Scilight

Pragmatic Cybersecurity
https://www.sciltp.com/journals/pc



Article

Check for updates

Σ-Orchestra: A Harmonized Lattice-Based Σ-Protocol for Polynomial Relations with Exact ℓ_2-Soundness †

Lizhen Zhang ‡, Shang Gao ‡ and Bin Xiao *

Department of Computing, The Hong Kong Polytechnic University, Hong Kong SAR, China
* Correspondence: b.xiao@polyu.edu.hk

† Extended from a non-archival presentation titled: Σ-Orchestra: A Harmonized Lattice-Based Σ-Protocol for Polynomial Relations with Exact ℓ_2-Soundness. Presented at the 31st Australasian Conference on Information Security and Privacy (ACISP 2026), Perth, WA, Australia, 6–9 July 2026.

‡ These authors contributed equally to this work.

How To Cite: Zhang, L.; Gao, S.; Xiao, B. Σ-Orchestra: A Harmonized Lattice-Based Σ-Protocol for Polynomial Relations with Exact ℓ_2-Soundness *Pragmatic Cybersecurity* **2026**, *1*(3), 20. https://doi.org/10.53941/pc.2026.100020

Received: 22 July 2026
Revised: 13 October 2026
Accepted: 5 November 2026
Published: 24 November 2026

Abstract: Compressed lattice-based Σ-protocols face a difficulty in preserving exact ℓ_2-norm bounds during knowledge extraction. Existing succinct constructions generally extract witnesses with relaxed norm bounds, whereas protocols supporting exact norms often incur substantially larger communication costs. In this work, we construct a lattice-based compressed Σ-protocol for polynomial relations with exact ℓ_2-soundness. Our approach first reduces polynomial and norm constraints to linear relations using multilinear extensions and sum-check. In particular, exact ℓ_2 bounds are enforced through a quadratic inner-product constraint together with an ℓ_∞ range check, preventing norm relaxation in the subsequent reduction. We then batch and amortize the resulting linear relations through a split-and-fold procedure, followed by a lattice-based proof compression step. The resulting protocol achieves logarithmic proof size and quasi-linear prover and verifier complexity while preserving the original ℓ_2-norm bound. We also describe applications to verifiable FHE ciphertext validity and lattice-based secret sharing.

Keywords: lattice-based cryptography; Σ-protocol; sum-check protocol

1. Introduction

Σ-protocol is a well-established proof-of-knowledge primitive, characterized by a three-move, public-coin structure (commit–challenge–response) [1]. They serve as fundamental building blocks for cryptographic applications, including identification, authentication, and digital signatures. Over the years, their expressiveness has been extended from basic algebraic statements to proofs of partial knowledge [2,3] and more general linear and polynomial relations [4–6]. More recently, compressed Σ-protocols [7] have employed Bulletproofs-inspired folding techniques [8] to achieve logarithmic communication.

Despite significant progress, most efficient Σ-protocols rely on discrete logarithm (DL) or modular-root assumptions and are therefore not post-quantum secure. Post-quantum alternatives in the lattice setting have been extensively studied, but adapting Σ-protocols to this setting introduces challenges, particularly in handling norm constraints. Specifically, lattice-based hardness assumptions typically require witnesses to satisfy strict norm bounds, which are difficult to enforce directly using standard algebraic proof techniques. Moreover, conventional rewinding-based extraction often incurs norm expansion, yielding only relaxed ℓ_2-norm knowledge soundness. Such expansion may accumulate through successive folding and amortization steps, causing extracted witnesses to violate the original norm bound. Consequently, achieving succinct lattice-based Σ-protocols with exact ℓ_2-norm knowledge soundness remains challenging.

Several approaches have been developed to address norm constraints. Stern-type protocols [9] use restricted challenge spaces to control norm growth, but require multiple repetitions and relatively large proofs. Other constructions [10,11]

Copyright: © 2026 by the authors. This is an open access article under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).
Publisher's Note: Scilight stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

achieve exact ℓ_∞-soundness for small bounds using CRT/NTT techniques, with prover complexity growing linearly in the bound. Direct ℓ_2-norm proofs based on inner-product arguments [12] avoid norm relaxation but incur (quasi-)linear communication. More recent constructions provide different trade-offs: LaBRADOR [13] obtains succinct proofs with constant ℓ_2-norm slack, whereas LatticeFold+ [14] achieves exact ℓ_∞-soundness using sum-check. However, no existing work integrates these ideas into a unified Σ-protocol framework while preserving logarithmic communication and the expressiveness needed for arbitrary polynomial relations. What is missing is not new primitives, but a carefully structured composition that avoids the accumulation effects and structural incompatibilities inherent in naively stacking prior techniques. This leads to the central question of this work:

> *Can we design a lattice-based Σ-protocol that achieves exact ℓ_2-soundness without sacrificing succinctness or practicality?*

1.1. Our Contributions

We construct a compressed lattice-based Σ-protocol for polynomial relations with exact ℓ_2-soundness. Our construction consists of three main components.

Direct polynomial-to-linear reduction. We develop a reduction from polynomial constraints to linear ones using multilinear extensions and sum-check. The reduction works directly with polynomial evaluations, without requiring circuit compilation, FFT-based techniques [7], MPC [5], or CCS/R1CS encodings [14,15]. It also applies to nested polynomial expressions, with logarithmic communication overhead for bounded-degree relations.

Exact ℓ_2-norm constraint reduction. Following the decomposition of [12], we express an exact ℓ_2-norm constraint as a quadratic inner-product relation together with an ℓ_∞ bound. Specifically, for $\vec{f} \in \mathbb{Z}_q^N$, we enforce $\langle \vec{f}, \vec{f} \rangle \bmod q \leq \beta^2$ and $\|\vec{f}\|_\infty \leq D$ under the condition $q > 2ND^2$ to prevent modular wrap-around. We linearize the quadratic relation using sum-check and reduce the ℓ_∞ constraint to linear relations using the range-checking technique of LatticeFold+ [14]. The resulting reductions have logarithmic communication and verification costs, with linear prover complexity.

Lattice-based compressed Σ-protocol. We combine the resulting linear constraints across multiple instances using batching and a split-and-fold amortization procedure. The amortized linear relation is then proved by a lattice-based three-move Σ-protocol, whose proof is further compressed using a lattice-based inner-product argument. The final construction achieves logarithmic proof size and quasi-linear prover and verifier complexity. Its knowledge soundness follows from the reductions of knowledge and the extraction of the final Σ-protocol, allowing recovery of witnesses satisfying the original polynomial and ℓ_2-norm constraints.

Table 1 compares our construction with existing Σ-protocols. Here, **P.Q.** denotes post-quantum security, **S.S.** standard soundness, and **P./V.** prover/verifier complexity, while m denotes the input size. Communication is measured in group or lattice elements rather than bits, under a common security level. For compressed Σ-protocols, the number of folded instances is treated as constant in the asymptotic comparison.

Table 1. Asymptotic performance comparison of our scheme and prior Σ-protocols.

Σ-Protocols	P.Q	S.S	Relation	Size	(Quasi-)Linear P./V.
[7]	×	/	Linear	$O(\log m)$	✓
[16]	×	/	Linear	$O(\log m)$	✓
[5]	×	/	Poly	$O(\mathsf{polylog}(m))$	✓
[17]	✓	×	Linear	$O(\log m)$	✓
[18]	✓	×	Linear	$O(\log m)$	✓
[19]	✓	×	Linear	$O(\log m)$	✓
Ours	✓	✓	Poly	$O(\log m)$	✓

1.2. Technique Overview: A Pipeline for Lattice-Based Σ-Protocols

We give an intuitive overview of our construction. At a high level, our protocol forms a structured pipeline that transforms a rich lattice relation–including arbitrary polynomial constraints and exact ℓ_2 bounds–into a single linear relation suitable for a compressed Σ-protocol. We work over the ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ and aim to prove

$$\mathcal{R}^{\beta}_{\mathsf{poly},2} := \left\{ \begin{array}{c} \mathbb{x} = (\mathsf{cm}_f \in \mathcal{R}_q^\kappa, \vec{\boldsymbol{v}} \in \mathcal{R}_q^m), \;\; \mathbb{w} = \vec{\boldsymbol{f}} \in \mathcal{R}_q^m, s.t. \\ \|\vec{\boldsymbol{f}}\|_2 \leq \beta, \;\; \mathsf{Com}(\vec{\boldsymbol{f}}) = \mathsf{cm}_f, \;\; h(\boldsymbol{f}_i) = \boldsymbol{v}_i \;\; \forall i \in [m] \end{array} \right\}, \tag{1}$$

where Com is the homomorphic Ajtai commitment and h is an arbitrary polynomial map.

Figure 1 summarizes the protocol flow. A batch of $\mathcal{R}^{\beta}_{\mathsf{poly},2}$ instances is first linearized, then batched and amortized into a single linear claim. A classical three-move Σ-protocol proves this linear relation, and a LaBRADOR-style compression step yields a succinct proof.

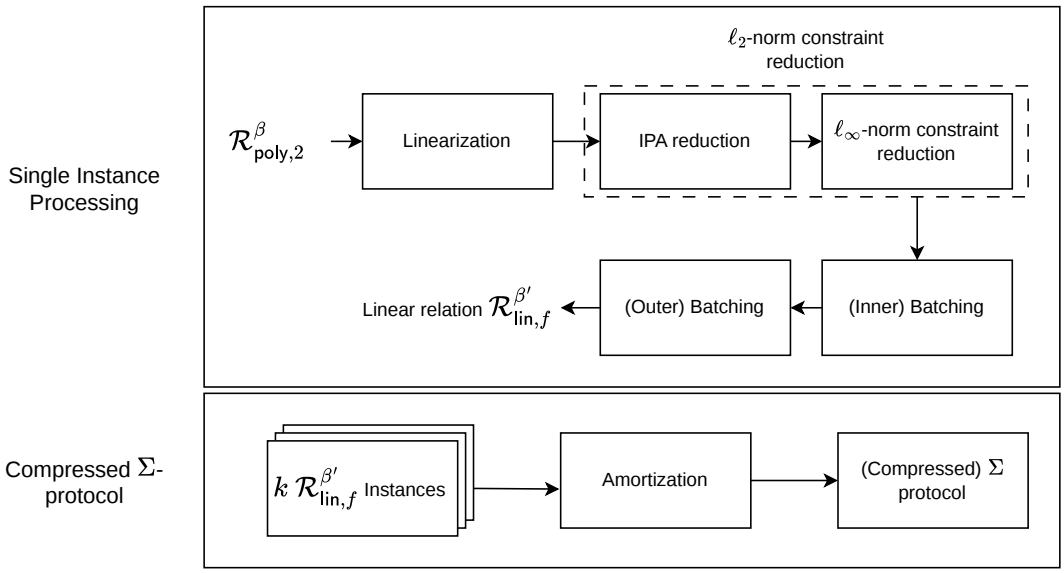


Figure 1. Workflow of our protocol.

1.2.1. RoK Toolkits

Section 3 introduces several reductions of knowledge (RoKs), all following the same paradigm: encode vectors as multilinear extensions (MLEs) and apply sum-check so that non-linear constraints reduce to a single linear evaluation claim.

Polynomial linearization. To linearize $h(\boldsymbol{f}_i) = \boldsymbol{v}_i$, we apply MLEs to $\vec{\boldsymbol{f}}$ and $\vec{\boldsymbol{v}}$, yielding $h(\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{b})) = \mathsf{mle}[\vec{\boldsymbol{v}}](\vec{b})$ for $\vec{b} \in \{0,1\}^{\log m}$. We encode this via $g_1(\vec{\boldsymbol{X}}) = \mathsf{eq}(\vec{\mu}, \vec{\boldsymbol{X}})\big(h(\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{\boldsymbol{X}})) - \mathsf{mle}[\vec{\boldsymbol{v}}](\vec{\boldsymbol{X}})\big)$, whose Boolean sum is 0. A sum-check reduces this to the evaluation claim

$$g_1(\vec{r}_1) = \boldsymbol{s}_{g_1} \quad \Leftrightarrow \quad \mathsf{eq}(\vec{\mu}, \vec{r}_1)\big(h(\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{r}_1)) - \mathsf{mle}[\vec{\boldsymbol{v}}](\vec{r}_1)\big) = \boldsymbol{s}_{g_1}.$$

All other terms are public, the prover reveals only $\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{r}_1)$. Since $\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{r}_1) = \langle \vec{\boldsymbol{f}}, \mathsf{tensor}(\vec{r}_1)\rangle$, the polynomial constraint reduces to a single linear constraint.

Exact ℓ_2-norm reduction. The exact norm bound $\|\vec{\boldsymbol{f}}\|_2 \le \beta$ is handled analogously. By Lemma 2, it reduces to an inner-product argument (IPA) together with an ℓ_∞ bound. We generate $\vec{a}$ with $\langle \vec{a}, \vec{a}\rangle = \beta^2 - \langle \vec{f}, \vec{f}\rangle$ and form $\vec{f}' = (\vec{f}, \vec{a})$, giving $\langle \vec{f}', \vec{f}'\rangle = \beta^2$. Applying MLEs and a sum-check to $g_2(\vec{\boldsymbol{X}}) = \langle \mathsf{mle}[\vec{f}'](\vec{\boldsymbol{X}}), \mathsf{mle}[\vec{f}'](\vec{\boldsymbol{X}})\rangle$ linearizes the quadratic IPA into constraints on $\vec{f}$ and $\vec{a}$. The ℓ_∞ bound is enforced via the LatticeFold+ range check, which likewise reduces to linear constraints using monomial matrices.

Batching. The same MLE–sum-check paradigm is used in our inner and outer batching steps, allowing many linear constraints across multiple witnesses to be aggregated into a single linear evaluation claim.

1.2.2. Lattice-Based Compressed Σ-Protocol

Our compressed Σ-protocol builds directly on the sub-protocols introduced above.

Amortization. For each instance, we first reduce $\mathcal{R}^{\beta}_{\mathsf{poly},2}$ to a linear relation via the RoKs. Across resulting k linear instances, we apply a split-and-fold procedure that controls witness growth and merges them into a single amortized relation. As shown in Theorem 12, this amortization constitutes a valid reduction of knowledge: although extraction may initially return a relaxed opening, all norm constraints have already been reduced to linear ones, so the knowledge soundness of the underlying RoKs ensures that the extracted witness satisfies the exact linear relations and thus matches the true opening.

Compressed Σ-protocol. After amortization, we run a standard lattice-based three-move Σ-protocol on the folded instance. Finally, following the compression method of [7], we incorporate a lattice-based IPA [13] to further compress the proof while preserving post-quantum security.

2. Preliminaries

We introduce the notation and basic tools used throughout; additional details appear in Appendix A.

2.1. Notation

Let q be a modulus and $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$. For $n \in \mathbb{N}$, write $[n] = \{0, \ldots, n-1\}$ and $[a{:}b] = \{a, \ldots, b-1\}$. Vectors in $\mathbb{Z}_q^n$ are written $\vec{f} = (f_0, \ldots, f_{n-1})$; $(\vec{a}, \vec{b})$ denotes concatenation. Sampling $a \leftarrow_{\$} \mathbb{S}$ means sampling uniformly from $\mathbb{S}$. For $\vec{r} \in \mathbb{Z}_q^{\log n}$, define $\mathsf{tensor}(\vec{r}) = \bigotimes_{i \in [\log n]} (1 - r_i, r_i) \in \mathbb{Z}_q^n$.

Cyclotomic rings. Let $\mathcal{R} = \mathbb{Z}[X]/(X^d + 1)$ and $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$ for a power-of-two $d > 1$. Elements of $\mathcal{R}_q$ are written in bold, e.g., $\boldsymbol{f} = \sum_{i=0}^{d-1} f_i X^i$, with coefficient vector $\vec{f} = (f_0, \ldots, f_{d-1})$ and constant term $\mathsf{ct}(\boldsymbol{f})$. For $\vec{\boldsymbol{f}} \in \mathcal{R}_q^m$, let $F \in \mathbb{Z}_q^{m \times d}$ denote the matrix of coefficient vectors. For $A \in \mathbb{Z}_q^{m \times n}$, $\vec{A}_j$ denotes its j-th column. (Elements of $\mathbb{Z}_q$ are not boldfaced, though they embed naturally as constant polynomials.)

Norms. For $\boldsymbol{f} = \sum_{i=0}^{d-1} f_i X^i \in \mathcal{R}_q$, define $\|\boldsymbol{f}\|_1 = \sum_i |f_i|$, $\|\boldsymbol{f}\|_2 = \sqrt{\sum_i f_i^2}$, $\|\boldsymbol{f}\|_\infty = \max_i |f_i|$. For $\vec{\boldsymbol{f}} = (\boldsymbol{f}_0, \ldots, \boldsymbol{f}_{m-1}) \in \mathcal{R}_q^m$, set $\|\vec{\boldsymbol{f}}\|_1 = \sum_{i=0}^{m-1} \|\boldsymbol{f}_i\|_1$, $\|\vec{\boldsymbol{f}}\|_2 = \sqrt{\sum_{i=0}^{m-1} \|\boldsymbol{f}_i\|_2^2}$, and $\|\vec{\boldsymbol{f}}\|_\infty = \max_i \|\boldsymbol{f}_i\|_\infty$. Unless stated otherwise, $\|\cdot\|$ denotes the ℓ_2 norm.

Challenge space. The challenge space is defined as follows [20].

Definition 1 (**Challenge Space**). *A subset $\mathcal{C} \subset \mathcal{R}_q$ is a* challenge space *if differences of distinct elements are non–zero divisors, and* strong *if they are invertible [20]. For a small-norm challenge space $\mathcal{C}_{\mathsf{small}}$, define the expansion factor*

$$\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} = \sup_{\rho \in \mathcal{C}_{\mathsf{small}},\, \boldsymbol{v} \in \mathcal{R}} \frac{\|\rho \boldsymbol{v}\|}{\|\boldsymbol{v}\|}.$$

A standard example is $\mathbb{Z}_q \subset \mathcal{R}_q$.

Lemma 1 (**Invertibility in Polynomial Rings** [21]). *If q is prime and $q \equiv 5 \bmod 8$, then any $\boldsymbol{f} \in \mathcal{R}_q$ with $0 < \|\boldsymbol{f}\| < q^{1/2}$ is invertible in $\mathcal{R}_q$.*

Integer-to-Ring Mapping. The map $M : \mathbb{Z}_q^{md} \to \mathcal{R}_q^m$ groups each block of d coefficients into one ring element. For example,
$M(\vec{a}) = (\sum_{i=0}^{d-1} a_i X^i, \sum_{i=0}^{d-1} a_{d+i} X^i, \ldots, \sum_{i=0}^{d-1} a_{(m-1)d+i} X^i)$ for $\vec{a} \in \mathbb{Z}_q^{md}$.

Exact ℓ_2-norm constraint decomposition. Following [12], an exact ℓ_2-norm constraint can be decomposed into the following sub-constraints.

Lemma 2. *Let β be the ℓ_2-norm bound and $\vec{f} \in \mathbb{Z}_q^{md}$ the coefficient vector of $\vec{\boldsymbol{f}} \in \mathcal{R}_q^m$. The constraint $\|\vec{\boldsymbol{f}}\|_2^2 \le \beta^2$ is equivalent to: (i) $\langle \vec{f}, \vec{f} \rangle \bmod q < \beta^2$, (ii) $\|\vec{f}\|_\infty \le D$ for $q > 2D^2 \cdot md$.*

Proof. From the ℓ_∞ bound, $\|\vec{\boldsymbol{f}}\|_2^2 = \sum_{i=1}^{md} f_i^2 \le D^2 md < \frac{q}{2}$. Hence the centered representative of $\langle \vec{f}, \vec{f} \rangle \bmod q$ equals its true integer value $\|\vec{\boldsymbol{f}}\|_2^2$. Therefore, $\langle \vec{f}, \vec{f} \rangle \bmod q < \beta^2 \iff \|\vec{\boldsymbol{f}}\|_2^2 \le \beta^2$. □

2.2. Standard ℓ_∞-Norm Proof in LatticeFold+

We recall the range-proof protocol of [14], which relies on the following notions. Let $\mathcal{M} = \{1, X, \ldots, X^{d-1}\}$ be the monomial set, and for $\boldsymbol{a} = \sum_{i=0}^{d-1} a_i X^i$ define the evaluation map $\mathsf{ev}_{\boldsymbol{a}}(v) = \sum_i a_i v^i$.

Lemma 3 (**Monomial Check (Corollary 4.1 in** [14])). *For $\boldsymbol{a} \in \mathcal{M}$ and $v \in \mathcal{C}$, $\mathsf{ev}_{\boldsymbol{a}}(v)^2 = \mathsf{ev}_{\boldsymbol{a}}(v^2)$. If $\boldsymbol{a} \notin \mathcal{M}$, the equality holds with probability $< 2d/|\mathcal{C}|$.*

For $a \in (-d, d]$, let $sgn(a) \in \{-1, 0, 1\}$ denote the sign of a and $exp(a) = sgn(a) X^{|a|} \in \mathcal{M}$. Define $EXP(a) = \{exp(a)\}$ if $a \neq 0$ and $\{0, 1, X^{d/2}\}$ otherwise. For a matrix $A \in (-d, d)^{m \times n}$, let $exp(A)$ be the entrywise application of $exp(\cdot)$, and let $EXP(A)$ be the Cartesian product of the corresponding $EXP(\cdot)$ sets.

Lemma 4 (**Integer Range Check (Lemma 2.2 in** [14])). *Let $d' := d/2$ and $\psi := \sum_{i=1}^{d'-1} i \cdot (X^i + X^{-i})$. If $a \in (-d', d')$, then for all $\boldsymbol{b} \in EXP(a)$, $\mathsf{ct}(\boldsymbol{b} \cdot \psi) = a$. Conversely, if $\mathsf{ct}(\boldsymbol{b} \cdot \psi) = a$ for some $\boldsymbol{b} \in \mathcal{M}$, then $a \in (-d', d')$ and $\boldsymbol{b} \in EXP(a)$.*

2.3. LaBRADOR Relation

We recall from [13] the following relation:

$$\mathcal{R} = \left\{ ((\mathcal{F}, \mathcal{F}', \beta), (\vec{z}_0, \ldots, \vec{z}_{r-1})) \;\middle|\; \begin{array}{l} f(\vec{z}_0, \ldots, \vec{z}_{r-1}) = 0 \quad \forall f \in \mathcal{F}, \\ \mathsf{ct}\big(f'(\vec{z}_0, \ldots, \vec{z}_{r-1})\big) = 0 \quad \forall f' \in \mathcal{F}', \\ \sum_{i=0}^{r-1} \|\vec{z}_i\|_2^2 \le \beta^2 \end{array} \right\}, \tag{2}$$

where $\mathcal{F}$ is a family of quadratic dot-product functions $f : \mathcal{R}_q^n \times \cdots \times \mathcal{R}_q^n \to \mathcal{R}_q$ of the form

$$f(\vec{z}_0, \ldots, \vec{z}_{r-1}) = \sum_{i,j=0}^{r-1} \alpha_{i,j} \langle \vec{z}_i, \vec{z}_j \rangle + \sum_{i=0}^{r-1} \langle \vec{\varphi}_i, \vec{z}_i \rangle - \boldsymbol{\beta}, \tag{3}$$

where $\boldsymbol{\alpha}_{i,j}, \boldsymbol{\beta} \in \mathcal{R}_q$, $\vec{\boldsymbol{\varphi}}_i \in \mathcal{R}_q^n$ for $n, r \ge 0$. $\mathcal{F}'$ is another family that focuses on the constant coefficients of the polynomials corresponding to the functions f', which have the same form as in Equation (3). The LaBRADOR protocol proves membership in $\mathcal{R}$ with $O(\log \log N)$ proof size and quasi-linear prover and verifier time.

3. A Toolbox for Protocol Construction

This section introduces the reduction of knowledge (RoK) protocols that serve as building blocks for our final Σ-protocol. Each subsection presents a modular component: Π_{lin} for polynomial linearization, Π_{rg} for ℓ_2-norm reduction, Π_{ba} for batching linear constraints, Π_{tool} used for amortization, and zero-knowledge variants. All protocols use challenge sets $\mathcal{C} := \mathbb{Z}_q$ for sum-check and $\mathcal{C}' \subseteq \mathcal{R}_q$ (Definition 1) for folding.

3.1. Linearization

We linearize the relation $h(\boldsymbol{f}_i) = \boldsymbol{v}_i$ for $i \in [m]$ via a sum-check over MLEs. Using MLEs, this is equivalent to $h(\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{b})) - \mathsf{mle}[\vec{\boldsymbol{v}}](\vec{b}) = 0$ for all $\vec{b} \in \{0,1\}^{\log m}$. Note that to show a $p(\vec{x})$ vanishes on $\{0,1\}^{\log m}$, the condition $\sum_{\vec{b}} p(\vec{b}) = 0$ is insufficient due to cancellations. Instead, to isolate a single Boolean point, we sample $\vec{\mu} \leftarrow \mathcal{C}^{\log m}$ and enforce $\sum_{\vec{b} \in \{0,1\}^{\log m}} \mathsf{eq}(\vec{\mu}, \vec{b})\, p(\vec{b}) = 0$. Thus, with $\vec{\mu} \leftarrow_{\$} \mathcal{C}^{\log m}$, we define:

$$g_1(\vec{\boldsymbol{X}}) = \mathsf{eq}(\vec{\mu}, \vec{\boldsymbol{X}}) \cdot \Big(h\big(\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{\boldsymbol{X}})\big) - \mathsf{mle}[\vec{\boldsymbol{v}}](\vec{\boldsymbol{X}}) \Big). \tag{4}$$

A sum-check is then run to prove $\sum_{\vec{b} \in \{0,1\}^{\log m}} g_1(\vec{b}) = 0$, which ultimately reduces to a single evaluation claim expressible as a linear constraint on $\vec{\boldsymbol{f}}$.

We define the output relation $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ as follows:

$$\mathcal{R}^{\beta}_{\mathsf{lin},2} := \left\{ \begin{array}{c} \mathbb{x} = (\mathsf{cm}_f \in \mathcal{R}_q^{\kappa},\ \vec{r}_1 \in \mathcal{C}^{\log m},\ \boldsymbol{s}_1 \in \mathcal{R}_q),\ \mathbb{w} = \vec{\boldsymbol{f}} \in \mathcal{R}_q^m,\ s.t. \\ \|\vec{\boldsymbol{f}}\|_2 \le \beta, \quad \mathsf{Com}(\vec{\boldsymbol{f}}) = \mathsf{cm}_f, \quad \langle \vec{\boldsymbol{f}}, \mathsf{tensor}(\vec{r}_1) \rangle = \boldsymbol{s}_1. \end{array} \right\}. \tag{5}$$

Protocol 1 details Π_{lin}, which reduces an input $(\mathsf{cm}_f, \vec{\boldsymbol{v}}; \vec{\boldsymbol{f}}) \in \mathcal{R}^{\beta}_{\mathsf{poly},2}$ to an output $(\mathsf{cm}_f, \vec{r}_1, \boldsymbol{s}_1; \vec{\boldsymbol{f}}) \in \mathcal{R}^{\beta}_{\mathsf{lin},2}$.

Protocol 1. Π_{lin}: reduce $\mathcal{R}^{\beta}_{\mathsf{poly},2}$ to $\mathcal{R}^{\beta}_{\mathsf{lin},2}$

$\mathcal{P}(\mathsf{cm}_f, \vec{\boldsymbol{v}}, \vec{\boldsymbol{f}})$, $\mathcal{V}(\mathsf{cm}_f, \vec{\boldsymbol{v}})$

1: $\mathcal{V} \to \mathcal{P}$: $\vec{\mu} \leftarrow_{\$} \mathcal{C}^{\log m}$.

2: $\mathcal{P}$ and $\mathcal{V}$: Construct $g_1(\vec{\boldsymbol{X}})$ as defined in Equation (4). Engage in a sum-check for the claim $\sum_{\vec{b} \in \{0,1\}^{\log m}} g_1(\vec{b}) = 0$, which reduces to check an evaluation claim $g_1(\vec{r}_1) = \boldsymbol{s}_{g_1}$, where $\vec{r}_1 \leftarrow_{\$} \mathcal{C}^{\log m}$.

3: $\mathcal{P} \to \mathcal{V}$: $\boldsymbol{s}_1 := \langle \vec{\boldsymbol{f}}, \mathsf{tensor}(\vec{r}_1) \rangle$.

4: $\mathcal{V}$: Set $\boldsymbol{s}_v := \langle \vec{\boldsymbol{v}}, \mathsf{tensor}(\vec{r}_1) \rangle$ and $\boldsymbol{e}_\mu := \mathsf{eq}(\vec{\mu}, \vec{r}_1)$. Check whether

$$\boldsymbol{s}_{g_1} \stackrel{?}{=} \boldsymbol{e}_\mu \big(h(\boldsymbol{s}_1) - \boldsymbol{s}_v \big).$$

5: $\mathcal{P}$: Output $\vec{\boldsymbol{f}}$.

6: $\mathcal{V}$: Output $(\mathsf{cm}_f, \vec{r}_1, \boldsymbol{s}_1)$.

Theorem 1. Π_{lin} *is a reduction of knowledge from* $\mathcal{R}^{\beta}_{\mathsf{poly},2}$ *to* $\mathcal{R}^{\beta}_{\mathsf{lin},2}$.

Proof. Public reducibility is immediate: from $(\mathsf{cm}_f, \vec{v})$ and transcript $(\vec{r}_1, \boldsymbol{s}_1, \vec{\mu})$, the verifier outputs $(\mathsf{cm}_f, \vec{r}_1, \boldsymbol{s}_1)$. Perfect completeness follows from the completeness of sum-check, and knowledge soundness from Theorem 2. □

Theorem 2. *When the commitment scheme* $\mathsf{Com}(\cdot)$ *is binding, the protocol* Π_{lin} *is knowledge sound with soundness error* $\epsilon_{\mathsf{lin}} = \frac{(\deg(h)+2)\log m}{|\mathcal{C}|}$.

Proof. Given public input $(\mathsf{cm}_f, \vec{v})$, the extractor rewinds the malicious prover, aborting if $\mathcal{V}$ rejects; otherwise it obtains $(\mathsf{cm}_f, \vec{r}_1, \boldsymbol{s}_1; \vec{\boldsymbol{f}})$ and outputs $\vec{\boldsymbol{f}}$ as the extracted witness. We bound the probability that $\vec{\boldsymbol{f}}$ fails to be a valid witness for $\mathcal{R}^{\beta}_{\mathsf{poly},2}$ by considering the following events.

(1) Event 1:The polynomial constraint is false but the batched sum-check accepts, which occurs with probability $\frac{\log m}{|\mathcal{C}|}$.

(2) Event 2: Both the constraint and the sum-check fail, yet the final evaluation test accepts, which occurs with probability at most $\frac{(D'+1)\log m}{|\mathcal{C}|}$ for $D' = \deg(h)$.

□

Efficiency. Let $D' = \deg(h)$. The protocol Π_{lin} has proof size $1 + (D'+1)\log m$, prover cost $O(mD')$, and verifier cost $O(D'\log m + m)$, yielding overall linear complexity for both parties.

Parallel composition. For k instances $(\mathsf{cm}_{f_j}, \vec{\boldsymbol{v}}_j; \vec{\boldsymbol{f}}_j) \in \mathcal{R}^{\beta}_{\mathsf{poly},2}$ $(j \in [k])$, we apply Theorem A2 to run k instances of Π_{lin} concurrently using a single sum-check. This reduces communication to $k + (D'+1)\log m$ and the verifier's sum-check cost to $O(D'\log m)$. Specifically, we define $g(\vec{\boldsymbol{X}}) := \sum_{j=0}^{k-1} \alpha_j\, g_{1,j}(\vec{\boldsymbol{X}})$, where $\vec{\alpha} \leftarrow_{\$} \mathcal{C}^k$ are random challenges and

$$\forall j \in [k]: \quad g_{1,j}(\vec{\boldsymbol{X}}) := \mathsf{eq}(\vec{\mu}, \vec{\boldsymbol{X}}) \cdot \Big(h\big(\mathsf{mle}[\vec{\boldsymbol{f}}_j](\vec{\boldsymbol{X}})\big) - \mathsf{mle}[\vec{\boldsymbol{v}}_j](\vec{\boldsymbol{X}})\Big). \tag{6}$$

Accordingly, after receiving $\boldsymbol{s}_j = \langle \vec{\boldsymbol{f}}_j, \mathsf{tensor}(\vec{r}_1)\rangle$ for all $j \in [k]$, the verifier $\mathcal{V}$ checks $\boldsymbol{s}_g \stackrel{?}{=} \sum_{j=0}^{k-1} \alpha_j\, \boldsymbol{e}_\mu\big(h(\boldsymbol{s}_{1,j}) - \boldsymbol{s}_{v,j}\big)$, where $\boldsymbol{s}_g = g(\vec{r}_1)$ is the output claim and $\boldsymbol{s}_{v,j} = \langle \vec{\boldsymbol{v}}_j, \mathsf{tensor}(\vec{r}_1)\rangle$.

3.2. ℓ_2-Norm Range Check

This section shows how to reduce the ℓ_2-norm constraint to linear constraints. We decompose $\|\vec{\boldsymbol{f}}\| \le \beta$ into an IPA instance enforcing $\langle \vec{f}, \vec{f}\rangle \bmod q = B \le \beta^2$ and an ℓ_∞-norm bound $\|\vec{\boldsymbol{f}}\|_\infty \le D$ (Lemma 2). The protocol Π_{IPA} (Protocol 2) linearizes the quadratic relation, while the ℓ_∞ check follows [14].

3.2.1. Quadratic Relation Reduction

The prover constructs $\vec{a} \in \mathbb{Z}_q^{md}$ such that $\langle \vec{a}, \vec{a}\rangle = \beta^2 - B$ and $\|\vec{a}\|_\infty \le D$. By Lagrange's Four Square Theorem, the non-negative integer $\beta^2 - B$ can be written as $x_1^2 + x_2^2 + x_3^2 + x_4^2$ with non-negative integers x_i, which can be found efficiently in $O((\beta^2 - B)^{1/4})$ time. Embedding these into an md-dimensional vector and padding with zeros yields $\vec{a}$ of the required norm. Choosing $D > \sqrt{\beta^2 - B}$ ensures $\|\vec{a}\|_\infty \le D$. After computing $\vec{a}$, the prover forms the ring vector $\vec{\boldsymbol{a}}$, commits as $\mathsf{cm}_a = \mathsf{Com}(\vec{\boldsymbol{a}})$, and publishes it. Then, letting $\vec{f'} = (\vec{f}, \vec{a}) \in \mathbb{Z}_q^{2md}$ so that $\langle \vec{f'}, \vec{f'}\rangle = \beta^2$, we write

$$\vec{f'} = \begin{bmatrix} \vec{f}_0'^{\,T} \\ \vec{f}_1'^{\,T} \\ \vdots \\ \vec{f}_{2m-1}'^{\,T} \end{bmatrix} \in \mathbb{Z}_q^{2m\times d}, \quad \mathsf{mle}(\vec{f'})[\vec{\boldsymbol{X}}] = \sum_{i=0}^{2m-1} \vec{f}_i' \cdot \mathsf{eq}(\vec{\boldsymbol{X}}, \text{to-bit}(i)).$$

That is, the MLE treats each d-element block as a single unit rather than individually. Construct

$$g_2(\vec{\boldsymbol{X}}) = \langle \mathsf{mle}(\vec{f'})[\vec{\boldsymbol{X}}],\ \mathsf{mle}(\vec{f'})[\vec{\boldsymbol{X}}]\rangle. \tag{7}$$

Applying the sum-check to $\sum_{\vec{b}\in\{0,1\}^{\log 2m}} g_2(\vec{b}) = \beta^2$ reduces it to an evaluation of g_2 and ultimately to a linear constraint on $\vec{f'} = (\vec{f}, \vec{a})$.

The following Π_{IPA} reduces $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ to $\mathcal{R}^{D}_{\mathsf{lin},\infty}$ defined in Equation (8).

$$\mathcal{R}^{D}_{\mathsf{lin},\infty} := \left\{ \begin{array}{c} \mathbb{x} = \left(\begin{array}{c} (\mathsf{cm}_f, \mathsf{cm}_a) \in \mathcal{R}_q^{\kappa},\ (\vec{r}_i)_{i=\{1,2\}} \in \mathcal{C}^{\log m}, \\ (\boldsymbol{s}_1, \boldsymbol{s}_{2,f}, \boldsymbol{s}_{2,a}) \in \mathcal{R}_q \end{array} \right), \\ \mathbb{w} = (\vec{\boldsymbol{f}},\ \vec{\boldsymbol{a}}) \in (\mathcal{R}_q^m)^2,\ s.t. \\ \mathsf{Com}(\vec{\boldsymbol{f}}) = \mathsf{cm}_f,\ \mathsf{Com}(\vec{\boldsymbol{a}}) = \mathsf{cm}_a, \\ \|\vec{\boldsymbol{f}}\|_\infty \le D,\ \|\vec{\boldsymbol{a}}\|_\infty \le D, \\ \langle \vec{\boldsymbol{f}}, \mathsf{tensor}(\vec{r}_1) \rangle = \boldsymbol{s}_1,\ \begin{bmatrix} \vec{\boldsymbol{f}}^T \\ \vec{\boldsymbol{a}}^T \end{bmatrix} \cdot \mathsf{tensor}(\vec{r}_2) = \begin{bmatrix} \boldsymbol{s}_{2,f} \\ \boldsymbol{s}_{2,a} \end{bmatrix}. \end{array} \right\}. \tag{8}$$

Protocol 2. Π_{IPA}: reduce $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ to $\mathcal{R}^{D}_{\mathsf{lin},\infty}$

$\mathcal{P}(\mathsf{cm}_f, \vec{r}_1, \boldsymbol{s}_1, \vec{\boldsymbol{f}})$, $\mathcal{V}(\mathsf{cm}_f, \vec{r}_1, \boldsymbol{s}_1)$

1: $\mathcal{P}$: Construct $\vec{a} \in \mathbb{Z}_q^{md}$ and $\vec{\boldsymbol{a}} \in \mathcal{R}_q^m$.
2: $\mathcal{P} \to \mathcal{V}$: $\mathsf{cm}_a := \mathsf{Com}(\vec{\boldsymbol{a}})$.
3: $\mathcal{P}$ and $\mathcal{V}$: Construct $g_2(\vec{\boldsymbol{X}})$ as in Equation (7). Engage in a sum-check for $\sum_{\vec{b}\in\{0,1\}^{\log m+1}} \vec{g}_2(\vec{b}) = \beta^2$, which reduces to check $g_2(\vec{r}') = s'$, where $\vec{r}' \leftarrow_{\$} \mathcal{C}^{\log m+1}$ and $s' \in \mathbb{Z}_q$.
4: $\mathcal{P} \to \mathcal{V}$: $\boldsymbol{s}_{2,f} := \langle \vec{\boldsymbol{f}}, \mathsf{tensor}(\vec{r}'[0, \log m]) \rangle$, $\boldsymbol{s}_{2,a} := \langle \vec{\boldsymbol{a}}, \mathsf{tensor}(\vec{r}'[0, \log m]) \rangle$.
5: $\mathcal{V}$: Compute $\boldsymbol{s}_{f'} := \big((1 - r'_{\log m})\boldsymbol{s}_{2,f} + r_{\log m}\boldsymbol{s}_{2,a}\big)$, then check $s' \stackrel{?}{=} \langle \vec{s}_{f'}, \vec{s}_{f'} \rangle$,
where $\vec{s}_{f'}$ denotes the coefficient vector of the ring element $\boldsymbol{s}_{f'}$.
6: $\mathcal{P}$: Output $(\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}})$.
7: $\mathcal{V}$: Output $(\mathsf{cm}_f, \mathsf{cm}_a, \vec{r}_1, \vec{r}_2 = \vec{r}'[0, \log m], \boldsymbol{s}_1, \boldsymbol{s}_{2,f}, \boldsymbol{s}_{2,a})$.

Theorem 3. *Π_{IPA} is a reduction of knowledge from $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ to $\mathcal{R}^{D}_{\mathsf{lin},\infty}$.*

Proof. The public reducibility is trivial. For perfect completeness, after the sum-check, the claim reduces to $g_2(\vec{r}') = \langle \mathsf{mle}(\vec{f}')(\vec{r}'), \mathsf{mle}(\vec{f}')(\vec{r}') \rangle = s'$. Let $\boldsymbol{s}_{f'} = \langle \vec{\boldsymbol{f}}', \mathsf{tensor}(\vec{r}') \rangle$. Then $\boldsymbol{s}_{f'} = \sum_{i=0}^{d-1} s_{f'_i} X^i$, and therefore $\langle \vec{s}_{f'}, \vec{s}_{f'} \rangle = s'$.

Let $\vec{r}_2 = \vec{r}'[0{:}\log m]$. Using the block-MLE decomposition,

$$\mathsf{mle}(\vec{f}')(\vec{r}') = (1 - r'_{\log m}) \sum_{i=0}^{m-1} \vec{f}_i\, \mathsf{eq}(\text{to-bit}(i), \vec{r}_2) + r'_{\log m} \sum_{i=0}^{m-1} \vec{a}_i\, \mathsf{eq}(\text{to-bit}(i), \vec{r}_2).$$

Since $\boldsymbol{s}_{2,f} = \langle \vec{\boldsymbol{f}}, \mathsf{tensor}(\vec{r}_2) \rangle$ and $\boldsymbol{s}_{2,a} = \langle \vec{\boldsymbol{a}}, \mathsf{tensor}(\vec{r}_2) \rangle$, we obtain $\boldsymbol{s}_{f'} = (1 - r'_{\log m})\, \boldsymbol{s}_{2,f} + r'_{\log m}\, \boldsymbol{s}_{2,a}$. Thus an honest prover satisfies $\langle \vec{s}_{f'}, \vec{s}_{f'} \rangle = s'$, and therefore always passes the evaluation check. The knowledge soundness follows from the theorem below. □

Theorem 4. *When the commitment scheme $\mathsf{Com}(\cdot)$ is binding, $q > 4D^2md$, the protocol Π_{IPA} is knowledge sound with soundness error $\epsilon_{\mathsf{IPA}} = \frac{2(\log m+1)}{|\mathcal{C}|}$.*

Proof. The extractor interacts with a malicious prover $\mathcal{P}^*$ and outputs its witness in polynomial time whenever the verifier accepts. Let $\mathbb{x}_{\mathbb{i}}, \mathbb{x}_{\mathbb{o}}$ denote the input/output statements, and let $\mathbb{w}_{\mathbb{o}} = (\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}})$. If $(\mathbb{x}_{\mathbb{i}}, \mathbb{w}_{\mathbb{i}} = \vec{\boldsymbol{f}}) \notin \mathcal{R}^{\beta}_{\mathsf{lin},2}$ but $(\mathbb{x}_{\mathbb{o}}, \mathbb{w}_{\mathbb{o}}) \in \mathcal{R}^{D}_{\mathsf{lin},\infty}$, then $\|\vec{\boldsymbol{f}}\|_\infty \le D$ but $\|\vec{\boldsymbol{f}}\|_2^2 > \beta^2$. Since $\|\vec{\boldsymbol{a}}\|_\infty \le D$ and $q > 4D^2md$, we have $\langle \vec{\boldsymbol{f}}, \vec{\boldsymbol{f}} \rangle + \langle \vec{\boldsymbol{a}}, \vec{\boldsymbol{a}} \rangle < q/2$, so $\langle \vec{\boldsymbol{f}}, \vec{\boldsymbol{f}} \rangle + \langle \vec{\boldsymbol{a}}, \vec{\boldsymbol{a}} \rangle \bmod q = \langle \vec{\boldsymbol{f}}, \vec{\boldsymbol{f}} \rangle + \langle \vec{\boldsymbol{a}}, \vec{\boldsymbol{a}} \rangle > \beta^2$, and the sum-check fails. Hence, the prover passes the evaluation check with probability at most $2(\log m + 1)/|\mathcal{C}|$. □

Efficiency. The proof contains $O(\log(md))$ ring elements, the prover runs in $O(md)$ time, and the verifier in $O(\log(md))$, dominated by the sum-check phase.

3.2.2. ℓ_∞-Norm Range Check

We follow LatticeFold+ [14] for the ℓ_∞-norm check. For simplicity, we set the ℓ_∞-norm bound to $D = d' = \frac{d}{2}$. A larger bound can be handled via decomposition as in Section 4.3 of [14], which we omit here.

Representing $\vec{f}$ and $\vec{a}$ as $m \times d$ matrices, the prover forms the monomial matrices $\boldsymbol{M}_f = EXP(\vec{f})$ and $\boldsymbol{M}_a = EXP(\vec{a})$. By Lemma 3, the prover and verifier then perform the monomial check on $\boldsymbol{M}_f$ and $\boldsymbol{M}_a$ via:

$$\begin{aligned}
\vec{\boldsymbol{M}}^{v}_{f,j} &:= \left(\mathsf{ev}_{\boldsymbol{M}_{f_{0,j}}}(v),\ \mathsf{ev}_{\boldsymbol{M}_{f_{1,j}}}(v),\ \cdots,\ \mathsf{ev}_{\boldsymbol{M}_{f_{m-1,j}}}(v)\right),\\
\vec{\boldsymbol{M}}^{v'}_{f,j} &:= \left(\mathsf{ev}_{\boldsymbol{M}_{f_{0,j}}}(v^2),\ \mathsf{ev}_{\boldsymbol{M}_{f_{1,j}}}(v^2),\ \cdots,\ \mathsf{ev}_{\boldsymbol{M}_{f_{m-1,j}}}(v^2)\right),\\
\vec{\boldsymbol{M}}^{v}_{a,j} &:= \left(\mathsf{ev}_{\boldsymbol{M}_{a_{0,j}}}(v),\ \mathsf{ev}_{\boldsymbol{M}_{a_{1,j}}}(v),\ \cdots,\ \mathsf{ev}_{\boldsymbol{M}_{a_{m-1,j}}}(v)\right),\\
\vec{\boldsymbol{M}}^{v'}_{a,j} &:= \left(\mathsf{ev}_{\boldsymbol{M}_{a_{0,j}}}(v^2),\ \mathsf{ev}_{\boldsymbol{M}_{a_{1,j}}}(v^2),\ \cdots,\ \mathsf{ev}_{\boldsymbol{M}_{a_{m-1,j}}}(v^2)\right).
\end{aligned}$$

for challenge $v \leftarrow_{\$} \mathcal{C}$ and $j \in [d]$. Then, we construct

$$\begin{aligned}
g_{M_{f_j}}(\vec{\boldsymbol{X}}) &= \mathsf{mle}[\vec{\boldsymbol{M}}^{v}_{f,j}](\vec{\boldsymbol{X}})^2 - \mathsf{mle}[\vec{\boldsymbol{M}}^{v'}_{f,j}](\vec{\boldsymbol{X}}),\\
g_{M_{a_j}}(\vec{\boldsymbol{X}}) &= \mathsf{mle}[\vec{\boldsymbol{M}}^{v}_{a,j}](\vec{\boldsymbol{X}})^2 - \mathsf{mle}[\vec{\boldsymbol{M}}^{v'}_{a,j}](\vec{\boldsymbol{X}}),
\end{aligned}$$

and define

$$g_3(\vec{\boldsymbol{X}}) = \mathsf{eq}(\vec{\alpha}_1, \vec{\boldsymbol{X}}) \cdot \sum_{j\in[d]} c^j g_{M_{f_j}}(\vec{\boldsymbol{X}}) + \mathsf{eq}(\vec{\alpha}_2, \vec{\boldsymbol{X}}) \cdot \sum_{j\in[d]} c^j g_{M_{a_j}}(\vec{\boldsymbol{X}}), \tag{9}$$

for a batch challenge $c \leftarrow_{\$} \mathcal{C}$ and random challenge $\vec{\alpha}_1, \vec{\alpha}_2 \leftarrow_{\$} \mathcal{C}^{\log m}$.

Running a sum-check on the claim $\sum_{\vec{b}\in\{0,1\}^{\log m}} g_3(\vec{b}) = 0$ further reduces the problem to a set of linear constraints, which can then be leveraged to perform the ℓ_∞-norm range check. Finally, based on Lemma 4, the prover $\mathcal{P}$ and the verifier $\mathcal{V}$ then execute the protocol $\Pi_{\mathsf{rg},\infty}$ to reduce $\mathcal{R}^{D}_{\mathsf{lin},\infty}$ to the linear relation $\mathcal{R}_{\mathsf{lin}}$ defined below:

$$\mathcal{R}_{\mathsf{lin}} := \left\{ \begin{array}{c}
\mathbb{x} = \left(\begin{array}{c} (\mathsf{cm}_f, \mathsf{cm}_a) \in \mathcal{R}_q^{\kappa},\ (\mathsf{cm}_{\boldsymbol{M}_f}, \mathsf{cm}_{\boldsymbol{M}_a}) \in \mathcal{R}_q^{\kappa\times d},\\ (\boldsymbol{s}_1, \boldsymbol{s}_{2,f}, \boldsymbol{s}_{2,a}, \boldsymbol{s}_{3,f}, \boldsymbol{s}_{3,a}) \in \mathcal{R}_q \\ (\vec{r}_i)_{i\in\{1,2,3\}} \in \mathcal{C}^{\log m},\ (\vec{\boldsymbol{s}}_{M_f}, \vec{\boldsymbol{s}}_{M_a}) \in \mathcal{R}_q^{d} \end{array} \right),\\
\mathbb{w} = \left((\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}}) \in \mathcal{R}_q^{m},\ (\boldsymbol{M}_f, \boldsymbol{M}_a) \in \mathcal{R}_q^{m\times d} \right),\ s.t.\\
\mathsf{Com}(\vec{\boldsymbol{f}}) = \mathsf{cm}_f,\ \mathsf{Com}(\vec{\boldsymbol{a}}) = \mathsf{cm}_a,\\
\mathsf{Com}(\boldsymbol{M}_f) = \mathsf{cm}_{\boldsymbol{M}_f},\ \mathsf{Com}(\boldsymbol{M}_a) = \mathsf{cm}_{\boldsymbol{M}_a},\\
\langle \vec{\boldsymbol{f}}, \mathsf{tensor}(\vec{r}_1)\rangle = \boldsymbol{s}_1,\ \begin{bmatrix} \vec{\boldsymbol{f}}^T \\ \vec{\boldsymbol{a}}^T \end{bmatrix} \cdot \mathsf{tensor}(\vec{r}_2) = \begin{bmatrix} \boldsymbol{s}_{2,f} \\ \boldsymbol{s}_{2,a} \end{bmatrix},\\
\begin{bmatrix} \vec{\boldsymbol{f}}^T \\ \vec{\boldsymbol{a}}^T \end{bmatrix} \cdot \mathsf{tensor}(\vec{r}_3) = \begin{bmatrix} \boldsymbol{s}_{3,f} \\ \boldsymbol{s}_{3,a} \end{bmatrix},\ \begin{bmatrix} \boldsymbol{M}_f^T \\ \boldsymbol{M}_a^T \end{bmatrix} \cdot \mathsf{tensor}(\vec{r}_3) = \begin{bmatrix} \vec{\boldsymbol{s}}_{M_f} \\ \vec{\boldsymbol{s}}_{M_a} \end{bmatrix}.
\end{array} \right\}. \tag{10}$$

The complete $\Pi_{\mathsf{rg},\infty}$ is provided in Appendix B.

Theorem 5. *$\Pi_{\mathsf{rg},\infty}$ is a reduction of knowledge from $\mathcal{R}^{D}_{\mathsf{lin},\infty}$ to $\mathcal{R}_{\mathsf{lin}}$. When the commitment scheme $\mathsf{Com}(\cdot)$ is binding, and $D = d' = \frac{d}{2}$, $\Pi_{\mathsf{rg},\infty}$ is knowledge sound with soundness error $\epsilon_{\mathsf{rg},\infty} = \frac{3d+5\log m}{|\mathcal{C}|}$.*

Proof. The proof closely follows the proofs in Sections 4.2 and 4.3 of LatticeFold+. The full details are provided in Appendix B. □

Theorem 6. *By sequentially composing Π_{IPA} and $\Pi_{\mathsf{rg},\infty}$ to obtain $\Pi_{\mathsf{rg}} = \Pi_{\mathsf{IPA}} \diamond \Pi_{\mathsf{rg},\infty}$, Theorem 3, 5 and A1 guarantee that Π_{rg} is a reduction of knowledge from $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ to $\mathcal{R}_{\mathsf{lin}}$, with knowledge soundness error $\epsilon_{\mathsf{rg}} = \epsilon_{\mathsf{IPA}} + \epsilon_{\mathsf{rg},\infty}$.*

Efficiency. The proof includes $O(3\log m)$ ring elements from sum-check plus $2(\kappa d + 2d)$ more, with linear-time proving and logarithmic-time verification.

3.3. Inner Linear Constraints Batching

This section batches all linear constraints inside $\mathcal{R}_{\mathsf{lin}}$ into a single one, reducing $\mathcal{R}_{\mathsf{lin}}$ to $\mathcal{R}_{\mathsf{lin,ba}}$ defined as follows:

$$\mathcal{R}_{\mathsf{lin,ba}} := \left\{ \begin{array}{c}
\mathbb{x} = \left(\begin{array}{c} (\mathsf{cm}_f, \mathsf{cm}_a) \in \mathcal{R}_q^{\kappa},\ (\mathsf{cm}_{\boldsymbol{M}_f}, \mathsf{cm}_{\boldsymbol{M}_a}) \in \mathcal{R}_q^{\kappa\times d},\\ \vec{r} \in \mathcal{C}^{\log m},\ \vec{\boldsymbol{s}} \in \mathcal{R}_q^{2d+2} \end{array} \right),\\
\mathbb{w} = \left((\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}}) \in \mathcal{R}_q^{m},\ (\boldsymbol{M}_f, \boldsymbol{M}_a) \in \mathcal{R}_q^{m\times d} \right),\ s.t.\\
\mathsf{Com}(\vec{\boldsymbol{f}}) = \mathsf{cm}_f,\ \mathsf{Com}(\vec{\boldsymbol{a}}) = \mathsf{cm}_a,\\
\mathsf{Com}(\boldsymbol{M}_f) = \mathsf{cm}_{\boldsymbol{M}_f},\ \mathsf{Com}(\boldsymbol{M}_a) = \mathsf{cm}_{\boldsymbol{M}_a},\\
\mathsf{tensor}(\vec{r})^T \cdot \left[\vec{\boldsymbol{f}},\quad \vec{\boldsymbol{a}},\quad \boldsymbol{M}_f,\quad \boldsymbol{M}_a \right] = \vec{\boldsymbol{s}}^T.
\end{array} \right\}, \tag{11}$$

The protocol Π_{ba} is given in Protocol 3. By applying MLEs on $\vec{\boldsymbol{f}}$, $\vec{\boldsymbol{a}}$, and on each column $\vec{\boldsymbol{M}}_{f,j}$, $\vec{\boldsymbol{M}}_{a,j}$ for $j \in [d]$, we construct

$$\begin{aligned} g_4(\vec{\boldsymbol{X}}) = \mathsf{mle}[\vec{\boldsymbol{f}}](\vec{\boldsymbol{X}}) \cdot \mathsf{eq}(\vec{r}_1, \vec{\boldsymbol{X}}) + \\ \left(c\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{\boldsymbol{X}}) + c^2\mathsf{mle}[\vec{\boldsymbol{a}}](\vec{\boldsymbol{X}})\right) \cdot \mathsf{eq}(\vec{r}_2, \vec{\boldsymbol{X}}) + \\ \left(c^3\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{\boldsymbol{X}}) + c^4\mathsf{mle}[\vec{\boldsymbol{a}}](\vec{\boldsymbol{X}})\right) \cdot \mathsf{eq}(\vec{r}_3, \vec{\boldsymbol{X}}) + \\ \left(\sum_{j=0}^{d-1} c^{j+5}\mathsf{mle}[\vec{\boldsymbol{M}}_{f,j}](\vec{\boldsymbol{X}}) + \sum_{j=0}^{d-1} c^{j+d+5}\mathsf{mle}[\vec{\boldsymbol{M}}_{a,j}](\vec{\boldsymbol{X}})\right) \cdot \mathsf{eq}(\vec{r}_3, \vec{\boldsymbol{X}}), \end{aligned} \tag{12}$$

where $c \leftarrow_{\$} \mathcal{C}$ is the batching challenge sent by the verifier. Let

$$\begin{aligned} \boldsymbol{v} = \boldsymbol{s}_1 + c \cdot \boldsymbol{s}_{2,f} + c^2 \cdot \boldsymbol{s}_{2,a} + c^3 \cdot \boldsymbol{s}_{3,f} + c^4 \cdot \boldsymbol{s}_{3,a} \\ + \sum_{j=0}^{d-1} c^{j+5} \cdot \boldsymbol{s}_{M_{f,j}} + \sum_{j=0}^{d-1} c^{j+d+5} \cdot \boldsymbol{s}_{M_{a,j}}. \end{aligned} \tag{13}$$

Engaging in the sum-check $\sum_{\vec{b}\in\{0,1\}^{\log m}} g(\vec{b}) = \boldsymbol{v}$, reduces the proof to linear claims over the unified vector $\vec{r}$.

Protocol 3. Π_{ba}: Reducing $\mathcal{R}_{\mathsf{lin}}$ to $\mathcal{R}_{\mathsf{lin,ba}}$

$\mathbb{x} = \begin{pmatrix} (\mathsf{cm}_f, \mathsf{cm}_a, \mathsf{cm}_{\boldsymbol{M}_f}, \mathsf{cm}_{\boldsymbol{M}_a}), (\vec{r}_i)_{i\in\{1,2,3\}} \\ (\boldsymbol{s}_1, (\boldsymbol{s}_{2,f}, \boldsymbol{s}_{3,f}), (\boldsymbol{s}_{2,a}, \boldsymbol{s}_{3,a}), (\vec{\boldsymbol{s}}_{M_f}, \vec{\boldsymbol{s}}_{M_a})) \end{pmatrix}$, $\mathbb{w} = \left(\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}}, \boldsymbol{M}_f, \boldsymbol{M_a}\right)$.
$\mathcal{P}(\mathbb{x}, \mathbb{w})$, $\mathcal{V}(\mathbb{x})$.

1: $\mathcal{V} \to \mathcal{P}$: $c \leftarrow_{\$} \mathcal{C}$.
2: $\mathcal{P}$ and $\mathcal{V}$: Construct $g_4(\vec{\boldsymbol{X}})$ as in Equation (12), and perform a sum-check on $\sum_{\vec{b}\in\{0,1\}^{\log m}} g_4(\vec{b}) = \boldsymbol{v}$, where $\boldsymbol{v}$ is defined in Equation (13), reducing to verify $g_4(\vec{r}) = \boldsymbol{s}_{g_4}$ for $\vec{r} \leftarrow_{\$} \mathcal{C}^{\log m}$.
3: $\mathcal{P} \to \mathcal{V}$: $\vec{\boldsymbol{s}} = (\boldsymbol{s}_0, \boldsymbol{s}_1, \cdots, \boldsymbol{s}_{2d+1}) := \begin{bmatrix} \vec{\boldsymbol{f}}^T \\ \vec{\boldsymbol{a}}^T \\ \boldsymbol{M}_f^T \\ \boldsymbol{M}_a^T \end{bmatrix} \cdot \mathsf{tensor}(\vec{r})$.
4: $\mathcal{V}$: Compute $\boldsymbol{e}_i := \mathsf{eq}(\vec{r}, \vec{r}_i)$ for $i \in \{1, 2, 3\}$, and check

$$\boldsymbol{s}_{g_4} \stackrel{?}{=} \boldsymbol{s}_0\boldsymbol{e}_1 + (c\boldsymbol{s}_0 + c^2\boldsymbol{s}_1)\boldsymbol{e}_2 + \left(c^3\boldsymbol{s}_0 + c^4\boldsymbol{s}_1 + \sum_{j=0}^{d-1} c^{j+5}(\boldsymbol{s}_{j+2} + c^d\boldsymbol{s}_{j+2+d})\right)\boldsymbol{e}_3.$$

5: $\mathcal{P}$: Output $(\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}}, \boldsymbol{M}_f, \boldsymbol{M}_a)$.
6: $\mathcal{V}$: Output $\left((\mathsf{cm}_f, \mathsf{cm}_a, \mathsf{cm}_{\boldsymbol{M}_f}, \mathsf{cm}_{\boldsymbol{M}_a}), \vec{r}, \vec{\boldsymbol{s}}\right)$.

Theorem 7. Π_{ba} *is a reduction of knowledge from* $\mathcal{R}_{\mathsf{lin}}$ *to* $\mathcal{R}_{\mathsf{lin,ba}}$. *Specifically, when the commitment scheme* $\mathsf{Com}(\cdot)$ *is binding, the protocol* Π_{ba} *achieves knowledge soundness with soundness error* $\epsilon_{\mathsf{ba}} = \frac{2d+5+2\log m}{|\mathcal{C}|}$.

Proof. Public reducibility is straightforward, and perfect completeness follows from that of the sum-check protocol. For knowledge soundness, the case where the output $\mathbb{w}$ satisfies $\mathcal{R}_{\mathsf{lin,ba}}$ but not $\mathcal{R}_{\mathsf{lin}}$ can occur in two cases: (i). The batched sum-check holds, which occurs with probability $\frac{2d+5}{|\mathcal{C}|}$. (ii). The batched sum-check fails, but the evaluation check passes, which occurs with probability $\frac{2\log m}{|\mathcal{C}|}$. □

Efficiency. The proof consists of $O(\log m)$ ring elements, together with an additional $2d + 2$ ring elements, with linear prover complexity $O(dm)$ and logarithmic verifier complexity $O(\log m)$.

3.4. Outer Linear Constraints Batching

This section presents a batching method for linear constraints across witnesses in $\mathcal{R}_{\mathsf{lin,ba}}$ and provides a toolkit reducing $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ to the following relation, used as a modular component in the amortization phase of our compressed Σ-protocol:

$$\mathcal{R}^{\beta'}_{\mathsf{lin,f}} = \left\{ \begin{array}{c} \mathbb{x} = \left(\mathsf{cm}_y \in \mathcal{R}_q^{\kappa},\ \vec{r}_0 \in \mathcal{C}^{\log m},\ \boldsymbol{s}_y \in \mathcal{R}_q\right),\ \mathbb{w} = \left(\vec{\boldsymbol{y}} \in \mathcal{R}_q^m\right),\ s.t. \\ \mathsf{Com}(\vec{\boldsymbol{y}}) = \mathsf{cm}_y, \quad \langle \vec{\boldsymbol{y}}, \mathsf{tensor}(\vec{r}_0)\rangle = \boldsymbol{s}_y, \quad \|\vec{\boldsymbol{y}}\| \le \beta'. \end{array} \right\}, \tag{14}$$

for $\beta' \geq 2Td\sqrt{m} + T\beta$, where T is the expansion factor from Definition 1.

The reduction first runs Π_{IPA}, $\Pi_{\mathsf{rg},\infty}$, and Π_{ba} to obtain the relation $\mathcal{R}_{\mathsf{lin,ba}}$, after which the verifier checks the linear claim in $\mathcal{R}_{\mathsf{lin,ba}}$ via a batched sum-check and folds the witnesses using its sampled challenges. The complete protocol is presented in Protocol 4. Here, we slightly abuse the notation mle by letting $\mathsf{mle}[\vec{a}+\vec{b}](\vec{\boldsymbol{X}}) = \mathsf{mle}[\vec{a}](\vec{\boldsymbol{X}}) + \mathsf{mle}[\vec{b}](\vec{\boldsymbol{X}})$.

Protocol 4. Π_{tool}: Reducing $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ to $\mathcal{R}^{\beta'}_{\mathsf{lin,f}}$

$\mathcal{P}((\mathsf{cm}_f, \vec{r}_1, \boldsymbol{s}_1), \vec{\boldsymbol{f}}), \quad \mathcal{V}(\mathsf{cm}_f, \vec{r}_1, \boldsymbol{s}_1)$.

1: $\mathcal{P}$ and $\mathcal{V}$: Run Π_{IPA} to reduce to $\mathcal{R}^{D}_{\mathsf{lin},\infty}$.
2: $\mathcal{P}$ and $\mathcal{V}$: Run $\Pi_{\mathsf{rg},\infty}$ to reduce to $\mathcal{R}_{\mathsf{lin}}$.
3: $\mathcal{P}$ and $\mathcal{V}$: Run Π_{ba} to reduce to $\mathcal{R}_{\mathsf{lin,ba}}$.
4: $\mathcal{V} \to \mathcal{P}$: $\vec{b} \leftarrow_{\$} \mathcal{C}'^{2d}$.
5: $\mathcal{P} \to \mathcal{V}$: $\mathsf{Com}(\vec{\boldsymbol{m}}_f) := \mathsf{cm}_{M_f} \cdot \vec{b}[0,d]$, $\mathsf{Com}(\vec{\boldsymbol{m}}_a) := \mathsf{cm}_{M_a} \cdot \vec{b}[d,2d]$.
6: $\mathcal{V} \to \mathcal{P}$: $c \leftarrow_{\$} \mathcal{C}$.
7: $\mathcal{P}$ and $\mathcal{V}$: Construct $g(\vec{\boldsymbol{X}}) = \left(\mathsf{mle}[\vec{\boldsymbol{f}} + c\vec{\boldsymbol{a}} + c^2\vec{\boldsymbol{m}}_f + c^3\vec{\boldsymbol{m}}_a](\vec{\boldsymbol{X}})\right) \cdot \mathsf{eq}(\vec{r}, \vec{\boldsymbol{X}})$, and perform a sum-check on $\sum_{\vec{b}\in\{0,1\}^{\log m}} g(\vec{b}) = \langle \vec{\boldsymbol{s}}, \vec{p}\rangle$, where $\vec{p} = (1, c, c^2 \cdot \vec{b}[0,d], c^3 \cdot \vec{b}[d,2d])$. This reduces to verify $g(\vec{r}_0) = \boldsymbol{s}_g$ for $\vec{r}_0 \leftarrow_{\$} \mathcal{C}^{\log m}$.
8: $\mathcal{P} \to \mathcal{V}$: $\vec{\boldsymbol{s}}_0 = (\boldsymbol{s}_0, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3) := \begin{bmatrix} \vec{\boldsymbol{f}}^T \\ \vec{\boldsymbol{a}}^T \\ \vec{\boldsymbol{m}}_f^T \\ \vec{\boldsymbol{m}}_a^T \end{bmatrix} \cdot \mathsf{tensor}(\vec{r}_0)$.
9: $\mathcal{V}$: Compute $\boldsymbol{e}_0 := \mathsf{eq}(\vec{r}, \vec{r}_0)$, and check $\boldsymbol{s}_g \stackrel{?}{=} \left(\sum_{i=0}^{3} c^i \boldsymbol{s}_{0,i}\right) \cdot \boldsymbol{e}_0$.
10: $\mathcal{V} \to \mathcal{P}$: The folding challenge $\vec{c}_f \leftarrow_{\$} \mathcal{C}'^2$.
11: $\mathcal{P}$: Output $\vec{\boldsymbol{y}} := c_{f,0}\vec{\boldsymbol{f}} + c_{f,1}\vec{\boldsymbol{a}} + \vec{\boldsymbol{m}}_f + \vec{\boldsymbol{m}}_a$.
12: $\mathcal{V}$: Output $\left(\mathsf{cm}_y, \boldsymbol{s}_y\right)$, where $\mathsf{cm}_y := c_{f,0}\mathsf{cm}_f + c_{f,1}\mathsf{cm}_a + \mathsf{Com}(\vec{\boldsymbol{m}}_f) + \mathsf{Com}(\vec{\boldsymbol{m}}_a)$,
$\boldsymbol{s}_y := c_{f,0}\boldsymbol{s}_{0,0} + c_{f,1}\boldsymbol{s}_{0,1} + \boldsymbol{s}_{0,2} + \boldsymbol{s}_{0,3}$.

Theorem 8. *When* $\beta' \geq 2Td \cdot \sqrt{m} + 2T \cdot \beta$ *and* $\mathsf{Com}(\cdot)$ *is binding for norm* $\max(2\beta', 4T\beta')$, Π_{tool} *is a reduction of knowledge from* $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ *to* $\mathcal{R}^{\beta'}_{\mathsf{lin,f}}$.

Proof. Public reducibility is trivial. For perfect completeness, the completeness of Π_{IPA}, $\Pi_{\mathsf{rg},\infty}$, Π_{ba}, and the sum-check protocol guarantees that an honest prover can pass the check.

For the norm bound on the output witness, we have:

$$\|\vec{\boldsymbol{y}}\|_2 \leq T \cdot \|\vec{\boldsymbol{f}}\|_2 + T \cdot \|\vec{\boldsymbol{a}}\|_2 + 2Td \cdot \sqrt{m} \leq 2T\beta + 2Td \cdot \sqrt{m}.$$

The knowledge soundness follows the theorems below. □

Theorem 9. *When* $\beta' \geq 2Td \cdot \sqrt{m} + T \cdot \beta$ *and* $\mathsf{Com}(\cdot)$ *is binding for norm* $\max(2\beta', 4T\beta')$, Π_{tool} *is knowledge sound with soundness error* $\epsilon_{\mathsf{tool}} = \epsilon_{\mathsf{tool,cwss}} + \frac{2d+2}{|\mathcal{C}'|}$, *for* $\epsilon_{\mathsf{tool,cwss}} = \epsilon_{\mathsf{ba}} + \epsilon_{\mathsf{rg}} + \frac{2\log m + 2d + 3}{|\mathcal{C}|}$.

A full proof is provided in Appendix C.

Efficiency. The proof contains $O(\log m)$ ring elements from the sum-check plus $2d+2$ extras, with prover time $O(m)$ and verifier time $O(\log m)$.

3.5. Achieving Zero-Knowledge

Since the plain claim $\langle \vec{\boldsymbol{f}}_j, \mathsf{tensor}(\vec{r})\rangle$ and the sum-check transcript may leak information about $\vec{\boldsymbol{f}}_j$, we employ zero-knowledge sum-check and zero-knowledge claim techniques. To hide $\vec{\boldsymbol{f}}_j \in \mathcal{R}_q^m$ inside a claim and the output of a sum-check, we sample $\vec{r}_f \leftarrow_{\$} \mathcal{R}_q^m$ and set $\vec{\boldsymbol{f}}_{j,r} := (\vec{\boldsymbol{f}}_j, \vec{r}_f)$, yielding the protected claim

$$\langle \vec{\boldsymbol{f}}_{j,r}, \mathsf{tensor}(\vec{r}_{j,r})\rangle = (1 - r_{\log m}) \langle \vec{\boldsymbol{f}}_j, \mathsf{tensor}(\vec{r})\rangle + r_{\log m} \langle \vec{r}_f, \mathsf{tensor}(\vec{r})\rangle,$$

with $\vec{r}_{j,r} = (\vec{r}, r_{\log m}) \in \mathcal{C}^{\log m+1}$. Note that $\|\vec{r}_f\|$ may be large, so it is excluded from both the norm constraint and the commitment.

To obtain zero-knowledge in the sum-check, we adapt the masking technique of [22] to rings. For a ring $\bar{\mathcal{R}}$ and a multilinear $g(\vec{\boldsymbol{X}}) \in \bar{\mathcal{R}}^{\leq 1}[\boldsymbol{X}_0, \ldots, \boldsymbol{X}_{\mu-1}]$, define the masking polynomial

$$g_r(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{\mu-1}) = \boldsymbol{a} + \boldsymbol{a}_0\boldsymbol{x}_0 + \cdots + \boldsymbol{a}_{\mu-1}\boldsymbol{x}_{\mu-1}, \quad \boldsymbol{a}, \boldsymbol{a}_i \leftarrow_{\$} \bar{\mathcal{R}}.$$

Let $\boldsymbol{v} = \sum_{\vec{b}} g(\vec{b})$ and $\boldsymbol{v}_r = \sum_{\vec{b}} g_r(\vec{b})$. Given $\zeta \leftarrow_{\$} \mathcal{C}$, the parties run sum-check on $\boldsymbol{v} + \zeta\boldsymbol{v}_r = \sum_{\vec{b}}(g(\vec{b}) + \zeta g_r(\vec{b}))$. The verifier receives a claim on $g(\vec{r}) + \zeta g_r(\vec{r})$, after which the prover reveals $g_r(\vec{r})$ to recover $g(\vec{r})$. Since ζ is uniform, zero-knowledge follows as in Theorem 3 of [22].

Zero-Knowledge Linearization

We illustrate how to transform Protocol 1 into a zero-knowledge variant by combining masked evaluation claims with zero-knowledge sum-checks.

The prover samples $\vec{r}_f \leftarrow_{\$} \mathcal{R}_q^m$, while the verifier samples $\vec{r}_v \leftarrow_{\$} \mathcal{R}_q^m$ and sends it to the prover. They define the extended vectors $\vec{\boldsymbol{f}}_r = (\vec{\boldsymbol{f}}, \vec{r}_f)$ and $\vec{\boldsymbol{v}}_r = (\vec{\boldsymbol{v}}, \vec{r}_v)$.

To hide the sum-check transcript, the prover additionally samples a masking polynomial $g_r(\boldsymbol{Y}, \vec{\boldsymbol{X}})$ of sufficient individual degree, following the zero-knowledge sum-check technique of [22]. The polynomial in Step 2 of Protocol 1 becomes

$$g_{1,r}(\boldsymbol{Y}, \vec{\boldsymbol{X}}) = (1 - \boldsymbol{Y})\mathsf{eq}(\vec{\mu}, \vec{\boldsymbol{X}}) \cdot \Big(h(\mathsf{mle}[\vec{\boldsymbol{f}}_r](\boldsymbol{Y}, \vec{\boldsymbol{X}})) - \mathsf{mle}[\vec{\boldsymbol{v}}_r](\boldsymbol{Y}, \vec{\boldsymbol{X}})\Big) + \zeta g_r(\boldsymbol{Y}, \vec{\boldsymbol{X}}), \tag{15}$$

where ζ is the verifier's random masking challenge.

Since $h(\boldsymbol{f}_i) = \boldsymbol{v}_i$ for all $i \in [m]$, the factor $(1 - \boldsymbol{Y})$ ensures that

$$\sum_{\substack{\vec{b} \in \{0,1\}^\ell \\ c \in \{0,1\}}} (1 - c)\mathsf{eq}(\vec{\mu}, \vec{b})\Big(h(\mathsf{mle}[\vec{\boldsymbol{f}}_r](c, \vec{b})) - \mathsf{mle}[\vec{\boldsymbol{v}}_r](c, \vec{b})\Big) = 0.$$

Consequently, the prover and verifier execute sum-check on the claim

$$\sum_{\substack{\vec{b} \in \{0,1\}^\ell \\ c \in \{0,1\}}} g_{1,r}(c, \vec{b}) = \boldsymbol{v}_{1,r}, \tag{16}$$

where $\boldsymbol{v}_{1,r} := \zeta \sum_{\substack{\vec{b} \in \{0,1\}^\ell \\ c \in \{0,1\}}} g_r(c, \vec{b})$.

The sum-check protocol reduces Equation (16) to the evaluation claim $g_{1,r}(\vec{r}_1, r_{\log m}) = \boldsymbol{s}_{g_{1,r}}$, where $\vec{r}_1$ and $r_{\log m}$ are verifier challenges.

The prover sends

$$\boldsymbol{s}_f = (1 - r_{\log m})\langle \vec{\boldsymbol{f}}, \mathsf{tensor}(\vec{r}_1)\rangle + r_{\log m}\langle \vec{r}_f, \mathsf{tensor}(\vec{r}_1)\rangle$$

and $\boldsymbol{s}_r = g_r(\vec{r}_1, r_{\log m})$. Since both $\vec{\boldsymbol{v}}$ and $\vec{r}_v$ are public, the verifier independently computes

$$\boldsymbol{s}_v = (1 - r_\ell)\langle \vec{\boldsymbol{v}}, \mathsf{tensor}(\vec{r}_1)\rangle + r_\ell\langle \vec{r}_v, \mathsf{tensor}(\vec{r}_1)\rangle$$

and $\boldsymbol{e}_\mu = \mathsf{eq}(\vec{\mu}, \vec{r}_1)$. Finally, the verifier checks

$$\boldsymbol{s}_{g_{1,r}} \stackrel{?}{=} (1 - r_\ell)\boldsymbol{e}_\mu\big(h(\boldsymbol{s}_f) - \boldsymbol{s}_v\big) + \zeta\boldsymbol{s}_r. \tag{17}$$

We denote the resulting protocol by $\Pi_{\mathsf{zk\text{-}lin}}$.

Theorem 10. $\Pi_{\mathsf{zk\text{-}lin}}$ *realizes the same reduction relation as* Π_{lin}*, with perfect completeness, knowledge soundness up to the additional errors of the auxiliary consistency checks, and honest-verifier zero knowledge.*

Proof. *Perfect completeness.* For every $\vec{b} \in \{0,1\}^\ell$, the original relation implies

$$h(\mathsf{mle}[\vec{\boldsymbol{f}}_r](\vec{b}, 0)) = \mathsf{mle}[\vec{\boldsymbol{v}}_r](\vec{b}, 0).$$

At $c = 1$, the factor $(1 - c)$ vanishes. Therefore,

$$\sum_{\vec{b},c} g_{1,r}(\vec{b}, c) = \zeta \sum_{\vec{b},c} g_r(\vec{b}, c) = \boldsymbol{v}_{1,r}.$$

Hence, an honest prover satisfies the sum-check and final evaluation checks.

Knowledge soundness. The soundness of sum-check reduces Equation (16) to

$$\sum_{\vec{b}\in\{0,1\}^\ell} \mathsf{eq}(\vec{\mu}, \vec{b})\big(h(\boldsymbol{f}_{\vec{b}}) - \boldsymbol{v}_{\vec{b}}\big) = 0,$$

except with the corresponding soundness error. The original knowledge-soundness argument of Theorem 1 then applies, together with the additional errors of the auxiliary consistency checks.

Honest-verifier zero knowledge. Fix the verifier challenges $(\vec{r}_1, r_{\log m})$ with $r_{\log m} \in \mathcal{R}_q^\times$. Since $\vec{r}_f$ is uniform over $\mathcal{R}_q^m$, the evaluation $\langle \vec{r}_f, \mathsf{tensor}(\vec{r}_1)\rangle$ is uniform over $\mathcal{R}_q$. Multiplication by the element $r_{\log m}$ preserves uniformity, and hence $\boldsymbol{s}_f \sim U(\mathcal{R}_q)$.

Moreover, $\boldsymbol{s}_v$ is determined entirely by public information and verifier randomness, and therefore reveals no private witness information.

During the sum-check phase, the intermediate polynomial coefficients are masked by the independently sampled polynomial g_r, thus can be simulated without knowledge of the original witness. □

Replacing all claims and sum-checks in Π_{lin}, Π_{IPA}, $\Pi_{\mathsf{rg},\infty}$, Π_{ba}, and Π_{tool} with their zero-knowledge counterparts yields the protocols $\Pi_{\mathsf{zk\text{-}lin}}$, $\Pi_{\mathsf{zk\text{-}IPA}}$, $\Pi_{\mathsf{zk\text{-}rg},\infty}$, $\Pi_{\mathsf{zk\text{-}ba}}$, and $\Pi_{\mathsf{zk\text{-}tool}}$.

Theorem 11. *The protocols* $\Pi_{\mathsf{zk\text{-}lin}}$, $\Pi_{\mathsf{zk\text{-}IPA}}$, $\Pi_{\mathsf{zk\text{-}rg},\infty}$, $\Pi_{\mathsf{zk\text{-}ba}}$, *and* $\Pi_{\mathsf{zk\text{-}tool}}$ *are reductions of knowledge that perform the same reductions as* Π_{lin}, Π_{IPA}, $\Pi_{\mathsf{rg},\infty}$, Π_{ba}, *and* Π_{tool}, *respectively.*

Proof. The claim follows from Theorems 1, 3, 5, 7, 8, together with the completeness and knowledge soundness of the sum-check protocol. □

4. Efficient ΣProtocol for Polynomial Relations

In this section, we build an efficient lattice-based Σ-protocol for proving k instances of $\mathcal{R}^\beta_{\mathsf{poly},2}$ using the toolbox from Section 3. Following the compressed Σ-protocol of [7], we amortize the k instances into a single one and verify it with a 3-move Σ-protocol, thus avoiding repeated proofs and achieving essentially the cost of a single instance.

4.1. Amortization

We first run $\Pi_{\mathsf{zk\text{-}lin}}$ in parallel on k instances $(\mathsf{cm}_{f_j}, \vec{\boldsymbol{v}}_j; \vec{\boldsymbol{f}}_j) \in \mathcal{R}^\beta_{\mathsf{poly},2}$, reducing to $(\mathsf{cm}_{f_j}, \vec{r}_1, \boldsymbol{s}_{1,j}; \vec{\boldsymbol{f}}_j) \in \mathcal{R}^\beta_{\mathsf{lin},2}$ for all $j \in [k]$. We then amortize these $\mathcal{R}^\beta_{\mathsf{lin},2}$ instances using the split-and-fold technique of [14, 15], which decomposes each witness and folds the pieces into a reduced-norm vector. Protocol 5 summarizes this phase, reducing $(\mathcal{R}^\beta_{\mathsf{poly},2})^k$ to $\mathcal{R}^{\mathcal{B}'}_{\mathsf{lin,f}}$ with $\mathcal{B}' \geq \frac{\sigma}{2}\sqrt{md} \cdot Tk\iota$. Here T is the expansion factor (Definition 1), $\iota = \lceil \log_\sigma \beta' \rceil$ is the number of decomposition pieces.

Theorem 12. *When* $\mathcal{B}' \geq \frac{\sigma}{2}\sqrt{md}\,Tk\iota$ *and* $\mathsf{Com}(\cdot)$ *is binding for the norm* $\max(2\mathcal{B}', 4T\mathcal{B}', 2\mathcal{B}'\beta', 4T\mathcal{B}'\beta')$, Π_{amor} *gives a knowledge reduction from* $(\mathcal{R}^\beta_{\mathsf{poly},2})^k$ *to* $\mathcal{R}^{\mathcal{B}'}_{\mathsf{lin,f}}$ *with knowledge soundness error* $\epsilon_{\mathsf{lin}} + \epsilon_{\mathsf{tool}} + \frac{6k+2\log m}{|\mathcal{C}|} + \frac{2k\iota}{|\mathcal{C}'|}$.

The complete proof is given in Appendix D.

4.2. Compressed Σ-Protocol

4.2.1. Σ-Protocol

After amortization, we can employ a 3-move Σ-protocol to prove $\mathcal{R}^{\mathcal{B}'}_{\mathsf{lin,f}}$, which proceeds as in Protocol 6.

Theorem 13. *For the ring* $\mathcal{R}_q$, *let the public parameters be* $\mathsf{pp} := \big(\boldsymbol{A} \in \mathcal{R}_q^{\kappa\times m},\ (\beta, \beta_t, \beta', \mathcal{B}', \gamma) \in \mathbb{Z}_q\big)$ *such that*

$$\beta' \geq 2Td \cdot \sqrt{m} + 2T \cdot \beta, \quad \mathcal{B}' \geq \frac{\sigma}{2}\sqrt{md} \cdot Tk\iota, \quad \gamma \geq \beta_t + T\mathcal{B}',$$

Suppose the MSIS *problem is hard for rank* κ *and norm bound* $\max(2\gamma, 4T\gamma, 2\mathcal{B}'\beta', 4T\mathcal{B}'\beta')$*, where* T *is the expansion factor of* $\mathcal{C}'$*. Then* Π_Σ *satisfies completeness, special soundness with soundness error* $\epsilon_\Sigma = \epsilon_{\mathsf{amor}} + \frac{1}{|\mathcal{C}'|}$*, and special honest-verifier zero-knowledge.*

Proof. Completeness follows directly from Π_{amor}. For special soundness, two accepting transcripts let the extractor recover a relaxed opening $\tilde{\vec{\boldsymbol{y}}}$ with $\mathsf{Com}(\tilde{\vec{\boldsymbol{y}}}) = \mathsf{cm}_y$ and $\|\bar{c}\tilde{\vec{\boldsymbol{y}}}\| \le 2\gamma$. Feeding $\tilde{\vec{\boldsymbol{y}}}$ into the extractor of Π_{amor}, leveraging its knowledge soundness with the binding of $\mathsf{Com}(\cdot)$ at norm $\max(2\gamma, 4T\gamma, 2\mathcal{B}'\beta', 4T\mathcal{B}'\beta')$, yields valid witnesses for the original relation. HVZK follows from the zero-knowledge of all sub-protocols together with the masking and rejection-sampling steps of the Σ-protocol. □

Protocol 5. Π_{amor}: Reducing $(\mathcal{R}^{\beta}_{\mathsf{poly},2})^k$ to $\mathcal{R}^{\mathcal{B}'}_{\mathsf{lin},\mathsf{f}}$

$\mathcal{P}(\mathsf{cm}_{f_i}, \vec{\boldsymbol{v}}_i, \vec{\boldsymbol{f}}_i)_{i\in[k]}$, $\mathcal{V}(\mathsf{cm}_{f_i}, \vec{\boldsymbol{v}}_i)_{i\in[k]}$.

1: $\mathcal{P}, \mathcal{V}$: Run k parallel $\Pi_{\mathsf{zk\text{-}lin}}$, obtaining $((\mathsf{cm}_{f,i}, \vec{r}_1, \boldsymbol{s}_{1,i}), \vec{\boldsymbol{f}}_i) \in \mathcal{R}^{\beta}_{\mathsf{lin},2}$ for $i \in [k]$.
2: $\mathcal{P}, \mathcal{V}$: Run k parallel $\Pi_{\mathsf{zk\text{-}tool}}$, reducing to $((\mathsf{cm}_{y,i}, \vec{r}_0, \boldsymbol{s}_{y,i}), \vec{\boldsymbol{y}}_i) \in \mathcal{R}^{\beta'}_{\mathsf{lin},\mathsf{f}}$.
3: $\mathcal{V} \to \mathcal{P}$: $c \leftarrow_{\$} \mathcal{C}$.
4: $\mathcal{P}, \mathcal{V}$: Set $g_a(\vec{\boldsymbol{X}}) = \sum_{i=0}^{k-1} c^i\, \mathsf{mle}[\vec{\boldsymbol{y}}_i](\vec{\boldsymbol{X}}) \cdot \mathsf{eq}(\vec{r}_0, \vec{\boldsymbol{X}})$, and run a zero-knowledge sum-check on $\sum_{\vec{b}\in\{0,1\}^{\log m}} g_a(\vec{b}) = \sum_{i=0}^{k-1} c^i\, \boldsymbol{s}_{y_i}$, reducing to an evaluation claim $g_a(\vec{r}_a) = \boldsymbol{s}_a$ for $\vec{r}_a \leftarrow_{\$} \mathcal{C}^{\log m}$.
5: $\mathcal{P} \to \mathcal{V}$: $\{\boldsymbol{s}^a_{y_i} := \langle \vec{\boldsymbol{y}}_i, \mathsf{tensor}(\vec{r}_a)\rangle\}_{i\in[k]}$.
6: $\mathcal{V}$: Compute $\boldsymbol{e}_a := \mathsf{eq}(\vec{r}_0, \vec{r}_a)$ and check $\boldsymbol{s}_a \overset{?}{=} \sum_{i=0}^{k-1} c^i\, \boldsymbol{s}^a_{y_i}\, \boldsymbol{e}_a$.
7: $\mathcal{P}$: Decompose each $\vec{\boldsymbol{y}}_i$ into base-σ components $(\vec{\boldsymbol{y}}_{i,j})_{j\in[\iota]}$, $\iota = \lceil \log_\sigma \beta' \rceil$.
8: $\mathcal{P}$: Compute $\mathsf{cm}_{y_{i,j}} := \mathsf{Com}(\vec{\boldsymbol{y}}_{i,j})$ and $\boldsymbol{s}_{y_{i,j}} := \langle \vec{\boldsymbol{y}}_{i,j}, \mathsf{tensor}(\vec{r}_a)\rangle$.
9: $\mathcal{P} \to \mathcal{V}$: $\{\mathsf{cm}_{y_{i,j}}, \boldsymbol{s}_{y_{i,j}}\}_{i\in[k], j\in[\iota]}$.
10: $\mathcal{V}$: For each i, verify $\sum_{j=0}^{\iota-1} \sigma^j\, \mathsf{cm}_{y_{i,j}} \overset{?}{=} \mathsf{cm}_{y_i}$, $\sum_{j=0}^{\iota-1} \sigma^j\, \boldsymbol{s}_{y_{i,j}} \overset{?}{=} \boldsymbol{s}^a_{y_i}$.
11: $\mathcal{V} \to \mathcal{P}$: $\vec{c}_a \leftarrow_{\$} \mathcal{C}'^{k\iota}$.
12: $\mathcal{P}$: Output $\vec{\boldsymbol{y}} := \sum_{i,j} c_{a_{i\iota+j}}\, \vec{\boldsymbol{y}}_{i,j}$.
13: $\mathcal{V}$: Output $\mathsf{cm}_y := \sum_{i,j} c_{a_{i\iota+j}} \mathsf{cm}_{y_{i,j}}$ and $\boldsymbol{s}_y := \sum_{i,j} c_{a_{i\iota+j}}\, \boldsymbol{s}^a_{y_{i,j}}$.

4.2.2. Compression via LaBRADOR

Note that the check on the response $\vec{\boldsymbol{z}}$ in Protocol 6 matches the dot-product form of the LaBRADOR relation (Section 2.3). Hence $\vec{\boldsymbol{z}}$ can be viewed as the witness of a LaBRADOR instance, allowing logarithmic proof size via LaBRADOR-style compression with only quasilinear prover and verifier overhead.

Protocol 6. Π_Σ: Compressed Σ-protocol for proving $(\mathcal{R}^{\beta}_{\mathsf{poly},2})^k$

$\mathcal{P}(\mathsf{cm}_{f_i}, \vec{\boldsymbol{v}}_i, \vec{\boldsymbol{f}}_i)_{i\in[k]}$, $\mathcal{V}(\mathsf{cm}_{f_i}, \vec{\boldsymbol{v}}_i)_{i\in[k]}$.

1: $\mathcal{P}$ and $\mathcal{V}$: Execute Π_{amor} to reduce $(\mathcal{R}^{\beta}_{\mathsf{poly},2})^k$ to $\mathcal{R}^{\mathcal{B}'}_{\mathsf{lin},\mathsf{f}}$.
2: $\mathcal{P}$: Sample $\vec{\boldsymbol{t}} \leftarrow_{\$} \mathcal{R}_q^m$ with $\|\vec{\boldsymbol{t}}\| \le \beta_t$.
3: $\mathcal{P} \to \mathcal{V}$: $\mathsf{cm}_t := \mathsf{Com}(\vec{\boldsymbol{t}})$, $\boldsymbol{s}_t = \langle \vec{\boldsymbol{t}}, \mathsf{tensor}(\vec{r}_0)\rangle$.
4: $\mathcal{V} \to \mathcal{P}$: $c_s \leftarrow_{\$} \mathcal{C}'$.
5: $\mathcal{P}$ computes $\vec{\boldsymbol{z}} := \vec{\boldsymbol{t}} + c_s \cdot \vec{\boldsymbol{y}}$ and runs $\mathsf{Rej}(\vec{\boldsymbol{z}}, c_s\vec{\boldsymbol{y}}, \delta)$ to output $\vec{\boldsymbol{z}}$.
6: $\mathcal{P} \to \mathcal{V}$: $\vec{\boldsymbol{z}}$.
7: $\mathcal{V}$: Check:

$$\mathsf{Com}(\vec{\boldsymbol{z}}) \overset{?}{=} c_s \cdot \mathsf{cm}_y + \mathsf{cm}_t, \quad \langle \vec{\boldsymbol{z}}, \mathsf{tensor}(\vec{r}_0)\rangle \overset{?}{=} c_s \cdot \boldsymbol{s}_y + \boldsymbol{s}_t, \quad \|\vec{\boldsymbol{z}}\| \le \gamma.$$

5. Instantiation and Implementation

5.1. Instantiation

Many lattice-based cryptographic applications involve polynomial relations subject to strict ℓ_2-norm bounds. We demonstrate how to instantiate our framework in two typical scenarios.

5.1.1. Verifiable Ciphertext Validity in FHE

In RLWE-based fully homomorphic encryption (FHE), controlling ciphertext noise is crucial for correct decryption and subsequent homomorphic operations. Consider a ciphertext $\mathsf{c} = (c_0, c_1)$ and a public key pk. We

consider the validity relation:

$$c_1 = \mathsf{pk} \cdot c_0 + m + e \pmod q, \qquad \|e\|_2 \le B,$$

where m is the publicly encoded message and e is a short error polynomial. By setting $\beta = B$, $\boldsymbol{f} = e$, and

$$h(\boldsymbol{f}) = c_1 - \mathsf{pk} \cdot c_0 - m - \boldsymbol{f},$$

this relation can be expressed within $\mathcal{R}^{\beta}_{\mathrm{poly},2}$. Our protocol then proves the existence of a bounded error satisfying $h(e) = 0$ without revealing the error itself. This provides a succinct proof of the specified ciphertext validity relation with exact ℓ_2-norm knowledge soundness.

5.1.2. Verifiable Secret-Sharing Consistency

In threshold and multi-party computation (MPC) applications, participants may need to prove that their shares are consistent with a committed polynomial while keeping the polynomial's coefficients private. Consider a norm-bounded polynomial $f(X) = \sum_{j=0}^{t} a_j X^j$ defined over $\mathcal{R}_q$, along with its shares $s_i = f(i)$. The consistency relation is given by

$$s_i = \sum_{j=0}^{t} a_j i^j, \qquad \|(a_0, \ldots, a_t)\|_2 \le B.$$

By treating the coefficient vector $\vec{f} = (a_0, \ldots, a_t)$ as a witness and defining h as the polynomial evaluation map, our framework enables proofs of share consistency and precise norm bounds without revealing the underlying coefficients.

These examples demonstrate how to handle polynomial correctness relations and precise norm constraints within a unified lattice-based proof framework.

5.2. Implementation

For a 128-bit security level ($\lambda = 128$), we select parameters satisfying the following conditions: (i) invertibility of the required ring elements; (ii) absence of modular wrap-around, guaranteed by $q > 4D^2md$ and $D > \sqrt{\beta^2 - B}$; (iii) ϵ_Σ is negligible; (iv) perfect completeness, requiring $\beta' \ge 2Td\sqrt{m}+2T\beta$, $\mathcal{B}' \ge \frac{\sigma}{2}\sqrt{md}\,Tk\iota$, and $\gamma \ge \beta_t + T\mathcal{B}'$; and (v) special knowledge soundness based on $\mathsf{MSIS}_{\kappa, B_{\mathsf{MSIS}}}$, where

$$B_{\mathsf{MSIS}} = \max\{2\gamma,\ 4T\gamma,\ 2\mathcal{B}'\beta',\ 4T\mathcal{B}'\beta'\}.$$

As described in Appendix A.2, the estimated MSIS hardness threshold is $B_{\mathsf{hard}} = \min\left\{q,\ 2^{2\sqrt{d\kappa \log q \log(1.0045)}}\right\}$. For $(q, d, \kappa) = (\approx 2^{128}, 64, 18)$, this threshold is approximately 2^{43}. We choose a strong challenge space $\mathcal{C}' \subseteq \mathcal{R}_q$, whose elements contain 23 zero coefficients, 31 coefficients taking values in $\{\pm 1\}$, and 10 coefficients taking values in $\{\pm 2\}$, yielding $|\mathcal{C}'| \approx 2^{128}$. The sum-check challenge values are drawn from $\mathcal{C} = \mathbb{Z}_q$. Under the aforementioned parameters, the overall soundness error is $O(2^{-\lambda})$.

Rejection sampling is employed to ensure an operator norm bound of $T \le 15$.

The communication volume for each sub-protocol is logarithmic, and the computational complexity for the prover/verifier is quasilinear.

Table 2 lists a set of example parameter instantiations along with the corresponding theoretical estimates for proof size. For $m = 2^{10}$, $D = 1$(i.e, binary case), and $k = 2$, the above constraints reduce to $\sigma\iota\beta' < 2^{26}$. Since $\sigma^\iota \approx \beta'$, setting $\iota = 2$ yields $2(\beta')^{3/2} < 2^{26}$, which allows $\beta' < 2^{50/3} \approx 2^{16.67}$.

Table 2. Concrete parameter choices and the resulting proof size for $m = 2^{10}$.

$\log q$	d	κ	T	$\deg(h)$	ι	D	k	Proof Size
128	64	18	15	6	2	1	2	2.90 MB

Author Contributions

L.Z.: writing—original draft; S.G.: conceptualization, writing—review and editing; B.X.: writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding

This work was partially supported by the Hong Kong RGC General Research Fund (GRF) under project numbers 15207522, 15202123, and 15205624; the RGC Collaborative Research Fund (CRF) under project number C5097-25G; the NSFC Young Scientists Fund (No. 62302418); and the Hong Kong Polytechnic University Start-up Fund (Project ID P0058467).

Data Availability Statement

Not applicable.

Conflicts of Interest

The authors declare no conflict of interest.

Use of AI and AI-Assisted Technologies

During the preparation of this work, the authors used ChatGPT for limited phrasing refinement and grammar correction. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

Appendix A. Additional Preliminaries

Appendix A.1. Sum-Checks and Multilinear Extensions over Rings

Sum-check over rings. We give a generalized sum-check protocol operating on an arbitrary ring $\bar{\mathcal{R}}$, utilizing challenges sampled from the sampling set $\mathcal{C}$.

Lemma A1 (**Generalized sum-check** [20]). *For an arbitrary ring $\bar{\mathcal{R}}$, let $f \in \bar{\mathcal{R}}^{\leq d}[\boldsymbol{X}_0, \cdots, \boldsymbol{X}_{\mu-1}]$ be a μ-variate nonzero polynomial with per-variable degree at most d. Let $\mathcal{C} \subset \bar{\mathcal{R}}$ be a strong sampling set, the following protocol for proving $\boldsymbol{s} = \sum_{\vec{b}\in\{0,1\}^\mu} f(\vec{b})$ has soundness error $\frac{\mu d}{|\mathcal{C}|}$.*

(1) In the i-th round ($i \in [\mu]$):

- *Upon receiving challenges $\boldsymbol{r}_0, \cdots, \boldsymbol{r}_{i-1}$ from the previous rounds, $\mathcal{P}$ sends the univariate polynomial $f_i(\boldsymbol{X}) := \sum_{\vec{b}\in\{0,1\}^{\mu-i}} f(\boldsymbol{r}_0, \cdots, \boldsymbol{r}_{i-1}, \boldsymbol{X}, \vec{b})$. More precisely, $\mathcal{P}$ sends $d+1$ evaluations of f_i at $d+1$ points in $\mathcal{C}$.*
- *$\mathcal{V}$ checks $f_i(0) + f_i(1) = f_{i-1}(\boldsymbol{r}_{i-1})$ and sends a challenge $\boldsymbol{r}_i \leftarrow_{\$} \mathcal{C}$.*

(2) $\mathcal{V}$ checks $f_{\mu-1}(\boldsymbol{r}_{\mu-1}) = f(\boldsymbol{r}_0, \cdots, \boldsymbol{r}_{\mu-1})$.

Sometimes the last check is deferred. Accordingly, the protocol outputs a claim of $f_{\mu-1}(\boldsymbol{r}_{\mu-1}) = f(\boldsymbol{r}_0, \cdots, \boldsymbol{r}_{\mu-1})$.

Multilinear extensions over rings. The formal definition of multilinear extensions (MLEs) over rings [20] is defined below.

Definition A1 (**MLEs over Rings**). *For an arbitrary ring $\bar{\mathcal{R}}$, given a function $f : \{0,1\}^\mu \to \bar{\mathcal{R}}$, the multilinear extension (MLE) of f is defined as $\mathsf{mle}[f](\vec{\boldsymbol{X}}) := \sum_{\vec{b}\in\{0,1\}^\mu} f(\vec{b}) \cdot \mathsf{eq}(\vec{b}, \vec{\boldsymbol{X}}) \in \bar{\mathcal{R}}^{\leq 1}[\boldsymbol{X}_0, \cdots, \boldsymbol{X}_{\mu-1}]$, where $\mathsf{eq}(\vec{b}, \vec{\boldsymbol{X}}) := \prod_{i=0}^{\mu-1} \big((1-b_i)(1-\boldsymbol{X}_i) + b_i \boldsymbol{X}_i\big)$.*

When given an m-size vector $\vec{\boldsymbol{f}} \in \bar{\mathcal{R}}^m$, denote the MLE of $\vec{\boldsymbol{f}}$ as $\mathsf{mle}[\vec{\boldsymbol{f}}](\vec{\boldsymbol{X}}) := \sum_{i\in[m]} \boldsymbol{f}_i \cdot \mathsf{eq}(\text{to-bit}(i), \vec{\boldsymbol{X}})$, where the "to-bit" operation for an integer i denotes decomposing it into its binary representation in base 2.

Similar to the field setting, the multilinear extension over rings of a Boolean function is unique [20] and has the following properties.

Corollary A1 (**Corollary 2.1 in** [15]). *For $\boldsymbol{f} \in \bar{\mathcal{R}}^{\leq 1}[\boldsymbol{X}_0, \cdots, \boldsymbol{X}_{\mu-1}]$ over a ring $\bar{\mathcal{R}}$, we have $\boldsymbol{f}(\vec{\boldsymbol{X}}) = \sum_{\vec{b}\in\{0,1\}^\mu} \boldsymbol{f}(\vec{b}) \cdot \mathsf{eq}(\vec{b}, \vec{\boldsymbol{X}})$.*

Appendix A.2. Lattice Problems and Ajtai Commitment

MSIS **.** We define the lattice problem known as the *module short integer solution* (MSIS) [23], upon which the security of our schemes relies.

Definition A2 (MSIS). $\mathsf{MSIS}(\kappa, m, q, \gamma)$. *Given a random matrix* $\boldsymbol{A} \leftarrow_{\$} \mathcal{R}_q^{\kappa \times m}$, *the goal of the problem is to find non-zero* $\vec{\boldsymbol{z}} \in \mathcal{R}_q^m$ *such that* $\boldsymbol{A}\vec{\boldsymbol{z}} = \vec{0}$ *over* $\mathcal{R}_q$ *and* $\|\vec{\boldsymbol{z}}\| \leqslant \gamma$.

Following the lattice reduction estimates in [24], the MSIS problem is considered hard when the solution norm bound satisfies

$$\beta_{\mathsf{MSIS}} < \min\left(q,\ 2^{2\sqrt{d\kappa \log q \log(1.0045)}}\right),$$

where d denotes the degree of the polynomial defining the ring $\mathcal{R}_q$.

Ajtai commitment scheme. We review the Ajtai commitment scheme [25] where the messages are ring elements with *small* norms.

Definition A3 (Ajtai commitment scheme). *Let* λ *be the security parameter and* κ, q, γ *be positive integers. Suppose a prover commits to a* m*-dimensional* short *vector* $\vec{\boldsymbol{f}} \in \mathcal{R}_q^m$ *(with small norms). The Ajtai commitment is as follows:*

- $\mathsf{Setup}(1^\lambda)$*: Sample a matrix* $\boldsymbol{A} \leftarrow_{\$} \mathcal{R}_q^{\kappa \times m}$. *Output* $\mathsf{ck} := \boldsymbol{A}$.
- $\mathsf{Commit}_{\mathsf{ck}}(\vec{\boldsymbol{f}})$*: Given* $\vec{\boldsymbol{f}}$ *such that* $\|\vec{\boldsymbol{f}}\| \leq \gamma$, *output* $\mathsf{Com}(\vec{\boldsymbol{f}}) := \boldsymbol{A}\vec{\boldsymbol{f}}$.

Definition A4. ***Hiding***. *The commitment scheme is hiding if for all PPT adversaries* $\mathcal{A}$,

$$\Pr\left[\begin{array}{l} \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda), (\vec{\boldsymbol{m}}_0, \vec{\boldsymbol{m}}_1) \leftarrow \mathcal{A}(\mathsf{ck}), \\ b \leftarrow_{\$} \{0,1\}, \vec{r} \leftarrow_{\$} D_{\mathsf{ck}}, C \leftarrow \mathsf{Commit}_{\mathsf{ck}}(\vec{\boldsymbol{m}}_b; \vec{r}) \end{array} : \mathcal{A}(C) = b\right] \approx \frac{1}{2}.$$

where D_{ck} *is the space of the randomness controller by* ck.

Definition A5. ***Binding***. *The commitment scheme is binding if for all PPT adversaries* $\mathcal{A}$,

$$\Pr\left[\begin{array}{l} \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda), \\ (\vec{\boldsymbol{m}}_0, \vec{r}_0, \vec{\boldsymbol{m}}_1, \vec{r}_1) \leftarrow \mathcal{A}(ck) \end{array} : \begin{array}{l} \vec{\boldsymbol{m}}_0 \neq \vec{\boldsymbol{m}}_1 \quad \wedge \\ \mathsf{Commit}_{\mathsf{ck}}(\vec{\boldsymbol{m}}_0; \vec{r}_0) = \mathsf{Commit}_{\mathsf{ck}}(\vec{\boldsymbol{m}}_b; \vec{r}_b) \end{array}\right] \approx 0.$$

A commitment scheme is *binding* if it is hard to find two openings for the same commitment, and *hiding* if it reveals no information about the message. The Ajtai commitment is binding under the hardness of $\mathsf{MSIS}(\kappa, m, q, 2\gamma)$, and supports hiding by appending a small random vector to the message. For simplicity, we treat this randomness as part of the witness, following [17,18].

Appendix A.3. Reject Sampling

Definition A6 (Discrete Gaussian Distribution). *The discrete Gaussian distribution over* $\mathcal{R}^\ell$, *centered at a vector* $\vec{\boldsymbol{v}} \in \mathcal{R}^\ell$ *with standard deviation* $\delta > 0$, *is defined as*

$$D_{\vec{\boldsymbol{v}},\delta}^\ell(\vec{\boldsymbol{z}}) = \frac{e^{-\|\vec{\boldsymbol{z}} - \vec{\boldsymbol{v}}\|^2/(2\delta^2)}}{\sum_{\vec{\boldsymbol{z}}' \in \mathcal{R}^\ell} e^{-\|\vec{\boldsymbol{z}}' - \vec{\boldsymbol{v}}\|^2/(2\delta^2)}}.$$

When it is centered at the origin $\vec{\mathbf{0}} \in \mathcal{R}_q^\ell$, *we simply write* $D_\delta^\ell = D_{\vec{\mathbf{0}},\delta}^\ell$.

In lattice-based zero-knowledge proofs, when the prover aims to ensure that the output distribution is independent of the secret and the randomness, the *rejection sampling* algorithm should be applied.

Lemma A2 (Reject Sampling [26]). *Let* $V \subseteq \mathcal{R}^\ell$ *be a set of polynomials with norm at most* T *and* $\rho : V \to [0,1]$ *be a probability distribution. Fix the standard deviation* $\delta = kT$ *for a constant* k. *Then, the following statement hold. Let* $M = \exp(14/k + 1/(2k^2))$. *Now, sample* $\vec{\boldsymbol{v}} \leftarrow \rho$ *and* $\vec{\boldsymbol{y}} \leftarrow D_\delta^\ell$, *set* $\vec{\boldsymbol{z}} = \vec{\boldsymbol{y}} + \vec{\boldsymbol{v}}$, *and run* $b \leftarrow \mathsf{Rej}_\delta(\vec{\boldsymbol{z}}, \vec{\boldsymbol{v}}, \delta)$ *as defined in Algorithm A1. Then, the probability that* $b = 0$ *is at least* $(1 - 2^{-128})/M$ *and the distribution of* $(\vec{\boldsymbol{v}}, \vec{\boldsymbol{z}})$, *conditioned on* $b = 0$, *is within statistical distance of* 2^{-128} *of the product distribution* $\rho \times D_\delta^\ell$.

Appendix A.4. Properties of Σ-Protocols

For any Nondeterministic Polynomial time (NP) relation $\mathcal{R}_*$, let $\mathcal{G}$ be a setup algorithm that generates public parameters pp for the Σ-protocol. Given an instance $(u; w)$ where u is the public statement and w is the witness, the Σ-protocol to prove $(u; w) \in \mathcal{R}_*$ works as follows: (i) $\mathcal{P}$ sends an initial message t, (ii) $\mathcal{V}$ issues a random

challenge $\lambda \leftarrow_{\$} \mathcal{C}$ from the challenge space, and (iii) $\mathcal{P}$ provides a response g. The properties of the Σ-protocol are defined as follows.

Algorithm A1. Rejection Sampling [26]

$\mathsf{Rej}(\vec{z}, \vec{v}, \delta)$

1: $u \leftarrow_{\$} [0, 1)$
2: **if** $u > \frac{1}{M} \cdot \exp(\frac{-2\langle \vec{z}, \vec{v} \rangle + \|\vec{v}\|^2}{2\delta^2})$ **then**
3: **return** $\perp$
4: **end if**
5: Return 1

Completeness. When $(u; w) \in \mathcal{R}_*$, the interaction between an honest prover and an honest verifier yields an accepting transcript with probability larger than $1 - \epsilon_c$, where ϵ_c is denoted as the completeness error.

Formally, for all PPT adversaries $\mathcal{A}$,

$$\Pr\left[\begin{array}{l} \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), (u, w) \leftarrow \mathcal{A}(\mathsf{pp}), \\ t \leftarrow \mathcal{P}(\mathsf{pp}, u, w), \lambda \leftarrow_{\$} \mathcal{C}, g \leftarrow \mathcal{P}(\lambda) \end{array} : \mathcal{V}(\mathsf{pp}, u, t, \lambda, g) = 1 \right] = 1 - \epsilon_c.$$

Special soundness. There exists an efficient extractor $\mathcal{E}$ that on any statement u and k accepting transcripts $(t, \lambda_i, g_i)_{i=0}^{k-1}$ with common t and distinct λ_i's and g_i's, outputs a witness w' such that $(u, w') \in \mathcal{R}_*$ with probability larger than $1 - \epsilon_s$, where ϵ_s is denoted as the soundness error. This is also known as the k-special soundness prosperity. A more generalized version, $(k_1, \cdots, k_\mu)$-special soundness, allows $\mathcal{E}$ to run with a $(k_1, \cdots, k_\mu)$-tree of accepting transcripts tr, where tr is a set of $\prod_{i=1}^{\mu} k_i$ accepting transcripts with the following structure. The nodes in tr correspond to the prover's messages and the edges correspond to the verifier's challenges. Every node at depth i has precisely k_i children corresponding to k_i pairwise distinct challenges. Every transcript corresponds to exactly one path from the root node to a leaf node.

Formally, for all PPT adversaries $\mathcal{A}$,

$$\Pr\left[\ \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), (u, \mathsf{tr}) \leftarrow \mathcal{A}(\mathsf{pp}), w \leftarrow \mathcal{E}(\mathsf{pp}, u, \mathsf{tr}) : (u; w) \in \mathcal{R} \ \right] = 1 - \epsilon_s.$$

Special HVZK. There exists an efficient simulator that on any statement u and a challenge λ from the challenge space, outputs (t, g) such that (t, g) is indistinguishable from an accepting transcript produced by running the protocol.

Formally, for all PPT adversaries $\mathcal{A}$,

$$\begin{aligned} &\Pr\left[\ \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), (u, w, \lambda) \leftarrow \mathcal{A}(\mathsf{pp}), t \leftarrow \mathcal{P}(\mathsf{pp}, u, w), g \leftarrow \mathcal{P}(\lambda) : \mathcal{A}(t, \lambda, g) = 1 \ \right] \\ &\approx \Pr\left[\ \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), (u, w, \lambda) \leftarrow \mathcal{A}(\mathsf{pp}), (t, \lambda, g) \leftarrow \mathcal{S}(\mathsf{pp}, u) : \mathcal{A}(t, \lambda, g) = 1 \ \right]. \end{aligned}$$

Appendix A.5. Coordinate-wise Special Soundness

Coordinate-wise special soundness extends the notion of special soundness and ensures knowledge soundness for a $(2\mu + 1)$-round interactive proof system [27].

Let $x = (x_1, x_2, \ldots, x_\ell)$, $y = (y_1, y_2, \ldots, y_\ell) \in S^\ell$. For $i \in [\ell]$, define $\equiv_i$ so that $\vec{x} \equiv_i \vec{y}$ means $\vec{x}$ and $\vec{y}$ agree in every coordinates except the i-th, i,e,

$$\vec{x} \equiv_i \vec{y} \Leftrightarrow (x_i \neq y_i \wedge \forall j \in [\ell] \setminus \{i\}, x_j = y_j).$$

Next, for $K = \ell(k - 1) + 1$, define the set:

$$SS(S, \ell, k) := \left\{ \{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_K\} \subseteq (S^\ell)^K \ \middle| \ \begin{array}{l} \exists e \in [K], \forall i \in [\ell], \\ \exists J_i \subseteq [K] \setminus \{e\}, |J_i| = k - 1, \\ \forall j \in J_i, \ \vec{x}_e \equiv_i \vec{x}_j \end{array} \right\}.$$

Equivalently, a "central" vector $\vec{x}_e$ exists such that, for each coordinate i, the set $\{\vec{x}_e\} \cup \{\vec{x}_j : j \in J_i\}$ contains k vectors that differ only in the i-th coordinate.

Definition A7 (ℓ-Coordinate-wise k-Special Soundness (Multi-round)). *Let $\Pi = (\mathcal{S}, \mathcal{P}, \mathcal{V})$ be a public-coin $(2\mu + 1)$-round interactive proof for a relation $\mathcal{R}$. In each round, the verifier samples a challenge $c \in \mathcal{C} = S^\ell$. Consider a full $(\ell(k - 1) + 1)$-ary tree of depth μ, where the outgoing edges at each internal node are labeled by*

$\ell(k-1)+1$ *distinct challenges whose set lies in* $\mathsf{SS}(S,\ell,k)$*. Each root-to-leaf path corresponds to one transcript, so the tree collects* $K=(\ell(k-1)+1)^{\mu}$ *transcripts for the same statement* x*. We say that* Π *satisfies* ℓ-coordinate-wise k-special soundness *if there exists a polynomial-time extractor that, given* x *and any such transcript tree for* x*, outputs a witness* w *such that* $(x,w)\in\mathcal{R}$*. This notion reduces to the standard* k*-special soundness when* $\ell=1$*.*

Lemma A3 (Lemma 2.31 in [27]). *Let* $\Pi=(\mathcal{S},\mathcal{P},\mathcal{V})$ *be public-coin a* $(2\mu+1)$*-round interactive proof system for relation* $\mathcal{R}$ *and* S^{ℓ} *be the challenge space of* $\mathcal{V}$ *in each round. If* Π *is* ℓ*-coordinate-wise* k*-special sound and* $(\ell(k-1))^{\mu}=\mathsf{poly}(\lambda)$*, it is knowledge sound with knowledge error* $\mu\ell(k-1)/|S|$*.*

Appendix A.6. Reduction of Knowledge Protocols

Definition A8 (Reduction of knowledge [28]). *Let* λ *be a security parameter. Given two relations* $\mathcal{R}_1$ *and* $\mathcal{R}_2$*, denote* $\langle\mathcal{P},\mathcal{V}\rangle$ *be an interactive protocol between a prover* $\mathcal{P}$ *and a verifier* $\mathcal{V}$*. A reduction of knowledge protocol* Π *from* $\mathcal{R}_1$ *to* $\mathcal{R}_2$ *consists of the following algorithms:*

- $\mathsf{Setup}(1^{\lambda})\to\mathsf{pp}$*: Output public parameters* pp*.*
- $\langle\mathcal{P}(\mathsf{pp},u_1,w_1),\mathcal{V}(\mathsf{pp},u_1)\rangle\to(u_2,w_2)$*: On input public parameters* pp*, shared public statement* u_1*,* $\mathcal{P}$ *(with a witness* w_1 *such that* $(u_1,w_1)\in\mathcal{R}_1$*) and* $\mathcal{V}$ *engage in* Π*. At the end of* Π*,* $\mathcal{P}$ *and* $\mathcal{V}$ *output* u_2 *(or* $\perp$ *if abort),* $\mathcal{P}$ *additionally outputs* w_2*.*

Reduction of knowledge protocols satisfy the following properties:

- Completeness. For every PPT adversary $\mathcal{A}$ that adaptively chooses an $\mathcal{R}_1$ instance $(u_1,w_1)\leftarrow\mathcal{A}(\mathsf{pp})$ when given $\mathsf{pp}\leftarrow\mathsf{Setup}(1^{\lambda})$, the protocol execution $(u_2,w_2)\leftarrow\langle\mathcal{P}(\mathsf{pp},u_1,w_1),\mathcal{V}(\mathsf{pp},u_1)\rangle$ satisfies $(u_2,w_2)\in\mathcal{R}_2$ if $(u_1,w_1)\in\mathcal{R}_1$.
- Knowledge soundness. For every expected polynomial-time adversary $\mathcal{A}$ and malicious prover $\mathcal{P}^*$, there is an expected polynomial-time extractor $\mathcal{E}$ such that given $\mathsf{pp}\leftarrow\mathsf{Setup}(1^{\lambda})$ and $(u_1,w_1^*)\leftarrow\mathcal{A}(\mathsf{pp})$,

$$\Pr\left[\left(u_1,\mathcal{E}(\mathsf{pp},u_1,w_1^*)\right)\in\mathcal{R}_1\right]\approx\Pr\left[\langle\mathcal{P}^*(\mathsf{pp},u_1,w_1^*),\mathcal{V}(\mathsf{pp},u_1)\rangle\in\mathcal{R}_2\right].$$

- **Public reducibility**. There is a deterministic polynomial-time algorithm f, such that for any PPT adversary $\mathcal{A}$ and a malicious expected polynomial-time prover $\mathcal{P}^*$, given

$$\mathsf{pp}\leftarrow\mathsf{Setup}(1^{\lambda}),\quad(u_1,w_1^*)\leftarrow\mathcal{A}(\mathsf{pp}),\quad(u_2,w_2)\leftarrow\langle\mathcal{P}^*(\mathsf{pp},u_1,w_1^*),\mathcal{V}(\mathsf{pp},u_1)\rangle,$$

and the transcript tr, we have $f(\mathsf{pp},u_1,\mathsf{tr})=u_2$.

By the composition theory [28], the reduction of knowledge protocols can be composed.

Theorem A1 (Sequential composition (Theorem 5 in [28])). *Let* $\mathcal{R}_1$*,* $\mathcal{R}_2$*,* $\mathcal{R}_3$ *be three relations. Given two reduction of knowledge protocols,* Π_1 *from* $\mathcal{R}_1$ *to* $\mathcal{R}_2$ *and* Π_2 *from* $\mathcal{R}_2$ *to* $\mathcal{R}_3$*, the composed protocol* $\Pi_1\diamond\Pi_2$ *is a reduction of knowledge from* $\mathcal{R}_1$ *to* $\mathcal{R}_3$*.*

Theorem A2 (Parallel composition (Theorem 6 in [28])). *Let* $\mathcal{R}_1$*,* $\mathcal{R}_2$*,* $\mathcal{R}_3$*,* $\mathcal{R}_4$ *be four relations. Given two reduction of knowledge protocols,* Π_1 *from* $\mathcal{R}_1$ *to* $\mathcal{R}_2$ *and* Π_2 *from* $\mathcal{R}_3$ *to* $\mathcal{R}_4$*, the composed protocol* $\Pi_1\times\Pi_2$ *is a reduction of knowledge from* $(\mathcal{R}_1\times\mathcal{R}_3)$ *to* $(\mathcal{R}_2\times\mathcal{R}_4)$*.*

Appendix B. Complete Protocol for $\Pi_{\mathsf{rg},\infty}$

The complete $\Pi_{\mathsf{rg},\infty}$ which reduces $\mathcal{R}^{D}_{\mathsf{lin},\infty}$ to the linear relation $\mathcal{R}_{\mathsf{lin}}$ is provided in Protocol A1.

Theorem A3. $\Pi_{\mathsf{rg},\infty}$ *is a reduction of knowledge from* $\mathcal{R}^{D}_{\mathsf{lin},\infty}$ *to* $\mathcal{R}_{\mathsf{lin}}$*.*

Proof. The public reducibility is straightforward. Perfect completeness follows from the completeness of the underlying sum-check protocol, Lemma 4, and Lemma 3. Specifically, an honest prover with a correctly constructed monomial and a bounded witness can pass both the sum-check in Step 4 and the range check in Step 8. The knowledge soundness follows the theorem below. □

Theorem A4. *When the commitment scheme* $\mathsf{Com}(\cdot)$ *is binding, by setting* $D=d'=\frac{d}{2}$*,* $\Pi_{\mathsf{rg},\infty}$ *is knowledge sound with soundness error* $\epsilon_{\mathsf{rg},\infty}=\frac{3d+5\log m}{|\mathcal{C}|}$*.*

Protocol A1. $\Pi_{\mathsf{rg},\infty}$: Reduction from $\mathcal{R}^{D}_{\mathsf{lin},\infty}$ to $\mathcal{R}_{\mathsf{lin}}$

$\mathbb{x} = (\mathsf{cm}_f, \mathsf{cm}_a, \vec{r}_1, \vec{r}_2, \boldsymbol{s}_1, \boldsymbol{s}_{2,f}, \boldsymbol{s}_{2,a}), \quad \mathbb{w} = (\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}})$.
$\mathcal{P}(\mathbb{x}, \mathbb{w}), \quad \mathcal{V}(\mathbb{x})$.

1: $\mathcal{P}$: Construct $\boldsymbol{M}_f := \mathrm{EXP}(\vec{f})$ and $\boldsymbol{M}_a := \mathrm{EXP}(\vec{a}) \in \mathcal{M}^{m\times d}$.
2: $\mathcal{P} \to \mathcal{V}$: $\mathsf{cm}_{\boldsymbol{M}_f} := \mathsf{Com}(\boldsymbol{M}_f)$, $\mathsf{cm}_{\boldsymbol{M}_a} := \mathsf{Com}(\boldsymbol{M}_a) \in \mathcal{R}_q^{\kappa\times d}$.
3: $\mathcal{V} \to \mathcal{P}$: $\vec{\alpha}_1, \vec{\alpha}_2 \leftarrow_{\$} \mathcal{C}^{\log m}$ and $v, c \leftarrow_{\$} \mathcal{C}$.
4: $\mathcal{P}$ and $\mathcal{V}$: Construct $g_3(\vec{\boldsymbol{X}})$ as in Equation (9), and apply the sum-check protocol on $\sum_{\vec{b}\in\{0,1\}^{\log m}} g_3(\vec{b}) = 0$, which reduces to $g_3(\vec{r}_3) = \boldsymbol{s}_{g_3}$ for $\vec{r}_3 \leftarrow_{\$} \mathcal{C}^{\log m}$.
5: $\mathcal{P} \to \mathcal{V}$: $\{\boldsymbol{s}_{M_{f,j}} := \langle \vec{\boldsymbol{M}}_{f,j}, \mathsf{tensor}(\vec{r}_3)\rangle, \boldsymbol{s}_{M_{a,j}} := \langle \vec{\boldsymbol{M}}_{a,j}, \mathsf{tensor}(\vec{r}_3)\rangle\}_{j\in[d]}$.
6: $\mathcal{V}$: Compute $\boldsymbol{e}_{\alpha_1} := \mathsf{eq}(\vec{\alpha}_1, \vec{r}_3)$ and $\boldsymbol{e}_{\alpha_2} := \mathsf{eq}(\vec{\alpha}_2, \vec{r}_3)$, and check

$$\begin{aligned} \boldsymbol{s}_{g_3} \stackrel{?}{=} \boldsymbol{e}_{\alpha_1} \cdot \sum_{j\in[d]} c^j \left(\mathsf{ev}_{\boldsymbol{s}_{M_{f,j}}}(v)^2 - \mathsf{ev}_{\boldsymbol{s}_{M_{f,j}}}(v^2)\right) \\ + \boldsymbol{e}_{\alpha_2} \cdot \sum_{j\in[d]} c^j \left(\mathsf{ev}_{\boldsymbol{s}_{M_{a,j}}}(v)^2 - \mathsf{ev}_{\boldsymbol{s}_{M_{a,j}}}(v^2)\right). \end{aligned} \tag{A1}$$

7: $\mathcal{P} \to \mathcal{V}$: $\vec{\boldsymbol{s}}_{3,f} := F^{\top}\mathsf{tensor}(\vec{r}_3) \in \mathcal{R}_q^d$, $\vec{\boldsymbol{s}}_{3,a} := A^{\top}\mathsf{tensor}(\vec{r}_3) \in \mathcal{R}_q^d$, where F and A denote the coefficient matrices of $\vec{\boldsymbol{f}}$ and $\vec{\boldsymbol{a}}$.
8: $\mathcal{V}$: Let $\psi := \sum_{i\in[1,d')} i\cdot(X^{-i} + X^i) \in \mathcal{R}_q$, and check $\mathsf{ct}(\psi\cdot(\boldsymbol{s}_{M_{f,j}})_{j\in[d]}) \stackrel{?}{=} \vec{\boldsymbol{s}}_{3,f}$, $\mathsf{ct}(\psi\cdot(\boldsymbol{s}_{M_{a,j}})_{j\in[d]}) \stackrel{?}{=} \vec{\boldsymbol{s}}_{3,a}$.
9: $\mathcal{P}$: Output $\mathbb{w} = (\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}}, \vec{r}_i)_{i\in\{1,2,3\}} \in \mathcal{R}_q^{\log m},)$.
10: $\mathcal{V}$: Set $\boldsymbol{s}_{3,f} := \sum_{i=0}^{d-1} \boldsymbol{s}_{3,f_i} X^i$, $\boldsymbol{s}_{3,a} := \sum_{i=0}^{d-1} \boldsymbol{s}_{3,a_i} X^i$, and output

$$\mathbb{x} = \begin{pmatrix} (\mathsf{cm}_f, \mathsf{cm}_a, \mathsf{cm}_{\boldsymbol{M}_f}, \mathsf{cm}_{\boldsymbol{M}_a}), (\vec{r}_i)_{i\in\{1,2,3\}} \\ (\boldsymbol{s}_1, (\boldsymbol{s}_{2,f}, \boldsymbol{s}_{3,f}), (\boldsymbol{s}_{2,a}, \boldsymbol{s}_{3,a}), (\vec{\boldsymbol{s}}_{M_f}, \vec{\boldsymbol{s}}_{M_a})) \end{pmatrix}.$$

Proof. The extractor interacts with the malicious prover $\mathcal{P}^*$ and outputs $\mathcal{P}^*$'s result if the verifier accepts within polynomial time. Let $(\mathbb{x}_{\mathbb{i}}, \mathbb{x}_{\mathbb{o}})$ denote the statements for the input and output relations, and let $\mathbb{w}_{\mathbb{o}} = (\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}}, \boldsymbol{M}_f, \boldsymbol{M}_a)$ be the extractor's output. Consider the case where $(\mathbb{x}_{\mathbb{o}}, \mathbb{w}_{\mathbb{o}}) \in \mathcal{R}_{\mathsf{lin}}$ while $(\mathbb{x}_{\mathbb{i}}, \mathbb{w}_{\mathbb{i}} = (\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}})) \notin \mathcal{R}^{D}_{\mathsf{lin},\infty}$.

(1) Case 1.The extracted $(\boldsymbol{M}_f, \boldsymbol{M}_a)$ satisfy the relation constraints but are not in $\mathcal{M}$, i,e, some entries fall outside the monomial set. In this case, the following three events may occur:
- **Event 1:** Though $(\boldsymbol{M}_f, \boldsymbol{M}_a) \notin \mathcal{M}$, it holds $\mathsf{ev}_{\boldsymbol{M}_{f_{i,j}}}(v)^2 = \mathsf{ev}_{\boldsymbol{M}_{f_{i,j}}}(v^2)$ and $\mathsf{ev}_{\boldsymbol{M}_{a_{i,j}}}(v)^2 = \mathsf{ev}_{\boldsymbol{M}_{a_{i,j}}}(v^2)$. According to Lemma 3, this happens with probability $\frac{2d}{|\mathcal{C}|}$.
- **Event 2:** $\mathsf{ev}_{\boldsymbol{M}_{f_{i,j}}}(v)^2 \neq \mathsf{ev}_{\boldsymbol{M}_{f_{i,j}}}(v^2)$ and $\mathsf{ev}_{\boldsymbol{M}_{a_{i,j}}}(v)^2 \neq \mathsf{ev}_{\boldsymbol{M}_{a_{i,j}}}(v^2)$, but the batched sum-check in Step 3 still holds, which occurs with probability $\frac{d+\log m}{|\mathcal{C}|}$.
- **Event 3:** The batched sum-check fails, but the evaluation check in Step 6 passes, which occurs with probability $\frac{3\log m}{|\mathcal{C}|}$.

(2) **Case 2.** The extracted $(\boldsymbol{M}_f, \boldsymbol{M}_a)$ form valid monomial sets, but

$$\|\vec{\boldsymbol{f}}\|_\infty \geq D \ \vee \ \|\vec{\boldsymbol{a}}\|_\infty \geq D \ \vee \ \boldsymbol{M}_f \notin \mathrm{EXP}(F) \ \vee \ \boldsymbol{M}_a \notin \mathrm{EXP}(A).$$

By Lemma 4, under such conditions we have $\mathsf{ct}(\psi\cdot\boldsymbol{M}_f) \neq F$ or $\mathsf{ct}(\psi\cdot\boldsymbol{M}_a) \neq A$, while the probability of passing the check in Step 8 is $\frac{\log m}{|\mathcal{C}|}$.

Therefore, the total knowledge soundness error is $\frac{3d+5\log m}{|\mathcal{C}|}$. □

Appendix C. Proof for Theorem 9

Theorem A5. *When* $\beta' \geq 2Td\cdot\sqrt{m} + T\cdot\beta$ *and* $\mathsf{Com}(\cdot)$ *is binding for norm* $\max(2\beta', 4T\beta')$*, the protocol* Π_{tool} *is* $(2d+2)$*-coordinate-wise* 2*-special sound with soundness error* $\epsilon_{\mathsf{tool,cwss}} = \epsilon_{\mathsf{ba}} + \epsilon_{\mathsf{rg}} + \frac{2\log m + 2d + 3}{|\mathcal{C}|}$.

Proof. Denote the witness matrix in the output of Step 3 in Π_{tool} as $\boldsymbol{F} = [\vec{\boldsymbol{f}}, \vec{\boldsymbol{a}}, \boldsymbol{M}_f, \boldsymbol{M}_a]$. The proof proceeds as follows.

(1) Extract the weak commitment opening. Given a set of accepted transcripts of $\{\mathsf{tr}_{\mathsf{IPA}}, \mathsf{tr}_{\mathsf{rg},\infty}, \mathsf{tr}_{\mathsf{ba}}, \mathsf{tr}_{\mathsf{tool}_i}\}$ for

$i \in [2d+3]$, where

$$\mathsf{tr}_{\mathsf{tool}_i} = \{\vec{b}_i, c, (\mathsf{tr}_{\mathsf{sc}}), \vec{c}_{f_i}\},$$

and $\mathsf{tr}_{\mathsf{sc}}$ denotes the same sum-check transcripts containing $(\vec{r}_0, \vec{s}_0)$ in the i-th $\mathsf{tr}_{\mathsf{tool}_i}$, with $\vec{c}_{t,i} = (\vec{b}_i, \vec{c}_{f_i}) \in \mathsf{SS}(\mathcal{C}', (2d+2), 2)$ as defined in Appendix A.5, and the corresponding output instances $(\mathbb{x} = (\mathsf{cm}_{y,i}, \vec{r}_0, \boldsymbol{s}_0), \mathbb{w}_i = \vec{\boldsymbol{y}}_i) \in \mathcal{R}^{\beta'}_{\mathsf{lin,f}}$.

Without loss of generality, denote $\vec{c}_{t,0}$ as the central vector, and assume each other challenge vector $\vec{c}_{t,i}$ differs from $\vec{c}_{t,0}$ only in the $(i-1)$-th position. The extractor can extract the $(i-1)$-th relaxed witness by solving the equations:

$$\begin{aligned} \mathsf{cm}_{y,0} &= \mathsf{Com}(\boldsymbol{F}) \cdot \vec{c}_{t,0}, \quad \vec{\boldsymbol{y}}_0 = \boldsymbol{F} \cdot \vec{c}_{t,0}, \\ \mathsf{cm}_{y,i} &= \mathsf{Com}(\boldsymbol{F}) \cdot \vec{c}_{t,i}, \quad \vec{\boldsymbol{y}}_i = \boldsymbol{F} \cdot \vec{c}_{t,i}. \end{aligned} \tag{A2}$$

This yields $\tilde{\vec{\boldsymbol{F}}}_{i-1} = \frac{\vec{\boldsymbol{y}}_i - \vec{\boldsymbol{y}}_0}{c_{t,i,i-1} - c_{t,0,i-1}}$ such that $\mathsf{Com}(\tilde{\vec{\boldsymbol{F}}}_{i-1}) = \frac{\mathsf{cm}_{y_i} - \mathsf{cm}_{y_0}}{c_{t,i,i-1} - c_{t,0,i-1}} = \mathsf{Com}(\vec{\boldsymbol{F}}_{i-1})$, and $\|\bar{c} \cdot \tilde{\vec{\boldsymbol{F}}}_{i-1}\|_2 \leq 2\beta'$, where $\bar{c} = c_{t,i,i-1} - c_{t,0,i-1}$.

(2) The relaxed opening satisfies the constraints in $\mathcal{R}_{\mathsf{lin,ba}}$. We now analyze the probability that the extracted $\tilde{\boldsymbol{F}}$ fails to satisfy the constraints in $\mathcal{R}_{\mathsf{lin,ba}}$. This failure scenario can be decomposed into two mutually exclusive events:

- **Event 1:** The extracted $\tilde{\boldsymbol{F}}$ violates the constraints in $\mathcal{R}_{\mathsf{lin,ba}}$, yet the sum-check claim in Step 7 holds. This event occurs with probability $\frac{2d+3}{|\mathcal{C}|}$.
- **Event 2:** The extracted $\tilde{\boldsymbol{F}}$ violates the constraints in $\mathcal{R}_{\mathsf{lin,ba}}$, the sum-check claim in Step 7 is rejected, but the subsequent evaluation check passes. This event occurs with probability at most $\frac{2\log m}{|\mathcal{C}|}$.

Therefore, the overall probability that the extracted $\tilde{\boldsymbol{F}}$ does not satisfy $\mathcal{R}_{\mathsf{lin,ba}}$ is bounded by $\frac{2\log m + 2d + 3}{|\mathcal{C}|}$.

(3) The relaxed opening $\tilde{\vec{\boldsymbol{F}}}_0$ satisfies the constraints in $\mathcal{R}^{\beta}_{\mathsf{lin},2}$. Based on Theorems 6 and 7, given an output instance in $\mathcal{R}_{\mathsf{lin,ba}}$, one can extract an instance in $\mathcal{R}^{\beta}_{\mathsf{lin},2}$ with a knowledge soundness error of at most $\epsilon_{\mathsf{ba}} + \epsilon_{\mathsf{rg}}$.

□

Theorem A6. *When* $\beta' \geq 2Td \cdot \sqrt{m} + T \cdot \beta$ *and* $\mathsf{Com}(\cdot)$ *is binding for norm bound* $\max(2\beta', 4T\beta')$*, the protocol* Π_{tool} *is knowledge sound with soundness error* $\epsilon_{\mathsf{tool}} = \epsilon_{\mathsf{tool,cwss}} + \frac{2d+2}{|\mathcal{C}|}$.

Proof. By Lemma A3 and Theorem A5, the claim holds. □

Appendix D. Proof for Theorem 12

Theorem A7. *When* $\mathcal{B}' \geq \frac{\sigma}{2}\sqrt{md}\,Tk\iota$ *and* $\mathsf{Com}(\cdot)$ *is binding for the norm* $\max(2\mathcal{B}', 4T\mathcal{B}', 2\mathcal{B}'\beta', 4T\mathcal{B}'\beta')$*,* Π_{amor} *gives a knowledge reduction from* $(\mathcal{R}^{\beta}_{\mathsf{poly},2})^k$ *to* $\mathcal{R}^{\mathcal{B}'}_{\mathsf{lin,f}}$ *with knowledge soundness error* $\epsilon_{\mathsf{lin}} + \epsilon_{\mathsf{tool}} + \frac{6k+2\log m}{|\mathcal{C}|} + \frac{2k\iota}{|\mathcal{C}'|}$.

Proof. Public reducibility follows from the properties of $\Pi_{\mathsf{zk\text{-}lin}}$ and $\Pi_{\mathsf{zk\text{-}tool}}$. Their completeness guarantees that an honest prover produces a witness satisfying all linear constraints in $\mathcal{R}^{\mathcal{B}'}_{\mathsf{lin,f}}$. For the norm bound, we have $\|\vec{\boldsymbol{y}}\|_2 \leq Tk\iota \cdot \|\vec{\boldsymbol{y}}_{i,j}\| = Tk\iota \cdot \frac{\sigma}{2}\sqrt{md} \leq \mathcal{B}'$, establishing perfect completeness.

Knowledge soundness follows from coordinate-wise special soundness. Given accepted transcripts

$$(\mathsf{tr}^k_{\mathsf{zk\text{-}lin}}, \mathsf{tr}^k_{\mathsf{zk\text{-}tool}}, (\mathsf{tr}_{\mathsf{de}}, \vec{c}_{a,\ell}))$$

for $\ell \in [k\iota+1]$, with $\vec{c}_{a,\ell} \in \mathsf{SS}(\mathcal{C}', k\iota, 2)$ and output instance $(\mathsf{cm}_{y_\ell}, s_{y_\ell}, \vec{y}_\ell)$, the extractor recovers relaxed openings $\widetilde{\vec{y}}_{i,j}$ such that $\mathsf{Com}(\widetilde{\vec{y}}_{i,j}) = \mathsf{cm}_{y_{i,j}}$ and $\|\bar{c}\,\widetilde{\vec{y}}_{i,j}\|_2 \leq 2B'$ for all $\bar{c} \in \mathcal{C}' - \mathcal{C}'$. The probability that the extracted witness violates the check in Step 10 while the transcript still verifies is at most $k\iota/|\mathcal{C}|$.

We reconstruct $\widetilde{\vec{y}}_i := \sum_{j=0}^{\iota-1} \sigma^j \widetilde{\vec{y}}_{i,j}$, $\mathsf{cm}_{y_i} := \sum_{j=0}^{\iota-1} \sigma^j \mathsf{cm}_{y_{i,j}}$, yielding $\mathsf{Com}(\widetilde{\vec{y}}_i) = \mathsf{cm}_{y_i}$. From Step 6 we also obtain $\langle \widetilde{\vec{y}}_i, \mathsf{tensor}(\vec{r}_0) \rangle = s_{y_i}$, incurring an additional soundness error of $(2\log m + k)/|\mathcal{C}|$. Hence $\widetilde{\vec{y}}_i$ is a valid opening for $\mathcal{R}^{\beta', B'}_{\mathsf{lin},f}$ —it satisfies all linear constraints in $\mathcal{R}^{\beta'}_{\mathsf{lin},f}$ except possibly the norm bound.

Apply the extractor of $\Pi_{\mathsf{zk\text{-}tool}}$ (Theorem 2) to $\widetilde{\vec{y}}_i$. Since $\widetilde{\vec{y}}_i$ satisfies all linear constraints of $\mathcal{R}^{\beta'}_{\mathsf{lin},f}$, and $\mathsf{Com}(\cdot)$ is binding at $\max(2B', 4TB', 2B'\beta', 4TB'\beta')$, the extractor outputs $(\widetilde{\vec{f}}_i, \widetilde{\vec{a}}_i)$ such that

$$\mathsf{Com}(\widetilde{\vec{f}}_i) = \mathsf{cm}_{f_i},\ \langle \widetilde{\vec{f}}_i, \mathsf{tensor}(\vec{r}_1) \rangle = s_{1,i},\ \langle \widetilde{\vec{f}}_i, \widetilde{\vec{f}}_i \rangle \bmod q \ \leq\ \beta^2,\ \|\widetilde{\vec{f}}_i\|_\infty \ \leq\ D, \tag{A3}$$

with knowledge-soundness error at most ϵ_{tool}. Here the third and fourth constraints arise because $\Pi_{\mathsf{zk\text{-}tool}}$ internally runs Π_{IPA} and $\Pi_{\mathsf{rg},\infty}$, which reduce the ℓ_2-norm bound to exactly these two linear constraints.

We now show that the extracted $\widetilde{\vec{f}}_i$ satisfies the *exact* norm bound $\|\widetilde{\vec{f}}_i\|_2 \le \beta$—not merely a relaxed version thereof. By the constraint set in (A3):

(i) $\langle \widetilde{\vec{f}}_i, \widetilde{\vec{f}}_i \rangle \bmod q \;\le\; \beta^2$, and

(ii) $\|\widetilde{\vec{f}}_i\|_\infty \;\le\; D$ where $q > 4D^2 \cdot md$.

Applying Lemma 2 with these two conditions: since $\|\widetilde{\vec{f}}_i\|_2^2 = \sum_{t=1}^{md} (\widetilde{f}_{i,t})^2 \le D^2 \cdot md < q/2$, the centered representative of $\langle \widetilde{\vec{f}}_i, \widetilde{\vec{f}}_i \rangle \bmod q$ equals its true integer value $\|\widetilde{\vec{f}}_i\|_2^2$. Therefore

$$\langle \widetilde{\vec{f}}_i, \widetilde{\vec{f}}_i \rangle \bmod q \;\le\; \beta^2 \quad\Longleftrightarrow\quad \|\widetilde{\vec{f}}_i\|_2^2 \;\le\; \beta^2 \quad\Longleftrightarrow\quad \|\widetilde{\vec{f}}_i\|_2 \;\le\; \beta.$$

Hence the extracted witness satisfies the original exact ℓ_2-norm bound. This is the key point: because Π_{IPA} encodes $\|\vec{f}\|_2 \le \beta$ as the linear constraints (i)–(ii), the extractor never handles a "relaxed norm" constraint; once those linear constraints are verified to hold exactly, Lemma 2 guarantees the original bound is recovered exactly, without any slack.

By Theorem 2, $\widetilde{\vec{f}}_i$ also satisfies $h(\widetilde{f}_{i,t}) = v_{i,t}$ for all $t \in [m]$ with knowledge-soundness error ϵ_{lin}. Hence $(\mathsf{cm}_{f_i}, \vec{v}_i; \widetilde{\vec{f}}_i) \in \mathcal{R}^{\beta}_{\mathsf{poly},2}$ for all $i \in [k]$. By Lemma A3, this establishes coordinate-wise special soundness and hence knowledge soundness with the stated error. Combining all error terms via a union bound:

$$\begin{aligned}\epsilon_{\mathsf{amor}} \;&\le\; \underbrace{\epsilon_{\mathsf{lin}}}_{\text{Thm. 2}} + \underbrace{\epsilon_{\mathsf{tool}}}_{\text{Thm. 9}} + \underbrace{\frac{k\iota}{|\mathcal{C}'|}}_{\text{Step 10}} + \underbrace{\frac{2\log m + k}{|\mathcal{C}|}}_{\text{Step 6}} + \underbrace{\frac{5k}{|\mathcal{C}|}}_{\text{parallel}} + \underbrace{\frac{k\iota}{|\mathcal{C}'|}}_{\text{cwss to ks}} \\ &=\; \epsilon_{\mathsf{lin}} + \epsilon_{\mathsf{tool}} + \frac{6k + 2\log m}{|\mathcal{C}|} + \frac{2k\iota}{|\mathcal{C}'|}.\end{aligned}$$

□

References

1. Goldwasser, S.; Micali, S.; Rackoff, C. The Knowledge Complexity of Interactive Proof-Systems. In *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 203–225. https://doi.org/10.1145/3335741.3335750.
2. Cramer, R.; Damgård, I.; Schoenmakers, B. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In *Advances in Cryptology—CRYPTO 1994, Proceedings of the 14th Annual International Cryptology Conference, Santa Barbara, CA, USA, 21–25 August 1994*; Springer: Berlin/Heidelberg, Germany, 1994; Volume 839, pp. 174–187. https://doi.org/10.1007/3-540-48658-5_19.
3. Abe, M.; Ambrona, M.; Bogdanov, A.; et al. Non-Interactive Composition of Sigma-Protocols via Share-Then-Hash. In *Advances in Cryptology—ASIACRYPT 2020, Proceedings of the 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, 7–11 December 2020*; Springer: Cham, Switzerland, 2020; Volume 12493, pp. 749–773. https://doi.org/10.1007/978-3-030-64840-4_25.
4. Maurer, U. Zero-Knowledge Proofs of Knowledge for Group Homomorphisms. *Des. Codes Cryptogr.* **2015**, *77*, 663–676. https://doi.org/10.1007/s10623-015-0103-5.
5. Zhang, M.; Chen, Y.; Yao, C.; et al. Sigma Protocols from Verifiable Secret Sharing and Their Applications. In *Advances in Cryptology—ASIACRYPT 2023, Proceedings of the 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, 4–8 December 2023*; Springer: Singapore, 2023; Volume 14439, pp. 208–242. https://doi.org/10.1007/978-981-99-8724-5_7.
6. Gao, S.; Qian, C.; Zheng, T.; et al. Σ-Check: Compressed Σ-Protocol Theory from Sum-Check. Cryptology ePrint Archive, Paper 2024/1654, 2024. Available online: https://eprint.iacr.org/2024/1654.pdf (accessed on 18 November 2026).
7. Attema, T.; Cramer, R. Compressed Σ-Protocol Theory and Practical Application to Plug & Play Secure Algorithmics. In *Advances in Cryptology—CRYPTO 2020, Proceedings of the 40th Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2020*; Springer: Cham, Switzerland, 2020; Volume 12172, pp. 513–543. https://doi.org/10.1007/978-3-030-56877-1_18.
8. Bünz, B.; Bootle, J.; Boneh, D.; et al. Bulletproofs: Short Proofs for Confidential Transactions and More. In Proceedings of the 2018 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 21–23 May 2018; pp. 315–334. https://doi.org/10.1109/sp.2018.00020.
9. Stern, J. A New Identification Scheme Based on Syndrome Decoding. In *Advances in Cryptology—CRYPTO 1993,*

Proceedings of the 13th Annual International Cryptology Conference, Santa Barbara, CA, USA, 22–26 August 1993; Springer: Berlin/Heidelberg, Germany, 1994; Volume 773, pp. 13–21. https://doi.org/10.1007/3-540-48329-2_2.

10. Esgin, M.F.; Steinfeld, R.; Liu, J.K.; et al. Lattice-Based Zero-Knowledge Proofs: New Techniques for Shorter and Faster Constructions and Applications. In *Advances in Cryptology—CRYPTO 2019, Proceedings of the 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2019*; Springer: Cham, Switzerland, 2019; Volume 11692, pp. 115–146. https://doi.org/10.1007/978-3-030-26948-7_5.
11. Attema, T.; Lyubashevsky, V.; Seiler, G. Practical Product Proofs for Lattice Commitments. In *Advances in Cryptology—CRYPTO 2020, Proceedings of the 40th Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2020*; Springer: Cham, Switzerland, 2020; Volume 12171, pp. 470–499. https://doi.org/10.1007/978-3-030-56880-1_17.
12. Lyubashevsky, V.; Nguyen, N.K.; Plançon, M. Lattice-Based Zero-Knowledge Proofs and Applications: Shorter, Simpler, and More General. In *Advances in Cryptology—CRYPTO 2022, Proceedings of the 42nd Annual International Cryptology Conference, Santa Barbara, CA, USA, 15–18 August 2022*; Springer: Cham, Switzerland, 2022; Volume 13508, pp. 71–101. https://doi.org/10.1007/978-3-031-15979-4_3.
13. Beullens, W.; Seiler, G. LaBRADOR: Compact Proofs for R1CS from Module-SIS. In *Advances in Cryptology—CRYPTO 2023, Proceedings of the 43rd Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 August 2023*; Springer: Cham, Switzerland, 2023; Volume 14085, pp. 518–548. https://doi.org/10.1007/978-3-031-38554-4_17.
14. Boneh, D.; Chen, B. LatticeFold+: Faster, Simpler, Shorter Lattice-Based Folding for Succinct Proof Systems. In *Advances in Cryptology—CRYPTO 2025, Proceedings of the 45th Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2025*; Springer: Cham, Switzerland, 2025; Volume 16006, pp. 327–361. https://doi.org/10.1007/978-3-032-01907-3_11.
15. Boneh, D.; Chen, B. LatticeFold: A Lattice-Based Folding Scheme and Its Applications to Succinct Proof Systems. In *Advances in Cryptology—ASIACRYPT 2025, Proceedings of the 31st International Conference on the Theory and Application of Cryptology and Information Security, Melbourne, VIC, Australia, 8–12 December 2025*; Springer: Singapore, 2026; Volume 16247, pp. 330–362. https://doi.org/10.1007/978-981-95-5099-9_11.
16. Attema, T.; Cramer, R.; Fehr, S. Compressing Proofs of *k*-out-of-*n* Partial Knowledge. In *Advances in Cryptology—CRYPTO 2021, Proceedings of the 41st Annual International Cryptology Conference, Online, 16–20 August 2021*; Springer: Cham, Switzerland, 2021; Volume 12828, pp. 65–91. https://doi.org/10.1007/978-3-030-84259-8_3.
17. Bootle, J.; Lyubashevsky, V.; Nguyen, N.K.; et al. A Non-PCP Approach to Succinct Quantum-Safe Zero-Knowledge. In *Advances in Cryptology—CRYPTO 2020, Proceedings of the 40th Annual International Cryptology Conference, Santa Barbara, CA, USA, 17–21 August 2020*; Springer: Cham, Switzerland, 2020; Volume 12171, pp. 441–469. https://doi.org/10.1007/978-3-030-56880-1_16.
18. Attema, T.; Cramer, R.; Kohl, L. A Compressed Σ-Protocol Theory for Lattices. In *Advances in Cryptology—CRYPTO 2021, Proceedings of the 41st Annual International Cryptology Conference, Online, 16–20 August 2021*; Springer: Cham, Switzerland, 2021; Volume 12826, pp. 549–579. https://doi.org/10.1007/978-3-030-84245-1_19.
19. Albrecht, M.R.; Lai, R.W.F. Subtractive Sets over Cyclotomic Rings: Limits of Schnorr-like Arguments over Lattices. In *Advances in Cryptology—CRYPTO 2021, Proceedings of the 41st Annual International Cryptology Conference, Online, 16–20 August 2021*; Springer: Cham, Switzerland, 2021; Volume 12826, pp. 519–548. https://doi.org/10.1007/978-3-030-84245-1_18.
20. Chen, S.; Cheon, J.H.; Kim, D.; et al. Verifiable Computing for Approximate Computation. Cryptology ePrint Archive, Paper 2019/762, 2019. Available online: https://eprint.iacr.org/2019/762.pdf (accessed on 18 November 2026).
21. Lyubashevsky, V.; Seiler, G. Short, Invertible Elements in Partially Splitting Cyclotomic Rings and Applications to Lattice-Based Zero-Knowledge Proofs. In *Advances in Cryptology—EUROCRYPT 2018, Proceedings of the 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, 29 April–3 May 2018*; Springer: Cham, Switzerland, 2018; Volume 10820, pp. 204–224. https://doi.org/10.1007/978-3-319-78381-9_8.
22. Xie, T.; Zhang, J.; Zhang, Y.; et al. Libra: Succinct Zero-Knowledge Proofs with Optimal Prover Computation. In *Advances in Cryptology—CRYPTO 2019, Proceedings of the 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, 18–22 August 2019*; Springer: Cham, Switzerland, 2019; Volume 11694, pp. 733–764. https://doi.org/10.1007/978-3-030-26954-8_24.
23. Langlois, A.; Stehlé, D. Worst-Case to Average-Case Reductions for Module Lattices. *Des. Codes Cryptogr.* **2015**, *75*, 565–599. https://doi.org/10.1007/s10623-014-9938-4.
24. Micciancio, D.; Regev, O. Lattice-Based Cryptography. In *Post-Quantum Cryptography*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 147–191. https://doi.org/10.1007/978-3-540-88702-7_5.
25. Ajtai, M. Generating Hard Instances of Lattice Problems (Extended Abstract). In Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC), Philadelphia, PA, USA, 22–24 May 1996; pp. 99–108. https://doi.org/10.1145/237814.237838.
26. Lyubashevsky, V. Lattice Signatures without Trapdoors. In *Advances in Cryptology—EUROCRYPT 2012, Proceedings of the 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, 15–19 April 2012*; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7237, pp. 738–755. https://doi.org/10.1007/978-3-642-29011-4_43.

27. Fenzi, G.; Moghaddas, H.; Nguyen, N.K. Lattice-Based Polynomial Commitments: Towards Asymptotic and Concrete Efficiency. *J. Cryptol.* **2024**, *37*, 31. https://doi.org/10.1007/s00145-024-09511-8.
28. Kothapalli, A.; Parno, B. Algebraic Reductions of Knowledge. In *Advances in Cryptology—CRYPTO 2023, Proceedings of the 43rd Annual International Cryptology Conference, Santa Barbara, CA, USA, 20–24 August 2023*; Springer: Cham, Switzerland, 2023; Volume 14084, pp. 669–701. https://doi.org/10.1007/978-3-031-38551-3_21.