Greyhound: Fast Polynomial Commitments from Lattices

Ngoc Khanh Nguyen[2] and Gregor Seiler[1]

[1] IBM Research Europe, Zurich
[2] King's College London, London

Abstract. In this paper, we propose Greyhound, the first concretely efficient polynomial commitment scheme from standard lattice assumptions. At the core of our construction lies a simple three-round protocol for proving evaluations for polynomials of bounded degree N with verifier time complexity $O(\sqrt{N})$. By composing it with the LaBRADOR proof system (CRYPTO 2023), we obtain a succinct proof of polynomial evaluation (i.e. polylogarithmic in N) that admits a sublinear verifier runtime.

To highlight practicality of Greyhound, we provide implementation details including concrete sizes and runtimes. Notably, for large polynomials of degree at most $N = 2^{30}$, the scheme produces evaluation proofs of size 53KB, which is more than 10^4 times smaller than the recent lattice-based framework, called SLAP (EUROCRYPT 2024), and around three orders of magnitude smaller than Ligero (CCS 2017) and Brakedown (CRYPTO 2023).

Keywords: lattices, polynomial commitment scheme, SNARK, implementation, NTT, AVX-512

1 Introduction

A polynomial commitment scheme [KZG10] is a cryptographic primitive that allows one to commit to a degree-bounded polynomial $f \in R^{<N}[\mathsf{X}]$ over a ring R, and later prove evaluation statements, such as $f(x) = y$ for public $x, y \in R$. It is crucial for real-world applications that the size of the evaluation proof is succinct and can be efficiently verified (i.e. sublinear in N). Polynomial commitments, and variations thereof, have found numerous applications in constructing succinct non-interactive arguments of knowledge (SNARKs) [BFS20, BHR$^+$21, CHM$^+$20, GWC19, MBKM19], look-up arguments [STW23], verifiable secret sharing [BDK13], and multi-party computation [BHV$^+$23].

Due to fast development in building quantum computers, there is currently a strong need in designing quantum-safe polynomial commitments. This is evidenced by the NIST Post-Quantum Competition for standardizing quantum-safe key encapsulation mechanisms and digital signatures, where three out of four schemes, that were recently selected for standardization, rely on lattice-based assumptions. Not only does it imply that algebraic lattices are a suitable candidate for building more advanced quantum-safe applications in general, but also that lattice-based SNARKs are the most natural choice for upgrading the newly-standardized encryption and signature schemes with privacy-preserving properties, e.g. verifiable encryption or anonymous credentials.

Prior works on lattice-based polynomial commitments have been mainly of theoretical interest. Starting with the construction by Libert et al. [LRY16], polynomial commitments were treated as a direct application of (inner-product) functional commitments from lattices [ACL$^+$22, BCFL23, dCP23, FLV23, WW23b]. Even though the constructions offer succinct proofs, their verification runtime is sublinear in the degree N of the committed polynomial (via preprocessing) only if the evaluation point x is known in advance[3]. This is unfortunately not the case in SNARK-related applications, where the evaluation points are chosen uniformly at random. Moreover, only the works of [ACL$^+$22, BCFL23, FLV23] provide extractability, although under a *knowledge* assumption that has independently been broken in both classical [WW23a] and quantum setting [DAFS24].

A different (yet still intuitive) approach for building polynomial commitments can be described as simply combining a standard commitment scheme with an interactive proof of polynomial evaluation. The latter can then be turned non-interactive using Fiat-Shamir transformation [FS86]. For instance, Bootle et al. [BCS23] recently proposed a "Bulletproofs-type" polynomial evaluation proof, which achieves succinct verification via a delegation protocol [Lee21]. The resulting polynomial commitment relies only on a standard Module-SIS problem and requires no trusted setup. Unfortunately, as inherited from the original lattice-Bulletproofs [BLNS20], soundness error of the core evaluation protocol is non-negligible. Even though parallel repetition can be used to amplify soundness in the interactive setting [AF22], the Fiat-Shamir transformed protocol would suffer a super-polynomial reduction loss in the random oracle model (ROM) [AFK22]. Similar limitation can be found in the polynomial commitment scheme by Cini et al. [CLM23], whose security relies on a new Vanishing-SIS problem.

More recent constructions depart from the Bulletproofs paradigm and focus on the "split-and-fold" approach used in FRI low-degree test [BBHR18]. Notably, Fenzi et al. [FMN23] proposed a non-interactive polynomial commitment secure in the ROM under a new assumption called Power-BASIS – a more structured variant of the BASIS assumption introduced in [WW23b]. Unfortunately, the scheme requires a trusted setup, and what is worse, both the common reference string (CRS) size and committing runtime are quadratic in the degree bound N. A follow-up work by Albrecht et al. [AFLN24], called SLAP, removed the need of a new assumption, thus relying only on Module-SIS, while making the prover runtime quasi-linear. However, the remaining requirement on a trusted setup, together with concrete proof sizes reaching tens of megabytes make the scheme very unlikely to be practical. Some issues have been circumvented by the recent work by Cini et al. [CMNW24] who built an elegant SIS-based polynomial commitment with transparent setup and polylogarithmic verifier runtime. The

[3] It is worth noting that Orbweaver [FLV23] explicitly circumvents this issue.

concrete instantiation of the scheme, however, provides proof sizes in the order of single-digit megabytes for $N > 2^{25}$.

Even though none of the currently state-of-the-art lattice-based polynomial commitments have shown any significant sign of practicality, concretely efficient proof of knowledge for NP can be constructed from standard lattice assumptions. Notably, Beullens and Seiler [BS23] proposed a succinct proof system called LaBRADOR that achieves impressive proofs of size $\approx$ 50KB for large N. As a drawback, the protocol suffers from having linear verifier runtime, which limits the range of applications where the proof system could be used.

Based on the discussion above, we focus on the following research question:

Can we build a concretely efficient polynomial commitment scheme with transparent setup, sublinear verification complexity, and secure under standard lattice assumptions?

1.1 Our Contributions

Polynomial commitment scheme. In this work we propose Greyhound, the first practical lattice-based polynomial commitment scheme in the random oracle model. The construction requires no trusted setup and relies on the well-studied Module-SIS assumption.

Asymptotically, our scheme produces evaluation proofs of size $\mathsf{polylog}(N)$ which can be verified in time $O(\sqrt{N})$. As for concrete efficiency, we provide more details, as well as comparison with prior (plausibly) post-quantum polynomial commitments, in Tables 1 and 2. Notably, for large degrees N Greyhound provides 10^4 smaller evaluation proofs than SLAP [AFLN24], and around three orders of magnitude smaller proofs than the hash-based constructions [AHIV17, BBHR18, GLS+21]. Our construction also produces much smaller proof sizes compared to the more recent lattice-based polynomial commitments [CMNW24, HSS24] by a factor of at least 30. As for the commit and prover running time, Greyhound performs around $5 - 10$X faster than Brakedown and Ligero. As a drawback, our verification time seems comparable with Brakedown and two times slower than Ligero.

Library for fast ring operations. We have implemented an AVX-512 optimized library for polynomial arithmetic over small-degree power-of-two cyclotomic ring modulo multi-precision primes of the form $q \equiv 5 \pmod 8$. This library includes functions for sampling polynomials from several standard distributions as well as computing ring automorphisms directly in several different polynomial representations such as coefficient representations and multi-modular NTT representations. Moreover, our library contains a very fast implementation of the Johnson-Lindenstrauss projection [GHL22] needed in recent lattice-based zero-knowledge protocols [BS23, LNP22]. The implementation uses the Four Russian algorithm and vector shuffle instructions for in-register lookups. See Section 6 for more details.

		Transparent	Proof sizes for		
Scheme	Structure	setup	$N = 2^{26}$	$N = 2^{28}$	$N = 2^{30}$
Brakedown-PC	Hashes	✓	49157	93767	181948
Ligero-PC	Hashes	✓	7256	14383	28631
FRI-PC	Hashes	✓	740	–	–
FMN23-PC	Lattices	✗	–	–	8499
SLAP-PC	Lattices	✗	–	–	785408
CMNW24-PC	Lattices	✓	1546	–	5294
HSS24-PC	Lattices	✓	48640	–	–
Greyhound	Lattices	✓	46	53	53

Table 1: Concrete evaluation proof sizes (in KB) of Greyhound and comparison with prior plausibly post-quantum extractable polynomial commitments. Here, N is the degree bound on the committed polynomial over a suitably chosen finite field $\mathbb{F}_q$. Concrete sizes are set to reach λ-bit security level, where $\lambda \approx 128$. Sizes for Brakedown-PC [GLS+21], Ligero-PC [AHIV17] (Reed-Solomon rate of $\rho = 1/4$) and FRI-PC [BBHR18] are taken directly from [GLS+21, Figure 8], where for simplicity we assume that sizes for degree 2^{25} and 2^{26} are the same (and identically for $N = 2^{28}, 2^{30}$). As stated in the aforementioned figure, for $N > 2^{25}$ no sizes are provided for FRI-PC since the prover ran out of memory. Similarly for CMNW24-PC [CMNW24] and HSS24-PC [HSS24], the reported sizes (taken from the respective works) correspond to the degree 2^{25} instead of 2^{26}, where the instantiation of the latter scheme additionally provides zero-knowledge. Proof sizes for SLAP-PC [AFLN24] and FMN23-PC [FMN23] are taken from the respective works.

1.2 Technical Overview

Denote λ as a security parameter. Let d be a power-of-two and $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ be the ring of integers of the $2d$-th cyclotomic field. Take an odd prime q and define $\mathcal{R}_q := \mathcal{R}/(q)$ and $\delta := \lfloor \log q \rfloor$. For the sake of the overview, we consider base-two gadget matrices $\mathbf{G}_n := \mathbf{I}_n \otimes \begin{bmatrix} 1 & 2 & 4 & \cdots & 2^\delta \end{bmatrix} \in \mathcal{R}_q^{n \times n\delta}$ for $n \geqslant 1$. We define the standard inverse function $\mathbf{G}_n^{-1} : \mathcal{R}_q^n \to \mathcal{R}_q^{n\delta}$, which decomposes each entry w.r.t. base 2. In particular, for any $\mathbf{t} \in \mathcal{R}_q^n$, $\mathbf{G}_n^{-1}(\mathbf{t})$ has binary coefficients and $\mathbf{G}_n \mathbf{G}_n^{-1}(\mathbf{t}) = \mathbf{t}$.

Inner and outer commitments. The starting point of our construction is the basic commitment scheme from LaBRADOR [BS23]. Let $n, m, r \in \mathbb{N}$ and define the commitment key as a pair of uniformly random matrices $\mathbf{A} \in \mathcal{R}_q^{n \times m\delta}$ and $\mathbf{B} \in \mathcal{R}_q^{n \times rn\delta}$. Suppose we want to commit to arbitrary r vectors $\mathbf{f}_1, \ldots, \mathbf{f}_r \in \mathcal{R}_q^m$ of length m. The first step is to compute inner commitments $\mathbf{t}_i := \mathbf{A}\mathbf{G}_m^{-1}(\mathbf{f}_i) \in \mathcal{R}_q^n$ and their binary decomposition $\hat{\mathbf{t}}_i := \mathbf{G}_n^{-1}(\mathbf{t}_i)$ for $i \in [r]$. Then, the final outer commitment is

$$\mathbf{u} := \mathbf{B} \begin{bmatrix} \hat{\mathbf{t}}_1 \\ \vdots \\ \hat{\mathbf{t}}_r \end{bmatrix} \in \mathcal{R}_q^n. \tag{1}$$

	$N = 2^{26}$			$N = 2^{28}$			$N = 2^{30}$		
Scheme	Commit	Prove	Verify	Commit	Prove	Verify	Commit	Prove	Verify
Brakedown-PC	36	3.21	0.703	150	13	2.56	605	48.6	2.96
Ligero-PC	39.9	3.11	0.196	169	12.4	0.402	717	50	0.846
FRI-PC	168	185	0.041	–	–	–	–	–	–
HSS24-PC	188		1.07	–	–	–	–	–	–
Greyhound	4.37	2.03	0.492	21.2	8.21	1.15	132	41.2	2.80

Table 2: Concrete running time (in seconds) of the Greyhound polynomial commitment scheme and comparison with Brakedown-PC [GLS+21], Ligero-PC [AHIV17] (Reed-Solomon rate of $\rho = 1/2$) and FRI-PC [BBHR18]. The Greyhound runtimes were obtained by running the code on a single Intel Xeon Sapphire Rapids core at 3.2 GHz. The values for the related works are taken directly from [GLS+21, Figure 8] and [HSS24, Table 3], where for simplicity we assume that running times for degree 2^{25} and 2^{26} are the same (and identically for $N = 2^{28}, 2^{30}$).

Commitment opening for the message $(\mathbf{f}_i)_{i\in[r]}$ consists of short vectors $(\mathbf{s}_i, \hat{\mathbf{t}}_i)_{i\in[r]}$ which satisfy (i) $\mathbf{f}_i = \mathbf{G}_m\mathbf{s}_i$, (ii) $\mathbf{A}\mathbf{s}_i = \mathbf{G}_n\hat{\mathbf{t}}_i$ for $i \in [r]$ and (iii) Equation (1). Computational binding property follows directly from the Module-SIS assumption.

Simple proof of quadratic relations. Our base for constructing proofs of polynomial evaluation is the following three-round proof of knowledge of a commitment opening $(\mathbf{s}_i, \hat{\mathbf{t}}_i)_{i\in[r]}$ for the message $(\mathbf{f}_i)_{i\in[r]}$ which satisfies

$$\mathbf{a}^\intercal \left[\mathbf{f}_1 | \cdots | \mathbf{f}_r\right] \mathbf{b} = y.$$

The protocol can be described as follows. The prover starts by sending

$$\mathbf{w}^\intercal := \mathbf{a}^\intercal \left[\mathbf{f}_1 | \cdots | \mathbf{f}_r\right] = \mathbf{a}^\intercal \mathbf{G}_m \left[\mathbf{s}_1 | \cdots | \mathbf{s}_r\right] \in \mathcal{R}_q^r$$

to the verifier. Then, given a short challenge vector $\mathbf{c} \in \mathcal{R}_q^r$, the prover outputs

$$(\hat{\mathbf{t}}_i)_{i\in[r]} \quad \text{and} \quad \mathbf{z} := \left[\mathbf{s}_1 | \cdots | \mathbf{s}_r\right] \mathbf{c}.$$

Finally, verifier checks whether $(\hat{\mathbf{t}}_i)_{i\in[r]}, \mathbf{z}$ are short and if the following hold:

$$\mathbf{w}^\intercal\mathbf{b} \overset{?}{=} y, \quad \mathbf{w}^\intercal\mathbf{c} \overset{?}{=} \mathbf{a}^\intercal\mathbf{G}_m\mathbf{z}, \quad \mathbf{A}\mathbf{z} \overset{?}{=} \sum_{i=1}^{r} c_i \mathbf{G}_n \hat{\mathbf{t}}_i \quad \text{and} \quad \mathbf{u} \overset{?}{=} \mathbf{B} \begin{bmatrix} \hat{\mathbf{t}}_1 \\ \vdots \\ \hat{\mathbf{t}}_r \end{bmatrix}. \qquad (2)$$

Communication complexity of the three-round protocol is $O(rn + m\delta)$ elements over $\mathcal{R}_q$, which is sublinear in the witness size $N = r \cdot m$.

Reducing the proof size. We propose two substantial changes to the protocol above. The first one is that instead of sending $\mathbf{w}$ in the clear, we commit to it by computing $\hat{\mathbf{w}} := \mathbf{G}_r^{-1}(\mathbf{w})$ and outputting $\mathbf{v} := \mathbf{D}\hat{\mathbf{w}}$, where $\mathbf{D} \in \mathcal{R}_q^{n \times r}$ is a uniformly random matrix. Then, in the final round, the prover reveals $\hat{\mathbf{w}}$, together with $(\hat{\mathbf{t}}_i)_{i \in [r]}, \mathbf{z}$. The verifer checks whether $\hat{\mathbf{w}}$ is short, $\mathbf{D}\hat{\mathbf{w}} \overset{?}{=} \mathbf{v}$, and if conditions in (2) hold for the reconstructed $\mathbf{w} := \mathbf{G}_r\hat{\mathbf{w}}$. The modified three-round protocol is summarized in Figure 1.

At a first sight, this modification gives no advantage, or even worse, makes the protocol less efficient. Indeed, instead of sending $\mathbf{w}$, the prover outputs the commitment $\mathbf{v}$, together with opening $\hat{\mathbf{w}}$ which has the same bit-length as $\mathbf{w}$. The key observation here is that the verification conditions can be described as a standard lattice-type statement, i.e. checking whether $\hat{\mathbf{w}}, \hat{\mathbf{t}} := (\hat{\mathbf{t}}_i)_{i \in [r]}, \mathbf{z}$ have small norm and they satisfy the following linear relation over $\mathcal{R}_q$:

$$\begin{bmatrix} \mathbf{D} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{b}^\mathsf{T}\mathbf{G}_r & \mathbf{0} & \mathbf{0} \\ \mathbf{c}^\mathsf{T}\mathbf{G}_r & \mathbf{0} & -\mathbf{a}^\mathsf{T}\mathbf{G}_m \\ \mathbf{0} & \mathbf{c}^\mathsf{T} \otimes \mathbf{G}_n & -\mathbf{A} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{w}} \\ \hat{\mathbf{t}} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{u} \\ y \\ 0 \\ \mathbf{0} \end{bmatrix}. \tag{3}$$

Therefore, instead of sending $\hat{\mathbf{w}}, \hat{\mathbf{t}}, \mathbf{z}$ in the clear, we apply the LaBRADOR [BS23] proof system to prove knowledge of short $\hat{\mathbf{w}}, \hat{\mathbf{t}}, \mathbf{z}$ which satisfy (3). This results in succinct proof sizes comparable with LaBRADOR, i.e. asymptotically $\mathsf{poly}(\lambda, \log N)$ bits. More importantly, we notice that by running the LaBRADOR sub-protocol on an instance and witness of size $O_\lambda(r+m) = O_\lambda(\sqrt{N})$ for a suitable choice of r and m, our verifier for the polynomial evaluation protocol has sublinear time complexity.

Polynomial evaluation proof. To transform the protocol above into a polynomial evaluation proof[4] over $\mathcal{R}_q$ we use the following (standard) observation. Suppose $N = m \cdot r$ for some $m, r \geqslant 1$. Then, for any $f = \sum_{i=0}^{N-1} f_i \mathsf{X}^i \in \mathcal{R}_q^{<N}[\mathsf{X}]$, and any evaluation point $x \in \mathcal{R}_q$ we have:

$$f(x) = \mathbf{a}^\mathsf{T} \left[\mathbf{f}_1 | \cdots | \mathbf{f}_r\right] \mathbf{b} \quad \text{where} \quad \begin{aligned} \mathbf{a}^\mathsf{T} &:= \left[1 \ x \ x^2 \ \cdots \ x^{m-1}\right] \\ \mathbf{b}^\mathsf{T} &:= \left[1 \ x^m \ (x^m)^2 \ \cdots \ (x^m)^{r-1}\right] \\ \mathbf{f}_i^\mathsf{T} &:= \left[f_{(i-1)m} \ f_{(i-1)m+1} \ \cdots \ f_{im-1}\right] \text{ for } i \in [r] \end{aligned} .$$

Hence, we can invoke the protocol described above to prove statements of the form $f(x) = y$ over $\mathcal{R}_q$. Finally, we apply the generic transformation from [AFLN24] to convert our construction into a polynomial commitment scheme over a finite field $\mathbb{F}_q$.

[4] In a similar fashion we can also construct bivariate polynomial commitments, which are used in, e.g. Sonic [MBKM19].

2 Preliminaries

2.1 Notation

Let q be an odd prime. Denote $\mathbb{Z}_q$ to be the ring of integers modulo q. 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 $T \cdot \mathsf{poly}(\lambda)$. 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 $\mathsf{negl}(\lambda)$ to denote an unspecified negligible function.

For a power of two d and a positive integer q, denote $\mathcal{R}$ and $\mathcal{R}_q$ 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 $\mathcal{R}$ or $\mathcal{R}_q$ and bold lower-case (resp. upper-case) letters represent column vectors (resp. matrices) with coefficients in $\mathcal{R}$ or $\mathcal{R}_q$. For $y = \sum_{i=0}^{d-1} y_i \cdot X^i \in \mathcal{R}$, we write $\mathsf{ct}(y) := y_0 \in \mathbb{Z}$ to denote the constant term of y.

We define $r' = r \bmod^{\pm} q$ to be the unique element r' in the range $-\frac{q-1}{2} \leqslant r' \leqslant \frac{q-1}{2}$ such that $r' = r \bmod q$. We also denote $r' = r \bmod^{+} q$ to be the unique element r' in the range $0 \leqslant r' < q$ 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 $w = w_0 + w_1 X + \ldots + w_{d-1} X^{d-1} \in \mathcal{R}$ as follows:

$$\|w\|_\infty = \max_j \|w_j\|_\infty, \quad \|w\|_p = \sqrt[p]{\|w_0\|_\infty^p + \ldots + \|w_{d-1}\|_\infty^p}.$$

If $\mathbf{w} = (w_1, \ldots, w_m) \in \mathcal{R}^k$, then

$$\|\mathbf{w}\|_\infty = \max_j \|w_j\|_\infty, \quad \|\mathbf{w}\|_p = \sqrt[p]{\|w_1\|^p + \ldots + \|w_k\|^p}.$$

By default, $\|\mathbf{w}\| := \|\mathbf{w}\|_2$. Similarly, we define the norms for vectors over $\mathbb{Z}_q$.

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

Lemma 2.1 ([LS18]). *Let $q \equiv 5 \pmod 8$ be a prime. Then, any $f \in \mathcal{R}_q$ which satisfies either $0 < \|f\|_\infty < \frac{1}{\sqrt{2}} q^{1/2}$ or $0 < \|f\| < q^{1/2}$ has an inverse in $\mathcal{R}_q$.*

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

Let $b, n \in \mathbb{N}$. We define the gadget vector $\mathbf{g}_b^\mathsf{T} := \begin{bmatrix} 1 & b & b^2 & \cdots & b^\delta \end{bmatrix}$, where $\delta = \lfloor \log_b q \rfloor$. Then, the matrix matrix $\mathbf{G}_{b,n}$ is defined as $\mathbf{G}_{b,n} := \mathbf{I}_n \otimes \mathbf{g}_b^\mathsf{T}$. Conversely, we define $\mathbf{G}_{b,n}^{-1} : \mathcal{R}_q^{n \times m} \to \mathcal{R}_q^{\delta n \times m}$ to be the inverse function which decomposes each entry w.r.t. base $b \geqslant 2$. Clearly, for any $\mathbf{t} \in \mathcal{R}_q^n$, we have

$$\mathbf{G}\mathbf{G}_{b,n}^{-1}(\mathbf{t}) = \mathbf{t} \quad \text{and} \quad \|\mathbf{G}_{b,n}^{-1}(\mathbf{t})\|_\infty \leqslant \frac{b}{2}.$$

Next, we recall the standard Module-SIS (MSIS) problem [LS15].

Definition 2.2 (Module-SIS). *Let $q = q(\lambda)$, $n = n(\lambda)$, $m = m(\lambda)$, $\beta = \beta(\lambda)$ and $d = d(\lambda)$. We say that the $\mathsf{MSIS}_{n,m,q,\beta}$ assumption holds if for any PPT adversary $\mathcal{A}$, the following holds:*

$$\Pr\left[\mathbf{Az} = \mathbf{0} \wedge 0 < \|\mathbf{z}\| \leqslant \beta \;\middle|\; \begin{array}{c}\mathbf{A} \leftarrow \mathcal{R}_q^{n\times m} \\ \mathbf{z} \leftarrow \mathcal{A}(\mathbf{A})\end{array}\right] = \mathsf{negl}(\lambda) \ .$$

2.2 Interactive Proofs

Let $\mathsf{R} \subseteq \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^*$ be a ternary relation. For a triple $(\mathsf{pp}, \mathbb{x}, \mathbb{w}) \in \mathsf{R}$, we call pp the public parameters, $\mathbb{x}$ is a statement and $\mathbb{w}$ is a witness for $\mathbb{x}$ w.r.t. pp. We denote $\mathsf{R}(\mathsf{pp}, \mathbb{x}) = \{\mathbb{w} : \mathsf{R}(\mathsf{pp}, \mathbb{x}, \mathbb{w}) = 1\}$. In this work, we only consider NP relations R for which a witness w can be verified in time $\mathsf{poly}(|\mathsf{pp}|, |\mathbb{x}|)$ for all $(\mathsf{pp}, \mathbb{x}, \mathbb{w}) \in \mathsf{R}$.

An interactive proof system $\mathit{\Pi} = (\mathcal{S}, \mathcal{P}, \mathcal{V})$ 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. We write $(tr, b) \leftarrow \langle \mathcal{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w}), \mathcal{V}(\mathsf{pp}, \mathbb{x})\rangle$ for running $\mathcal{P}$ and $\mathcal{V}$ on inputs $\mathsf{pp}, \mathbb{x}, \mathbb{w}$ and $\mathsf{pp}, \mathbb{x}$ respectively and getting communication transcript tr and the verifier's decision bit b. We use the convention that $b = 0$ means reject and $b = 1$ means accept the prover's claim of knowing $\mathbb{w}$ such that $(\mathbb{x}, \mathbb{w}) \in R$. Unless stated otherwise, we will assume that the first and the last message are sent from a prover. Hence, the protocol between $\mathcal{P}$ and $\mathcal{V}$ has an odd number of rounds. Further, we say a protocol is *public coin* if the verifier's challenges are chosen uniformly at random independently of the prover's messages.

Definition 2.3 (Completeness). *A proof system $\mathit{\Pi} = (\mathcal{S}, \mathcal{P}, \mathcal{V})$ for the relation R satisfies completeness with completeness error $\epsilon(\cdot)$ if for all adversaries $\mathcal{A}$,*

$$\Pr\left[b = 0 \wedge (\mathsf{pp}, \mathbb{x}, \mathbb{w}) \in \mathsf{R} \;\middle|\; \begin{array}{r}\mathsf{pp} \leftarrow \mathcal{S}(1^\lambda) \\ (\mathbb{x}, \mathbb{w}) \leftarrow \mathcal{A}(\mathsf{pp}) \\ (tr, b) \leftarrow \langle \mathcal{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w}), \mathcal{V}(\mathsf{pp}, \mathbb{x})\rangle\end{array}\right] = \epsilon(\lambda) + \mathsf{negl}(\lambda).$$

If $\epsilon(\cdot)$ is a zero-function then we say $\mathit{\Pi}$ satisfies perfect completeness.

Definition 2.4 (Knowledge Soundness). *A proof system $\mathit{\Pi} = (\mathcal{S}, \mathcal{P}, \mathcal{V})$ for the relation R is knowledge sound with knowledge error $\varepsilon(\lambda)$ if there exists an expected PPT extractor $\mathcal{E}$ such that for any stateful PPT adversary $\mathcal{P}^*$:*

$$\Pr\left[b = 1 \wedge (\mathsf{pp}, \mathbb{x}, \mathbb{w}) \notin \mathsf{R} \;\middle|\; \begin{array}{r}\mathsf{pp} \leftarrow \mathcal{S}(1^\lambda) \\ (\mathbb{x}, \mathsf{st}^*) \leftarrow \mathcal{P}^*(\mathsf{pp}) \\ (tr, b) \leftarrow \langle \mathcal{P}^*(\mathsf{pp}, \mathbb{x}, \mathsf{st}^*), \mathcal{V}(\mathsf{pp}, \mathbb{x})\rangle \\ \mathbb{w} \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{pp}, \mathbb{x})\end{array}\right] = \varepsilon(\lambda) + \mathsf{negl}(\lambda).$$

Here, the extractor $\mathcal{E}$ has a black-box oracle access to the (malicious) prover $\mathcal{P}^*$ and can rewind it to any point in the interaction.

To prove knowledge soundness, we will show that our protocols satisfy coordinate-wise special soundness (CWSS) defined in [FMN23]. Namely, let $\mathcal{C}$ be a finite set and $\ell \in \mathbb{N}$. For any two vectors $\vec{x} := (x_1, \dots, x_\ell), \vec{y} := (y_1, \dots, y_\ell) \in \mathcal{C}^\ell$, define the following relation "$\equiv_i$" for fixed $i \in [\ell]$ as:

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

That is, vectors $\vec{x}$ and $\vec{y}$ have the same values in all coordinates apart from the i-th one. Next, we define the set

$$\mathsf{SS}(\mathcal{C}, \ell) := \left\{ (\vec{x}_1, \dots, \vec{x}_{\ell+1}) \in \mathcal{C}^{\ell+1} : \begin{array}{l} \exists\ k \in [\ell+1],\ \forall\ i \in [\ell], \\ \exists\ j \in [\ell+1]\backslash\{k\},\ \vec{x}_k \equiv_i \vec{x}_j \end{array} \right\} .$$

We are ready to define the notion of coordinate-wise special soundness for three-round protocols (the general definition for multi-round protocols is not needed here).

Definition 2.5 (CWSS for three-round protocols). *Let $\Pi = (\mathcal{S}, \mathcal{P}, \mathcal{V})$ be a public-coin three-round interactive proof system for relation* R*, and suppose the challenge space of $\mathcal{V}$ is $\mathcal{C}^\ell$. We say that Π is ℓ-coordinate-wise special sound if there exists a polynomial time algorithm that on input public parameters* pp*, statement $\mathbb{x}$ and $\ell + 1$ accepting transcripts $(a, \vec{c}_i, z_i)_{i \in [\ell+1]}$, with $\{\vec{c}_1, \dots, \vec{c}_{\ell+1}\} \in \mathsf{SS}(\mathcal{C}, \ell)$ and common first message a, outputs a witness $\mathbb{w} \in \mathsf{R}(\mathsf{pp}, \mathbb{x})$.*

It was shown in [FMN23] that coordinate-wise special sound protocols are knowledge sound [5].

Lemma 2.6 (Lemma 2.31 of [FMN23]). *Let $\Pi = (\mathcal{S}, \mathcal{P}, \mathcal{V})$ be public-coin three-round protocol for relation* R *with the challenge space of $\mathcal{C}$. If Π is ℓ-coordinate-wise special sound, then it is knowledge sound with knowledge error $\ell/|\mathcal{C}|$.*

2.3 Polynomial Commitment Scheme

Polynomial commitment schemes can be seen as standard commitments to polynomials f (e.g. by committing to the coefficients of f) equipped with the ability to prove evaluations of f. We define polynomial commitments in the interactive setting. Due to the slack occurring in the lattice setting, we define the slack space SL and fix a (public) identity element $\mathsf{e} \in \mathsf{SL}$.

Definition 2.7. *Let* $\mathsf{PCS} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Eval})$ *be a tuple of algorithms.* PCS *is a polynomial commitment scheme over a ring R with degree bound N if:*

- $\mathsf{Setup}(1^\lambda) \rightarrow \mathsf{pp}$ *takes a security parameter λ (specified in unary) and outputs public parameters* pp.
- $\mathsf{Commit}(\mathsf{pp}, f) \rightarrow (C, \mathsf{st})$ *takes public parameters* pp *a message $f \in R^{<N}[\mathsf{X}]$ and outputs a commitment C and decommitment state* st.

[5] See [FMN23, Lemma 2.32] for the non-interactive version in the random oracle model.

- $\mathsf{Open}(\mathsf{pp}, C, f, \mathsf{st}, c) \to 0/1$ *takes public parameters* pp*, a commitment* C*, a message* $f \in R^{<N}[\mathsf{X}]$*, a decommitment state* st *and a relaxation factor* $c \in \mathsf{SL}$ *and outputs a bit indicating whether* C *is a valid commitment to* f *under* pp*. We implicitly assume that if* $c \notin \mathsf{SL}$ *then* Open *outputs* 0*.*
- $\mathsf{Eval} := (\mathsf{Eval}.\mathcal{P}, \mathsf{Eval}.\mathcal{V})$ *is a pair of probabilistic polynomial-time algorithms. Here* $\mathsf{Eval}.\mathcal{P}(\mathsf{pp}, (C, x, y), (f, \mathsf{st}))$ *is the evaluation prover,* $\mathsf{Eval}.\mathcal{V}(\mathsf{pp}, (C, x, y))$ *is the evaluation verifier.*

An interactive polynomial commitment scheme can be transformed into a non-interactive one using the Fiat-Shamir transformation [FS86].

We require that the polynomial commitment scheme satisfies evaluation completeness, weak binding and knowledge soundness.

Definition 2.8 (Evaluation Completeness). *We say that a polynomial commitment scheme* $\mathsf{PCS} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Eval})$ *satisfies evaluation completeness with completeness error* $\epsilon(\cdot)$ *if for every polynomial* $f \in R^{<N}[\mathsf{X}]$ *and any evaluation point* $x \in R$*:*

$$\Pr\left[\begin{array}{l|r} & \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{Open}(\mathsf{pp}, C, f, \mathsf{st}, \mathsf{e}) = 0 & C, \mathsf{st} \leftarrow \mathsf{Commit}(\mathsf{pp}, f) \\ \vee b = 0 & \mathbb{x} := (C, x, f(x)), \mathbb{w} := (f, \mathsf{st}) \\ & (tr, b) \leftarrow \langle \mathsf{Eval}.\mathcal{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w}), \mathsf{Eval}.\mathcal{V}(\mathsf{pp}, \mathbb{x}) \rangle \end{array}\right] = \epsilon(\lambda) + \mathsf{negl}(\lambda).$$

Definition 2.9 (Weak Binding). *A polynomial commitment scheme* $\mathsf{PCS} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Eval})$ *satisfies weak binding if for every PPT adversary* $\mathcal{A}$*:*

$$\Pr\left[\begin{array}{l|r} f \neq f' \wedge f, f' \in R^{<N}[\mathsf{X}] \wedge & \\ \mathsf{Open}(\mathsf{pp}, C, f, \mathsf{st}, c) & \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ = \mathsf{Open}(\mathsf{pp}, C, f', \mathsf{st}', c') = 1 & (C, (f, \mathsf{st}, c), (f', \mathsf{st}', c')) \leftarrow \mathcal{A}(\mathsf{pp}) \end{array}\right] = \mathsf{negl}(\lambda).$$

Definition 2.10 (Knowledge Soundness). *We say that a polynomial commitment scheme* $\mathsf{PCS} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Eval})$ *is knowledge sound with knowledge error* ε *if for all stateful PPT adversaries* $\mathcal{P}^*$*, there exists an expected PPT extractor* $\mathcal{E}$ *such that*

$$\Pr\left[\begin{array}{l|r} & \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{Open}(\mathsf{pp}, C, f, \mathsf{st}, c) \neq 1 \vee f(x) \neq y) & \mathbb{x} := (C, x, y), \mathsf{st}^* \leftarrow \mathcal{P}^*(\mathsf{pp}) \\ \wedge b = 1 & (tr, b) \leftarrow \langle \mathcal{P}^*(\mathsf{pp}, \mathbb{x}, \mathsf{st}^*), \mathcal{V}(\mathsf{pp}, \mathbb{x}) \rangle \\ & (f, \mathsf{st}, c) \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{pp}, \mathbb{x}) \end{array}\right] = \varepsilon(\lambda) + \mathsf{negl}(\lambda).$$

Here, the extractor $\mathcal{E}$ *has a black-box oracle access to the (malicious) prover* $\mathcal{P}^*$ *and can rewind it to any point in the interaction.*

2.4 Principal Relation

We recall the principal relation (alternatively called a dot-product relation) defined in [BS23]. The relation is characterised by the rank $n \geqslant 1$, multiplicity

$r \geqslant 1$ and the norm bound $\beta > 0$. The statement is a triple $(\mathcal{F}, \mathcal{F}', \beta)$, where $\mathcal{F}$ and $\mathcal{F}'$ are families of functions $f : \mathcal{R}_q^n \times \ldots \times \mathcal{R}_q^n \to \mathcal{R}_q$ of the form

$$f(\mathbf{s}_1, \ldots, \mathbf{s}_r) = \sum_{i,j=1}^{r} a_{i,j} \langle \mathbf{s}_i, \mathbf{s}_j \rangle + \sum_{i=1}^{r} \langle \phi_i, \mathbf{s}_i \rangle - b$$

for $a_{i,j}, b \in \mathcal{R}_q$, $\phi_i \in \mathcal{R}_q^n$, and $\beta \geqslant 0$. Then, a valid witness is a sequence of vectors $\mathbf{s}_1, \ldots, \mathbf{s}_r \in \mathcal{R}_q^n$ which satisfy:

$$\begin{aligned} & f(\mathbf{s}_1, \ldots, \mathbf{s}_r) = 0 \quad \forall f \in \mathcal{F} \\ & \mathsf{ct}(f'(\mathbf{s}_1, \ldots, \mathbf{s}_r)) = 0 \quad \forall f' \in \mathcal{F}' \\ & \sum_{i=1}^{r} \|\mathbf{s}_i\|^2 \leqslant \beta. \end{aligned}$$

It was shown in [BS23, Section 7] that the Rank-1 Constraint System (R1CS) can be reduced to the principal relation.

2.5 Inner and Outer Commitments

We recall the inner and outer commitments from [BS23], which will be the base of our polynomial commitment. Let $n, m, r, b, q \in \mathbb{N}$ and set $\delta := \lfloor \log_b q \rfloor$. Denote $\bar{\beta}, \bar{\gamma}, \bar{\kappa} > 0$ as the security-related norm bounds. Let $(\mathbf{A} \in \mathcal{R}_q^{n \times m}, \mathbf{B} \in \mathcal{R}_q^{n \times n\delta r})$ be the public parameters.

Suppose we want to commit to a matrix $\mathbf{S} \in \mathcal{R}_q^{m \times r}$, which can be represented as r column vectors $\mathbf{s}_1, \ldots, \mathbf{s}_r \in \mathcal{R}_q^m$. The *inner commitments* are the r vectors $\mathbf{t}_i := \mathbf{A}\mathbf{s}_i \in \mathcal{R}_q^n$. Then, the *outer commitment* $\mathbf{u}$ is generated by computing

$$\mathbf{u} := \mathbf{B} \begin{bmatrix} \hat{\mathbf{t}}_1 \\ \vdots \\ \hat{\mathbf{t}}_r \end{bmatrix} \in \mathcal{R}_q^n, \quad \text{where} \quad \hat{\mathbf{t}}_i := \mathbf{G}_{b,n}^{-1}(\mathbf{t}_i) \text{ for } i \in [r]. \tag{4}$$

The decommitment state consists of $(\hat{\mathbf{t}}_i)_{i \in [r]}$. A *weak* opening for the commitment $\mathbf{u}$ is a tuple $(\mathbf{s}_i, \hat{\mathbf{t}}_i, c_i)_{i \in [r]}$, which satisfies all the following conditions

$$\begin{aligned} & \forall i \in [r] : \quad \|c_i \cdot \mathbf{s}_i\| \leqslant \bar{\beta}, \quad \|c_i\|_1 \leqslant \bar{\kappa}, \quad c_i \in \mathcal{R}_q^\times, \quad \mathbf{A}\mathbf{s}_i = \mathbf{G}_{b,n}\hat{\mathbf{t}}_i \\ & \mathbf{B} \begin{bmatrix} \hat{\mathbf{t}}_1 \\ \vdots \\ \hat{\mathbf{t}}_r \end{bmatrix} = \mathbf{u} \quad \text{and} \quad \left\| \begin{bmatrix} \hat{\mathbf{t}}_1 \\ \vdots \\ \hat{\mathbf{t}}_r \end{bmatrix} \right\| \leqslant \bar{\gamma}. \end{aligned}$$

Next, we show that the commitment scheme described above satisfies binding with respect to weak openings under Module-SIS assumption [ALS20].

Lemma 2.11 (Weak Binding). *There is a deterministic algorithm, that given two weak openings $(\mathbf{s}_i, \hat{\mathbf{t}}_i, c_i)_{i \in [r]}$ and $(\mathbf{s}'_i, \hat{\mathbf{t}}'_i, c'_i)_{i \in [r]}$ for the commitment $\mathbf{u} \in \mathcal{R}_q^n$ such that $\mathbf{s}_j \neq \mathbf{s}'_j$ for some $j \in [r]$, outputs a vector $\mathbf{z} \in \mathcal{R}_q^{m+n\delta r}$ such that $[\mathbf{A} \mid \mathbf{B}]\mathbf{z} = \mathbf{0}$ and $0 < \|\mathbf{z}\| \leqslant \max(4\bar{\kappa}\bar{\beta}, 2\bar{\gamma})$.*

Proof. Note that if $\hat{\mathbf{t}}_i \neq \hat{\mathbf{t}}'_i$ for some $i \in [r]$, then we have automatically found a short, non-zero solution $\mathbf{z}_B$ for the matrix $\mathbf{B}$ of norm at most $2\bar{\gamma}$. Suppose this is not the case. In particular, we have

$$\mathbf{A}\mathbf{s}_j = \mathbf{G}_{b,n}\hat{\mathbf{t}}_j = \mathbf{G}_{b,n}\hat{\mathbf{t}}'_j = \mathbf{A}\mathbf{s}'{}_j.$$

Although $\mathbf{s}_j - \mathbf{s}'_j \neq \mathbf{0}$ is not short, we know that

$$\|c_j c'_j(\mathbf{s}_j - \mathbf{s}'_j)\| \leqslant \|c'_j(c_j\mathbf{s}_j)\| + \|c_j(c'_j\mathbf{s}'_j)\| \leqslant 2\bar{\kappa}\bar{\beta}.$$

Finally, since both c_j, c'_j are invertible over $\mathcal{R}_q$, we deduce that $\mathbf{z}_A := c_j c'_j(\mathbf{s}_j - \mathbf{s}'_j)$ is a short non-zero solution for $\mathbf{A}$. We conclude the proof by combining the two cases. □

3 Proofs of Quadratic Relations with Sublinear Verification

In this section, we propose a simple proof of knowledge of a commitment opening which satisfies certain quadratic relations. More concretely, using the notation from Section 2.5 we consider a relation:

$$\mathsf{R}_{b_0,b_1} := \left\{ \left(\begin{array}{c} (\mathbf{A},\mathbf{B},\mathbf{D}), \\ (\mathbf{a},\mathbf{b},\mathbf{u},y), \\ ((\mathbf{s}_i)_{i\in[r]}, \hat{\mathbf{t}} = (\hat{\mathbf{t}}_i)_{i\in[r]}) \end{array} \right) \middle| \begin{array}{c} \forall i \in [r], \mathbf{A}\mathbf{s}_i = \mathbf{G}_{b_1,n}\hat{\mathbf{t}}_i; \\ \mathbf{B}\hat{\mathbf{t}} = \mathbf{u}; \quad \mathbf{a}^\intercal \left[\mathbf{s}_1|\cdots|\mathbf{s}_r\right]\mathbf{b} = y; \\ \forall i \in [r], \|\mathbf{s}_i\|_\infty \leqslant \frac{b_0}{2}; \quad \|\hat{\mathbf{t}}\|_\infty \leqslant \frac{b_1}{2} \end{array} \right\}. \quad (5)$$

Here, width of the matrix $\mathbf{G}_{b_1,n}$ is $\delta \cdot n$, where $\delta := \lfloor \log_{b_1} q \rfloor$. Matrix $\mathbf{D} \in \mathcal{R}_q^{n \times \delta r}$ has a role of an additional commitment key, used to commit to various prover messages in order to preserve succinctness.

3.1 Simple Protocol

The three-round protocol is presented in Figure 1. As for the security analysis, we focus on completeness and coordinate-wise special soundness.

Public parameters: $\mathbf{A} \in \mathcal{R}_q^{n \times m}, \mathbf{B} \in \mathcal{R}_q^{n \times n\delta r}, \mathbf{D} \in \mathcal{R}_q^{n \times \delta r}$
Witness: $(\mathbf{s}_i \in \mathcal{R}_q^m)_{i \in [r]}, \hat{\mathbf{t}} = (\hat{\mathbf{t}}_i)_{i \in [r]} \in \mathcal{R}_q^{n\delta r}$
Statement: $\mathbf{a} \in \mathcal{R}_q^m, \mathbf{b} \in \mathcal{R}_q^r, \mathbf{u} \in \mathcal{R}_q^n, y \in \mathcal{R}_q$,

Prover		Verifier
$\mathbf{w}^\intercal := \mathbf{a}^\intercal \left[\mathbf{s}_1	\cdots	\mathbf{s}_r\right] \in \mathcal{R}_q^r$ $\hat{\mathbf{w}} := \mathbf{G}_{b_1,r}^{-1}(\mathbf{w}) \in \mathcal{R}_q^{\delta r}$ $\mathbf{v} := \mathbf{D}\hat{\mathbf{w}} \in \mathcal{R}_q^n$	
	$\xrightarrow{\mathbf{v}}$	$\mathbf{c} = (c_1, \ldots, c_r) \leftarrow \mathcal{C}^r$
	$\xleftarrow{\mathbf{c}}$
$\mathbf{z} := \left[\mathbf{s}_1	\cdots	\mathbf{s}_r\right] \mathbf{c}$	
	$\xrightarrow{\hat{\mathbf{w}}, \hat{\mathbf{t}}, \mathbf{z}}$	Accept iff:

1. $\left\| \begin{bmatrix} \hat{\mathbf{w}} \\ \hat{\mathbf{t}} \\ \mathbf{z} \end{bmatrix} \right\| \leqslant \frac{1}{2}\sqrt{b_1^2(n+1)\delta r d + (r\kappa b_0)^2 md}$

2. $\begin{bmatrix} \mathbf{D} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{b}^\intercal \mathbf{G}_{b_1,r} & \mathbf{0} & \mathbf{0} \\ \mathbf{c}^\intercal \mathbf{G}_{b_1,r} & \mathbf{0} & -\mathbf{a}^\intercal \\ \mathbf{0} & \mathbf{c}^\intercal \otimes \mathbf{G}_{b_1,n} & -\mathbf{A} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{w}} \\ \hat{\mathbf{t}} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{u} \\ y \\ 0 \\ \mathbf{0} \end{bmatrix}$

Fig. 1: Proof of knowledge for the relation R_{b_0,b_1} in (5). Here, $\delta := \lfloor \log_{b_1} q \rfloor$.

Lemma 3.1 (Completeness). *The protocol in Figure 1 for relation* R_{b_0,b_1} *satisfies perfect completeness.*

Proof. We start with the norm check. Since all coefficients of $\hat{\mathbf{t}}$ and $\hat{\mathbf{w}}$ are at most b in the absolute value, we have $\|\hat{\mathbf{t}}\| \leqslant \frac{b_1}{2}\sqrt{n\delta r d}$ and $\|\hat{\mathbf{w}}\| \leqslant \frac{b_1}{2}\sqrt{\delta r d}$. Combining with $\|\mathbf{z}\|_\infty \leqslant \sum_{i=1}^r \|c_i \cdot \mathbf{s}_i\|_\infty \leqslant r\kappa b_0/2$, this yields the first verification check. As for the algebraic equations, directly from the relation $\mathsf{R}_{\beta,b}$ we have the outer-commitment equation $\mathbf{B}\hat{\mathbf{t}} = \mathbf{u}$. Also, by construction $\mathbf{D}\hat{\mathbf{w}} = \mathbf{v}$. Next, we obtain

$$\mathbf{b}^\intercal \mathbf{G}_{b,r}\hat{\mathbf{w}} = \mathbf{b}^\intercal \mathbf{w} = \mathbf{w}^\intercal \mathbf{b} = \mathbf{a}^\intercal \left[\mathbf{s}_1 | \cdots | \mathbf{s}_r\right] \mathbf{b} = y$$

and

$$\mathbf{c}^\intercal \mathbf{G}_{b,r}\hat{\mathbf{w}} = \mathbf{c}^\intercal \mathbf{w} = \mathbf{w}^\intercal \mathbf{c} = \mathbf{a}^\intercal \left[\mathbf{s}_1 | \cdots | \mathbf{s}_r\right] \mathbf{c} = \mathbf{a}^\intercal \mathbf{z}.$$

Finally,

$$(\mathbf{c}^\intercal \otimes \mathbf{G}_{b,r})\hat{\mathbf{t}} = \sum_{i=1}^{r} c_i \mathbf{G}_{b,r}\hat{\mathbf{t}}_i = \sum_{i=1}^{r} c_i \mathbf{A}\mathbf{s}_i = \mathbf{A}\mathbf{z}$$

which concludes the proof. □

As standard in lattice-based proof systems, we only manage to extract a relaxed openings (cf. Section 2.5). This corresponds to the following relaxed relation R^*:

$$\mathsf{R}^*_{b_1,\bar{\beta},\bar{\gamma},\bar{\kappa}} := \left\{ \left(\begin{array}{c} (\mathbf{A},\mathbf{B},\mathbf{D}), \\ (\mathbf{a},\mathbf{b},\mathbf{u},y), \\ ((\mathbf{s}_i)_{i\in[r]}, \hat{\mathbf{t}} = (\hat{\mathbf{t}}_i)_{i\in[r]}, (c_i)_{i\in[r]}) \end{array} \right) \middle| \begin{array}{c} \forall i \in [r], \mathbf{A}\mathbf{s}_i = \mathbf{G}_{b_1,n}\hat{\mathbf{t}}_i; \\ \mathbf{B}\hat{\mathbf{t}} = \mathbf{u}; \quad \mathbf{a}^\intercal \left[\mathbf{s}_1|\cdots|\mathbf{s}_r\right]\mathbf{b} = y; \\ \forall i \in [r], \|c_i \cdot \mathbf{s}_i\| \leqslant \bar{\beta}; \|c_i\|_1 \leqslant \bar{\kappa}; c_i \in \mathcal{R}_q^\times; \\ \|\hat{\mathbf{t}}\| \leqslant \bar{\gamma} \end{array} \right\}. \tag{6}$$

Recall that, as shown in Lemma 2.11, the commitment scheme satisfies weak binding under the Module-SIS assumption.

Next, we show that the three-round protocol satisfies coordinate-wise special soundness under the Module-SIS assumption.

Lemma 3.2 (CWSS). *Let* $\bar{\gamma} := \frac{1}{2}\sqrt{b_1^2(n+1)\delta r d + (r\kappa b_0)^2 m d}$, $\bar{\beta} := 2\bar{\gamma}$ *and* $\bar{\kappa} := 2\kappa$. *Suppose that* $\bar{\kappa} < \frac{1}{\sqrt{2}}\sqrt{q}$. *Then, there exists a polynomial time algorithm that on input public parameters* $\mathsf{pp} := (\mathbf{A},\mathbf{B},\mathbf{D})$, *statement* $\mathbb{x} := (\mathbf{a},\mathbf{b},\mathbf{u},y)$ *and* $r+1$ *accepting transcripts*

$$\mathsf{tr}_i := (\mathbf{v}, \mathbf{c}_i, (\hat{\mathbf{w}}_i, \hat{\mathbf{t}}_i, \mathbf{z}_i)) \quad \textit{for } i = 0,1,\ldots,r$$

with $(\mathbf{c}_0,\ldots,\mathbf{c}_r) \in \mathsf{SS}(\mathcal{C}, r)$ *and common first message* $\mathbf{v}$, *either outputs a witness* $\mathbb{w} \in \mathsf{R}^*_{b_1,\bar{\beta},\bar{\gamma},\bar{\kappa}}(\mathsf{pp},\mathbb{x})$, *or* $\mathbf{z} \in \mathcal{R}_q^{(n+1)\delta r}$ *so that* $[\mathbf{B} \mid \mathbf{D}]\mathbf{z} = \mathbf{0}$ *and* $0 < \|\mathbf{z}\| \leqslant \bar{\beta}$.

Proof. Assume without loss of generality that $\mathbf{c}_0$ differs from each $\mathbf{c}_i$ exactly in the i coordinate for $i \in [r]$. First, if for some distinct $i,j \in \{0,1,\ldots,r\}$ we have $\hat{\mathbf{t}}_i \neq \hat{\mathbf{t}}_j$, then we immediately yield a non-zero solution $\mathbf{z} := \hat{\mathbf{t}}_i - \hat{\mathbf{t}}_j$ to $\mathbf{B}$ of norm at most $\bar{\beta}$. Similarly we argue for all $\hat{\mathbf{w}}_i$. Thus, from now on we assume that $\hat{\mathbf{t}} := \hat{\mathbf{t}}_0 = \ldots = \hat{\mathbf{t}}_r$ and $\hat{\mathbf{w}} := \hat{\mathbf{w}}_0 = \ldots = \hat{\mathbf{w}}_r$. For presentation, set $\mathbf{w} := (w_1,\ldots,w_r) = \mathbf{G}_{b_1,r}\hat{\mathbf{w}}$.

Fix $i \in [r]$ and denote $\mathbf{c}_0 := (c_1,\ldots,c_r)$ and $\mathbf{c}_i := (c_1,\ldots,c_{i-1},c_i',c_{i+1},\ldots,c_r)$ where $c_i \neq c_i'$. The L_1 norm of $\bar{c}_i := c_i - c_i'$ is at most $\bar{\kappa} = 2\kappa$ and thus it is invertible over $\mathcal{R}_q$ by Lemma 2.1. Now, define $\bar{\mathbf{s}}_i := (\mathbf{z}_0 - \mathbf{z}_i)/\bar{c}_i$. Clearly, $\|\bar{c}_i \cdot \bar{\mathbf{s}}_i\| \leqslant \bar{\beta}$. Next, from the verification equations for tr_0 and tr_i we have

$$w_i = \frac{(\mathbf{c}_0^\intercal - \mathbf{c}_i^\intercal)\mathbf{w}}{\bar{c}_i} = \frac{\mathbf{a}^\intercal(\mathbf{z}_0 - \mathbf{z}_1)}{\bar{c}_i} = \mathbf{a}^\intercal\bar{\mathbf{s}}_i.$$

In particular, combined with $\mathbf{b}^\intercal\mathbf{w} = y$, we obtain

$$\mathbf{a}^\intercal\left[\bar{\mathbf{s}}_1|\cdots|\bar{\mathbf{s}}_r\right]\mathbf{b} = y.$$

Moreover, by parsing $\hat{\mathbf{t}} := (\hat{\mathbf{t}}^{(1)},\ldots,\hat{\mathbf{t}}^{(r)})$, where each $\hat{\mathbf{t}}^{(j)} \in \mathcal{R}_q^{n\delta}$, we have

$$\mathbf{G}_{b_1,n}\hat{\mathbf{t}}^{(i)} = \mathbf{G}_{b_1,n}\left(\frac{\mathbf{c}_0^\intercal - \mathbf{c}_i^\intercal}{\bar{c}_i} \otimes \mathbf{I}_{n\delta}\right)\hat{\mathbf{t}} = \left(\frac{\mathbf{c}_0^\intercal - \mathbf{c}_i^\intercal}{\bar{c}_i}\right) \otimes \mathbf{G}_{b_1,n}\hat{\mathbf{t}} = \mathbf{A}\left(\frac{\mathbf{z}_0 - \mathbf{z}_i}{\bar{c}_i}\right) = \mathbf{A}\bar{\mathbf{s}}_i.$$

Therefore, we conclude that

$$\mathbb{w} := \left((\bar{\mathbf{s}}_i)_{i\in[r]}, (\hat{\mathbf{t}}^{(i)})_{i\in[r]}, (\bar{c}_i)_{i\in[r]}\right)$$

belongs to $\mathsf{R}^*_{b_1,\bar{\beta},\bar{\gamma},\bar{\kappa}}(\mathsf{pp},\mathbb{x})$. □

Efficiency. The communication complexity from the prover's side can be bounded by

$$nd\lceil \log q \rceil + (n+1)\delta r d \lceil \log(2b_1) \rceil + md\lceil \log(2r\kappa b_0) \rceil.$$

The prover's running time is $O(r(m+n+\delta))$ operations over $\mathcal{R}_q$. On the other hand, the verifier's time complexity is $O(n \cdot (n\delta r + m))$ operations over $\mathcal{R}_q$. Since the witness size is $m \cdot r$ elements in $\mathcal{R}_q$, we deduce that the verifier runtime is sublinear.

3.2 Batching

In this section we consider a full generalisation of the relation in Equation (5). Namely, let $k \geqslant 1$ and fix $\mathbf{a}_j \in \mathcal{R}_q^m, \mathbf{b}_j \in \mathcal{R}_q^r$ for $j \in [k]$. Next, consider any k positive integers $L_1, \ldots, L_k$. In the context of polynomial commitments, k is the number of *distinct* evaluation points, and for the j-th point, we will prove L_j polynomial evaluations. Clearly, the previous protocol corresponds to the case $k = 1$ and $L_1 = 1$.

We focus on proving knowledge of short vectors $(\mathbf{s}_{j,\iota,i})_{j \in [k], \iota \in [L_j], i \in [r]}$, such that

$$\mathbf{a}_j^\intercal \left[\mathbf{s}_{j,\iota,1} | \cdots | \mathbf{s}_{j,\iota,r}\right] \mathbf{b}_j = y_{j,\iota} \quad \forall j \in [k], \iota \in [L_j],$$

where all $y_{j,\iota}$ are public. By including the commitment opening relation, we define (for presentation we fix the indices $j \in [k], \iota \in [L_j]$ and $i \in [r]$):

$$\mathsf{BR}_{b_0,b_1} := \left\{ \left(\begin{array}{c} (\mathbf{A}, (\mathbf{B}_j, \mathbf{D}_j)_j), \\ ((\mathbf{a}_j, \mathbf{b}_j)_j, \mathbf{u}, (y_{j,\iota})_{j,\iota}), \\ ((\mathbf{s}_{j,\iota,i})_{j,\iota,i}, (\hat{\mathbf{t}}_j = (\hat{\mathbf{t}}_{j,\iota,i})_{\iota,i})_j) \end{array} \right) \middle| \begin{array}{c} \forall j, \iota, i, \mathbf{A}\mathbf{s}_{j,\iota,i} = \mathbf{G}_{b_1,n}\hat{\mathbf{t}}_{j,\iota,i}; \\ \sum_{j=1}^k \mathbf{B}_j \hat{\mathbf{t}}_j = \mathbf{u}; \\ \forall j, \iota, \mathbf{a}_j^\intercal \left[\mathbf{s}_{j,\iota,1} | \cdots | \mathbf{s}_{j,\iota,r}\right] \mathbf{b}_j = y_{j,\iota}; \\ \forall j, \iota, i, \|\mathbf{s}_{j,\iota,i}\|_\infty \leqslant b_0; \quad \|\hat{\mathbf{t}}_j\|_\infty \leqslant b_1 \end{array} \right\}. \tag{7}$$

As before, width of the matrix $\mathbf{G}_{b_1,n}$ is $\delta \cdot n$ for $\delta := \lfloor \log_{b_1} q \rceil$.

We present a three-round protocol for BR_{b_0,b_1} in Figure 2 and provide an informal description below due to more involved notation. The prover starts by computing for all $j \in [k], \iota \in [L_j]$:

$$\hat{\mathbf{w}}_{j,\iota} := \mathbf{G}_{b_1,r}^{-1}(\mathbf{w}_{j,\iota}) \in \mathcal{R}_q^{\delta r}, \quad \text{where} \quad \mathbf{w}_{j,\iota}^\intercal := \mathbf{a}_j^\intercal \left[\mathbf{s}_{j,\iota,1} | \cdots | \mathbf{s}_{j,\iota,r}\right] \in \mathcal{R}_q^r$$

and sets $\hat{\mathbf{w}}_j^\intercal := \left[\hat{\mathbf{w}}_{j,1}^\intercal | \cdots | \hat{\mathbf{w}}_{j,L_j}^\intercal\right]$ for $j \in [k]$. Finally, it commits to all $\hat{\mathbf{w}}_1, \ldots, \hat{\mathbf{w}}_k$ by sending

$$\mathbf{v} := \mathbf{D}_1 \hat{\mathbf{w}}_1 + \ldots + \mathbf{D}_k \hat{\mathbf{w}}_k$$

to the verifier. Then, $L_1 + \ldots + L_k$ vectors $\mathbf{c}_{1,1}, \ldots, \mathbf{c}_{k,L_k}$ generated uniformly at random from $\mathcal{C}^r$ are sent by the verifier. The prover responds by computing

$$\mathbf{z}_j := \sum_{\iota=1}^{L_j} \left[\mathbf{s}_{j,\iota,1} | \cdots | \mathbf{s}_{j,\iota,r}\right] \mathbf{c}_{j,\iota} \quad \text{for } j = 1, \ldots, k$$

and outputting $(\hat{\mathbf{w}}_1, \ldots, \hat{\mathbf{w}}_k), \hat{\mathbf{t}}, (\mathbf{z}_1, \ldots, \mathbf{z}_k)$. The verifier then checks whether

$$\left\| \begin{bmatrix} (\hat{\mathbf{w}}_j)_{j\in[k]} \\ (\hat{\mathbf{t}}_j)_{j\in[k]} \\ (\mathbf{z}_j)_{j\in[k]} \end{bmatrix} \right\| \leqslant \frac{1}{2}\sqrt{b_1^2(n+1)\delta r \left(\sum_{j=1}^{k} L_j\right) d + (r\kappa b_0)^2 \left(\sum_{j=1}^{k} \ell_j^2\right) md} \tag{8}$$

and

$$\begin{aligned} &\sum_{j=1}^{k} \mathbf{D}_j \hat{\mathbf{w}}_j = \mathbf{v} \\ &\sum_{j=1}^{k} \mathbf{B}_j \hat{\mathbf{t}}_j = \mathbf{u} \\ &\forall j \in [k], \begin{bmatrix} \mathbf{b}_j^\intercal \mathbf{G}_{b_1,r} & & \\ & \ddots & \\ & & \mathbf{b}_j^\intercal \mathbf{G}_{b_1,r} \end{bmatrix} \hat{\mathbf{w}}_j = \begin{bmatrix} y_{j,1} \\ \vdots \\ y_{j,L_j} \end{bmatrix} \\ &\forall j \in [k], \left[\mathbf{c}_{j,1}^\intercal \mathbf{G}_{b_1,r} \cdots \mathbf{c}_{j,L_j}^\intercal \mathbf{G}_{b_1,r}\right] \hat{\mathbf{w}}_j = \mathbf{a}_j^\intercal \mathbf{z}_j \\ &\forall j \in [k], \left[\mathbf{c}_{j,1}^\intercal \otimes \mathbf{G}_{b_1,n} \cdots \mathbf{c}_{j,L_j}^\intercal \otimes \mathbf{G}_{b_1,n}\right] \hat{\mathbf{t}}_j = \mathbf{A}\mathbf{z}_j. \end{aligned} \tag{9}$$

```
Public parameters: $\mathbf{A} \in \mathcal{R}_q^{n\times m}, (\mathbf{B}_j \in \mathcal{R}_q^{n\times n\delta r L_j}, \mathbf{D}_j \in \mathcal{R}_q^{n\times \delta r L_j})_{j\in[k]}$
Witness: $(\mathbf{s}_{j,\iota,i} \in \mathcal{R}_q^{m})_{j\in[k],\iota\in[L_j],i\in[r]}, (\hat{\mathbf{t}}_j \in \mathcal{R}_q^{n\delta r L_j})_{j\in[k]}$
Statement: $(\mathbf{a}_j \in \mathcal{R}_q^{m}, \mathbf{b}_j \in \mathcal{R}_q^{r})_{j\in[k]}, \mathbf{u} \in \mathcal{R}_q^{n}, (y_{j,\iota} \in \mathcal{R}_q)_{j\in[k],\iota\in[L_j]}$,

Prover                                                  Verifier

For $j \in [k]$ :
  For $\iota \in [L_j]$ :
    $\mathbf{w}_{j,\iota}^\intercal := \mathbf{a}_j^\intercal \left[\mathbf{s}_{j,\iota,1} | \cdots | \mathbf{s}_{j,\iota,r}\right] \in \mathcal{R}_q^r$
    $\hat{\mathbf{w}}_{j,\iota} := \mathbf{G}_{b_1,r}^{-1}(\mathbf{w}_{j,\iota}) \in \mathcal{R}_q^{\delta r}$
  $\hat{\mathbf{w}}_j^\intercal := \left[\hat{\mathbf{w}}_{j,1}^\intercal | \cdots | \hat{\mathbf{w}}_{j,L_j}^\intercal\right]$
$\mathbf{v} := \sum_{j=1}^{k} \mathbf{D}_j \hat{\mathbf{w}}_j \in \mathcal{R}_q^n$
                              $\xrightarrow{\mathbf{v}}$
                                                        $\mathbf{c}_{1,1}, \ldots, \mathbf{c}_{k,L_k} \leftarrow \mathcal{C}^r$
                              $\xleftarrow{\mathbf{c}_{1,1}, \ldots, \mathbf{c}_{k,L_k}}$
For $j \in [k]$ :
  $\mathbf{z}_j := \sum_{\iota=1}^{L_j} \left[\mathbf{s}_{j,\iota,1} | \cdots | \mathbf{s}_{j,\iota,r}\right] \mathbf{c}_{j,\iota}$
                              $\xrightarrow{(\hat{\mathbf{w}}_j, \hat{\mathbf{t}}_j, \mathbf{z}_j)_{j\in[k]}}$
                                                        Accept iff (8) and (9) hold
```

Fig. 2: Proof of knowledge for the relation BR_{b_0,b_1} in (7).

Security analysis. We prove completeness and coordinate-wise special soundness.

Lemma 3.3. *The protocol in Figure 2 for relation BR_{b_0,b_1} satisfies perfect completeness.*

Proof. We start with the norm checks. We know that for $j \in [k]$, $\|\hat{\mathbf{w}}_j\|_\infty \leqslant \frac{b_1}{2}$ and $\|\hat{\mathbf{t}}_j\|_\infty \leqslant \frac{b_1}{2}$. Also, $\|\mathbf{z}_j\|_\infty \leqslant r\kappa b_0 L_j/2$. Hence, (8) holds by applying the naive ℓ_∞-to-ℓ_2 inequality. Now, we move on to (9). The first two equations hold trivially. As for the third one, we note that for any $j \in [k]$ and $\iota \in [L_j]$:

$$\mathbf{b}_j^\intercal \mathbf{G}_{b_1,r}\hat{\mathbf{w}}_{j,\iota} = \mathbf{b}_j^\intercal \mathbf{w}_{j,\iota} = \mathbf{w}_{j,\iota}^\intercal \mathbf{b}_j = \mathbf{a}_j^\intercal \left[\mathbf{s}_{j,\iota,1}|\cdots|\mathbf{s}_{j,\iota,r}\right]\mathbf{b}_j = y_{j,\iota}.$$

As for the fourth item:

$$\begin{aligned}\left[\mathbf{c}_{j,1}^\intercal \mathbf{G}_{b_1,r} \cdots \mathbf{c}_{j,L_j}^\intercal \mathbf{G}_{b_1,r}\right]\hat{\mathbf{w}}_j &= \sum_{\iota=1}^{L_j} \mathbf{c}_{j,\iota}^\intercal \mathbf{G}_{b_1,r}\hat{\mathbf{w}}_{j,\iota} \\ &= \sum_{\iota=1}^{L_j} \mathbf{w}_{j,\iota}^\intercal \mathbf{c}_{j,\iota} \\ &= \sum_{\iota=1}^{L_j} \mathbf{a}_j^\intercal \left[\mathbf{s}_{j,\iota,1}|\cdots|\mathbf{s}_{j,\iota,r}\right]\mathbf{c}_{j,\iota} \\ &= \mathbf{a}_j^\intercal \mathbf{z}_j.\end{aligned}$$

For the last equation, we know that:

$$\begin{aligned}\left[\mathbf{c}_{j,1}^\intercal \otimes \mathbf{G}_{b_1,n} \cdots \mathbf{c}_{j,L_j}^\intercal \otimes \mathbf{G}_{b_1,n}\right]\hat{\mathbf{t}}_j &= \sum_{\iota}^{L_j}\sum_{i=1}^{r} c_{j,\iota,i}\mathbf{G}_{b_1,n}\hat{\mathbf{t}}_{j,\iota,i} \\ &= \mathbf{A}\left(\sum_{\iota}^{L_j}\sum_{i=1}^{r} c_{j,\iota,i}\mathbf{s}_{j,\iota,i}\right) \\ &= \mathbf{A}\mathbf{z}_j.\end{aligned}$$

This concludes the proof. □

Similarly as before, we consider a relaxed relation for proving coordinate-wise special soundness:

$$\mathsf{BR}^*_{b_1,\bar{\beta},\bar{\gamma},\bar{\kappa}} := \left\{ \left(\begin{array}{c} (\mathbf{A},(\mathbf{B}_j,\mathbf{D}_j)_j), \\ ((\mathbf{a}_j,\mathbf{b}_j)_j,\mathbf{u},(y_{j,\iota})_{j,\iota}), \\ ((\mathbf{s}_{j,\iota,i})_{j,\iota,i},(\hat{\mathbf{t}}_j = (\hat{\mathbf{t}}_{j,\iota,i})_{\iota,i})_j,(c_{j,\iota,i})_{j,\iota,i}) \end{array} \right) \middle| \begin{array}{c} \forall j,\iota,i, \mathbf{A}\mathbf{s}_{j,\iota,i} = \mathbf{G}_{b_1,n}\hat{\mathbf{t}}_{j,\iota,i}; \\ \sum_{j=1}^k \mathbf{B}_j\hat{\mathbf{t}}_j = \mathbf{u}; \\ \forall j,\iota, \mathbf{a}_j^\intercal\left[\mathbf{s}_{j,\iota,1}|\cdots|\mathbf{s}_{j,\iota,r}\right]\mathbf{b}_j = y_{j,\iota}; \\ \forall j,\iota,i, \|c_{j,\iota,i}\cdot\mathbf{s}_{j,\iota,i}\| \leqslant \bar{\beta}; \|c_{j,\iota,i}\|_1 \leqslant \bar{\kappa}; \\ c_{j,\iota,i} \in \mathcal{R}_q^\times, \|\hat{\mathbf{t}}_j\| \leqslant \bar{\gamma} \end{array} \right\}. \tag{10}$$

Lemma 3.4. *Define* $\bar{\gamma} := \frac{1}{2}\sqrt{b_1^2(n+1)\delta r\left(\sum_{j=1}^k L_j\right)d + (r\kappa b_0)^2\left(\sum_{j=1}^k \ell_j^2\right)md}$, $\bar{\beta} := 2\bar{\gamma}$ *and* $\bar{\kappa} := 2\kappa$. *Suppose that* $\bar{\kappa} < \frac{1}{\sqrt{2}}\sqrt{q}$. *Then, there exists a polynomial* *time algorithm that on input public parameters* $\mathsf{pp} := (\mathbf{A}, (\mathbf{B}_j, \mathbf{D}_j)_j)$*, statement* $\mathbb{x} := ((\mathbf{a}_j, \mathbf{b}_j)_j, \mathbf{u}, (y_{j,\iota})_{j,\iota})$ *and* $(\sum_{j=1}^{k} L_j)r + 1$ *accepting transcripts*

$$\mathsf{tr}_i := (\mathbf{v}, \mathbf{c}^{(e)}, (\hat{\mathbf{w}}_j^{(e)}, \hat{\mathbf{t}}_j^{(e)}, \mathbf{z}_j^{(e)})_{j \in [k]}) \quad \textit{for } e = 0, 1, \ldots, \left(\sum_{j=1}^{k} L_j\right) r$$

with $(\mathbf{c}^{(e)})_e \in \mathsf{SS}(\mathcal{C}, (\sum_{j=1}^{k} L_j)r)$ *and common first message* $\mathbf{v}$*, either outputs a witness* $\mathbb{w} \in \mathsf{BR}^{*}_{b_1,\bar{\beta},\bar{\gamma},\bar{\kappa}}(\mathsf{pp}, \mathbb{x})$*, or* $\mathbf{z} \in \mathcal{R}_q^{(n+1)\delta r(\sum_{j=1}^{k} L_j)}$ *so that*

$$\left[\mathbf{B}_1 | \cdots | \mathbf{B}_k | \mathbf{D}_1 | \cdots | \mathbf{D}_k\right] \mathbf{z} = \mathbf{0} \quad \textit{and} \quad 0 < \|\mathbf{z}\| \leqslant \bar{\beta}.$$

The proof follows almost identically as for Lemma 3.2. That is, we first claim that unless we found a short Module-SIS solution, all $\hat{\mathbf{w}}_j^{(e)}$, for $e = 0, 1, \ldots, (\sum_{j=1}^{k} L_j)r$, must be the same (and similarly for $\hat{\mathbf{t}}_j^{(e)}$). Then, using the coordinate-wise special soundness property, we extract each vector $\mathbf{s}_{j,\iota,i}$.

Efficiency. Communication complexity from the prover's side can be bounded by

$$nd\lceil \log q \rceil + (n+1)\delta r \left(\sum_{j=1}^{k} L_j\right) d\lceil \log(2b_1) \rceil + \sum_{j=1}^{k} md\lceil \log(2r\kappa b_0 L_j) \rceil.$$

The prover's running time is $O((\sum_{j=1}^{k} L_j)r(m + n + \delta))$ operations over $\mathcal{R}_q$. On the other hand, the verifier's time complexity is dominated by the last equation of (9), which takes $O(n^2 \delta r(\sum_{j=1}^{k} L_j) + knm)$ ring operations. Thus, if each $L_j = O(1)$ then the verifier runtime becomes asymptotically linear in k.

Remark 3.5. Note that trivially concatenating proofs would result in the verification time

$$O\left((n^2 \cdot \delta \cdot r + n \cdot m) \cdot \left(\sum_{j=1}^{k} L_j\right)\right).$$

When $L_1 = ... = L_k = 1$ (i.e. we prove k polynomial evaluations at k different points) then our proposed batching method does not differ from trivially concatenating proofs. The main advantage of our approach comes when one wants prove multiple polynomial evaluations at the *same evaluation point.*

4 Efficient Polynomial Commitments over $\mathbb{Z}_q$

In this section we show how to utilise the proofs of quadratic relations from Section 3 to efficiently prove polynomial evaluations. The key idea is that for a bivariate polynomial

$$f(\mathsf{X}, \mathsf{Y}) = \sum_{i=0}^{m-1} \sum_{j=0}^{r-1} f_{i,j} \mathsf{X}^i \mathsf{Y}^j,$$

where the individual degrees of X and Y are $m-1$ and $r-1$ respectively, we can write

$$f(\mathsf{X},\mathsf{Y}) = \begin{bmatrix}1 \; \mathsf{X} \; \mathsf{X}^2 \cdots \mathsf{X}^{m-1}\end{bmatrix} \begin{bmatrix} f_{0,0} & \cdots & f_{0,r-1} \\ f_{1,0} & \cdots & f_{1,r-1} \\ \vdots & \vdots & \vdots \\ f_{m-1,0} & \cdots & f_{m-1,r-1} \end{bmatrix} \begin{bmatrix} 1 \\ \mathsf{Y} \\ \mathsf{Y}^2 \\ \vdots \\ \mathsf{Y}^{r-1} \end{bmatrix} \tag{11}$$

which is of the same form as in (5) by setting $\mathbf{s}_i$ to be the i-th column of the middle matrix for $i \in [r]$, $\mathbf{a}^\intercal := \begin{bmatrix}1 \; \mathsf{X} \; \mathsf{X}^2 \cdots \mathsf{X}^{m-1}\end{bmatrix}$ and $\mathbf{b}^\intercal := \begin{bmatrix}1 \; \mathsf{Y} \; \mathsf{Y}^2 \cdots \mathsf{Y}^{r-1}\end{bmatrix}$. Therefore, the protocols in Section 3 can intuitively prove polynomial evaluations over $\mathcal{R}_q$. However, there are two caveats. First, the protocols only support witnesses $(\mathbf{s}_i)_i$ with short coefficients. Additionally, to achieve compatibility with Polynomial IOPs, the polynomial commitments should be over finite fields, which is not the case for $\mathcal{R}_q$. We deal with these issues as follows.

4.1 Adapting the Protocols from Section 3

Short coefficients. If we denote the i-th row of the middle matrix in (11) as $\mathbf{f}_i \in \mathcal{R}_q^m$ for $i \in [r]$, then we can define $\mathbf{s}_i := \mathbf{G}_{b_0,m}^{-1}(\mathbf{f}_i) \in \mathcal{R}_q^{\delta_0 m}$ for $\delta_0 := \lfloor \log_{b_0} q \rceil$. Then, the coefficients of all $\mathbf{s}_i$ are indeed short and

$$f(\mathsf{X},\mathsf{Y}) = \mathbf{a}^\intercal \begin{bmatrix}\mathbf{s}_1 | \cdots | \mathbf{s}_r\end{bmatrix} \mathbf{b} \quad \text{where} \quad \begin{array}{l} \mathbf{a}^\intercal := \begin{bmatrix}1 \; \mathsf{X} \; \mathsf{X}^2 \cdots \mathsf{X}^{m-1}\end{bmatrix} \mathbf{G}_{b_0,m} \in \mathcal{R}_q^{\delta_0 m} \\ \mathbf{b}^\intercal := \begin{bmatrix}1 \; \mathsf{Y} \; \mathsf{Y}^2 \cdots \mathsf{Y}^{r-1}\end{bmatrix} \in \mathcal{R}_q^r \end{array} .$$

Hence, by setting $\mathsf{Y} := \mathsf{X}^m$ and using the protocols in Section 3 we can prove arbitrary polynomial evaluations of degree strictly less than $m \cdot r$ over $\mathcal{R}_q$.

Working over $\mathbb{Z}_q$. We recall how one translates proving polynomial evaluations over $\mathbb{Z}_q$ to $\mathcal{R}_q$ as shown in [AFLN24]. Suppose $f(x) = y$ over $\mathbb{Z}_q$ and f has degree at most $N-1$, where N is divisible by the ring dimension d. Then

$$y = \sum_{i=0}^{N-1} f_i x^i = \sum_{i=0}^{N/d-1} \sum_{j=0}^{d-1} f_{id+j} x^{id+j} = \sum_{i=0}^{N/d-1} \left(\sum_{j=0}^{d-1} f_{id+j} x^j \right) \cdot \left(x^d\right)^i .$$

Let $\sigma_{-1} : \mathcal{R} \to \mathcal{R}$ be the Galois automorphism, which maps $X \mapsto X^{-1}$. Thus, if we define the following $\mathcal{R}_q$-elements:

$$\mathsf{x} := \sum_{j=0}^{d-1} x^j \cdot X^j, \qquad \mathsf{f}_i := \sum_{j=0}^{d-1} f_{id+j} \cdot X^j \quad \text{for } i = 0, 1, \ldots, N/d-1,$$

then the constant term (as defined in Section 2) of

$$\mathsf{y} := \sum_{i=0}^{N/d-1} \sigma_{-1}(\mathsf{x}) \cdot \mathsf{f}_i \cdot \left(x^d\right)^i$$

variable	description	instantiation
q	prime modulus, $q \equiv 5 \pmod 8$	
N	degree bound on the polynomials	
d	ring dimension, power-of-two	$\mathsf{poly}(\lambda)$
m	folding parameter	$O(\sqrt{N/d})$
r	folding parameter	$O(\sqrt{N/d})$
n	height of matrices $\mathbf{A}, \mathbf{B}, \mathbf{D}$	$O(1)$
b_0	ℓ_∞ norm of $\mathbf{s}_1, \ldots, \mathbf{s}_r$	$q^{1/O(1)}$
b_1	ℓ_∞ norm of $\hat{\mathbf{t}}_1, \ldots, \hat{\mathbf{t}}_r$	$q^{1/O(1)}$
δ_0	$\lfloor \log_{b_0} q \rfloor$	$O(1)$
δ_1	$\lfloor \log_{b_1} q \rfloor$	$O(1)$
κ	ℓ_1 norm of a challenge	$\omega(1)$
$\bar{\kappa}$	slack parameter	$\sqrt{q/2} > \bar{\kappa} \geqslant 2\kappa$
$\bar{\gamma}$	ℓ_2 norm of $\mathbf{z}$	$\bar{\gamma} := \sqrt{b_1^2(n+1)\delta_1 rd + (r\kappa b_0)^2 \delta_0 md}$
$\bar{\beta}$	ℓ_2 norm bound on extracted witness	$2\bar{\gamma}$
$\mathcal{C}$	$\mathcal{C}^r$ is the challenge space	$\{c \in \mathcal{R} : \|c\|_1 \leqslant \kappa\}$
SL	slack space	$\{c \in \mathcal{R} : \|c\|_1 \leqslant \bar{\kappa}\}^r$
e	identity element in SL	$(1, \ldots, 1)$

Fig. 3: Overview of the notation.

is equal to y [LNP22]. Also, the equation above is a polynomial evaluation statement $\sigma_{-1}(\mathsf{x}) \cdot \mathsf{f}(x^d) = \mathsf{y}$ over $\mathcal{R}_q$, where the polynomial $\mathsf{f} \in \mathcal{R}_q^{<N/d}[\mathsf{X}]$ has coefficients

$$(\mathsf{f}_0, \ldots, \mathsf{f}_{N/d-1}) \in \mathcal{R}_q^{N/d},$$

the evaluation point is $x^d \in \mathcal{R}_q$ and the image is y defined above. Therefore, the prover can first send $\mathsf{y} \in \mathcal{R}_q$ in the clear and proceed with proving knowledge of f such that $\mathsf{f}(x^d) = \sigma_{-1}(\mathsf{x})^{-1} \cdot \mathsf{y}$. Note that now we prove evaluations for polynomials of degree less than N/d rather than N. We refer to [AFLN24, Section 5.5] for more details.

4.2 Construction

We present our basic construction $\mathsf{PCS} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Eval}, \mathsf{Verify})$ for-polynomials over $\mathbb{Z}_q[\mathsf{X}]$ of degree less than $N := m \cdot r \cdot d$ in Figure 4. Basic notation is summarized in Figure 3. The slack space is defined as $\mathsf{SL} := \{c \in \mathcal{R} : \|c\|_1 \leqslant \bar{\kappa}\}^r$ for $\bar{\kappa} \geqslant 1$. We set the identity $\mathsf{e} := (1, \ldots, 1) \in \mathsf{SL}$. As before, we define $\mathcal{C} := \{c \in \mathcal{R} : \|c\|_1 \leqslant \kappa\}$. As a building block, we need a proof system $\Pi' = (\mathcal{S}', \mathcal{P}', \mathcal{V}')$ for the relation R' defined as follows:

$$\mathsf{R}' := \{(\mathsf{pp}, (\mathbf{P}, \mathbf{h}, \bar{\gamma}), \mathbf{z}) : \mathbf{P}\mathbf{z} = \mathbf{h} \wedge \|\mathbf{z}\| \leqslant \bar{\gamma}\}\,. \tag{12}$$

We are ready to summarise the security properties of our polynomial commitment.

```
$\mathsf{Setup}(1^\lambda)$:
 1: $\mathbf{A} \leftarrow \mathcal{R}_q^{n\times\delta_0 m}$
 2: $\mathbf{B} \leftarrow \mathcal{R}_q^{n\times n\delta r}$
 3: $\mathbf{D} \leftarrow \mathcal{R}_q^{n\times\delta r}$
 4: $\mathsf{pp}' \leftarrow \mathcal{S}'(1^\lambda)$
 5: return $\mathsf{pp} := (\mathbf{A},\mathbf{B},\mathbf{D},\mathsf{pp}')$
```

```
$\mathsf{Commit}(\mathsf{pp}, f \in \mathbb{Z}_q^{<N}[X])$:
 1: $f(\mathsf{X}) := \sum_{i=0}^{N-1} f_i \mathsf{X}^i$
 2: for $i = 0,1,\ldots,N/d-1$ :
 3:     $\mathsf{f}_i := \sum_{j=0}^{d-1} f_{id+j}X^j \in \mathcal{R}_q$
 4: for $i = 1,\ldots,r$:
 5:     $\mathbf{f}_i^\intercal := (\mathsf{f}_{(i-1)m},\ldots,\mathsf{f}_{im-1}) \in \mathcal{R}_q^m$
 6:     $\mathbf{s}_i := \mathbf{G}_{b_0,m}^{-1}(\mathbf{f}_i)$
 7:     $\mathbf{t}_i := \mathbf{A}\mathbf{s}_i$
 8:     $\hat{\mathbf{t}}_i := \mathbf{G}_{b_1,n}^{-1}(\mathbf{t}_i)$
 9: $\hat{\mathbf{t}} := (\hat{\mathbf{t}}_i)_{i\in[r]} \in \mathcal{R}_q^{n\delta r}$
10: $\mathbf{u} := \mathbf{B}\hat{\mathbf{t}}$
11: $\mathsf{st} := (\mathbf{s}_i,\hat{\mathbf{t}}_i)_{i\in[r]}$
12: return $(\mathbf{u},\mathsf{st})$
```

```
$\mathsf{Open}(\mathsf{pp},\mathbf{u},f,\mathsf{st} := (\mathbf{s}_i,\hat{\mathbf{t}}_i)_{i\in[r]},(c_i)_{i\in[r]})$
 1: $f(\mathsf{X}) := \sum_{i=0}^{N-1} f_i \mathsf{X}^i$
 2: for $i = 0,1,\ldots,N/d-1$ :
 3:     $\mathsf{f}_i := \sum_{j=0}^{d-1} f_{id+j}X^j \in \mathcal{R}_q$
 4: for $i = 1,\ldots,r$:
 5:     $\mathbf{f}_i^\intercal := (\mathsf{f}_{(i-1)m},\ldots,\mathsf{f}_{im-1}) \in \mathcal{R}_q^m$
 6:     if $\mathbf{G}_{b_0,m}\mathbf{s}_i \neq \mathbf{f}_i \vee \mathbf{A}\mathbf{s}_i \neq \mathbf{G}_{b_1,n}\hat{\mathbf{t}}_i$
 7:         return 0
 8:     if $\|c_i\cdot\mathbf{s}_i\| > \bar{\beta} \vee \|c_i\|_1 > \bar{\kappa} \vee c_i \notin \mathcal{R}_q^\times$
 9:         return 0
10: $\hat{\mathbf{t}}^\intercal := [\hat{\mathbf{t}}_1^\intercal|\cdots|\hat{\mathbf{t}}_r^\intercal]$
11: if $\|\hat{\mathbf{t}}\| > \bar{\gamma} \vee \mathbf{B}\hat{\mathbf{t}} \neq \mathbf{u}$
12:     return 0
13: return 1
```

```
$\mathsf{Eval}.\mathcal{P}(\mathsf{pp},(\mathbf{u},x,y),(f,\mathsf{st} := (\mathbf{s}_i,\hat{\mathbf{t}}_i)))$:
 1: $f(\mathsf{X}) := \sum_{i=0}^{N-1} f_i \mathsf{X}^i$
 2: for $i = 0,1,\ldots,N/d-1$ :
 3:     $\mathsf{f}_i := \sum_{j=0}^{d-1} f_{id+j}X^j \in \mathcal{R}_q$
 4: $\mathsf{x} := \sum_{j=0}^{d-1} x^j \cdot X^j$
 5: $\mathsf{y} := \sum_{i=0}^{N/d-1} \sigma_{-1}(\mathsf{x})\cdot\mathsf{f}_i\cdot\left(x^d\right)^i$
 6: $\mathbf{a}^\intercal := \left[1\ x^d\ x^{2d}\ \cdots\ x^{(m-1)d}\right]\mathbf{G}_{b_0,m}$
 7: $\mathbf{b}^\intercal := \left[1\ x^{md}\ x^{2md}\ \cdots\ x^{(r-1)md}\right]$
 8: $\mathbf{w}^\intercal := \mathbf{a}^\intercal\left[\mathbf{s}_1|\cdots|\mathbf{s}_r\right] \in \mathcal{R}_q^r$
 9: $\hat{\mathbf{w}} := \mathbf{G}_{b_1,r}^{-1}(\mathbf{w}) \in \mathcal{R}_q^{\delta r}$
10: $\mathbf{v} := \mathbf{D}\hat{\mathbf{w}} \in \mathcal{R}_q^n$
11: send $(\mathsf{y},\mathbf{v})$ to $\mathsf{Eval}.\mathcal{V}$
12: receive $\mathbf{c} \in \mathcal{C}^r$ from $\mathsf{Eval}.\mathcal{V}$
13: $\mathbf{z} := \left[\mathbf{s}_1|\cdots|\mathbf{s}_r\right]\mathbf{c}$
14: $\mathbf{P} := \begin{bmatrix} \mathbf{D} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} & \mathbf{0} \\ \mathbf{b}^\intercal\mathbf{G}_{b_1,r} & \mathbf{0} & \mathbf{0} \\ \mathbf{c}^\intercal\mathbf{G}_{b_1,r} & \mathbf{0} & -\mathbf{a}^\intercal \\ \mathbf{0} & \mathbf{c}^\intercal\otimes\mathbf{G}_{b_1,n} & -\mathbf{A} \end{bmatrix}$
15: $\mathbf{z} := \begin{bmatrix} \hat{\mathbf{w}} \\ \hat{\mathbf{t}} \\ \mathbf{z} \end{bmatrix}, \quad \mathbf{h} := \begin{bmatrix} \mathbf{v} \\ \mathbf{u} \\ \sigma_{-1}(\mathsf{x})^{-1}\cdot\mathsf{y} \\ 0 \\ \mathbf{0} \end{bmatrix}$
16: $\bar{\gamma} := \sqrt{b_1^2(n+1)\delta rd + (r\kappa b_0)^2\delta_0 md}$
17: run $\mathcal{P}'(\mathsf{pp}',(\mathbf{P},\mathbf{h},\bar{\gamma}),\mathbf{z})$
```

```
$\mathsf{Eval}.\mathcal{V}(\mathsf{pp},(\mathbf{u},x,y))$:
 1: $\mathsf{x} := \sum_{j=0}^{d-1} x^j \cdot X^j$
 2: $\mathbf{a}^\intercal := \left[1\ x^d\ x^{2d}\ \cdots\ x^{(m-1)d}\right]\mathbf{G}_{b_0,m}$
 3: $\mathbf{b}^\intercal := \left[1\ x^{md}\ x^{2md}\ \cdots\ x^{(r-1)md}\right]$
 4: receive $(\mathsf{y},\mathbf{v})$ from $\mathsf{Eval}.\mathcal{P}$
 5: send $\mathbf{c} \leftarrow \mathcal{C}^r$ to $\mathsf{Eval}.\mathcal{P}$
 6: compute $\mathbf{P},\mathbf{h},\bar{\gamma}$ as in Lines 14 to 16 of
    $\mathsf{Eval}.\mathcal{P}$
 7: if $\mathsf{ct}(\mathsf{y}) \neq y$
 8:     return 0
 9: else run $\mathcal{V}'(\mathsf{pp}',(\mathbf{P},\mathbf{h},\bar{\gamma}))$
```

Fig. 4: Description of the $\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}$ and $\mathsf{Eval} = (\mathsf{Eval}.\mathcal{P}, \mathsf{Eval}.\mathcal{V})$ algorithms.

Theorem 4.1. *The polynomial commitment* PCS *defined in Figure 4 satisfies evaluation completeness, weak binding, and knowledge soundness under the Module-SIS assumption. Namely, let* $\Pi' = (\mathcal{S}', \mathcal{P}', \mathcal{V}')$ *be a proof system for the relation* R'*. Then, the following hold.*

1. *For evaluation completeness,* PCS *satisfies evaluation completeness with completeness error* ϵ'*, where* ϵ' *is the completeness error for* Π'*.*
2. *For weak binding, there is a deterministic algorithm, that given public parameters* $(\mathbf{A}, \mathbf{B}, \mathbf{D}, \mathsf{pp}') \leftarrow \mathsf{Setup}(1^\lambda)$*, and two weak openings* $(f, (\mathbf{s}_i, \hat{\mathbf{t}}_i, c_i)_{i \in [r]})$ *and* $(f', (\mathbf{s}'_i, \hat{\mathbf{t}}'_i, c'_i)_{i \in [r]})$ *for the commitment* $\mathbf{u} \in \mathcal{R}_q^n$ *such that* $f \neq f'$*, outputs a vector* $\mathbf{z} \in \mathcal{R}_q^{\delta_0 m + n\delta_1 r}$ *such that* $[\mathbf{A} \mid \mathbf{B}]\mathbf{z} = \mathbf{0}$ *and* $0 < \|\mathbf{z}\| \leqslant \max(4\bar{\kappa}\bar{\beta}, 2\bar{\gamma})$*.*
3. *As for knowledge soundness, there is an expected PPT extractor* $\mathcal{E}$ *with the folowing properties. Given rewindable black-box access to a PPT prover* $\mathcal{P}^*$ *that convinces* $\mathsf{Eval}.\mathcal{V}(\mathsf{pp}, (\mathbf{u}, x, y))$*, where* $\mathsf{pp} := (\mathbf{A}, \mathbf{B}, \mathbf{D}, \mathsf{pp}') \leftarrow \mathsf{Setup}(1^\lambda)$*, with probability* ε*, extractor* $\mathcal{E}$ *with probability at least*
$$\varepsilon - \varepsilon' - \frac{r}{|\mathcal{C}|}$$
either outputs $f, \mathsf{st}, (c_i)_{i \in [r]}$ *such that* $\mathsf{Open}(\mathsf{pp}, \mathbf{u}, f, \mathsf{st}, (c_i)_{i \in [r]}) = 1$*, or a vector* $\mathbf{z} \in \mathcal{R}_q^{(n+1)\delta_1 r}$ *such that* $[\mathbf{B}|\mathbf{D}]\mathbf{z} = \mathbf{0}$ *and* $0 < \|\mathbf{z}\| \leqslant \bar{\beta}$*, where* ε' *is the knowledge error of* Π'*.*

Proof. We first show that a modified scheme, where instead of running Π' the prover outputs $\mathbf{z}$ in the clear, satisfies perfect evaluation completeness. The statement then follows by composition. Take any polynomial $f \in \mathbb{Z}_q^{<N}[\mathsf{X}]$. Then, for $\mathsf{pp} := (\mathbf{A}, \mathbf{B}, \mathbf{D}, \mathsf{pp}') \leftarrow \mathsf{Setup}(1^\lambda)$ and $(\mathbf{u}, \mathsf{st} := (\mathbf{s}_i, \hat{\mathbf{t}}_i)_{i \in [r]}) \leftarrow \mathsf{Commit}(\mathsf{pp}, f)$ we have

$$\mathbf{G}_{b_0,m}\mathbf{s}_i = \mathbf{G}_{b_0,m}\mathbf{G}_{b_0,m}^{-1}(\mathbf{f}_i) = \mathbf{f}_i \quad \text{and} \quad \mathbf{A}\mathbf{s}_i = \mathbf{t}_i = \mathbf{G}_{b_1,n}(\hat{\mathbf{t}}_i) \quad \text{for } i \in [r].$$

Moreover, $\|\mathbf{s}_i\| \leqslant b_0\sqrt{\delta_0 m d} \leqslant \bar{\beta}$ for all i. Therefore, $\mathsf{Open}(\mathsf{pp}, \mathbf{u}, f, \mathsf{st}, \mathsf{e}) = 1$. Finally, by applying the methodology described in Section 4.1 together with Lemma 3.3, we conclude that the underlying evaluation protocol satisfies perfect completeness, and thus the claim holds.

We move on to weak binding. From Lemma 2.11 we deduce that either all $\mathbf{s}_i = \mathbf{s}'_i$ for all i, or there is an efficient algorithm which finds a short solution to $[\mathbf{A}|\mathbf{B}]$. Suppose the former case. Since $\mathbf{f}_i = \mathbf{G}_{b_0,m}\mathbf{s}_i = \mathbf{G}_{b_0,m}\mathbf{s}'_i = \mathbf{f}'_i$ for all i, and therefore we conclude that $f = f'$, which leads to a contradiction.

As for knowledge soundness, we first consider the modified evaluation protocol, where instead of running Π', the prover outputs $\mathbf{z}$ in the clear. The statement then follows by the composition result [BS23, Lemma 3.7]. To begin with, we use Lemmas 3.2 and 2.6 to deduce that the knowledge error of the evaluation protocol is at least $r/|\mathcal{C}|$. This means that we can define an extractor that with probability at least $\varepsilon - r/|\mathcal{C}|$ either outputs a short solution to $[\mathbf{B}|\mathbf{D}]$, or $\mathsf{st} := (\bar{\mathbf{s}}_i, \hat{\mathbf{t}}_i)_{i \in [r]}$ and $(\bar{c}_i)_{i \in [r]} \in \mathsf{SL}$ such that for $\|(\hat{\mathbf{t}}_i)_{i \in [r]}\| \leqslant \bar{\gamma}$ and $\|\bar{c}_i \cdot \bar{\mathbf{s}}_i\| \leqslant \bar{\beta}$

size		runtime	
commitment	eval. proof	prover	verifier
$O_\lambda(1)$	$O_\lambda(\log\log N)$	$O_\lambda(N)$	$O_\lambda(\sqrt{N})$

Table 3: Asymptotic efficiency in terms of $\mathbb{Z}_q$ elements and operations.

for all $i \in [r]$ and

$$\begin{bmatrix}1 \; x^d \; x^{2d} \; \cdots \; x^{(m-1)d}\end{bmatrix} \mathbf{G}_{b_0,m} \begin{bmatrix}\bar{\mathbf{s}}_1 | \cdots | \bar{\mathbf{s}}_r\end{bmatrix} \begin{bmatrix}1\\ x^{md}\\ x^{2md}\\ \vdots \\ x^{(r-1)md}\end{bmatrix} = \sigma_{-1}(\mathsf{x})^{-1} \cdot \mathsf{y}.$$

Then, by defining $\mathbf{f}_i := \mathbf{G}_{b_0,m}\bar{\mathbf{s}}_i$ for $i \in [r]$ and following the strategy from Section 4.1, one can extract $f \in \mathbb{Z}_q^{<N}[\mathsf{X}]$ so that $f(x) = y$ over $\mathbb{Z}_q$. This concludes the proof. □

Remark 4.2. We highlight that matrices $\mathbf{A}, \mathbf{B}, \mathbf{D}$ can be generated uniformly at random from a seed. Thus, by embedding a Module-SIS challenge inside the aforementioned matrices yields weak binding and knowledge soundness under the Module-SIS assumption.

4.3 Instantiation and Asymptotic Efficiency

We set asymptotic parameters for our polynomial commitment scheme as described in Figure 3. We instantiate our evaluation protocol with LaBRADOR [BS23] as the underlying proof system Π'. We first show that R' is a folklore lattice-type relation that is a special case of *principal relations* (cf. Section 2.4). Thus, we can directly apply the LaBRADOR proof system [BS23] to produce a succinct proof.

The length of the vector $\mathbf{z}$ is $(n+1)\delta_1 r + m = O(\sqrt{N/d})$ elements in $\mathcal{R}_q$, while the height of the matrix $\mathbf{P}$ is $3n+2 = O(1)$. Denote by $\mathbf{p}_i^\intercal$ the i-th row of $\mathbf{P}$. We can then split the vector $\mathbf{z}$ into r' subvectors $\mathbf{z}_1, \ldots, \mathbf{z}_{r'}$ of length n' each, where $r' \cdot n' = (n+1)\delta_1 r + m$. We proceed similarly for all row vectors $\mathbf{p}_i^\intercal := [\mathbf{p}_{i,1}^\intercal | \cdots | \mathbf{p}_{i,r'}^\intercal]$. Then, the linear equation of (12) can be rewritten as $3n+2$ constraints of the form:

$$f_i(\mathbf{z}) := \sum_{j=1}^{r'} \langle \mathbf{p}_{i,j}, \mathbf{z}_j \rangle - h_i = 0 \quad \text{for } i \in [3n+2]$$

where $\mathbf{h} := (h_1, \ldots, h_{3n+2})$. Hence, we formulated the relation in (12) using the native language of LaBRADOR. We apply the LaBRADOR proof system as an underlying building block and pick the most asymptotically optimal parameters as described in [BS23, Section 1.1]. In particular, we set the multiplicity r' and rank n' as follows:

$$r' = O_\lambda\left(N^{\frac{1}{6}}\right) \quad \text{and} \quad n' = O_\lambda\left(N^{\frac{1}{3}}\right).$$

Then, the LaBRADOR sub-protocol has $O(\log\log N)$ rounds and the total size of prover's messages in our evaluation protocol, in terms of the number of $\mathcal{R}_q$-elements, is $O_\lambda(\log\log N)$.

The prover runtime (in terms of the number of $\mathcal{R}_q$-operations) of our evaluation protocol can be split the two parts. The first one is running the protocol in Figure 1, which takes $O_\lambda(r \cdot m) = O_\lambda(N)$ operations. As for running the LaBRADOR building block, the main bottleneck is computing the so-called garbage cross-terms, which takes at most $O_\lambda({r'}^2 \cdot n') = O_\lambda(N^{2/3})$ operations over $\mathbb{Z}_q$. Hence, the naive upper-bound on the prover time for this sub-protocol is $O_\lambda(N^{2/3}\log\log N)$. By combining the two parts, we conclude that the total prover runtime is $O_\lambda(N)$.

Similarly as above, the verifier runtime can be analysed in two parts. The first is receiving the vector $\mathbf{v}$ and generating the challenge $\mathbf{c}$, which takes $O_\lambda(r) = O_\lambda(\sqrt{N})$ time. Further, the verifier runs the verification algorithm from the LaBRADOR protocol, where the statement size is $O_\lambda(r + m) = O_\lambda(\sqrt{N})$ elements over $\mathcal{R}_q$. Since the verifier time for LaBRADOR is linear in the size of the statement, we conclude that the total verifier runtime is $O_\lambda(\sqrt{N})$.

4.4 Batching Evaluation Proofs

Suppose we want to prove knowledge of L polynomials $(f_{j,\iota})_{j\in[k],\iota\in[\ell_j]}$ over $\mathbb{Z}_q$ such that

$$f_{j,\iota}(x_j) = y_{j,\iota} \quad \text{for } j \in [k], \iota \in [\ell_j].$$

We can do this similarly as before by adapting the protocol in Figure 2, where k is now the number of distinct evaluation points, and for the j-th point, we want to prove $\ell_j \geqslant 1$ polynomial evaluations. Then, by following the strategy from Section 4.1, the prover needs to send $L := \sum_{j=1}^{k} \ell_j$ ring elements $(\mathsf{y}_{j,\iota})_{j\in[k],\iota\in[\ell_j]}$ in the clear.

Even though in many Polynomial IOPs we have $L = O(1)$, and thus succinctness is asymptotically preserved, sending all L full-sized elements in $\mathcal{R}_q$ can be costly in practice. To circumvent this problem, one can instead commit to the vector $\mathbf{y} := (\mathsf{y}_{j,\iota})_{j\in[k],\iota\in[\ell_j]} \in \mathcal{R}_q^L$ and later prove its well-formedness, as well as that the constant term of each $\mathsf{y}_{j,\iota}$ equals $y_{j,\iota}$. The key observation here is that these "constant term"-type statements are also natively supported by principal relations, and therefore we can still apply LaBRADOR in a black-box manner.

4.5 Hiding

Our current construction of the polynomial commitment scheme does not natively satisfy the hiding property. Namely, both the commitment and the evaluation protocol may reveal information about the committed values. To remedy this, we introduce the following simple changes.

Computationally hiding commitment scheme. First, we use the hiding version of the outer commitment scheme by sampling a randomness vector $\mathbf{r} \leftarrow \chi^{\mu}$ and computing the commitment $\mathbf{u} := \mathbf{B}\hat{\mathbf{t}} + \mathbf{E}\mathbf{r}$ (instead of $\mathbf{u} = \mathbf{B}\hat{\mathbf{t}}$), where $\mathbf{E} \in \mathcal{R}_q^{n \times \mu}$ is an additional uniformly random matrix. By the (knapsack) Module-LWE assumption, the commitment $\mathbf{u}$ looks pseudorandom. Hence, one needs to choose the parameter μ big enough to ensure that $\mathbf{u}$ does not leak any information about $\hat{\mathbf{t}}$, while not too big since it directly affects efficiency of the underlying scheme.

Defining weak binding. In the hiding version of the commitment, we define a weak opening to additionally contain a short randomness vector $\mathbf{r}$, such that $\mathbf{u} = \mathbf{B}\hat{\mathbf{t}} + \mathbf{E}\mathbf{r}$. More concretely, a weak opening for the commitment $\mathbf{u}$ is a tuple $((\mathbf{s}_i, \hat{\mathbf{t}}_i, c_i)_{i \in [r]}, \mathbf{r})$, which satisfies all the following conditions

$$\forall i \in [r]: \quad \|c_i \cdot \mathbf{s}_i\| \leqslant \bar{\beta}, \quad \|c_i\|_1 \leqslant \bar{\kappa}, \quad c_i \in \mathcal{R}_q^{\times}, \quad \mathbf{A}\mathbf{s}_i = \mathbf{G}_{b,n}\hat{\mathbf{t}}_i$$

$$\mathbf{B}\begin{bmatrix} \hat{\mathbf{t}}_1 \\ \vdots \\ \hat{\mathbf{t}}_r \end{bmatrix} + \mathbf{E}\mathbf{r} = \mathbf{u} \quad \text{and} \quad \left\| \begin{bmatrix} \hat{\mathbf{t}}_1 \\ \vdots \\ \hat{\mathbf{t}}_r \\ \mathbf{r} \end{bmatrix} \right\| \leqslant \bar{\gamma}.$$

Suppose we have two weak openings $((\mathbf{s}_i, \hat{\mathbf{t}}_i, c_i)_{i \in [r]}, \mathbf{r})$ and $((\mathbf{s}'_i, \hat{\mathbf{t}}'_i, c'_i)_{i \in [r]}, \mathbf{r}')$ for the same commitment $\mathbf{u}$. Note that if $\hat{\mathbf{t}}_i \neq \hat{\mathbf{t}}'_i$ for some i, then we immediately yield a short solution for the uniformly random concatenated matrix $[\mathbf{B} \mid \mathbf{E}]$. We argue analogously for the case $\mathbf{r} \neq \mathbf{r}'$. The rest of the proof follows similarly as in Lemma 2.11.

Finally, we highlight that in the knowledge soundness argument, we will be able to extract such a weak opening, since the additional randomness vector $\mathbf{r}$ is a part of the witness for the LaBRADOR subroutine (see (14)).

HVZK Evaluation Proof. We modify the evaluation protocol to achieve honest-verifier zero-knowledge (HVZK) as follows. To begin with, note that sending $\mathsf{y} \in \mathcal{R}_q$ in the clear, and in particular the non-constant terms of y, may naturally reveal some information about the secret polynomial f. To circumvent this issue, we follow the strategy from [ENS20, LNP22]. Let $L \geqslant 1$ be the soundness parameter. The prover at the beginning samples masking terms $\mathbf{l} := (l_1, \ldots, l_L) \leftarrow \{l \in \mathcal{R}_q : \mathsf{ct}(l) = 0\}^L$. Then, it computes $\hat{\mathbf{l}} := \mathbf{G}_{b_1,L}^{-1}(\mathbf{l})$. Next, it commits to both $\hat{\mathbf{w}}, \hat{\mathbf{l}}$ by sampling $\mathbf{r}_v \leftarrow \chi^{\mu}$ and computing

$$\mathbf{v} := \mathbf{D}_0\hat{\mathbf{w}} + \mathbf{D}_1\hat{\mathbf{l}} + \mathbf{E}\mathbf{r}_v$$

where $\mathbf{D}_0, \mathbf{D}_1, \mathbf{E}$ are part of public parameters. Similarly as before, $\mathbf{v}$ is computationally indistinguishable from random. The first prover message is $\mathbf{v}$.

In the second round, the verifier provides L challenges $\alpha_1, \ldots, \alpha_L \leftarrow \mathbb{Z}_q$. The prover replies with $\mathbf{j} := (j_1, \ldots, j_L)$ where

$$j_i := l_i + \alpha_i \cdot \mathsf{y} \quad \text{for } i = 1, \ldots, L. \tag{13}$$

Note that $\mathsf{ct}(j_i) = \mathsf{ct}(l_i + \alpha_i \cdot \mathsf{y}) = \alpha_i \cdot \mathsf{ct}(\mathsf{y}) = \alpha_i \cdot y$ by definition of $l_1, \ldots, l_L$. In particular, the verifier can manually check whether constant terms of each j_i are exactly $\alpha_i \cdot y$. Moreover, sending all j_i reveals no information about the coefficients of y apart from the constant term.

Finally, we have to prove well-formedness of polynomials $j_1, \ldots, j_L$ in Equation (13). That is,

$$\begin{aligned} j_i &= l_i + \alpha_i \cdot \sigma_{-1}(\mathsf{x}) \cdot \mathbf{a}^\intercal \left[\mathbf{s}_1 | \cdots | \mathbf{s}_r\right] \mathbf{b} \\ &= \mathbf{e}_i \mathbf{G}_{b_1,L} \hat{\mathbf{l}} + \alpha_i \cdot \sigma_{-1}(\mathsf{x}) \cdot \mathbf{b}^\intercal \mathbf{G}_{b_1,r} \hat{\mathbf{w}} \end{aligned}$$

for $i \in [L]$, where $\mathbf{e}_i \in \mathcal{R}_q^L$ is the binary vector with 1-entry in exactly i-th position and $\mathbf{a}, \mathbf{b}, \hat{\mathbf{w}}$ are constructed as before. Then, given a challenge $\mathbf{c} \leftarrow \mathcal{C}^r$, the prover now runs the proof system Π' to prove knowledge of short vectors $\hat{\mathbf{w}}, \hat{\mathbf{l}}, \mathbf{r}_v, \hat{\mathbf{t}}, \mathbf{r}, \mathbf{z}$ which satisfy

$$\begin{bmatrix} \mathbf{D}_0 & \mathbf{D}_1 & \mathbf{E} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{E} & \mathbf{0} \\ \alpha_1 \cdot \sigma_{-1}(\mathsf{x}) \cdot \mathbf{b}^\intercal \mathbf{G}_{b_1,r} & \mathbf{e}_1 \mathbf{G}_{b_1,L} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_L \cdot \sigma_{-1}(\mathsf{x}) \cdot \mathbf{b}^\intercal \mathbf{G}_{b_1,r} & \mathbf{e}_L \mathbf{G}_{b_1,L} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{c}^\intercal \mathbf{G}_{b_1,r} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{a}^\intercal \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{c}^\intercal \otimes \mathbf{G}_{b_1,n} & \mathbf{0} & -\mathbf{A} \end{bmatrix} \begin{bmatrix} \hat{\mathbf{w}} \\ \hat{\mathbf{l}} \\ \mathbf{r}_v \\ \hat{\mathbf{t}} \\ \mathbf{r} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{u} \\ j_1 \\ \vdots \\ j_L \\ 0 \\ \mathbf{0} \end{bmatrix}. \qquad (14)$$

Finally, we require Π' to satisfy HVZK. As demonstrated in [BS23, Section 6], we can still apply LaBRADOR to achieve a hiding polynomial commitment scheme.

The intuition for knowledge soundness comes from the following observation, which is used to formally argue (coordinate-wise) special soundness. Suppose we are given two distinct tuples $(\alpha_{0,1}, \ldots, \alpha_{0,L}) \neq (\alpha_{1,1}, \ldots, \alpha_{1,L})$, along with $2L$ polynomials $(j_{b,i})_{b \in \{0,1\}, i \in [L]}$ such that

$$j_{b,i} = l_i + \alpha_{b,i} \mathsf{y} \quad \text{and} \quad \mathsf{ct}(j_{b,i}) = \alpha_{b,i} \cdot y \quad \text{for } b \in \{0,1\}, i \in [L].$$

First, there exists some index i for which $\alpha_{0,i} \neq \alpha_{1,i}$, and thus $\alpha_{0,i} - \alpha_{1,i}$ is invertible over $\mathbb{Z}_q$. Also, the constant term of $j_{0,i} - j_{1,i} = (\alpha_{0,i} - \alpha_{1,i})\mathsf{y}$ is $(\alpha_{0,i} - \alpha_{1,i})y$. Therefore, we conclude that the $\mathsf{ct}(\mathsf{y}) = y$, which is what we wanted. Hence, the soundness error of our HVZK protocol is increased by an additive factor of q^{-L}.

Remark 4.3. We note that the prover actually does not need to reveal all the L ring elements $j_1, \ldots, j_L$ defined in (13). The reason is that LaBRADOR natively also allows to prove statements related to constant terms (see Section 2.4) by applying the same "masking non-constant term" technique as shown above. Thus, we can directly use the framework to prove that $\mathsf{ct}(\mathsf{y}) = y$.

5 Concrete Parameters

We now discuss how to set the various parameters in Greyhound. Similar strategies as in LaBRADOR are employed. We use the standard power-of-two cyclotomic ring of dimension $d = 64$ and modulus $q \approx 2^{32}$, and challenges with $\tau_1 = 32$

non-zero coefficients that are ±1 and $\tau_2 = 8$ non-zero coefficients that are ±2. For simplicity, in the above presentation of the protocol we have used the same SIS rank n for the inner and outer commitments (i.e. height of the matrices $\mathbf{A}$ and $\mathbf{B}$). However, it is indeed more efficient to allow for different ranks and we denote them by n and n_1 for the inner and outer commitments, respectively. They need to be chosen large enough to achieve (weak) binding. We do this in the standard way with respect to the relevant norm bounds, c.f. [MR08].

The N/d witness polynomials that make up the polynomial $f \in \mathbb{Z}_q[X]$ are distributed over r vectors of length m. So we need to have $rm \geqslant N/d$. Then the vectors are decomposed into δ_0 parts with respect to the small integer basis b_0 in order to commit to them. So here we want to have $\delta_0 \log(b_0) \approx \log(q)$. In the protocol the last prover message, i.e. the witness for the LaBRADOR statement, consists of r commitments that are each decomposed into δ parts of length n, the decomposed $\hat{\boldsymbol{w}}$ vector of length δr, and the amortized opening $\boldsymbol{z}$ of length $\delta_0 m$, which is decomposed into two parts with respect to the basis b before handing it over to LaBRADOR. Our goal is thus to minimize $r\delta(n+1) + 2\delta_0 m$ under the constraint $rm \geqslant N/d$. We approximate $b_0 = b$ and hence $\delta_0 = \delta$. Then we find

$$m = \left\lceil \sqrt{\frac{N(n+1)}{2d}} \right\rceil \quad \text{and} \quad r = \left\lceil \frac{N}{md} \right\rceil .$$

We predict the variance of the decomposed $\boldsymbol{z}$ vectors to be $v_z = \frac{b^2}{12} r(\tau_1 + 4\tau_2)$, where we have used $b^2/12$ for the variance of the discrete uniform distribution on $\{-b/2, \dots, b/2 - 1\}$. Then we use $\log(b) = \left\lceil \frac{\log(12 v_z)}{4} \right\rceil$ as the decomposition basis for $\boldsymbol{z}$, and $\delta = \lceil \log(q)/\log(b) \rfloor$. Finally, the square of the predicted total norm for the LaBRADOR statement turns out to be

$$\left(\frac{b^2}{12} + \frac{z_v}{b^2} \right) m\delta_0 d + \left((\delta - 1)\frac{b^2}{12} + \frac{q^2}{12 b^{2(\delta-1)}} \right) (n+1) r d.$$

We summarize the concrete parameters that we have used in our implementation in Table 4. For the parameters inside LaBRADOR and how to optimize them see the Labrador paper. The concrete contributions from the Greyhound protocol to the proof sizes for $N = 2^{26}$, $N = 2^{28}$ and $N = 2^{30}$ due to the parameter choices in Table 4 are 3.75 KB, 3.75 KB and 4.25 KB, respectively.

Making the protocol zero-knowledge. As explained in Section 4.5 for adding zero-knowledge it suffices to add LWE randomness to the outer commitments and mask the polynomial y where the uniformly random masks need to be put into the first outer commitment. This is similar to LaBRADOR. Unlike in LaBRADOR there is no Johnson-Lindenstrauss projection in Greyhound which would be more complicated to mask since it would need a short mask and rejection sampling. We refer to [BS23] for the details. The relatively low-dimensional randomness vectors and masks do not increase the total norm of the output witness much and hence the SIS ranks for the outer commitments can stay the same. Since $q \approx 2^{32}$ we need $L = 4$ masking terms for y so the proof size of Greyhound

	$N = 2^{26}$	$N = 2^{28}$	$N = 2^{30}$
m	3156	6312	12625
r	333	665	1329
n	18	18	18
n_1	7	7	7
b_0	6	5	4
δ_0	5	6	8
b	7	6	6
δ	5	5	5

Table 4: Concrete parameter choices for Greyhound for three different polynomial lengths N.

goes up by three additional polynomials, or 0.75KB. For the LWE randomness we use the uniform distribution modulo b. Then the required LWE rank to achieve the hiding property can be computed in the usual way, c.f. [ADPS15].

6 Implementation

We have implemented Greyhound and LaBRADOR in C with intrinsics for vectorization using the AVX-512 instruction set. The source can be found here:

`https://github.com/lattice-dogs/labrador`.

Our code is single-threaded and so we do not make use of parallization beyond SIMD. We have deviated from the LaBRADOR paper in a few ways. Most importantly we only use power-of-two bases for decomposing vectors, and sample the matrices for the Johnson-Lindenstrauss projections to have coefficients that are ± 1 instead of $-1, 0, 1$. The heuristic from [GHL22, BS23] regarding the tail of the distribution of the projected vectors still applies. The power-of-two decomposition bases mean that we do not achieve the best possible proof sizes. Also we have not yet implemented the most elaborated parameter selection strategy and optimize the parameters for each LaBRADOR layer locally instead of globally optimizing over all layers. The focus of this paper is on runtime and we leave the proof size optimization to later work. The proof sizes are determined by the later LaBRADOR layers where the instance sizes are already so small that those layers do not contribute significantly to the runtime. Therefore we believe that one can improve our proof sizes without influencing the runtime.

Since vectorized code on the Intel architecture often bottlenecks on the front-end of the CPU pipeline we tried to structure our code in a way that is friendly to the μop cache. Concretely, this means that we try to compute on chunks of polynomial vectors that are short enough to fit into the data caches but long enough that the same small code section (for example implementing an NTT) is used on many polynomials and comes from the μop cache rather than the L1 instruction cache and decoding.

For sampling randomness we use the new vectorized AES instructions from the VAES instruction set that compute four (independent) AES-128 rounds simultaneously. Together with hiding the instruction latencies by computing sufficiently many AES blocks in parallel this results in our sampler outputting blocks of 512 bytes of randomness at a time. For the hashing needed in the Fiat-Shamir transform we use SHAKE128.

6.1 Polynomial Arithmetic Library

As part of our implementation we provide an optimized library for polynomial arithmetic modulo (low-degree) power-of-two cyclotomics and primes q of the form $q = 2^d - a$ for $d = 3, \dots, 263$ and minimal a such that $q \equiv 5 \pmod 8$. The library is fully vectorized and includes functions for sampling polynomials from various distributions and applying ring automorphisms.

For theoretical reasons the prime q defining the quotient polynomial ring $\mathcal{R}_q$ for the LaBRADOR proof system needs to have high inertia degree. Therefore we can not use NTT-based multiplication directly for the ring $\mathcal{R}_q$. Instead we use a multi-modular algorithm with NTT-based multiplication modulo several small primes p_i. This is similar to [CHK$^+$21]. Unlike [CHK$^+$21] where a divided-difference based CRT algorithm is used to lift the results from mod p_i, followed by reduction modulo q, we use the explicit CRT mod q from [BS07]. This is advantageous in our case since we compute modulo more small primes primes p_i.

For the p_i we use primes between 2^{12} and 2^{14} that are fully splitting in the main ring $\mathbb{Z}[X]/(X^{64} + 1)$ in LaBRADOR. Such 16-bit primes allow us to use the fast Montgomery arithmetic from [Sei18] and [LS19] on the x86 instruction set.

For the multi-precision arithmetic modulo q we use 14-bit limbs. This is not optimal but allows us to always compute on vectors of 16-bit integers. This also includes the fixed-point approximation to the quotient in the explicit CRT. Moreover, the 14-bit limbs enable a fast forward CRT-map using a (modified) Montgomery reduction algorithm. The arithmetic mod q is much less relevant for the overall speed of our protocols compared to the arithmetic modulo the p_i where most of the operations take place.

We keep the computation of CRT maps and NTTs to a minimum and compute in the multi-modular NTT representation as much as possible. This is the main advantage of NTT-based multiplication in lattice-based protocols with arithmetic in high-rank modules. In the case of Greyhound and LaBRADOR this means that the arithmetic becomes effectively linear.

Instead of the usual sign-and-magnitude representation for the multi-precision arithmetic modulo q we use two's complement and allow for signed limbs in our representation. The main advantage of this is that conversion to and from short polynomials that are stored in signed single-precision representations are very fast. This explains the reason for the 14 bits: positive limbs can go up to $2^{15} - 1$ to not overflow into negative values and we need one nail bit to handle the carries in our vectorized algorithms.

For computing commitments we compute over extension rings by viewing them as vector spaces over our base ring. This reduces the randomness that has to be sampled for the commitment matrices. The computational cost stays quadratic in the extension degree (resp. the SIS rank).

6.2 Johnson-Lindenstrauss Projection

For fast computation of the Johnson-Lindenstrauss reductions in LaBRADOR which essentially entails a matrix-vector product where the matrix has coefficients that are ±1, we use the Four Russians algorithm on blocks of 4 integers. We precompute 16 vector registers at a time, each containing the 16 possible signed summations of 4 vector coefficients. Then, for every matrix row and four times four columns we lookup the correct summations from the precomputed vectors using vector shuffle instructions.

6.3 Future Work

Unlike AVX2, AVX-512 has 52-bit integer instructions that include fused low and high half multiply and add instructions. These instructions enable fast vectorized NTTs modulo 52-bit primes p_i, c.f. [BKS$^+$21]. The advantage of this approach would be that one could compute the commitments directly with NTTs for the extension rings instead of implementing the extension ring arithmetic using quadratic linear algebra over the 64 dimensional base ring. Concretely we often compute commitments in extension rings of rank 16 over $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^{64} + 1)$ with a cost of 16^2 pointwise multiplications of length 64 (note that the NTTs don't matter as they can be precomputed in case of the commitment matrices and reused many times in case of the matrices and the vectors). By directly computing length-1024 pointwise products when the p_i are fully splitting in $\mathbb{Z}_q[X]/(X^{1024} + 1)$ one can reduce the computational cost to only one pointwise product of length 1024 and hence reduce the cost by a factor of 16.

Acknowledgements

We would like to thank the reviewers for their useful comments. Ngoc Khanh Nguyen was supported by the Protocol Labs RFP-013: Cryptonet network grant. Gregor Seiler was supported by the EU H2020 ERC Project 101002845 PLAZA.

References

ACL$^+$22. Martin R. Albrecht, Valerio Cini, Russell W. F. Lai, Giulio Malavolta, and Sri Aravinda Krishnan Thyagarajan. Lattice-based snarks: Publicly verifiable, preprocessing, and recursively composable - (extended abstract). In *CRYPTO (2)*, volume 13508 of *Lecture Notes in Computer Science*, pages 102–132. Springer, 2022.

ADPS15. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - a new hope. *IACR Cryptol. ePrint Arch.*, 2015:1092, 2015.

AF22. Thomas Attema and Serge Fehr. Parallel repetition of $(k_1, \ldots, k_\mu)$-special-sound multi-round interactive proofs. In *CRYPTO (1)*, volume 13507 of *Lecture Notes in Computer Science*, pages 415–443. Springer, 2022.

AFK22. Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs. In *TCC (1)*, volume 13747 of *Lecture Notes in Computer Science*, pages 113–142. Springer, 2022.

AFLN24. Martin R. Albrecht, Giacomo Fenzi, Oleksandra Lapiha, and Ngoc Khanh Nguyen. Slap: Succinct lattice-based polynomial commitments from standard assumptions. To appear at EUROCRYPT 2024, 2024. `https://eprint.iacr.org/2023/1469`.

AHIV17. Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *ACM Conference on Computer and Communications Security*, pages 2087–2104. ACM, 2017.

ALS20. Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments. In *CRYPTO (2)*, volume 12171 of *Lecture Notes in Computer Science*, pages 470–499. Springer, 2020.

BBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In *ICALP*, volume 107 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

BCFL23. David Balbás, Dario Catalano, Dario Fiore, and Russell W. F. Lai. Chainable functional commitments for unbounded-depth circuits. In *TCC (3)*, volume 14371 of *Lecture Notes in Computer Science*, pages 363–393. Springer, 2023.

BCS23. Jonathan Bootle, Alessandro Chiesa, and Katerina Sotiraki. Lattice-based succinct arguments for NP with polylogarithmic-time verification. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023*, volume 14082 of *Lecture Notes in Computer Science*, pages 227–251. Springer, 2023.

BDK13. Michael Backes, Amit Datta, and Aniket Kate. Asynchronous computational VSS with reduced communication complexity. In *CT-RSA*, volume 7779 of *Lecture Notes in Computer Science*, pages 259–276. Springer, 2013.

BFS20. Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent snarks from DARK compilers. In *EUROCRYPT (1)*, volume 12105 of *Lecture Notes in Computer Science*, pages 677–706. Springer, 2020.

BHR$^+$21. Alexander R. Block, Justin Holmgren, Alon Rosen, Ron D. Rothblum, and Pratik Soni. Time- and space-efficient arguments from groups of unknown order. In *CRYPTO (4)*, volume 12828 of *Lecture Notes in Computer Science*, pages 123–152. Springer, 2021.

BHV$^+$23. Rishabh Bhadauria, Carmit Hazay, Muthuramakrishnan Venkitasubramaniam, Wenxuan Wu, and Yupeng Zhang. Private polynomial commitments and applications to MPC. In *Public Key Cryptography (2)*, volume 13941 of *Lecture Notes in Computer Science*, pages 127–158. Springer, 2023.

BKS$^+$21. Fabian Boemer, Sejun Kim, Gelila Seifu, Fillipe D. M. de Souza, and Vinodh Gopal. Intel HEXL: accelerating homomorphic encryption with intel AVX512-IFMA52. In *WAHC@CCS*, pages 57–62. WAHC@ACM, 2021.

BLNS20. Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. A non-pcp approach to succinct quantum-safe zero-knowledge. In *CRYPTO (2)*, volume 12171 of *Lecture Notes in Computer Science*, pages 441–469. Springer, 2020.

BS07. Daniel J. Bernstein and Jonathan P. Sorenson. Modular exponentiation via the explicit chinese remainder theorem. *Math. Comput.*, 76(257):443–454, 2007.

BS23. Ward Beullens and Gregor Seiler. Labrador: Compact proofs for R1CS from module-sis. In *CRYPTO (5)*, volume 14085 of *Lecture Notes in Computer Science*, pages 518–548. Springer, 2023.

CHK$^+$21. Chi-Ming Marvin Chung, Vincent Hwang, Matthias J. Kannwischer, Gregor Seiler, Cheng-Jhih Shih, and Bo-Yin Yang. NTT multiplication for ntt-unfriendly rings new speed records for saber and NTRU on cortex-m4 and AVX2. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(2):159–188, 2021.

CHM$^+$20. Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In *Proceedings of the 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '20, pages 738–768, 2020.

CLM23. Valerio Cini, Russell W. F. Lai, and Giulio Malavolta. Lattice-based succinct arguments from vanishing polynomials. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 72–105, Cham, 2023. Springer Nature Switzerland.

CMNW24. Valerio Cini, Giulio Malavolta, Ngoc Khanh Nguyen, and Hoeteck Wee. Polynomial commitments from lattices: Post-quantum security, fast verification and transparent setup. Cryptology ePrint Archive, Paper 2024/281, 2024.

DAFS24. Thomas Debris-Alazard, Pouria Fallahpour, and Damien Stehlé. Quantum oblivious lwe sampling and insecurity of standard model lattice-based snarks. Cryptology ePrint Archive, Paper 2024/030, 2024. `https://eprint.iacr.org/2024/030`.

dCP23. Leo de Castro and Chris Peikert. Functional commitments for all functions, with transparent setup and from SIS. In *EUROCRYPT (3)*, volume 14006 of *Lecture Notes in Computer Science*, pages 287–320. Springer, 2023.

ENS20. Muhammed F. Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In *ASIACRYPT (2)*, pages 259–288, 2020.

FLV23. Ben Fisch, Zeyu Liu, and Psi Vesely. Orbweaver: Succinct linear functional commitments from lattices. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 106–131, Cham, 2023. Springer Nature Switzerland.

FMN23. Giacomo Fenzi, Hossein Moghaddas, and Ngoc Khanh Nguyen. Lattice-based polynomial commitments: Towards asymptotic and concrete efficiency. Cryptology ePrint Archive, Paper 2023/846, 2023. `https://eprint.iacr.org/2023/846`.

FS86. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194, 1986.

GHL22. Craig Gentry, Shai Halevi, and Vadim Lyubashevsky. Practical non-interactive publicly verifiable secret sharing with thousands of parties. In

EUROCRYPT (1), volume 13275 of *Lecture Notes in Computer Science*, pages 458–487. Springer, 2022.

GLS$^+$21. Alexander Golovnev, Jonathan Lee, Srinath Setty, Justin Thaler, and Riad S. Wahby. Brakedown: Linear-time and field-agnostic SNARKs for R1CS. Cryptology ePrint Archive, Paper 2021/1043, 2021. `https://eprint.iacr.org/2021/1043`.

GWC19. Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019.

HSS24. Intak Hwang, Jinyeong Seo, and Yongsoo Song. Concretely efficient lattice-based polynomial commitment from standard assumptions. Cryptology ePrint Archive, Paper 2024/306, 2024.

KZG10. Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 177–194. Springer, 2010.

Lee21. Jonathan Lee. Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In *TCC (2)*, volume 13043 of *Lecture Notes in Computer Science*, pages 1–34. Springer, 2021.

LNP22. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In *CRYPTO (2)*, volume 13508 of *Lecture Notes in Computer Science*, pages 71–101. Springer, 2022.

LRY16. Benoît Libert, Somindu C. Ramanna, and Moti Yung. Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions. In *ICALP*, volume 55 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

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

LS18. Vadim Lyubashevsky and Gregor Seiler. Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In *EUROCRYPT (1)*, pages 204–224. Springer, 2018.

LS19. Vadim Lyubashevsky and Gregor Seiler. NTTRU: truly fast NTRU using NTT. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(3):180–201, 2019.

MBKM19. Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updateable structured reference strings. Cryptology ePrint Archive, Report 2019/099, 2019.

MR08. Daniele Micciancio and Oded Regev. Lattice-based cryptography. In Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen, editors, *Chapter in Post-quantum Cryptography*, pages 147–191. Springer, 2008.

Sei18. Gregor Seiler. Faster AVX2 optimized NTT multiplication for ring-lwe lattice cryptography. *IACR Cryptol. ePrint Arch.*, page 39, 2018.

STW23. Srinath Setty, Justin Thaler, and Riad Wahby. Unlocking the lookup singularity with lasso. Cryptology ePrint Archive, Paper 2023/1216, 2023. `https://eprint.iacr.org/2023/1216`.

WW23a. Hoeteck Wee and David J. Wu. Lattice-based functional commitments: Fast verification and cryptanalysis. In *ASIACRYPT (5)*, volume 14442 of *Lecture Notes in Computer Science*, pages 201–235. Springer, 2023.

WW23b. Hoeteck Wee and David J. Wu. Succinct vector, polynomial, and functional commitments from lattices. In *EUROCRYPT (3)*, volume 14006 of *Lecture Notes in Computer Science*, pages 385–416. Springer, 2023. Full version: `https://eprint.iacr.org/2022/1515`.