Hachi: Efficient Lattice-Based Multilinear Polynomial Commitments over Extension Fields

Abstract. In this work, we present **Hachi**, a concretely efficient multilinear polynomial commitment scheme that offers succinct proof sizes of $\text{poly}(\ell, \lambda)$ and achieves "square-root" verifier time complexity of $\tilde{O}(\sqrt{2^\ell \lambda})$ for ℓ-variate polynomials, under the Module-SIS assumption. Compared to the current state-of-the-art scheme, Greyhound (CRYPTO 2024), Hachi provides an asymptotic improvement of $\tilde{O}(\lambda)$ in verification time, which translates to a practical speedup of at least 35%, while maintaining compact proofs of approximately 63KB.

As a separate contribution, we introduce a generic reduction that converts polynomial evaluation proofs over extension fields $\mathbb{F}_{q^k}$ (under suitable parameter regimes) into equivalent statements over power-of-two cyclotomic rings $\mathbb{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$. This reduction is compatible with existing lattice-based polynomial commitment schemes and can be integrated as a modular enhancement to broaden applicability to statements over extension fields.

1 Introduction

Multilinear polynomial commitment scheme (PCS) [KZG10; BBHR18] is a strong cryptographic primitive that allows a prover to commit to a ℓ-dimensional multilinear polynomial $f \in \mathbb{F}^{\leq 1}[X_1, \ldots, X_\ell]$ and later prove its evaluation

$$f(x_1, \ldots, x_\ell) = y$$

where the evaluation point $(x_1, \ldots, x_\ell) \in \mathbb{F}^\ell$ and the image $y \in \mathbb{F}$ are public. Multilinear polynomial commitments are a core building block for succinct non-interactive arguments of knowledge (SNARKs) [WTs+18; Set20; CHM+20; GLS+23; CBBZ23], look-up arguments [STW24] and multi-party computation [BHV+23]. In such applications, it is crucial for the polynomial evaluation proof to be succinct, i.e. polynomial in the number of variables ℓ, and be verified efficiently.

With the rapid advancement of quantum computing, there is increasing interest in quantum-safe PCS constructions. Among these, lattice-based and hash-based approaches have emerged as the leading candidates. Lattice-based PCS are particularly attractive due to their efficient prover runtime and compact proof sizes, both concretely [NS24] and asymptotically [KLNO25]. In contrast, hash-based constructions currently offer significantly faster verification (by two orders of magnitude) compared to lattice-based schemes [ACFY24; ACFY25]. This discrepancy raises a natural research question whether one can design a lattice-based multilinear polynomial commitment scheme that matches the verification efficiency of hash-based constructions, while retaining the compact proof sizes of the prior works [NS24].

1.1 Prior Works

Split-and-fold. At the core of state-of-the-art lattice-based succinct proof systems (including polynomial commitments) lies proving knowledge of vectors $\mathbf{s}_1, \ldots, \mathbf{s}_r \in \mathbf{R}_q^m$ such that

$$\mathbf{B}\begin{bmatrix}\mathbf{A} & & \\ & \ddots & \\ & & \mathbf{A}\end{bmatrix}\begin{bmatrix}\mathbf{s}_1 \\ \vdots \\ \mathbf{s}_r\end{bmatrix} = \mathbf{u} \pmod q \quad \text{and} \quad \|\mathbf{s}_i\| \leq \beta \quad \text{for } i = 1, \ldots, r \tag{1}$$

where $\mathbf{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$ for a power-of-two d, and the matrices $\mathbf{A}, \mathbf{B}$, vector $\mathbf{t}$ and norm bound β are public. The first interactive protocol for this relation was proposed by Baum et al. [BBC+18] which can be summarized as follows. The prover starts by sending the "partial evaluations" $\mathbf{t}_i = \mathbf{A}\mathbf{s}_i$ for $i \in [r]$ (this step is called *split*). The verifier responds by outputting short challenges $c_1, \ldots, c_r \in C$ over the cyclotomic ring $\mathbf{R}_q$. Finally, the prover sends returns the linear combination of witness vectors (this step is called *fold*)

$$\mathbf{z} := c_1\mathbf{s}_1 + \ldots c_r\mathbf{s}_r \in \mathbf{R}_q^m. \tag{2}$$

The verifier then checks three conditions:

$$\mathbf{B}\begin{bmatrix}\mathbf{t}_1 \\ \vdots \\ \mathbf{t}_r\end{bmatrix} = \mathbf{u}, \qquad \mathbf{A}\mathbf{z} = \sum_{i=1}^{r} c_i\mathbf{t}_i \qquad \text{and} \qquad \|\mathbf{z}\| \text{ is short.} \tag{3}$$

The complexity of the communication of the prover is $r + m$ ring elements, which is strictly sublinear in the original witness length $N := r \cdot m \cdot d$ in terms of $\mathbb{Z}_q$-elements. By picking parameters appropriately, [BBC+18] obtains "square-root" proofs of size $\tilde{O}(\sqrt{N\lambda})$, where λ is the security parameter.

Multiple follow-up works [BLNS20; AFLN24; CMNW24; KLNO24] managed to reduce the proof size further to polylog(N) ring elements. The key insight lies in making the matrix $\mathbf{A}$ to have a tensor structure similar to the matrix on the left-hand side of Equation (1). This allows the prover, instead of sending the final message $\mathbf{z}$ in the clear, to prove knowledge of $\mathbf{z}$ satisfying the verification equation of (3) in a recursive fashion.

Exact norm proof. Arguably, the hardest part of designing proof systems for relations such as (1) is proving that vectors $\mathbf{s}_i$ are actually short. Indeed, knowledge soundness of the original protocol of [BBC+18] only guarantees extraction of (not necessarily short) $\bar{\mathbf{s}}_1, \ldots, \bar{\mathbf{s}}_r \in \mathbf{R}_q^m$ of the form

$$\bar{\mathbf{s}}_i := \frac{\bar{\mathbf{z}}_i}{\bar{c}_i} \quad \text{where both } \bar{\mathbf{z}}_i, \bar{c}_i \text{ are short.}$$

There is a line of research [BLNS20; AL21; ACK21; CLM23; FMN24; AFLN24; KLNO24] which addresses this issue using so-called subtractive sets. A set $C \subseteq \mathbf{R}_q$ is called subtractive if for any two distinct elements $c, c' \in C$, inverse of the difference $c - c'$

over $\mathbf{R}_q$ is short. This allows the knowledge extractor to deduce that the extracted $\bar{\mathbf{s}}_i$ are short. This approach has one crucial disadvantage, which is that all subtractive sets are only of polynomial size [AL21]. Consequently, to obtain negligible soundness error, multiple repetitions are needed which affect the efficiency of the scheme significantly.

An alternative approach to proving shortness, which avoids subtractive sets, involves applying random projections to the witness vector $\mathbf{s}$ [BL17; LNS21; LNP22; GHL22; BS23; KLNO25]. Let $\mathbf{s}' \in \mathbb{Z}_q^{md}$ denote the coefficient vector of $\mathbf{s} \in \mathbf{R}_q^m$. Consider a matrix $\mathbf{J} \in \mathbb{Z}^{r \times md}$ with small, independently chosen random entries, where r is a soundness parameter. The Johnson-Lindenstrauss Lemma (and heuristic variants [Ach03; GHL22]) implies that

$$\|\mathbf{J}\mathbf{s}'\| \approx \|\mathbf{s}'\| = \|\mathbf{s}\|.$$

Using this fact, the verifier sends the projection matrix $\mathbf{J}$ to the prover (or in the non-interactive case it is generated from a hash function), who responds with $\mathbf{v} := \mathbf{J}\mathbf{s}' \in \mathbb{Z}^r$. The verifier then checks whether $\mathbf{v}$ is short. For soundness, previous works show that if $\mathbf{s}$ were large (i.e. a prover was malicious), then with probability $\exp(-r)$, $\mathbf{v}$ would also have large norm, and thus result in a contradiction. Hence, $r = O(\lambda)$ is required for soundness.

The prover must also prove that $\mathbf{v}$ is well-formed. As shown in [LNP22; BS23], this reduces to proving that certain r polynomials $u_1, u_2, \ldots, u_r \in \mathbf{R}_q$ have zero constant coefficients. One method is to send all $(u_i)_{i \in [r]}$ in the clear for the verifier to check, but this requires sending at least $O(\lambda)$ ring elements, resulting in large proofs. A more efficient method batches the r statements: the verifier sends r random challenges $\alpha_1, \ldots, \alpha_r \leftarrow \mathbb{Z}_q$, and the prover returns the linear combination

$$u := \sum_{i=1}^{r} \alpha_i \cdot u_i \in \mathbf{R}_q.$$

The verifier checks that the constant coefficient of u is zero. For soundness, if some u_i had a non-zero constant coefficient, then with probability $1/q$, so would u, leading to a contradiction. Since $1/q$ may not be negligible, this must be repeated $O(\lambda/\log q)$ times to achieve $\exp(-\lambda)$ soundness error.

Concretely, the Johnson-Lindenstrauss projection has shown to be a practical approach to prove norm bounds compared to subtractive sets. The prime example is the LaBRADOR proof system introduced by Beullens and Seiler, which achieves $\approx$ 60KB proofs for large statements (of size 2^{30}). The main disadvantage of the random projection methodology is the verification time. Indeed, the verifier has to operate on the projection matrix $\mathbf{J}$ of size $O(\lambda \cdot md)$ – linear in witness length (in terms of $\mathbb{Z}_q$-elements). This makes the LaBRADOR verifier linear time, and thus not suitable for various SNARK-oriented applications. More recently, Klooß et al. [KLNO25] focused on tensor-structured projection matrices J, and managed to reduce the verification time to polylog(md), at the cost of 40X larger proof sizes compared to LaBRADOR.

Nguyen and Seiler [NS24] proposed a promising hybrid polynomial commitment, called Greyhound. Their evaluation proof intuitively follows the three-round protocol from [BBC+18], but instead of sending the prover message in the clear, the prover

Scheme	Multi -linear	Extension fields	Asymptotic proof	Asymptotic verifier	Concrete proof	Concrete verifier
Greyhound	✗	✗	$\text{poly}(\ell, \lambda)$	$\tilde{O}(\lambda \cdot \sqrt{2^\ell \lambda})$	53KB	2.8s
Hachi	✓	✓	$\text{poly}(\ell, \lambda)$	$\tilde{O}(\sqrt{2^\ell \lambda})$	63KB	1.8s

Fig. 1. Overview of Hachi and comparison with the state-of-the-art lattice-based polynomial commitment scheme Greyhound [NS24]. The asymptotic verifier time is measured in $\mathbb{Z}_q$-operations. We use the $\tilde{O}$-notation to hide logarithmic factors. The concrete numbers are taken from evaluating the polynomial with $\ell = 30$ variables.

commits to $(\mathbf{t}_i)_i, \mathbf{z}$ and later proves that they satisfy the verification equations in (3) via the LaBRADOR proof system. The key advantage of this strategy is that invoking LaBRADOR on a smaller statement yields square-root $\tilde{O}(\lambda\sqrt{N\lambda})$ verification time, while benefiting from the attractive proof sizes of [BS23].

Several recent works have used the sumcheck protocol [LFKN90] to prove infinity norm bounds of the committed vector, albeit in the context of folding schemes [KST22]. This strategy was first applied in the lattice setting by Boneh and Chen in LatticeFold [BC24] to prove binary (NTT) coefficients of the committed vector. As a drawback, the protocol runs sumcheck over the cyclotomic ring $\mathbf{R}_q$, which significantly increases the proof size. Neo [NS25] later addresses this issue by mapping all the necessary $\mathbf{R}_q$-relations into $\mathbb{Z}_q$ via rotational (i.e. skew-circulant) matrices, which allows them to run sumcheck directly over $\mathbb{Z}_q$ (or over the extension field $\mathbb{F}_{q^k}$ to ensure negligible soundness). Concurrently, LatticeFold+ [BC25] proposed a novel approach to prove range as follows. Namely, to prove $a \in [0, d)$, the prover commits to the corresponding monomial $X^a \in \mathbf{R}_q$ and later proves that the committed ring element is indeed a monomial. LatticeFold+ reduces this (and other related statements) to sumcheck claims over $\mathbf{R}_q$. As opposed to [BC24; NS25], the authors notice that the claims contain no "full-fledged" $\mathbf{R}_q$ multiplications, and can thus be naturally reduced to $O(d)$ sumcheck claims over $\mathbb{Z}_q$. The main efficiency bottleneck of this approach is that for a witness $\mathbf{s}' \in \mathbb{Z}_q^{md}$ of length md, the prover has to commit to md $\mathbf{R}_q$-elements, as opposed to m as in [BS23; NS24; NS25] (excluding decompositions).

1.2 Our Contributions

In this work, we propose Hachi, a concretely efficient multilinear polynomial commitment scheme with succinct $\text{poly}(\ell, \lambda)$ proof size and $\tilde{O}(\sqrt{2^\ell \lambda})$ verifier time for ℓ-variate polynomials based on standard lattice assumptions (and the random oracle model for the Fiat-Shamir transformation). Hence, asymptotically we achieve $\tilde{O}(\lambda)$ improvement in verification time compared to Greyhound [NS24]. In Figure 1 we provide more details on the comparison. Our results can be summarized as follows.

Embedding multilinear evaluation claims inside $\mathbf{R}_q$. Our first contribution is a generic transformation from proving ℓ-variate polynomial evaluations over finite fields $\mathbb{F}_{q^k}$ to proving $(\ell - \alpha + \kappa)$-variate polynomial evaluations over the cyclotomic ring $\mathbf{R}_q$, where $d = 2^\alpha$ and $k = 2^\kappa$. Thus, we generalize the result of Albrecht et al. [AFLN24] in

two orthogonal directions: (i) we support polynomials over field extensions (for specific parameter sets), rather than just prime-order fields, and (ii) we extend the strategy from univariate to the multivariate case. Using our methodology, one can already adapt Greyhound [NS24] to prove multilinear polynomial evaluations out-of-the-box. We also provide an optimization if the polynomial f itself is only over $\mathbb{F}_q$ while the evaluation points still lie in $\mathbb{F}_{q^k}$.

Faster verification via ring switching. Unlike Greyhound [NS24], we move away from the random projection strategy and consider the sumcheck-based approach to prove infinity norm bounds. To this end, we apply the ring switching technique introduced in [HMZ25], which lifts statements over $\mathbf{R}_q$ to the polynomial ring $\mathbb{Z}_q[X]$. This allows us to evaluate the equations over a random point $X := \alpha \in \mathbb{F}_{q^k}$ and run the sumcheck protocol over the extension field $\mathbb{F}_{q^k}$. The total complexity of the verifier of our scheme is therefore dominated by $\tilde{O}(\sqrt{2^\ell \lambda})$.

Flexibility in picking ring dimension and implications. Both Greyhound and LaBRADOR have certain limitations in picking the ring dimension d. One of the main obstacles is when applying random projection strategy, one has to prove that certain ring elements have constant coefficients equal to zero. As described above, this comes at the cost of the prover sending $O(\lambda/\log q)$ ring elements after each iteration of LaBRADOR. Hence, choosing d (equal to 64 in [BS23; NS24]) as small as possible is essential to achieve small proof sizes. In our construction, since we focus on the sumcheck-based norm proofs, the prover only sends 1 $\mathbf{R}_q$-element in every iteration, which gives us more freedom in choosing ring dimension d with smaller impact on the communication size. In particular, picking larger d has two efficiency benefits: (i) the committing time is significantly faster thanks to the optimized NTT-based ring multiplication, and (ii) this allows us to define a challenge space C of *sparse* ring elements - thus speeding up the ring multiplication for computing the folded witness as in (2).

Implementation. We have developed a prototype implementation of Hachi in Rust[1]. While the implementation is still in its early stages, it is amenable to several optimizations—particularly those leveraging SIMD—which could significantly enhance performance. Even at this stage, Hachi demonstrates faster verification times than Greyhound, especially when handling large witnesses (i.e., polynomials with 30 or more variables). Additionally, thanks to the aforementioned flexibility in the choice of d, the commitment phase also benefits from improved efficiency[2]. For a detailed comparison with Greyhound, we refer the reader to Appendix C.9.

1.3 Technical Overview

Denote λ as a security parameter. Let $d := 2^\alpha$ be a power-of-two and $\mathbf{R} := \mathbb{Z}[X]/(X^d+1)$ be the ring of integers of the $2d$-th cyclotomic field. Take an odd prime q and define

[1] Source code is available at https://github.com/logger-pileup/hachi.

[2] Concretely, we obtain 25X speed-up by picking $d = 1024$ compared to $d = 64$ as in Greyhound.

$\mathbf{R}_q := \mathbf{R}/(q)$ and $\delta := \lceil \log q \rceil$. For $k \geq 1$, we denote $\mathbb{F}_{q^k}$ to be a finite field of order q^k. In this overview, we consider base-two gadget matrices $\mathbf{G}_n := \mathbf{I}_n \otimes \left[1\ 2\ 4 \cdots\ 2^{\delta-1}\right] \in \mathbf{R}_q^{n \times n\delta}$ for $n \geq 1$. We define the standard inverse function $\mathbf{G}_n^{-1} : \mathbf{R}_q^n \to \mathbf{R}_q^{n\delta}$, which decomposes each entry w.r.t. base 2. In particular, for any $\mathbf{t} \in \mathbf{R}_q^n$, $\mathbf{G}_n^{-1}(\mathbf{t})$ has binary coefficients and $\mathbf{G}_n\mathbf{G}_n^{-1}(\mathbf{t}) = \mathbf{t}$.

For $i \in \mathbb{Z}_{2d}^\times$, we define the Galois automorphism $\sigma_i : \mathbf{R} \to \mathbf{R}$ defined as $X \mapsto X^i$, and denote the Galois group of automorphisms as $\mathsf{Aut}(\mathbf{R}) := \{\sigma_i : i \in \mathbb{Z}_{2d}^\times\}$. For a subgroup H of $\mathsf{Aut}(\mathbf{R})$, we denote $\mathbf{R}_q^H := \{x \in \mathbf{R}_q : \forall \sigma \in H, \sigma(x) = x\}$, i.e. the ring of elements of $\mathbf{R}_q$ fixed by all automorphisms in H. Also, we denote the $\mathbf{R}_q$-trace map $\mathrm{Tr}_H : \mathbf{R}_q \to \mathbf{R}_q^H$ as $\mathrm{Tr}_H(a) := \sum_{\sigma \in H} \sigma(a) \in \mathbf{R}_q^H$.

Reducing multilinear evaluation claims to relations over $\mathbf{R}_q$. Our first step will be to translate the multilinear evaluation over $\mathbb{F}_{q^k}$, i.e. $f(x_1, \ldots, x_\ell) = y$, to an equivalent relation over $\mathbf{R}_q$ (or a related ring). The hardness lies in translating operations over field extensions $\mathbb{F}_{q^k}$, such as addition and multiplication, to equivalent operations over the cyclotomic ring $\mathbf{R}_q$. We proceed in two steps. First, we directly identify the finite fields within $\mathbf{R}_q$.

Lemma 1 (Informal). *Let $q = 5 \pmod 8$ and $k \geq 1$ be a divisor of $d/2$ and consider the subgroup $H := \langle -1, 4k+1 \rangle$ of $\mathbb{Z}_{2d}^\times$. Then, $\mathbf{R}_q^H$ is a subfield of $\mathbf{R}_q$ isomorphic to $\mathbb{F}_{q^k}$.*

This lemma allows us to translate proving inner products over $\mathbb{F}_{q^k}$ into relations directly over the cyclotomic ring $\mathbf{R}_q$ using the $\mathbf{R}_q$-trace map.

Theorem 1 (Informal). *Let $q = 5 \pmod 8$, k divide $d/2$ and $H := \langle \sigma_{-1}, \sigma_{4k+1} \rangle \subseteq \mathbb{Z}_{2d}^\times$. Then, there exists an efficiently computable bijection $\psi : (\mathbf{R}_q^H)^{d/k} \to \mathbf{R}_q$ such that for any $\mathbf{a}, \mathbf{b} \in (\mathbf{R}_q^H)^{d/k}$,*

$$\mathrm{Tr}_H(\psi(\mathbf{a}) \cdot \sigma_{-1}(\psi(\mathbf{b}))) = \frac{d}{k} \cdot \langle \mathbf{a}, \mathbf{b} \rangle.$$

We now show how to use these two theoretical results. Let $k = 2^\kappa$ be a divisor of $d/2$ and suppose we want to prove a polynomial evaluation claim over $\mathbf{R}_q^H$ ($\cong \mathbb{F}_{q^k}$ by Lemma 1). Denote the coefficients of polynomial f as $(f_\iota)_{\iota \in \{0,1\}^\ell} \in \mathbb{F}_{q^k}$. We can explicitly rewrite the evaluation equation as

$$y = \sum_{i \in \{0,1\}^{\ell-\alpha+\kappa}} \left(x_1^{i_1} \cdot \ldots \cdot x_{\ell-\alpha+\kappa}^{i_{\ell-\alpha+\kappa}}\right) \cdot \left(\sum_{j \in \{0,1\}^{\alpha-\kappa}} f_{i\|j} \cdot x_{\ell-\alpha+\kappa+1}^{j_1} \cdot \ldots \cdot x_\ell^{j_{\alpha-\kappa}} \right). \tag{4}$$

For each $i \in \{0,1\}^{\ell-\alpha+\kappa}$ define the ring element

$$F_i := \psi\left((f_{i||j})_{j \in \{0,1\}^{\alpha-\kappa}}\right) \quad \text{and also} \quad v := \psi\left((x_{\ell-\alpha+\kappa+1}^{j_1} \cdot \ldots \cdot x_\ell^{j_{\alpha-\kappa}})_{j \in \{0,1\}^{\alpha-\kappa}}\right).$$

Then, using the fact that $\mathbf{R}_q^H$ is stable under the additively homomorphic map tr_H, and by applying Theorem 1 to Equation (4) we get

$$\frac{d}{k} \cdot y = \mathrm{Tr}_H \left(\underbrace{\sum_{i \in \{0,1\}^{\ell-\alpha+\kappa}} \left(x_1^{i_1} \cdot \ldots \cdot x_{\ell-\alpha}^{i_{\ell-\alpha+\kappa}} \right) \cdot F_i}_{Y:=} \cdot \sigma_{-1}(v) \right).$$

Therefore, the prover can send a single ring element $Y \in \mathbf{R}_q$, that allows the verifier to check if the $\mathbf{R}_q$-trace map of $Y \cdot \sigma_{-1}(v)$ is equal to dy/k. Then, we only need to prove that Y is well-formed, which is now an evaluation proof for the $(\ell - \alpha + \kappa)$-variate polynomial $F := (F_i)_i$ over $\mathbf{R}_q$.

In certain scenarios, the evaluation point might be defined over the extension field $\mathbb{F}_{q^k}$, while the coefficients of the committed polynomial f are over $\mathbb{F}_q$ (which is the case after the first iteration of our recursive proof system). Can we optimize our transformation in this setting without bluntly assuming that $f \in \mathbb{F}_{q^k}^{\leq 1}[X_1, \ldots, X_\ell]$? To this end, we use the similar observation as before that a polynomial evaluation claim can be written as

$$y = \sum_{i \in \{0,1\}^\kappa} \left(x_1^{i_1} \cdot \ldots \cdot x_\kappa^{i_\kappa} \right) \cdot \left(\underbrace{\sum_{j \in \{0,1\}^{\ell-\kappa}} f_{i \| j} \cdot x_{\kappa+1}^{j_1} \cdot \ldots \cdot x_\ell^{j_{\ell-\kappa}}}_{y_i:=} \right). \tag{5}$$

Hence, we let the prover send k partial evaluations $y_i \in \mathbb{F}_{q^k}$ to the verifier, who can directly check the equation above[3]. What we have left to prove is that all the y_i are well-formed, or in other words $f_i(x_{\kappa+1}, \ldots, x_\ell) = y_i$ where $f_i \in \mathbb{F}_q[X_{\kappa+1}, \ldots, X_\ell]$.

Next, denote $\mathbb{F}_{q^k} := \mathbb{F}_q[Z]/\varphi(Z)$, where φ is an irreducible polynomial in $\mathbb{F}_q[Z]$ of degree k. Then, define the $(\ell-\kappa)$-variate multilinear polynomial $f' \in \mathbb{F}_{q^k}[X_{\kappa+1}, \ldots, X_\ell]$:

$$f'(X_{\kappa+1}, \ldots, X_\ell) := \sum_{i \in \{0,1\}^\kappa} f_i(X_{\kappa+1}, \ldots, X_\ell) \cdot Z^{\sum_{t=1}^{\kappa} i_t \cdot 2^{t-1}}.$$

Then, it is easy to see that

$$f'(x_{\kappa+1}, \ldots, x_\ell) = \sum_{i \in \{0,1\}^\kappa} y_i \cdot Z^{\sum_{t=1}^{\kappa} i_t \cdot 2^{t-1}} \in \mathbb{F}_{q^k}$$

which is now a full-fledged polynomial evaluation over $\mathbb{F}_{q^k}$. By applying the generic transformation above, we can reduce the statement into an evaluation claim over $\mathbf{R}_q$ for a $(\ell - \alpha)$-variate multilinear polynomial (instead of $\ell - \alpha + \kappa$, if done generically).

[3] A simple optimization is for the prover to send only $k - 1$ partial evaluation y_i, excluding $y_{0\ldots0} \in \mathbb{F}_{q^k}$, which can be deterministically derived from the verification equation.

Polynomial evaluation claim as a quadratic equation. Now, consider proving an evaluation claim of a μ-variate multilinear polynomial f over $\mathbf{R}_q$. We use the following folklore strategy. Suppose $\mu = m + r$ for some $m, r \geq 1$ and let $\mathbf{f} := (f_\iota)_{\iota \in \{0,1\}^\mu}$ be the coefficient vector of f. Then, for any

$$f(X_1, \ldots, X_\mu) := \sum_{i \in \{0,1\}^r} \sum_{j \in \{0,1\}^m} f_{i||j} \cdot (X_1^{i_1} \cdot \ldots \cdot X_r^{i_r}) \cdot (X_{r+1}^{j_1} \cdot \ldots \cdot X_\mu^{j_m})$$

and any evaluation point $(x_1, \ldots, x_\mu) \in \mathbf{R}_q^\mu$ we can write:

$$f(x_1, \ldots, x_\mu) = \mathbf{b}^\intercal \begin{bmatrix} \mathbf{a}^\intercal & & \\ & \ddots & \\ & & \mathbf{a}^\intercal \end{bmatrix} \begin{bmatrix} \mathbf{f}_{0\ldots0} \\ \vdots \\ \mathbf{f}_{1\ldots1} \end{bmatrix} \quad \begin{aligned} \mathbf{b}^\intercal &:= (x_1^{i_1} \cdot \ldots \cdot x_r^{i_r})_{i \in \{0,1\}^r} \in \mathbf{R}_q^{2^r} \\ \text{where } \mathbf{a}^\intercal &:= (x_{r+1}^{j_1} \cdot \ldots \cdot x_\mu^{j_m})_{j \in \{0,1\}^m} \in \mathbf{R}_q^{2^m} \\ \mathbf{f}_i^\intercal &:= (f_{i||j})_{j \in \{0,1\}^m} \in \mathbf{R}_q^{2^m} \text{ for } i \in \{0,1\}^r \end{aligned}.$$

Finally, by defining the decomposition $\mathbf{s} := \mathbf{G}_{2^\mu}^{-1}(\mathbf{f}) \in \mathbf{R}_q^{2^m \delta}$ and using the mixed product of the tensor product, we deduce that

$$\begin{aligned} f(x_1, \ldots, x_\mu) &= \mathbf{b}^\intercal (\mathbf{a}^\intercal \otimes \mathbf{I}_{2^r}) \mathbf{f} \\ &= \mathbf{b}^\intercal (\mathbf{a}^\intercal \otimes \mathbf{I}_{2^r}) (\mathbf{g}^\intercal \otimes \mathbf{I}_{2^{m+r}}) \mathbf{s} \\ &= \mathbf{b}^\intercal (\mathbf{a}^\intercal (\mathbf{g}^\intercal \otimes \mathbf{I}_{2^r}) \otimes \mathbf{I}_{2^r}) \mathbf{s}. \end{aligned}$$

Hence, the goal is to prove that the (committed) vector $\mathbf{s}$ is short and satisfies the quadratic equation above, which exactly the relation formulated in Equation (1). To prove knowledge of such $\mathbf{s}$, we closely follow the initial methodology from Greyhound [NS24]. That is, we consider the square-root interactive proof from [BBC+18], but instead of sending the prover messages in the clear, we commit to them. Later, we need to prove knowledge of the prover messages that satisfy all the verification equations (as well as proving knowledge of the commitment opening). All these relations can be phrased as proving knowledge of a short vector $\mathbf{z}$ of dimension $(2^m + 2^r) \cdot \text{poly}(\lambda)$ such that

$$\mathbf{M}\mathbf{z} = \mathbf{w} \quad \text{and} \quad \mathbf{z} \text{ is short.} \tag{6}$$

This is the point where our strategy diverges from [NS24]. Instead of running LaBRADOR [BS23] to prove the relation above, we propose a new sumcheck-based solution.

Ring switching and sumcheck over extension fields. We take inspiration from the ring switching approach proposed recently by Huang, Mao, and Zhang [HMZ25].To begin with, recall that a multilinear extension $\mathsf{mle}[f] \in \mathbb{F}_{q^k}^{\leq 1}[X_1, \ldots, X_\mu]$ of f is defined as

$$\mathsf{mle}[f](\mathbf{x}) := \sum_{i \in \{0,1\}^\mu} f(\mathbf{b}) \cdot eq(i, \mathbf{x})$$

where $eq(i, \mathbf{x}) := \prod_{j=1}^{\mu} \big((1 - i_j) \cdot (1 - x_j) + i_j x_j\big)$ is the equality polynomial.

The first step is to lift Equation (6) from $\mathbf{R}_q$ to $\mathbb{Z}_q[X]$. For the sake of overview, we only focus on the unstructured linear relation and suppose matrix $\mathbf{M}$ has only one

row (dealing with multiple rows and proving norm bounds is presented in Section 4.3). Then, we can rewrite it as

$$\sum_k M_k(X) \cdot z_k(X) = w(X) + (X^d + 1) \cdot r(X) \quad \text{for some } r \in \mathbb{Z}_q^{<d-1}[X].$$

Let $\mathbf{z}'$ (resp. $\mathbf{r}'$) be the $\mathbb{Z}_q$-coefficient vector of $\mathbf{z}$ (resp. r). Then, the prover commits, by using a polynomial commitment scheme, to the multilinear extension $P := \mathsf{mle}[(\mathbf{z}', \mathbf{r}')] \in \mathbb{F}_q^{\leq 1}[X_1, \ldots, X_\mu]$ of $(\mathbf{z}', \mathbf{r}')$[4], before obtaining a challenge $\alpha \leftarrow \mathbb{F}_{q^k}$ from the verifier. The challenge is then used to substitute $X = \alpha$, which simplifies to an inner product claim purely over $\mathbb{F}_{q^k}$. We note that the size of P is much smaller than that of the original committed polynomial f; hence, we can recursively repeat this process. Similarly as in [HMZ25] this can further get transformed into proving a sumcheck-type relation

$$\sum_{i \in \{0,1\}^\mu} P(i) \cdot Q(i) = V$$

for a public polynomial Q over $\mathbb{F}_{q^k}[X_1, \ldots, X_\mu]$ and a public value $V \in \mathbb{F}_{q^k}$.

On the other hand, to prove that $\mathbf{z}$ is short, we only need to show that all coordinates of $\mathbf{z}'$ are small. Here we can use a standard range proof over a finite field, since the ring switching has transformed the statements from polynomial rings into a finite field. In contrast, both LaBRADOR and LatticeFold+ demonstrate smallness directly over polynomial rings, which is considerably more complicated and less efficient for the verifier.

Finally, by running the sumcheck protocol [LFKN90], we end up with a polynomial evaluation claim of the form:

$$P(r_1^*, \ldots, r_\mu^*) \cdot Q(r_1^*, \ldots, r_\mu^*) = y^*,$$

where the evaluation point $(r_1^*, \ldots, r_\mu^*)$ and the image y^* are public. Hence, by sending the evaluation $P(r_1^*, \ldots, r_\mu^*)$ to the verifier, they can directly check the equation above. What is left for the prover to do now is to prove the evaluation claim for the polynomial P, and thus the protocol can be repeated in a further way to obtain smaller proof sizes.

Committing to P and avoiding re-decomposition. A naive approach to commit to the multilinear polynomial $P = \mathsf{mle}[(\mathbf{z}', \mathbf{r}')]$ is to first decompose the coefficients of P (and thus the committed witness gets longer), and then apply an Ajtai-style [Ajt96] commitment scheme. A more natural and efficient way is to directly commit to $(\mathbf{z}', \mathbf{r}')$, as we know that the coefficients of $\mathbf{z}'$ are already small and thus no decomposition for $\mathbf{z}'$ is needed. But this directly leads to a question of how to directly prove that $\mathsf{mle}[(\mathbf{z}', \mathbf{r}')](x_1, \ldots, x_L) = y'$, when having only committed to $(\mathbf{z}', \mathbf{r}')$. We solve this issue by observing a very simple homomorphic property of the *eq* polynomials. Namely, for any vectors i, j and $\mathbf{x}_0, \mathbf{x}_1$ we have

$$eq(i, \mathbf{x}_0) \cdot e(j, \mathbf{x}_1) = eq(i||j, \mathbf{x}_0||\mathbf{x}_1).$$

[4] Here, we implicitly treat a vector $\mathbf{a} \in \mathbb{F}_q^{2^L}$ as an indexing function from $\{0,1\}^L$ to $\mathbb{F}_q$.

Then, by definition of the multilinear extension and for $\mu = m + r$, we can write

$$\begin{aligned}\mathsf{mle}[(\mathbf{z}', \mathbf{r}')](\mathbf{x}_0 \parallel \mathbf{x}_1) &= \sum_{i \in \{0,1\}^r} \sum_{j \in \{0,1\}^m} (\mathbf{z}', \mathbf{r}')_{i \parallel j} \cdot eq(i \parallel j, \mathbf{x}_0 \parallel \mathbf{x}_1) \\ &= \sum_{i \in \{0,1\}^r} eq(i, \mathbf{x}_0) \cdot \left(\sum_{j \in \{0,1\}^m} (\mathbf{z}', \mathbf{r}')_{i \parallel j} \cdot eq(j, \mathbf{x}_1) \right)\end{aligned}$$

which is exactly of the same form as (4). Therefore, we can recursively run our protocol to prove evaluation of P (with minor modifications) without explicitly calculating its coefficients.

2 Preliminaries

Notation. Let $q = 5 \pmod 8$ be an odd prime. Denote $\mathbb{Z}_q := \{0, 1, \ldots, q-1\}$ to be the ring of integers of order q and $\mathbb{F}_{q^k}$ be the Galois field of dimension k and size q^k. In this paper, we will use $\mathbb{Z}_q$ and $\mathbb{F}_q$ interchangeably. For $n \in \mathbb{N}$, we define $[n] := \{1, 2, \ldots, n\}$. Let λ to be the security parameter. We write $O_\lambda(T)$ to denote $T \cdot \mathrm{poly}(\lambda)$. For a probability distribution $\mathcal{X}$ (resp. finite set $\mathcal{X}$), $x \leftarrow \mathcal{X}$ means that x is sampled from $\mathcal{X}$ (resp. x is chosen uniformly at random from the set $\mathcal{X}$). We write negl to denote an unspecified negligible function.

We will mostly use letters i, j for indexing. For $n \in \mathbb{N}$, we write $[n] := \{1, 2, \ldots, n\}$. When we denote $i \in \{0,1\}^k$, we treat i as a binary vector $(i_1, \ldots, i_k)$ of length k. For presentation purposes, when we say $i \in [2^k]$, we may also mean its binary encoding $i \in \{0,1\}^k$. This will be useful when describing the sumcheck-based norm protocol (cf. Section 4.3).

2.1 Cyclotomic Rings

For a power of two $d := 2^\alpha$ and prime q, denote $\mathbf{R}$ and $\mathbf{R}_q$ respectively to be the rings $\mathbb{Z}[X]/(X^d+1)$ and $\mathbb{Z}_q[X]/(X^d+1)$. Lower-case letters denote elements in $\mathbf{R}$ or $\mathbf{R}_{q^k}$ and bold lower-case (resp. upper-case) letters represent column vectors (resp. matrices) with coefficients in $\mathbf{R}$ or $\mathbf{R}_{q^k}$. For $y = \sum_{i=0}^{d-1} y_i \cdot X^i \in \mathbf{R}_q$, we write $\mathtt{cf}(y) := (y_0, \ldots, y_{d-1}) \in \mathbb{F}_q^d$ to denote the coefficient vector of y. We naturally extend this notation to vectors.

We consider base-b gadget matrices $\mathbf{G}_{b,n} := \mathbf{I}_n \otimes \left[1 \; b \; b^2 \; \cdots \; b^{\delta-1}\right] \in \mathbf{R}_q^{n \times n\delta}$ for $n \geq 1$ and $\delta = \lceil \log_b q \rceil$. We define the inverse function $\mathbf{G}_{b,n}^{-1} : \mathbf{R}_q^n \to \mathbf{R}_q^{n\delta}$, which decomposes each entry w.r.t. base b. In particular, for any $\mathbf{t} \in \mathbf{R}_q^n$, $\mathbf{G}_{b,n}^{-1}(\mathbf{t})$ has coefficients between 0 and $b-1$ and $\mathbf{G}_{b,n}\mathbf{G}_{b,n}^{-1}(\mathbf{t}) = \mathbf{t}$. We will omit the subscript b if the base is known and fixed from the context.

Galois Automorphisms and trace map. For $i \in \mathbb{Z}_{2d}^\times$, we define the Galois automorphism $\sigma_i : \mathbf{R} \to \mathbf{R}$ defined as $X \mapsto X^i$, and denote the Galois group of automorphisms as $\mathrm{Aut}(\mathbf{R}) := \{\sigma_i : i \in \mathbb{Z}_{2d}^\times\}$. For a subgroup H of $\mathrm{Aut}(\mathbf{R})$, we denote $\mathbf{R}_q^H := \{x \in \mathbf{R}_q : \forall \sigma \in H, \sigma(x) = x\}$, i.e. the ring of elements of $\mathbf{R}_q$ fixed by all automorphisms in H. Also, we define the $\mathbf{R}_q$-trace map $\mathrm{Tr}_H : \mathbf{R}_q \to \mathbf{R}_q^H$ as $\mathrm{Tr}_H(a) := \sum_{\sigma \in H} \sigma(a) \in \mathbf{R}_q^H$.

Norms. We define $r' = r \bmod^{\pm} q$ to be the unique element r' in the range $-\frac{q-1}{2} \leq r' \leq \frac{q-1}{2}$ such that $r' = r \bmod q$. We also denote $r' = r \bmod^{+} q$ to be the unique element r' in the range $0 \leq r' < q$ such that $r' = r \bmod q$. When the exact representation is not important, we simply write $r \bmod q$. For an element $w \in \mathbb{Z}_q$, we write $\|w\|_\infty$ to mean $|w \bmod^{\pm} q|$. Define the ℓ_∞ and ℓ_p norms for $w = w_0 + w_1 X + \ldots + w_{d-1} X^{d-1} \in \mathbf{R}$ as $\|w\|_\infty = \max_j \|w_j\|_\infty$, $\|w\|_p = \sqrt[p]{\|w_0\|_\infty^p + \ldots + \|w_{d-1}\|_\infty^p}$. If $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_m) \in \mathbf{R}^k$, then $\|\mathbf{w}\|_\infty = \max_j \|\mathbf{w}_j\|_\infty$, $\|\mathbf{w}\|_p = \sqrt[p]{\|\mathbf{w}_1\|^p + \ldots + \|\mathbf{w}_k\|^p}$. By default, $\|\mathbf{w}\| := \|\mathbf{w}\|_2$.

For $\beta \geq 1$, we define S_β to be the set of ring elements in $\mathbf{R}$ with coefficients between 0 and $\beta - 1$.

Short elements are invertible. We recall the main result by Lyubashevsky and Seiler [LS18] which says that short polynomials over $\mathbf{R}_q$ are invertible over $\mathbf{R}_q$.

Lemma 2 ([LS18]). *Let $q \equiv 5 \pmod 8$ be a prime. Then, any $f \in \mathbf{R}_q$ which satisfies $0 < \|f\|_\infty < \frac{1}{\sqrt{2}} q^{1/2}$ or $0 < \|f\| < q^{1/2}$ has an inverse in $\mathbf{R}_q$.*

The set of invertible elements of $\mathbf{R}_q$ is denoted by $\mathbf{R}_q^\times$.

Module-SIS Assumption. We recall the Module-SIS hardness assumption [LS15] adapted to the infinity norm setting.

Definition 1 (Module-SIS). *Let $q = q(\lambda), n = n(\lambda), m = m(\lambda), d = d(\lambda)$ and $\beta = \beta(\lambda)$. We say that the $\mathsf{MSIS}_{q,d,n,m,\beta}$ assumption holds if for any PPT algorithm $\mathcal{A}$, the following holds:*

$$\mathbf{Pr}\left[\mathbf{Az} = \mathbf{0} \wedge 0 < \|\mathbf{z}\|_\infty \leq \beta \mid A \leftarrow \mathbf{R}_q^{n \times m}, \mathbf{z} \leftarrow \mathcal{A}(\mathbf{A})\right] = \mathsf{negl}(\lambda)$$

2.2 Multilinear Extensions over Rings

Definition 2 (Multilinear extension over rings). *Let R be an arbitrary ring with zero 0 and identity 1. Given a function $f : \{0,1\}^\mu \to R$, we define the multilinear extension $\mathsf{mle}[f] \in R^{\leq 1}[X_1, \ldots, X_\mu]$ of f as*

$$\mathsf{mle}[f](\mathbf{x}) := \sum_{i \in \{0,1\}^\mu} f(i) \cdot eq(i, \mathbf{x})$$

where $eq(i, \mathbf{x}) := \prod_{j=1}^{\mu} \big((1 - i_j) \cdot (1 - x_j) + i_j x_j\big)$.

By definition, $\mathsf{mle}[f](\mathbf{x}) = f(\mathbf{x})$ for all $\mathbf{x} \in \{0,1\}^\mu$.

For a vector $\mathbf{f} := (f_i)_{i \in \{0,1\}^\mu} \in R^{2^\mu}$, we denote $\tilde{\mathbf{f}}$ as the multilinear extension of the function $f : \{0,1\}^\mu \to R$ defined by $i \mapsto f_i$.

2.3 Coordinate-Wise Special Soundness

Let $\mathsf{R} \subseteq \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$ be a ternary relation. For a triple $(\mathsf{pp}, \mathsf{x}, \mathsf{w}) \in \mathsf{R}$, we call pp the public parameters, x is a statement and w is a witness for x w.r.t. pp [5]. We denote $\mathsf{R}(\mathsf{pp}, \mathsf{x}) = \{\mathsf{w} : \mathsf{R}(\mathsf{pp}, \mathsf{x}, \mathsf{w}) = 1\}$.

An interactive proof system $\Pi = (\mathcal{S}, \mathcal{P}, \mathcal{V})$ for relation R consists of three PPT algorithms: the setup algorithm $\mathcal{S}$, prover $\mathcal{P}$, and verifier $\mathcal{V}$. The latter two are interactive and stateful. Further, we say a protocol is *public coin* if the verifier's challenges are chosen uniformly at random independently of the prover's messages.

We recall the notion of coordinate-wise special soundness from [FMN24]. Let S be a set and $\ell, k \in \mathbb{N}$. Namely, take two vectors $\mathbf{x} \coloneqq (x_1, \ldots, x_\ell), \mathbf{y} \coloneqq (y_1, \ldots, y_\ell) \in S^\ell$. Then, we define the following relation "$\equiv_i$" for fixed $i \in [\ell]$ as:

$$\mathbf{x} \equiv_i \mathbf{y} \iff x_i \neq y_i \wedge \forall j \in [\ell] \backslash \{i\}, x_j = y_j \ .$$

That is, vectors $\mathbf{x}$ and $\mathbf{y}$ have the same values in all coordinates apart from the i-th one. For $\ell = 1$, the relations boil down to checking whether two elements are distinct. Further, we can define the set

$$\mathrm{SS}(S, \ell, k) \coloneqq \left\{ \{\mathbf{x}_1, \ldots, \mathbf{x}_K\} \subseteq (S^\ell)^K : \begin{array}{l} \exists e \in [K], \forall i \in [\ell], \\ \exists J = \{j_1, \ldots, j_{k-1}\} \subseteq [K] \setminus \{e\}, \\ \forall j \in J, \mathbf{x}_e \equiv_i \mathbf{x}_j \end{array} \right\} \ ,$$

where $K \coloneqq \ell(k-1)+1$. In other words, there is a *central* vector $\mathbf{x}_e \in X$ such that for each coordinate of $\mathbf{x}_e$, there are $k-1$ other vectors in X that differ from $\mathbf{x}_e$ only in that coordinate. In other words, for each coordinate, there are k vectors in X that differ from each other only in that coordinate, and $\mathbf{x}_e$ is always one of them.

We are ready to define the notion of coordinate-wise special soundness.

Definition 3 (CWSS for Multi-Round Protocols). *Let Π be public-coin $(2\mu+1)$-round interactive proof system for relation R, where in the $2i$-th round, for $i \in [\mu]$, the verifier picks a uniformly random challenge from $S_i^{\ell_i}$. A tree of transcripts is a set of $K \coloneqq \prod_{i=1}^{\mu}(\ell_i(k_i-1)+1)$ arranged in the following tree structure. The nodes in the tree correspond to the prover's messages and the edges correspond to the verifier's challenges. Each node at depth i has exactly $\ell_i(k_i-1)+1$ children corresponding to $\ell_i(k_i-1)+1$ distinct challenges which, as a set of vectors, lie in $\mathrm{SS}(S_i, \ell_i, k_i)$. Every transcript corresponds to exactly one path from the root to a leaf node.*

We say that Π is $(\ell_1, \ldots, \ell_\mu)$-coordinate-wise $(k_1, \ldots, k_\mu)$-special sound for the relation R if there is a polynomial time algorithm that given public parameters pp, statement x and the tree of transcripts, outputs a witness $\mathsf{w} \in R(\mathsf{pp}, \mathsf{x})$. If $\ell_1 = \ldots = \ell_\mu = 1$ then we say the protocol is $(k_1, \ldots, k_\mu)$-special sound.

We recall the main result of [FMN24, Section 7] which translates CWSS to knowledge soundness.

[5] We omit pp and simply write $\mathsf{R}(\mathsf{x}, \mathsf{w})$ if the public parameters are not relevant to the relation.

Lemma 3. *Let Π be a $(\ell_1, \ldots, \ell_\mu)$-coordinate-wise $(k_1, \ldots, k_\mu)$-special sound interactive proof system as in Definition 3 and assume $K := \prod_{i=1}^{\mu}(\ell_i(k_i - 1) + 1) = \mathrm{poly}(\lambda)$. Then, it is knowledge sound with knowledge error*

$$\sum_{i=1}^{\mu} \frac{\ell_i \cdot k_i}{|S_i|^{\ell_i}}.$$

A similar result also holds in the random oracle model if Π is turned non-interactive via Fiat-Shamir transformation (see [FMN24, Section 8]). In this work, we will aim to show that our protocol satisfies coordinate-wise special soundness.

3 Evaluation Claims over Extension Fields as Relations over $\mathbf{R}_q$

In this section, we provide a generic transformation from proving ℓ-variate polynomial evaluations over finite fields $\mathbb{F}_{q^k}$ to proving $(\ell - \alpha + \kappa)$-variate polynomial evaluations over the cyclotomic ring $\mathbf{R}_q$. Next, we improve the result if the committed polynomial is only over the base field $\mathbb{F}_q$, instead of $\mathbb{F}_{q^k}$. At the core of our transformation is the process of embedding vectors of extension fields $\mathbb{F}_{q^k}$ inside ring elements $\mathbf{R}_q$ and translating inner products over $\mathbb{F}_{q^k}$ as "inner-product-related" statements over $\mathbf{R}_q$. In Figure 2 we provide a summary table of how much the $\mathbb{F}_{q^k}$-to-$\mathbf{R}_q$ transformations presented in this section cost in terms of prover communication.

3.1 Generic Transformation over $\mathbb{F}_{q^k}$

Finding subfields in $\mathbf{R}_q$. Let H be a subgroup of $\mathrm{Aut}(\mathbf{R}_q)$. We introduce a lemma that states that subring $\mathbf{R}_q^H$ of $\mathbf{R}_q$ is actually a field as long as σ_{-1} is contained in H.

Lemma 4 (Subfields of $\mathbf{R}_q$). *Let $q = 5 \pmod 8$ and $k \geq 1$ be a divisor of $d/2$. Then, $\mathbf{R}_q^{\langle -1, 4k+1 \rangle}$ is a subfield of $\mathbf{R}_q$ isomorphic to $\mathbb{F}_{q^k}$.*

Proof. By definition, $\mathbf{R}_q^{\langle -1, 4k+1 \rangle}$ is a subset of the set of all ring elements in $\mathbf{R}_q$ stable under the σ_{-1} automorphism, which itself is a field [LNP22, Lemma 2.6]. Using the homomorphic properties of Galois automorphisms, the set $\mathbf{R}_q^{\langle -1, 4k+1 \rangle}$ is closed under both addition and multiplication. Finally, the size of $\mathbf{R}_q^{\langle -1, 4k+1 \rangle}$ is q^k since each element of $\mathbf{R}_q^{\langle -1, 4k+1 \rangle}$ is of the form

$$a := a_0 + \sum_{j=1}^{k-1} a_{k-j} \cdot \left(X^{\frac{d}{2k} \cdot (k-j)} - X^{\frac{d}{2k} \cdot (k+j)} \right)$$

with k degrees of freedom: $a_0, a_1, \ldots, a_{k-1} \in \mathbb{Z}_q$. □

Inner products. Our main theorem of this section generalizes [LNP22], and reduces proving inner products over the subfield into proving statements over $\mathbf{R}_q$ related to the trace map. Informally, it introduces a bijection ψ which allows packing d/k elements of field $\mathbb{F}_{q^k}$ into one single $\mathbf{R}_q$ element. The fact that it is an invertible function is crucial in arguing knowledge soundness.

Theorem 2 (Inner product as output of the trace). *Let k divide $d/2$ and $H := \langle \sigma_{-1}, \sigma_{4k+1} \rangle \subseteq \mathbb{Z}_{2d}^{\times}$. Consider a map*

$$\begin{aligned} &\psi : (\mathbf{R}_q^H)^{d/k} \to \mathbf{R}_q, \\ &(a_0, a_1, \ldots, a_{d/k-1}) \mapsto \left(\sum_{i=0}^{d/2k-1} a_i \cdot X^i \right) + X^{d/2} \cdot \left(\sum_{i=0}^{d/2k-1} a_{d/2k+i} \cdot X^i \right). \end{aligned} \tag{7}$$

Then, ψ is a bijection. Moreover, for any $\mathbf{a}, \mathbf{b} \in (\mathbf{R}_q^H)^{d/k}$,

$$\mathrm{Tr}_H(\psi(\mathbf{a}) \cdot \sigma_{-1}(\psi(\mathbf{b}))) = \frac{d}{k} \cdot \langle \mathbf{a}, \mathbf{b} \rangle.$$

Proof. To prove the first part, we only need to show that ψ is injective since the domain and co-domain size are the same. Indeed, suppose that for some $\mathbf{a} = (a_0, \ldots, a_{d/k-1})$ we have $\psi(\mathbf{a}) = 0$ (the statement holds by the linearly homomorphic property of ψ). We prove that $\mathbf{a}$ is the zero-vector.

First, we note that $\psi(\mathbf{a})$ can be expanded as

$$\begin{aligned} \sum_{i=0}^{\frac{d}{2k}-1} X^i a_{i,0} + X^{\frac{d}{2}+i} a_{d/2k+i,0} + \sum_{j=1}^{k-1} a_{i,k-j} \cdot (X^{\frac{d}{2k}\cdot(k-j)+i} - X^{\frac{d}{2k}\cdot(k+j)+i}) \\ + \sum_{j=1}^{k-1} a_{d/2k+i,k-j} \cdot (X^{\frac{d}{2k}\cdot(2k-j)+i} - X^{\frac{d}{2k}\cdot(2k+j)+i}). \end{aligned}$$

Then, the u-th coefficient of the polynomial above, where $0 \le u < d/(2k)$, is exactly $a_{u,0}$. Similarly, the $(d/2k+u)$-th coefficient is $a_{d/2k+u,0}$. Hence, $a_{u,0} = a_{d/2k+u,0} = 0$.

Now, we claim that for any $1 \le u < k$ and $0 \le v < \frac{d}{2k}$, $a_{v,u} = a_{d/2k+v,u} = 0$. Indeed, first consider the $(\frac{d}{2k} \cdot u + v)$-th coefficient of the polynomial above. It is equal to $a_{v,u} + a_{d/2k+v,k-u} = 0$. On the other hand, if we focus on the $(\frac{d}{2k} \cdot (k+u) + v)$-th coefficient, then it is exactly equal to $-a_{v,k-u} + a_{d/2k+v,u} = 0$. By combining the two observations we obtain

$$a_{d/2k+v,u} = a_{v,k-u} = -a_{d/2k+v,u}$$

which implies $a_{d/2k+v,u} = a_{v,u} = 0$, and thus $\mathbf{a} = \mathbf{0}$.

We move on to the second part. We will make use of the following three claims.

Claim. Consider the multiplicative group $\mathbb{Z}_{2d}^{\times}$ and let $k \le d/2$ be a power-of-two. Then,

$$\langle 4k+1 \rangle = \{(4k) \cdot \alpha + 1 : 0 \le \alpha \le d/(2k) - 1\}.$$

Proof. First, using the property that $4k$ divides $2d$, we have for any i,

$$((4k+1)^i \bmod 2d) \bmod 4k = (4k+1)^i \bmod 4k = \sum_{j=0}^{i} \binom{i}{j} (4k)^j \bmod 4k = 1.$$

Therefore, $\langle 4k+1 \rangle \subseteq \{(4k)\cdot\alpha+1 : 0 \le \alpha \le d/(2k)-1\}$. Now, the statement holds since by [LS18, Lemma 2.4], we know that the order of $4k+1 \in \mathbb{Z}_{2d}^{\times}$ is exactly $\frac{2d}{4k} = d/(2k)$. □

Claim. Let $i \in \mathbb{Z}$ be not divisible by $d/2k$. Then, $\mathrm{Tr}_H(X^i) = 0$.

Proof. Indeed, by assumption $4ki$ cannot be divisible $2d$ and thus $X^{4ki} \neq 1$. Now, using the first claim, we get

$$\begin{aligned}
\mathrm{Tr}_H(X^i) &= \sum_{h \in \langle -1, 4k+1 \rangle} X^{h\cdot i} \\
&= \sum_{h \in \langle 4k+1 \rangle} X^{h\cdot i} + X^{-h\cdot i} \\
&= \sum_{\alpha=0}^{d/(2k)-1} X^{(4k\cdot\alpha+1)\cdot i} + \sum_{\alpha=0}^{d/(2k)-1} X^{-(4k\cdot\alpha+1)\cdot i} \\
&= X^i \cdot \frac{(X^{4ki})^{d/(2k)} - 1}{X^{4ki} - 1} + X^{-i} \cdot \frac{(X^{-4ki})^{d/(2k)} - 1}{X^{-4ki} - 1} \\
&= 0.
\end{aligned}$$

□

Claim. $\mathrm{Tr}_H(X^{d/2}) = 0$.

Proof. Using the first claim we obtain

$$\begin{aligned}
\mathrm{Tr}_H(X^{d/2}) &= \sum_{\alpha=0}^{d/(2k)-1} X^{(4k\cdot\alpha+1)\cdot d/2} + \sum_{\alpha=0}^{d/(2k)-1} X^{-(4k\cdot\alpha+1)\cdot d/2} \\
&= \sum_{\alpha=0}^{d/(2k)-1} X^{d/2} + \sum_{\alpha=0}^{d/(2k)-1} X^{-d/2} \\
&= \sum_{\alpha=0}^{d/(2k)-1} X^{d/2} - X^{d/2} = 0.
\end{aligned}$$

□

To conclude the proof of the theorem, we notice that

$$\begin{aligned}
\psi(\mathbf{a}) \cdot \sigma_{-1}(\psi(\mathbf{b})) = &\left(\sum_{i=0}^{d/2k-1} \sum_{j=0}^{d/2k-1} a_i b_j \cdot X^{i-j} + \sum_{i=0}^{d/2k-1} \sum_{j=0}^{d/2k-1} a_{d/2k+i} b_{d/2k+j} \cdot X^{i-j} \right) \\
&+ X^{d/2} \cdot \left(\sum_{i=0}^{d/2k-1} \sum_{j=0}^{d/2k-1} a_{d/2k+i} b_j \cdot X^{i-j} - \sum_{i=0}^{d/2k-1} \sum_{j=0}^{d/2k-1} a_i b_{d/2k+j} \cdot X^{i-j} \right).
\end{aligned}$$

Hence,

$$\begin{aligned}
&\mathrm{Tr}_H(\psi(\mathbf{a}) \cdot \sigma_{-1}(\psi(\mathbf{b}))) \\
&= \left(\sum_{i=0}^{d/2k-1} \sum_{j=0}^{d/2k-1} a_i b_j \cdot \mathrm{Tr}_H(X^{i-j}) + \sum_{i=0}^{d/2k-1} \sum_{j=0}^{d/2k-1} a_{d/2k+i} b_{d/2k+j} \cdot \mathrm{Tr}_H(X^{i-j}) \right) \\
&+ \left(\sum_{i=0}^{d/2k-1} \sum_{j=0}^{d/2k-1} a_{d/2k+i} b_j \cdot \mathrm{Tr}_H(X^{d/2+(i-j)}) - \sum_{i=0}^{d/2k-1} \sum_{j=0}^{d/2k-1} a_i b_{d/2k+j} \cdot \mathrm{Tr}_H(X^{d/2+(i-j)}) \right).
\end{aligned}$$

Now, using the fact that for two distinct $0 \leq i, j < d/2k$, both $i - j$ and $d/2 + (i - j)$ cannot be divisible by $d/2k$, and the second claim, we deduce that $\mathrm{Tr}_H(X^{i-j}) = \mathrm{Tr}_H(X^{d/2+i-j}) = 0$. However, when $i = j$, then $\mathrm{Tr}_H(X^{i-j}) = \frac{d}{k}$ and $\mathrm{Tr}_H(X^{d/2+(i-j)}) = 0$ by the third claim. Hence, we deduce that

$$\mathrm{Tr}_H(\psi(\mathbf{a}) \cdot \sigma_{-1}(\psi(\mathbf{b}))) = \frac{d}{k} \sum_{i=0}^{d/k-1} a_i b_i = \frac{d}{k} \langle \mathbf{a}, \mathbf{b} \rangle$$

which concludes the proof. □

Reducing to multilinear evaluation over $\mathbf{R}_q$. We use the results above to translate the multilinear evaluation claim over $\mathbf{R}_q^H$

$$f(x_1, \ldots, x_\ell) = y$$

to an equivalent relation over the ring $\mathbf{R}_q$. Denote the coefficients of polynomial f as $(f_\iota)_{\iota \in \{0,1\}^\ell} \in \mathbf{R}_q^H$. We can explicitly rewrite the evaluation equation as

$$y = \sum_{i \in \{0,1\}^{\ell-\alpha+\kappa}} \left(x_1^{i_1} \cdot \ldots \cdot x_{\ell-\alpha+\kappa}^{i_{\ell-\alpha+\kappa}} \right) \cdot \left(\sum_{j \in \{0,1\}^{\alpha-\kappa}} f_{i\|j} \cdot x_{\ell-\alpha+\kappa+1}^{j_1} \cdot \ldots \cdot x_\ell^{j_{\alpha-\kappa}} \right). \tag{8}$$

For each $i \in \{0,1\}^{\ell-\alpha+\kappa}$ we define

$$F_i := \psi\left((f_{i||j})_{j \in \{0,1\}^{\alpha-\kappa}} \right) \quad \text{and also} \quad v := \psi\left((x_{\ell-\alpha+\kappa+1}^{j_1} \cdot \ldots \cdot x_\ell^{j_{\alpha-\kappa}})_{j \in \{0,1\}^{\alpha-\kappa}} \right).$$

Now, recall that for every $x \in \mathbf{R}_q^H$ we have $\mathrm{Tr}_H(x) = x$ and trace is additively homomorphic. Hence, by applying Theorem 2 to Equation (8) we obtain

$$\frac{d}{k} \cdot y = \mathrm{Tr}_H\left(\underbrace{\sum_{i \in \{0,1\}^{\ell-\alpha+\kappa}} \left(x_1^{i_1} \cdot \ldots \cdot x_{\ell-\alpha}^{i_{\ell-\alpha+\kappa}} \right) \cdot F_i}_{Y :=} \cdot \sigma_{-1}(v) \right).$$

Therefore, the prover will send a single ring element $Y \in \mathbf{R}_q$, which allows the verifier to check if $\mathrm{Tr}_H(Y \cdot \sigma_{-1}(v)) = dy/k$. Then, we only need to prove that Y is well-formed, which is an evaluation proof for the $(\ell - \alpha + \kappa)$-variate multilinear polynomial $F := (F_i)_i$ over $\mathbf{R}_q$.

3.2 Dealing with $f \in \mathbb{Z}_q^{\leq 1}[X_1, \ldots, X_\ell]$

We now optimize the special case where the evaluation point $(x_1, \ldots, x_\ell)$ is defined in the extension field $\mathbb{F}_{q^k}$, while the coefficients of the committed polynomial f are over $\mathbb{Z}_q$. Note that if we naively apply the generic transformation from Section 3.1 then we would end up with an evaluation proof for the $(\ell - \alpha + \kappa)$-variate multilinear polynomial over $\mathbf{R}_q$. This subsection focuses on how to lower it further to $\ell - \alpha$ variables.

We start with a similar observation as in Equation (8) that a polynomial evaluation claim can be written as

$$y = \sum_{i \in \{0,1\}^\kappa} \left(x_1^{i_1} \cdot \ldots \cdot x_\kappa^{i_\kappa} \right) \cdot \left(\underbrace{\sum_{j \in \{0,1\}^{\ell - \kappa}} f_{i \| j} \cdot x_{\kappa+1}^{j_1} \cdot \ldots \cdot x_\ell^{j_{\ell-\kappa}}}_{y_i :=} \right). \tag{9}$$

Hence, we let the prover send $k - 1$ partial evaluations $y_i \in \mathbb{F}_{q^k}$, for $i \neq \mathbf{0}$, to the verifier who can directly compute:

$$y_{0 \ldots 0} := y - \sum_{i \neq \mathbf{0}} \left(x_1^{i_1} \cdot \ldots \cdot x_\kappa^{i_\kappa} \right) \cdot yi.$$

What we have left to prove is that all the y_i are well-formed, or in other words $f_i(x_{\kappa+1}, \ldots, x_\ell) = y_i$ for $i \in \{0,1\}^\kappa$ where $f_i \in \mathbb{Z}_q[X_{\kappa+1}, \ldots, X_\ell]$.

For the next part, we need a somewhat explicit definition of the extension field (it could as well be $\mathbf{R}_q^H$). Let $\mathbb{F}_{q^k} := \mathbb{F}_q[Z]/\varphi(Z)$, where φ is an irreducible polynomial in $\mathbb{F}_q[Z]$ of degree k. Then, we define the $(\ell - \kappa)$-variate multilinear polynomial $f' \in \mathbb{F}_{q^k}[X_{\kappa+1}, \ldots, X_\ell]$ as

$$f'(X_{\kappa+1}, \ldots, X_\ell) := \sum_{i \in \{0,1\}^\kappa} f_i(X_{\kappa+1}, \ldots, X_\ell) \cdot Z^{\sum_{t=1}^{\kappa} i_t \cdot 2^{t-1}}.$$

Then, by definition of the partial evaluations y_i we obtain

$$f'(x_{\kappa+1}, \ldots, x_\ell) = \sum_{i \in \{0,1\}^\kappa} y_i \cdot Z^{\sum_{t=1}^{\kappa} i_t \cdot 2^{t-1}} \in \mathbb{F}_{q^k}$$

which is now a full-fledged polynomial evaluation over $\mathbb{F}_{q^k}$. By applying the generic transformation from Section 3.1, we can reduce the statement into an evaluation claim over $\mathbf{R}_q$ for a μ-variate multilinear polynomial, where

$$\mu = \underbrace{\underbrace{(\ell - \kappa)}_{\text{Section 3.2}} - \alpha + \kappa}_{\text{Section 3.1}} = \ell - \alpha.$$

Committed polynomial	# variables after the $\mathbf{R}_q$-transformation	Prover cost
$f \in \mathbb{F}_{q^k}[X_1, \ldots, X_\ell]$	$\ell - \alpha + \kappa$	$d \log q$
$f \in \mathbb{F}_q[X_1, \ldots, X_\ell]$	$\ell - \alpha$	$(d + k(k-1)) \log q$

Fig. 2. Summary of our transformation in Sections 3.1 and 3.2. The prover cost is measured in the size of the prover messages in bits.

4 Multilinear Polynomial Commitment Scheme over $\mathbf{R}_q$

This section focuses on proving knowledge of a *committed* multilinear polynomial $f \in \mathbf{R}_q^{\leq 1}[X_1, \ldots, X_\ell]$ with coefficients $(f_i)_{i \in \{0,1\}^\ell}$ such that

$$f(x_1, \ldots, x_\ell) := \sum_{i \in \{0,1\}^\ell} f_i \cdot x_1^{i_1} \cdot \ldots \cdot x_\ell^{i_\ell} = u$$

where the evaluation point $(x_1, \ldots, x_\ell) \in \mathbf{R}_q^\ell$ and the image $u \in \mathbf{R}_q$ are public.

We follow the initial strategy from Greyhound [NS24]. Suppose $\ell = m + r$ for some $m, r \geq 1$ (keep in mind $m \approx r$). Then, we can rewrite the equation above as

$$f(x_1, \ldots, x_\mu) := \sum_{i \in \{0,1\}^r} \sum_{j \in \{0,1\}^m} f_{i||j} \cdot (x_1^{i_1} \cdot \ldots \cdot x_r^{i_r}) \cdot (x_{r+1}^{j_1} \cdot \ldots \cdot x_\ell^{j_m}),$$

or in other words

$$f(x_1, \ldots, x_\mu) = \mathbf{b}^\intercal \begin{bmatrix} \mathbf{a}^\intercal & & \\ & \ddots & \\ & & \mathbf{a}^\intercal \end{bmatrix} \begin{bmatrix} \mathbf{f}_{0\ldots0} \\ \vdots \\ \mathbf{f}_{1\ldots1} \end{bmatrix} \quad \begin{aligned} \mathbf{b}^\intercal &:= (x_1^{i_1} \cdot \ldots \cdot x_r^{i_r})_{i \in \{0,1\}^r} \in \mathbf{R}_q^{2^r} \\ \text{for } \mathbf{a}^\intercal &:= (x_{r+1}^{j_1} \cdot \ldots \cdot x_\ell^{j_m})_{j \in \{0,1\}^m} \in \mathbf{R}_q^{2^m} \\ \mathbf{f}_i^\intercal &:= (f_{i||j})_{j \in \{0,1\}^m} \in \mathbf{R}_q^{2^m} \text{ for } i \in \{0,1\}^r \end{aligned} . \tag{10}$$

This representation of a polynomial evaluation claim gives intuition how to commit to the polynomial f.

4.1 Inner and Outer Commitment

The starting point of our polynomial commitment scheme construction is the basic commitment scheme from LaBRADOR [BS23]. Let $b > 1$ be the base used for decomposition and denote $\delta := \lceil \log_b q \rceil$. The commitment key is a pair of uniformly random matrices $\mathbf{A} \in \mathbf{R}_q^{n_A \times \delta 2^m}$ and $\mathbf{B} \in \mathbf{R}_q^{n_B \times n_A \delta 2^r}$ where n_A, n_B are the security parameters which determine hardness of the underlying Module-SIS problem.

Suppose we want to commit to the 2^r vectors $(\mathbf{f}_i^\intercal)_i$ of length 2^m. The first step is to compute the decompositions:

$$\mathbf{s}_{1+\sum_{z=1}^r i_z \cdot 2^{z-1}} := \mathbf{G}_{2^m}^{-1}(\mathbf{f}_i) \tag{11}$$

and the inner Ajtai commitments $\mathbf{t}_i := \mathbf{A}\mathbf{s}_i \in \mathbf{R}_q^{n_A}$ for $i \in [2^r]$. The next step is to compute their binary decomposition $\hat{\mathbf{t}}_i := \mathbf{G}_{n_A}^{-1}(\mathbf{t}_i)$. The final outer commitment is

$$\mathbf{u} := \mathbf{B} \begin{bmatrix} \hat{\mathbf{t}}_1 \\ \vdots \\ \hat{\mathbf{t}}_{2^r} \end{bmatrix} \in \mathbf{R}_q^{n_B}. \tag{12}$$

The commitment opening becomes a tuple of $(\mathbf{s}_i, \hat{\mathbf{t}}_i)_i$.

In the knowledge extraction, we will only be able to extract so-called weak openings. We define a weak opening for the commitment $\mathbf{u}$ is a tuple $(\mathbf{s}_i, \hat{\mathbf{t}}_i, c_i)_{i\in[2^r]}$, which satisfies all the following conditions

$$\forall i \in [2^r]: \quad \|c_i \cdot \mathbf{s}_i\| \leq \bar{\beta}, \quad \|c_i\|_1 \leq \bar{\omega}, \quad c_i \in \mathbf{R}_q^{\times}, \quad \mathbf{A}\mathbf{s}_i = \mathbf{G}_{n_B}\hat{\mathbf{t}}_i$$
$$\mathbf{B}\begin{bmatrix}\hat{\mathbf{t}}_1\\ \vdots \\ \hat{\mathbf{t}}_r\end{bmatrix} = \mathbf{u} \quad \text{and} \quad \left\|\begin{bmatrix}\hat{\mathbf{t}}_1\\ \vdots \\ \hat{\mathbf{t}}_{2^r}\end{bmatrix}\right\|_{\infty} \leq \bar{\gamma}.$$

Clearly, a message can be derived from the weak opening by Equation (11). We recall the known result that the commitment scheme above satisfies binding w.r.t. weak openings under the Module-SIS assumption. We omit the proof since it follows identically as in [NS24, Lemma 2.11].

Lemma 5 (Weak Binding). *There is a deterministic algorithm that given two weak openings $(\mathbf{s}_i, \hat{\mathbf{t}}_i, c_i)_{i\in[2^r]}$ and $(\mathbf{s}'_i, \hat{\mathbf{t}}'_i, c'_i)_{i\in[2^r]}$ for the commitment $\mathbf{u} \in \mathbf{R}_q^{n_B}$ such that $\mathbf{s}_j \neq \mathbf{s}'_j$ for some $j \in [2^r]$, outputs a vector $\mathbf{z} \in \mathbf{R}_q^{2^m+n_A\delta 2^r}$ such that $[\mathbf{A} \mid \mathbf{B}]\mathbf{z} = \mathbf{0}$ and $0 < \|\mathbf{z}\|_\infty \leq \max(2\bar{\omega}\bar{\beta}, 2\bar{\gamma})$.*

4.2 Polynomial Evaluation as Quadratic Equation

We can now rephrase Equation (10) in terms of the commitment opening $(\mathbf{s}_i)_i$. Namely, we can write

$$u = f(x_1, \ldots, x_\ell) = \mathbf{b}^\intercal \begin{bmatrix}\mathbf{a}^\intercal\mathbf{G}_{2^m} & & \\ & \ddots & \\ & & \mathbf{a}^\intercal\mathbf{G}_{2^m}\end{bmatrix}\begin{bmatrix}\mathbf{s}_1\\ \vdots \\ \mathbf{s}_{2^r}\end{bmatrix}. \tag{13}$$

From now on, we focus on proving knowledge of a commitment opening $(\mathbf{s}_i, \hat{\mathbf{t}}_i)_i$ for the commitment $\vec{u}$ which satisfies the above equation. We summarize the protocol in Figure 3.

To begin with, the prover commits to the vector $\mathbf{w} = (w_1, \ldots, w_{2^r}) \in \mathbf{R}_q^{2^r}$ defined as $w_i := \mathbf{a}^\intercal\mathbf{G}_{b_0,2^m}\mathbf{s}_i \in \mathbf{R}_q$. That is, $\mathcal{P}$ first decomposes each w_i w.r.t. base b_1, thus obtaining $\hat{\mathbf{w}}_i := \mathbf{G}^{-1}_{b_1,1}(w_i) \in \mathbf{R}_q^{\delta}$. Finally, the prover sends the commitment

$$\mathbf{v} := \mathbf{D}\hat{\mathbf{w}} \in \mathbf{R}_q^{n_D} \tag{14}$$

where $\mathbf{D} \in \mathbf{R}_q^{n_D\times\delta\cdot 2^r}$ is a public uniformly random matrix and $\hat{\mathbf{w}}^\intercal := (\hat{\mathbf{w}}_1^\intercal, \ldots, \hat{\mathbf{w}}_{2^r}^\intercal)$. Note that Equation (13) can be rephrased as

$$u = \mathbf{b}^\intercal\mathbf{w} = \mathbf{b}^\intercal\mathbf{G}_{2^r}\hat{\mathbf{w}}. \tag{15}$$

The verifier $\mathcal{V}$ responds with a challenge vector $\mathbf{c} := (c_1, \ldots, c_{2^r}) \leftarrow C^{2^r}$, where the challenge space C is currently defined as a subset of short challenges in $\mathbf{R}_q$:

$$C \subseteq \{c \in \mathbf{R}_q : \|c\|_1 \leq \omega\}$$

for some ω instantiated later. Then, the prover $\mathcal{P}$ computes the short folded witness

$$\mathbf{z} := c_1\mathbf{s}_1 + c_2\mathbf{s}_2 + \ldots + c_{2^r}\mathbf{s}_{2^r} \in \mathbf{R}_q^{2^m}.$$

We note the relationship between $\mathbf{z}$ and $\hat{\mathbf{w}}$

$$\mathbf{a}^\intercal \mathbf{G}_{2^m}\mathbf{z} = \sum_{i=1}^{2^r} c_i \mathbf{a}^\intercal \mathbf{G}_{2^m}\mathbf{s}_i = \sum_{i=1}^{2^r} c_i w_i = \sum_{i=1}^{2^r} c_i \mathbf{G}_1 \hat{w}_i = (\mathbf{c}^\intercal \otimes \mathbf{G}_{b_1,1})\hat{\mathbf{w}}. \tag{16}$$

We also note the relationship between $\mathbf{z}$ and $\hat{\mathbf{t}}$:

$$\mathbf{A}\mathbf{z} := \sum_{i=1}^{2^r} c_i \mathbf{A}\mathbf{s}_i = \sum_{i=1}^{2^r} c_i \mathbf{G}_{n_A}\hat{\mathbf{t}}_i = (\mathbf{c} \otimes \mathbf{G}_{n_A})\hat{\mathbf{t}}. \tag{17}$$

Now, combining Equations 12, 14, 15, 16, and 17, the prover $\mathcal{P}$ will prove that short $\mathbf{z}, \hat{\mathbf{t}}, \hat{\mathbf{w}}$ satisfy the linear relation over $\mathbf{R}_q$:

$$\begin{bmatrix} \mathbf{D} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{b}^\intercal \mathbf{G}_{2^r} & \mathbf{0} & \mathbf{0} \\ (\mathbf{c}^\intercal \otimes \mathbf{G}_1) & \mathbf{0} & -\mathbf{a}^\intercal \mathbf{G}_{2^m} \\ \mathbf{0} & \mathbf{c}^\intercal \otimes \mathbf{G}_{n_A} & -\mathbf{A} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{w}} \\ \hat{\mathbf{t}} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{u} \\ u \\ 0 \\ \mathbf{0} \end{bmatrix}.$$

We see that while coefficients of $\hat{\mathbf{w}}, \hat{\mathbf{t}}$ are between 0 and $b-1$, the coefficients of $\mathbf{z}$ will be moderately larger than b, and may as well be negative, depending how the challenge space C is chosen. Consequently, we further consider a decomposition $\hat{\mathbf{z}}$ of $\mathbf{z}$ similar to two's complement. Namely, let β be the infinity norm upper-bound on $\mathbf{z}$ (i.e. the prover aborts if the norm is larger than β). Let τ be the smallest integer such that $\beta < b^\tau$. Then, we can decompose an integer $z \in [-\beta, \beta]$ as

$$z := \sum_{i=0}^{\tau-1} \hat{z}_i \cdot b^i - \hat{z}_\tau \cdot b^\tau \quad \text{for} \quad \hat{z}_0, \ldots, \hat{z}_{\tau-1} \in \{0, 1, \ldots, b-1\} \text{ and } \hat{z}_\tau \in \{0, 1\}.$$

This decomposition immediately implies the existence of a (sparse) matrix $\mathbf{J}$ such that

$$\mathbf{z} = \mathbf{J} \underbrace{\begin{bmatrix} \hat{\mathbf{z}}_1 \\ \vdots \\ \hat{\mathbf{z}}_{k2^m\delta} \end{bmatrix}}_{\hat{\mathbf{z}}}, \quad \text{where} \quad \begin{matrix} \hat{\mathbf{z}}_{(\tau+1)2^m\delta} \in S_2 \\ \hat{\mathbf{z}}_i \in S_b \text{ for } i < \tau 2^m \delta \end{matrix}. \tag{18}$$

Hence, the prover now needs to prove knowledge of $(\hat{\mathbf{w}}, \hat{\mathbf{t}}, \hat{\mathbf{z}}) \in S_b^{((n_A+1)2^r+\tau 2^m)\delta} \times S_1^{2^m\delta}$ such that

$$\begin{bmatrix} \mathbf{D} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{b}^\intercal \mathbf{G}_{2^r} & \mathbf{0} & \mathbf{0} \\ (\mathbf{c}^\intercal \otimes \mathbf{G}_1) & \mathbf{0} & -\mathbf{a}^\intercal \mathbf{G}_{2^m}\mathbf{J} \\ \mathbf{0} & \mathbf{c}^\intercal \otimes \mathbf{G}_{n_A} & -\mathbf{A}\mathbf{J} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{w}} \\ \hat{\mathbf{t}} \\ \hat{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{u} \\ u \\ 0 \\ \mathbf{0} \end{bmatrix}. \tag{19}$$

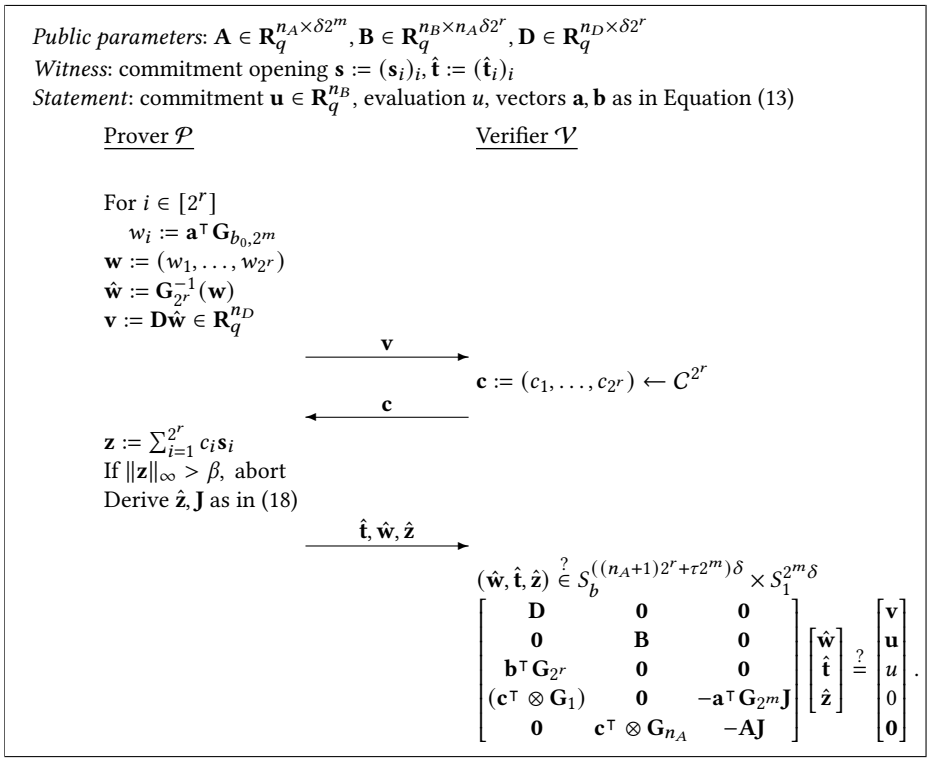

Fig. 3. Interactive proof of knowledge of the commitment opening $(\mathbf{s}_i, \hat{\mathbf{t}}_i)$ which satisfies (13). In our final scheme, the prover does not send the final message in the clear, but proves knowledge of valid $(\hat{\mathbf{w}}, \hat{\mathbf{t}}, \hat{\mathbf{z}})$.

The completeness follows directly from the protocol rationale described above. Hence, we move on the soundness guarantees. Identically as [NS24, Lemma 3.2] we show that given CWSS-structured transcripts, we can extract a valid weak opening which satisfies Equation (13). Due to space constraints, we provide a more formal statement in Lemma 6.

4.3 Unstructured Linear Relations and Sumcheck

In this section, we show how to reduce proving unstructured linear relations over $\mathbf{R}_q$ with infinity-norm constraints, as in Equation (19). Concretely, we focus of the following relation $\mathsf{R}^{\mathsf{lin}}$[6]:

$$\mathsf{R}^{\mathsf{lin}}_{q,d,n,\mu,\beta} := \left\{ (\mathbf{z} \in \mathbf{R}_q^{\mu}, (\mathbf{M} \in \mathbf{R}_q^{n \times \mu}, \mathbf{y} \in \mathbf{R}_q^{n})) : \mathbf{M}\mathbf{z} = \mathbf{y} \wedge \|\mathbf{z}\|_\infty \leq \beta \right\}.$$

[6] We note that it is almost identical to (19), apart from two differences. First, we consider coefficients between $\{0, \ldots, \beta\}$ (for $\beta := b - 1$) instead, and some coefficients of $\mathbf{z}$ are actually binary. Our methodology can be generalized to the setting of (19) as we point out later.

At a high level, our approach is to $\mathbf{z}$ using a (potentially another) polynomial commitment scheme, and then apply the sumcheck protocol to prove the relation. However, one challenge is that the relation $\mathsf{R}^{\mathsf{lin}}_{q,d,n,\mu,\beta}$ is defined over polynomial rings $\mathbf{R}_q$, so the standard sumcheck protocol over finite fields cannot be directly applied (or one would have to run it over $\mathbf{R}_q$ as in [BC24]). To bridge this gap, we leverage the ring-switching technique introduced by Huang, Mao, and Zhang [HMZ25].

Ring switching. In the relation $\mathsf{R}^{\mathsf{lin}}_{q,d,n,\mu,\beta}$, the input $\mathbf{M} \in \mathbf{R}_q^{n\times\mu}$ and $\mathbf{y} \in \mathbf{R}_q^n$ are public information, and the prover needs to prove knowledge of a witness $\mathbf{z} \in \mathbf{R}_q^\mu$ such that

- $\mathbf{Mz} = \mathbf{y}$
- $\|\mathbf{z}\|_\infty \le \beta$

For the first identity, the authors of [HMZ25] observe that $\mathbf{Mz} = \mathbf{y}$ if and only if there is a polynomial $\mathbf{r} \in (\mathbb{Z}_q^{<d}[X])^n$ such that

$$\mathbf{Mz} = \mathbf{y} + (X^d + 1) \cdot \mathbf{r}$$

We now naturally consider $\mathbf{M}$, $\mathbf{z}$, and $\mathbf{y}$ as matrices or vectors over $\mathbb{Z}_q^{<d}[X]$ (instead of over $\mathbf{R}_q$). Since we will apply the sumcheck protocol later, we further consider $\mathbf{M}, \mathbf{x}, \mathbf{y}, \mathbf{r}$ as elements over $\mathbb{F}_{q^k}^{<d}[X]$. Now, we can divide the process into two steps: the prover first commits to $(\mathbf{z}, \mathbf{r})$ using a polynomial commitment scheme Com, and then proves the following constraints:

- For each $i \in [n]$, $\sum_j \mathbf{M}_{i,j}\mathbf{z}_j = \mathbf{y}_i + (X^d + 1) \cdot \mathbf{r}_i$, where $\mathbf{M}_{i,j}, \mathbf{y}_i \in \mathbb{Z}_q^{<d}[X]$
- $\mathbf{z}$ is a vector over $\mathbb{Z}_q^{<d}[X]$ and $\|\mathbf{z}\|_\infty \le \beta$.

Notice that the verifier can check whether $\mathbf{M}$ and $\mathbf{y}$ are over $\mathbb{Z}_q^{<d}[X]$ by themselves, since $\mathbf{M}$ and $\mathbf{y}$ are public information. However, the prover has to prove that $\mathbf{z}$ is a vector over $\mathbb{Z}_q^{<d}[X]$ if we run the sumcheck protocol over $\mathbb{F}_{q^k}^{<d}[X]$. Since $\mathbf{M}_{i,j}, \mathbf{z}_j, \mathbf{y}_i, \mathbf{r}_i \in \mathbb{Z}_q^{<d}[X]$, we can represent them as

$$\mathbf{M}_{i,j} = \sum_{\ell<d} M_{i,j,\ell} \cdot X^\ell,$$

and similarly for all $\mathbf{z}_j, \mathbf{y}_i$, and $\mathbf{r}_i$. For each $\alpha \in \mathbb{F}_{q^k}$, we denote

$$\mathbf{M}_{i,j}(\alpha) = \sum_{\ell<d} M_{i,j,\ell} \cdot \alpha^\ell \in \mathbb{F}_{q^k},$$

and similarly for all $\mathbf{z}_j, \mathbf{y}_i$, and $\mathbf{r}_i$. Now, instead of proving the constraints over $\mathbb{F}_{q^k}^{<d}[X]$, which could be very costly, we optimize by letting the verifier first send a random α to the prover. Then, the prover proves the following constraints:

- For each $i \in [n]$, $\sum_j \mathbf{M}_{i,j}(\alpha)\mathbf{z}_j(\alpha) = \mathbf{y}_i(\alpha)+(\alpha^d+1)\cdot\mathbf{r}_i(\alpha)$, where $\mathbf{M}_{i,j}, \mathbf{y}_i \in \mathbb{Z}_q^{<d}[X]$
- $\mathbf{z}$ is a vector over $\mathbb{Z}_q^{<d}[X]$ and $\|\mathbf{z}\|_\infty \le \beta$.

Having proof composition in mind, we consider the following simple three-round protocol in Figure 4 which captures the idea above.

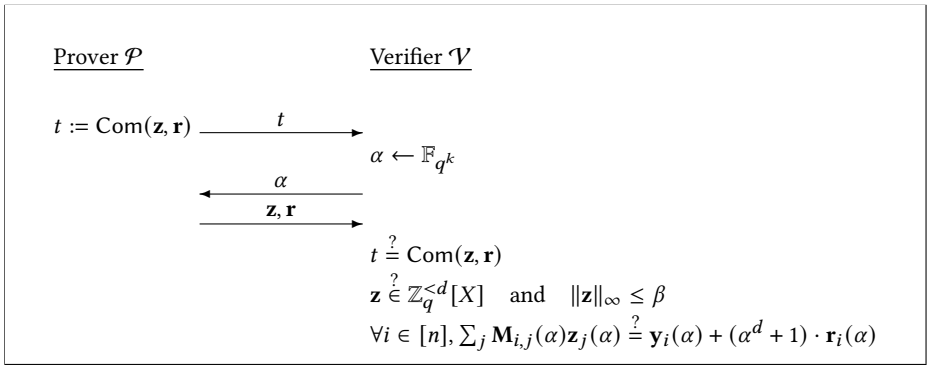

Fig. 4. Reducing proving linear equations over $\mathbb{Z}_q^{<d}q[X]$ to equations over $\mathbb{F}_{q^k}$.

We note that this simplification could introduce a $(2d-1)/|\mathbb{F}_{q^k}|$ soundness error. This is because the polynomial $\sum_j \mathbf{M}_{i,j}\mathbf{z}_j - \mathbf{y}_i - (X^d+1)\cdot\mathbf{r}_i$ has degree at most $2d-1$. Formally, we argue this by proving $2d$-special soundness in Lemma 7, which we do not include here due to space constraints.

From now on, we focus on proving knowledge of the prover's last message which satisfies the verification conditions.

Remark 1. Looking ahead, we will instantiate Com with the inner-outer commitment from Section 4. In this case, for special soundness we will consider breaking the weak binding property (cf. Lemma 5), rather than binding.

Represent the constraints by polynomials. In our sumcheck protocol, we first commit $(\mathbf{z}, \mathbf{r})$ as a multilinear polynomial over $\mathbb{F}_{q^k}$. Let $\widetilde{w}$ be a multilinear polynomial such that

$$\widetilde{w}(u,\ell) = \begin{cases} z_{u,\ell}, & \text{if } u \le \mu \\ r_{u-\mu,\ell}, & \text{if } \mu < u \le \mu + n \end{cases} \tag{20}$$

Here, we slightly abuse the notation; that is, u and ℓ actually refer to the binary representations of numbers in $[\mu+n]$ and $[d]$, respectively. Now we can assume that α is fixed. We define multilinear polynomials $\widetilde{\alpha}$ and $\widetilde{M}_\alpha$ such that:

$$\forall i,u,\ell, \quad \widetilde{\alpha}(\ell) = \alpha^\ell \quad \text{and} \quad \widetilde{M}_\alpha(i,u) = \begin{cases} \mathbf{M}_{i,u}(\alpha), & \text{if } u \le \mu \text{ and } i \le n \\ -(\alpha^d+1), & \text{if } i+\mu = u \\ 0, & \text{otherwise} \end{cases}$$

Now, for each fixed α, we can represent the constraints as polynomials:

– For each $i \in [n]$,

$$\sum_{u=1}^{\mu+n} \widetilde{M}_\alpha(i,u) \sum_{\ell} \widetilde{w}(u,\ell) \cdot \widetilde{\alpha}(\ell) = \mathbf{y}_i(\alpha),$$

where $\mathbf{M}_{i,j}, \mathbf{y}_i \in \mathbb{Z}_q^{<d}[X]$.

– For each $u \in [\mu]$ and $\ell \in [d]$,

$$\widetilde{w}(u,\ell) \cdot (\widetilde{w}(u,\ell)-1) \cdot (\widetilde{w}(u,\ell)+1) \cdots (\widetilde{w}(u,\ell)-\beta) \cdot (\widetilde{w}(u,\ell)+\beta) = 0.^{7}$$

Following standard approaches [Set20; HMZ25], we use an equality function to batch these constraints. Since $i \in [n]$, its binary representation is a string in $\{0,1\}^{\log n}$. Recall that the equality function $\widetilde{eq} : \mathbb{F}_{q^k}^{\log n} \times \mathbb{F}_{q^k}^{\log n} \to \mathbb{F}_{q^k}$ is defined

$$\widetilde{eq}(t,i) = \prod_j (t_j \cdot i_j + (1-t_j) \cdot (1-i_j))$$

Using $\widetilde{eq}$, we can then batch the first type of constraints to show that the following polynomial $H_\alpha(t) : \mathbb{F}_{q^k}^{\log n} \to \mathbb{F}_{q^k}$ is a zero polynomial.

$$H_\alpha(t) := \sum_{i \in [n]} \widetilde{eq}(t,i) \left(\sum_{u,\ell} \widetilde{M}_\alpha(i,u) \cdot \widetilde{w}(u,\ell) \cdot \widetilde{\alpha}(\ell) - \mathbf{y}_i(\alpha) \right), \tag{21}$$

Similarly, we can also batch the second type of constraints (smallness check) as proving that the following polynomial $H_0 : \mathbb{F}_{q^k}^{\log\mu+\log d} \to \mathbb{F}_{q^k}$ is a zero polynomial.

$$H_0(t) := \sum_{\ell,u \le \mu} \widetilde{eq}(t,(u,\ell)) \cdot \widetilde{w}(u,\ell)(\widetilde{w}(u,\ell)-1)(\widetilde{w}(u,\ell)+1) \cdots (\widetilde{w}(u,\ell)-\beta)(\widetilde{w}(u,\ell)+\beta). \tag{22}$$

Finish the proof using sumcheck protocols. Now, the goal of the honest prover is to prove that $H_0(t) \equiv 0$ and $H_\alpha(t) \equiv 0$. We adopt a standard approach again: the verifier sends two random points, τ_0 and τ_1, and the prover then proves that $H_0(\tau_0) = 0$ and $H_\alpha(\tau_1) = 0$. We capture this with the following protocol. Similarly as in Lemma 7, we can show coordinate-wise special soundness of this protocol by treating (τ_0, τ_1) as a vector of $\log\mu + \log d + \log n$ coordinates. Due to space constraints, we provide a more formal analysis in Lemma 8.

Now, fix τ_0, τ_1 and recall that by definition of H_0 and H_α:

$$H_0(\tau_0) := \sum_{\ell,u \le \mu} \widetilde{eq}(\tau_0,(u,\ell)) \widetilde{w}(u,\ell)(\widetilde{w}(u,\ell)-1)(\widetilde{w}(u,\ell)+1) \cdots (\widetilde{w}(u,\ell)-\beta)(\widetilde{w}(u,\ell)+\beta),$$

and

$$H_\alpha(\tau_1) = \sum_{u,\ell} \widetilde{w}(u,\ell) \cdot \widetilde{\alpha}(\ell) \sum_i \widetilde{eq}(\tau_1,i) \cdot \widetilde{M}_\alpha(i,u) - \sum_i \widetilde{eq}(\tau_1,i) \cdot \mathbf{y}_i(\alpha).$$

[7] If we need to prove different ranges for different (u,ℓ), as in the setting of (19), we can simply use different polynomials for the range proof.

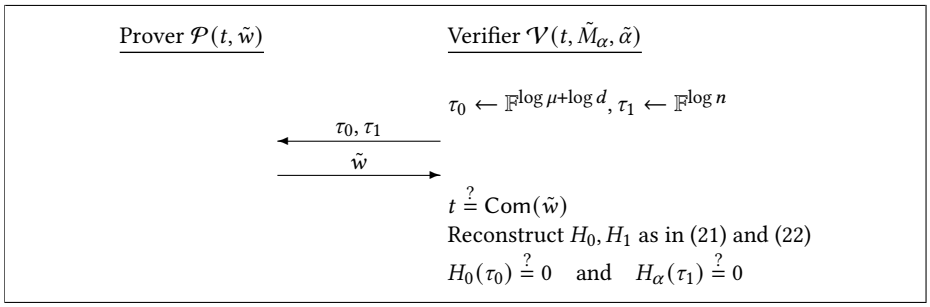

Fig. 5. Proving that polynomials H_0 and H_α are zero via random evaluation.

We then use sumcheck to prove that $H_0(\tau_0) = H_\alpha(\tau_1) = 0$. Defining polynomials

$$F_{0,\tau_0}(x,y) := \widetilde{eq}(\tau_0,(x,y))\widetilde{w}(x,y)(\widetilde{w}(x,y)-1)(\widetilde{w}(x,y)+1)\cdots(\widetilde{w}(x,y)-\beta)\cdot(\widetilde{w}(x,y)+\beta),$$

and

$$F_{\alpha,\tau_1}(x,y) := \widetilde{w}(x,y)\cdot\widetilde{\alpha}(y)\cdot\sum_i \widetilde{eq}(\tau_1,i)\cdot\widetilde{M}_\alpha(i,x),$$

we have that $H_0(\tau_0) = 0$ and $H_\alpha(\tau_1) = 0$ if and only if

$$\sum_{u,\ell} F_{0,\tau_0}(u,\ell) = 0 \quad \text{and} \quad \sum_{u,\ell} F_{\alpha,\tau_1}(u,\ell) = a,$$

where $a := \sum_i \widetilde{eq}(\tau_1, i)\cdot \mathbf{y}_i(\alpha)$, which the verifier can compute on their own. Therefore, we lastly apply sumcheck on F_{0,τ_0} and F_{α,τ_1}.

A reinterpretation of the sumcheck protocol. In the standard sumcheck protocol, the prover and the verifier communicate univariate polynomials and random field elements. In the last round of the protocol, the verifier makes a single evaluation query to the polynomial. However, since we rely on the special soundness notion (see Lemma 9), we reinterpret the sumcheck protocol as a composition of single-round communication protocols to simplify the analysis. In Figure 6, we demonstrate the sumcheck protocol for polynomials of the form $H(\mathbf{X}) := P(\mathbf{X})\cdot Q(\tilde{w}(\mathbf{X}))$, where P and Q are public information and $\tilde{w}(\mathbf{X})$ is the witness. We note that both F_{0,τ_0} and F_{α,τ_1} have this form.

Compared to the standard sumcheck protocol, we note that the prover usually does not send $\widetilde{w}$ to the verifier directly. Instead, it typically proves knowledge of $\widetilde{w}$ by continuing the sumcheck protocol, and finally by providing a proof of the evaluation of $\widetilde{w}$. Since special soundness is composable, we can obtain the special soundness of the sumcheck protocol by composing Figure 6 for all $i \in [\ell]$. Finally, the prover outputs $y' := \tilde{w}(a_1,\ldots,a_\ell)$, which allows the verifier to compute the evaluation $H(a_1,\ldots,a_\ell)$. What is left for the prover to do now is prove the polynomial evaluation claim for $\tilde{w}$. This ends the first iteration of our protocol.

The complete protocol. The final protocol is a natural composition of subprotocols in Figures 4, 5 and 6. By combining Lemmas 7, 8 and 9, we conclude that our final protocol satisfies coordinate-wise special soundness.

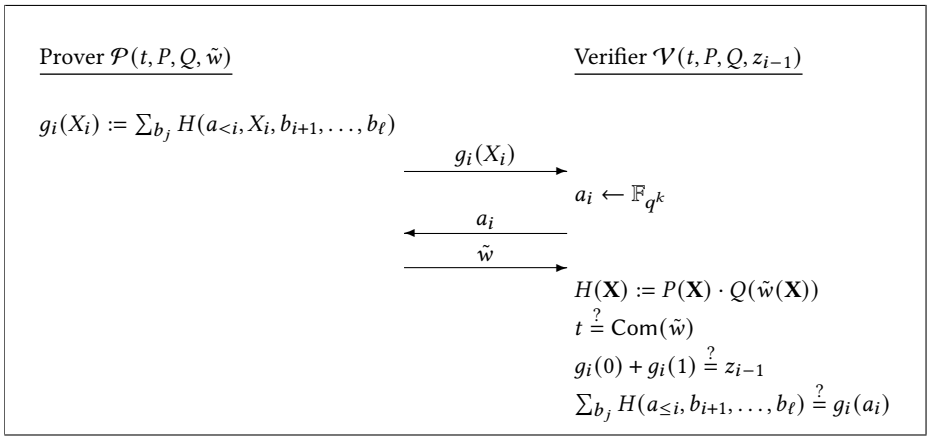

Fig. 6. Proving $\sum_{b_i,\ldots,b_\ell} H(a_1, \ldots, a_{i-1}, b_i, \ldots, b_\ell) = z_{i-1}$ for $i \in [\ell]$, where $H(\mathbf{X}) := P(\mathbf{X}) \cdot Q(\tilde{w}(\mathbf{X}))$. We set z_0 to be the initial sum, and $z_i := g_i(a_i)$.

4.4 Asymptotic Efficiency Analysis

Suppose we want to prove knowledge of the committed multilinear polynomial $f \in \mathbb{Z}_q^{\leq 1}[X_1, \ldots, X_\ell]$ which satisfies a polynomial evaluation claim over $\mathbb{F}_{q^k}$. In the following, let $L := 2^\ell$.

Basic parameters. Let $q = \text{poly}(\lambda, L)$ and $k = \lambda / \log q$, so that $\mathbb{F}_{q^k}$ is of exponential size to ensure negligible knowledge soundness in Lemmas 7, 8 and 9. We set the ring dimension $d = \text{poly}(\lambda)$, so $\alpha = \log \lambda$, and $n_A = n_B = n_D = O(1)$ for hardness of the Module-SIS problem. Then, we pick $\omega = O(d)$ for sufficiently large challenge space so that knowledge soundness error in Lemma 6 becomes negligible. We pick the decomposition base $b = O(1)$ and therefore $\delta = O(\log q)$. Recall that β in Section 4.2 was the upper-bound on the vector $\mathbf{z}$ (see Figure 3). A naive bound for β is:

$$\left\| \sum_{i=1}^{2^r} c_i \mathbf{s}_i \right\|_\infty \leq \sum_{i=1}^{2^r} \omega \cdot b = 2^r \omega b =: \beta.$$

Hence, if we choose the smallest integer τ such that $b^\tau > \beta$, then $\tau = O(r + \alpha)$.

Witness length reduction after one iteration. By Section 3, we can consider polynomials over $\mathbf{R}_q$ with $\ell - \alpha$ variables. As in Section 4.2, we define variables m, r such that $m + r = \ell - \alpha$. Let $m = r = (\ell - \alpha)/2$ (we implicitly assume $\ell - \alpha$ is even, which does not have any *asymptotic* impact). Then, the witness $(\hat{\mathbf{t}}, \hat{\mathbf{w}}, \hat{\mathbf{z}})$, parsed in Section 4.3 as $\mathbf{z}$, together with the "quotient" $\mathbf{r}$ from the ring switching methodology, is together of length – in terms of $\mathbb{Z}_q$ elements:

$$((n_A + 1)2^r + (\tau + 1)2^m) \cdot \delta \cdot d = O((\ell + \alpha)^2 \cdot 2^{\frac{\ell+\alpha}{2}}).$$

Hence, we conclude that after one round of iteration, the number of variables for the polynomial evaluation gets reduced (at most) to:

$$\ell \mapsto \frac{\ell + \alpha}{2} + 2\log(\ell + \alpha) + O(1) \leq \frac{2}{3}(\ell + \alpha) + O(1).$$

After t iterations, the length of the witness is reduced to (at most):

$$(\frac{2}{3})^t \cdot \ell + \sum_{i=1}^{t} (\frac{2}{3})^t \cdot \alpha + O(t) \leq (\frac{2}{3})^t \cdot \ell + 2\alpha + O(t).$$

Thus, after running our protocol $t = O(\log \ell)$ times we will output a witness polynomial with at most $O(\log \ell + \alpha)$ variables. Consequently, sending all the $\mathbb{Z}_q$-coeffients of the final polynomial in the clear costs $\text{poly}(\ell, d) = \text{poly}(\ell, \lambda)$ bits.

Proof size. Having non-interactive proofs via Fiat-Shamir transformation in mind, we only count the prover messages. The proof size of a single iteration can be split in three parts. The first is the transformation from $\mathbb{Z}_q$ to $\mathbf{R}_q$ from Section 3, where the prover sends $d + k(k-1)$ $\mathbb{Z}_q$-elements. The second is the commitment $\mathbf{v}$ from Section 4.2, which is of size $n_D \cdot d$ elements in $\mathbb{Z}_q$. Finally, in the protocol in Section 4.3, for each sumcheck round (out of $O(\ell + \alpha)$) the prover sends 2 (resp. $b + 1$) elements in $\mathbb{F}_{q^k}$ corresponding to the coefficients of the univariate polynomials obtained from F_{α,τ_1} (resp. F_{0,τ_0}). Finally, the prover sends the commitment to $\tilde{w}$ which is of length $n_B \cdot d$ $\mathbb{Z}_q$ elements (resp. Section 4). In total, a single iteration of the protocol has proof size, in terms of $\mathbb{Z}_q$ elements:

$$(n_B + n_D + 1) \cdot d + (k-1) \cdot k + (b+2) \cdot k \cdot O(\ell + \alpha) = O\left(\frac{\lambda}{\log \lambda} \cdot \ell\right) + \text{poly}(\lambda).$$

By recursively running the protocol $O(\log \ell)$ times and by having $q = \text{poly}(\lambda, L)$, we obtain succinct $\text{poly}(\ell, \lambda)$ proof size.

Verifier time. We split the verifier algorithm in three parts. For the transformation in Section 3 the verifier has to compute the $\mathbf{R}_q$-trace map consistency, which is of time $\text{poly}(\lambda)$. For the protocol in Section 4.2 the verifier does not check anything. Finally, in Section 4.3 the verifier has two tasks. First is checking that the univariate polynomials sent by the prover every sumcheck round are consistent. This takes $O(b) = O(1)$ $\mathbb{Z}_q$-operations. A much more expensive part is the final evaluation of the polynomial F_{α,τ_1} after sumcheck, which involves evaluating the (publicly known) multilinear extensions $\tilde{M}_\alpha$ $n := n_B + n_D + 3 = O(1)$ times. As shown in [VSBW13], to evaluate $\tilde{M}_\alpha$ once, it would take

$$O((n_A + 1)2^r + (\tau + 1)2^m \delta) = \tilde{O}(\sqrt{2^\ell / \lambda}) \text{ time}$$

in terms of field operations over $\mathbb{F}_{q^k}$ using dynamic programming. A single multiplication over $\mathbb{F}_{q^k}$ can be computed with $\tilde{O}(k)$ operations over $\mathbb{Z}_q$. Since $k = O(\lambda / \log q)$ the total verifier time is dominated by $\tilde{O}(\sqrt{2^\ell \cdot \lambda})$.

References

[ACFY24] G. Arnon, A. Chiesa, G. Fenzi, and E. Yogev, "STIR: Reed-solomon proximity testing with fewer queries," in *CRYPTO 2024, Part X*, L. Reyzin and D. Stebila, Eds., ser. LNCS, vol. 14929, Springer, Cham, Aug. 2024, pp. 380–413. DOI: 10.1007/978-3-031-68403-6_12.

[ACFY25] G. Arnon, A. Chiesa, G. Fenzi, and E. Yogev, "WHIR: Reed-solomon proximity testing with super-fast verification," in *EUROCRYPT 2025, Part IV*, S. Fehr and P.-A. Fouque, Eds., ser. LNCS, vol. 15604, Springer, Cham, May 2025, pp. 214–243. DOI: 10.1007/978-3-031-91134-7_8.

[Ach03] D. Achlioptas, "Database-friendly random projections: Johnson-lindenstrauss with binary coins," *J. Comput. Syst. Sci.*, vol. 66, no. 4, pp. 671–687, 2003.

[ACK21] T. Attema, R. Cramer, and L. Kohl, "A compressed Σ-protocol theory for lattices," in *CRYPTO 2021, Part II*, T. Malkin and C. Peikert, Eds., ser. LNCS, vol. 12826, Virtual Event: Springer, Cham, Aug. 2021, pp. 549–579. DOI: 10.1007/978-3-030-84245-1_19.

[AFLN24] M. R. Albrecht, G. Fenzi, O. Lapiha, and N. K. Nguyen, "SLAP: Succinct lattice-based polynomial commitments from standard assumptions," in *EUROCRYPT 2024, Part VII*, M. Joye and G. Leander, Eds., ser. LNCS, vol. 14657, Springer, Cham, May 2024, pp. 90–119. DOI: 10.1007/978-3-031-58754-2_4.

[Ajt96] M. Ajtai, "Generating hard instances of lattice problems (extended abstract)," in *28th ACM STOC*, ACM Press, May 1996, pp. 99–108. DOI: 10.1145/237814.237838.

[AL21] M. R. Albrecht and R. W. F. Lai, "Subtractive sets over cyclotomic rings - limits of Schnorr-like arguments over lattices," in *CRYPTO 2021, Part II*, T. Malkin and C. Peikert, Eds., ser. LNCS, vol. 12826, Virtual Event: Springer, Cham, Aug. 2021, pp. 519–548. DOI: 10.1007/978-3-030-84245-1_18.

[APS15] M. R. Albrecht, R. Player, and S. Scott, *Journal of Mathematical Cryptology*, vol. 9, no. 3, pp. 169–203, 2015. DOI: doi:10.1515/jmc-2015-0016. [Online]. Available: https://doi.org/10.1515/jmc-2015-0016.

[BBC+18] C. Baum, J. Bootle, A. Cerulli, R. del Pino, J. Groth, and V. Lyubashevsky, "Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits," in *CRYPTO 2018, Part II*, H. Shacham and A. Boldyreva, Eds., ser. LNCS, vol. 10992, Springer, Cham, Aug. 2018, pp. 669–699. DOI: 10.1007/978-3-319-96881-0_23.

[BBHR18] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Fast reed-solomon interactive oracle proofs of proximity," in *ICALP 2018*, I. Chatzigiannakis, C. Kaklamanis, D. Marx, and D. Sannella, Eds., ser. LIPIcs, vol. 107, Schloss Dagstuhl, Jul. 2018, 14:1–14:17. DOI: 10.4230/LIPIcs.ICALP.2018.14.

[BC24] D. Boneh and B. Chen, *LatticeFold: A lattice-based folding scheme and its applications to succinct proof systems*, Cryptology ePrint Archive, Report 2024/257, 2024. [Online]. Available: https://eprint.iacr.org/2024/257.

[BC25] D. Boneh and B. Chen, "LatticeFold+: Faster, simpler, shorter lattice-based folding for succinct proof systems," in *CRYPTO 2025, Part VII*, Y. T. Kalai

and S. F. Kamara, Eds., ser. LNCS, vol. 16006, Springer, Cham, Aug. 2025, pp. 327–361. DOI: 10.1007/978-3-032-01907-3_11.

[BHV+23] R. Bhadauria, C. Hazay, M. Venkitasubramaniam, W. Wu, and Y. Zhang, "Private polynomial commitments and applications to MPC," in *PKC 2023, Part II*, A. Boldyreva and V. Kolesnikov, Eds., ser. LNCS, vol. 13941, Springer, Cham, May 2023, pp. 127–158. DOI: 10.1007/978-3-031-31371-4_5.

[BL17] C. Baum and V. Lyubashevsky, *Simple amortized proofs of shortness for linear relations over polynomial rings*, Cryptology ePrint Archive, Report 2017/759, 2017. [Online]. Available: https://eprint.iacr.org/2017/759.

[BLNS20] J. Bootle, V. Lyubashevsky, N. K. Nguyen, and G. Seiler, "A non-PCP approach to succinct quantum-safe zero-knowledge," in *CRYPTO 2020, Part II*, D. Micciancio and T. Ristenpart, Eds., ser. LNCS, vol. 12171, Springer, Cham, Aug. 2020, pp. 441–469. DOI: 10.1007/978-3-030-56880-1_16.

[BS23] W. Beullens and G. Seiler, "LaBRADOR: Compact proofs for R1CS from module-SIS," in *CRYPTO 2023, Part V*, H. Handschuh and A. Lysyanskaya, Eds., ser. LNCS, vol. 14085, Springer, Cham, Aug. 2023, pp. 518–548. DOI: 10.1007/978-3-031-38554-4_17.

[CBBZ23] B. Chen, B. Bünz, D. Boneh, and Z. Zhang, "HyperPlonk: Plonk with linear-time prover and high-degree custom gates," in *EUROCRYPT 2023, Part II*, C. Hazay and M. Stam, Eds., ser. LNCS, vol. 14005, Springer, Cham, Apr. 2023, pp. 499–530. DOI: 10.1007/978-3-031-30617-4_17.

[CHM+20] A. Chiesa, Y. Hu, M. Maller, P. Mishra, P. Vesely, and N. P. Ward, "Marlin: Preprocessing zkSNARKs with universal and updatable SRS," in *EUROCRYPT 2020, Part I*, A. Canteaut and Y. Ishai, Eds., ser. LNCS, vol. 12105, Springer, Cham, May 2020, pp. 738–768. DOI: 10.1007/978-3-030-45721-1_26.

[CLM23] V. Cini, R. W. F. Lai, and G. Malavolta, "Lattice-based succinct arguments from vanishing polynomials - (extended abstract)," in *CRYPTO 2023, Part II*, H. Handschuh and A. Lysyanskaya, Eds., ser. LNCS, vol. 14082, Springer, Cham, Aug. 2023, pp. 72–105. DOI: 10.1007/978-3-031-38545-2_3.

[CMNW24] V. Cini, G. Malavolta, N. K. Nguyen, and H. Wee, "Polynomial commitments from lattices: Post-quantum security, fast verification and transparent setup," in *CRYPTO 2024, Part X*, L. Reyzin and D. Stebila, Eds., ser. LNCS, vol. 14929, Springer, Cham, Aug. 2024, pp. 207–242. DOI: 10.1007/978-3-031-68403-6_7.

[FMN24] G. Fenzi, H. Moghaddas, and N. K. Nguyen, "Lattice-based polynomial commitments: Towards asymptotic and concrete efficiency," *Journal of Cryptology*, vol. 37, no. 3, p. 31, Jul. 2024. DOI: 10.1007/s00145-024-09511-8.

[GHL22] C. Gentry, S. Halevi, and V. Lyubashevsky, "Practical non-interactive publicly verifiable secret sharing with thousands of parties," in *EUROCRYPT 2022, Part I*, O. Dunkelman and S. Dziembowski, Eds., ser. LNCS,

vol. 13275, Springer, Cham, May 2022, pp. 458–487. DOI: 10.1007/978-3-031-06944-4_16.

[GLS+23] A. Golovnev, J. Lee, S. T. V. Setty, J. Thaler, and R. S. Wahby, "Brakedown: Linear-time and field-agnostic SNARKs for R1CS," in *CRYPTO 2023, Part II*, H. Handschuh and A. Lysyanskaya, Eds., ser. LNCS, vol. 14082, Springer, Cham, Aug. 2023, pp. 193–226. DOI: 10.1007/978-3-031-38545-2_7.

[HMZ25] M.-Y. M. Huang, X. Mao, and J. Zhang, *Sublinear proofs over polynomial rings*, Cryptology ePrint Archive, Report 2025/199, 2025. [Online]. Available: https://eprint.iacr.org/2025/199.

[KLNO24] M. Klooß, R. W. F. Lai, N. K. Nguyen, and M. Osadnik, "RoK, paper, SISsors toolkit for lattice-based succinct arguments - (extended abstract)," in *ASIACRYPT 2024, Part V*, K.-M. Chung and Y. Sasaki, Eds., ser. LNCS, vol. 15488, Springer, Singapore, Dec. 2024, pp. 203–235. DOI: 10.1007/978-981-96-0935-2_7.

[KLNO25] M. Klooß, R. W. F. Lai, N. K. Nguyen, and M. Osadnik, "Rok and roll - verifier-efficient random projection for $\tilde{o}(\lambda)$-size lattice arguments," *IACR Cryptol. ePrint Arch.*, p. 1220, 2025.

[KST22] A. Kothapalli, S. Setty, and I. Tzialla, "Nova: Recursive zero-knowledge arguments from folding schemes," in *CRYPTO 2022, Part IV*, Y. Dodis and T. Shrimpton, Eds., ser. LNCS, vol. 13510, Springer, Cham, Aug. 2022, pp. 359–388. DOI: 10.1007/978-3-031-15985-5_13.

[KZG10] A. Kate, G. M. Zaverucha, and I. Goldberg, "Constant-size commitments to polynomials and their applications," in *ASIACRYPT 2010*, M. Abe, Ed., ser. LNCS, vol. 6477, Springer, Berlin, Heidelberg, Dec. 2010, pp. 177–194. DOI: 10.1007/978-3-642-17373-8_11.

[LFKN90] C. Lund, L. Fortnow, H. J. Karloff, and N. Nisan, "Algebraic methods for interactive proof systems," in *31st FOCS*, IEEE Computer Society Press, Oct. 1990, pp. 2–10. DOI: 10.1109/FSCS.1990.89518.

[LNP22] V. Lyubashevsky, N. K. Nguyen, and M. Plançon, "Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general," in *CRYPTO 2022, Part II*, Y. Dodis and T. Shrimpton, Eds., ser. LNCS, vol. 13508, Springer, Cham, Aug. 2022, pp. 71–101. DOI: 10.1007/978-3-031-15979-4_3.

[LNS21] V. Lyubashevsky, N. K. Nguyen, and G. Seiler, "Shorter lattice-based zero-knowledge proofs via one-time commitments," in *PKC 2021, Part I*, J. Garay, Ed., ser. LNCS, vol. 12710, Springer, Cham, May 2021, pp. 215–241. DOI: 10.1007/978-3-030-75245-3_9.

[LS15] A. Langlois and D. Stehlé, "Worst-case to average-case reductions for module lattices," *DCC*, vol. 75, no. 3, pp. 565–599, 2015. DOI: 10.1007/s10623-014-9938-4.

[LS18] V. Lyubashevsky and G. Seiler, "Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs," in *EUROCRYPT 2018, Part I*, J. B. Nielsen and V. Rijmen, Eds., ser. LNCS, vol. 10820, Springer, Cham, Apr. 2018, pp. 204–224. DOI: 10.1007/978-3-319-78381-9_8.

[NS24] N. K. Nguyen and G. Seiler, "Greyhound: Fast polynomial commitments from lattices," in *CRYPTO 2024, Part X*, L. Reyzin and D. Stebila, Eds., ser. LNCS, vol. 14929, Springer, Cham, Aug. 2024, pp. 243–275. DOI: 10.1007/978-3-031-68403-6_8.

[NS25] W. Nguyen and S. Setty, *Neo: Lattice-based folding scheme for CCS over small fields and pay-per-bit commitments*, Cryptology ePrint Archive, Report 2025/294, 2025. [Online]. Available: https://eprint.iacr.org/2025/294.

[Set20] S. Setty, "Spartan: Efficient and general-purpose zkSNARKs without trusted setup," in *CRYPTO 2020, Part III*, D. Micciancio and T. Ristenpart, Eds., ser. LNCS, vol. 12172, Springer, Cham, Aug. 2020, pp. 704–737. DOI: 10.1007/978-3-030-56877-1_25.

[STW24] S. T. V. Setty, J. Thaler, and R. S. Wahby, "Unlocking the lookup singularity with Lasso," in *EUROCRYPT 2024, Part VI*, M. Joye and G. Leander, Eds., ser. LNCS, vol. 14656, Springer, Cham, May 2024, pp. 180–209. DOI: 10.1007/978-3-031-58751-1_7.

[VSBW13] V. Vu, S. T. V. Setty, A. J. Blumberg, and M. Walfish, "A hybrid architecture for interactive verifiable computation," in *2013 IEEE Symposium on Security and Privacy*, IEEE Computer Society Press, May 2013, pp. 223–237. DOI: 10.1109/SP.2013.48.

[WTs+18] R. S. Wahby, I. Tzialla, a. shelat, J. Thaler, and M. Walfish, "Doubly-efficient zkSNARKs without trusted setup," in *2018 IEEE Symposium on Security and Privacy*, IEEE Computer Society Press, May 2018, pp. 926–943. DOI: 10.1109/SP.2018.00060.

A Coordinate-Wise Special Soundness Proofs

In this section, we include proofs that our subprotocols satisfy the notion of coordinate-wise special soundness (cf. Section 2.3).

Lemma 6 (CWSS of Figure 3). *Let* $(\bar{\beta}, \bar{\omega}, \bar{\gamma}) := (2b^k, 2\omega, b)$. *Suppose* $\omega < \frac{1}{2\sqrt{2}} q^{1/2}$. *Then, given the public parameters, the statement* $(\mathbf{u}, u)$, *and* $2^r + 1$ *valid transcripts of the form*

$$\mathrm{tr}^{(j)} := (\mathbf{v}, \mathbf{c}^{(j)}, (\hat{\mathbf{w}}^{(j)}, \hat{\mathbf{t}}^{(j)}, \hat{\mathbf{z}}^{(j)})) \quad \textit{where} \quad (\mathbf{c}^{(j)})_{0 \le j \le 2^r} \in \mathsf{SS}(C, 2^r, 2),$$

one can either efficiently extract a weak opening $(\bar{\mathbf{s}}^{(j)}, \bar{\mathbf{t}}^{(j)}, c^{(j)})_{j \in [2^r]}$ *for the commitment* $\mathbf{u}$ *such that*

$$u = \mathbf{b}^\intercal \begin{bmatrix} \mathbf{a}^\intercal \mathbf{G}_{2^m} & & \\ & \ddots & \\ & & \mathbf{a}^\intercal \mathbf{G}_{2^m} \end{bmatrix} \begin{bmatrix} \bar{\mathbf{s}}_1 \\ \vdots \\ \bar{\mathbf{s}}_{2^r} \end{bmatrix},$$

or solve $\mathsf{MSIS}_{q,d,n_B,\delta 2^r n_A,2b}$ *or* $\mathsf{MSIS}_{q,d,n_D,\delta 2^r,2b}$.

Proof. For each $j \in [2^r]$, let us reconstruct the vector $\mathbf{z}^{(j)} := \mathbf{J}\hat{\mathbf{z}}^{(j)}$ as in (18). Then, we have $\|\mathbf{z}^{(j)}\|_\infty \le b^k$.

Without loss of generality, assume that $\mathbf{c}^{(0)}$ is the "central vector". First, suppose that for two distinct $i \neq j$ we have $\hat{\mathbf{t}}^{(i)} \neq \hat{\mathbf{t}}^{(j)}$. In particular, we have $\mathbf{B}\hat{\mathbf{t}}^{(i)} = \mathbf{u} = \mathbf{B}\hat{\mathbf{t}}^{(j)}$, and so we have found a short solution of norm at most $2b$ for the matrix $\mathbf{B}$. Similarly, we argue for the vectors $\hat{\mathbf{w}}^{(j)}$. Hence, from now on we assume that all $\hat{\mathbf{t}}^{(j)} := \hat{\mathbf{t}}$ and $\hat{\mathbf{w}}^{(j)} := \hat{\mathbf{w}}$ are the same. Denote $\mathbf{w} := \mathbf{G}_{2^r}\hat{\mathbf{w}}$ and parse $\mathbf{w} = (w_i)_i, \hat{\mathbf{t}} := (\hat{\mathbf{t}}_i)_i$. Then, we have $\mathbf{b}^\intercal \mathbf{w} = u$.

Fix $j \in [2^r]$ and consider the transcripts $\mathrm{tr}^{(j)}$ and $\mathrm{tr}^{(0)}$ with the verifier challenges denoted as $\mathbf{c}^{(j)} := (c_{j,i})_i$ and $\mathbf{c}^{(0)} := (c_{0,i})_i$. Recall that the two challenge vectors differ*exactly* in the j-th coordinate and $c_{j,i} = c_{0,i}$ for all $i \neq j$. By subtracting the two corresponding verification equations, we obtain

$$\mathbf{b}^\intercal \mathbf{G}_{2^m}(\mathbf{z}^{(j)} - \mathbf{z}^{(0)}) = (c_{j,j} - c_{0,j}) \cdot w_j \quad \text{and} \quad \mathbf{A}(\mathbf{z}^{(j)} - \mathbf{z}^{(0)}) = (c_{j,j} - c_{0,j}) \cdot \mathbf{G}_{b,n}\hat{\mathbf{t}}_j.$$

Hence, define our extracted opening as $\bar{\mathrm{st}} := (\mathbf{s}_j, \hat{\mathbf{t}}_j)_j$ where

$$\mathbf{s}_j := \frac{\mathbf{z}^{(j)} - \mathbf{z}^{(0)}}{c_{j,j} - c_{0,j}} \quad \text{for all } j \in \{0,1\}^r$$

and the slack vector $(\bar{c}_j)_j$ is defined as $\bar{c}_j := c_{j,j} - c_{0,j}$. By construction, we have $\|\bar{c}_j\|_1 \leq 2\omega$ and $\|\bar{c}_j \mathbf{s}_j\|_\infty \leq 2b^k$. By Lemma 2, $\bar{c}_j$ is invertible over $\mathbf{R}_q$ since $\|\bar{c}_j\|_\infty \leq 2\omega < \frac{1}{\sqrt{2}} q^{1/2}$. Finally, $\mathbf{b}^\intercal \mathbf{G}_{2^m} \mathbf{s}_j = w_j$ for all $j \in [2^r]$, which concludes the proof. □

Lemma 7 (Special soundness of Figure 4). *There is an efficient, deterministic algorithm that given a statement $(\mathbf{M}_{i,j}, \mathbf{y}_i, \beta)$ and $2d$ valid transcripts $(t_i, \alpha_i, (\mathbf{z}_i, \mathbf{r}_i))$ of the protocol in Figure 4 such that $(\alpha_i)_i \in \mathrm{SS}(\mathbb{F}_{q^k}, 1, 2d)$, can either extract $\bar{\mathbf{z}}, \bar{\mathbf{r}}$ which satify $\mathbf{Mz} = \mathbf{y} + (X^d + 1) \cdot \mathbf{r}$ over $\mathbb{Z}_q[X]$, or break binding of the commitment scheme* Com.

Proof. If for some $i \neq j$ we have $(\mathbf{z}_i, \mathbf{r}_i) \neq (\mathbf{z}_j, \mathbf{r}_j)$, then we automatically break the binding property of the commitment scheme. Otherwise, define $\bar{\mathbf{z}} := \mathbf{z}_i, \bar{\mathbf{r}} := \mathbf{r}_i$ for all i. Then, for $2d$ distinct evaluation points α_j, we have

$$\forall i \in [n], \sum_j \mathbf{M}_{i,j}(\alpha_j)\mathbf{z}_j(\alpha_j) \stackrel{?}{=} \mathbf{y}_i(\alpha_j) + (\alpha^d + 1) \cdot \mathbf{r}_i(\alpha_j).$$

Since the polynomial $\sum_j \mathbf{M}_{i,j}\mathbf{z}_j - \mathbf{y}_i - (X^d + 1) \cdot \mathbf{r}_i$ has degree at most $2d - 1$, this implies that it must be a zero polynomial, which finishes the proof. □

Lemma 8 (CWSS of Figure 5). *There is an efficient, deterministic algorithm that given a statement $(t, \tilde{M}_\alpha, \tilde{\alpha})$ and $D := \max(2d, 2\beta + 1)$ valid transcripts $((\tau_{i,0}, \tau_{i,1}), \tilde{w}_i)$ of the protocol in Figure 5 such that $(\tau_{i,0}, \tau_{i,1})_i \in \mathrm{SS}(\mathbb{F}_{q^k}, 2, D)$, can either extract a valid opening $\tilde{w}$ of t, i.e. $t = \mathrm{Com}(\tilde{w})$, which satisfies $H_0(t_0) \equiv 0$ and $H_\alpha(t) \equiv 0$ (defined in Equations* (21) *and* (22)*), or break binding of the commitment scheme* Com.

Proof. If for some $i \neq j$ we have $\tilde{w}_i \neq \tilde{w}_j$, then we break the binding property of the commitment scheme. Otherwise, define $\tilde{w} := \tilde{w}_i$ for all i. By definition of $D = \max(2d, 2\beta + 1)$ and the coordinate-wise special soundness, we have found at least $2d$ (resp. $2\beta + 1$) distinct roots for H_α (H_0), which implies that the aforementioned polynomials are equal to zero. □

Lemma 9 (Special soundness of Figure 6). *There is an efficient, deterministic algorithm that given a statement and $D := \deg(H) + 1$ valid transcripts $(g_i(X_i), a_{i,j}, \tilde{w})_j$ of the protocol in Figure 6 such that $(a_{i,j})_j \in \mathsf{SS}(\mathbb{F}_{q^k}, 1, D)$, can either extract a valid opening $\tilde{w}$ of t, i.e. $t = \mathsf{Com}(\tilde{w})$, which satisfies $\sum_{b_j} H(a_1, \ldots, a_i, b_{i+1}, \ldots, b_\ell) = g_i(a_i)$, or break binding of the commitment scheme* Com.

Proof. If for some $i \neq j$ we have $\tilde{w}_i \neq \tilde{w}_j$, then we break the binding property of the commitment scheme. Otherwise, define $\tilde{w} := \tilde{w}_i$ for all i. By definition of $D = \deg(H) + 1$ and the coordinate-wise special soundness, we have found at least $\deg(H) + 1$ distinct roots for $\sum_{b_j} H(a_{<i}, X_i, b_{i+1}, \ldots, b_\ell) - g_i(X_i)$, which implies that $\sum_{b_j} H(a_{<i}, X_i, b_{i+1}, \ldots, b_\ell) - g_i(X_i) \equiv 0$.

B Concrete Instantiation

Avoiding re-decomposition. Having a recursive proof system in mind, a naive approach to commit to the multilinear polynomial $\tilde{w}$ (cf. Equation (20)) is to run again the commitment scheme from Section 4. The first part is the inner commitment, where we decompose the coefficients of $\tilde{w}$ (and thus the committed witness gets longer), and then run an Ajtai commitment scheme to get $(\mathbf{t}_i)_i$. A more natural and efficient way is to directly commit to $(\mathbf{z}', \mathbf{r}')$, as we know that the coefficients of $\mathbf{z}'$ are already short and no decomposition for $\mathbf{z}'$ is needed. It is fine from committing point of view, but how should one prove an evaluation claim

$$\mathsf{mle}[(\mathbf{z}', \mathbf{r}')](x'_1, \ldots, x'_\ell) = y',$$

when having only committed to $(\mathbf{z}', \mathbf{r}')$? We solve this issue using a simple observation. Namely, for any vectors i, j and $\mathbf{x}_0, \mathbf{x}_1$ we have

$$eq(i, \mathbf{x}_0) \cdot e(j, \mathbf{x}_1) = eq(i||j, \mathbf{x}_0||\mathbf{x}_1).$$

Then, by definition of the multilinear extension and for $\ell = m + r$, we can write

$$\begin{aligned}\mathsf{mle}[(\mathbf{z}', \mathbf{r}')](\mathbf{x}_0 \parallel \mathbf{x}_1) &= \sum_{i \in \{0,1\}^r} \sum_{j \in \{0,1\}^m} (\mathbf{z}', \mathbf{r}')_{i\parallel j} \cdot eq(i \parallel j, \mathbf{x}_0 \parallel \mathbf{x}_1) \\ &= \sum_{i \in \{0,1\}^r} eq(i, \mathbf{x}_0) \cdot \left(\sum_{j \in \{0,1\}^m} (\mathbf{z}', \mathbf{r}')_{i\parallel j} \cdot eq(j, \mathbf{x}_1) \right)\end{aligned}$$

which is exactly of the same form as (10). Therefore, we can recursively run our protocol to prove the evaluation of $\tilde{w}$ without explicitly calculating its coefficients.

Composing with LaBRADOR. Similarly as in Greyhound [NS24], instead of running the protocol from Section 4.3, at some point we can naturally switch to LaBRADOR [BS23]. This can be done when the witness length is small enough so that performing the Johnson-Lindenstrauss projection becomes cheap. In fact, although the sum check-based approach provides faster verification time, reducing witness length after a single

operation is slower. The reason is that we have to pick relatively small base b, as the sumcheck proof size consists of $O(b)$ field elements. This implies longer decomposition δ, and thus longer witness for the next round. On the other hand, the random projection approach allows larger bases b (at a fairly small cost), which makes δ concretely smaller. This allows LaBRADOR to recurse further until the final witness is cheap enough to send in the clear ($\approx$ 30KB).

Concrete parameters. To showcase the efficiency, let us set parameters for $\ell = 30$. We refer to Appendix C (and in particular Figure 8) for more rationale on the parameters. We aim for $\lambda = 128$ security level and use the Lattice Estimator [APS15] to measure hardness of the Module-SIS problem in the infinity norm.

In short, we pick $q \approx 2^{32}$, $\alpha = 10$, $r = 10$ and $m = 10$ and $(n_A, n_B, n_D) = (1, 1, 1)$. The decomposition base is $b = 16$, thus $\delta = 8$. For comparison with Greyhound, we consider polynomial evaluations over $\mathbb{F}_q$ instead of F_{q^k}. Then, the cost of the transformation in Section 3 is $d \log q \leq$ 4KB. Next is the commitment $\mathbf{v}$ is also of size 4KB. Length of the witness $\tilde{w}$ for sumcheck can be bounded by 2^{26}. We consider extension fields $\mathbb{F}_{q^k}$ for $k = 4$. Then, the cost of sumcheck is bounded by $26 \cdot 4 \cdot 32 \cdot (16 + 2)$ bits $\approx$ 7.31KB. Finally, we also need to commit to $\tilde{w}$ (to get ready for the next iteration), which again takes 4KB. Hence, a single iteration costs us 19.31KB.

For the next iteration, we can directly apply Greyhound for the witness length 2^{26} (see [NS24, Table 1]), where the proof size is $\approx$ 43KB. Hence, the total evaluation proof can be approximately 63KB.

C Experimental Results

To explore the concrete runtime of our protocol, we have developed a prototype implementation of Hachi in Rust. The source can be found here:

https://github.com/logger-pileup/hachi

We highlight that our implementation is still in an early stage of development and is amenable to a number of optimizations which would lead to considerable increases in performance. Despite this, we achieve reasonable concrete performance - particularly in verification time for large witnesses (30+ variables).

Variables	Commit	Prove	Verify
l=26	11.5	8.24	0.455
l=28	55.7	37.7	0.909
l=30	220	138	1.83

Fig. 7. Concrete running (in seconds) of the Hachi PCS with prototype implementation.

variable	description	value
ℓ	number of variables of witness polynomial	30
q	prime modulus	4294967197
k	degree of the extension field $\mathbb{F}_{q^k}$ for sumcheck	4
α	log of ring dimension	10
d	$d := 2^\alpha$ is the ring dimension of $\mathbf{R}_q$	1024
m	folding parameter	10
r	folding parameter	10
n_A	height of the matrix $\mathbf{A}$	1
n_B	height of the matrix $\mathbf{B}$	1
n_D	height of the matrix $\mathbf{D}$	1
z	maximum L_∞ norm of $\mathbf{z}$	4096
b	decomp. base (inner commitments)	16
δ	expansion factor for decomposing base witness	8
τ	expansion factor for decomposing $\mathbf{z}$	4
ω	L_1 norm of a challenge element	16
c	number of non-zero coefficients in challenge	16
Next witness size		2^{26}

Fig. 8. Concrete parameters for an $\ell = 30$ variable multilinear polynomial over $\mathbb{Z}_q$.

C.1 Supplying the witness

To ensure our implementation can be used for arbitrarily large polynomials without large memory requirements, we provide the witness polynomial as a stream of coefficients (signed 64 bit integers) so that only a small portion of the witness is stored in memory at any time. This design also decouples the processing of the witness from its underlying storage or format. For testing, we produce a sample witness by writing uniform random coefficients to a file and implement a file stream by memory mapping the file and allowing sequential access. This adds some overhead to the running time of the functions (compared to storing the whole witness in memory) however our testing has shown this overhead to be relatively insignificant.

C.2 Sampling in $\mathbb{Z}_q$

To generate each commitment matrix $\mathbf{A}$, $\mathbf{B}$, $\mathbf{D}$; we expand a seed using a PRNG (ChaCha12 in our experiments) into a new seed for each row. We then use these to seed PRNGs which are used to produce each row as a stream of random coefficients. To sample uniformly randomly modulo q, we employ a simple rejection sampling technique. If q is an n bit prime, we first sample a 64 bit integer randomly and mask to n bits. We test if the result is below q and if not, we reject and try again. In our experiments, we set q as the largest suitable prime below 2^{32} - in this case rejection happens with probability close to zero. We have found this technique to be efficient and the overhead of sampling these matrices is insignificant in the running time of the protocol.

C.3 Parameter Selection and Decomposition

We heuristically set the ring dimension $d = 2^\alpha$ of $\mathbf{R}_q$ and the height of the commitment matrices, n, so that $n \cdot d \geq 2^{10}$. Computing the inner commitments $\mathbf{t}_i = \mathbf{A}\mathbf{s}_i$ therefore requires computing $2^r \cdot 2^m \cdot \delta \cdot 2^{max(10-\alpha,0)}$, where δ is the expansion factor from decomposing the witness. If we set $\alpha \leq 10$, we must do $2^{r+m+10-\alpha} = 2^{l+10-2\alpha}$ ring multiplications. This is inversely proportional to the square of the chosen ring dimension. However, using the NTT, the cost of doing one ring multiplication scales with $d \log d$. It follows that choosing a larger ring dimension (up to $\alpha = 10$) should reduce the running time of the commitment function. Indeed this is corroborated by our experiments. Choosing α larger than 10 is not worthwhile since it does not lead to the same reduction in the number of ring multiplications. Once d is chosen, we set r and m equally as $\lceil \frac{l-\alpha}{2} \rceil$ and $\lfloor \frac{l-\alpha}{2} \rfloor$. Our experiments show that choice of r and m does not hugely impact the running time. For decomposition, we choose power of two basis to employ fast decomposition with bitwise operations. Specifically, we decompose base 16 with an expansion factor of 8 for decomposition of arbitrary elements of $\mathbb{Z}_q$. We employ b's compliment decomposition so that both positive and negative values are decomposed into small non-negative integers.

C.4 Arithmetic over $\mathbf{R}_q$

To perform fast multiplication over $\mathbf{R}_q$, we use the concrete-ntt library. This library provides SIMD implementations of multiplication over NTT-friendly $\mathbb{Z}_q[X]/(X^d + 1)$ as well over $\mathbb{Z}[X]/(X^d + 1)$. The latter is achieved by first calculating the product over several smaller rings $\mathbb{Z}_{p_i}[X]/(X^d + 1)$ before using the Chinese remainder theorem to recover the result over $\mathbb{Z}[X]/(X^d + 1)$. This holds provided the product $p_1 \cdot ... \cdot p_k$ is larger than the largest coefficient in the product over $\mathbb{Z}[X]/(X^d + 1)$. Since we must choose q NTT-unfriendly, we use this 'native' interface to first compute the product over $\mathbb{Z}[X]/(X^d + 1)$ before explicitly reducing modulo q. This allows us to be fully flexible with the ring dimension and modulus - both of which can be chosen at run time. The library exposes three interfaces which accept polynomials with 32 bit, 64 bit and 128 bit unsigned integer coefficients respectively. The execution time of the multiplication is mostly determined by how many smaller primes p_i are required. This increases with the bit size of the coefficients and so the most optimal choice would be to use the 32 bit interface. Unfortunately, despite setting q to be 32 bits, we are not able to do this since the *product* of two 32 bit polynomials has coefficients much larger. Therefore, the product will have wrapped modulo 2^{32} before we are able to reduce it modulo q. In particular, we need to ensure the largest coefficient of the product fits in the bit width we choose. Naively, this means we must choose the bit width at least $d \cdot q^2 = 2^{74}$ so that we must use the least efficient 128 bit implementation. However, we note that in the protocol, one of the operands in the product is always small as we have decomposed it, or, in the case of the prover response z, it is a product of two small elements. Provided we decompose into non-negative integers $0, ..., b - 1$ (as concrete-ntt only works on unsigned integers), we are able to use the more efficient 64 bit implementation. In our experiments, we use the AVX-512 optimized 64 bit implementation of multiplication over $\mathbb{Z}[X]/(X^d + 1)$ provided by concrete-ntt and reduce naively modulo q. The cost

of the reduction modulo q is sub-optimal but is still only around 5% of the overall cost of the multiplication.

C.5 Arithmetic over $\mathbb{Z}_q[X]$

In the second part of the evaluation proof, the prover is required to 'lift' several equations from $\mathbf{R}_q$ to $\mathbb{Z}_q[X]$. This involves recomputing the relevant functions over $\mathbb{Z}_q[X]$. To do this, we note that we only ever need to compute a sum over a product of *two* polynomials, so that the degree does not exceed $2d - 1$. We therefore pad the polynomials to degree $2d - 1$ and use the concrete-ntt library, setting the dimension to $2d$. The result therefore does not incur any reduction modulo $X^{2d} + 1$. Given the full convolution, we perform polynomial division to recover the quotient part and the original result over $\mathbf{R}_q$. We note that division by $X^d + 1$ has a closed form solution and can be very efficiently implemented. In particular, considering the coefficients as two halves of length d, the quotient is simply the 'high' half whilst the remainder is the high half subtracted from the 'low' half.

C.6 Arithmetic for sparse polynomials

As we are able to choose a large ring dimension, we are able to use very sparse polynomials as challenges and maintain an exponential sized challenge set. In particular, for $d = 2^{10}$, each challenge polynomial has exactly $k = 16$ non-zero coefficients, each of which is -1 or 1. We sample such challenges using the Fisher-Yates algorithm to partially perform a permutation on $0, ..., d - 1$. After the first k elements of the permutation have been sampled, we stop and set these as the indices of the non-zero coefficients. Simultaneously, we sample an additional bit independently for each of these non-zero coefficients to determine its sign. We have implemented an optimized ring multiplication function for the case when one of the operands is a sparse polynomial. Instead of using the NTT, we simply perform $c \cdot k$ additions over the integers. This has shown to be considerably faster than (even the binary coefficient optimized) concrete-ntt library; despite not making use of any SIMD instructions. This is particularly important since computing the prover response $\mathbf{z}$ is the most expensive part of the evaluation proof. Moreover, since $\mathbf{z}$ is a sum of products of a sparse challenge polynomial and a small (decomposed) witness element, its coefficients do not wrap modulo q, provided we allow for both positive and negative numbers (hence the use of signed 64 bit integers). This means we do not need to reduce modulo q during these multiplications. Our experiments show that for $l = 30$ variables, the maximum coefficients of $\mathbf{z}$ are usually no more in magnitude than 2^{12}. This allows as to use an expansion factor of 4 when decomposing $\mathbf{z}$.

C.7 Arithmetic over $\mathbb{F}_{q^k}$

We use the popular ark-ff crate to define our extension fields and perform efficient field arithmetic. Unfortunately, our experiments showed that it was slower to use ark-ff to perform reductions over $\mathbb{Z}_q$ than simply reducing naively due to the repeated type

conversions required to make use of the concrete-ntt library. During verification, the verifier is required to evaluate the MLE $\widetilde{M}_\alpha(i, x)$ over the field extension at several points. We use the ark-poly to do this very efficiently so that the main cost to the verifier is computing the function table for $\widetilde{M}_\alpha(i, x)$.

C.8 Future Work

Our implementation is still in the early stages of development and there are several improvements that can be made to greatly improve its performance.

Reducing NTTs A very significant source of inefficiency in our current implementation is that, every time we want to compute a product over $\mathbf{R}_q$, we perform three sets of NTT operations: two sets of forward NTT for each operand, and an inverse NTT on the product. By sampling our commitment matrices directly in NTT form we could avoid having to do any forward NTTs on one of the operands, immediately reducing the cost of a multiplication almost by a third. Furthermore, if we keep the products in NTT form whilst performing the summation, we greatly reduce the number of inverse NTTs that must be performed, further reducing the cost of multiplications by about another one sixth. Since these NTTs are by far the dominant cost in the commitment function and a very significant cost in the evaluation proof and verification functions; this optimization should yield greatly improved run times.

SIMD instructions With the exception of the ring multiplication provided by concrete-ntt, none of our implementation makes use of SIMD instructions. By utilizing AVX-2/AVX-512 to perform, in particular, our optimized sparse polynomial arithmetic and decomposition (which are the next most expensive parts of the protocol) we could hope to see a substantial improvement in running time, as has been shown by implementations of existing protocols.

C.9 Comparison to existing implementation

Since our commitment function is the same as that presented in Greyhound [NS24] (except with different parameters) we are able to directly compare our implementation of the commitment function with existing implementations. When setting the parameters as close to those presented in Greyhound as possible; for a 26 variable polynomial our implementation took around 100 seconds. This is about 25 times slower than the time achieved (4.37 seconds) for the Greyhound implementation. This discrepancy is largely because the existing Greyhound implementation has already implemented the optimizations highlighted above and is highly optimized for a particular ring (whilst ours is very flexible).

To understand why the parameter regime from Greyhound is so much slower than our parameters (for our implementation), note that using the smaller ring dimension of $d = 64$ increases the number of ring multiplications by an order of magnitude compared to $d = 1024$. These multiplications are essentially the entire cost of the commitment function, and so - using the NTT - it is beneficial to perform as few as possible.

Whilst, on one hand, this demonstrates that there is significant work still to do on our implementation to compete with existing implementations, it may also indicate that the `scheme` itself is far more efficient than existing schemes. Should we achieve efficiency anywhere close to the Greyhound implementation, we will have concrete running times several times faster than existing schemes.