceedings of ISSAC 2003*, pages 37–44. ACM, 2003.

BMNW24. B. Bünz, P. Mishra, W. Nguyen, and W. Wang. Arc: Accumulation for reed–solomon codes. Cryptology ePrint Archive, Paper 2024/1731, 2024.

BNP26. D. Balbás, A. Nitulescu, and M. Plançon. ProtogaLattice: Lattice-based algebraic folding. Cryptology ePrint Archive, Paper 2026/1317, 2026.

BS05. A. Bostan and É. Schost. Polynomial evaluation and interpolation on special sets of points. *Journal of Complexity*, 21(4):420–446, 2005.

BS22. W. Beullens and G. Seiler. LaBRADOR: Compact proofs for R1CS from module-SIS. Cryptology ePrint Archive, Paper 2022/1341, 2022.

Che25. B. Chen. Symphony: Scalable SNARKs in the random oracle model from lattice-based high-arity folding. Cryptology ePrint Archive, Paper 2025/1905, 2025.

CLM23. V. Cini, R. W. F. Lai, and G. Malavolta. Lattice-based succinct arguments from vanishing polynomials. Cryptology ePrint Archive, Paper 2023/1405, 2023.

CN11. Y. Chen and P. Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *Advances in Cryptology—ASIACRYPT 2011*, *Lecture Notes in Computer Science* 7073, pages 1–20. Springer, 2011.

DGMV24. N. Dimitriou, A. Garreta, I. Manzur, and I. Vlasov. Mova: Nova folding without committing to error terms. Cryptology ePrint Archive, Paper 2024/1220, 2024.

DP25. B. E. Diamond and J. Posen. Succinct arguments over towers of binary fields. In *Advances in Cryptology—EUROCRYPT 2025, Part IV*, *Lecture Notes in Computer Science* 15604, pages 93–122. Springer, 2025.

EG23. L. Eagen and A. Gabizon. ProtoGalaxy: Efficient ProtoStar-style folding of multiple instances. Cryptology ePrint Archive, Paper 2023/1106, 2023.

FKNP25. G. Fenzi, C. Knabenhans, N. K. Nguyen, and D. T. Pham. Lova: Lattice-based folding scheme from unstructured lattices. In *Advances in Cryptology—ASIACRYPT 2024, Part IV*, *Lecture Notes in Computer Science* 15487, pages 303–326. Springer, 2025.

GLLO26. A. Garreta, H. Lipmaa, U. Luhaäär, and M. Osadnik. Cyclo: Lightweight lattice-based folding via partial range checks. In *Advances in Cryptology—EUROCRYPT 2026, Part VII*, *Lecture Notes in Computer Science* 16547, pages 33–62. Springer, 2026.

GLS+21. A. Golovnev, J. Lee, S. Setty, J. Thaler, and R. S. Wahby. Brakedown: Linear-time and field-agnostic SNARKs for R1CS. Cryptology ePrint Archive, Paper 2021/1043, 2021.

HLSS26. I. Hwang, H. Lee, J. Seo, and Y. Song. Jindo: Practical lattice-based polynomial commitments for client-side proving. Cryptology ePrint Archive, Paper 2026/044, 2026.

KK26. A. Kallesøe and H. Khoshakhlagh. Grand danois: Succinct multilinear polynomial commitments over lattices. Cryptology ePrint Archive, Paper 2026/1196, 2026.

KLN+26. M. Klooss, R. W. F. Lai, N. K. Nguyen, M. Osadnik, and L. Tucci. RoKoko: Lattice-based succinct arguments, a committed refinement. Cryptology ePrint Archive, Paper 2026/575, 2026.

KLNO24. M. Klooß, R. W. F. Lai, N. K. Nguyen, and M. Osadnik. RoK, paper, SISsors – toolkit for lattice-based succinct arguments. Cryptology ePrint Archive, Paper 2024/1972, 2024.

KLNO25. M. Klooß, R. W. F. Lai, N. K. Nguyen, and M. Osadnik. RoK and roll – verifier-efficient random projection for $\tilde{O}(\lambda)$-size lattice arguments. Cryptology ePrint Archive, Paper 2025/1220, 2025.

KLOT25. S. Kuriyama, R. W. F. Lai, M. Osadnik, and L. Tucci. SALSAA – sumcheck-aided lattice-based succinct arguments and applications. Cryptology ePrint Archive, Paper 2025/2124, 2025.

KLPS26. E. Külaots, H. Lipmaa, R. Parisella, and J. Siim. Special soundness and binding properties: A framework for tightly secure zk-SNARKs. Cryptology ePrint Archive, Paper 2026/326, 2026.

KP23. A. Kothapalli and B. Parno. Algebraic reductions of knowledge. In *Advances in Cryptology—CRYPTO 2023, Part IV*, *Lecture Notes in Computer Science* 14084, pages 669–701. Springer, 2023.

KS22. A. Kothapalli and S. Setty. SuperNova: Proving universal machine executions without universal circuits. Cryptology ePrint Archive, Paper 2022/1758, 2022.

KS24a. A. Kothapalli and S. Setty. HyperNova: Recursive arguments for customizable constraint systems. In *Advances in Cryptology—CRYPTO 2024, Part X*, *Lecture Notes in Computer Science* 14929, pages 345–379. Springer, 2024.

KS24b. A. Kothapalli and S. Setty. NeutronNova: Group-based folding done right. Cryptology ePrint Archive, Paper 2024/1606, 2024.

KST22. A. Kothapalli, S. Setty, and I. Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. In *Advances in Cryptology—CRYPTO 2022, Part IV*, *Lecture Notes in Computer Science* 13510, pages 359–388. Springer, 2022.

LFKN92. C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992.

LS15. A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015.

LS18. V. Lyubashevsky and G. Seiler. Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In *Advances in Cryptology—EUROCRYPT 2018, Part I*, *Lecture Notes in Computer Science* 10820, pages 204–224. Springer, 2018.

MAT22. MATZOV. Report on the security of LWE: Improved dual lattice attack. Technical report, The Center of Encryption and Information Security, IDF, 2022.

Nis92. N. Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

NOZ26. N. K. Nguyen, G. O'Rourke, and J. Zhang. Hachi: Efficient lattice-based multilinear polynomial commitments over extension fields. Cryptology ePrint Archive, Paper 2026/156, 2026.

NS24. N. K. Nguyen and G. Seiler. Greyhound: Fast polynomial commitments from lattices. Cryptology ePrint Archive, Paper 2024/1293, 2024.

NS26. W. Nguyen and S. Setty. Neo and SuperNeo: Post-quantum folding with pay-per-bit costs over small fields. In *Advances in Cryptology—CRYPTO 2026, Part IX*, *Lecture Notes in Computer Science* 16808, pages 492–523. Springer, 2026.

Osa26. M. Osadnik. PikkuFold: Efficient folding in a few kilobytes. Cryptology ePrint Archive, Paper 2026/1809, 2026.

Sch80. J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

Set19. S. Setty. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. Cryptology ePrint Archive, Paper 2019/550, 2019.

STW23. S. Setty, J. Thaler, and R. Wahby. Customizable constraint systems for succinct arguments. Cryptology ePrint Archive, Paper 2023/552, 2023.

vzGG13. J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 3 edition, 2013.

WF26. X. Wang and S. Fu. Compare before clearing: Exact integral-comparison frontiers for lattice extraction. Cryptology ePrint Archive, Paper 2026/1962, 2026.

WX26. X. Wang and J. Xiang. The residue is the resource: Exact fresh-state complexity, reconstruction limits, and challenge design in folding. Cryptology ePrint Archive, Paper 2026/2019, 2026.

XZZ+19. T. Xie, J. Zhang, Y. Zhang, C. Papamanthou, and D. Song. Libra: Succinct zero-knowledge proofs with optimal prover computation. Cryptology ePrint Archive, Paper 2019/317, 2019.

A Carrier Algebra and Algorithms

This appendix records the polynomial routines used in Section 4. The input consists of an explicit homogeneous quadratic map, its image-basis maps, distinct poles, nonzero scales, and $k+1$ source vectors. It does not include the verifier's Cauchy challenge.

A.1 Numerator and evaluation trees

For an index set $I \subseteq \{1, \dots, k\}$, write

$$D_I(T) = \prod_{i \in I} (T - \xi_i), \qquad N_I(T) = \sum_{i \in I} a_i \frac{D_I(T)}{T - \xi_i},$$

where the a_i lie in any fixed finite-dimensional $\mathbb{K}$-vector space. The leaves contain $D_{\{i\}} = T - \xi_i$ and $N_{\{i\}} = a_i$. For disjoint children I_0, I_1, combine them by

$$D_I = D_{I_0} D_{I_1}, \qquad N_I = N_{I_0} D_{I_1} + N_{I_1} D_{I_0}. \tag{41}$$

Each summand acquires precisely the missing denominator factors, so induction on the tree proves the displayed definition of N_I. For carrier construction, the weights are respectively $\lambda_i z_i$, $\lambda_i^2 Q(z_i)$, or $J(R_i)$.

At a level whose subproblems have size at most m, the total scalar work is $O((k/m)M(m)) = O(M(k))$. The equality of these bounds follows from the assumptions on M. There are $O(\log(k+1))$ levels. For an a-dimensional numerator, applying the recurrence to every coordinate costs $O(a\mathsf{P}(k))$. Unequal subtrees handle values of k that are not powers of two.

For evaluation at distinct points $t_1, \dots, t_k$, construct the product tree of $T - t_j$. Starting at its root, reduce the input polynomial modulo each child's product. The leaf remainders are the evaluations. This remainder tree has the same asymptotic cost as the numerator tree. It applies to each coordinate of N and to the derivatives used in the two algorithms. The scalar tree is shared by all coordinates.

Interpolation also uses (41). If $L(T) = \prod_j (T - t_j)$ and the values are h_j, form

$$F(T) = \sum_j \frac{h_j}{L'(t_j)} \frac{L(T)}{T - t_j}.$$

Distinctness gives $L'(t_j) \neq 0$ and $F(t_j) = h_j$. Since $\deg F < k$, this is the unique interpolant. The evaluations of L' and the weighted reconstruction give an $O(aM(k)\log(k+1))$ algorithm for a-dimensional data. These polynomial routines are standard [vzGG13,BS05]. Transposed multiplication and division also give fast evaluation and interpolation routines [BLS03]. Here the trees avoid explicit Cauchy matrix multiplication.

A.2 Multipoint construction in detail

The rational function M_Q/D has simple poles:

$$\frac{M_Q(T)}{D(T)} = \sum_i \frac{\lambda_i^2 Q(z_i)}{T - \xi_i}.$$

Differentiate in $\mathbb{K}(T)$, coordinate by coordinate. The quotient rule gives

$$-\left(\frac{M_Q}{D}\right)' = \frac{M_Q D' - M_Q' D}{D^2} = \sum_i \frac{\lambda_i^2 Q(z_i)}{(T - \xi_i)^2}. \tag{42}$$

This uses formal derivatives and involves no division by factorials. The identity is valid in positive characteristic as well.

A complete execution proceeds as follows.

1. Construct D, N, M_Q by the numerator tree. If source values $Q(z_i)$ are not part of the available data, evaluate them first.
2. Form D', M_Q' and evaluate D, D', N, M_Q, M_Q' at the fixed nonpole points $\tau_1, \dots, \tau_k$.
3. Invert the nonzero $D(\tau_j)$ and compute the vectors s_j and h_j in Section 4.2. Equation (42) removes the diagonal source terms.
4. Interpolate the h_j in ambient Y. Apply J to each coefficient to obtain the serialized image-coordinate order.

The interpolation polynomial has degree below k by construction. Equation (5) identifies its k values with those of H, whose degree is also below k. This proves equality of every coefficient, not just equality at the interpolation points.

The need for k distinct nonpole points is an algorithmic precondition of this version. The concrete instance uses $\xi_i = i - 1$ and $\tau_j = 15 + j$, for $1 \le i, j \le 16$. They lie in $\mathbb{F}_q \subset \mathbb{K}$ and are distinct because $q > 31$. They are not sampled during the protocol.

A.3 Deriving the pole-residue formula

Fix a pole ξ_i. The rational source vector can be written locally as

$$z_0 + \frac{N(T)}{D(T)} = \frac{\lambda_i z_i}{T - \xi_i} + z_0 + \sum_{j \neq i} \frac{\lambda_j z_j}{T - \xi_j}.$$

The second part is regular at ξ_i and takes the value v_i. After the double-pole diagonal term in (5) is subtracted, the coefficient of $(T - \xi_i)^{-1}$ is $R_i = \lambda_i B(z_i, v_i)$. The following elementary calculation also obtains it without manipulating Laurent series.

Since $D = (T - \xi_i)P_i$, formal differentiation gives

$$D'(\xi_i) = P_i(\xi_i), \qquad D''(\xi_i) = 2P_i'(\xi_i).$$

For $j \neq i$, the polynomial P_j contains the factor $T - \xi_i$, so

$$P_j'(\xi_i) = \prod_{h \neq i,j} (\xi_i - \xi_h) = \frac{D'(\xi_i)}{\xi_i - \xi_j}.$$

Consequently

$$\begin{aligned} N'(\xi_i) &= \lambda_i z_i P_i'(\xi_i) + \sum_{j \neq i} \lambda_j z_j P_j'(\xi_i) \\ &= \lambda_i z_i D''(\xi_i)/2 + D'(\xi_i) \sum_{j \neq i} \frac{\lambda_j z_j}{\xi_i - \xi_j}. \end{aligned}$$

Dividing by $D'(\xi_i)$ and subtracting the first summand yields the formula for v_i in (6). The only characteristic restriction is that 2 be invertible. In particular, no condition $\operatorname{char}\mathbb{K} > k$ is needed for this derivative argument.

To verify the reconstruction, use

$$P_i - P_j = \big((T-\xi_j) - (T-\xi_i)\big)P_{ij} = (\xi_i - \xi_j)P_{ij}.$$

The two residue contributions of every unordered source pair therefore combine into its single term in (2). The contributions involving z_0 already have the required coefficient $\lambda_i P_i$. Since each R_i is in Y_B, applying J before polynomial reconstruction loses no information: $UJ(R_i) = R_i$.

For $k = 1$, the formula specializes to $D'' = 0$, $v_1 = z_0$, and $H = \lambda_1 B(z_1, z_0)$. Thus the same algorithm covers this endpoint.

A.4 Inversions, basis changes, and the concrete count model

For nonzero $a_1, \ldots, a_k$, all inverses can be obtained from the inverse of their product. Form the prefix products, invert the final product, and traverse the prefixes backwards to recover each a_i^{-1}. This uses one field inversion and $O(k)$ multiplications. It applies both to $D(\tau_j)$ and to $D'(\xi_i)$. Interpolation denominators can be handled in the same way or precomputed with the fixed interpolation nodes. The arithmetic bounds of Section 4 allow either implementation.

The concrete count uses an explicitly specified schoolbook polynomial routine. Its fixed public preparation uses direct inversions, so it counts those inversions rather than claiming the batch-inversion saving. The asymptotic algorithms and this literal count model compute the same polynomials; they answer different cost questions.

For the concrete relation, one Q evaluation uses 36 multiplications and 32 additions or subtractions in $\mathbb{K}$. One direct B evaluation uses 72 multiplications and 68 additions or subtractions. The maps J, U are identities on $\mathbb{K}^4$, so each application copies four coordinates and uses no field arithmetic. With the source Q-values cached, the multipoint algorithm makes 16 new Q calls. Without that cache it makes 33 in total. The pole-residue algorithm makes 16 calls to B in either case. These map calls are expanded in the operation counts; they are not added a second time to the expanded multiplication and addition counts.

For a general relation, a dense J can dominate the coordinate conversion. Its cost is therefore explicit in (7) and (8). The coefficient representation alone does not provide an efficient basis map. If the basis is computed from the values of B on source basis pairs, both those evaluations and the ensuing linear algebra belong to relation preprocessing.

B Encoding and Compiler Specification

The deterministic compiler fixes the relation, public-coordinate map, committed bit segments, and field constraint matrices. This appendix specifies these objects and their layouts for the instance in Section 7.

B.1 Semantic relation and mixed-output coordinates

The source space is $S = \mathbb{K}^{69}$, with coordinates ordered as

$$z = \big(u, x_0, x_1, x_2, x_3, (a_{j,l}, b_{j,l})_{j=0,\dots,3;\, l=0,\dots,7}\big).$$

Within the last group, j is the outer index and l the inner index; each $a_{j,l}$ is followed by $b_{j,l}$. Define

$$Q_j(z) = \sum_{l=0}^{7} a_{j,l} b_{j,l} - u x_j, \qquad 0 \le j < 4. \tag{43}$$

Thus $Y = \mathbb{K}^4$. The public-coordinate map selects u, x_0, x_1, x_2, x_3. A fresh record has $u = 1$, residual $E = 0$, and all its z coordinates in the embedded base field. Its four equations state that the indicated dot products equal the public values x_j. These are strict constraints: changing $a_{0,0}$ while keeping a nonzero $b_{0,0}$, x_0, and $E = 0$ fixed makes the first equation false.

An accumulator satisfies $Q(z) = E$ over $\mathbb{K}$, with its public coordinates fixed by the instance. The compiler includes a satisfying assignment with $u_0 = 2$ and a nonzero residual, which also exercises the relaxed relation. The public statement and the sparse monomials of (43) are stored separately.

Let $e_{a_{j,0}}, e_{b_{j,0}}$ denote the corresponding standard source basis vectors, and let e_j denote an output basis vector. Then $B(e_{a_{j,0}}, e_{b_{j,0}}) = e_j$. These four explicit pairs span Y, while B always takes values in Y. It follows that $Y_B = Y$, $r = y = 4$, and $J = U = I_4$. This algebraic calculation fixes the basis used by the compiler.

B.2 Canonical field values and comparison helpers

The field representation is

$$q = 281474976710597, \qquad \mathbb{K} = \mathbb{F}_q[\theta]/(\theta^4 - 4\theta^2 + 2).$$

An element of $\mathbb{K}$ is encoded in the basis $(1, \theta, \theta^2, \theta^3)$. Each of its four coefficients is the unique integer in $[0, q)$ representing that base-field element. Its binary encoding has 48 bits, least significant first. Therefore one $\mathbb{K}$ value occupies 192 coefficient bits. Decoding is the linear map

$$\mathrm{Dec}(b) = \sum_{v=0}^{3} \theta^v \sum_{j=0}^{47} 2^j b_{v,j} \quad \text{in } \mathbb{K}.$$

Range constraints make this linear decoding canonical. Boolean constraints alone would also accept the 48-bit representation of q.

For one base-field coefficient, let q_j be the binary digits of q. Store Boolean helpers $e_0, \dots, e_{47}$, and let e_{48} alias the fixed-one coordinate. For $j = 47, \dots, 0$, enforce

$$e_j = \begin{cases} e_{j+1} b_j, & q_j = 1, \\ e_{j+1}(1 - b_j), & q_j = 0, \end{cases} \tag{44}$$
$$e_{j+1} b_j = 0 \text{ if } q_j = 0, \qquad e_0 = 0.$$

Every value bit and every helper is constrained to be Boolean.

Lemma 1 (Canonical comparison). *For Boolean digits $b_0, \dots, b_{47}$, the constraints in* (44) *have a solution in the helper bits exactly when*

$$\sum_{j=0}^{47} 2^j b_j < q.$$

The solution, when it exists, is unique.

Proof. The recurrence forces $e_j = 1$ precisely when all digits from position 47 down to j equal those of q. Starting from $e_{48} = 1$ determines every helper. At the first differing digit, a zero digit of q cannot be replaced by a one, because $e_{j+1} b_j = 0$ would fail. A one digit of q may be replaced by zero; the equality flag then becomes zero and remains zero. Finally, $e_0 = 0$ excludes equality with q. This is exactly strict comparison of the represented nonnegative integers. The forced helpers are Boolean, so they satisfy the additional helper Boolean constraints.

Helpers are single base-field bits. They are not themselves encoded as $\mathbb{K}$ values, so checking them does not create a recursive encoding problem.

B.3 Segments, packing, and commitment aliases

A state segment contains z followed by its four residual coordinates. The carrier segment contains the 16 coefficients of $\widehat{H}$ in increasing powers of T, with the four image coordinates inside each coefficient. Basis coefficients and bits then follow the order specified above. Packing puts each consecutive group of 64 bits into the coefficients of an element of $R_q = \mathbb{F}_q[X]/(X^{64} + 1)$, in increasing powers of X. It applies no field-to-ring homomorphism.

The compiler inlines affine combinations in R1CS linear forms. Its private arithmetic gates are the 32 products $a_{j,l} b_{j,l}$ and the four products $u x_j$ needed to check $Q(z^*) = E^*$. Thus $g_K = 36$ canonical $\mathbb{K}$ gate values are added. Their $192 g_K = 6912$ encoding bits occur first in the auxiliary segment. Gates are ordered by $j = 0, \dots, 3$; within each j, the eight products $a_{j,l} b_{j,l}$ occur in increasing l, followed by $u x_j$. The gate bits are followed by the comparison helpers for all encoded base-field coefficients, in the order of their associated values. For each coefficient the stored helper order is $e_0, \dots, e_{47}$, although their recurrence is evaluated from high to low indices. There are

$$V_K = 18(69 + 4) + 16 \cdot 4 + 36 = 1414$$

encoded $\mathbb{K}$ values in total. Hence

$$h_{\mathrm{aux}} = 48 \cdot 4 V_K = 271488, \qquad L_{\mathrm{aux}} = 192 \cdot 36 + h_{\mathrm{aux}} = 278400.$$

Each state uses $L_{\mathrm{st}} = 192(69 + 4) = 14016$ bits, and the carrier uses $L_H = 192 \cdot 16 \cdot 4 = 12288$ bits. All three lengths are multiples of 64, giving 219, 192, and 4350 ring columns, respectively.

Table 7. Compiler segment layout. Ranges are zero-based and half-open. The order describes the handoff vector, not the order of protocol messages.

Segment	Coefficient range	Commitment matrix
Accumulator	$[0, 14016)$	A_{st}
Fresh record i, $1 \le i \le 16$	$[14016i, 14016(i+1))$	A_{st}
Folded output	$[238272, 252288)$	A_{st}
Carrier	$[252288, 264576)$	A_H
Auxiliary	$[264576, 542976)$	A_{aux}
Constant one	$[542976, 542977)$	Public constraint
Fixed zeros	$[542977, 1048576)$	Public constraints

Table 7 partitions the handoff vector. Its actual unpadded length, including the constant one, is 542977. The reserved capacity is $N_0 = 2^{20}$, leaving $P_{0,\mathrm{zero}} = 505599$ fixed zeros. The first backend layer adds another 320 block-padding coordinates; these are outside the displayed capacity and are counted separately.

The state matrix is reused by the accumulator, the 16 fresh records, and the folded output. The carrier and auxiliary matrices are independently sampled. Thus these 20 logical commitment uses require three independent matrices. In the transcript, the carrier is committed before c; the output and auxiliary segments are committed after c and before field randomness. All private comparison helpers are covered by A_{aux}.

B.4 Field constraints and the linear handoff

Let $\mathbf{b} \in \mathbb{F}_q^{N_0}$ be the vector in Table 7. Every decoded semantic coordinate is a $\mathbb{K}$-linear form in $\mathbf{b}$. A multiplication gate is therefore one row of

$$(\mathcal{A}\mathbf{b}) \odot (\mathcal{B}\mathbf{b}) - \mathcal{C}\mathbf{b} = 0 \quad \text{over } \mathbb{K}.$$

An affine equality is a row whose first factor is the fixed-one coordinate. Inlining affine intermediate values eliminates addition wires without changing the equations or introducing uncommitted private data.

The matrices check canonical encodings; the public u, x_j coordinates of all 18 states; the strict and base-field fresh conditions; the two Cauchy fold equations; and $Q(z^*) = E^*$. They also check the fixed-zero handoff coordinates explicitly. The public coordinates of the output are computed from those of the sources with the Cauchy weights. They are not independent prover-selected inputs.

The exact constraint count is given in Table 8. The circuit does not separately test all input equations $Q(z_i) = E_i$ or assert that the supplied carrier is the honest one. Those properties are the subject of Cauchy consistency extraction in Section 6. Adding such tests here would describe a different front end.

The integer q has four zero bits, which accounts for $4 \cdot 5656 = 22624$ first-difference restrictions. There are $16 \cdot 69 \cdot 3 = 3312$ base-field restrictions on fresh

Table 8. Unpadded field constraints in the concrete compiler.

Constraint category	Rows
Boolean value bits and helpers	542976
Canonical prefix recurrences	271488
First-difference restrictions	22624
Exclusion of equality to q	5656
Public coordinates of 18 states	90
Fresh $u = 1$ and $E = 0$	80
Nonconstant extension components of fresh z	3312
Folded source-coordinate equations	69
Folded residual equations	4
Quadratic product gates	36
Output residual equations	4
Fixed-zero handoff coordinates	505599
Total	1351938

source coordinates. The residual-zero equations already force all four components of each fresh residual to vanish.

Padding adds 745214 zero rows, giving $M_c = 2097152 = 2^{21}$ and $\ell_F = 21$. These rows have zero entries in all three matrices and impose only $0 = 0$. The sumcheck table uses variable zero as the most significant bit of its row index. This convention is independent of the least-significant-first encoding of a field coefficient. An equality table is generated in the same row order, so the final transposes refer to precisely the padded R1CS tables.

Each of the three terminal $\mathbb{K}$-linear claims becomes four $\mathbb{F}_q$-linear claims because $\mathbf{b}$ has base-field coordinates. Together with the commitment equations, the fixed one, and the zeros, the initial scalar linear system has

$$20 \cdot 32 \cdot 64 + 12 + 1 + 505599 = 546572$$

rows before backend block padding, and 546892 afterwards. The 20 in this formula counts commitment uses, whereas the assumption matrix count refers to independently sampled matrices.

B.5 Challenge independence and sparse-matrix representation

For $c \notin \{0, \ldots, 15\}$, the values $(c-i)^{-1}$ and $D(c)^{-1}$ are public coefficients. They change the coefficients of the fold equations and of the output public-coordinate equations. They never change a segment, gate, helper, row, or column allocation. The compiler therefore emits one fixed matrix layout together with coefficient expressions parameterized by c and the public statement.

The three sparse matrices are stored in compressed sparse row form. Their coefficient IDs refer to a dictionary of $\mathbb{K}$ values and to the corresponding public expressions. At $c = 16$, the numbers of nonzero entries in $\mathcal{A}, \mathcal{B}, \mathcal{C}$ are 1358814,

2384833, and 278400. The storage layout remains fixed over the supported challenges. A coefficient that vanishes at a particular challenge does not alter the allocated position.

Each binary matrix has a 40-byte little-endian header containing the format marker `CFDCSR01`, the unpadded and padded row counts, the column count, and the stored-entry count. The data consists of unsigned 32-bit row offsets, column indices, and coefficient IDs. Trailing zero rows are implicit. The coefficient dictionary, expression template, and wire-to-bit map complete the representation.

The relation and parameter files determine the semantic data, the coefficient expressions, and the sparse layouts. Appendix E uses these layouts to derive the communication and operation counts.

C Finite Extraction and Statistical Proofs

C.1 A comparison gate for raw recovery trials

The gate used in Section 6.3 takes a raw recovery algorithm $\mathcal{O}$ at one complete checkpoint. Independent calls restore the same public and prover state and use fresh continuation coins. A call returns failure, a verified short kernel, or a candidate. Write $s = \Pr[\mathcal{O} \neq \bot]$, and let μ bound its entire expected cost.

Two incompatible candidates must yield a verifiable kernel by the specified comparison procedure. For each fixed bad candidate u, assume

$$\Pr[\mathcal{O} \text{ returns a candidate compatible with } u] \leq \varepsilon. \tag{45}$$

Compatibility depends on the objects fixed at this checkpoint, not on messages that may change in a fresh continuation.

Lemma 2 (Raw-trial gate). *There is an algorithm that returns a good candidate or a kernel with probability at least $s - \varepsilon$. Its expected raw-trial cost is at most 2μ, in addition to comparison and checking.*

Proof. Run $\mathcal{O}$ once. On failure, stop; on a kernel, return it. Otherwise retain the candidate u and repeat independent calls to $\mathcal{O}$ until a nonfailure result is obtained. A kernel is returned immediately. Incompatible candidates produce their checked kernel. For compatible candidates, check the target predicate on u and return it only if it is good.

If $s = 0$, the loop is never reached. If $s > 0$, its last sample has the success-conditioned distribution of $\mathcal{O}$. For a fixed bad u, equation (45) bounds the probability of a compatible candidate in that distribution by ε/s. The total probability that the first call returns a bad candidate is at most s. Thus the gate's additional failure probability is at most ε.

For cost, let C_n be the cost of trial n in an independent sequence. The event that the first $n-1$ calls fail is independent of C_n, even if C_n is correlated with its own outcome. Tonelli's theorem gives the expected cost until first success as

$$\sum_{n \geq 1} (1-s)^{n-1} \mathbb{E} C_n \leq \frac{\mu}{s}.$$

The loop is reached with probability at most s, so it contributes at most μ on average. The initial call contributes another μ.

No successful sample is asserted to be uniform over the original challenge set. The first candidate is held fixed only for the second raw trial. This is why a fixed-candidate statistical test can be used for candidates recovered after seeing the first trial's randomness.

C.2 Coordinate sampling with challenge-dependent success

Let the challenge space be $\prod_{j=0}^{s-1} \mathcal{D}_j$ with $|\mathcal{D}_j| = M_j \geq 3$. A call at challenge c succeeds with an arbitrary probability $a(c)$. Calls all restore the same complete short-challenge checkpoint. The success event can include a kernel, which terminates the algorithm immediately.

Choose a uniform center and call the oracle once. If successful, seek two further successful values on every coordinate fiber. When $r \in \{1, 2\}$ values of coordinate j are already excluded, draw uniformly from the remaining $M_j - r$ values. On failure, stop with probability r/M_j; otherwise repeat. A success supplies the next point. These stopping coins are extractor coins.

Lemma 3 (Three-point sampling). *Let p be the success probability of a uniform center. The procedure returns a complete three-point star or a kernel with probability at least*

$$p - 3\sum_{j=0}^{s-1} M_j^{-1}.$$

It makes at most $1 + 2s$ oracle calls in expectation.

Proof. Fix all center coordinates except j, and let δ be the average of a on the resulting fiber. If E is the excluded set of size r, the next attempt succeeds with probability

$$u = \frac{M_j\delta - \sum_{e\in E} a(e)}{M_j - r}.$$

The probability that an attempt either succeeds or stops is

$$D = u + (1-u)\frac{r}{M_j} = \delta + \frac{r - \sum_{e\in E} a(e)}{M_j} \geq \delta.$$

Consequently the expected number of calls in this stage is $1/D$, and its stopping probability is $(1-u)r/(M_jD)$. These formulas apply conditional on the history that selected E.

For each possible original center, the probability of reaching the stage is at most its original success probability $a(c)$. Averaging this bound over the original uniform center gives δ on the fixed fiber. Since $D \geq \delta$, the stage costs at most one call in expectation and stops with mass at most r/M_j. A fiber with $\delta = 0$ cannot reach the stage. Summing for $r = 1, 2$ and all coordinates proves both bounds.

This calculation is taken under the original center distribution. Conditioning first on reaching a later stage would generally bias that distribution. Uniform sampling with at most two excluded values can be implemented by rejection sampling, with expected overhead at most $M_j/(M_j-2) \le 3$ per retained draw.

For the actual bounded retry protocol, apply the lemma separately at each reached response attempt. Its checkpoint is fixed by the past; its current challenge is fresh. A retry in a coordinate fork counts as an unsuccessful oracle call. Summing over at most R_C attempts gives $3R_C \sum_j M_j^{-1}$.

C.3 Algebraic recovery and short integer comparisons

Suppose two accepted responses differ only in coordinate j. Their difference satisfies $AX = t_j \Delta$, where X is the difference of responses and Δ the difference of challenges. The latter is a unit. Thus $w_j = X\Delta^{-1}$ is a well-defined modular slope. Given two such certificates (X_1, Δ_1) and (X_2, Δ_2), unequal slopes produce $v = X_1\Delta_2 - X_2\Delta_1$. Then $Av = 0 \bmod q$, and

$$v \bmod q = (w_j^{(1)} - w_j^{(2)})\Delta_1\Delta_2 \neq 0.$$

If every response has norm at most Z and every challenge has operator norm at most T^{op}, then $\|X_b\|_2 \le 2Z$, $\|\Delta_b\|_{\mathrm{op}} \le 2T^{\mathrm{op}}$, and $\|v\|_2 \le 8T^{\mathrm{op}}Z$.

If both slopes agree, put $w_j = w_j^{(1)} = w_j^{(2)}$. For any aggregation e, define

$$P_{e,j}(T) = \left\langle \sum_a \boldsymbol{c}_a(T)\phi_a^{(e)}, z + (T - \boldsymbol{c}_j)w_j \right\rangle - \sum_{a,b} \boldsymbol{c}_a(T)\boldsymbol{c}_b(T)h_{ab}^{(e)},$$

where only coordinate j depends on T. This degree-two polynomial vanishes at the center and both new coordinate values. Its evaluation matrix has determinant equal, up to sign, to the product of their pairwise differences. That determinant is a unit, so the polynomial is zero over R_q. Its quadratic coefficient yields $\langle \phi_j^{(e)}, w_j \rangle = h_{jj}^{(e)}$. The first response equation gives $Aw_j = t_j$. These facts suffice to recover all aggregate scalar equations.

At a nonterminal layer, any discrepancy in the recovered $\widehat{t}$ or $\widehat{h}$ is handled before this polynomial argument. Both vectors open commitments fixed at the short-challenge checkpoint. Their differences have norm at most $2\gamma_{i+1}$. At the terminal, $t_1, \ldots, t_{s-1}$ and h are public and fixed. Two different reconstructions of t_0 have different canonical pivot digit vectors. Their difference is nonzero modulo q, opens the same pivot commitment to zero, and has norm at most $\rho_p\sqrt{da_L\ell_p}$.

C.4 A modular projection bound

The projection certifies the centered modular slope, rather than the short numerator used to recover it. The following bound is uniform over every fixed candidate vector.

Lemma 4 (One-row test). *Let $\beta \geq 1$, $q \geq 128\beta$, and let w be a centered integer coefficient vector with $\|w\|_2 \geq \beta$. Let the coordinates of π be independent, with $\Pr[\pi_j = -1] = \Pr[\pi_j = 1] = 1/4$ and $\Pr[\pi_j = 0] = 1/2$. Then*

$$\mathbb{E}_\pi \exp\left(-\frac{\operatorname{cen}(\langle \pi, w\rangle)^2}{3\beta^2}\right) \leq \frac{7}{8} + 2^{-1000}.$$

Proof. Consider the q-periodic Gaussian

$$G(x) = \sum_{l\in\mathbb{Z}} e^{-(x+lq)^2/(3\beta^2)}.$$

Its absolutely convergent Fourier expansion is

$$G(x) = \frac{\sqrt{3\pi}\beta}{q} \sum_{h\in\mathbb{Z}} e^{-3\pi^2\beta^2h^2/q^2} e^{2\pi \mathrm{i} hx/q}.$$

Write $S_w = \sum_j \pi_j w_j$. Its characteristic function at frequency h/q is

$$\mathbb{E}e^{2\pi \mathrm{i} hS_w/q} = \prod_j \cos^2(\pi h w_j/q) \in [0,1].$$

Removing coordinates can only increase this product. All Fourier coefficients are nonnegative. Hence for every subvector v of w,

$$\mathbb{E}e^{-\operatorname{cen}(S_w)^2/(3\beta^2)} \leq \mathbb{E}G(S_w) \leq \mathbb{E}G(S_v). \tag{46}$$

If some $|w_j| \geq \beta$, retain that coordinate alone. The $l = 0$ contribution is at most $(1 + e^{-1/3})/2 \leq 7/8$. The coordinate is centered, so $|w_j| \leq q/2$. For $l \neq 0$, the distance to the lth Gaussian peak is at least $|l|q/2$, giving a total contribution at most $2\sum_{l\geq 1} e^{-l^2(q/\beta)^2/12} < 2^{-1000}$.

Otherwise every coordinate has magnitude below β. Retain coordinates until $V = \sum_j v_j^2$ first reaches β^2. Then $\beta^2 \leq V < 2\beta^2$ and

$$\mathbb{E}S_v^2 = V/2, \qquad \mathbb{E}S_v^4 = \frac{3}{4}V^2 - \frac{1}{4}\sum_j v_j^4 \leq \frac{3}{4}V^2.$$

Using $e^{-x} \leq 1 - x + x^2/2$ for $x \geq 0$ bounds the zero peak by

$$1 - \frac{V}{6\beta^2} + \frac{V^2}{24\beta^4} \leq \frac{7}{8}.$$

For the other peaks, independence and $\cosh(tv_j/2)^2 \leq e^{t^2v_j^2/4}$ give $\Pr[|S_v| \geq x] \leq 2e^{-x^2/V}$. For a nonzero integer l, split according to whether $|S_v| \geq |l|q/2$. The former event has probability at most $2e^{-l^2q^2/(8\beta^2)}$; on its complement the Gaussian term is at most $e^{-l^2q^2/(12\beta^2)}$. Summing both signs of l, the nonzero peaks contribute at most

$$4\sum_{l\geq 1} e^{-l^2(q/\beta)^2/8} + 2\sum_{l\geq 1} e^{-l^2(q/\beta)^2/12} \leq \frac{4}{2^{2048}-1} + \frac{2}{2^{1365}-1} < 2^{-1000}.$$

Equation (46) completes the proof.

Corollary 1 (Independent rows). *For $\gamma = \sigma\sqrt{S}$, $\sigma \geq 3$, $q \geq 128\gamma$, and a fixed centered w with $\|w\|_2 \geq \gamma$, an m-row projection satisfies*

$$\Pr[\|\operatorname{cen}(\Pi w)\|_2^2 \leq mS] \leq \left(e^{1/(3\sigma^2)}\theta_P\right)^m \leq \left(\frac{3\sigma^2}{3\sigma^2-1}\theta_P\right)^m.$$

Proof. On the event in question,

$$\exp\left(-\frac{\|\operatorname{cen}(\Pi w)\|_2^2}{3\gamma^2}\right) \geq e^{-m/(3\sigma^2)}.$$

Markov's inequality and row independence bound its probability by $e^{m/(3\sigma^2)}\theta_P^m$. The last inequality uses $e^x \leq (1-x)^{-1}$ for $0 \leq x < 1$.

The last expression is rational when S, σ, m are integers. For the concrete $m = 864$, $\sigma = 4$, its comparison with 2^{-140} is an exact integer-power calculation. The protocol may choose a projection based on earlier failed attempts. For a fixed bad candidate, each new matrix still satisfies the bound conditional on the past. A union bound over R_P attempts therefore gives $R_P\epsilon_P$.

C.5 Transport to seeded projections

We now replace the $R = 2mN$ independent input bits used by the lazy-ternary projection with the output of Nisan's generator [Nis92]. The seed includes the initial block and every hash description. All of these bits are freshly and independently sampled in each raw projection attempt.

Lemma 5 (Fixed-test transport). *Let $F_w : \{0,1\}^R \to \{0,1\}$ be the complete projection PASS predicate for a vector w fixed before the seed. Suppose F_w is computed by a finite-state machine with at most $2^{w_{\mathrm{fsm}}}$ states. For block length b, depth h, and any $\delta > 0$, the recursive affine-Toeplitz generator satisfies*

$$|\Pr[F_w(G) = 1] - \Pr[F_w(U_R) = 1]| \leq (2^h - 1)\delta + \frac{2^{6w_{\mathrm{fsm}}}h}{\delta^2 2^b}.$$

Proof. Nisan's good-hash lemma bounds the probability of a bad hash sequence by $2^{6w_{\mathrm{fsm}}}h/(\delta^2 2^b)$ [Nis92, Lemma 2]. Conditioned on good hashes, the recursive hybrid changes the acceptance probability of any Boolean event by at most $(2^h - 1)\delta$. Splitting on the good-hash event gives the stated inequality. The probability is over the initial block and every hash choice; keeping the hashes fixed and refreshing only the initial block would not satisfy this premise.

For completeness, an affine Toeplitz hash on b bits is

$$h_{a,t}(x)_j = t_j + \sum_{r=0}^{b-1} a_{r+j}x_r \pmod 2, \qquad |a| = 2b-1, \quad |t| = b.$$

It requires $3b-1$ description bits. Starting from $G_0(x)=x$, define

$$G_h(x;h_1,\dots,h_h)=G_{h-1}(x;h_1,\dots,h_{h-1})\,\|\,G_{h-1}(h_h(x);h_1,\dots,h_{h-1}).$$

The complete seed length is $b+h(3b-1)$. The expansion may be truncated after R bits.

The PASS computation consumes two bits per trit. Its state stores the current bit position, one pending bit, the residue modulo q, and the accumulated squared energy clipped at $T+1$. It therefore has at most

$$(R+1)\,4q(T+2)$$

states. The vector w is fixed in the transition function; no table of this size is materialized by the streaming evaluator.

Put

$$w_{\mathrm{fsm}}=\left\lceil \log_2\big((R+1)4q(T+2)\big)\right\rceil, \qquad K=\lceil \log_2 R\rceil, \qquad e=150,$$

and choose

$$b=6w_{\mathrm{fsm}}+2K+3e+\lceil \log_2 K\rceil+3.$$

Let h be the least integer satisfying $b2^h\ge R$, and set $\delta=2^{-e}/(2(2^h-1))$. The first term in Lemma 5 is 2^{-e-1}. Since every concrete layer has $h\le K$ and $b<R$, the second term is strictly below

$$2^{-e-1}\frac{h}{2^{\lceil \log_2 K\rceil}}\le 2^{-e-1}.$$

This proves (23). The six concrete substitutions are evaluated with exact integers and rational numbers. Table 9 records their block lengths, depths, seed lengths, and expansion costs.

Table 9. Seeded-projection parameters. The prefix-hash count is for one expansion scan.

Layer	Block bits	Depth	Seed bits	Seed bytes	Prefix hashes
0	1 186	21	75 883	9 486	1 528 239
1	1 226	20	74 766	9 346	581 825
2	1 218	19	70 625	8 829	345 940
3	1 212	18	66 642	8 331	259 507
4	1 206	18	66 312	8 289	223 018
5	1 204	18	66 202	8 276	204 466

The lemma is pointwise for arbitrary fixed w. A vector chosen from earlier adaptive history is covered after conditioning on that history, provided the next complete seed is independent. The conclusion concerns this one PASS test. It does not compare the joint distributions (d,Π_d) and (d,Π_{iid}), nor does it fool an arbitrary interactive prover.

C.6 Short-challenge moments and honest retries

For $\mathcal{D} = \{-2, -1, 0, 1, 2\}^d$, every integer representative $\widetilde{\boldsymbol{c}}$ has $\|\widetilde{\boldsymbol{c}}\|_1 \le 2d$. Negacyclic multiplication is a sum of signed coordinate shifts, so its operator norm on coefficient ℓ_2 is at most this ℓ_1 norm. Distinct challenges have coefficients differing by at most four. Under the field and cyclotomic conditions used here, the short-difference unit criterion [BC25] applies when $4 < \sqrt{q/2}$.

Each challenge coefficient has mean zero and variance two. For an integer ring vector $\widetilde{v}$, expansion of each output coefficient and cancellation of cross terms gives

$$\mathbb{E}\|\widetilde{\boldsymbol{c}}\widetilde{v}\|_2^2 = 2d\|\widetilde{v}\|_2^2. \tag{47}$$

Centering modulo q can only reduce the absolute value of each coefficient. Thus for modular multiplication the conclusion is the inequality $\mathbb{E}\|\operatorname{cen}(\boldsymbol{c}v)\|_2^2 \le 2d\|v\|_2^2$.

For several blocks, let $Y_j = \operatorname{cen}(\boldsymbol{c}_j w_j)$. Since q is odd and the challenge distribution is symmetric, $\mathbb{E}Y_j = 0$. The Y_j are independent, so

$$\mathbb{E}\left\|\operatorname{cen}\left(\sum_j \boldsymbol{c}_j w_j\right)\right\|_2^2 \le \mathbb{E}\left\|\sum_j Y_j\right\|_2^2 = \sum_j \mathbb{E}\|Y_j\|_2^2 \le \mu S.$$

Consequently a response threshold $G \ge 2\mu S$ succeeds with probability at least $1/2$ by Markov's inequality. At the pivot terminal, conditioning the first challenge on being nonzero preserves symmetry and raises its moment by at most $M/(M-1)$. Equation (22) gives the same $1/2$ success bound.

For the projection, the uncentered variance calculation is $\mathbb{E}\|\Pi\operatorname{coef}(w)\|_2^2 = m\|w\|_2^2/2$. Centering again decreases the norm. Hence an honest projection response under independent trits passes $\|p\|_2^2 \le mS$ with probability at least $1/2 + 1/(2(mS+1))$, using the integrality of the energy in Markov's inequality. Direct substitution gives $2^{-150} < 1/(2(mS+1))$ in every layer. Lemma 5 therefore leaves seeded honest success at least $1/2$. Fresh challenges in each bounded retry family give exhaustion probability at most 2^{-R_P} and 2^{-R_C}, respectively. The response bound remains valid after projection succeeds, because the witness is fixed and the short challenges are newly sampled. A union bound across layers proves (29).

C.7 Norm propagation through a child

The radix decomposition used by the honest prover is deterministic. For a nonnegative coefficient x, repeatedly set $x' = \lfloor (x + b_i^{\mathrm{dig}} - 1)/\rho_i \rfloor$ and use the remainder $x - \rho_i x'$. The remainder lies in $[-b_i^{\mathrm{dig}} + 1, b_i^{\mathrm{dig}}]$. Apply this to $|x|$ and restore the sign for negative x. After $\ell_i = \lceil \log_{\rho_i} q \rceil$ digits, every centered coefficient has been represented. This fixes the tie convention at the radix endpoints.

For an honest split $z = z_{\mathrm{lo}} + \rho_i z_{\mathrm{hi}}$, there are dn_i low coefficients, each bounded by b_i^{dig}. The triangle inequality yields

$$\|z_{\mathrm{hi}}\|_2 \le \frac{\sqrt{G_i} + b_i^{\mathrm{dig}}\sqrt{dn_i}}{\rho_i}.$$

Adding the separate $\widehat{t}$ and $\widehat{h}$ digit energies proves the sharper honest bound

$$S_{i+1} \geq dn_i(b_i^{\mathrm{dig}})^2 + \frac{(\sqrt{G_i} + b_i^{\mathrm{dig}}\sqrt{dn_i})^2}{\rho_i^2} + E_{t,i} + E_{h,i}.$$

Using $(a+b)^2 \leq 2a^2 + 2b^2$ gives the rational version (20).

An extracted child only has the total norm bound $\|x\|_2 \leq \gamma_{i+1}$. Its blocks need not be canonical radix digits. Nevertheless, Cauchy–Schwarz gives

$$\|\operatorname{cen}(z_{\mathrm{lo}} + \rho_i z_{\mathrm{hi}})\|_2 \leq \sqrt{1+\rho_i^2}\sqrt{\|z_{\mathrm{lo}}\|_2^2 + \|z_{\mathrm{hi}}\|_2^2} \leq \sqrt{1+\rho_i^2}\,\gamma_{i+1}.$$

Each auxiliary block also has norm at most γ_{i+1}. Combining these inequalities with the integer comparison above proves (27).

C.8 Conditional costs and finite composition

The number of invocations is random and can depend on previous successes. Let N be this count, and let $\mathcal{F}_{n-1}$ contain the complete history before call n. The event $N \geq n$ is $\mathcal{F}_{n-1}$-measurable. If every reached checkpoint satisfies $\mathbb{E}[C_n \mid \mathcal{F}_{n-1}] \leq H + T$, then

$$\begin{aligned}\mathbb{E}\sum_{n=1}^{N} C_n &= \sum_{n\geq 1} \mathbb{E}\left[\mathbf{1}_{\{N\geq n\}}\mathbb{E}[C_n \mid \mathcal{F}_{n-1}]\right] \\ &\leq (H+T)\sum_{n\geq 1}\Pr[N \geq n] = (H+T)\mathbb{E}N.\end{aligned}$$

Tonelli applies because the costs are nonnegative. This argument includes histories that select unusual challenges or child checkpoints; it requires a uniform conditional cost bound there, rather than a bound averaged only over an honest challenge.

For success, let the actual forward execution select child checkpoint t with subprobability measure ν. It selects at most one child, so ν has mass at most one. If the child extractor succeeds with probability $p_t - \Lambda_{i+1}$ or better at every t, averaging gives

$$\int (p_t - \Lambda_{i+1})\, d\nu(t) \geq \int p_t\, d\nu(t) - \Lambda_{i+1}.$$

The coordinate sampler then applies to this raw oracle's actual success function. Its loss is additive; the child loss is not charged again at every requested fork. The raw-trial gate doubles the expected trial cost and adds only the fixed-candidate return error. These two facts give (30) and the induction $p_i - \Lambda_i$.

Throughout this induction a successful child return includes its actual accepting center branch. Prefix it by the parent's recorded messages to retain one accepting continuation. It is not necessary that the extracted witness, when used by an honest prover, reproduce all those messages. The record serves to identify the actual challenge and output commitment attached to the recovered source witnesses.

D Computational Reduction and Module-SIS Instantiation

D.1 One assumption term per sampled matrix

The setup distribution is a product of uniform module-matrix distributions. Its coordinates are the three front matrices $A_{\mathrm{st}}, A_H, A_{\mathrm{aux}}$, the three matrices A_i, B_i, D_i at every nonterminal layer, and the two matrices A_L, B_p at the pivot terminal. With L nonterminal layers, there are $3L+5$ coordinates. The concrete six-layer chain has $L = 5$ and $J = 20$.

A matrix can be used in several commitment equations. In particular, A_{st} authenticates $k+2$ state records: one accumulator, k fresh states, and one folded output. These uses neither create independent setup coordinates nor require guessing which commitment will collide. A short difference for any one of them solves the same A_{st} instance.

For role j, the challenge is a uniform matrix $A_j \in R_q^{a_j \times b_j}$. The solver's output is an integer vector in $\mathcal{O}^{b_j}$ satisfying

$$A_j v = 0 \pmod{q}, \qquad v \bmod q \neq 0, \qquad \|\operatorname{coef}(v)\|_2 \leq \beta_j.$$

Define $\mathrm{Adv}^{\mathrm{MSIS}}_{\vartheta_j}(t)$ as the maximum success probability of a solver with runtime at most t for this game. The norm is coefficient ℓ_2 for every role in this instantiation. Where $\beta_j < q$, requiring an integer vector to be nonzero is equivalent to the nonzero-modulo-q requirement within the permitted radius: a nonzero vector in $q\mathcal{O}^{b_j}$ has norm at least q.

Coefficient expansion produces a structured negacyclic matrix of size da_j by db_j. It is an exact representation of the module equation and coefficient norm. Its entries are not an independently sampled scalar matrix. The attack estimates reported later use the specified coefficient-lattice model; the computational theorem retains the role-specific Module-SIS assumptions.

D.2 Embedding a challenge without changing setup

Fix a role j. The reduction receives a Module-SIS matrix with exactly the dimensions and ring of that role. It places this matrix in coordinate j and independently samples every other coordinate from its declared distribution. Public data derived from these matrices is then generated by the protocol's usual algorithm. The reduction runs the adversary's initialization to reach the pre-Cauchy checkpoint and invokes the truncated extractor.

This is a coordinate-preserving embedding into the product setup. No change of basis, rejection predicate on the supplied matrix, or extra correlation between sampled keys is used. Public statements and commitments may depend on the entire setup; they are generated after this embedding in the same order as in the real experiment. The full joint distribution, rather than only each marginal matrix distribution, is consequently preserved.

When the extractor returns a role-j kernel, the reduction checks the equation, the norm, and modular nonzeroness, then returns the vector. Any other role or

failure is a failure for this reduction. This defines a separate reduction for every fixed j.

To make the role events disjoint, the extractor returns immediately on the first verified kernel. Its role label is part of the output. If a local comparison can produce several kernels, use a fixed public order to select the first. Writing K_j for the event that the truncated extractor returns role j, we have

$$\Pr\left[\bigcup_j K_j\right] = \sum_j \Pr[K_j].$$

Each fixed-role reduction gives the corresponding bound on $\Pr[K_j]$. A random choice of role would incur a guessing factor, but it is unnecessary for this summed-advantage theorem.

D.3 A single global clock and complete runtime charges

To keep the formulas readable, this appendix writes $\bar{p}, P, B, E_B, S_j^{\max}$ for the seeded quantities $\bar{p}_N, P_N, B_N, E_{B_N}, S_{j,N}^{\max}$ of Theorem 3. Fix the cost model before interpreting an advantage bound. The extractor's clock covers its entire execution from the supplied pre-Cauchy checkpoint. It includes verifier sampling, prover continuations, copying and restoring complete states, nested extractor calls, field and ring arithmetic, integer norm checks, kernel validation, and output construction. The expectation P in Theorem 3 bounds this whole clock, averaged over the actual setup and initialized prover state.

Let T be that runtime and stop the full computation after B steps. Markov's inequality gives $\Pr[T > B] \le P/B$. The same clock runs across all layers and all recovery trials. There is no separate truncation loss per layer or per matrix.

For the fixed-role reduction, write

$$S_j^{\max} = S_{\mathrm{keys},j}^{\max} + S_{\mathrm{public},j}^{\max} + S_{\mathrm{init},j}^{\max} + S_{\mathrm{interface},j}^{\max}.$$

The terms bound, respectively: generation of the other setup matrices; generation of the remaining public data; the adversary's initialization up to the supplied checkpoint; and any challenge conversion, input/output handling, or final return work outside the extractor's clock. Any cost already inside that clock need not be counted a second time. A common larger bound is also valid.

With these charges, the solver runs in at most $B + S_j^{\max}$.

When the adversary is strict polynomial time and the protocol dimensions are polynomially bounded, the initialization and interface terms have worst-case polynomial bounds. Exact finite-alphabet sampling can be charged as a primitive operation in a stated arithmetic model. In a random-bit model it requires the additional treatment below; an unbounded rejection loop is not a worst-case setup bound.

D.4 Quantifiers when averaging over setup

Let ω contain the complete setup and initialized prover state, and let $s(\omega)$ be its pre-Cauchy checkpoint. The statistical extractor is the same algorithm for all ω. Its pointwise guarantee is

$$\Pr[\mathrm{Src} \cup \mathrm{Ker} \mid \omega] \geq \max\{0, p(s(\omega)) - \kappa_{\mathrm{alg}}\}.$$

The error bound is determined by the public parameter contract and does not depend on a lucky setup. Therefore

$$\Pr[\mathrm{Src} \cup \mathrm{Ker}] \geq \mathbb{E}_\omega p(s(\omega)) - \kappa_{\mathrm{alg}} = \bar{p} - \kappa_{\mathrm{alg}}.$$

The runtime assumption is separately $\mathbb{E}_\omega \mathbb{E}[T \mid \omega] \leq P$. It suffices for the one averaged Markov bound. There is no step that chooses a favorable setup after seeing the adversary's acceptance behavior, or a different role after receiving a single external challenge.

After truncation, subtracting the labelled kernel events gives

$$\begin{aligned}\Pr[E_B \text{ returns Src}] &\geq \bar{p} - \kappa_{\mathrm{alg}} - P/B - \sum_j \Pr[K_j] \\ &\geq \bar{p} - \kappa_{\mathrm{alg}} - P/B - \sum_j \mathrm{Adv}^{\mathrm{MSIS}}_{\vartheta_j}(B + S_j^{\max}).\end{aligned}$$

Taking the maximum with zero yields the stated theorem. In a parameter family with a growing number of roles, the assumption must control the entire matrix family uniformly, or directly bound its summed advantage. Negligibility for each fixed role index, with no uniform bound as the index grows, would not suffice.

D.5 Bounded setup sampling from random bits

A uniform $\mathbb{F}_q$ coefficient can be sampled by drawing $m_q = \lceil \log_2 q \rceil$ unbiased bits and accepting the integer if it is below q. Its acceptance probability is greater than $1/2$. For the extra keys in reduction j, let

$$N_{\mathrm{set},j} = d \sum_{h \neq j} a_h b_h$$

be the number of required base-field coefficients. Bound every coefficient sampler by r_{set} attempts and fail the reduction on exhaustion. Then

$$\eta_{\mathrm{set},j} \leq N_{\mathrm{set},j} 2^{-r_{\mathrm{set}}}. \tag{48}$$

If other public setup randomness uses rejection sampling, its bounded samplers and failure probabilities are included in the same accounting.

Couple this implementation with exact rejection sampling using the same random bits. On the event that no cap is reached, the two setups agree, including

all derived public data. Successful sampled coefficients have the required uniform distribution. Thus

$$\Pr[K_j] \le \mathrm{Adv}^{\mathrm{MSIS}}_{\vartheta_j}(B + S_j^{\max}) + \eta_{\mathrm{set},j},$$

where $S_j^{\max}$ now includes at most $N_{\mathrm{set},j} r_{\mathrm{set}}$ coefficient-sampling attempts. Summing these bounds adds $\sum_j \eta_{\mathrm{set},j}$ to the theorem. Sampling inside the extractor itself remains covered by its full clock and the one global truncation.

D.6 Interpreting the concrete estimates

The concrete table below lists each distinct matrix together with its coefficient ℓ_2 radius. The estimator input mapping is $n_{\mathrm{SIS}} = da_j$, $m_{\mathrm{SIS}} = db_j$, and length bound β_j. It also states the classical or quantum cost convention used for each estimate.

A finite estimated lattice-reduction cost describes the specified attack model. It is not a value for the full time-dependent advantage function in (36). In particular, the statistical exponent of κ_{alg}, an estimated work factor, and the loss P/B measure different quantities. Their conversion into an end-to-end computational claim would require a common time and success model and the complete summed-advantage accounting. The concrete section reports them separately.

D.7 Concrete commitment matrices

Table 10 contains the matrix set used in the computational theorem. For each nonterminal layer $i = 0, \ldots, 4$, A_i commits a response block, B_i commits the radix digits of its block commitments, and D_i commits symmetric-data digits. The terminal uses A_5 and the pivot matrix B_p. Adding the three front-end keys gives twenty independent matrices. Repeated uses of A_{st} do not add independent roles.

All radii are derived from the kernel comparisons in Section 6. In particular, the front-end radius $8,192 = 2\sigma_0\sqrt{S_0}$ uses the extracted whole-handoff bound, rather than replacing it with the smaller honest size of one segment. Table 11 separates these two quantities. For the nonterminal matrices, the child norm budget determines the extracted response radius before the three-point comparison is applied. The terminal pivot radius instead bounds a difference of radix-digit openings.

D.8 Attack models and estimator configuration

For a role with a rows and b columns over R_q, the estimator input is

$$(n_{\mathrm{SIS}}, m_{\mathrm{SIS}}, q, \beta, \mathrm{norm}) = (64a, 64b, 281474976710597, \beta, 2).$$

No conversion from an infinity-norm radius is used. The official `SIS.estimate` API selects the returned attack; the reported branch is the Euclidean lattice

Table 10. All twenty independent matrix roles. Dimensions are over R_q and every radius is a coefficient ℓ_2 radius. The classical and quantum columns are $\log_2(\text{rop})$ under the stated models. † marks a BKZ block size above the model's fitted range of 1,024.

Matrix	a	b	β	BKZ block	Classical	Quantum
A_{st}	32	219	8 192	10 912†	3124.36	2819.20
A_H	32	192	8 192	11 555†	3305.55	2981.96
A_{aux}	32	4 350	8 192	10 832†	3102.03	2799.16
A_0	24	1 821	4 567 851 477	797	255.49	238.73
B_0	16	1 728	139 383	2 406†	708.03	645.76
D_0	16	1 080	139 383	2 406†	708.03	645.76
A_1	24	1 075	3 861 716 751	814	260.25	243.01
B_1	16	1 152	117 836	2 493†	732.60	667.86
D_1	16	504	117 836	2 493†	732.60	667.86
A_2	24	762	3 353 291 926	828	264.18	246.54
B_2	16	960	102 322	2 570†	754.35	687.43
D_2	16	360	102 322	2 570†	754.35	687.43
A_3	24	711	3 025 014 479	838	266.98	249.07
B_3	16	768	92 305	2 628†	770.74	702.16
D_3	16	240	92 305	2 628†	770.74	702.16
A_4	24	608	2 902 545 478	843	268.38	250.33
B_4	16	768	88 568	2 652†	777.52	708.26
D_4	16	240	88 568	2 652†	777.52	708.26
A_5	24	318	181 386 952	1 206†	370.28	341.99
B_p	4	384	1 255	1 500†	450.69	414.10

attack. We ran the official lattice estimator [A+26] with SageMath 10.9 and Python 3.12.14 under Ubuntu 24.04, using its SIS implementation without local changes.

Classical costs use the estimator's `MATZOV` option [MAT22]. Quantum costs use its near-neighbor option with depth-times-width cost. BKZ cost estimation depends on both the required reduction quality and the assumed cost of the reduction algorithm [CN11]. The software descends from the concrete-hardness framework of [APS15]; its SIS implementation determines the present estimates. The table therefore records block size as well as the reported operation exponent. For fifteen matrices, in both models, the block size exceeds the fitted range of 1,024. Table 10 identifies these extrapolated entries.

The Euclidean branch does not return a memory estimate or an explicit success probability, so Table 10 reports neither quantity. The quantum column concerns an attack-cost model only. The node extractor remains the classical resettable extractor of the theorem.

Table 11. Honest opening squared-norm bounds and extraction kernel radii. The kernel radius is a different quantity from the honest opening norm.

Matrix	Honest ℓ_2^2 bound	Kernel radius β
A_{st}	14 016	8 192
A_H	12 288	8 192
A_{aux}	278 400	8 192
A_0	1 048 576	4 567 851 477
B_0	113 246 208	139 383
D_0	70 778 880	139 383
A_1	303 555 488	3 861 716 751
B_1	75 497 472	117 836
D_1	33 030 144	117 836
A_2	216 957 652	3 353 291 926
B_2	62 914 560	102 322
D_2	23 592 960	102 322
A_3	163 590 043	3 025 014 479
B_3	50 331 648	92 305
D_3	15 728 640	92 305
A_4	133 127 892	2 902 545 478
B_4	50 331 648	88 568
D_4	15 728 640	88 568
A_5	122 566 619	181 386 952
B_p	393 216	1 255

E Concrete Serialization and Operation Counts

This appendix derives the serialization and operation counts for the instance in Table 3. The lattice estimates use the configuration in Appendix D.8.

E.1 Serialization grammar

All integers in this appendix refer to the single instance of Table 3. A canonical $\mathbb{F}_q$-element occupies six little-endian bytes; a value at least q is rejected. An extension-field value lists its four coefficients in the basis $(1, \theta, \theta^2, \theta^3)$. A ring element lists the coefficients of $1, X, \ldots, X^{63}$. These encodings require 24 and 384 bytes, respectively.

A projection attempt sends the complete seed for the fixed Nisan generator. Seed bits are packed least-significant bit first, and nonzero padding bits in the final byte are rejected. A short-challenge coefficient uses three bits for one of five symbols. Unused symbols and nonzero padding bits are rejected. All messages are byte-aligned separately. The public parameter identifier determines all vector lengths and attempt caps, so no variable-length dimension header is part of a fold. This grammar counts protocol messages. The application's public-instance envelope and the CRS are separate objects.

The three newly sent front-end commitments have 32 ring entries each, totaling 36, 864 bytes. The field protocol sends four $\mathbb{K}$-coefficients per sumcheck round and three terminal $\mathbb{K}$-claims:

$$B_{P\to V}^{\text{field}} = 4 \cdot 24\ell_F + 3 \cdot 24 = 2,088.$$

Its two length-ℓ_F verifier vectors occupy $48\ell_F = 1,008$ bytes. The Cauchy challenge is another 24 bytes. The carrier's private coefficients are committed, not transmitted as a raw field vector.

E.2 Retry-dependent wire lengths

Let P_i and C_i be the projection and short-response attempt counts at layer i, each in $\{1, \ldots, 160\}$ for an accepted execution. A failed attempt sends only its tag. The selected projection response is sent once, and its checkpoint is then locked. A malformed disclosed object causes rejection; it does not open another retry.

Table 12. Per-layer payloads, in bytes. The prover column excludes the one-byte tag at each projection or response attempt. Aggregation is sent once after a projection is selected.

Layer	Fixed $P \to V$	Seed/attempt	Aggregation	Short/attempt
0	17 472	9 486	9 859 608	216
1	17 472	9 346	83 574	144
2	17 472	8 829	88 182	120
3	17 472	8 331	83 574	96
4	17 472	8 289	85 878	96
5	216 366	8 276	85 878	168

The fixed prover payloads in Table 12 sum to 303, 726 bytes. Thus

$$B_{P\to V}^{\text{fold}} = 340,662 + 96\ell_F + \sum_{i=0}^{5}(P_i + C_i). \tag{49}$$

The sum of tags is 12 without retries and at most 1,920. For the verifier, let R_i^{bytes} and A_i^{bytes} be the projection-seed and aggregation columns of the table. The packed seed and three-bit short encoding give

$$B_{V\to P} = 24 + 48\ell_F + \sum_{i=0}^{5}\left(R_i^{\text{bytes}}P_i + A_i^{\text{bytes}} + 24s_iC_i\right). \tag{50}$$

The six no-retry seeds occupy 52,557 bytes after separate byte alignment. The terminal condition on the first short coordinate changes the distribution, not its fixed serialized length.

Substituting $\ell_F = 21$ yields

	No retry	Maximum accepted attempts
$P \to V$	342,690	344,598
$V \to P$	10,341,123	18,831,246
Interactive total	10,683,813	19,175,844

Fresh commitments contribute a separate $16 \cdot 32 \cdot 384 = 196,608$ bytes. The incoming accumulator contributes $32 \cdot 384 = 12,288$ bytes if it too must be transported. These totals are exact lengths under the selected syntax, not measurements of generated proof files.

E.3 Carrier operations

Table 13 uses the same literal polynomial-arithmetic model for all algorithms: schoolbook products, monic remainder division at fixed padded degrees, and zero-initialized accumulators. Because the poles and internal interpolation nodes lie in $\mathbb{F}_q$, their polynomial constants act by extension-by-base-field multiplication. Such a product uses four base-field products in the chosen basis, and is kept separate from a general $\mathbb{K}$-product.

Table 13. Carrier arithmetic under the fixed schoolbook reference model. Source Q-values are cached for the multipoint row. Public polynomial preprocessing is separate.

Algorithm	$\mathbb{K}$ products	$\mathbb{K}$–$\mathbb{F}_q$ products	$\mathbb{K}$ adds/subs	J calls
Streaming pairs	9 792	8 224	17 472	136
Multipoint	576	43 372	48 624	16
Pole-residue	1 152	41 995	46 720	16

The map costs, in extension-field operations, are

Map	Products	Additions/subtractions
Q	36	32
B	72	68

The basis maps $J = U = I_4$ copy four coordinates per call and perform no field arithmetic. If the seventeen source Q-values are not already available, the multipoint row adds seventeen evaluations, or 612 $\mathbb{K}$-products and 544 additions/subtractions.

The public constant builder shared by the fast algorithms uses 1,693 base-field multiplications, 1,630 additions/subtractions, and 48 inversions under its literal reference implementation. It prepares product trees, derivative values, interpolation reciprocals, and the pole-residue constants. We report this one-time public preprocessing separately from the online counts. The streaming reference's additional independently constructed P_i and pair polynomials cost 29,040 multiplications and the same number of additions. The asymptotic algorithms

may replace these schoolbook routines by faster polynomial multiplication; the concrete table does not assume that replacement.

E.4 Field circuit and public operators

The compiler has 1,414 canonical $\mathbb{K}$-values, or 5,656 base-field coefficients. Their 271,488 value bits require another 271,488 prefix-equality helper bits. Witness construction performs 1,232 $\mathbb{K}$-products and 1,232 additions for the two fold equations, followed by 36 product-gate evaluations. The challenge-dependent public coefficient preparation performs 9,391 $\mathbb{K}$-products, 96 $\mathbb{K}$-additions, one $\mathbb{K}$-inversion, and 360 base-field negations. These costs are charged once after the Cauchy challenge.

For a sparse matrix application, the counted implementation processes each stored entry, including an entry whose coefficient happens to vanish for a special public challenge. The witness coordinate is in $\mathbb{F}_q$, so the product is a $\mathbb{K}$-by-$\mathbb{F}_q$ product. For a transpose, the equality weight is in $\mathbb{K}$, so the general product is counted in $\mathbb{K}$. The exact counts are

Operation	$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$
Apply: $\mathbb{K}$–$\mathbb{F}_q$ products	1,358,814	2,384,833	278,400
Apply: $\mathbb{K}$ additions	6,876	1,032,895	6,876
Transpose: $\mathbb{K}$ products	1,358,814	2,384,833	278,400
Transpose: $\mathbb{K}$ additions	815,837	1,336,257	0

The public transposes are computed once by each party. They are reused through projection retries.

An equality table is built by replacing an entry x with xr and $x - xr$. For $M_c = 2^{21}$, this takes $M_c - 1 = 2,097,151$ products and the same number of subtractions. For every pair of entries in the sumcheck's four tables, expanding $E(AB - C)$ and folding the tables uses fourteen products and seventeen additions/subtractions. There are $M_c - 1$ pairs across all rounds. This gives the sumcheck counts in the main text. The verifier's twenty-one round checks use 63 products, 147 additions/subtractions, and 21 equalities; its final scalar check adds 64 products, 64 additions/subtractions, and one equality.

The initial linear relation also includes commitments and fixed coordinates. One reference application uses 271,488 ring products and 270,848 ring additions for its commitment blocks, followed by 3,145,728 extension-by-base-field products and 3,145,725 extension-field additions for the three terminal field rows. The explicit coordinate equations require 505,920 copies or checks after backend block padding. The protocol incurs this application cost at the call sites described in Section 5.

One transpose application uses 271,488 ring products, 263,004 ring additions, 12,582,912 base-field products, and 13,631,808 coefficient accumulations. Each party calls it three times for the first layer's independent aggregation rows. The resulting field transpose vectors are reused across projection retries.

E.5 Commitments, projections, and lattice layers

Under ordinary dense ring matrix-vector multiplication, the new output, carrier, and auxiliary commitments use respectively

$$\begin{aligned} &(7,008,\ 6,144,\ 139,200) && R_q\text{-products},\\ &(6,976,\ 6,112,\ 139,168) && R_q\text{-additions}. \end{aligned}$$

Incoming commitment formation is separate. The table uses ordinary dense ring products; the bit-valued openings also admit a bounded-coefficient implementation.

Table 14. Main lattice-layer operation counts. The first two columns count full ring products once per layer. Projection additions are expectations per raw attempt. The last column counts bounded-integer–field-coefficient products per response attempt.

Layer	Block $A_i w_j$	Symmetric data	Projection adds	Response products
0	393 336	442 503	453 123 072	67 129 344
1	154 800	116 100	178 329 600	26 419 200
2	91 440	57 150	105 338 880	15 605 760
3	68 256	34 128	78 630 912	11 649 024
4	58 368	29 184	67 239 936	9 961 472
5	53 424	46 746	61 544 448	9 117 696

For every projection attempt, the verifier samples a complete seed and both parties expand the required prefix of $864N_i$ lazy-ternary symbols. A row sum uses one addition or subtraction for each nonzero trit. Each raw trit is uniform, so the expected addition count is $432N_i$; the exact count depends on the selected seed. The verifier does not multiply this matrix by the private witness. Both parties nevertheless use the selected projection in the public aggregation transpose. For three aggregation rows this projection part has expected addition count $1,296N_i$. These are expectations for raw attempts, not an assertion that every accepted attempt has that count.

The complete seeds contain 75,883, 74,766, 70,625, 66,642, 66,312, and 66,202 bits at layers zero through five. One scan uses 1,528,239, 581,825, 345,940, 259,507, 223,018, and 204,466 prefix-hash evaluations, respectively. Counting one prover scan and one selected-adjoint scan by each party gives 9,428,985 prefix hashes on a no-retry path. The literal binary and carry-less multiplication realizations are alternative generator implementations and remain separate operation units.

A short response uses bounded challenge coefficients. The last column of Table 14 counts the schoolbook coefficient products used in the negacyclic block combinations. After reduction, each response coefficient is centered and squared for the norm check. The selected response alone is decomposed for the child.

Table 15. One application of each child linear circuit, and the coefficient accumulations for its transpose. The transpose preserves the typed multiplication counts. These are operator costs, not an additional call charged unconditionally to both parties.

Layer	R_q mult.	Bounded R_q mult.	Radix mult.	R_q adds	Transpose adds
0	94 230	216	273 792	94 360	18 947 531
1	55 584	144	161 536	55 642	11 177 136
2	41 739	120	122 688	41 773	8 397 479
3	35 355	96	101 952	35 365	7 105 054
4	32 574	96	95 360	32 584	6 551 326

Retry-dependent total work is obtained by counting the actual attempts; the once-per-layer symmetric data are not recomputed for each short challenge.

The child circuits combine the public A_i, B_i, D_i products, radix recomposition, and the principal relation. Their transposes reverse the same linear circuit, using coefficient-involuted public ring constants. A transpose therefore preserves the typed multiplication counts, while its zero-initialized adjoints accumulate contributions at every reversed edge. Table 15 records these accumulations explicitly. It counts the specified circuit; other implementations may use different addition schedules.

E.6 Terminal payload and verification

The terminal's fixed 216, 366-byte payload consists of 55, 296 bytes for six disclosed block commitments, a 1, 536-byte pivot commitment, a 5, 184-byte selected projection, 32, 238 bytes of symmetric data, and a 122, 112-byte response. The symmetric-data encoding omits only the three constant coefficients determined by the diagonal equations. All other coefficients are present.

After receiving an accepted response, the verifier computes $A_5 z$ with 7,632 ring products and 7,608 ring additions. It subtracts the six known challenge-weighted block commitments, using 144 bounded ring products and 144 additions or subtractions. Reconstructing the pivot then uses one ring-unit inversion and 24 ring products. The verifier extracts 24,576 radix digits and checks the pivot commitment with 1,536 ring products and 1,532 ring additions.

The principal relation has three rows. Its left side costs 954 ring products and 951 additions; its symmetric right side costs 84 products and 81 additions. Combining the public φ-vectors uses 6,678 bounded ring products and 5,724 additions. The cached short-challenge products require 28 bounded-by-bounded ring products and 1,344 coefficient doublings. Eighteen base-field additions or subtractions recover the omitted diagonal constants. The response norm check uses 20,352 integer squares and 20,351 additions. Public aggregation and its three source transposes are separate from these final checks.