Concretely Efficient Lattice-based Polynomial Commitment from Standard Assumptions

Intak Hwang[1], Jinyeong Seo[1], and Yongsoo Song[1]

Seoul National University
{intak.hwang, jinyeong.seo, y.song}@snu.ac.kr

Abstract. Polynomial commitment is a crucial cryptographic primitive in constructing zkSNARKs. To date, most practical constructions are either insecure against quantum adversaries or lack homomorphic properties, which are useful in recursive compositions of SNARKs. Recently, lattice-based constructions from functional commitments have drawn attention for possessing all the desirable properties, but they yet lack concrete efficiency, and their extractability, which is essential for SNARKs, requires further analysis.
In this paper, we propose a novel construction of an extractable polynomial commitment scheme based on standard lattice-based assumptions, which is transparent and publicly verifiable. Our polynomial commitment has a square-root proof size and verification complexity, but it provides concrete efficiency in proof size, proof generation, and verification. When compared with the recent code-based construction based on Brakedown (CRYPTO 23), our construction provides comparable performance in all aspects.

Keywords: Polynomial Commitment, Lattice, Zero-Knowledge

1 Introduction

Polynomial commitment (PC) is a cryptographic primitive that allows a prover to commit to a polynomial $f \in \mathbb{Z}_p[X]$ by publishing a short commitment. Later, given two public values x and y, the prover can generate a proof π convincing a verifier that $y = f(x)$. A polynomial commitment scheme is said to be evaluation binding if it cannot be proven to evaluate to any other value $y' \neq y$ for x, evaluation hiding if the proof does not reveal any information on the coefficients of f, and extractable if it provides a proof of knowledge. It has been a core building block for many cryptographic protocols, such as verifiable secret sharing [ZXH+22, TCZ+20], or composing zero-knowledge arguments for membership or range relations [BG18].

The most prominent use case of polynomial commitments lies in constructing zero-knowledge non-interactive succinct arguments of knowledge (zkSNARKs), since recent zkSNARKs are based on polynomial interactive oracle proofs [BFS20] combined with polynomial commitment schemes. Therefore, constructing efficient polynomial commitment schemes in terms of prover complexity, verifier

complexity, and proof size has been an important recent research direction. To date, most efficient polynomial commitment schemes [BBB+18, KZG10, Lee21] are based on cryptographic assumptions that are insecure against quantum adversaries. When considering post-quantum security, alternatives from code-based constructions [BSBHR18, GLS+23] exist, but they lack homomorphic properties, which are useful in recursive proof composition techniques [KST22, BCL+21]. Therefore, to achieve both security in terms of post-quantum resistance and efficiency with homomorphic properties, lattice-based constructions have recently drawn attention.

In the realm of lattice-based cryptography, significant progress has been made, rapidly replacing its pre-quantum counterparts in signature and encryption schemes (e.g. [DKL+18, BDK+18]). This advances also extended to lattice-based zero-knowledge proof systems [ALS20, ENS20, LNS20, LNP22], where the size of proofs scales linearly with the input size based on the efficient commitment scheme called BDLOP [BDL+18], providing practical performance. However, there have been relatively limited results on the construction of practically efficient lattice-based polynomial commitment systems. The existing studies on lattice-based polynomial commitments, or more general functional commitments (e.g. [ACL+22, dCP23, WW23b, FLV23]) asymptotically achieve polylogarithmic proof size and verification complexity, but their concrete efficiency has not yet been discussed. Additionally, constructions such as [dCP23, WW23b] do not offer extractability, a crucial requirement in constructing SNARKs, and the extractability of [ACL+22, FLV23] relies on newly introduced assumptions, the knowledge k-R-ISIS assumption, which has been heuristically broken in [WW23a]. There is another recent work [AFLN23], which is based on standard assumptions, provides extractability, and offers concrete proof sizes. However, the concrete performance of the prover or verifier has not been discussed.

1.1 Our Contribution

In this paper, we propose a new lattice-based polynomial commitment scheme that is not only practical in terms of proof size but also efficient in the concrete performance of proof generation and verification. As mentioned above, there have been numerous research efforts on constructing polynomial commitments in the lattice-based setting, but their concrete efficiency has not been presented. Hence, our aim is to provide a concretely efficient lattice-based polynomial commitment scheme, addressing practical concerns that arise in actual implementations.

Modified Ajtai Commitment Scheme. For the basic building block, we use the Ajtai [Ajt96] commitment scheme. There have been several research efforts [BBC+18, BLNS20, NS22, BS23] aimed at constructing SNARKs directly from Ajtai commitment schemes. This is due to its intrinsic compressing property, where commitment size is almost independent of the input size, in contrast to the BDLOP scheme, where commitment size grows linearly with the input size. We also utilize the Ajtai commitment scheme to commit coefficients of the input polynomial, aiming to reduce the asymptotic proof size. We note that the

security of the Ajtai commitment scheme is based on standard assumptions: the MSIS and MLWE problems. However, there are several practical issues in the previous Ajtai-based proof systems: namely, they can only commit small messages for the binding property, and they require rejection sampling [Lyu12] to achieve zero-knowledgeness. These two issues can be critical in the context of polynomial commitment schemes. First, due to soundness, the modulus p of the polynomial is usually set to exponentially large values, such as 255 bits in size, and this incurs parameter blow-ups to handle large size messages. Second, multiple polynomials are committed during the instantiation of PIOP, and this degrades the prover's performance since the rejection rate grows exponentially with respect to the number of committed polynomials.

To address the first issue, we propose a new encoding method inspired by [CIL21, CLPX18], which maps vectors of elements in $\mathbb{Z}_p$ into small normed polynomial ring elements while preserving the homomorphic property. For example, our method can encode 255-bit-sized messages into polynomials with 16-bit-sized coefficients. As a result, the new encoding method effectively resolves overhead from the large modulus p. For the second issue, we devise a randomized encoding technique for the simulatability of proof systems. There is recent work by Kim et al. [KLSS23] that attains simulatability of BDLOP-based proof systems without rejection sampling using the property of the discrete Gaussian distribution. Inspired by this work, we randomize encoded messages so that they follow a discrete Gaussian distribution over a coset of a lattice. As a consequence, we successfully eliminate rejection sampling from our protocol.

Polynomial Evaluation Protocol. With the modified Ajtai commitment scheme, we design a protocol for proving polynomial evaluation. We adapt the methodology from [BCC$^+$16, BG18], which builds polynomial commitment schemes based on the Pedersen commitment scheme, with proof sizes that are square-root in polynomial degree. However, due to the incompatibility in the proof techniques between the Pedersen commitment scheme and the Ajtai commitment scheme, the direct conversion is not straightforward.

We first redesign the knowledge extractor based on the batched sigma protocol in [BBC$^+$18], as the knowledge extractor in [BG18, BCC$^+$16] relies on the invertibility of the Vandermonde matrix over $\mathbb{Z}_p$, which is not compatible with the Ajtai commitment scheme. Second, we adapt the blinding technique from [BG18, BCC$^+$16] into the Ajtai commitment scheme. To achieve the evaluation hiding property, which ensures the proof does not reveal information other than evaluation results, the previous construction uses random blinders from $\mathbb{Z}_p$ to hide committed polynomial coefficients. We adapt this by sampling random blinders from the discrete Gaussian distribution, extending the randomized encoding technique. As a result, we successfully construct a polynomial commitment scheme that provides square-root proof size and verification complexity in terms of polynomial degree, along with extractability and evaluation hiding properties. Additionally, we note that our construction is publicly verifiable and uses a transparent setup.

PoC Implementation. We implement our polynomial commitment scheme at a proof-of-concept level to evaluate its concrete performance. Since our protocol requires sampling from a discrete Gaussian distribution over a coset of the integer lattice, we use the algorithm by Karney [Kar16]. Additionally, we utilize Residue Number System (RNS) representation and Number Theoretic Transform (NTT) to instantiate efficient polynomial ring arithmetic. Finally, we use the optimization technique from [BG14] to further reduce the proof size. The benchmark results indicate that our polynomial commitment scheme offers comparable performance in terms of proof generation, verification, and proof size when compared to Brakedown [GLS+23], another code-based polynomial commitment scheme, which also yields square-root proof size. Moreover, in comparison to SLAP [AFLN23], a lattice-based polynomial commitment scheme, our construction achieves approximately 4.1 times smaller proof size for a polynomial degree of 2^{20}.

1.2 Related Works

The most relevant work to ours is [KSSL21], which also constructs a polynomial commitment scheme based on BDLOP, following the methodology from [BCC+16,BG18]. However, due to the usage of the BDLOP commitment scheme, its proof size is not sublinear in the input polynomial degree. Additionally, they utilize rejection sampling to achieve zero-knowledgeness. Recently, there has been a line of research [ACL+22, WW23b, FLV23] for polynomial commitment, which builds efficient commitment schemes from newly introduced assumptions. However, as discussed before, their extractability requires further analysis, and their setup phase is not transparent. Meanwhile, there are other constructions [dCP23, AFLN23] which provide an efficient polynomial commitment scheme based on standard assumptions. However, [dCP23] does not provide extractability, and its concrete performance was not analyzed in the literature. While [AFLN23] provides extractability with a concrete proof size, it requires a trusted setup, and concrete performance for the prover and verifier was not provided.

Among recent research, there has been a series of work [BBC+18, BLNS20, NS22, BS23], that directly builds SNARKs from the Ajtai commitment scheme. To address issues related to small messages, [BBC+18] presented an encoding method for finite field $GF(p^k)$ where p is small, [NS22] used base representation to decompose inputs into small elements, and [BS23] utilized non-adjacent form. The SNARK constructions in [BBC+18, NS22] provide square-root size with respect to the input size. The leveled Ajtai commitment scheme in [BLNS20] further reduces the asymptotic proof size by generalizing the approach in [BBC+18]. Meanwhile, the proof size in [BS23] and the Bulletproof variant of [BLNS20] provide a polylogarithmic scale. To achieve zero-knowledgeness, [BBC+18,BLNS20,NS22] utilize rejection sampling, but the zero-knowledgeness of [BS23] and the Bulletproof variant of [BLNS20] was not precisely described in the literature. [NS22] and [BS23] provide concrete proof sizes that can be

practically deployed, but the performance of the prover and verifier was not provided.

2 Preliminaries

2.1 Notation

For a positive integer q, we use $\mathbb{Z} \cap (-q/2, q/2]$ as a representative set of $\mathbb{Z}_q$, and denote by $[a]_q$ the reduction of a modulo q. Vectors over $\mathbb{Z}$ or $\mathbb{Z}_q$ are denoted with regular lowercase letters and arrows, such as $\vec{v}$, and matrices over $\mathbb{Z}$ or $\mathbb{Z}_q$ are represented by regular uppercase letters. We regard all vectors as column vectors, and we use the symbol $\|$ for the concatenation of two vectors.

Let d be a power of two. We denote by $R = \mathbb{Z}[X]/(X^d + 1)$ the ring of integers of the $2d$-th cyclotomic field and $R_q = \mathbb{Z}_q[X]/(X^d + 1)$ the residue ring of R modulo q. For polynomials in R or R_q, we use bold lowercase letters to denote them e.g. $\boldsymbol{f}$. We often regard them as d-dimensional vectors over $\mathbb{Z}$ or $\mathbb{Z}_q$ with components corresponding to coefficients. Vectors over R or R_q are denoted with bold lowercase letters and arrows, such as $\vec{\boldsymbol{f}}$, and matrices over R or R_q are represented by bold uppercase letters.

For a vector $\vec{v} = (v_0, \ldots, v_{n-1}) \in \mathbb{Z}^n$, the ℓ^p $(p \geq 1)$ and ℓ^∞ norms are defined as follows:

$$\|v\|_p := \sqrt[p]{\sum_{i=0}^{n-1} |v_i|^p}, \qquad \|v\|_\infty := \max_{0 \leq i < n} |v_i|$$

For a polynomial $\boldsymbol{f}$ or a vector of polynomials $\vec{\boldsymbol{f}}$, $\|\boldsymbol{f}\|_p$ and $\left\|\vec{\boldsymbol{f}}\right\|_p$ are calculated by regarding them as coefficient vectors. For a matrix $A \in \mathbb{R}^{n \times n}$, we denote the matrix norm of A by $\|A\|_2 := \max_{0 \neq \vec{x} \in \mathbb{R}^n} \frac{\|A\vec{x}\|_2}{\|\vec{x}\|_2}$.

2.2 Probability Distributions

We denote sampling x from the distribution $\mathcal{D}$ by $x \leftarrow \mathcal{D}$. For distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ over a countable set S (e.g. $\mathbb{Z}^n$), the statistical distance of $\mathcal{D}_1$ and $\mathcal{D}_2$ is defined as $\frac{1}{2} \cdot \sum_{x \in S} |\mathcal{D}_1(x) - \mathcal{D}_2(x)| \in [0, 1]$. We denote the uniform distribution over S by $\mathcal{U}(S)$ when S is finite.

We define the n-dimensional spherical Gaussian function $\rho : \mathbb{R}^n \to (0, 1]$ as $\rho(\vec{x}) := \exp(-\pi \cdot \vec{x}^\top \vec{x})$. In general, for a positive definite matrix $\Sigma \in \mathbb{R}^{n \times n}$, we define the elliptical Gaussian function $\rho_{\sqrt{\Sigma}} : \mathbb{R}^n \to (0, 1]$ as $\rho_{\sqrt{\Sigma}}(\vec{x}) := \exp(-\pi \cdot \vec{x}^\top \Sigma^{-1} \vec{x})$. Let $\Lambda \subseteq \mathbb{R}^n$ be a lattice and $\vec{c} \in \mathbb{R}^n$. The discrete Gaussian distribution $\mathcal{D}_{\vec{c}+\Lambda,\sqrt{\Sigma}}$ is defined as a distribution over the coset $\vec{c} + \Lambda$, whose probability mass function is $\mathcal{D}_{\vec{c}+\Lambda,\sqrt{\Sigma}}(\vec{x}) = \rho_{\sqrt{\Sigma}}(\vec{x}) / \rho_{\sqrt{\Sigma}}(\vec{c} + \Lambda)$ for $\vec{x} \in \vec{c} + \Lambda$ where $\rho_{\sqrt{\Sigma}}(\vec{c} + \Lambda) := \sum_{\vec{v} \in \vec{c}+\Lambda} \rho_{\sqrt{\Sigma}}(\vec{v}) < \infty$. When $\Sigma = \sigma^2 \cdot I_n$ for $\sigma > 0$ where I_n is the n-dimensional identity matrix, then we substitute $\sqrt{\Sigma}$ by σ in the subscript and refer to σ as the width parameter. For a polynomial $\boldsymbol{x}$, we denote by $\boldsymbol{x} \leftarrow \mathcal{D}_{\vec{c}+\Lambda,\sqrt{\Sigma}}$ if we sample its coefficient vector from $\mathcal{D}_{\vec{c}+\Lambda,\sqrt{\Sigma}}$.

2.3 Useful Lemmas

Definition 1 ([MR07, Def. 3.1]). *For an n-dimensional lattice Λ and positive real $\varepsilon > 0$, the smoothing parameter $\eta_\varepsilon(\Lambda)$ is the smallest s such that $\rho_{1/s}(\Lambda^* \backslash \{0\}) \leq \varepsilon$.*

Definition 2 ([Pei10, Def. 2.3]). *Let Σ be a positive-definite matrix. We say that $\sqrt{\Sigma} \geq \eta_\varepsilon(\Lambda)$ if $\eta_\varepsilon(\sqrt{\Sigma}^{-1} \cdot \Lambda) \leq 1$.*

Lemma 1 ([MR07, Lem. 3.3]). *For any n-dimensional lattice Λ and $\varepsilon > 0$,*

$$\eta_\varepsilon(\Lambda) \leq \sqrt{\frac{\ln(2n(1+1/\varepsilon))}{\pi}} \cdot \lambda_n(\Lambda)$$

where $\lambda_n(\Lambda)$ is the smallest real number $r > 0$ such that $\dim(\mathrm{span}(\Lambda \cap \mathcal{B}(r))) = n$ and $\mathcal{B}(r)$ is the n-dimensional ball with radius r centered at the origin.

Lemma 2 ([KLSS23, Lem. 5]). *For a positive-definite matrix Σ, $\sqrt{\Sigma} \geq \eta_\varepsilon(\Lambda)$ holds if $\left\|\Sigma^{-1}\right\|_2 \leq \eta_\varepsilon(\Lambda)^{-2}$.*

Lemma 3 ([Pei10, Lem. 2.4]). *Let $\Lambda \subseteq \mathbb{R}^n$ be any n-dimensional lattice. For any $0 < \varepsilon < 1$, $\Sigma > 0$ such that $\sqrt{\Sigma} \geq \eta_\varepsilon(\Lambda)$, and any $\vec{c} \in \mathbb{R}^n$,*

$$\rho_{\sqrt{\Sigma}}(\vec{c} + \Lambda) \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right] \cdot \rho_{\sqrt{\Sigma}}(\Lambda)$$

Lemma 4 ([Ban95, Lem. 2.4]). *Let $\Lambda \subseteq \mathbb{R}^n$ be any n-dimensional lattice. For any $0 < \varepsilon < 1/3$, $\sigma \geq \eta_\varepsilon(\Lambda)$, and any $\vec{c} \in \mathbb{R}^n$,*

$$\Pr\left[\|\vec{x}\|_\infty > 5\sigma \mid \vec{x} \leftarrow \mathcal{D}_{\vec{c}+\Lambda,\sigma}\right] \leq n \cdot 2^{-111}$$

Lemma 5 ([MR07, Lem. 4.4]). *Let $\Lambda \subseteq \mathbb{R}^n$ be any n-dimensional lattice. For any $0 < \varepsilon < 1/3$, $\sigma \geq \eta_\varepsilon(\Lambda)$, and any $\vec{c} \in \mathbb{R}^n$,*

$$\Pr\left[\|\vec{x}\|_2 > \sigma\sqrt{n} \mid \vec{x} \leftarrow \mathcal{D}_{\vec{c}+\Lambda,\sigma}\right] \leq 2^{-n+1}$$

Lemma 6 ([Pei10, Fact 2.1]). *Let $\Sigma_1, \Sigma_2 > 0$ be positive-definite matrices. Define $\Sigma_0^{-1} = \Sigma_1^{-1} + \Sigma_2^{-1} > 0$ and $\Sigma_3 = \Sigma_1 + \Sigma_2 > 0$. Let $\vec{x}, \vec{c}_1, \vec{c}_2 \in \mathbb{R}^n$ be arbitrary, and let $\vec{c}_0 \in \mathbb{R}^n$ be such that $\Sigma_0^{-1}\vec{c}_0 = \Sigma_1^{-1}\vec{c}_1 + \Sigma_2^{-1}\vec{c}_2$. Then,*

$$\rho_{\sqrt{\Sigma_1}}(\vec{x} - \vec{c}_1) \cdot \rho_{\sqrt{\Sigma_2}}(\vec{x} - \vec{c}_2) = \rho_{\sqrt{\Sigma_0}}(\vec{x} - \vec{c}_0) \cdot \rho_{\sqrt{\Sigma_3}}(\vec{c}_1 - \vec{c}_2)$$

Lemma 7 (Simplified [Pei10, Thm. 1]). *Let $\Sigma_1, \Sigma_2 > 0$ be positive-definite matrices. Let $\Lambda_1, \Lambda_2 \subseteq \mathbb{R}^n$ be n-dimensional lattices such that $\Lambda_2 \subseteq \Lambda_1$, $\sqrt{\Sigma_1} \geq \eta_\varepsilon(\Lambda_1)$, and $\sqrt{(\Sigma_1^{-1} + \Sigma_2^{-1})^{-1}} \geq \eta_\varepsilon(\Lambda_2)$ for some $0 \leq \varepsilon < 1/2$. Then, for any $\vec{c}_1, \vec{c}_2 \in \mathbb{R}^n$, the distribution $D_{\vec{c}_1+\Lambda_1,\sqrt{\Sigma_1}} + D_{\vec{c}_2+\Lambda_2,\sqrt{\Sigma_2}}$ is within statistical distance 8ε of $D_{\vec{c}_1+\vec{c}_2+\Lambda_1,\sqrt{\Sigma_1+\Sigma_2}}$.*

Lemma 8 ([BCK+14, Lem. 3.1]). *Let d be a power of two, and let $0 \leq i, j < 2d$ such that $i \neq j$. Then, $2(X^i - X^j)^{-1}$ is an element of R such that*

$$\left\|2(X^i - X^j)^{-1}\right\|_\infty \leq 1,$$

where the inverse of $(X^i - X^j)$ is taken over the field $\mathbb{Q}[X]/(X^d + 1)$.

2.4 Module SIS/LWE

Definition 3. *Let μ, ℓ be positive integers, and $0 < \beta < q$. Then, the goal of the Module-SIS (MSIS) problem is to find, for a given matrix $\mathbf{A} \leftarrow \mathcal{U}(R_q^{\mu \times \ell})$, $\vec{\boldsymbol{x}} \in R_q^{\mu+\ell}$ such that $[\mathbf{I}_\mu | \mathbf{A}]\vec{\boldsymbol{x}} = \mathbf{0} \pmod q$ and $\|\vec{\boldsymbol{x}}\|_2 < \beta$. We say that a PPT adversary $\mathcal{A}$ has advantages ε in solving $\mathsf{MSIS}_{R,\mu,q,\beta}$ if*

$$\Pr\left[\|\vec{\boldsymbol{x}}\|_2 < \beta \wedge [\mathbf{I}_\mu | \mathbf{A}]\vec{\boldsymbol{x}} = \mathbf{0} \pmod q \mid \mathbf{A} \leftarrow \mathcal{U}(R_q^{\mu \times \ell}); \vec{\boldsymbol{x}} \leftarrow \mathcal{A}(\mathbf{A})\right] \geq \varepsilon.$$

Definition 4. *Let ν, ℓ be positive integer, and χ be a distribution over $R^{\nu+\ell}$. Then, the goal of the Module-LWE (MLWE) problem is to distinguish $(\mathbf{A}, \vec{\boldsymbol{u}})$ from $(\mathbf{A}, [\mathbf{A}|\mathbf{I}_\ell]\vec{\boldsymbol{r}})$ for $\mathbf{A} \leftarrow \mathcal{U}(R_q^{\ell \times \nu})$, $\vec{\boldsymbol{u}} \leftarrow \mathcal{U}(R_q^\ell)$, and $\vec{\boldsymbol{r}} \leftarrow \chi$. We say that a PPT adversary $\mathcal{A}$ has advantages ε in solving $\mathsf{MLWE}_{R,\nu,q,\chi}$ if*

$$\begin{aligned} &|\Pr\left[b = 1 \mid \mathbf{A} \leftarrow \mathcal{U}(R_q^{\ell \times \nu});\ \vec{\boldsymbol{r}} \leftarrow \chi;\ b \leftarrow \mathcal{A}\left(\mathbf{A}, [\mathbf{A}|\mathbf{I}_\ell]\vec{\boldsymbol{r}}\right)\right] \\ &- \Pr\left[b = 1 \mid (\mathbf{A}, \vec{\boldsymbol{u}}) \leftarrow \mathcal{U}(R_q^{\ell \times \nu} \times R_q^\ell);\ b \leftarrow \mathcal{A}(\mathbf{A}, \vec{\boldsymbol{u}})\right]| \geq \varepsilon. \end{aligned}$$

For spherical discrete Gaussian distributions, we replace them with their width parameters in the MLWE notation for simplicity.

For both problems, the value of ℓ in the hardness estimation is not significant, so we omit it in the parameters for both problems.

2.5 Hint-MLWE

The *Hint-MLWE* problem is a variant of MLWE introduced in recent literature [MKMS22, KLSS23]. This hardness problem is useful when proving the simulatability of lattice-based proof systems without relying on the rejection sampling technique [Lyu12].

Definition 5 (Hint-MLWE). *Let ν, ℓ, n be positive integers, χ and $\psi_0, \ldots, \psi_{n-1}$ be distributions over $R^{\nu+\ell}$, and $\boldsymbol{c}_0, \ldots \boldsymbol{c}_{n-1}$ be elements in R. The Hint-MLWE problem, denoted by $\mathsf{HintMLWE}^{\boldsymbol{c}_0,\ldots,\boldsymbol{c}_{n-1}}_{R,\nu,q,\chi,\psi_0,\ldots,\psi_{n-1}}$, asks the adversary $\mathcal{A}$ to distinguish the following two distributions:*

1. $\left(\mathbf{A}, [\mathbf{A}|\mathbf{I}_\ell]\vec{\boldsymbol{r}}, \vec{\boldsymbol{z}}_0, \ldots, \vec{\boldsymbol{z}}_{k-1}\right)$ *for* $\mathbf{A} \leftarrow \mathcal{U}(R_q^{\ell \times \nu})$, $\vec{\boldsymbol{r}} \leftarrow \chi$, $\vec{\boldsymbol{y}}_i \leftarrow \psi_i$, *and* $\vec{\boldsymbol{z}}_i = \boldsymbol{c}_i \cdot \vec{\boldsymbol{r}} + \vec{\boldsymbol{y}}_i$ *for* $0 \leq i < n$.
2. $\left(\mathbf{A}, \vec{\boldsymbol{u}}, \vec{\boldsymbol{z}}_0, \ldots, \vec{\boldsymbol{z}}_{n-1}\right)$ *for* $\mathbf{A} \leftarrow \mathcal{U}(R_q^{\ell \times \nu})$, $\vec{\boldsymbol{u}} \leftarrow \mathcal{U}(R_q^\ell)$, $\vec{\boldsymbol{r}} \leftarrow \chi$, $\vec{\boldsymbol{y}}_i \leftarrow \psi_i$, *and* $\vec{\boldsymbol{z}}_i = \boldsymbol{c}_i \cdot \vec{\boldsymbol{r}} + \vec{\boldsymbol{y}}_i$ *for* $0 \leq i < n$.

When χ and $\psi_0, \ldots, \psi_{n-1}$ are spherical discrete Gaussian distributions, we simply use their width parameters to denote the Hint-MLWE problem for simplicity.

Theorem 1 (Generalized [KLSS23, Thm 1]). *Let ν, n be positive integers and $\boldsymbol{c}_0, \ldots, \boldsymbol{c}_{n-1}$ be elements of R. For $\sigma_1, \sigma_{2,0}, \ldots \sigma_{2,n-1} > 0$, let $\sigma > 0$ be defined as $1/\sigma^2 = 2(1/\sigma_1^2 + \sum_{i=0}^{n-1} B_i^2/\sigma_{2,i}^2)$, where B_i's are upper bounds for $\|\boldsymbol{c}_i\|_1$'s. If $\sigma \geq \sqrt{2} \cdot \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$ for some $\varepsilon = \mathsf{negl}(\lambda)$, then there exists an efficient reduction from $\mathsf{MLWE}_{R,\nu,q,\sigma}$ to $\mathsf{HintMLWE}^{\boldsymbol{c}_0,\ldots,\boldsymbol{c}_{n-1}}_{R,\nu,q,\sigma_1,\sigma_{2,0},\ldots,\sigma_{2,n-1}}$.*

2.6 Commitment Scheme

We recall the notion of a commitment scheme. A commitment scheme consists of three PPT algorithms $(\mathtt{Setup}, \mathtt{Com}, \mathtt{Open})$.

- $\mathtt{Setup}(1^\lambda) \to \mathsf{ck}$: Given a security parameter λ, it generates a commitment key ck.
- $\mathtt{Com}(\mathsf{ck}, m) \to (c, \delta)$: Given a message m, it returns a commitment c and an opening δ.
- $\mathtt{Open}(\mathsf{ck}, c, m, \delta) \to b$: Given a commitment c, a message m, and an opening δ, it outputs 0 or 1.

Definition 6 (Hiding). *A commitment scheme satisfies hiding if for all PPT adversary $\mathcal{A}$,*

$$\left| \Pr\left[b = b' \middle| \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{Setup}(1^\lambda) \\ (m_0, m_1) \leftarrow \mathcal{A}(\mathsf{ck}) \\ b \leftarrow \mathcal{U}(\{0,1\}) \\ (c, \delta) \leftarrow \mathtt{Com}(\mathsf{ck}, m_b) \\ b' \leftarrow \mathcal{A}(c) \end{array} \right] - \frac{1}{2} \right| \leq \mathsf{negl}(\lambda).$$

Definition 7 (Binding). *A commitment scheme satisfies binding if for all PPT adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{r} \mathtt{Open}(\mathsf{ck}, c, m, \delta) = 1 \wedge \\ \mathtt{Open}(\mathsf{ck}, c, m', \delta') = 1 \wedge \\ m \neq m' \end{array} \middle| \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{Setup}(1^\lambda) \\ (c, m, m', \delta, \delta') \leftarrow \mathcal{A}(\mathsf{ck}) \end{array} \right] \leq \mathsf{negl}(\lambda).$$

In some applications of commitment schemes, such as zero-knowledge proof systems, an interactive protocol called proof of opening knowledge is required, which proves the knowledge of opening δ for given commitments without revealing the committed messages. Its security is defined as follows, adapted from [LNS20, KLSS23].

Definition 8. *An interactive protocol $(\mathcal{P}, \mathcal{V})$ is called a secure proof of opening knowledge protocol for a commitment scheme $(\mathtt{Setup}, \mathtt{Com}, \mathtt{Open})$ if it satisfies the followings:*

- ***Completeness.*** *For all message m,*

$$\Pr\left[b = 1 \wedge \mathtt{Open}(\mathsf{ck}, c, m, \delta) = 1 \middle| \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{Setup}(1^\lambda) \\ (c, \delta) \leftarrow \mathtt{Com}(\mathsf{ck}, m) \\ (\pi, b) \leftarrow \langle \mathcal{P}(\mathsf{ck}, c, \delta), \mathcal{V}(\mathsf{ck}, c) \rangle \end{array} \right] \geq 1 - \mathsf{negl}(\lambda).$$

– ***Soundness.*** *For all PPT adversaries $\mathcal{P}^*$, there exists an expected polynomial time extractor $\mathcal{E}$ such that*

$$\Pr\left[b = 1 \wedge \mathtt{Open}(\mathsf{ck}, c, m, \delta) = 0 \;\middle|\; \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{Setup}(1^\lambda) \\ c \leftarrow \mathcal{P}^*(\mathsf{ck}) \\ (\pi, b) \leftarrow \langle \mathcal{P}^*(\mathsf{ck}, c), \mathcal{V}(\mathsf{ck}, c)\rangle \\ (m, \delta) \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{ck}, c) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

– ***Simulatability.*** *There exists a PPT simulator $\mathcal{S}$ such that for all PPT adversaries $\mathcal{A}$,*

$$\left|\Pr\left[\mathcal{A}(c,\pi) = 1 \;\middle|\; \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{Setup}(1^\lambda) \\ m \leftarrow \mathcal{A}(\mathsf{ck}) \\ (c,\pi) \leftarrow \mathcal{S}(\mathsf{ck}) \end{array}\right] - \Pr\left[\mathcal{A}(c,\pi) = 1 \;\middle|\; \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{Setup}(1^\lambda) \\ m \leftarrow \mathcal{A}(\mathsf{ck}) \\ (c,\delta) \leftarrow \mathtt{Com}(\mathsf{ck}, m) \\ (\pi, b) \leftarrow \langle \mathcal{P}(\mathsf{ck}, c, \delta), \mathcal{V}(\mathsf{ck}, c)\rangle \end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

2.7 Polynomial Commitment Scheme

We recall the notion of a polynomial commitment scheme, adapted from [KZG10]. A polynomial commitment scheme consists of five PPT algorithms ($\mathtt{PC.Setup}$, $\mathtt{PC.Com}$, $\mathtt{PC.Open}$, $\mathtt{PC.Eval}$, $\mathtt{PC.Verify}$).

– $\mathtt{PC.Setup}(1^\lambda, N) \to \mathsf{ck}$: Given a security parameter λ and a polynomial degree upper bound N, it generates a commitment key ck.

– $\mathtt{PC.Com}(\mathsf{ck}, h(X)) \to (c, \delta)$: Given a polynomial $h(X) \in \mathbb{Z}_p[X]$ of degree $< N$, it generates a commitment c and an opening δ.

– $\mathtt{PC.Open}(\mathsf{ck}, c, h(X), \delta) \to b$: Given a commitment c, an opening δ, and polynomial $h(X)$, it outputs 0 or 1.

– $\mathtt{PC.Eval}(x, \delta) \to (y, \rho)$: Given an opening δ and an evaluation point $x \in \mathbb{Z}_p$, it returns an evaluation result y, and an evaluation proof ρ.

– $\mathtt{PC.Verify}(\mathsf{ck}, c, x, y, \rho) \to b$: Given a commitment c, an evaluation point x, an evaluation result y, and an evaluation proof ρ, it outputs 0 or 1.

Definition 9 (Hiding). *A polynomial commitment scheme satisfies hiding if for all PPT adversary $\mathcal{A}$,*

$$\left|\Pr\left[b = b' \;\middle|\; \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, N) \\ (h_0(X), h_1(X)) \leftarrow \mathcal{A}(\mathsf{ck}) \\ b \leftarrow \mathcal{U}(\{0,1\}) \\ (c,\delta) \leftarrow \mathtt{PC.Com}(\mathsf{ck}, h_b(X)) \\ b' \leftarrow \mathcal{A}(c) \end{array}\right] - \frac{1}{2}\right| \leq \mathsf{negl}(\lambda).$$

Definition 10 (Binding). *A polynomial commitment scheme satisfies binding if for all PPT adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{r|r} \mathtt{PC.Open}(\mathsf{ck}, c, h(X), \delta) = 1 \wedge & \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, N) \\ \mathtt{PC.Open}(\mathsf{ck}, c, h'(X), \delta') = 1 \wedge & (c, h(X), h'(X), \delta, \delta') \leftarrow \mathcal{A}(\mathsf{ck}) \\ h(X) \neq h'(X) & \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 11 (Evaluation Binding). *A polynomial commitment scheme satisfies evaluation binding if for all PPT adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{r|r} \mathtt{PC.Verify}(\mathsf{ck}, c, x, y, \rho) = 1 \wedge & \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, N) \\ \mathtt{PC.Verify}(\mathsf{ck}, c, x, y', \rho') = 1 \wedge & (c, x, y, y', \rho, \rho') \leftarrow \mathcal{A}(\mathsf{ck}) \\ y \neq y' & \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 12 (Evaluation Hinding). *A polynomial commitment scheme satisfies evaluation hiding if there exists a PPT simulator $\mathcal{S}$ such that for all PPT adversary $\mathcal{A}$,*

$$\left|\Pr\left[\mathcal{A}(c,\rho) = 1 \middle| \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, N) \\ (h(X), x) \leftarrow \mathcal{A}(\mathsf{ck}) \\ (c, \rho) \leftarrow \mathcal{S}(\mathsf{ck}, x, h(x)) \end{array}\right] - \Pr\left[\mathcal{A}(c,\rho) = 1 \middle| \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, N) \\ (h(X), x) \leftarrow \mathcal{A}(\mathsf{ck}) \\ (c, \delta) \leftarrow \mathtt{PC.Com}(\mathsf{ck}, h(X)) \\ (y, \rho) \leftarrow \mathtt{PC.Eval}(x, \delta) \end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

Similar to commitment schemes, some applications of polynomial commitment schemes, such as zk-SNARK, require an interactive protocol called proof of opening knowledge. It proves the knowledge of opening δ for given commitments without revealing the committed polynomials We adapt its definition from [AFLN23, GLS+23].

Definition 13. *An interactive protocol $(\mathcal{P}, \mathcal{V})$ is called a secure proof of opening knowledge protocol for a polynomial commitment scheme* (PC.Setup, PC.Com, PC.Open, PC.Eval, PC.Verify) *if it satisfies the followings:*

- ***Completeness.*** *For every polynomial $h(X) \in \mathbb{Z}_p[X]$ of degree $< N$ and every point $x \in \mathbb{Z}_p$,*

$$\Pr\left[\begin{array}{r|r} b = 1 \wedge \mathtt{PC.Open}(\mathsf{ck}, c, h(X), \delta) = 1 \wedge & \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^{\lambda,N}) \\ \mathtt{PC.Verify}(\mathsf{ck}, c, x, h(x), \rho) = 1 & (c, \delta) \leftarrow \mathtt{PC.Com}(\mathsf{ck}, h(X)) \\ & (y, \rho) \leftarrow \mathtt{PC.Eval}(\delta, x) \\ & (\pi, b) \leftarrow \langle \mathcal{P}(\mathsf{ck}, c, \delta), \mathcal{V}(\mathsf{ck}, c)\rangle \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

- ***Soundness.*** *For all PPT adversaries $\mathcal{P}^*$, there exists an expected polynomial time extractor $\mathcal{E}$ such that*

$$\Pr\left[\begin{array}{r|r} b = 1 \wedge \mathtt{PC.Verify}(\mathsf{ck}, c, x, y, \rho) = 1 \wedge & \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, N) \\ \big(\mathtt{PC.Open}(\mathsf{ck}, c, h(X), \delta) = 0 \vee y \neq h(x)\big) & (c, x, y, \rho) \leftarrow \mathcal{P}^*(\mathsf{ck}) \\ & (\pi, b) \leftarrow \langle \mathcal{P}^*(\mathsf{ck}, c), \mathcal{V}(\mathsf{ck}, c)\rangle \\ & (h(X), \delta) \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{ck}, c, x, y, \rho) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

– ***Simulatability.*** *There exists a PPT simulator $\mathcal{S}$ such that for all PPT adversaries $\mathcal{A}$,*

$$\left| \Pr\left[\mathcal{A}(c,\rho,\pi)=1 \;\middle|\; \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, N) \\ (h(X), x) \leftarrow \mathcal{A}(\mathsf{ck}) \\ (c,\pi,\rho) \leftarrow \mathcal{S}(\mathsf{ck}, x, h(x)) \end{array} \right] \right.$$
$$\left. - \Pr\left[\mathcal{A}(c,\rho,\pi)=1 \;\middle|\; \begin{array}{r} \mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda) \\ (h(X), x) \leftarrow \mathcal{A}(\mathsf{ck}) \\ (c,\delta) \leftarrow \mathtt{PC.Com}(\mathsf{ck}, h(X)) \\ (y,\rho) \leftarrow \mathtt{PC.Eval}(\delta, x) \\ (\pi, b) \leftarrow \langle \mathcal{P}(\mathsf{ck}, c, \delta), \mathcal{V}(\mathsf{ck}, c) \rangle \end{array} \right] \right| \leq \mathsf{negl}(\lambda).$$

3 Revisiting the Ajtai Commitment Scheme

In this section, we revisit the Ajtai commitment scheme [Ajt96] and its proof of opening knowledge (POK) protocol, which we utilize as the main building block for our polynomial commitment scheme. Our construction is based on previous work by Baum et al. [BBC+18], which builds a SNARK from the Ajtai commitment scheme.

For the Ajtai commitment scheme, we use it to commit coefficients of an input polynomial from $\mathbb{Z}_p[X]$, so it needs to provide a compact commitment for messages in $\mathbb{Z}_p$. In the applications of polynomial commitments, p is usually set to a large prime to attain negligible soundness error. However, the previous approach does not effectively handle such cases, so naive adaptation results in parameter blow-ups.[1] To address this, we introduce a novel encoding map for the Ajtai commitment scheme, which transforms integers in $\mathbb{Z}_p$ into a polynomial with coefficients much smaller than p while still preserving the homomorphic property. This results in a more compact commitment size, preventing parameter blow-ups.

For the POK protocol, we utilize it to provide extractability for our polynomial commitment scheme. The previous construction uses the rejection sampling method [Lyu12] to achieve simulatability. However, it can incur degradation in prover's performance since the repetition rate grows exponentially with respect to the number of committed polynomials, where some use cases of polynomial commitments, such as PIOP, require committing multiple polynomials. Hence, we resolve this issue by eliminating rejection sampling in the POK protocol via a newly developed randomized encoding technique. In the rest of this section, we present more details on our new approaches to the Ajtai commitment scheme and its POK protocol.

[1] The previous work [BBC+18] provides an effective encoding method for a finite field $GF(p^k)$ for a small prime p, but it does not cover the large prime case.

3.1 New Encoding Method

Let us first recall the basic Ajtai commitment scheme. For a message $\vec{\boldsymbol{m}} \in R^\ell$ and a randomness $\vec{\boldsymbol{\mu}}$ sampled from a distribution over $R^{\mu+\nu}$, the commitment $\vec{\mathbf{m}} \in R_q^\mu$ is obtained as

$$\vec{\mathbf{m}} = \mathbf{A}_0\vec{\boldsymbol{m}} + \mathbf{A}_1\vec{\boldsymbol{\mu}} \pmod q$$

where $\mathbf{A}_0 \in R_q^{\mu\times\ell}$ and $\mathbf{A}_1 \in R_q^{\mu\times(\mu+\nu)}$ are (fixed) random matrices. In particular, the binding property of the Ajtai commitment scheme heavily depends on the size of $\vec{\boldsymbol{m}}$.

In the prior work [BBC+18], a simple encoding method is used, which embeds elements of $\mathbb{Z}_p$ into the coefficients of a polynomial in R. Therefore, the commitment modulus q should be sufficiently larger than p to meet the security requirements. If p is a large prime, this approach can lead to significant inefficiencies due to the scaling of commitments. To address this issue, we introduce a novel encoding map, which maps messages from $\mathbb{Z}_p$ for large primes into elements of a polynomial ring with small coefficients. Our encoding map is inspired by [CLPX18, CIL21], which proposes a novel packing method for homomorphic encryption to instantiate high-precision arithmetic.

Lemma 9. *Let b and r be positive integers such that $r \mid d$ and $p = b^r + 1$ is prime. Then, $R/(X^{d/r} - b)$ is isomorphic to $\mathbb{Z}_p^{d/r}$ as $\mathbb{Z}$-modules.*

Proof. Using the fact that $R/(X^{d/r} - b) = \mathbb{Z}[X]/(X^d + 1, X^{d/r} - b)$, we obtain the following isomorphism.

$$\begin{aligned} R/(X^{d/r} - b) &= \bigoplus_{i=0}^{d/r-1} X^i \cdot \mathbb{Z}[X^{d/r}]/(X^d + 1, X^{d/r} - b) \\ &= \bigoplus_{i=0}^{r-1} X^i \cdot \mathbb{Z}_p[X^{d/r}]/(X^{d/r} - b) \cong \prod_{i=0}^{d/r-1} \mathbb{Z}_p[X^{d/r}]/(X^{d/r} - b). \end{aligned}$$

We note that $\mathbb{Z}_p[X^{d/r}]/(X^{d/r} - b) \cong \mathbb{Z}_p$ via an isomorphism $f(X^{d/r}) \mapsto f(b)$. Therefore, we have $R/(X^{d/r} - b) \cong \mathbb{Z}_p^{d/r}$ with respect to the following isomorphism φ from $R/(X^{d/r} - b)$ to $\mathbb{Z}_p^{d/r}$:

$$\varphi : \bar{\boldsymbol{a}} = \sum_{i=0}^{d/r-1}\sum_{j=0}^{r-1} a_{(d/r)j+i}X^{(d/r)j+i} \mapsto \left(\sum_{j=0}^{r-1} a_{(d/r)j}b^j, \ldots, \sum_{j=0}^{r-1} a_{(d/r)(j+1)-1}b^j\right)$$

We note that the isomorphism φ is defined over the polynomial representation on R of an element $\bar{\boldsymbol{a}}$ in $R/(X^{d/r} - b)$. It is well-defined, because for any two polynomial representations $\boldsymbol{a}$ and $\boldsymbol{a}'$ in R of $\bar{\boldsymbol{a}}$, we have $\boldsymbol{a} = \boldsymbol{a}' \pmod{X^{d/r} - b}$ and φ maps $X^{d/r} - b$ to zero. □

We recall that the space of encoded message in the Ajtai commitment scheme is R^ℓ, so we need to design an encoding map from $\mathbb{Z}_p^{d/r}$ to R. The main idea is to utilize the isomorphism between $\mathbb{Z}_p^{d/r}$ and $R/(X^{d/r}-b)$, with R serving as the representation set for elements of $R/(X^{d/r}-b)$. As stated in the above lemma, the isomorphism $\varphi : R/(X^{d/r}-b) \to \mathbb{Z}_p^{d/r}$ can be naturally extended to the map over R, which can be considered as a decoding procedure, the inverse of encoding. Hence, we aim to construct an encoding map that outputs a polynomial representation of $\boldsymbol{a} = \varphi^{-1}(\vec{a}) \in R/(X^{d/r}-b)$ in R with small coefficients for $\vec{a} \in \mathbb{Z}_p^{d/r}$. To achieve this, we present an algorithm in Alg. 1 that outputs a polynomial representation of $\boldsymbol{a}$ with an upper bound $\|\boldsymbol{a}\|_\infty \leq \frac{1}{2}(b+2)$ since $|a_{i,j}| \leq \frac{1}{2}b$ and $|c_{i,j}| \leq 1$.

Algorithm 1 New encoding method

Input: $\vec{a} = (a_0, \ldots, a_{d/r-1}) \in \mathbb{Z}_p^{d/r}$
Output: $\boldsymbol{a} \in R$

for $0 \leq i < d/r$ **do**
  if $a_i = p - 1 \pmod p$ **then**
    $(a_{i,0}, \ldots, a_{i,r-1}) \leftarrow (0, \ldots, 0, b)$
  else
    $(a_{i,0}, \ldots, a_{i,r-1})$ is the base-b representation of $0 \leq a_i < b^r$
  end if
end for
for $0 \leq i < d/r$, $0 \leq j < r$ **do**
  if $a_{i,j} > b/2$ **then**
    $a_{i,j} \leftarrow a_{i,j} - b, \quad c_{i,j} \leftarrow 1$
  else
    $c_{i,j} \leftarrow 0$
  end if
end for
$\boldsymbol{a} \leftarrow \sum_{i=0}^{d/r-1} \sum_{j=0}^{r-1} a_{i,j} X^{(d/r)j+i} + \sum_{i=0}^{d/r-1} \sum_{j=0}^{r-1} c_{i,j} X^{(d/r)(j+1)+i}$

Below, we provide the encoding and decoding algorithms for our commitment scheme.

- $\texttt{Ecd}(\vec{a}) \to \boldsymbol{a}$: Given an element $\vec{a} = (a_0, \ldots, a_{d/r-1}) \in \mathbb{Z}_p^{d/r}$, it runs Alg. 1 and outputs a ring element $\boldsymbol{a}$ where $\|\boldsymbol{a}\|_\infty \leq \frac{b+2}{2}$.
- $\texttt{Dcd}(\boldsymbol{a}) \to \vec{a}$: Given a ring element $\boldsymbol{a} = \sum_{i=0}^{d/r-1} \sum_{j=0}^{r-1} a_{i,j} X^{(d/r)j+i} \in R$, it outputs $\vec{a} = \varphi(\boldsymbol{a}) = \left(\sum_{j=0}^{r-1} a_{0,j} b^j, \ldots, \sum_{j=0}^{r-1} a_{d/r-1,j} b^j \right) \in \mathbb{Z}_p^{d/r}$.

We note that our encoding maps a message vector in $\mathbb{Z}_p^{d/r}$ into a polynomial in R whose norm is bounded by $\frac{b+2}{2}$, which is much smaller than p, while preserving homomorphic property. This allows for a compact commitment size when committing messages from large prime fields, as the commitment modulus can be

chosen to be small while still maintaining the binding properties. As an abuse of notation, we often put an integer $a \in \mathbb{Z}_p$ as an input for the encoding algorithms. In this case, $\mathtt{Ecd}(a)$ outputs a ring element $\boldsymbol{a} = \mathtt{Ecd}(\vec{a})$, where $\vec{a} = (a, 0, \ldots, 0)$ such that $\|\boldsymbol{a}\|_1 \leq \frac{(b+2)r}{2}$ since there are at most r non-zero coefficients.

Now we explain how our encoding method affects the commitment modulus q. As mentioned above, the binding property of the Ajtai commitment $\vec{\mathbf{m}} = \mathbf{A}_0\vec{\boldsymbol{m}} + \mathbf{A}_1\vec{\boldsymbol{\mu}} \pmod q$ is closely related to the size of a message $\vec{\boldsymbol{m}}$, which is reduced to the hardness of $\mathsf{MSIS}_{R,\mu,q,\beta}$ where $\beta = 2\|\vec{\boldsymbol{m}}\|\vec{\boldsymbol{\mu}}\|_2$. Following the analysis from [GN08], the MSIS problem is believed to be computationally hard when β satisfies the following condition for a constant $\delta \approx 1.005$:

$$\beta \leq \min\{q, 2^{2\sqrt{\mu d \log q \log \delta}}\}$$

Assuming the second term is smaller than q, it holds that $\log q = \Omega(\log^2 \beta)$ when other parameters are fixed. If we naively embed the elements in $\mathbb{Z}_p$ into coefficients of a polynomial in R, it holds that $\beta = O(\log p) = O(r \log b)$, but with our method, it holds that $\beta = O(\log b + \log r)$. Hence, our encoding method roughly reduces the size $\log q$ of the commitment modulus by a factor of r^2.

3.2 Randomized Encoding

The rejection sampling is yet another bottleneck in the POK protocol for the Ajtai commitment scheme. To be precise, in the POK protocol of the Ajtai commitment scheme, the response of the form $\vec{\boldsymbol{t}} = \vec{\boldsymbol{g}} + \boldsymbol{c} \cdot \vec{\boldsymbol{m}}$ and $\vec{\boldsymbol{\tau}} = \vec{\boldsymbol{\gamma}} + \boldsymbol{c} \cdot \vec{\boldsymbol{\mu}}$ is transmitted to the verifier, where $\vec{\boldsymbol{m}}$ is an encoded message, $\vec{\boldsymbol{\mu}}$ is a commitment randomness, $\boldsymbol{c}$ is a random challenge from a verifier, and $\vec{\boldsymbol{g}}$, $\vec{\boldsymbol{\gamma}}$ are masks for these values. Since the responses $\vec{\boldsymbol{t}}$ and $\vec{\boldsymbol{\tau}}$ contain partial information about $\vec{\boldsymbol{m}}$ and $\vec{\boldsymbol{\mu}}$, which inhibits the simulatability, the previous construction of the POK protocol used rejection sampling method [Lyu12] to ensure that they follow an independent distribution from $\vec{\boldsymbol{m}}$ and $\vec{\boldsymbol{\mu}}$.

Recently, Kim et al. [KLSS23] presented a POK protocol for BDLOP [BDL+18], which is another lattice-based commitment scheme that does not rely on the rejection sampling technique. This work is based on the observation that the protocol can be still simulatable even if the verifier obtains from the response some partial information on the commitment randomness which is sampled from a discrete Gaussian distribution. Unfortunately, this framework is not simply compatible with the Ajtai scheme where the commitment is represented as a linear combination of both the message $\vec{\boldsymbol{t}}$ and randomness $\vec{\boldsymbol{\tau}}$, different from BDLOP whose POK statement depends only on the commitment randomness. If we directly apply the same approach, then a POK would leak nonnegligible amount of information about the committed message.

To overcome this limitation, we propose a randomized message encoding method where the output follows a discrete Gaussian distribution. We point out that the decoding procedure outputs the same result if inputs are congruent modulo $X^{d/r} - b$, i.e., $\mathtt{Dcd}(\boldsymbol{a}) = \mathtt{Dcd}(\boldsymbol{a}')$ whenever $\boldsymbol{a} = \boldsymbol{a}' \pmod{X^{d/r} - b}$. Since we aim to construct a commitment scheme for messages in $\mathbb{Z}_p$, it is allowed

to incorporate a randomization procedure during the message encoding process as long as its decoding remains the same. To be precise, let $\vec{m} \in \mathbb{Z}_p^{d/r}$ be an input message, and $\boldsymbol{m}$ be the output of the randomized encoding. Since our goal is to have $\boldsymbol{m} = \mathtt{Ecd}(\vec{m}) \pmod{X^{d/r} - b}$, we can sample it from a discrete Gaussian over a coset $\mathtt{Ecd}(\vec{m}) + P\mathbb{Z}^d$, where $P \in \mathbb{R}^{d \times d}$ is the negacyclic matrix corresponding to $X^{d/r} - b$, then it satisfies all the required conditions. Below, we present our randomized encoding algorithm.

- $\mathtt{R.Ecd}(\vec{a}, \mathfrak{s}) \to \boldsymbol{a}$: Given an element $\vec{a} \in \mathbb{Z}_p^{d/r}$ and a positive real $\mathfrak{s} > 0$, output a ring element $\boldsymbol{a} \leftarrow \mathcal{D}_{\mathtt{Ecd}(\vec{a}) + P\mathbb{Z}^d, \mathfrak{s}P}$ where $P \in \mathbb{Z}^{d \times d}$ is the negacyclic matrix of $X^{d/r} - b$.

As an abuse of notation, we often put an $\ell d/r$-dimensional vector $\vec{a} = \vec{a}_0 \| \cdots \| \vec{a}_{\ell-1} \in \mathbb{Z}_p^{\ell d/r}$ as an input for the above algorithms. In this case, $\mathtt{R.Ecd}(\vec{a})$ outputs a vector of ring element $\vec{\boldsymbol{a}} = (\boldsymbol{a}_0, \ldots, \boldsymbol{a}_{\ell-1}) \in R^\ell$ where $\boldsymbol{a}_i = \mathtt{R.Ecd}(\vec{a}_i; \mathfrak{s})$ for $0 \le i < \ell$. We also note that we use the covariant matrix of the form $\mathfrak{s}P$ for the sake of efficiency. We first sample from $\mathcal{D}_{P^{-1}\mathtt{Ecd}(\vec{a}) + \mathbb{Z}^d, \mathfrak{s}}$, and then multiply it by P, since sampling from a spherical discrete Gaussian over a coset of the integer lattice is much more efficient than sampling from a coset of an arbitrary lattice. Finally, we remark that P^{-1} corresponds to the negacyclic matrix of the polynomial $(X^{d/r} - b)^{-1} = -\frac{1}{b^r+1}\left(X^{d-d/r} + bX^{d-2d/r} + \cdots + b^{r-1}\right)$. Thus, it holds the followings:

$$\|P\|_2 \le \left\|X^{d/r} - b\right\|_1 = b + 1, \quad \left\|P^{-1}\right\|_2 \le \left\|(X^{d/r} - b)^{-1}\right\|_1 = \frac{b^r - 1}{(b-1)(b^r+1)} \le \frac{1}{b-1}$$

3.3 A New Proof of Opening Knowledge Protocol

In this subsection, we present our variant of the Ajtai commitment scheme, incorporating the new encoding method and the randomized encoding technique. The algorithms $(\mathtt{Setup}, \mathtt{Com}, \mathtt{Open})$ are defined as follows for the message space $\mathbb{Z}_p^n$.

- $\mathtt{Setup}(1^\lambda) \to \mathsf{ck}$: Given a security parameter λ, it generates a commitment key $\mathsf{ck} = (\mathbf{A}_0, \mathbf{A}_1)$ where $n = \ell d/r$, $\mathbf{A}_0 \leftarrow \mathcal{U}(R_q^{\mu \times \ell})$, and $\mathbf{A}_1 = [\mathbf{A}_1' | \mathbf{I}_\mu] \in R_q^{\mu \times (\mu+\nu)}$ with $\mathbf{A}_1' \leftarrow \mathcal{U}(R_q^{\mu \times \nu})$.
- $\mathtt{Com}(\mathsf{ck}, \vec{m}) \to (\vec{\mathfrak{m}}, \delta)$: Given a message $\vec{m} \in \mathbb{Z}_p^n$, it returns a commitment $\vec{\mathfrak{m}} = \mathbf{A}_0\vec{\boldsymbol{m}} + \mathbf{A}_1\vec{\boldsymbol{\mu}} \in R_q^\mu$ and an opening $\delta = (\vec{\boldsymbol{m}}, \vec{\boldsymbol{\mu}})$, where $\vec{\boldsymbol{m}} \leftarrow \mathtt{R.Ecd}(\vec{m}; \mathfrak{s}_1)$ and $\vec{\boldsymbol{\mu}} \leftarrow \mathcal{D}_{\mathbb{Z}^d, \sigma_1}^{\mu+\nu}$.
- $\mathtt{Open}(\mathsf{ck}, \vec{\mathfrak{m}}, \vec{m}, \delta) \to b$: Given a commitment $\vec{\mathfrak{m}}$, a message $\vec{m}$, and an opening $\delta = (\vec{\boldsymbol{m}}, \vec{\boldsymbol{\mu}})$, it outputs 1 if it satisfies $\|2\vec{\boldsymbol{m}}\|2\vec{\boldsymbol{\mu}}\|_2 < 2d\beta_{\mathtt{Open}}$, $\vec{\mathfrak{m}} = \mathbf{A}_0\vec{\boldsymbol{m}} + \mathbf{A}_1\vec{\boldsymbol{\mu}} \pmod q$, and $\vec{m} = \frac{p+1}{2} \cdot \mathtt{Dcd}(2\vec{\boldsymbol{m}}) \pmod p$; otherwise, it outputs 0.

The commitment scheme defined above is computationally hiding if $\mathsf{MLWE}_{R,\nu,q,\sigma_1}$ is hard, and computationally binding if $\mathsf{MSIS}_{R,\mu,q,4d\beta_{\mathtt{Open}}}$ is hard. Next, we define the POK protocol $\Pi_{\mathtt{Open}}$ as in Fig. 1. Our POK protocol adapts the batching

technique from [BBC+18], hence it can prove the knowledge of openings of k commitments simultaneously. For the challenge set $\mathcal{C}$, we use the set of monomials $\{1, X, \cdots, X^{2d-1}\}$, following the technique from [BCK+14]. The completeness and soundness of the protocol are as follows.

Π_{0pen}

	Prover $\mathcal{P}$		**Verifier** $\mathcal{V}$
	Input: $\mathsf{ck} = (\mathbf{A}_0, \mathbf{A}_1)$		ck
	$\vec{\mathbf{m}}_0, \ldots, \vec{\mathbf{m}}_{k-1} \in R_q^{\mu}$		$\vec{\mathbf{m}}_0, \ldots, \vec{\mathbf{m}}_{k-1}$
	$\vec{\boldsymbol{m}}_0, \ldots, \vec{\boldsymbol{m}}_{k-1} \in R_q^{\ell}$		
	$\vec{\boldsymbol{\eta}}_0, \ldots, \vec{\boldsymbol{\mu}}_{k-1} \in R_q^{\mu+\nu}$		
1 :	For $0 \le j < \kappa = \lceil \lambda / \log(2d) \rceil$,		
2 :	$\quad \vec{g}_j \leftarrow \mathcal{U}(\mathbb{Z}_p^n)$,		
3 :	$\quad \vec{\boldsymbol{g}}_j \leftarrow \mathtt{R.Ecd}(\vec{g}_j, \sqrt{k+1} \cdot \mathfrak{s}_2)$		
4 :	$\quad \vec{\boldsymbol{\gamma}}_j \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sqrt{k+1}\cdot\sigma_2}$		
5 :	$\quad \vec{\mathbf{g}}_j = \mathbf{A}_0 \vec{\boldsymbol{g}}_j + \mathbf{A}_1 \vec{\boldsymbol{\gamma}}_j$	$\xrightarrow{\vec{\mathbf{g}}_j}$	
6 :		$\xleftarrow{\boldsymbol{c}_{j,i}}$	$\boldsymbol{c}_{0,0}, \ldots, \boldsymbol{c}_{\kappa-1,k-1} \leftarrow \mathcal{U}(\mathcal{C})$
7 :	For $0 \le j < \kappa$,		
8 :	$\quad \vec{\boldsymbol{t}}_j = \vec{\boldsymbol{g}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{m}}_i$		
9 :	$\quad \vec{\boldsymbol{\tau}}_j = \vec{\boldsymbol{\gamma}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{\mu}}_i$	$\xrightarrow{\vec{\boldsymbol{t}}_j, \vec{\boldsymbol{\tau}}_j}$	
10 :			For $0 \le j < \kappa$,
11 :			$\left\| \vec{\boldsymbol{t}}_j \| \vec{\boldsymbol{\tau}}_j \right\|_2 \overset{?}{\le} \beta_{\mathsf{0pen}}$
12 :			$\mathbf{A}_0 \vec{\boldsymbol{t}}_j + \mathbf{A}_1 \vec{\boldsymbol{\tau}}_j \overset{?}{=} \vec{\mathbf{g}}_j + \sum_{i=0}^{m} \boldsymbol{c}_{j,i} \cdot \vec{\mathbf{m}}_i$

Fig. 1. Batched proof of opening protocol for the Ajtai commitment scheme

Theorem 2 (Completeness). *The proof of opening knowledge protocol Π_{0pen} in Fig. 1 has completeness if it satisfies the following conditions for $\varepsilon = \mathsf{negl}(\lambda)$:*

- $\mathfrak{s}_1, \sqrt{k+1} \cdot \mathfrak{s}_2 \ge \eta_\varepsilon(\mathbb{Z}^{d\ell})$
- $\sigma_1, \sqrt{k+1} \cdot \sigma_2 \ge \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$
- $\beta_{\mathsf{0pen}} = \sqrt{\left(\left(k\sigma_1 + \sqrt{k+1}\cdot\sigma_2\right)^2 \cdot (\mu+\nu) + (b+1)^2 \cdot \left(k\mathfrak{s}_1 + \sqrt{k+1}\cdot\mathfrak{s}_2\right)^2 \cdot \ell\right) d}$

Proof. See Appendix A.

Theorem 3 (Soundness). *The proof of opening knowledge protocol $\Pi_{\mathtt{Open}}$ in Fig. 1 satisfies soundness.*

Proof. Let $\mathsf{ck} \leftarrow \mathtt{Setup}(1^\lambda)$, and $\mathcal{P}^*$ be a PPT adversary which works as follows:

$$c = (\vec{\mathbf{m}}_0, \ldots, \vec{\mathbf{m}}_{k-1}) \leftarrow \mathcal{P}^*(\mathsf{ck}),\ (\pi, b) \leftarrow \langle \mathcal{P}^*(\mathsf{ck}, c), \mathcal{V}(\mathsf{ck}, c)\rangle$$

Let α be the probability that $b = 1$. By Lem.5 in [BBC+18], there exists an extractor $\mathcal{E}^{\mathcal{P}^*}$ that can extract $(\vec{\boldsymbol{m}}'_i, \vec{\boldsymbol{\mu}}'_i)$ such that $\left\|\vec{\boldsymbol{m}}'_i \| \vec{\boldsymbol{\mu}}'_i\right\|_2 < 2d\beta_{\mathtt{Open}}$, and $\mathbf{A}_0\vec{\boldsymbol{m}}'_i + \mathbf{A}_1\vec{\boldsymbol{\mu}}'_i = 2\vec{\mathbf{m}}_i$ for $0 \le i < k$ in expected time $poly(\lambda)/\alpha$. Then, for $\delta_i = (\frac{q+1}{2} \cdot \vec{\boldsymbol{m}}'_i, \frac{q+1}{2} \cdot \vec{\boldsymbol{\mu}}'_i)$ and $\vec{m}_i = \frac{p+1}{2} \cdot \mathtt{Dcd}(\vec{\boldsymbol{m}}'_i)$, it always holds that $\mathtt{Open}(\mathsf{ck}, \vec{\mathbf{m}}_i, \vec{m}_i, \delta_i) = 1$ for $0 \le i < k$. Thus, in case of non-negligible α, there exists an expected polynomial time extractor that outputs valid message and opening. □

We now prove the simulatability of our POK protocol. The core idea of our proof is to utilize the property of discrete Gaussian distribution similar to the technique from [KLSS23]. We first provide the following lemma, which can be considered as a generalized version of the convolution lemma in Lemma 7.

Lemma 10. *Let $k > 1$ be an integer, $\Sigma \in \mathbb{R}^{n\times n}$ be positive-definitive matrices, and $\Lambda \subseteq \mathbb{Z}^n$ be an n-dimensional lattice. If $\sqrt{\Sigma/2} \ge \eta_\varepsilon(\Lambda)$ for some $0 < \varepsilon < 1/2$, then for any $\vec{u}_0, \ldots, \vec{u}_{k-1} \in \mathbb{R}^n$, the distribution of $\sum_{i=0}^{k-1} \vec{x}_i$, where $\vec{x}_i \leftarrow \mathcal{D}_{\vec{u}_i+\Lambda,\sqrt{\Sigma}}$, is within a statistical distance of $8(k-1)\varepsilon$ from $\mathcal{D}_{\vec{u}+\Lambda,\sqrt{k\Sigma}}$, where $\vec{u} = \sum_{i=0}^{k-1} \vec{u}_i$.*

Proof. We first define $\vec{y}_j := \sum_{i=0}^{j-1} \vec{x}_i$, $\vec{v}_j := \sum_{i=0}^{j-1} \vec{u}_i$. We note that the sample space of $\vec{y}_j$ is identical to $\vec{t}_j + \Lambda$. By induction, it suffices to show that $\vec{y}_{j+1}$ is within a statistical distance of $8(j+1)\varepsilon$ from $\mathcal{D}_{\vec{t}_{j+1}+\Lambda,\sqrt{(j+1)\Sigma}}$ whenever $\vec{y}_j$ is within a statistical distance of $8j\varepsilon$ from $\mathcal{D}_{\vec{t}_j+\Lambda,\sqrt{j\Sigma}}$. By Lem. 7, $\vec{y}_j + \vec{x}_{j+1}$ is within statistical distance of $8(j+1)\varepsilon$ from $\mathcal{D}_{\vec{t}_{j+1}+\Lambda,\sqrt{(j+1)\Sigma}}$ if $\sqrt{\Sigma} \ge \eta_\varepsilon(\Lambda)$ and $\sqrt{\frac{j+1}{j+2}\Sigma} \ge \eta_\varepsilon(\Lambda)$, which are already satisfied since we assume $\sqrt{\Sigma/2} \ge \eta_\varepsilon(\Lambda)$. □

Using the above lemma, one can freely add or decompose discrete Gaussian distributions defined over cosets of the same lattice Λ, under proper conditions. Next, we provide a core lemma for our simulatability.

Lemma 11. *Let $k > 0$ be an integer, $\mathfrak{S}, \Sigma_0, \ldots, \Sigma_{k-1} \in \mathbb{R}^{n\times n}$ be positive-definitive matrices, $C_0, \ldots, C_{k-1} \in \mathbb{Z}^{n\times n}$ be invertible matrices, and $\Lambda \subseteq \mathbb{Z}^n$ be an n-dimensional lattice. If $\sqrt{(\mathfrak{S}^{-1} + \sum_{i=0}^{k-1} C_i^\top \Sigma_i^{-1} C_i)^{-1}} \ge \eta_\varepsilon(\Lambda)$, then for any $\vec{u} \in \mathbb{R}^n$, the following two distributions are within a statistical distance of 2ε.*

$$\left\{(\vec{z}_0, \ldots, \vec{z}_{k-1}) \mid \vec{x} \leftarrow \mathcal{D}_{\vec{u}+\Lambda,\sqrt{\mathfrak{S}}},\ \vec{y}_i \leftarrow \mathcal{D}_{-C_i\vec{u}+\Lambda,\sqrt{\Sigma_i}},\ \vec{z}_i = C_i\vec{x} + \vec{y}_i\right\}$$

$$\left\{(\vec{z'}_0, \ldots, \vec{z'}_{k-1}) \mid \vec{x'} \leftarrow \mathcal{D}_{\Lambda,\sqrt{\mathfrak{S}}},\ \vec{y'_i} \leftarrow \mathcal{D}_{\Lambda,\sqrt{\Sigma_i}},\ \vec{z'_i} = C_i\vec{x'} + \vec{y'_i}\right\}$$

Proof. See Appendix B.

The above lemma essentially says that if we suitably choose the masking vector $\vec{y}_i$ in the linear combination $\vec{z}_i = C_i \cdot \vec{x} + \vec{y}_i$, we can efficiently simulate the resulting distribution without knowing each $\vec{u}_i$. We sketch our proof briefly, incorporating the above lemmas. To achieve simulatability without rejection sampling, we sample mask vectors $\vec{\boldsymbol{g}}_j$ for the responses $\vec{\boldsymbol{t}}_j$ as a randomized encoding of $\vec{g}_j$, a uniformly sampled vector from $\mathbb{Z}_p^n$. Then, it can be decomposed into multiple discrete Gaussian distribution over cosets $\mathtt{Ecd}(-C_{i,j}\vec{m}_i) + P\mathbb{Z}^{d\ell}$ for $0 \le i \le m$ and $\mathtt{Ecd}(\vec{g}_j + \sum_{i=0}^{k-1} C_{i,j}\vec{m}_i) + P\mathbb{Z}^{d\ell}$, using Lem. 10. We note that the distribution of $\vec{g}_j + \sum_{i=0}^{k-1} C_{i,j}\vec{m}_i$ is identical to $\vec{g}_j$ since $\vec{g}_j$ follows the uniform random distribution over $\mathbb{Z}_p^n$. Finally, we use Lem. 11 to simulate the distribution of the responses $\vec{\boldsymbol{t}}_j$. Below, we provide a proof of the simulatability of our protocol.

Theorem 4 (Simulatability). *The proof of opening knowledge protocol $\Pi_{\mathtt{Open}}$ in Fig. 1 satisfies simulatability if it satisfies the followings for $\varepsilon = \mathsf{negl}(\lambda)$:*

- $\frac{\mathfrak{s}_2}{\sqrt{2}}, \mathfrak{s}_{\mathtt{Open}} \ge \frac{1}{b-1} \cdot \eta_\varepsilon((P \otimes I_\ell)\mathbb{Z}^{d\ell})$, *where* $\mathfrak{s}_{\mathtt{Open}} = (\mathfrak{s}_1^{-2} + \kappa\mathfrak{s}_2^{-2})^{-1/2}$
- $\frac{\sigma_2}{\sqrt{2}}, \frac{\sigma_{\mathtt{Open}}}{\sqrt{2}} \ge \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$, *where* $\sigma_{\mathtt{Open}} = \frac{1}{\sqrt{2}} \cdot \left(\sigma_1^{-2} + \kappa\sigma_2^{-2}\right)^{-1/2}$
- *The advantage of* $\mathsf{MLWE}_{R,\nu,q,\sigma_{\mathtt{Open}}}$ *is* $\mathsf{negl}(\lambda)$

$\mathcal{S}_{\mathtt{Open}}(\mathsf{ck})$

1 : $\vec{\mathbf{m}}_i \leftarrow \mathcal{U}(R_q^\mu), \boldsymbol{c}_{i,0}, \ldots, \boldsymbol{c}_{i,\kappa-1} \leftarrow \mathcal{U}(\mathcal{C})$ for $0 \le i < k$

2 : $\vec{\boldsymbol{m}}_i \leftarrow \mathtt{R.Ecd}(\vec{0}, \mathfrak{s}_1),\ \vec{\boldsymbol{\mu}}_i \leftarrow \mathcal{D}_{\mathbb{Z}^d,\sigma_1}$ for $0 \le i < k$

3 : $\vec{g}_j \leftarrow \mathcal{U}(\mathbb{Z}_p^n),\ \vec{\boldsymbol{g}}_j \leftarrow \mathtt{R.Ecd}(\vec{g}_j, \sqrt{k+1} \cdot \mathfrak{s}_2),\ \boldsymbol{\gamma}_j \leftarrow \mathcal{D}_{\mathbb{Z}^d,\sqrt{k+1}\cdot\sigma_2}$ for $0 \le j < \kappa$

4 : $\vec{\boldsymbol{t}}_j = \vec{\boldsymbol{g}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{m}}_i,\ \vec{\boldsymbol{\tau}}_j = \vec{\boldsymbol{\gamma}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{\mu}}_i$ for $0 \le j < \kappa$

5 : $\vec{\mathbf{g}}_j = \mathbf{A}_0\vec{\boldsymbol{t}}_j + \mathbf{A}_1\vec{\boldsymbol{\tau}}_j - \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\mathbf{m}}_i \pmod q$ for $0 \le j < \kappa$

6 : Output $(\vec{\mathbf{m}}, \vec{\mathbf{g}}, \vec{\boldsymbol{c}}, \vec{\boldsymbol{t}}, \vec{\boldsymbol{\tau}})$

Fig. 2. Simulator for $\Pi_{\mathtt{Open}}$

Proof. Let $\vec{m} = \vec{m}_0 \| \ldots \| \vec{m}_{k-1}$ be messages chosen by a PPT adversary $\mathcal{A}(\mathsf{ck})$, where $\mathsf{ck} \leftarrow \mathtt{Setup}(1^\lambda)$. We prove that the algorithm $\mathcal{S}_{\mathtt{Open}}$ in Fig. 2 is an efficient simulator for the protocol $\Pi_{\mathtt{Open}}$ using hybrid arguments. We first define a distribution $\mathcal{H}_0(\mathsf{ck}, \vec{m})$ as follows, where $\vec{\mathbf{m}} = \vec{\mathbf{m}}_0 \| \cdots \| \vec{\mathbf{m}}_{k-1}$, $\vec{\mathbf{g}} = \vec{\mathbf{g}}_0 \| \cdots \| \vec{\mathbf{g}}_{\kappa-1}$,

$\vec{\boldsymbol{c}} = (\boldsymbol{c}_{0,0}, \ldots, \boldsymbol{c}_{k-1,\kappa-1})$, $\vec{\boldsymbol{t}} = \vec{\boldsymbol{t}}_0 \| \cdots \| \vec{\boldsymbol{t}}_{\kappa-1}$, and $\vec{\boldsymbol{\tau}} = \vec{\boldsymbol{\tau}}_0 \| \cdots \| \vec{\boldsymbol{\tau}}_{\kappa-1}$.

$$\left\{ (\vec{\mathfrak{m}}, \vec{\mathfrak{g}}, \vec{\boldsymbol{c}}, \vec{\boldsymbol{t}}, \vec{\boldsymbol{\tau}}) \;\middle|\; \begin{array}{r} \vec{\boldsymbol{m}}_i \leftarrow \mathtt{R.Ecd}(\vec{m}_i, \mathfrak{s}_1),\ \vec{\boldsymbol{\mu}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sigma_1},\ \vec{\mathfrak{m}}_i = \mathbf{A}_0\vec{\boldsymbol{m}}_i + \mathbf{A}_1\vec{\boldsymbol{\mu}}_i \text{ for } 0 \le i < k \\ \boldsymbol{c}_{j,0}, \ldots, \boldsymbol{c}_{j,k-1} \leftarrow \mathcal{U}(\mathcal{C}),\ \vec{g}_j \leftarrow \mathcal{U}(\mathbb{Z}_p^n) \\ \vec{\boldsymbol{g}}_j \leftarrow \mathtt{R.Ecd}(\vec{g}_j, \sqrt{2}\mathfrak{s}_2),\ \vec{\boldsymbol{\gamma}}_j \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sqrt{2}\sigma_2} \\ \vec{\boldsymbol{t}}_j = \vec{\boldsymbol{g}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{m}}_i,\ \vec{\boldsymbol{\tau}}_j = \vec{\boldsymbol{\gamma}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{\mu}}_i \\ \vec{\mathfrak{g}}_j = \mathbf{A}_0\vec{\boldsymbol{t}}_j + \mathbf{A}_1\vec{\boldsymbol{\tau}}_j - \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\mathfrak{m}}_i \text{ for } 0 \le j < \kappa \end{array} \right\}$$

We note that the above distribution corresponds to the distribution of transcripts from $\mathit{\Pi}_{\mathtt{Open}}$.

Claim 1: $\mathcal{H}_0(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ are statistically indistinguishable, where $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ is defined as follows:

$$\left\{ (\vec{\mathfrak{m}}, \vec{\mathfrak{g}}, \vec{\boldsymbol{c}}, \vec{\boldsymbol{t}}, \vec{\boldsymbol{\tau}}) \;\middle|\; \begin{array}{r} \vec{\boldsymbol{m}}_i \leftarrow \mathtt{R.Ecd}(\vec{m}_i, \mathfrak{s}_1),\ \vec{\boldsymbol{\mu}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sigma_1},\ \vec{\mathfrak{m}}_i = \mathbf{A}_0\vec{\boldsymbol{m}}_i + \mathbf{A}_1\vec{\boldsymbol{\mu}}_i \text{ for } 0 \le i < k \\ \boldsymbol{c}_{j,0}, \ldots, \boldsymbol{c}_{j,k-1} \leftarrow \mathcal{U}(\mathcal{C}),\ \vec{g}_j \leftarrow \mathcal{U}(\mathbb{Z}_p^n) \\ \underline{\vec{\boldsymbol{g}}_{j,i} \leftarrow \mathtt{R.Ecd}(-C_{j,i}\vec{m}_i, \cdot\mathfrak{s}_2) \text{ for } 0 \le i < k,\ \vec{\boldsymbol{g}}_{j,k} \leftarrow \mathtt{R.Ecd}(\vec{g}_j + \sum_{i=0}^{k-1} C_{j,i}\vec{m}_i, \mathfrak{s}_2)} \\ \underline{\vec{\boldsymbol{\gamma}}_{j,0}, \ldots, \vec{\boldsymbol{\gamma}}_{j,k} \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sqrt{m+2}\cdot\sigma_2}} \\ \vec{\boldsymbol{t}}_j = \sum_{i=0}^{k-1} (\vec{\boldsymbol{g}}_{j,i} + \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{m}}_i) + \vec{\boldsymbol{g}}_{j,k},\ \vec{\boldsymbol{\tau}}_j = \sum_{i=0}^{k-1} (\vec{\boldsymbol{\gamma}}_{j,i} + \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{\mu}}_i) + \vec{\boldsymbol{\gamma}}_{j,k} \\ \vec{\mathfrak{g}}_j = \mathbf{A}_0\vec{\boldsymbol{t}}_j + \mathbf{A}_1\vec{\boldsymbol{\tau}}_j - \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\mathfrak{m}}_i \text{ for } 0 \le j < \kappa \end{array} \right\}$$

The difference between $\mathcal{H}_0(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ lies in how $\vec{\boldsymbol{g}}_j$ and $\vec{\boldsymbol{\gamma}}_j$ are sampled, which we underlined in the above. Let $P_\ell = P \otimes I_\ell \in \mathbb{R}^{d\ell \times d\ell}$. In $\mathcal{H}_0(\mathsf{ck}, \vec{m})$, $\vec{\boldsymbol{g}}_j \sim \mathcal{D}_{\mathtt{Ecd}(\vec{g}_j)+P_\ell\mathbb{Z}^{d\ell},\sqrt{k+1}\mathfrak{s}_2 P_\ell}$ and $\vec{\boldsymbol{\gamma}}_j \sim \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sqrt{k+1}\sigma_2}$. Meanwhile, in $\mathcal{H}_1(\mathsf{ck}, \vec{m})$,

$$\vec{\boldsymbol{g}}_j = \sum_{i=0}^{k} \vec{\boldsymbol{g}}_{j,i} \sim \sum_{i=0}^{k-1} \mathcal{D}_{\mathtt{Ecd}(-C_{j,i}\vec{m}_i)+P_\ell\mathbb{Z}^{d\ell},\mathfrak{s}_2 P_\ell} + \mathcal{D}_{\mathtt{Ecd}(\vec{g}_j+\sum_{i=0}^{k-1} C_{j,i}\vec{m}_i)+P_\ell\mathbb{Z}^{d\ell},\mathfrak{s}_2 P_\ell}$$
$$\vec{\boldsymbol{\gamma}}_j = \sum_{i=0}^{k} \vec{\boldsymbol{\gamma}}_{j,i} \sim \sum_{i=0}^{k} \mathcal{D}^{\mu+\nu}_{Z^d,\sigma_2}$$

where $C_{j,i}$ is the negacyclic matrix of $\boldsymbol{c}_{j,i}$. By Lem. 10, the distribution of $\vec{\boldsymbol{g}}_j$ in $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ is within a statistical distance of $8k\varepsilon$ from $\mathcal{D}_{\mathtt{Ecd}(\vec{g}_j)+P_\ell\mathbb{Z}^{d\ell},\sqrt{k+1}\mathfrak{s}_2 P_\ell}$ if $\frac{\mathfrak{s}_2}{\sqrt{2}} P_\ell \ge \eta_\varepsilon(P_\ell\mathbb{Z}^{d\ell})$, which is implied by $\frac{\mathfrak{s}_2}{\sqrt{2}} \ge \frac{1}{b-1}\eta_\varepsilon(P_\ell\mathbb{Z}^{d\ell})$ from Lem. 2 and $\left\|P^{-1}\right\|_2 \le \frac{1}{b-1}$. In a similar manner, by Lem. 10, $\sum_{i=0}^{k} \mathcal{D}^{\mu+\nu}_{Z^d,\sigma_2}$ is within a statistical distance of $8k\varepsilon$ from $\mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sqrt{k+1}\sigma_2}$ if $\frac{\sigma_2}{\sqrt{2}} \ge \eta_\varepsilon(\mathbb{Z}^d)$. Therefore, $\mathcal{H}_0(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ are within a statistical distance $16k\varepsilon$, which is $\mathsf{negl}(\lambda)$.

Claim 2: $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ are computationally indistinguishable, where $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ is defined as follows:

$$\left\{ (\vec{\mathfrak{m}}, \vec{\mathfrak{g}}, \vec{c}, \vec{t}, \vec{\tau}) \middle| \begin{array}{r} \vec{\boldsymbol{m}}_i \leftarrow \mathtt{R.Ecd}(\vec{m}_i, \mathfrak{s}_1),\ \vec{\boldsymbol{\mu}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_1},\ \underline{\vec{\mathfrak{m}}_i \leftarrow \mathcal{U}(R_q^\mu) \text{ for } 0 \le i < k} \\ \text{The rest is same as } \mathcal{H}_1(\mathsf{ck}, \vec{m}) \end{array} \right\}$$

The difference between $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ lies in how $\vec{\mathfrak{m}}_i$'s are sampled. Then, distinguishing $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ is at least hard as $\mathsf{HintMLWE}^{\boldsymbol{c}_0, \ldots, \boldsymbol{c}_{\kappa-1}}_{R,\nu,q,\sigma_1,\sigma_2,\ldots,\sigma_2}$ since $\left(\vec{\mathfrak{m}}_i, \vec{\boldsymbol{\gamma}}_{0,i} + \boldsymbol{c}_{0,i} \cdot \vec{\boldsymbol{\mu}}_i, \ldots, \vec{\boldsymbol{\gamma}}_{\kappa-1,i} + \boldsymbol{c}_{\kappa-1,i} \cdot \vec{\boldsymbol{\mu}}_i\right)$ can be considered as a $\mathsf{HintMLWE}$ instance. By Thm. 1, there is an efficient reduction from $\mathsf{MLWE}_{R,\nu,q,\sigma_{\mathtt{Open}}}$ to $\mathsf{HintMLWE}^{\boldsymbol{c}_0, \ldots, \boldsymbol{c}_{\kappa-1}}_{R,\nu,q,\sigma_1,\sigma_2,\ldots,\sigma_2}$ if $\sigma_{\mathtt{Open}} \ge \sqrt{2} \cdot \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$. Therefore, $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ are computationally indistinguishable due to the hardness of $\mathsf{MLWE}_{R,\nu,q,\sigma_{\mathtt{Open}}}$.

Claim 3: $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_3(\mathsf{ck})$ are statistically indistinguishable, $\mathcal{H}_3(\mathsf{ck})$ is defined as follows:

$$\left\{ (\vec{\mathfrak{m}}, \vec{\mathfrak{g}}, \vec{c}, \vec{t}, \vec{\tau}) \middle| \begin{array}{r} \underline{\vec{\boldsymbol{m}}_i \leftarrow \mathtt{R.Ecd}(\vec{0}, \mathfrak{s}_1)},\ \vec{\boldsymbol{\mu}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_1},\ \vec{\mathfrak{m}}_i \leftarrow \mathcal{U}(R_q^\mu) \text{ for } 0 \le i < k \\ \boldsymbol{c}_{j,0}, \ldots, \boldsymbol{c}_{j,k-1} \leftarrow \mathcal{U}(\mathcal{C}),\ \vec{g}_j \leftarrow \mathcal{U}(\mathbb{Z}_p^n) \\ \underline{\vec{\boldsymbol{g}}_{j,i} \leftarrow \mathtt{R.Ecd}(\vec{0}, \mathfrak{s}_2) \text{ for } 0 \le i < k,\ \vec{\boldsymbol{g}}_{j,k} \leftarrow \mathtt{R.Ecd}(\vec{g}_j, \mathfrak{s}_2)} \\ \text{The rest is same as } \mathcal{H}_2(\mathsf{ck}, \vec{m}) \end{array} \right\}$$

The difference between $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_3(\mathsf{ck})$ lies in how $\vec{\boldsymbol{m}}_i$, $\vec{\boldsymbol{g}}_{j,i}$, and $\vec{\boldsymbol{g}}_{j,i}$ are sampled. By Lem. 11, the distribution of $\left(\vec{\boldsymbol{g}}_{0,i} + \boldsymbol{c}_{0,i} \cdot \vec{\boldsymbol{m}}_i, \ldots, \vec{\boldsymbol{g}}_{\kappa-1,i} + \boldsymbol{c}_{\kappa-1,i} \cdot \vec{\boldsymbol{m}}_i\right)$ in $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_3(\mathsf{ck}, \vec{m})$ are within statistical distance 2ε, if the following condition holds.

$$\sqrt{\left(\mathfrak{s}_1^{-2} (P_\ell P_\ell^\top)^{-1} + \mathfrak{s}_2^{-2} \cdot \sum_{j=0}^{\kappa-1} C_j^\top (P_\ell P_\ell^\top)^{-1} C_j \right)^{-1}} \ge \eta_\varepsilon(P_\ell \mathbb{Z}^{d\ell})$$

By Lem. 2, the above condition is implied by $\mathfrak{s}_{\mathtt{Open}} \ge \frac{1}{b-1} \cdot \eta_\varepsilon(P_\ell \mathbb{Z}^{d\ell})$ since $\|C_j\|_2 = 1$ and $\left\|P^{-1}\right\|_2 \le \frac{1}{b-1}$. For the distribution of $\vec{\boldsymbol{g}}_{j,k}$, it is a randomized encoding of $\vec{g}_j + \sum_{i=0}^{k-1} C_{j,i} \vec{m}_i$ in $\mathcal{H}_2(\mathsf{ck}, \vec{m})$, while it corresponds to a randomized encoding of $\vec{g}_j$ in $\mathcal{H}_3(\mathsf{ck})$. We note that the distributions of $\vec{g}_j + \sum_{i=0}^{k-1} C_{j,i} \vec{m}_i$ and $\vec{g}_j$ are identical since $\vec{g}_j$ is uniformly sampled from $\mathbb{Z}_p^n$. Thus, we can regard $\vec{\boldsymbol{g}}_{j,1}$ in $\mathcal{H}_2$ and $\mathcal{H}_3$ as following an identical distribution. Therefore, $\mathcal{H}_2(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_3(\mathsf{ck})$ are within a statistical distance 2ε, which is $\mathsf{negl}(\lambda)$.

Claim 4: $\mathcal{H}_3(\mathsf{ck})$ and $\mathcal{H}_4(\mathsf{ck})$ are statistically indistinguishable, where $\mathcal{H}_4(\mathsf{ck})$ is defined as follows.

$$\left\{ (\vec{\mathfrak{m}}, \vec{\mathfrak{g}}, \vec{c}, \vec{t}, \vec{\tau}) \middle| \begin{array}{r} \vec{\boldsymbol{m}}_i \leftarrow \mathtt{R.Ecd}(\vec{0}, \mathfrak{s}_1),\ \vec{\boldsymbol{\mu}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_1},\ \vec{\mathfrak{m}}_i \leftarrow \mathcal{U}(R_q^\mu) \text{ for } 0 \le i < k \\ \boldsymbol{c}_{j,0}, \ldots, \boldsymbol{c}_{j,k-1} \leftarrow \mathcal{U}(\mathcal{C}),\ \vec{g}_j \leftarrow \mathcal{U}(\mathbb{Z}_p^n) \\ \underline{\vec{\boldsymbol{g}}_j \leftarrow \mathtt{R.Ecd}(\vec{g}_j, \sqrt{2}\mathfrak{s}_2),\ \vec{\boldsymbol{\gamma}}_j \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sqrt{2}\sigma_2}} \\ \vec{t}_j = \vec{\boldsymbol{g}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{m}}_i,\ \vec{\tau}_j = \vec{\boldsymbol{\gamma}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{\mu}}_i \\ \vec{\mathfrak{g}}_j = \mathbf{A}_0 \vec{t}_j + \mathbf{A}_1 \vec{\tau}_j - \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\mathfrak{m}}_i \text{ for } 0 \le j < \kappa \end{array} \right\}$$

We note that the above distribution is identical to the distribution of simulated transcript from $\mathcal{S}_{\mathtt{Open}}(\mathsf{ck})$. The difference between $\mathcal{H}_3(\mathsf{ck})$ and $\mathcal{H}_4(\mathsf{ck})$ lies in how $\vec{\boldsymbol{g}}_j$ and $\vec{\boldsymbol{\gamma}}_j$ are sampled, similar to Claim 1. Thus, following the proof of Claim 1, they are within a statistical distance of $16k\varepsilon$, which is $\mathsf{negl}(\lambda)$.

Therefore, we can conclude that $\mathcal{H}_0(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_4(\mathsf{ck})$ are computationally indistinguishable under the given conditions. Since $\mathcal{H}_0(\mathsf{ck}, \vec{m})$ corresponds to the distribution of real transcripts from $\Pi_{\mathtt{Open}}$, and $\mathcal{H}_4(\mathsf{ck})$ corresponds to the distribution of simulated transcripts from $\mathcal{S}_{\mathtt{Open}}$, there is no PPT adversary $\mathcal{A}$ that can distinguish between the two distributions with non-negligible probability. □

4 Lattice-based Polynomial Commitment Scheme

In this section, we present our polynomial commitment scheme based on the modified Ajtai commitment scheme and its POK protocol described in the previous section. We basically follow the construction in [BCC$^+$16, BG18], which builds a polynomial commitment scheme based on the Pedersen commitment scheme. Their main strategy in polynomial evaluation is to rearrange the coefficients of a polynomial to reduce communication cost while preserving secrecy. To be precise, let $h(X) = \sum_{i=0}^{N-1} h_i X^i \in \mathbb{Z}_p[X]$, and $N = nm$. If we define $\vec{h}_i = (h_{ni}, h_{ni+1}, \cdots, h_{n(i+1)-i})$ for $0 \leq i < m$, then $h(x) = \left\langle \sum_{i=0}^{m-1} x^{ni} \cdot \vec{h}_i,\ \vec{x} \right\rangle$ for $\vec{x} = (1, x, \ldots, x^{n-1})$ for all $x \in \mathbb{Z}_p$. Hence, it suffices to return $\sum_{i=0}^{m-1} x^{ni} \cdot \vec{h}_i$ as an evaluation proof, which is of size $O(n)$. However, it still contains information about each coefficient, which inhibits the evaluation hiding property. To resolve this issue, we pick random blinders $b_1, \ldots, b_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p)$, and define $h_m = (b_1, \ldots, b_{n-1}, 0)$, and $h_{m+1} = (0, -b_1, \ldots, -b_{n-1})$. Then, $\left\langle x \cdot \vec{h}_m + \vec{h}_{m+1}, \vec{x} \right\rangle = 0$, so it follows the distribution of random vectors whose inner product with $\vec{x}$ is 0. Thus, sending $\sum_{i=0}^{m-1} x^{ni} \cdot \vec{h}_i + x \cdot \vec{h}_m + \vec{h}_{m+1}$ achieves the desired goal of the evaluation proof, revealing nothing about the coefficients but the evaluation result $h(x)$.

However, directly adapting their technique to the Ajtai commitment scheme is not straightforward, as it requires converting all the proof techniques based on the properties of $\mathbb{Z}_p$ into the setting of R. First, there is an issue with the extractability of the polynomial commitment scheme. In [BCC$^+$16, BG18], extractability is directly attained from evaluation proofs at N different points by computing the inverse of the Vandermonde matrix over $\mathbb{Z}_p$. However, in the Ajtai commitment scheme, such an extraction technique is not allowed since encoded messages and randomness are elements of R. Hence, we additionally attach the proof of opening knowledge to provide extractability. Second, the aforementioned blinding technique is also not converted straightforwardly due to the difference between $\mathbb{Z}_p$ and R. To resolve this, we extend the randomized encoding in the previous method so that blinders sampled from proper discrete Gaussian distributions effectively hide information of coefficients. In the rest of this section, we present more details on how we address these issues arising from the difference between $\mathbb{Z}_p$ and R.

4.1 Our Polynomial Commitment

Our polynomial commitment scheme is defined as follows.

- $\texttt{PC.Setup}(1^\lambda, N) \to \mathsf{ck}$: Given a security parameter λ and a polynomial degree upper bound $N = mn$, it generates a commitment key $\mathsf{ck} = (\mathbf{A}_0, \mathbf{A}_1)$ where $n = d\ell/r$, $\mathbf{A}_0 \leftarrow \mathcal{U}(R_q^{\mu\times\ell})$, and $\mathbf{A}_1 = [\mathbf{A}_1'|\mathbf{I}_\mu] \in R_q^{\mu\times(\mu+\nu)}$ with $\mathbf{A}_1' \leftarrow \mathcal{U}(R_q^{\mu\times\nu})$.

- $\texttt{PC.Com}(\mathsf{ck}, h(X)) \to (\vec{\mathfrak{h}}, \delta)$: Given a polynomial $h(X) = \sum_{i=0}^{N-1} h_i X^i \in \mathbb{Z}_p[X]$, it generates a commitment $\vec{\mathfrak{h}}$ and an opening $\delta = (\vec{\boldsymbol{h}}, \vec{\boldsymbol{\eta}})$ as follows, where $\vec{\mathfrak{h}} = \vec{\mathfrak{h}}_0 \| \cdots \| \vec{\mathfrak{h}}_{m+1}$, $\vec{\boldsymbol{h}} = \vec{\boldsymbol{h}}_0 \| \cdots \| \vec{\boldsymbol{h}}_{m+1}$, and $\vec{\boldsymbol{\eta}} = \vec{\boldsymbol{\eta}}_0 \| \cdots \| \vec{\boldsymbol{\eta}}_{m+1}$:
  1. Define $\vec{h}_i = (h_{ni}, \ldots, h_{(n+1)i-1})$ and $0 \le i < m$.
  2. Sample $b_1, \ldots, b_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p)$, define $\vec{h}_m = (b_1, \ldots, b_{n-1}, 0)$ and $\vec{h}_{m+1} = (0, -b_1, \ldots, -b_{n-1})$.
  3. $\vec{\boldsymbol{h}}_i \leftarrow \texttt{R.Ecd}(\vec{h}_i; \mathfrak{s}_1)$, $\vec{\boldsymbol{\eta}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_1}$, and $\vec{\mathfrak{h}}_i = \mathbf{A}_0\vec{\boldsymbol{h}}_i + \mathbf{A}_1\vec{\boldsymbol{\eta}}_i \pmod q$ for $0 \le i \le m$.
  4. $\vec{\boldsymbol{h}}_{m+1} \leftarrow \texttt{R.Ecd}(\vec{h}_{m+1}, \sqrt{m+2} \cdot \mathfrak{s}_3)$, $\vec{\boldsymbol{\eta}}_{m+1} \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sqrt{m+2}\cdot\sigma_3}$, and $\vec{\mathfrak{h}}_{m+1} = \mathbf{A}_0\vec{\boldsymbol{h}}_{m+1} + \mathbf{A}_1\vec{\boldsymbol{\eta}}_{m+1} \pmod q$.

- $\texttt{PC.Open}(\mathsf{ck}, \vec{\mathfrak{h}}, h(X), \delta) \to b$: Given a commitment $\vec{\mathfrak{h}}$, an opening $\delta = (\vec{\boldsymbol{h}}, \vec{\boldsymbol{\eta}})$, and polynomial $h(X)$ of degree $< N$, output 1 if it satisfies the following conditions; otherwise, output 0:
  1. $\left\| 2\vec{\boldsymbol{h}}_i \| 2\vec{\boldsymbol{\eta}}_i \right\|_2 \overset{?}{\le} 2d\beta_{\texttt{PC.Open}}$ for $0 \le i \le m$
  2. $\left\| 2\vec{\boldsymbol{h}}_{m+1} \| 2\vec{\boldsymbol{\eta}}_{m+1} \right\|_2 \overset{?}{\le} 2\beta_{\texttt{PC}}$, where $\beta_{\texttt{PC}} = \beta_{\texttt{PC.Eval}} + \frac{(b+1)(m+1)dr}{2} \cdot \beta_{\texttt{PC.Open}}$
  3. $\vec{\mathfrak{h}}_i \overset{?}{=} \mathbf{A}_0\vec{\boldsymbol{h}}_i + \mathbf{A}_1\vec{\boldsymbol{\eta}}_i \pmod q$ for $0 \le i \le m+1$
  4. $h(X) \overset{?}{=} \sum_{i=0}^{m-1} X^{ni} \cdot \left\langle \vec{h}_i, \vec{X} \right\rangle + X \cdot \left\langle \vec{h}_m, \vec{x} \right\rangle + \left\langle \vec{h}_{m+1}, \vec{X} \right\rangle \pmod p$, where $\vec{X} = (1, X, \cdots, X^{n-1})$, and $\vec{h}_i = \frac{p+1}{2} \cdot \texttt{Dcd}(2\vec{\boldsymbol{h}}_i) \pmod p$ for $0 \le i \le m+1$

- $\texttt{PC.Eval}(x, \delta) \to (y, \rho)$: Given an opening $\delta = (\vec{\boldsymbol{h}}, \vec{\boldsymbol{\eta}})$ and an evaluation point $x \in \mathbb{Z}_p$, it generates an evaluation result y and an evaluation proof ρ as follows:
  1. Compute $\vec{\boldsymbol{e}} = \sum_{i=0}^{m-1} \texttt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{h}}_i + \texttt{Ecd}(x) \cdot \vec{\boldsymbol{h}}_m + \vec{\boldsymbol{h}}_{m+1}$ and $\vec{\boldsymbol{\varepsilon}} = \sum_{i=0}^{m-1} \texttt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{\eta}}_i + \texttt{Ecd}(x) \cdot \vec{\boldsymbol{\eta}}_m + \vec{\boldsymbol{\eta}}_{m+1}$.
  2. Return an evaluation proof $\rho = (\vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}})$, and an evaluation result $y = \left\langle \texttt{Dcd}(\vec{\boldsymbol{e}}), (1, x, \ldots, x^{n-1}) \right\rangle \pmod p$.

- $\texttt{PC.Verify}(\mathsf{ck}, \vec{\mathfrak{h}}, x, y, \rho) \to b$: Given a commitment $\vec{\mathfrak{h}}$, an evaluation point x, an evaluation result y, and a proof $\rho = (\vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}})$, output 1 if $\|\vec{\boldsymbol{e}} \| \vec{\boldsymbol{\varepsilon}}\|_2 \le \beta_{\texttt{PC.Eval}}$, $y = \left\langle \texttt{Dcd}(\vec{\boldsymbol{e}}), (1, x, \ldots, x^{n-1}) \right\rangle \pmod p$, and $\mathbf{A}_0\vec{\boldsymbol{e}} + \mathbf{A}_1\vec{\boldsymbol{\varepsilon}} = \sum_{i=0}^{m-1} \texttt{Ecd}(x^{ni}) \cdot \vec{\mathfrak{h}}_i + \texttt{Ecd}(x) \cdot \vec{\mathfrak{h}}_m + \vec{\mathfrak{h}}_{m+1} \pmod q$; otherwise, output 0.

The above polynomial commitment scheme is computationally hiding if $\mathsf{MLWE}_{R,\nu,q,\sigma_1}$ and $\mathsf{MLWE}_{R,\nu,q,\sigma_3}$ are hard, and computationally binding if $\mathsf{MSIS}_{R,\mu,q,4\beta_{\mathtt{PC}}}$ is hard. We also define a POK protocol $\Pi_{\mathtt{PC.Open}}$ for the polynomial commitment as $\Pi_{\mathtt{Open}}$ in Fig. 1, with the prover's input $(\mathsf{ck}, \vec{\mathfrak{h}}_0\|\cdots\|\vec{\mathfrak{h}}_m, \vec{\boldsymbol{h}}_0\|\cdots\|\vec{\boldsymbol{h}}_m, \vec{\boldsymbol{\eta}}_0\|\cdots\|\vec{\boldsymbol{\eta}}_m)$, the verifier's input $(\mathsf{ck}, \vec{\mathfrak{h}}_0\|\cdots\|\vec{\mathfrak{h}}_m)$, and replacing the value $\beta_{\mathtt{Open}}$ by $\beta_{\mathtt{PC.Open}}$. The evaluation binding property is ensured by the hardness of the MSIS problem as follows.

Theorem 5 (Evaluation Binding). *The polynomial commitment scheme* ($\mathtt{PC.Setup}$, $\mathtt{PC.Com}$, $\mathtt{PC.Open}$, $\mathtt{PC.Eval}$, $\mathtt{PC.Verify}$) *is evaluation biding if* $\mathsf{MSIS}_{R,\mu,q,2\beta_{\mathtt{PC.Eval}}}$ *is hard.*

Proof. Let $\mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda, N)$, and $\mathcal{A}(\mathsf{ck})$ be a PPT adversary that outputs $(c, x, y, y', \rho, \rho')$. Let α be the probability that $\mathtt{PC.Verify}(\mathsf{ck}, c, x, y, \rho) = 1$, $\mathtt{PC.Verify}(\mathsf{ck}, c, x, y', \rho') = 1$, and $y \neq y'$. Then, for $\rho = (\vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}})$ and $\rho' = (\vec{\boldsymbol{e}}', \vec{\boldsymbol{\varepsilon}}')$, it holds that $\mathbf{A}_0(\vec{\boldsymbol{e}} - \vec{\boldsymbol{e}}') + \mathbf{A}_1(\vec{\boldsymbol{\varepsilon}} - \vec{\boldsymbol{\varepsilon}}') = \mathbf{0}$, and $\left\|(\vec{\boldsymbol{e}} - \vec{\boldsymbol{e}}')\|(\vec{\boldsymbol{\varepsilon}} - \vec{\boldsymbol{\varepsilon}}')\right\|_2 \leq 2\beta_{\mathtt{PC.Eval}}$. Also, $\vec{\boldsymbol{e}} \neq \vec{\boldsymbol{e}}'$ since $y = \left\langle \mathtt{Dcd}(\vec{\boldsymbol{e}}), (1, x, \ldots, x^{n-1}) \right\rangle$, $y' = \left\langle \mathtt{Dcd}(\vec{\boldsymbol{e}}'), (1, x, \ldots, x^{n-1}) \right\rangle$, and $y \neq y'$. Then, $\mathcal{A}$ solves $\mathsf{MSIS}_{R,\mu,q,2\beta_{\mathtt{PC.Eval}}}$ with probability α. Therefore, α should be $\mathsf{negl}(\lambda)$ due to the hardness of $\mathsf{MSIS}_{R,\mu,q,2\beta_{\mathtt{PC.Eval}}}$. □

For the evaluation hiding property, we extend the randomized encoding technique to our modified Ajtai commitment scheme. The core idea is that we sample the blinders $\vec{\boldsymbol{h}}_m$ and $\vec{\boldsymbol{h}}_{m+1}$ via randomized encoding of $\vec{\boldsymbol{h}}_m$ and $\vec{\boldsymbol{h}}_{m+1}$. This offers simulatability not only in the encoded state in R but also in the decoded state in $\mathbb{Z}_p$. The proof follows a similar workflow as the simulatability proof for the POK protocol $\Pi_{\mathtt{Open}}$.

Theorem 6 (Evaluation Hiding). *The polynomial commitment scheme* ($\mathtt{PC.Setup}$, $\mathtt{PC.Com}$, $\mathtt{PC.Open}$, $\mathtt{PC.Eval}$, $\mathtt{PC.Verify}$) *is evaluation hiding if it satisfies the followings for* $\varepsilon = \mathsf{negl}(\lambda)$*:*

- $\frac{\mathfrak{s}_3}{\sqrt{2}}, \mathfrak{s}_{\mathtt{PC.Eval}} \geq \frac{1}{b-1} \cdot \eta_\varepsilon((P \otimes I_\ell)\mathbb{Z}^{d\ell})$*, where* $\mathfrak{s}_{\mathtt{PC.Eval}} = \left(\mathfrak{s}_1^{-2} + \left(\frac{(b+1)r}{2}\right)^2 \mathfrak{s}_3^{-2}\right)^{-1/2}$
- $\frac{\sigma_3}{\sqrt{2}}, \frac{\sigma_{\mathtt{PC.Eval}}}{\sqrt{2}} \geq \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$*, where* $\sigma_{\mathtt{PC.Eval}} = \frac{1}{\sqrt{2}} \cdot \left(\sigma_1^{-2} + \left(\frac{(b+1)r}{2}\right)^2 \sigma_3^{-2}\right)^{-1/2}$
- *The advantage of* $\mathsf{MLWE}_{R,\nu,q,\sigma_{\mathtt{PC.Eval}}}$ *is* $\mathsf{negl}(\lambda)$

Proof. See Appendix. C

The security of the POK protocol $\Pi_{\mathtt{PC.Open}}$ for the polynomial commitment scheme is as follows. The basic idea of the proof is similar to the case of $\Pi_{\mathtt{Open}}$.

Theorem 7 (Completeness). *The proof of opening knowledge protocol* $\Pi_{\mathtt{PC.Open}}$ *has completeness if it satisfies the following conditions for* $\varepsilon = \mathsf{negl}(\lambda)$*:*

- $\mathfrak{s}_1, \sqrt{m+2} \cdot \mathfrak{s}_2, \sigma_1, \sqrt{m+2} \cdot \sigma_2 \geq \eta_\varepsilon(\mathbb{Z}^d)$
- $\beta_{\mathtt{PC.Open}} = \sqrt{\left(\left((m+1)\cdot\sigma_1 + \sqrt{m+2}\cdot\sigma_2\right)^2 \cdot (\mu+\nu) + (b+1)^2 \cdot \left((m+1)\cdot\mathfrak{s}_1 + \sqrt{m+2}\cdot\mathfrak{s}_2\right)^2 \cdot \ell\right) d}$

– $\beta_{\texttt{PC.Eval}} = \sqrt{\left(\left((m+1)(b+1)r/2\cdot\sigma_1+\sqrt{m+2}\cdot\sigma_3\right)^2\cdot(\mu+\nu)+(b+1)^2\cdot\left((m+1)(b+1)r/2\cdot\mathfrak{s}_1+\sqrt{m+2}\cdot\mathfrak{s}_3\right)^2\cdot\ell\right)d}$

Proof. Following the proof of Theorem 2, we obtain the following upper bounds. For $\beta_{\texttt{PC.Open}}$, setting $k = m+1$ in Thm. 2 yields the given upper bound. For $\beta_{\texttt{PC.Eval}}$, we can estimate upper bounds for $\vec{\boldsymbol{e}}$ and $\vec{\boldsymbol{\varepsilon}}$ as follows

$$\|\vec{\boldsymbol{e}}\|_2 \le \sum_{i=0}^{m}\frac{(b+1)r}{2}\left\|\vec{\boldsymbol{h}}_i\right\|_2 + \left\|\vec{\boldsymbol{h}}_{m+1}\right\|_2, \quad \|\vec{\boldsymbol{\varepsilon}}\|_2 \le \sum_{i=0}^{m}\frac{(b+1)r}{2}\|\vec{\boldsymbol{\eta}}_i\|_2 + \left\|\vec{\boldsymbol{\eta}}_{m+1}\right\|_2$$

since $\left\|\texttt{Ecd}(x^k)\right\| \le \frac{(b+1)r}{2}$ for any k. Then, applying upper bounds for $\vec{\boldsymbol{h}}_i$ and $\vec{\boldsymbol{\eta}}_i$ estimated from Lemma 5 yields the given upper bound for $\beta_{\texttt{PC.Eval}}$ □

Since we cannot compute the inverse Vandermonde matrix in R in general, we bypass the problem in extractability by additionally offering the proof of opening knowledge for the commitments. Then, our extractor extracts openings by rewinding the POK protocol, unlike [BCC+16, BG18].

Theorem 8 (Soundness). *The proof of opening knowledge protocol $\Pi_{\texttt{PC.Open}}$ satisfies soundness.*

Proof. Let $\mathsf{ck} \leftarrow \texttt{PC.Setup}(1^\lambda, N)$, and $\mathcal{P}^*$ be a PPT adversary which works as follows:

$$(\vec{\mathfrak{h}}, x, y, \vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}}) \leftarrow \mathcal{P}^*(\mathsf{ck}),\ (\pi, b) \leftarrow \left\langle \mathcal{P}^*(\mathsf{ck}, \vec{\mathfrak{h}}), \mathcal{V}(\mathsf{ck}, \vec{\mathfrak{h}}) \right\rangle$$

Let α be the probability that $b = 1$ and $\texttt{PC.Verify}(\mathsf{ck}, \vec{\mathfrak{h}}, x, y, \vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}}) = 1$. Using Lem. 5 in [BBC+18], there exists an extractor $\mathcal{E}^{\mathcal{P}^*}$ that can extract $(\vec{\boldsymbol{h}}_i', \vec{\boldsymbol{\eta}}_i')$ such that $\left\|\vec{\boldsymbol{h}}_i' \| \vec{\boldsymbol{\eta}}_i'\right\|_2 \le 2d\beta_{\texttt{PC.Open}}$ and $\mathbf{A}_0\vec{\boldsymbol{h}}_i' + \mathbf{A}_1\vec{\boldsymbol{\eta}}_i' = 2\vec{\mathfrak{h}}_i$ for $0 \le i \le m$ in expected time $poly(\lambda)/\alpha$. Let $\vec{\boldsymbol{h}}_{m+1}' = 2\vec{\boldsymbol{e}} - \sum_{i=0}^{m-1}\texttt{Ecd}(x^{ni})\cdot\vec{\boldsymbol{h}}_i' - \texttt{Ecd}(x)\cdot\vec{\boldsymbol{h}}_m'$, and $\vec{\boldsymbol{\eta}}_{m+1}' = 2\vec{\boldsymbol{\varepsilon}} - \sum_{i=0}^{m-1}\texttt{Ecd}(x^{ni})\cdot\vec{\boldsymbol{\eta}}_i' - \texttt{Ecd}(x)\cdot\vec{\boldsymbol{\eta}}_m'$. Then, it holds that

$$\left\|\vec{\boldsymbol{h}}_{m+1}' \| \vec{\boldsymbol{\eta}}_{m+1}'\right\|_2 \le 2\beta_{\texttt{PC.Eval}} + (b+1)(m+1)dr\cdot\beta_{\texttt{PC.Open}} = 2\beta_{\texttt{PC}}$$

since $\left\|\texttt{Ecd}(x^k)\right\| \le \frac{(b+1)r}{2}$ for any k. Also, it holds that

$$\mathbf{A}_0\vec{\boldsymbol{h}}_{m+1}' + \mathbf{A}_1\vec{\boldsymbol{\eta}}_{m+1}' = 2\vec{\mathfrak{h}}_{m+1} \pmod q$$

since $\mathbf{A}_0\vec{\boldsymbol{e}} + \mathbf{A}_1\vec{\boldsymbol{\varepsilon}} = \sum_{i=0}^{m-1}\texttt{Ecd}(x^{ni})\cdot\vec{\mathfrak{h}}_i + \texttt{Ecd}(x)\cdot\vec{\mathfrak{h}}_m + \vec{\mathfrak{h}}_{m+1} \pmod q$ from $\texttt{PC.Verify}(\mathsf{ck}, \vec{\mathfrak{h}}, x, y, \vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}}) = 1$. Then, for $\delta = (\frac{q+1}{2}\cdot\vec{\boldsymbol{h}}', \frac{q+1}{2}\cdot\vec{\boldsymbol{\eta}}')$ and $h(X) = \sum_{i=0}^{m-1}X^{ni}\cdot\left\langle\vec{h}_i, \vec{X}\right\rangle + X\cdot\left\langle\vec{h}_m, \vec{X}\right\rangle + \left\langle\vec{h}_{m+1}, \vec{X}\right\rangle$, it holds that $y = h(x)$ and $\texttt{PC.Open}(\mathsf{ck}, \vec{\mathfrak{h}}, h(X), \delta) = 1$, where $\vec{X} = (1, X, \cdots, X^{n-1})$, and $\vec{h}_i = \frac{p+1}{2}\cdot\texttt{Dcd}(\vec{\boldsymbol{h}}_i') \pmod p$ for $0 \le i \le m+1$. Thus, in case of non-negligible α, there exists an expected polynomial time extractor that outputs $(h(X), \delta)$ such that $\texttt{PC.Open}(\mathsf{ck}, \vec{\mathfrak{h}}, h(X), \delta) = 1$ and $y = h(x)$. □

Theorem 9 (Simulatability). *The proof of opening knowledge protocol* $\Pi_{\mathtt{PC.Open}}$ *satisfies simulatability if it satisfies the followings for* $\varepsilon = \mathsf{negl}(\lambda)$*:*

- $\frac{\mathfrak{s}_2}{\sqrt{2}}, \frac{\mathfrak{s}_3}{\sqrt{2}}, \mathfrak{s}_{\mathtt{PC.Open}} \geq \frac{1}{b-1} \cdot \eta_\varepsilon((P \otimes I_\ell)\mathbb{Z}^{d\ell})$*, where* $\mathfrak{s}_{\mathtt{PC.Open}} = \left(\mathfrak{s}_1^{-2} + \kappa\mathfrak{s}_2^{-2} + \left(\frac{(b+1)r}{2}\right)^2 \mathfrak{s}_3^{-2}\right)^{-1/2}$
- $\frac{\sigma_2}{\sqrt{2}}, \frac{\sigma_3}{\sqrt{2}}, \frac{\sigma_{\mathtt{PC.Open}}}{\sqrt{2}} \geq \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$*, where* $\sigma_{\mathtt{PC.Open}} = \frac{1}{\sqrt{2}} \cdot \left(\sigma_1^{-2} + \kappa\sigma_2^{-2} + \left(\frac{(b+1)r}{2}\right)^2 \sigma_3^{-2}\right)^{-1/2}$
- *The advantage of* $\mathsf{MLWE}_{R,\nu,q,\sigma_{\mathtt{PC.Open}}}$ *is* $\mathsf{negl}(\lambda)$

$\mathcal{S}_{\mathtt{PC.Open}}(\mathsf{ck}, x, y)$

1 : $\vec{\mathfrak{h}}_0, \ldots, \vec{\mathfrak{h}}_m \leftarrow \mathcal{U}(R_q^\mu),\ \boldsymbol{c}_{0,0}, \ldots, \boldsymbol{c}_{\kappa-1,m} \leftarrow \mathcal{U}(\mathcal{C})$

2 : $\vec{\boldsymbol{h}}_0, \ldots, \vec{\boldsymbol{h}}_m \leftarrow \mathtt{R.Ecd}(\vec{0}, \mathfrak{s}_1),\ \vec{\boldsymbol{\eta}}_0, \ldots, \vec{\boldsymbol{\eta}}_m \leftarrow \mathcal{D}_{\mathbb{Z}^d, \sigma_1}$

3 : $u_1, \ldots, u_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p),\ u_0 = y - \sum_{j=1}^{n} u_j x^j,\ \vec{u} = (u_0, \ldots, u_{n-1})$

4 : $\vec{\boldsymbol{h}}_{m+1} \leftarrow \mathtt{R.Ecd}(\vec{u}, \sqrt{m+2} \cdot \mathfrak{s}_3),\ \vec{\boldsymbol{\eta}}_{m+1} \leftarrow \mathcal{D}_{\mathbb{Z}^d, \sqrt{m+2}\cdot\sigma_3}$

5 : $\vec{\boldsymbol{e}} = \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{h}}_i + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{h}}_m + \vec{\boldsymbol{h}}_{m+1}$

6 : $\vec{\boldsymbol{\varepsilon}} = \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{\eta}}_i + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{\eta}}_m + \vec{\boldsymbol{\eta}}_{m+1}$

7 : $\vec{\mathfrak{h}}_{m+1} = \mathbf{A}_0\vec{\boldsymbol{e}} + \mathbf{A}_1\vec{\boldsymbol{\varepsilon}} - \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\mathfrak{h}}_i - \mathtt{Ecd}(x) \cdot \vec{\mathfrak{h}}_m \pmod q$

8 : $\vec{g}_j \leftarrow \mathcal{U}(\mathbb{Z}_p^n),\ \vec{\boldsymbol{g}}_j \leftarrow \mathtt{R.Ecd}(\vec{g}_j, \sqrt{m+2} \cdot \mathfrak{s}_2),\ \boldsymbol{\gamma}_j \leftarrow \mathcal{D}_{\mathbb{Z}^d, \sqrt{m+2}\cdot\sigma_2}$ for $0 \leq j < \kappa$

9 : $\vec{\boldsymbol{t}}_j = \vec{\boldsymbol{g}}_j + \sum_{i=0}^{m} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{h}}_i,\ \vec{\boldsymbol{\tau}}_j = \vec{\boldsymbol{\gamma}}_j + \sum_{i=0}^{m} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{\eta}}_i$ for $0 \leq j < \kappa$

10 : $\vec{\mathfrak{g}}_j = \mathbf{A}_0\vec{\boldsymbol{t}}_j + \mathbf{A}_1\vec{\boldsymbol{\tau}}_j - \sum_{i=0}^{m} \boldsymbol{c}_{j,i} \cdot \vec{\mathfrak{h}}_i \pmod q$ for $0 \leq j < \kappa$

11 : Output $(\vec{\mathfrak{h}}, \vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}}, \vec{\mathfrak{g}}, \vec{\boldsymbol{t}}, \vec{\boldsymbol{\tau}})$

Fig. 3. Simulator for $\Pi_{\mathtt{PC.Open}}$

Proof. Let $\mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda)$, and $(h(X), x)$ be a polynomial and a point chosen by a PPT adversary $\mathcal{A}$. We prove that the algorithm $\mathcal{S}_{\mathtt{PC.Open}}$ in Fig. 3 is an efficient simulator for the distribution of evaluation proofs generated by `PC.Eval`, and real transcripts generated from $\Pi_{\mathtt{PC.Open}}$. The proof can be directly derived from combining Thm. 4 and Thm. 6 together. □

4.2 Performance

We now estimate the complexity of our polynomial commitment scheme when committing a polynomial of degree less than $N = nm$.

Prover Complexity. We analyze the prover's complexity in generating commitments, evaluation proofs, and proofs of opening knowledge. For generating each commitment $\vec{\mathfrak{h}}_i = \mathbf{A}_0\vec{\boldsymbol{h}}_i + \mathbf{A}_1\vec{\boldsymbol{\eta}}_i$ for $0 \leq i \leq m+1$, it takes $O(m \cdot \mu(\mu+\nu+\ell))$ ring arithmetic operations over R_q. For generating an evaluation proof, it takes $O(m \cdot (\mu+\nu+\ell))$ ring arithmetic operations over R_q and $O(n)$ arithmetic operations over $\mathbb{Z}_p$. Finally, for the proof of opening knowledge, it takes $O(m \cdot (\mu+\nu+\ell))$ ring arithmetic operations over R_q Assuming polynomial multiplications in R_q are done in $O(d \log d)$ complexity, it takes $O(md \log d \cdot \mu(\mu+\nu+\ell))$ computational complexity in total. We note that the quantity μd affects the security of MSIS and νd affects the security of MLWE and they follow $O(\mathsf{polylog}(N))$. Then, the overall prover's complexity follows $O(\frac{1}{d} \log d \cdot N \cdot \mathsf{polylog}(N))$, which is essentially $\tilde{O}(N)$ since d can be considered independent of N. However, taking larger values of d improves the prover's concrete performance.

Verifier Complexity. We analyze the verifier's complexity in verifying evaluation proofs and proofs of opening knowledge. For verifying evaluation proofs, it takes $O(\mu(\mu + \nu + \ell + m))$ arithmetic operations over R_q and $O(n)$ arithmetic operations over $\mathbb{Z}_p$. For verifying proofs of opening knowledge, it takes $O(\mu(\mu + \nu + \ell + m))$ arithmetic operations over R_q. Then, the overall verifier's complexity follows $O((n/d + m) \cdot \mathsf{polylog}(N))$ using the same assumption as the prover's complexity.

Communication Complexity. We analyze the communication cost in sending commitments, evaluation proofs, and proofs of opening knowledge. For commitments, it sends $O(m\mu)$ elements in R_q. For evaluation proofs, it sends $O(\mu+\nu+\ell)$ elements in R_q. Finally, for proofs of opening knowledge, it sends $O(\mu + \nu + \ell)$ elements in R_q. Hence, the communication cost is $O(d(m\mu + \nu + \ell)) = O(m \cdot \mathsf{polylog}(N) + n)$ in total.

Therefore, setting $O(n) = O(m) = O(\sqrt{N})$ results in verifier and communication complexity in $\tilde{O}(\sqrt{N})$, which is sublinear in N.

5 Experimental Results

In this section, we present a proof-of-concept implementation of our sublinear polynomial commitment scheme and demonstrate its concrete performance. We implement our protocol using the Rust programming language and convert the interactive protocol into a non-interactive one using the Fiat-Shamir transform. [2] For the functionality of the random oracle, we use the SHA-3 hash function. All experiments were performed with a single thread on a machine with an Intel(R) Xeon(R) Platinum 8268 CPU running at 2.90GHz and 384GB of RAM.

5.1 Parameter Setting

We summarize all parameters that appear in our protocol in Table. 1. Firstly, we set the message modulus $p = b^r + 1$ to be 255-bit sized primes using $b = 63388$

[2] The source code is available at `https://github.com/SNUCP/celpc`

Type	Parameter	Description
Binding & Hiding	q	commitment modulus
	d	ring dimension
	μ	MSIS rank
	ν	MLWE rank
	$\mathfrak{s}_1, \mathfrak{s}_2, \mathfrak{s}_3, \sigma_1, \sigma_2, \sigma_3$	width parameters
Soundness	b	base
	r	digit lengths
	p	message modulus ($= b^r + 1$)
	κ	# of repetition ($= \lceil \lambda / \log(2d) \rceil$
Proof Size	n	# of messages per commitments
	ℓ	ring element per commitment ($= rn/d$)
	m	# of commitment
	N	input polynomial degree ($= nm$)

Table 1. Parameter set

and $r = 16$. Then, we set the parameters that determine the security of the binding and hiding properties of our polynomial commitment scheme, namely $q, n, \mu, \nu, \mathfrak{s}_1, \mathfrak{s}_2, \mathfrak{s}_3, \sigma_1, \sigma_2, \sigma_3$. We recall that, to ensure the binding and hiding properties in our protocol, $\mathsf{MSIS}_{R,\mu,q,4\beta_{\mathtt{PC}}}$ and $\mathsf{MLWE}_{R,\nu,q,\sigma_{\mathtt{PC.Open}}}$ should be hard. We set the width parameters as follows for $P_\ell := P \otimes I_\ell$, so that they satisfy the conditions in Thm. 9.

$$\mathfrak{s}_1 \geq \frac{\sqrt{3}}{b-1} \cdot \eta_\varepsilon(P_\ell \mathbb{Z}^{d\ell}), \quad \mathfrak{s}_2 \geq \frac{\sqrt{3\kappa}}{b-1} \cdot \eta_\varepsilon(P_\ell \mathbb{Z}^{d\ell}), \quad \mathfrak{s}_3 \geq \frac{\sqrt{3}(b+1)r}{2b-2} \cdot \eta_\varepsilon(P_\ell \mathbb{Z}^{d\ell})$$
$$\sigma_1 \geq 2\sqrt{3} \cdot \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)}), \quad \sigma_2 \geq 2\sqrt{3\kappa} \cdot \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)}), \quad \sigma_3 \geq \sqrt{3}(b+1)r \cdot \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$$

We estimate the upper bounds for the smoothing parameters using Lem. 1 for $\varepsilon = 2^{-\lambda}$ and $\lambda = 128$. Once the width parameters are determined, the values of q, d, μ, ν determine the hardness of the MSIS and MLWE problem. To estimate their hardness, we compute the root Hermite factor δ and set it to be approximately 1.005. As a result, we set $\log q \approx 112$, $d = 2048$, $\mu = 1$, and $\nu = 2$. We note that d is set to be the largest possible value for the concrete efficiency of the prover and verifier. Finally, we adjust the parameters n, ℓ, m to yield the smallest proof size depending on the input polynomial degree N, as summarized in Table 2.

5.2 Implementation Technique

In this subsection, we present optimization techniques used in our implementation.

Ring Arithmetic. For efficient ring arithmetic over R_q, we use Residue Number System (RNS) representation and Number Theoretic Transform (NTT). We

N	n	m	ℓ
2^{19}	2^{12}	2^7	2^5
2^{21}	2^{13}	2^8	2^6
2^{23}	2^{14}	2^9	2^7
2^{25}	2^{15}	2^{10}	2^8

Table 2. Parameters for the smallest proof size

recall that we use a commitment modulus q of size 112 bits, which cannot be covered by 64-bit native integer operations. Hence, we set $q = q_1 \cdot q_2$, where q_1 and q_2 are coprime 56-bit integers so that $R_q \cong R_{q_1} \times R_{q_2}$ by the Chinese remainder theorem. Then, elements in R_q can be represented as a pair of elements in R_{q_1} and R_{q_2}, which is often referred to as a RNS representation. Also, arithmetic over R_q can be instantiated with arithmetic over R_{q_1} and R_{q_2} due to the isomorphism, which can be implemented with 64-bit integer operations. For efficient multiplication in R_{q_1} and R_{q_2}, we use NTT operations, which reduce the complexity of polynomial multiplication from $O(d^2)$ to $O(d \log d)$, by setting q_1 and q_2 to be primes such that $q_1, q_2 = 1 \pmod{2d}$.

Discrete Gaussian Distribution. To sample from discrete Gaussian distributions, we utilize Karney's sampler [Kar16]. For fixed Gaussian distributions such as distributions for randomness, we round the width parameter up to the nearest integer as Karney's sampler requires exact integer arithmetic to sample from the correct distribution. However, in randomized encoding, we cannot use this trick since the sample space, a coset of the integer lattice, keeps changing depending on the input. In this case, we utilize a slight variation of Karney's sampler using double-precision floating point values, proposed in [Wal19].

Commitment Compression. We use the compression technique in [BG14] to reduce the size of commitments. Instead of sending the full size of commitments $\vec{\mathfrak{h}}_0, \cdots, \vec{\mathfrak{h}}_m$, we can omit the lower D bits of them while maintaining the binding property. To be precise, suppose we omit the lower D bits of commitments. Then, it increases the bound of $\beta_{\mathtt{PC.Open}}$ in the POK protocol $\Pi_{\mathtt{PC.Open}}$ by $(m+1)2^D \cdot \sqrt{\mu d}$, but reduces the commitment size by a factor of $(\log q - D)/\log q$. Hence, in estimating proof size, we omit the lower D bits of commitments, where D is set to 24, the largest value that does not alter the MSIS parameter μ.

5.3 Benchmark Results

We present the benchmark results for our polynomial commitment scheme in Table 4 together with other post-quantum secure polynomial commitment schemes from Brakedown [GLS+23] and FRI [BSBHR18] for the 255-bit base field. Brakedown aims to provide faster prover's performance, but it provides square root

proof size like ours. FRI aims to provide a compact proof size, which is polylogarithmic in size with respect to N, but it has a slower prover's performance. We note that both of their constructions are based on cryptographic hash functions. To provide comparison with them, we measure the elapsed time for the prover and the verifier, as well as the communication cost, which includes the size of commitments, evaluation proofs, and the proof of knowledge. The benchmark results for other polynomial commitment schemes are referenced from Fig. 8 in [GLS+23].

N		2^{19}	2^{21}	2^{23}	2^{25}
Prover(seconds)	Ours	4.96	18.3	70.3	277
	Ours w/o ZK	0.97	3.47	13.0	50.9
	Brakedown	0.60	2.41	9.85	39.2
	FRI	6.42	31.9	60.8	353
Verifier(seconds)	Ours	0.14	0.27	0.53	1.07
	Brakedown	0.15	0.30	0.61	0.70
	FRI	0.03	0.03	0.04	0.04
Communication(MB)	Ours	6.07	11.9	23.6	47.5
	Brakedown	10.0	15.8	27.1	49.2
	FRI	0.39	0.49	0.61	0.74

Table 3. Benchmark results

For the prover's performance, we measure the elapsed time both with and without randomized encoding, since the benchmark results in [GLS+23] only measured in the non-zero-knowledge setting. Considering these differences, our scheme provides comparable performance with Brakedown and faster performance compared to FRI in the non-zero knowledge setting. For the verifier's performance and communication costs, our scheme provides comparable results with Brakedown due to similar asymptotic complexity, while FRI-based constructions offer better performance due to its polylogarithmic complexity. We note that another advantage of our polynomial commitment scheme lies in its homomorphic property, which may benefit in the recent proof composition techniques from [BCL+21, KST22], while Brakedown and FRI don't offer homomorphic properties.

To compare with other lattice-based constructions, we reference the proof size from SLAP [AFLN23], the recent construction by Albrecht et al. We note that they also did not provide concrete performance for the prover and verifier. Although SLAP has asymptotically better proof size, which is polylogarithmic in

N, ours has a much smaller concrete proof size, which is about 4.1 times smaller for the same polynomial degree $N = 2^{20}$ and a similar base field size $\log p$.

	N	$\log p$	Proof size
Ours	2^{20}	255	8.93 MB
SLAP	2^{20}	276	36.5 MB

Table 4. Performance comparison with [AFLN23]

6 Conclusion

In this paper, we have introduced a novel polynomial commitment scheme based on lattice-based cryptography. Prior to this work, there have been few studies addressing the construction of concretely efficient lattice-based polynomial commitments, which primarily focused on addressing asymptotic performance. Our proof system not only successfully achieves practical proof sizes through an efficient encoding method for large prime fields, but also provides practical proof generation and verification performance. Compared to other post-quantum secure polynomial commitments, such as Brakedown [GLS+23], our proof system yields comparable proof sizes, proof generation, and verification performance.

There are still several ways to further improve our proof system. In terms of concrete performance, improving the discrete Gaussian sampling algorithm can lead to faster proof generation, as generating random samples currently consumes a significant portion of the total elapsed time. For reducing the proof size, we can enhance its asymptotic scale by adopting the leveled Ajtai commitment [BLNS20], which achieves $\tilde{O}(N^{1/c})$ complexity by generalizing the POK protocols.

References

ACL+22. Martin R Albrecht, Valerio Cini, Russell WF Lai, Giulio Malavolta, and Sri AravindaKrishnan Thyagarajan. Lattice-based snarks: Publicly verifiable, preprocessing, and recursively composable. In *Annual International Cryptology Conference*, pages 102–132. Springer, 2022.

AFLN23. Martin R Albrecht, Giacomo Fenzi, Oleksandra Lapiha, and Ngoc Khanh Nguyen. Slap: Succinct lattice-based polynomial commitments from standard assumptions. *Cryptology ePrint Archive*, 2023.

Ajt96. Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108, 1996.

ALS20. Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments. In *Annual International Cryptology Conference*, pages 470–499. Springer, 2020.

Ban95. Wojciech Banaszczyk. Inequalities for convex bodies and polar reciprocal lattices in r n. *Discrete & Computational Geometry*, 13:217–231, 1995.

BBB+18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE symposium on security and privacy (SP)*, pages 315–334. IEEE, 2018.

BBC+18. Carsten Baum, Jonathan Bootle, Andrea Cerulli, Rafaël Del Pino, Jens Groth, and Vadim Lyubashevsky. Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits. In *Annual International Cryptology Conference*, pages 669–699. Springer, 2018.

BCC+16. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*, pages 327–357. Springer, 2016.

BCK+14. Fabrice Benhamouda, Jan Camenisch, Stephan Krenn, Vadim Lyubashevsky, and Gregory Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 551–572. Springer, 2014.

BCL+21. Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data without succinct arguments. In *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part I 41*, pages 681–710. Springer, 2021.

BDK+18. Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber: a cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367. IEEE, 2018.

BDL+18. Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. More efficient commitments from structured lattice assumptions. In *International Conference on Security and Cryptography for Networks*, pages 368–385. Springer, 2018.

BFS20. Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent snarks from dark compilers. In *Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part I 39*, pages 677–706. Springer, 2020.

BG14. Shi Bai and Steven D Galbraith. An improved compression technique for signatures based on learning with errors. In *Topics in Cryptology–CT-RSA 2014: The Cryptographer's Track at the RSA Conference 2014, San Francisco, CA, USA, February 25-28, 2014. Proceedings*, pages 28–47. Springer, 2014.

BG18. Jonathan Bootle and Jens Groth. Efficient batch zero-knowledge arguments for low degree polynomials. In *IACR International Workshop on Public Key Cryptography*, pages 561–588. Springer, 2018.

BLNS20. Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. A non-pcp approach to succinct quantum-safe zero-knowledge. In *Annual International Cryptology Conference*, pages 441–469. Springer, 2020.

BS23. Ward Beullens and Gregor Seiler. Labrador: Compact proofs for r1cs from module-sis. In *Annual International Cryptology Conference*, pages 518–548. Springer, 2023.

BSBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In *45th international colloquium on automata, languages, and programming (icalp 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

CIL21. Hao Chen, Ilia Iliashenko, and Kim Laine. When heaan meets fv: a new somewhat homomorphic encryption with reduced memory overhead. In *IMA International Conference on Cryptography and Coding*, pages 265–285. Springer, 2021.

CLPX18. Hao Chen, Kim Laine, Rachel Player, and Yuhou Xia. High-precision arithmetic in homomorphic encryption. In *Cryptographers' Track at the RSA Conference*, pages 116–136. Springer, 2018.

dCP23. Leo de Castro and Chris Peikert. Functional commitments for all functions, with transparent setup and from sis. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 287–320. Springer, 2023.

DKL+18. Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pages 238–268, 2018.

ENS20. Muhammed F Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 259–288. Springer, 2020.

FLV23. Ben Fisch, Zeyu Liu, and Psi Vesely. Orbweaver: succinct linear functional commitments from lattices. In *Annual International Cryptology Conference*, pages 106–131. Springer, 2023.

GLS+23. Alexander Golovnev, Jonathan Lee, Srinath Setty, Justin Thaler, and Riad S Wahby. Brakedown: Linear-time and field-agnostic snarks for r1cs. In *Annual International Cryptology Conference*, pages 193–226. Springer, 2023.

GN08. Nicolas Gama and Phong Q Nguyen. Predicting lattice reduction. In *Advances in Cryptology–EUROCRYPT 2008: 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings 27*, pages 31–51. Springer, 2008.

Kar16. Charles FF Karney. Sampling exactly from the normal distribution. *ACM Transactions on Mathematical Software (TOMS)*, 42(1):1–14, 2016.

KLSS23. Duhyeong Kim, Dongwon Lee, Jinyeong Seo, and Yongsoo Song. Toward practical lattice-based proof of knowledge from hint-mlwe. In *Annual International Cryptology Conference*, pages 549–580. Springer, 2023.

KSSL21. Veronika Kuchta, Amin Sakzad, Ron Steinfeld, and Joseph K Liu. Efficient lattice-based polynomial evaluation and batch zk arguments. In *Selected Areas in Cryptography: 27th International Conference, Halifax, NS, Canada (Virtual Event), October 21-23, 2020, Revised Selected Papers 27*, pages 3–33. Springer, 2021.

KST22. Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. In *Annual International Cryptology Conference*, pages 359–388. Springer, 2022.

KZG10. Aniket Kate, Gregory M Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *Advances in Cryptology-ASIACRYPT 2010: 16th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 5-9, 2010. Proceedings 16*, pages 177–194. Springer, 2010.

Lee21. Jonathan Lee. Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In *Theory of Cryptography Conference*, 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 *Annual International Cryptology Conference*, pages 71–101. Springer, 2022.

LNS20. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Practical lattice-based zero-knowledge proofs for integer relations. In *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 1051–1070, 2020.

Lyu12. Vadim Lyubashevsky. Lattice signatures without trapdoors. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 738–755. Springer, 2012.

MKMS22. Jose Maria Bermudo Mera, Angshuman Karmakar, Tilen Marc, and Azam Soleimanian. Efficient lattice-based inner-product functional encryption. In *IACR International Conference on Public-Key Cryptography*, pages 163–193. Springer, 2022.

MR07. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal on Computing*, 37(1):267–302, 2007.

NS22. Ngoc Khanh Nguyen and Gregor Seiler. Practical sublinear proofs for r1cs from lattices. In *Annual International Cryptology Conference*, pages 133–162. Springer, 2022.

Pei10. Chris Peikert. An efficient and parallel gaussian sampler for lattices. In *Annual Cryptology Conference*, pages 80–97. Springer, 2010.

TCZ+20. Alin Tomescu, Robert Chen, Yiming Zheng, Ittai Abraham, Benny Pinkas, Guy Golan Gueta, and Srinivas Devadas. Towards scalable threshold cryptosystems. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 877–893. IEEE, 2020.

Wal19. Michael Walter. Sampling the integers with low relative error. In *Progress in Cryptology–AFRICACRYPT 2019: 11th International Conference on Cryptology in Africa, Rabat, Morocco, July 9–11, 2019, Proceedings 11*, pages 157–180. Springer, 2019.

WW23a. Hoeteck Wee and David J Wu. Lattice-based functional commitments: Fast verification and cryptanalysis. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 201–235. Springer, 2023.

WW23b. Hoeteck Wee and David J Wu. Succinct vector, polynomial, and functional commitments from lattices. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 385–416. Springer, 2023.

ZXH+22. Jiaheng Zhang, Tiancheng Xie, Thang Hoang, Elaine Shi, and Yupeng Zhang. Polynomial commitment with a {One-to-Many} prover and applications. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 2965–2982, 2022.

A Proof of Thm. 2

Proof. We use Lem. 5 to attain upper bounds for $\left\|\vec{\boldsymbol{t}}_j\right\|_2$ and $\|\vec{\boldsymbol{\tau}}_j\|_2$. Let $P_\ell = P \otimes I_\ell \in \mathbb{R}^{d\ell \times d\ell}$. Since $\boldsymbol{g}_j \sim \mathcal{D}_{\mathtt{Ecd}(\vec{g}_j)+P_\ell\mathbb{Z}^{d\ell},\sqrt{k+1}\mathfrak{s}_2 P_\ell} \equiv P_\ell \cdot \mathcal{D}_{P_\ell^{-1}\mathtt{Ecd}(\vec{g}_j)+\mathbb{Z}^{d\ell},\sqrt{k+1}\mathfrak{s}_2}$, the following holds with overwhelming probability, assuming $\sqrt{k+1}\cdot\mathfrak{s}_2 \geq \eta_\varepsilon(\mathbb{Z}^{d\ell})$:

$$\left\|\boldsymbol{g}_j\right\|_2 \leq \mathfrak{s}_2 \cdot \|P_\ell\|_2 \cdot \sqrt{(k+1)d\ell} \leq (b+1)\mathfrak{s}_2 \cdot \sqrt{(k+1)d\ell}$$

Similarly, we have $\|\boldsymbol{m}_i\|_2 \leq (b+1)\mathfrak{s}_1 \cdot \sqrt{d\ell}$, assuming that $\mathfrak{s}_1 \geq \eta_\varepsilon(\mathbb{Z}^{d\ell})$. Thus, for $\vec{\boldsymbol{t}}_j = \vec{\boldsymbol{g}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i} \cdot \vec{\boldsymbol{m}}_i$, the following holds with overwhelming probability, given that $\boldsymbol{c}_{j,i}$'s are monomials:

$$\left\|\vec{\boldsymbol{t}}_j\right\|_2 \leq \left\|\vec{\boldsymbol{g}}_j\right\|_2 + \sum_{i=0}^{k-1} \|\vec{\boldsymbol{m}}_i\|_2 \leq (b+1)(k\mathfrak{s}_1 + \sqrt{k+1}\cdot\mathfrak{s}_2)\sqrt{d\ell}$$

Similarly, we obtain the following result for $\vec{\boldsymbol{\tau}}_j = \vec{\boldsymbol{\gamma}}_j + \sum_{i=0}^{k-1} \boldsymbol{c}_{j,i}\vec{\boldsymbol{\mu}}_i$, assuming that $\sigma_1, \sqrt{k+1}\cdot\sigma_2 \geq \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$:

$$\|\vec{\boldsymbol{\tau}}_j\|_2 \leq \left\|\vec{\boldsymbol{\gamma}}_j\right\|_2 + \sum_{i=0}^{k-1} \|\vec{\boldsymbol{\mu}}_i\|_2 \leq (b+1)(k\sigma_1 + \sqrt{k+1}\cdot\sigma_2)\sqrt{d(\mu+\nu)}$$

As a result, we have $\left\|\vec{\boldsymbol{t}}_j \| \vec{\boldsymbol{\tau}}_j\right\|_2 \leq \beta_{\mathtt{Open}}$ with overwhelming probability. □

B Proof of Lem. 11

Proof. We note that the sample spaces of the above two distributions are identical to Λ^k since C_i's are integer matrices and $\Lambda \in \mathbb{Z}^n$. Thus, it is enough to show that

$$\Pr[\vec{z} = \vec{w}] \in [1-2\varepsilon, 1+2\varepsilon] \cdot \Pr[\vec{z'} = \vec{w}] \tag{1}$$

for all $\vec{w} = \vec{w}_0 || \cdots || \vec{w}_{k-1} \in \Lambda^k$ where $\vec{z} = \vec{z}_0 || \cdots || \vec{z}_{k-1}$, and $\vec{z'} = \vec{z'_0} || \cdots || \vec{z'}_{k-1}$. The first term of Eq. (1) can be computed as follows:

$$\begin{aligned}\Pr[\vec{z} = \vec{w}] &= \sum_{\vec{v} \in \vec{u}+\Lambda} \Pr[\vec{x} = \vec{v}] \cdot \prod_{i=0}^{k-1} \Pr[\vec{y}_i = \vec{w}_i - C_i \cdot \vec{v}] \\ &= \sum_{\vec{v} \in \vec{u}+\Lambda} \rho_{\sqrt{\mathfrak{S}}}(\vec{v}) \cdot \prod_{i=0}^{k-1} \rho_{C_i^{-1}\sqrt{\Sigma_i}}(\vec{v} - C_i^{-1} \cdot \vec{w}_i) \end{aligned} \tag{2}$$

Let $\mathfrak{S}_i^{-1} = \mathfrak{S}^{-1} + \sum_{j=0}^{i-1} C_j^\top \Sigma_j^{-1} C_j$ for $0 \leq i < k$. Then, we have the following by Lem. 6

$$\rho_{\sqrt{\mathfrak{S}_i}}(\vec{v} - \vec{a}_i) \cdot \rho_{C_i^{-1}\sqrt{\Sigma_i}}(\vec{v} - C_i^{-1}\vec{w}_i) = \rho_{\sqrt{\mathfrak{S}_{i+1}}}(\vec{v} - \vec{a}_{i+1}) \cdot \rho_{\sqrt{\mathfrak{S}_i + C_i^{-1}\Sigma_i C_i^{-\top}}}(\vec{b}_{i+1})$$

where $\vec{a}_0 = 0, \vec{b}_0 = C_0^{-1} \cdot \vec{w}_0$ and, $\vec{a}_{i+1} = \left(\mathfrak{S}_i + C_i^{-1}\Sigma_i C_i^{-\top}\right)\left(\mathfrak{S}_i^{-1}\vec{a}_i + C_i^{\top}\Sigma_i^{-1}\vec{w}_i\right)$, and $\vec{b}_{i+1} = \vec{a}_i - \vec{b}_i$. Note that both $\vec{a}_i$ and $\vec{b}_i$ are independent of the value of $\vec{v}$. Then, Eq. (2) can be rewritten as follows:

$$\Pr[\vec{z} = \vec{w}] = \prod_{i=0}^{k-1} \rho_{\sqrt{\mathfrak{S}_i + C_i^{-1}\Sigma_i C_i^{-\top}}}(\vec{b}_{i+1}) \cdot \left(\sum_{\vec{v} \in \vec{u} + \Lambda} \rho_{\sqrt{\mathfrak{S}_k}}(\vec{v} - \vec{a}_k) \right)$$

Similarly, the second term of Eq. (1) can be computed as follows:

$$\Pr[\vec{z'} = \vec{w}] = \prod_{i=0}^{k-1} \rho_{\sqrt{\mathfrak{S}_i + C_i^{-1}\Sigma_i C_i^{-\top}}}(\vec{b}_{i+1}) \cdot \left(\sum_{\vec{v} \in \Lambda} \rho_{\sqrt{\mathfrak{S}_k}}(\vec{v} - \vec{a}_k) \right)$$

By the given condition, we know that $\sqrt{\mathfrak{S}_k} \geq \eta_\varepsilon(\Lambda)$. Thus, we obtain the following by Lem. 3.

$$\sum_{\vec{v} \in \vec{u} + \Lambda} \rho_{\sqrt{\mathfrak{S}_k}}(\vec{v} - \vec{a}_k) \in [1 - 2\varepsilon, 1 + 2\varepsilon] \cdot \sum_{\vec{v} \in \Lambda} \rho_{\sqrt{\mathfrak{S}_k}}(\vec{v} - \vec{a}_k)$$

Therefore, we prove that $\Pr[\vec{z} = \vec{w}] \in [1 - 2\varepsilon, 1 + 2\varepsilon] \cdot \Pr[\vec{z'} = \vec{w}]$ □

C Proof of Thm. 6

$\mathcal{S}_{\texttt{PC.Eval}}(\mathsf{ck}, x, y)$

1 : $\vec{\mathfrak{h}}_0, \ldots, \vec{\mathfrak{h}}_m \leftarrow \mathcal{U}(R_q^\mu)$

2 : $\vec{\boldsymbol{h}}_0, \ldots, \vec{\boldsymbol{h}}_m \leftarrow \texttt{R.Ecd}(\vec{0}, \mathfrak{s}_1),\ \vec{\boldsymbol{\eta}}_0, \ldots, \vec{\boldsymbol{\eta}}_m \leftarrow \mathcal{D}_{\mathbb{Z}^d, \sigma_1}$

3 : $u_1, \ldots, u_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p),\ u_0 = y - \sum_{j=1}^{n} u_j x^j,\ \vec{u} = (u_0, \ldots, u_{n-1})$

4 : $\vec{\boldsymbol{h}}_{m+1} \leftarrow \texttt{R.Ecd}(\vec{u}, \sqrt{m+2} \cdot \mathfrak{s}_3),\ \vec{\boldsymbol{\eta}}_{m+1} \leftarrow \mathcal{D}_{\mathbb{Z}^d, \sqrt{m+2} \cdot \sigma_3}$

5 : $\vec{\boldsymbol{e}} = \sum_{i=0}^{m-1} \texttt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{h}}_i + \texttt{Ecd}(x) \cdot \vec{\boldsymbol{h}}_m + \vec{\boldsymbol{h}}_{m+1}$

6 : $\vec{\boldsymbol{\varepsilon}} = \sum_{i=0}^{m-1} \texttt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{\eta}}_i + \texttt{Ecd}(x) \cdot \vec{\boldsymbol{\eta}}_m + \vec{\boldsymbol{\eta}}_{m+1}$

7 : $\vec{\mathfrak{h}}_{m+1} = \mathbf{A}_0 \vec{\boldsymbol{e}} + \mathbf{A}_1 \vec{\boldsymbol{\varepsilon}} - \sum_{i=0}^{m-1} \texttt{Ecd}(x^{ni}) \cdot \vec{\mathfrak{h}}_i - \texttt{Ecd}(x) \cdot \vec{\mathfrak{h}}_m \pmod q$

8 : Output $(\vec{\mathfrak{h}}, \vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}})$

Fig. 4. Simulator for `PC.Eval`

Proof. Let $\mathsf{ck} \leftarrow \mathtt{PC.Setup}(1^\lambda)$, and $(h(X), x)$ be a polynomial and a point chosen by a PPT adversary $\mathcal{A}$. We prove that the algorithm $\mathcal{S}_{\mathtt{PC.Eval}}$ in Fig. 4 is an efficient simulator for the distribution of evaluation proofs generated by $\mathtt{PC.Eval}$ using hybrid arguments. We first define a distribution $\mathcal{H}_0(\mathsf{ck}, h(X), x)$ as follows.

$$\left\{(\vec{\mathfrak{h}}, \vec{e}, \vec{\varepsilon}) \;\middle|\; \begin{array}{r} b_1, \ldots, b_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p),\ \vec{h}_m = (b_1, \ldots, b_{n-1}, 0),\ \vec{h}_{m+1} = (0, -b_1, \ldots, -b_{n-1}) \\ \vec{\boldsymbol{h}}_i \leftarrow \mathtt{R.Ecd}(\vec{h}_i, \mathfrak{s}_1),\ \vec{\boldsymbol{\eta}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_1},\ \vec{\mathfrak{h}}_i = \mathbf{A}_0 \vec{\boldsymbol{h}}_i + \mathbf{A}_1 \vec{\boldsymbol{\eta}}_i \text{ for } 0 \le i \le m \\ \vec{\boldsymbol{h}}_{m+1} \leftarrow \mathtt{R.Ecd}(\vec{h}_{m+1}, \sqrt{m+2} \cdot \mathfrak{s}_3),\ \vec{\boldsymbol{\eta}}_{m+1} \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sqrt{m+2}\cdot\sigma_3} \\ \vec{e} = \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{h}}_i + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{h}}_m + \vec{\boldsymbol{h}}_{m+1} \\ \vec{\varepsilon} = \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{\eta}}_i + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{\eta}}_m + \vec{\boldsymbol{\eta}}_{m+1} \\ \vec{\mathfrak{h}}_{m+1} = \mathbf{A}_0 \vec{e} + \mathbf{A}_1 \vec{\varepsilon} - \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\mathfrak{h}}_i - \mathtt{Ecd}(x) \cdot \vec{\mathfrak{h}}_m \end{array} \right\}$$

We note that the above distribution is identical to the distribution of evaluation proofs generated from $\mathtt{PC.Eval}$

Claim 1: $\mathcal{H}_0(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_1(\mathsf{ck}, h(X), x)$ are statistically indistinguishable, where $\mathcal{H}_1(\mathsf{ck}, h(X), x)$ is defined as follows.

$$\left\{(\vec{\mathfrak{h}}, \vec{e}, \vec{\varepsilon}) \;\middle|\; \begin{array}{r} b_1, \ldots, b_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p),\ \vec{h}_m = (b_1, \ldots, b_{n-1}, 0),\ \vec{h}_{m+1} = (0, -b_1, \ldots, -b_{n-1}) \\ \vec{\boldsymbol{h}}_i \leftarrow \mathtt{R.Ecd}(\vec{h}_i, \mathfrak{s}_1),\ \vec{\boldsymbol{\eta}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_1},\ \vec{\mathfrak{h}}_i = \mathbf{A}_0 \vec{\boldsymbol{h}}_i + \mathbf{A}_1 \vec{\boldsymbol{\eta}}_i \text{ for } 0 \le i \le m \\ \underline{\vec{\boldsymbol{h}}_{m+1,i} \leftarrow \mathtt{R.Ecd}(-x^{ni} \cdot \vec{h}_i, \mathfrak{s}_3) \text{ for } 0 \le i < m,\ \vec{\boldsymbol{h}}_{m+1,m} \leftarrow \mathtt{R.Ecd}(-x \cdot \vec{h}_m, \mathfrak{s}_3)} \\ \underline{\vec{\boldsymbol{h}}_{m+1,m+1} \leftarrow \mathtt{R.Ecd}(\sum_{i=0}^{m-1} x^{ni} \cdot \vec{h}_i + x \cdot \vec{h}_m + \vec{h}_{m+1}, \mathfrak{s}_3)} \\ \underline{\vec{\boldsymbol{\eta}}_{m+1,0}, \ldots, \vec{\boldsymbol{\eta}}_{m+1,m+1} \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_3}} \\ \vec{e} = \sum_{i=0}^{m-1} (\vec{\boldsymbol{h}}_{m+1,i} + \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{h}}_i) + (\vec{\boldsymbol{h}}_{m+1,m} + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{h}}_m) + \vec{\boldsymbol{h}}_{m+1,m+1} \\ \vec{\varepsilon} = \sum_{i=0}^{m-1} (\vec{\boldsymbol{\eta}}_{m+1,i} + \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{\eta}}_i) + (\vec{\boldsymbol{\eta}}_{m+1,m} + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{\eta}}_m) + \vec{\boldsymbol{\eta}}_{m+1,m+1} \\ \vec{\mathfrak{h}}_{m+1} = \mathbf{A}_0 \vec{e} + \mathbf{A}_1 \vec{\varepsilon} - \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\mathfrak{h}}_i - \mathtt{Ecd}(x) \cdot \vec{\mathfrak{h}}_m \end{array} \right\}$$

The difference between $\mathcal{H}_0(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_1(\mathsf{ck}, h(X), x)$ lies in how $\vec{\boldsymbol{h}}_{m+1}$ and $\vec{\boldsymbol{\eta}}_{m+1}$ are sampled. Let $P_\ell = P \otimes I_\ell \in \mathbb{R}^{d\ell \times d\ell}$. In $\mathcal{H}_0(\mathsf{ck}, h(X), x)$, $\vec{\boldsymbol{h}}_{m+1} \sim \mathcal{D}_{\mathtt{Ecd}(\vec{h}_{m+1}) + P_\ell \mathbb{Z}^{d\ell}, \sqrt{m+2}\cdot\mathfrak{s}_3 P_\ell}$ and $\vec{\boldsymbol{\eta}}_{m+1} \sim \mathcal{D}^{\mu+\nu}_{Z^d, \sqrt{m+2}\cdot\sigma_3}$. Meanwhile, in $\mathcal{H}_1(\mathsf{ck}, h(X), x)$,

$$\vec{\boldsymbol{h}}_{m+1} = \sum_{i=0}^{m+1} \vec{\boldsymbol{h}}_{m+1,i} \sim \begin{array}{l} \sum_{i=0}^{m-1} \mathcal{D}_{\mathtt{Ecd}(-x^{ni}\cdot\vec{h}_i) + P_\ell\mathbb{Z}^{d\ell}, \mathfrak{s}_3 P_\ell} + \mathcal{D}_{\mathtt{Ecd}(-x\cdot\vec{h}_m) + P_\ell\mathbb{Z}^{d\ell}, \mathfrak{s}_3 P_\ell} + \\ \mathcal{D}_{\sum_{i=0}^{m-1} x^{ni}\cdot\vec{h}_i + x\cdot\vec{h}_m + \vec{h}_{m+1} + P_\ell\mathbb{Z}^{d\ell}, \mathfrak{s}_3 P_\ell} \end{array}$$

$$\vec{\boldsymbol{\eta}}_{m+1} = \sum_{i=0}^{m+1} \vec{\boldsymbol{\eta}}_{m+1,i} \sim \sum_{i=0}^{m+1} \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_3}$$

By Lem. 10, the distribution of $\vec{\boldsymbol{h}}_{m+1}$ in $\mathcal{H}_1(\mathsf{ck}, h(X), x)$ is within a statistical distance of $8(m+1)\varepsilon$ from $\mathcal{D}_{\mathtt{Ecd}(\vec{h}_{m+1})+P_\ell\mathbb{Z}^{d\ell},\sqrt{m+2}\cdot\mathfrak{s}_3 P_\ell}$ if $\frac{\mathfrak{s}_3}{\sqrt{2}}P_\ell \geq \eta_\varepsilon(P_\ell\mathbb{Z}^{d\ell})$, which is implied by $\frac{\mathfrak{s}_3}{\sqrt{2}} \geq \frac{1}{b-1}\eta_\varepsilon(P_\ell\mathbb{Z}^{d\ell})$. In a similar manner, by Lem. 10, $\sum_{i=0}^{m+1}\mathcal{D}^{\mu+\nu}_{Z^d,\sigma_3}$ is within a statistical distance of $8(m+1)\varepsilon$ from $\mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sqrt{m+2}\cdot\sigma_3}$ if $\frac{\sigma_3}{\sqrt{2}} \geq \eta_\varepsilon(\mathbb{Z}^{d(\mu+\nu)})$. Therefore, $\mathcal{H}_0(\mathsf{ck}, \vec{m})$ and $\mathcal{H}_1(\mathsf{ck}, \vec{m})$ are within a statistical distance $16(m+1)\varepsilon$, which is $\mathsf{negl}(\lambda)$.

Claim 2: $\mathcal{H}_1(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ are computationally indistinguishable, where $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ is defined as follows.

$$\left\{ (\vec{\mathfrak{h}}, \vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}}) \;\middle|\; \begin{array}{r} b_1, \ldots, b_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p),\ \vec{h}_m = (b_1, \ldots, b_{n-1}, 0),\ \vec{h}_{m+1} = (0, -b_1, \ldots, -b_{n-1}) \\ \vec{\boldsymbol{h}}_i \leftarrow \mathtt{R.Ecd}(\vec{h}_i, \mathfrak{s}_1),\ \vec{\boldsymbol{\eta}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sigma_1},\ \underline{\vec{\mathfrak{h}}_i \leftarrow \mathcal{U}(R_q^\mu) \text{ for } 0 \leq i \leq m} \\ \text{The rest is same as } \mathcal{H}_1(\mathsf{ck}, h(X), x). \end{array} \right\}$$

The difference between $\mathcal{H}_1(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ lies in how $\vec{\mathfrak{h}}$ is sampled. Then, distinguishing $\mathcal{H}_1(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ is at least hard as $\mathsf{HintMLWE}^{\mathtt{Ecd}(x^{ni})}_{R,\nu,q,\sigma_1,\sigma_3}$ from some k since $(\vec{\mathfrak{h}}_i, \vec{\boldsymbol{\eta}}_{m+1,i} + \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{\eta}}_i)$ can be considered as a $\mathsf{HintMLWE}$ instance for $0 \leq i < m$. Similarly, $(\vec{\mathfrak{h}}_m, \vec{\boldsymbol{\eta}}_{m+1,m} + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{\eta}}_m)$ can be considered as an instance from $\mathsf{HintMLWE}^{\mathtt{Ecd}(x)}_{R,\nu,q,\sigma_1,\sigma_3}$. By Thm. 1, there is an efficient reduction from $\mathsf{MLWE}_{R,\nu,q,\sigma_{\mathtt{PC.Eval}}}$ to $\mathsf{HintMLWE}^{\mathtt{Ecd}(x^k)}_{R,\nu,q,\sigma_1,\sigma_3}$ for any k if $\sigma_{\mathtt{PC.Eval}} \geq \sqrt{2} \cdot \eta_\varepsilon(\mathbb{Z}^d)$, given that $\left\|\mathtt{Ecd}(x^k)\right\|_1 \leq \frac{(b+1)r}{2}$. Therefore, $\mathcal{H}_1(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ are computationally indistinguishable due to the hardness of $\mathsf{MLWE}_{R,\nu,q,\sigma_{\mathtt{PC.Eval}}}$.

Claim 3: $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_3(\mathsf{ck}, x, h(x))$ are statistically indistinguishable, where $\mathcal{H}_3(\mathsf{ck}, x, h(x))$ is defined as follows.

$$\left\{ (\vec{\mathfrak{h}}, \vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}}) \;\middle|\; \begin{array}{r} u_1, \ldots, u_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p),\ u_0 = h(x) - \sum_{j=1}^{n} b_j x^j,\ \vec{u} = (u_0, \ldots, u_{n-1}) \\ \hline \underline{\vec{\boldsymbol{h}}_i \leftarrow \mathtt{R.Ecd}(\vec{0}, \mathfrak{s}_1)},\ \vec{\boldsymbol{\eta}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d,\sigma_1},\ \vec{\mathfrak{h}}_i \leftarrow \mathcal{U}(R_q^\mu) \text{ for } 0 \leq i \leq m \\ \underline{\vec{\boldsymbol{h}}_{m+1,i} \leftarrow \mathtt{R.Ecd}(\vec{0}, \mathfrak{s}_3) \text{ for } 0 \leq i \leq m,\ \vec{\boldsymbol{h}}_{m+1,m+1} \leftarrow \mathtt{R.Ecd}(\vec{u}, \mathfrak{s}_3)} \\ \text{The rest is same as } \mathcal{H}_2(\mathsf{ck}, h(X), x). \end{array} \right\}$$

The difference between $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_3(\mathsf{ck}, x, h(x))$ lies in how $\vec{\boldsymbol{h}}_0, \ldots \vec{\boldsymbol{h}}_m$, and $\vec{\boldsymbol{h}}_{m+1,0}, \ldots, \vec{\boldsymbol{h}}_{m+1,m+1}$ are sampled. We note that $\vec{\boldsymbol{e}} = \sum_{i=0}^{m-1}(\vec{\boldsymbol{h}}_{m+1,i} + \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{h}}_i) + (\vec{\boldsymbol{h}}_{m+1,m} + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{h}}_m) + \vec{\boldsymbol{h}}_{m+1,m+1}$, so it suffices to consider only the distribution of $(\vec{\boldsymbol{h}}_{m+1,i} + \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{h}}_i)$ for $0 \leq i < m$, $(\vec{\boldsymbol{h}}_{m+1,m} + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{h}}_m)$, and $\vec{\boldsymbol{h}}_{m+1,m+1}$. By Lem. 11, the first two distributions in $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_3(\mathsf{ck}, x, h(x))$ are within a statistical distance 2ε, if the following holds

$$\sqrt{\left(\mathfrak{s}_1^{-2}(P_\ell P_\ell^\top)^{-1} + \mathfrak{s}_3^{-2} \cdot X_k^\top (P_\ell P_\ell^\top)^{-1} X_k\right)^{-1}} \geq \eta_\varepsilon(P_\ell\mathbb{Z}^{d\ell})$$

where X_k is the negacyclic matrix of $\mathtt{Ecd}(x^k)$. By Lem. 2, this condition is implied by $\mathfrak{s}_{\mathtt{PC.Eval}} \geq \frac{1}{b-1} \cdot \eta_\varepsilon(P_\ell\mathbb{Z}^{d\ell})$ since $\|X_k\|_2 \leq \frac{(b+1)r}{2}$ and $\left\|P^{-1}\right\|_2 \leq \frac{1}{b-1}$.

For $\vec{\boldsymbol{h}}_{m+1,m+1}$, it is a randomized encoding of $\sum_{i=0}^{m-1} x^{ni} \cdot \vec{h}_i + x \cdot \vec{h}_m + \vec{h}_{m+1}$ in $\mathcal{H}_2(\mathsf{ck}, h(X), x)$, whereas that of $\mathcal{H}_3(\mathsf{ck}, x, h(x))$ is a randomized encoding of $\vec{u}$. We note that these distributions are identical in the sense that a uniformly sampled vector, whose inner product with $(1, x, \ldots, x^{n-1})$, results in $h(x)$. Thus, we can regard $\vec{\boldsymbol{b}}_{0,m+1}$ in $\mathcal{H}_2$ and $\mathcal{H}_3$ as following an identical distribution, given the evaluation point x and the evaluation result $h(x)$. Therefore, $\mathcal{H}_2(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_3(\mathsf{ck}, x, h(x))$ are within a statistical distance $2(m+1)\varepsilon$, which is $\mathsf{negl}(\lambda)$.

Claim 4: $\mathcal{H}_3(\mathsf{ck}, x, h(x))$ and $\mathcal{H}_4(\mathsf{ck}, x, h(x))$ are statistically indistinguishable, where $\mathcal{H}_4(\mathsf{ck}, x, h(x))$ is defined as follows.

$$\left\{ (\vec{\mathfrak{h}}, \vec{\boldsymbol{e}}, \vec{\boldsymbol{\varepsilon}}) \;\middle|\; \begin{array}{r} u_1, \ldots, u_{n-1} \leftarrow \mathcal{U}(\mathbb{Z}_p),\ u_0 = h(x) - \sum_{j=1}^{n} b_j x^j,\ \vec{u} = (u_0, \ldots, u_{n-1}) \\ \vec{\boldsymbol{h}}_i \leftarrow \mathtt{R.Ecd}(\vec{0}, \mathfrak{s}_1),\ \vec{\boldsymbol{\eta}}_i \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sigma_1},\ \vec{\mathfrak{h}}_i \leftarrow \mathcal{U}(R_q^{\mu}) \text{ for } 0 \le i \le m \\ \underline{\vec{\boldsymbol{h}}_{m+1} \leftarrow \mathtt{R.Ecd}(\vec{u}, \sqrt{m+2} \cdot \mathfrak{s}_3)},\ \vec{\boldsymbol{\eta}}_{m+1} \leftarrow \mathcal{D}^{\mu+\nu}_{\mathbb{Z}^d, \sqrt{m+2} \cdot \sigma_3} \\ \vec{\boldsymbol{e}} = \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{h}}_i + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{h}}_m + \vec{\boldsymbol{h}}_{m+1} \\ \vec{\boldsymbol{\varepsilon}} = \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\boldsymbol{\eta}}_i + \mathtt{Ecd}(x) \cdot \vec{\boldsymbol{\eta}}_m + \vec{\boldsymbol{\eta}}_{m+1} \\ \vec{\mathfrak{h}}_{m+1} = \mathbf{A}_0 \vec{\boldsymbol{e}} + \mathbf{A}_1 \vec{\boldsymbol{\varepsilon}} - \sum_{i=0}^{m-1} \mathtt{Ecd}(x^{ni}) \cdot \vec{\mathfrak{h}}_i - \mathtt{Ecd}(x) \cdot \vec{\mathfrak{h}}_m \end{array} \right\}$$

The difference between $\mathcal{H}_3$ and $\mathcal{H}_4$ lies in how $\vec{\boldsymbol{h}}_{m+1}$ and $\vec{\boldsymbol{\eta}}_{m+1}$ are sampled, similar to Claim 1. Thus, following the proof of Claim 1, they are within a statistical distance of $16(m+1)\varepsilon$, which is $\mathsf{negl}(\lambda)$.

Therefore, we can conclude that $\mathcal{H}_0(\mathsf{ck}, h(X), x)$ and $\mathcal{H}_4(\mathsf{ck}, x, h(x))$ are computationally indistinguishable under the given conditions. Since $\mathcal{H}_0(\mathsf{ck}, h(X), x)$ corresponds to the distribution of evaluation transcripts from $\mathtt{PC.Eval}$, and $\mathcal{H}_4(\mathsf{ck}, x, h(x))$ corresponds to the distribution of simulated proofs from $\mathcal{S}_{\mathtt{PC.Eval}}$, there is no PPT adversary $\mathcal{A}$ that can distinguish between the two distributions with non-negligible probability. □