Practical Exact Proofs from Lattices: New Techniques to Exploit Fully-Splitting Rings*

Muhammed F. Esgin[1,2**], Ngoc Khanh Nguyen[3,4], and Gregor Seiler[3,4]

[1] Monash University, Australia
[2] Data61, CSIRO, Australia
[3] IBM Research – Zurich, Switzerland
[4] ETH Zurich, Switzerland

Abstract. We propose a lattice-based zero-knowledge proof system for exactly proving knowledge of a ternary solution $\vec{s} \in \{-1, 0, 1\}^n$ to a linear equation $A\vec{s} = \vec{u}$ over $\mathbb{Z}_q$, which produces proofs that are 7.5× shorter than the state-of-the-art result by Bootle, Lyubashevsky and Seiler (CRYPTO 2019). At the core lies a technique that utilizes the module-homomorphic BDLOP commitment scheme (SCN 2018) over the fully splitting cyclotomic ring $\mathbb{Z}_q[X]/(X^d + 1)$ to prove scalar products with the NTT vector of a secret polynomial.

1 Introduction

In a continuous effort towards migration to post-quantum cryptography, there has recently been many works with the aim towards realizing zero-knowledge proofs based on computational lattice problems, e.g. [dLNS17, dLS18, YAZ+19, BLS19, ESLL19, EZS+19]. That is because zero-knowledge proofs are a fundamental component in much more complex cryptographic protocols, such as verifiable encryption or circuit satisfiability. In almost all applications, it is crucial to be able to prove in zero-knowledge that one knows how to open a cryptographic commitment, and to prove that the committed values have particular properties or satisfy certain relations.

For lattice-based schemes, the relations of interest are linear equations of the form

$$A\vec{s} = \vec{u}, \tag{1}$$

where A is a publicly known matrix defined over some ring $\mathfrak{R}$ (often $\mathbb{Z}_q$ or a cyclotomic ring $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$) and $\vec{s}$ is a short vector with small coefficients over the ring. For example, in the case of an encryption scheme based on the (Ring)-LWE problem, $(A, \vec{u})$ is the public key and $\vec{s}$ is the secret key. In numerous applications such as group signatures [Cv91], ring signatures [RST01], and verifiable encryption, one must prove (in a zero-knowledge fashion) knowledge of a secret $\vec{s}$ that satisfies a relation of the form in (1), which is where lattice-based zero-knowledge proofs come into play. Throughout this manuscript, we assume that (1) is defined over $\mathbb{Z}_q$, which is the most general "unstructured" case. The ring-based cases are easily obtained by having a structured matrix A.

1.1 Lattice-Based Zero-Knowledge Proofs

We describe main strategies used in the literature for proving knowledge of a secret short vector $\vec{s}$ satisfying (1). The first one is to adapt Stern's protocol [Ste94] into proof systems as in [KTX08, LNSW13] to prove knowledge of an *exact* witness $\vec{s}$ satisfying (1). What we mean by *exact* is that there is no knowledge gap between a witness known by an honest prover and that the proof system convinces the verifier of. The main drawback of Stern-based "combinatorial" protocols is that the concrete efficiency of such schemes are far behind practical expectations (see Table 1). The reason behind it is that a single protocol execution provides

* This research was supported by the SNSF ERC starting transfer grant FELICITY.
** Work done while at IBM Research – Zurich

a very poor soundness level of 2/3, and thus many protocol repetitions (in the order of hundreds) are required to reach a negligible soundness error.

A more "algebraic" approach in the hope of proving knowledge of $\vec{s}$ satisfying (1) is adapting Schnorr's protocol [Sch90] to the lattice setting. Although this so-called Fiat-Shamir with Aborts technique [Lyu09, Lyu12] offers very practically efficient solutions, this only proves knowledge of a much longer $\vec{s}'$ satisfying $A\vec{s}' = c\vec{u}$ for some scalar $c \in \mathfrak{R}$. This is an example of a knowledge gap referred to before and thus these protocols are often called "relaxed" (or "approximate"). Observe here that there are two aspects of relaxation: 1) there is an extra c term in the proved relation, and 2) $\vec{s}'$ has larger coefficients than $\vec{s}$.

The efficiency of these protocols comes mainly from the fact that a single protocol execution is sufficient to reach a negligible soundness error. Even though relaxed proofs are sufficient and lead to efficient instantiations for some applications such as (ordinary) signatures [Lyu12, DKL$^+$18] and ring signatures [ESLL19, EZS$^+$19], for the settings of, for example, verifiable encryption and group signatures, the relaxation in the underlying zero-knowledge proof leads to further complications and relaxations in the higher level construction. We discuss more on this in Section 5. Another disadvantage of proving knowledge of larger secrets in the relaxed proofs is that the system modulus q is forced to be larger due to security reasons. If one is interested in using the zero-knowledge proof to prove a relation in a different protocol, then relaxed proofs may require the parameters of the latter protocol to be increased, which may not be desirable and/or a worth-while tradeoff.

An alternative approach to this problem is using hash-based argument systems such as STARKs [BBHR18] and Aurora [BCR$^+$19]. They stem from the PCP-based framework of Kilian [Kil92] and produce asymptotically logarithmic sized arguments. Furthermore, they can be instantiated based on collision resistance of hash functions. In particular, one could naively construct such a scheme with lattice-based hash functions. However, we believe that utilising the additional algebraic structure of structured computational assumptions should result in small sizes over the generic PCP-based approaches. Even though efficient lattice-based zero-knowledge protocols offer asymptotically linear proof size, making use of the underlying mathematical structure results in much smaller constants than generic constructions. Moreover, implementations of the generic proof systems for our linear equations are known to be very slow with running times in the order of tens of seconds. On the other hand, lattice-based constructions are usually very fast with running times in the order of 1ms.

A pair of very recent works [BLS19, YAZ$^+$19] explored a different way of addressing the problem using lattice assumptions. The approach in [BLS19, YAZ$^+$19] has two parts: 1) a relaxed proof of knowledge: proving knowledge of $\vec{s}'$ satisfying $A\vec{s}' = c\vec{u} \mod q$, and 2) a relaxed binary/ternary proof: proving that $\vec{s}' = c\vec{s}$ for $\vec{s} \in \{-1, 0, 1\}^n$. Combining these relations and assuming that c is invertible, then one ends up with proving (1) exactly for $\vec{s} \in \{-1, 0, 1\}^n$.

The drawback of these works is that the concrete practical efficiency is still not at a satisfactory level even though they are much more efficient than Stern-based proofs. Our contribution in this work is to address this problem. In particular, we introduce novel techniques, using the structure in fully-splitting rings, for constructing efficient lattice-based zero-knowledge proofs of exact relations.

1.2 Our Approach

Let us define $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$, where $X^d + 1$ splits fully into linear factors over $\mathbb{Z}_q$, i.e., $X^d + 1 = \prod_{i=0}^{d-1}(X - r_i) \mod q$. In this case, $\mathcal{R}_q$ is isomorphic to d "copies" of $\mathbb{Z}_q$ and we call an integer stored in a copy as *NTT coefficient* (see Section 2.2 for more details). Our goal is to introduce an efficient proof system that allows to prove that the NTT coefficients of a committed message $\boldsymbol{s} \in \mathcal{R}_q$ satisfy a relation of the form in (1). For the sake of simplicity, we assume here that the dimensions of $\vec{s}$ and $\mathcal{R}_q$ are the same. The general case is studied in Section 3.3.

We know that the i-th NTT coefficient of a polynomial $\boldsymbol{s} \in \mathcal{R}_q$ is equal to the evaluation of $\boldsymbol{s}$ at the i-th primitive $2d$-th root of unity r_i. Therefore, if $\vec{s} = \mathsf{NTT}(\boldsymbol{s})$ and $\vec{\gamma} \xleftarrow{\$} \mathbb{Z}_q^d$ is a random vector, we have

$$\begin{aligned}\langle A\vec{s} - \vec{u}, \vec{\gamma} \rangle &= \langle A\vec{s}, \vec{\gamma} \rangle - \langle \vec{u}, \vec{\gamma} \rangle = \langle \vec{s}, A^T\vec{\gamma} \rangle - \langle \vec{u}, \vec{\gamma} \rangle \\ &= \sum_{i=0}^{d-1} \boldsymbol{s}(r_i) \left(\mathsf{NTT}^{-1}(A^T\vec{\gamma})\right)(r_i) - \langle \vec{u}, \vec{\gamma} \rangle = \frac{1}{d}\sum_{i=0}^{d-1} \boldsymbol{f}(r_i) = f_0,\end{aligned}$$

where $\boldsymbol{f} := d\mathsf{NTT}^{-1}(A^T\vec{\gamma})\boldsymbol{s} - \langle \vec{u}, \vec{\gamma} \rangle \in \mathcal{R}_q$ and $f_0 \in \mathbb{Z}_q$ is the constant coefficient of $\boldsymbol{f}$. The last equality follows from Lemma 2.1. The idea is then to prove that f_0, the constant coefficient of $\boldsymbol{f}$, is zero. This proves that $\langle A\vec{s} - \vec{u}, \vec{\gamma} \rangle = 0$. For a uniformly random $\vec{\gamma} \in \mathbb{Z}_q^d$, the probability that $\langle A\vec{s} - \vec{u}, \vec{\gamma} \rangle = 0$ when $A\vec{s} \neq \vec{u} \pmod q$ is $1/q$. Therefore, allowing the verifier to choose a random $\vec{\gamma} \in \mathbb{Z}_q^d$ as a challenge, proving $f_0 = 0$ proves that $A\vec{s} = \vec{u}$ with a soundness error of $1/q$.

To prove that $\boldsymbol{f}$ has vanishing constant coefficient, the prover initially commits to $\boldsymbol{s}$ and a polynomial $\boldsymbol{g}$ with vanishing constant coefficient. The polynomial $\boldsymbol{g}$ will be used to mask $\boldsymbol{f}$. Upon receiving a challenge $\vec{\gamma} \in \mathbb{Z}_q^d$, the prover computes $\boldsymbol{f}$ and sets $\boldsymbol{h} = \boldsymbol{f} + \boldsymbol{g}$. Using the given information, we show that the verifier can compute a commitment to $\boldsymbol{f}$ (without requiring it to be sent by the prover). This allows to prove that h is of the correct form and the verifier can simply observe that $\boldsymbol{h}$ has a zero constant coefficient.

The above proof system has a soundness error of $1/q$, which is not negligibly small for typical choices of q. We show in Section 3.2 how to amplify the soundness of this protocol at a low cost using Galois automorphisms. Informally, consider k uniformly random vectors $\vec{\gamma}_0, \ldots, \vec{\gamma}_{k-1}$ such that $1/q^k$ is negligible. Similarly as before, we can write

$$\boldsymbol{f}_i := d\mathsf{NTT}^{-1}(A^T\vec{\gamma})\boldsymbol{s} - \langle \vec{u}, \vec{\gamma}_i \rangle$$

and thus the constant coefficient of $\boldsymbol{f}_i$ is $\langle A\vec{s} - \vec{u}, \vec{\gamma}_i \rangle$. For each $i = 0, \ldots, k-1$, we will define a map $\mathsf{L}_i : \mathcal{R}_q \to \mathcal{R}_q$ which satisfies the following property. Denote $\boldsymbol{p} := \mathsf{L}_i(\boldsymbol{f}_i)$ and $(p_0, \ldots, p_{d-1})$ to be the coefficient vector of $\boldsymbol{p}$. Then, $p_0 = \ldots = p_{i-1} = p_{i+1} = \ldots = p_{k-1} = 0$ and $p_i = \langle A\vec{s} - \vec{u}, \vec{\gamma}_i \rangle$. We can observe that if $A\vec{s} = \vec{u}$ then $\boldsymbol{f}$ defined now as

$$\boldsymbol{f} = \mathsf{L}_0(\boldsymbol{f}_0) + \ldots + \mathsf{L}_{k-1}(\boldsymbol{f}_{k-1})$$

has the first k coefficients equal to 0. Therefore, we can construct a protocol for proving this similarly as above. On the other hand, when $A\vec{s} \neq \vec{u}$ then the probability that all the first k coefficients of $\boldsymbol{f}$ are equal to zero is $1/q^k$.

The advantage of this approach over the standard way of having k-parallel repetitions is that the size of the commitment part of the non-interactive proof remains the same as that of a single protocol run. Therefore, the overall cost is significantly reduced.

We believe that this protocol can be useful in other settings, where one wants to "switch" from the original relation's domain to another one where proofs can be done more efficiently.

Notice that the above proof system does not fully answer our main question because we only proved that $A\vec{s} = \vec{u}$, but not that $\vec{s} \in \{-1, 0, 1\}^n$.

Proving that the secret is short. Now that we have a way to prove linear relations among the NTT coefficients of a polynomial, we can exploit the technique from [BLS19] to prove that the coefficients are small. Observe that for $\boldsymbol{s} \in \mathcal{R}_q$, if

$$\boldsymbol{s}(\boldsymbol{s} - 1)(\boldsymbol{s} + 1) = 0 \quad \text{in } \mathcal{R}_q, \tag{2}$$

then $s_i \in \{-1, 0, 1\}$ for any NTT coefficient s_i of $\boldsymbol{s}$. Therefore, by proving (2), we can perform d ternary proofs in parallel for d NTT coefficients, and then link these NTT coefficients to the coordinates of $\vec{s}$ in our main relation (1) using the aforementioned technique.

Another obstacle against practical efficiency (as encountered in [BLS19, YAZ+19]) is that a proof of such a non-linear relation as in (2) requires communication of "garbage terms". These garbage terms end up being a substantial cost in the proofs in [BLS19, YAZ+19]. In the companion paper [ALS20] to this work, a better product proof is presented that reduces the cost of the garbage terms, also using Galois automorphisms.

Table 1. Proof length comparison for proving knowledge of LWE secrets in dimension 1024. The result of Stern-like proofs is taken from [BLS19].

Stern-like proofs	3522 KB
[BLS19]	384 KB
Our work	**51 KB**

Applications. Having an efficient proof system to prove knowledge of $\vec{s} \in \{-1, 0, 1\}^n$ satisfying (1) paves the way for various efficient zero-knowledge proofs that can be used in many applications. To show the effectiveness of our new techniques, we present two example applications with concrete parameters. The first one is to prove knowledge of secrets in LWE samples. This is an important proof system to be used, for example, with fully homomorphic encryption (FHE) schemes. The goal here is to prove that $\vec{u}$ is a proper LWE sample such that $\vec{u} = A'\vec{s}' + \vec{e} \bmod q$ for $\vec{s}', \vec{e} \in \{-1, 0, 1\}^k$, which is equivalent to proving $\vec{u} = (A' \parallel I_k) \cdot \vec{s} \bmod q$ for $\vec{s} = (\vec{s}', \vec{e}) \in \{-1, 0, 1\}^{2k}$. As shown in Table 1, our proof system achieves an improvement of 7.5× in terms of proof length over the state-of-the-art result by Bootle et al. [BLS19], and is dramatically shorter than the Stern-based proofs.

Our other example application is a proof of plaintext knowledge. In this case, the goal is to create a ciphertext and a zero-knowledge proof to prove that the ciphertext is a proper encryption of a message known by the prover. Proofs of plaintext knowledge have applications, for example, in the settings of verifiable encryption, verifiable secret sharing and group signatures.

Being a very core proof system, there are many other applications beyond the two examples above, where our main protocol and our new techniques can be useful. For example, one can apply our unstructured linear proof to prove that one vector is a NTT representation of a polynomial (written as a vector of coefficients). Indeed, the matrix A in (1) simply becomes a Vandermonde matrix. Furthermore, one can see [YAZ+19] for various applications that all build on a similar core proof system.

2 Preliminaries

2.1 Notation

The following table summarizes the notation and parameters that will appear in this paper.

Let q be an odd prime, and $\mathbb{Z}_q$ denote the ring of integers modulo q. For $r \in \mathbb{Z}$, we define $r \bmod q$ to be the unique element in the interval $[-\frac{q-1}{2}, \frac{q-1}{2}]$ that is congruent to r modulo q. We write $\vec{v} \in \mathbb{Z}_q^m$ to denote vectors over $\mathbb{Z}_q$ and matrices over $\mathbb{Z}_q$ will be written as regular capital letters M. By default, all vectors are column vectors. We write $\vec{v} \parallel \vec{w}$ for the concatenation of $\vec{v}$ and $\vec{w}$ (which is still a column vector).

Let d be a power of two and denote $\mathcal{R}$ and $\mathcal{R}_q$ to be the rings $\mathbb{Z}[X]/(X^d + 1)$ and $\mathbb{Z}_q[X]/(X^d + 1)$, respectively. Bold lower-case letters $\boldsymbol{p}$ denote elements in $\mathcal{R}$ or $\mathcal{R}_q$ and bold lower-case letters with arrows $\vec{\boldsymbol{b}}$ represent column vectors with coefficients in $\mathcal{R}$ or $\mathcal{R}_q$. We also use bold upper-case letters for matrices $\boldsymbol{B}$ over $\mathcal{R}$ or $\mathcal{R}_q$. For a polynomial denoted as a bold letter, we write its i-th coefficient as the corresponding regular font letter with subscript i, e.g. $f_0 \in \mathbb{Z}_q$ is the constant coefficient of $\boldsymbol{f} \in \mathcal{R}_q$.

We write $x \xleftarrow{\$} S$ when $x \in S$ is sampled uniformly at random from the set S and similarly $x \xleftarrow{\$} D$ when x is sampled according to the distribution D.

Norms and Sizes. For an element $w \in \mathbb{Z}_q$, we write $|w|$ to mean $|w \bmod q|$. Define the ℓ_∞ and ℓ_2 norms for $\boldsymbol{w} \in \mathcal{R}_q$ as follows,

$$\|\boldsymbol{w}\|_\infty = \max_i |w_i| \quad \text{and} \quad \|\boldsymbol{w}\|_2 = \sqrt{|w_0|^2 + \ldots + |w_{d-1}|^2}.$$

Similarly, for $\vec{\boldsymbol{w}} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k) \in \mathcal{R}^k$, we define

$$\|\vec{\boldsymbol{w}}\|_\infty = \max_i \|\boldsymbol{w}_i\|_\infty \quad \text{and} \quad \|\vec{\boldsymbol{w}}\|_2 = \sqrt{\|\boldsymbol{w}_1\|_2^2 + \ldots + \|\boldsymbol{w}_k\|_2^2}.$$

Parameter	Explanation
d	Degree of the polynomial $X^d + 1$, power of two
q	Rational prime modulus
$\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$	The field over which the linear system is defined
$m \in \mathbb{Z}$	The number of rows in the linear system
$n \in \mathbb{Z}$	The number of columns in the linear system
$\mathcal{R} = \mathbb{Z}[X]/(X^d + 1)$	The ring of integers in the $2d$-th cyclotomic number field
$\mathcal{R}_q = \mathbb{Z}_q[X]/(X^d + 1)$	The ring of integers $\mathcal{R}$ modulo q
$k \in \mathbb{Z}$	Repetition rate
$\sigma = \sigma_{2d/k+1}$	Automorphism in $\mathsf{Aut}(\mathcal{R}_q)$ of order k
$\mathcal{C} \subset \mathcal{R}$	Challenge set
C	Probability distribution over $\mathcal{C}$ for challenges
T	Bound for honest prover's $\boldsymbol{c}\vec{\boldsymbol{r}}$ in the Euclidean norm
$\mathfrak{s} = 2T$	Standard deviation for sampling $\vec{\boldsymbol{y}}'$
$M = \exp(6 + 1/16)$	Rate for rejection sampling
λ	M-LWE dimension
κ	M-SIS dimension
$\beta = \mathfrak{s}\sqrt{2(\lambda + \kappa + 2)d}$	Bound for honest prover's $\vec{\boldsymbol{z}}_i$ in the Euclidean norm
χ	Error distribution on $\mathcal{R}$ in the M-LWE problem
$D_{\mathfrak{s}}$	Discrete Gaussian distribution on $\mathcal{R}$ with st. dev. $\mathfrak{s}$

Table 2. Overview of Parameters and Notation

2.2 Prime Splitting and Galois Automorphisms

Let l be a power of two dividing d and suppose $q - 1 \equiv 2l \pmod{4l}$. Then, $\mathbb{Z}_q$ contains primitive $2l$-th roots of unity but no elements with order a higher power of two, and the polynomial $X^d + 1$ factors into l irreducible binomials $X^{d/l} - \zeta$ modulo q where ζ runs over the $2l$-th roots of unity in $\mathbb{Z}_q$ [LS18, Theorem 2.3].

The ring $\mathcal{R}_q$ has a group of automorphisms $\mathsf{Aut}(\mathcal{R}_q)$ that is isomorphic to $\mathbb{Z}_{2d}^\times$,

$$i \mapsto \sigma_i \colon \mathbb{Z}_{2d}^\times \to \mathsf{Aut}(\mathcal{R}_q),$$

where σ_i is defined by $\sigma_i(X) = X^i$. In fact, these automorphisms come from the Galois automorphisms of the $2d$-th cyclotomic number field which factor through $\mathcal{R}_q$.

The group $\mathsf{Aut}(\mathcal{R}_q)$ acts transitively on the prime ideals $(X^{d/l} - \zeta)$ in $\mathcal{R}_q$ and every σ_i factors through field isomorphisms

$$\mathcal{R}_q/(X^{d/l} - \zeta) \to \mathcal{R}_q/(\sigma^i(X^{d/l} - \zeta)).$$

Concretely, for $i \in \mathbb{Z}_{2d}^\times$ it holds that

$$\sigma_i(X^{d/l} - \zeta) = (X^{id/l} - \zeta) = (X^{d/l} - \zeta^{i^{-1}})$$

To see this, observe that the roots of $X^{d/l} - \zeta^{i^{-1}}$ (in an appropriate extension field of $\mathbb{Z}_q$) are also roots of $X^{id/l} - \zeta$. Then, for $f \in \mathcal{R}_q$,

$$\sigma_i\left(f \bmod (X^{d/l} - \zeta)\right) = \sigma_i(f) \bmod (X^{d/l} - \zeta^{i^{-1}}).$$

The cyclic subgroup $\langle 2l + 1 \rangle < \mathbb{Z}_{2d}^\times$ has order d/l [LS18, Lemma 2.4] and stabilizes every prime ideal $(X^{d/l} - \zeta)$ since ζ has order $2l$. The quotient group $\mathbb{Z}_{2d}^\times / \langle 2l + 1 \rangle$ has order l and hence acts simply transitively on the l prime ideals. Therefore, we can index the prime ideals by $i \in \mathbb{Z}_{2d}^\times / \langle 2l + 1 \rangle$ and write

$$(X^d + 1) = \prod_{i \in \mathbb{Z}_{2d}^\times / \langle 2l+1 \rangle} \left(X^{d/l} - \zeta^i\right)$$

Now, the product of the $k \mid l$ prime ideals $(X^{d/l} - \zeta^i)$ where i runs over $\langle 2l/k+1\rangle/\langle 2l+1\rangle$ is given by the ideal $(X^{kd/l} - \zeta^k)$. So, we can partition the l prime ideals into l/k groups of k ideals each, and write

$$\left(X^d+1\right) = \prod_{j \in \mathbb{Z}_{2d}^\times/\langle 2l/k+1\rangle} \left(X^{kd/l} - \zeta^{jk}\right) = \prod_{j \in \mathbb{Z}_{2d}^\times/\langle 2l/k+1\rangle} \prod_{i \in \langle 2l/k+1\rangle/\langle 2l+1\rangle} \left(X^{\frac{d}{l}} - \zeta^{ij}\right).$$

Another way to write this, which we will use in our protocols, is to note that $\mathbb{Z}_{2d}^\times/\langle 2l/k+1\rangle \cong \mathbb{Z}_{2l/k}^\times$ and the powers $(2l/k+1)^i$ for $i = 0, \ldots, k-1$ form a complete set of representatives for $\langle 2l/k+1\rangle/\langle 2l+1\rangle$. So, if $\sigma = \sigma_{2l/k+1} \in \mathsf{Aut}(\mathcal{R}_q)$, then

$$\left(X^d+1\right) = \prod_{j \in \mathbb{Z}_{2l/k}^\times} \prod_{i=0}^{k-1} \sigma^i \left(X^{\frac{d}{l}} - \zeta^j\right),$$

and the prime ideals are indexed by $(i,j) \in I = \{0, \ldots, k-1\} \times \mathbb{Z}_{2l/k}^\times$.

The fully splitting case. In this paper our main attention lies on the setup where $q \equiv 1 \pmod{2d}$ and hence q splits completely. In this case there is a primitive $2d$-th root of unity $\zeta \in \mathbb{Z}_q$ and

$$(X^d+1) = \prod_{i \in \mathbb{Z}_{2d}^\times} (X - \zeta^i).$$

Then, for a divisor k of d and $\sigma = \sigma_{2d/k+1}$ of order k, we have the partitioning

$$(X^d+1) = \prod_{j \in \mathbb{Z}_{2d}^\times/\langle 2d/k+1\rangle} \prod_{i \in \langle 2d/k+1\rangle} (X - \zeta^{ij}) = \prod_{j \in \mathbb{Z}_{2d/k}^\times} \prod_{i=0}^{k-1} \sigma^i(X - \zeta^j)$$

2.3 The Number Theoretic Transform

The Number Theoretic Transform (NTT) of a polynomial $\boldsymbol{f} \in \mathcal{R}_q$ is defined by

$$\mathsf{NTT}(\boldsymbol{f}) = (\hat{\boldsymbol{f}}_i)_{i \in \mathbb{Z}_{2l}^\times} \in \prod_{i \in \mathbb{Z}_{2l}^\times} \mathbb{Z}_q[X]/(X^{d/l} - \zeta^i) \cong (\mathbb{F}_{q^{d/l}})^l$$

where $\hat{\boldsymbol{f}}_i = \boldsymbol{f} \bmod (X^{d/l} - \zeta^i)$. We write $\mathsf{NTT}^{-1}(\hat{\boldsymbol{f}}) = \boldsymbol{f}$ for the inverse map, which exists due to the Chinese remainder theorem. Note that for $\boldsymbol{f}, \boldsymbol{g} \in \mathcal{R}_q$, $\mathsf{NTT}(\boldsymbol{fg}) = \mathsf{NTT}(\boldsymbol{f}) \circ \mathsf{NTT}(\boldsymbol{g})$ where $\circ$ denotes the coefficient-wise multiplication of vectors.

The sum of the NTT coefficients of a polynomial $\boldsymbol{f} \in \mathcal{R}_q$ is equal to first d/l coefficients. This will be later used when proving unstructured linear relations over $\mathbb{Z}_q$.

Lemma 2.1. *Let $\boldsymbol{f} \in \mathcal{R}_q$. Then $\frac{1}{l}\sum_{i \in \mathbb{Z}_{2l}^\times} \hat{\boldsymbol{f}}_i = f_0 + f_1 X + \cdots + f_{d/l-1} X^{d/l-1}$, when we lift the $\hat{\boldsymbol{f}}_i$ to $\mathbb{Z}_q[X]$.*

Proof. Write $\boldsymbol{f}(X) = \boldsymbol{f}_0(X^{d/l}) + \boldsymbol{f}_1(X^{d/l})X + \cdots + \boldsymbol{f}_{d/l-1}(X^{d/l})X^{d/l-1}$ Then, it suffices to prove

$$\frac{1}{l} \sum_{i \in \mathbb{Z}_{2l}^\times} \boldsymbol{f}_j(\zeta^i) = f_j$$

for all $j = 0, \ldots, d/l-1$, which is the sum over the coefficients of a fully splitting length-l NTT. We find

$$\sum_{i \in \mathbb{Z}_{2l}^\times} \boldsymbol{f}_j(\zeta^i) = \sum_{i \in \mathbb{Z}_{2l}^\times} \sum_{\nu=0}^{l-1} f_{\nu d/l+j} \zeta^{i\nu} = \sum_{\nu=0}^{l-1} f_{\nu d/l+j} \sum_{i \in \mathbb{Z}_{2l}^\times} \zeta^{i\nu}$$

and it remains to show that for every $\nu \in \{1, \ldots, l-1\}$, $\sum_{i \in \mathbb{Z}_{2l}^\times} \zeta^{i\nu} = 0$. Indeed,

$$\sum_{i \in \mathbb{Z}_{2l}^\times} \zeta^{i\nu} = \sum_{i=0}^{l-1} \zeta^{(2i+1)\nu} = \zeta^\nu \sum_{i=0}^{l-1} \zeta^{2i\nu} = \zeta^\nu \frac{\zeta^{2l\nu} - 1}{\zeta^{2\nu} - 1} = 0$$

since $\zeta^{2l\nu} = 1$. □

2.4 Challenge Space

Let $\mathcal{C} = \{-1, 0, 1\}^d \subset \mathcal{R}_q$ be the set of ternary polynomials, which have coefficients in $\{-1, 0, 1\}$. We define $C \colon \mathcal{C} \to [0, 1]$ to be the probability distribution on $\mathcal{C}$ such that the coefficients of a challenge $\boldsymbol{c} \xleftarrow{\$} C$ are independently identically distributed with $\Pr(0) = 1/2$ and $\Pr(1) = \Pr(-1) = 1/4$.

In [ALS20] it is shown that if $\boldsymbol{c} \xleftarrow{\$} C$ then the distribution of $\boldsymbol{c} \bmod X^{kd/l} - \zeta^k$ are almost uniform in $\mathbb{Z}_q$.

Lemma 2.2. *Let $\boldsymbol{c} \xleftarrow{\$} C$. The coefficients of $\boldsymbol{c} \bmod X^{kd/l} - \zeta^k$ are identically independently distributed, say with distribution X. Then, for $x \in \mathbb{Z}_q$,*

$$\Pr(X = x) \leq \frac{1}{q} + \frac{2l/k}{q} \sum_{j \in \mathbb{Z}_q^* / \langle \zeta^k \rangle} \prod_{i=0}^{l/k-1} \left| \frac{1}{2} + \frac{1}{2} \cos(2\pi j \zeta^{ki} / q) \right|. \tag{3}$$

In particular, if $d = 128$, $q \approx 2^{32}$ is fully splitting, i.e. $l = d$, and $k = 4$, then the maximum probability for the coefficients of $\boldsymbol{c} \bmod X^4 - \zeta^4$ is bounded by $2^{-31.4}$.

2.5 Module-SIS and Module-LWE Problems

We employ the computationally binding and computationally hiding commitment scheme from [BDL+18] in our protocols, and rely on the well-known Module-LWE (MLWE) and Module-SIS (MSIS) [PR06, LM06, LPR10, LS15] problems to prove the security of our constructions. Both problems are defined over a ring $\mathcal{R}_q$ for a positive modulus $q \in \mathbb{Z}^+$.

Definition 2.3 ($\mathsf{MSIS}_{n,m,\beta_{\mathrm{SIS}}}$)**.** *The goal in the* Module-SIS *problem with parameters $n, m > 0$ and $0 < \beta_{\mathrm{SIS}} < q$ is to find, for a given matrix $\boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{n \times m}$, $\vec{\boldsymbol{x}} \in \mathcal{R}_q^m$ such that $\boldsymbol{A}\vec{\boldsymbol{x}} = \vec{\boldsymbol{0}}$ over $\mathcal{R}_q$ and $0 < \|\vec{\boldsymbol{x}}\|_2 \leq \beta_{\mathrm{SIS}}$. We say that a PPT adversary $\mathcal{A}$ has advantage ϵ in solving $\mathsf{MSIS}_{n,m,\beta_{\mathrm{SIS}}}$ if*

$$\Pr\left[0 < \|\vec{\boldsymbol{x}}\|_2 \leq \beta_{\mathrm{SIS}} \wedge \boldsymbol{A}\vec{\boldsymbol{x}} = \vec{\boldsymbol{0}} \text{ over } \mathcal{R}_q \,\middle|\, \boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{n \times m};\ \vec{\boldsymbol{x}} \leftarrow \mathcal{A}(\boldsymbol{A})\right] \geq \epsilon.$$

Definition 2.4 ($\mathsf{MLWE}_{n,m,\chi}$)**.** *In the* Module-LWE *problem with parameters $n, m > 0$ and an error distribution χ over $\mathcal{R}$, the PPT adversary $\mathcal{A}$ is asked to distinguish $(\boldsymbol{A}, \vec{\boldsymbol{t}}) \xleftarrow{\$} \mathcal{R}_q^{m \times n} \times \mathcal{R}_q^m$ from $(\boldsymbol{A}, \boldsymbol{A}\vec{\boldsymbol{s}} + \vec{\boldsymbol{e}})$ for $\boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{m \times n}$, a secret vector $\vec{\boldsymbol{s}} \xleftarrow{\$} \chi^n$ and error vector $\vec{\boldsymbol{e}} \xleftarrow{\$} \chi^m$. We say that $\mathcal{A}$ has advantage ϵ in solving $\mathsf{MLWE}_{n,m,\chi}$ if*

$$\begin{aligned} \Big| \Pr\left[b = 1 \,\middle|\, \boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{m \times n};\ \vec{\boldsymbol{s}} \xleftarrow{\$} \chi^n;\ \vec{\boldsymbol{e}} \xleftarrow{\$} \chi^m;\ b \leftarrow \mathcal{A}(\boldsymbol{A}, \boldsymbol{A}\vec{\boldsymbol{s}} + \vec{\boldsymbol{e}})\right] \\ - \Pr\left[b = 1 \,\middle|\, \boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{m \times n};\ \vec{\boldsymbol{t}} \xleftarrow{\$} \mathcal{R}_q^m;\ b \leftarrow \mathcal{A}(\boldsymbol{A}, \vec{\boldsymbol{t}})\right] \Big| \geq \epsilon. \end{aligned} \tag{4}$$

For our practical security estimations of these two problems against known attacks, the parameter m in both of the problems does not play a crucial role. Therefore, we sometimes simply omit m and use the notations $\mathsf{MSIS}_{n,B}$ and $\mathsf{MLWE}_{n,\chi}$. The parameters κ and λ denote the *module ranks* for MSIS and MLWE, respectively.

2.6 Error Distribution, Discrete Gaussians and Rejection Sampling

For sampling randomness in the commitment scheme that we use, and to define the particular variant of the Module-LWE problem that we use, we need to specify the error distribution χ^d on $\mathcal{R}$. In general any of the standard choices in the literature is fine. So, for example, χ can be a narrow discrete Gaussian distribution or the uniform distribution on a small interval. In the numerical examples in Section 4.2 we assume that χ is the computationally simple centered binomial distribution on $\{-1, 0, 1\}$ where ± 1 both have probability 5/16 and 0 has probability 6/16. This distribution is chosen (rather than the more "natural" uniform one) because it is easy to sample given a random bitstring by computing $a_1 + a_2 - b_1 - b_2 \bmod 3$ with uniformly random bits a_i, b_i.

Rejection Sampling. In our zero-knowledge proof, the prover will want to output a vector $\vec{\boldsymbol{z}}$ whose distribution should be independent of a secret randomness vector $\vec{\boldsymbol{r}}$, so that $\vec{\boldsymbol{z}}$ cannot be used to gain any information on the prover's secret. During the protocol, the prover computes $\vec{\boldsymbol{z}} = \vec{\boldsymbol{y}} + \boldsymbol{c}\vec{\boldsymbol{r}}$ where $\vec{\boldsymbol{r}}$ is the randomness used to commit to the prover's secret, $\boldsymbol{c} \xleftarrow{\$} C$ is a challenge polynomial, and $\vec{\boldsymbol{y}}$ is a "masking" vector. To remove the dependency of $\vec{\boldsymbol{z}}$ on $\vec{\boldsymbol{r}}$, we use the rejection sampling technique by Lyubashevsky [Lyu09, Lyu12]. In the two variants of this technique the masking vector is either sampled uniformly from some bounded region or using a discrete Gaussian distribution. In the high dimensions we will encounter, the Gaussian variant is far superior as it gives acceptable rejection probabilities for much narrower distributions. We first define the discrete Gaussian distribution and then state the rejection sampling algorithm in Figure 1, which plays a central role in Lemma 2.6.

Definition 2.5. *The* discrete Gaussian distribution *on $\mathcal{R}^\ell$ centered around $\vec{\boldsymbol{v}} \in \mathcal{R}^\ell$ with standard deviation $\mathfrak{s} > 0$ is given by*

$$D^{\ell d}_{\boldsymbol{v},\mathfrak{s}}(\vec{\boldsymbol{z}}) = \frac{e^{-\|\vec{\boldsymbol{z}}-\vec{\boldsymbol{v}}\|_2^2/2\mathfrak{s}^2}}{\sum_{\vec{\boldsymbol{z}}'\in\mathcal{R}^\ell} e^{-\|\vec{\boldsymbol{z}}'\|_2^2/2\mathfrak{s}^2}}.$$

When it is centered around $\vec{\mathbf{0}} \in \mathcal{R}^\ell$ we write $D^{\ell d}_{\mathfrak{s}} = D^{\ell d}_{\vec{\mathbf{0}},\mathfrak{s}}$

Lemma 2.6 (Rejection Sampling). *Let $V \subseteq \mathcal{R}^\ell$ be a set of polynomials with norm at most T and $\rho\colon V \to [0,1]$ be a probability distribution. Also, write $\mathfrak{s} = 2T$ and $M = \exp(6 + 1/16)$. Now, sample $\vec{\boldsymbol{v}} \xleftarrow{\$} \rho$ and $\vec{\boldsymbol{y}} \xleftarrow{\$} D^{\ell d}_{\mathfrak{s}}$, set $\vec{\boldsymbol{z}} = \vec{\boldsymbol{y}} + \vec{\boldsymbol{v}}$, and run $b \leftarrow \mathsf{Rej}(\vec{\boldsymbol{z}}, \vec{\boldsymbol{v}}, \mathfrak{s})$ Then, the probability that $b = 0$ is at least $(1-2^{-100})/M$ and the distribution of $(\vec{\boldsymbol{v}}, \vec{\boldsymbol{z}})$, conditioned on $b = 0$, is within statistical distance of $2^{-100}/M$ of the product distribution $\rho \times D^{\ell d}_{\mathfrak{s}}$.*

```
Rej(z, v, s)
01 u <-$ [0, 1)
02 If u > (1/M) · exp((−2⟨z, v⟩ + ‖v‖²) / (2s²))
03    return 1
04 Else
05    return 0
```

Fig. 1. Rejection Sampling [Lyu12].

We will also use the following tail bound, which follows from [Ban93, Lemma 1.5(i)].

Lemma 2.7. *Let $\vec{\boldsymbol{z}} \xleftarrow{\$} D^{\ell d}_{\mathfrak{s}}$. Then*

$$\Pr\left[\|\vec{\boldsymbol{z}}\|_2 \le \mathfrak{s}\sqrt{2\ell d}\right] \ge 1 - 2^{-\log(e/2)\ell d/4}.$$

2.7 Commitment Scheme

In our protocol, we use a variant of the commitment scheme from [BDL+18], which allows to commit to a vector of messages in $\mathcal{R}_q$. Our basic proof of knowledge of multiplicative relations will prove that $\boldsymbol{m}_1\boldsymbol{m}_2 = \boldsymbol{m}_3$, so for simplicity, we just describe the commitment scheme for three messages. Suppose that we want to commit to a message vector $\vec{\boldsymbol{m}} = (\boldsymbol{m}_1, \ldots, \boldsymbol{m}_l)^T \in \mathcal{R}_q^l$ and that module ranks of κ and λ are required for MSIS and MLWE security, respectively. Then, in the key generation, a uniformly random matrix $\boldsymbol{B}_0 \xleftarrow{\$} \mathcal{R}_q^{\kappa\times(\lambda+\kappa+l)}$ and vectors $\vec{\boldsymbol{b}}_1, \ldots, \vec{\boldsymbol{b}}_l \xleftarrow{\$} \mathcal{R}_q^{\lambda+\kappa+l}$ are generated and output as public parameters.

To commit to the message $\vec{\boldsymbol{m}}$, we first sample $\vec{\boldsymbol{r}} \xleftarrow{\$} \chi^{(\lambda+\kappa+l)d}$. Now, there are two parts of the commitment scheme; the binding part and the message encoding part. Particularly, we compute

$$\begin{aligned}\vec{\boldsymbol{t}}_0 &= \boldsymbol{B}_0\vec{\boldsymbol{r}},\\ \boldsymbol{t}_i &= \langle\vec{\boldsymbol{b}}_i, \vec{\boldsymbol{r}}\rangle + \boldsymbol{m}_i \quad \text{for } i = 1, \ldots, l,\end{aligned}$$

where $\vec{\boldsymbol{t}}_0$ forms the binding part and each $\boldsymbol{t}_i$ encodes a message polynomial $\boldsymbol{m}_i$. The commitment scheme is computationally hiding under the Module-LWE assumption and computationally binding under the Module-SIS assumption; see [BDL+18].

The utility of the commitment scheme for zero-knowledge proof systems stems from the fact that one can compute module homomorphisms on committed messages. For example, let $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ be from $\mathcal{R}_q$. Then

$$\boldsymbol{a}_1\boldsymbol{t}_1 + \boldsymbol{a}_2\boldsymbol{t}_2 = \langle\boldsymbol{a}_1\vec{\boldsymbol{b}}_1 + \boldsymbol{a}_2\vec{\boldsymbol{b}}_2, \vec{\boldsymbol{r}}\rangle + \boldsymbol{a}_1\boldsymbol{m}_1 + \boldsymbol{a}_2\boldsymbol{m}_2$$

is a commitment to the message $\boldsymbol{a}_1\boldsymbol{m}_1+\boldsymbol{a}_2\boldsymbol{m}_2$ with matrix $\boldsymbol{a}_1\vec{\boldsymbol{b}}_1+\boldsymbol{a}_2\vec{\boldsymbol{b}}_2$. This module homomorphic property together with a proof that a commitment is a commitment to the zero polynomial allows to prove linear relations among committed messages over $\mathcal{R}_q$.

2.8 Opening and Product Proof

We use the opening proof from [ALS20, Figure 2] that we sketch now. Suppose that the prover knows an opening to the commitment

$$\begin{aligned}\vec{\boldsymbol{t}}_0 &= \boldsymbol{B}_0\vec{\boldsymbol{r}},\\ \boldsymbol{t}_1 &= \langle\vec{\boldsymbol{b}}_1, \vec{\boldsymbol{r}}\rangle + \boldsymbol{m}_1.\end{aligned}$$

As in previous opening proofs the prover gives an approximate proof for the first equation. To this end, the prover and verifier engage in k parallel executions of a sigma protocol with challenges $\sigma^i(\boldsymbol{c})$, $i = 0, \ldots, k-1$, that are the rotations of a global challenge $\boldsymbol{c} \xleftarrow{\$} C$. Concretely, in the first flow the prover samples k short masking vectors $\vec{\boldsymbol{y}}_i$ from the discrete Gaussian distribution $D_{\mathfrak{s}}^{(\lambda+\kappa+1)d}$ and sends them over to the verifier. The verifier replies with the challenge $\boldsymbol{c}$. Then the prover applies rejection sampling, and, if this does not reject, sends $\vec{\boldsymbol{z}}_i = \vec{\boldsymbol{y}}_i+\sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}$. The verifier checks that the $\vec{\boldsymbol{z}}_i$ are short and the equations $\boldsymbol{B}_0\vec{\boldsymbol{z}}_i = \vec{\boldsymbol{w}}_i+\sigma^i(\boldsymbol{c})\vec{\boldsymbol{t}}_0$.

Now, unlike in previous protocols, the algebraic setup is such that it is not possible to extract a pair of accepting transcript with invertible challenge difference $\bar{\boldsymbol{c}} = \boldsymbol{c} - \boldsymbol{c}'$. Instead, extraction works by piecing together l/k accepting transcripts where for each ideal $(X^{kd/l} - \zeta^{kj})$ there is a transcript pair with challenge difference $\bar{\boldsymbol{c}}_j \bmod (X^{kd/l} - \zeta^{kj}) \neq 0$. For this to work out it is required that the maximum probability p over $\mathbb{Z}_q$ of the coefficients of $\boldsymbol{c} \bmod (X^{kd/l} - \zeta^k)$, as given by Lemma 2.2, is such that $p^{kd/l}$ is negligible. For example, if $d = 128$, $q \approx 2^{-32}$ fully splits so that $l = d$, and $k = 4$, then $p^{kd/l} = p^4 \approx 2^{-128}$.

Next, the analysis of the protocol given in [ALS20, Theorem 4.4] shows that it is possible to extract a weak opening from a prover with non-negligible high success probability, as given in the following definition.

Definition 2.8. *A* weak opening *for the commitment* $\vec{\boldsymbol{t}} = \vec{\boldsymbol{t}}_0 \parallel \boldsymbol{t}_1$ *consists of* l *polynomials* $\sigma^i(\bar{\boldsymbol{c}}_j) \in \mathcal{R}_q$*, a randomness vector* $\vec{\boldsymbol{r}}^*$ *over* $\mathcal{R}_q$ *and a message* $\boldsymbol{m}_1^* \in \mathcal{R}_q$ *such that*

$$\begin{aligned}
&\left\|\sigma^i(\bar{\boldsymbol{c}}_j)\right\|_1 \leq 2d \text{ and } \sigma^i(\bar{\boldsymbol{c}}_j) \bmod \sigma^i(X^{d/l} - \zeta^j) \neq 0 \text{ for all } (i,j) \in I,\\
&\left\|\sigma^i(\bar{\boldsymbol{c}}_j)\vec{\boldsymbol{r}}^*\right\|_2 \leq 2\beta \text{ for all } (i,j) \in I,\\
&\boldsymbol{B}_0\vec{\boldsymbol{r}}^* = \vec{\boldsymbol{t}}_0,\\
&\langle \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{r}}^* \rangle + \boldsymbol{m}_1^* = \boldsymbol{t}_1.
\end{aligned}$$

The commitment scheme is binding with respect to weak openings, c.f. [ALS20, Lemma 4.3]. Furthermore, in the extraction it is also possible to obtain vectors $\vec{\boldsymbol{y}}_i^*$ such that every accepting transcript is such that

$$\vec{\boldsymbol{z}}_i = \vec{\boldsymbol{y}}^* + \sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}^*,$$

when it contains the same prover commitments $\vec{\boldsymbol{w}}_i$ that were used in the extraction.

We also apply the product proof from [ALS20, Figure 4], adapted to the case of a cubic relation, to prove that our secret vector has ternary coefficients. In addition to the opening proof, the product proof only requires two additional commitments to garbage terms, and the masking vectors $\vec{\boldsymbol{y}}_i$ can be used as the randomness vectors in these commitments. So, the prover sends two polynomials

$$\begin{aligned}
\boldsymbol{t}_1' &= \langle \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{y}}_0 \rangle + \boldsymbol{g}_1,\\
\boldsymbol{t}_2' &= \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{y}}_0 \rangle + \boldsymbol{g}_2.
\end{aligned}$$

3 Proving Unstructured Linear Relations over $\mathbb{Z}_q^n$

Our goal for this section is to construct an efficient protocol for proving unstructured linear relations among committed $\mathbb{Z}_q$-elements. By this we meant that we want to be able to commit to a vector $\vec{s} \in \mathbb{Z}_q^n$ and prove that it fulfills an arbitrary linear equation $A\vec{s} = \vec{u}$ with public matrix $A \in \mathbb{Z}_q^{m \times n}$ and right hand side $\vec{u} \in \mathbb{Z}_q^m$. We borrow LWE terminology and call the linear equation unstructured to highlight the fact that A can be an arbitrary matrix over $\mathbb{Z}_q$ that does not necessarily express linear relations over some ring of higher rank.

Proofs of linear relations are only useful for applications in lattice cryptography if it is possible to amend them by a proof of shortness. So, we will also want to be able to prove that the vector $\vec{s}$ is short. As opposed to the so-called approximate proofs that are ubiquitous in lattice cryptography and where the prover only proves knowledge of a vector that is much longer than the one he actually knows, we are interested in exact proofs of shortness. These have the advantage that the parameters of underlying cryptographic schemes do not have to account for the longer vectors that can be extracted from a prover, i.e. the schemes do not need to be secure with respect to the longer vectors. This results in more efficient schemes. For example, one interesting goal of this line of research is to construct a proof of plaintext knowledge or a verifiable encryption scheme for a standard unmodified lattice-based public-key encryption scheme. In particular, for one of the schemes submitted to the NIST PQC standardization effort.

The most efficient lattice-based exact proofs of shortness work by encoding the vector $\vec{s}$ in the NTT representations $\mathsf{NTT}(\check{\boldsymbol{s}}_i)$ of possibly several polynomials $\check{\boldsymbol{s}}_i \in \mathcal{R}_q$. In a first step we restrict to the case where q splits completely in $\mathcal{R}$. Then $\mathsf{NTT}(\check{\boldsymbol{s}}_i)$ is a vector in $\mathbb{Z}_q^d$.

Now, for simplicity assume that n is divisible by d. Suppose the prover $\mathcal{P}$ knows an opening to a commitment $\vec{\boldsymbol{t}} = \vec{\boldsymbol{t}}_0 \parallel \boldsymbol{t}_1 \parallel \cdots \parallel \boldsymbol{t}_{n/d}$ to n/d secret polynomials $\check{\boldsymbol{s}}_1, \ldots, \check{\boldsymbol{s}}_{n/d} \in \mathcal{R}_q$. More precisely,

$$\begin{aligned}
\vec{\boldsymbol{t}}_0 &= \boldsymbol{B}_0\vec{\boldsymbol{r}},\\
\boldsymbol{t}_i &= \langle \vec{\boldsymbol{b}}_i, \vec{\boldsymbol{r}} \rangle + \check{\boldsymbol{s}}_i \text{ for } i \in \{1, \ldots, n/d\}.
\end{aligned}$$

Then, the goal of $\mathcal{P}$ is to prove that the vector

$$\vec{s} = \mathsf{NTT}(\check{\boldsymbol{s}}_1) \parallel \cdots \parallel \mathsf{NTT}(\check{\boldsymbol{s}}_{n/d}) \in \mathbb{Z}_q^n$$

satisfies the linear equation $A\vec{s} = \vec{u}$ over $\mathbb{Z}_q$ where $A \in \mathbb{Z}_q^{m \times n}$ and $\vec{u} \in \mathbb{Z}_q^m$ are public.

Firstly, we describe the main ideas and present a protocol which achieves soundness error $1/q$. Then, we present two methods to efficiently decrease the soundness error to negligible quantities.

3.1 Basic Protocol

Let us assume that $n = d$ and denote $\check{\boldsymbol{s}} := \check{\boldsymbol{s}}_1$. We show how to deal with the case $n > d$ in Section 3.3. The first protocol relies on the following simple observation. Suppose that $A\vec{s} = \vec{u}$. This means that for all $\vec{\gamma} \in \mathbb{Z}_q^m$ we have $\langle A\vec{s} - \vec{u}, \vec{\gamma} \rangle = 0$. On the contrary, if $A\vec{s} \neq \vec{u}$, then for a uniformly random $\vec{\gamma} \in \mathbb{Z}_q^m$, $\langle A\vec{s} - \vec{u}, \vec{s} \rangle = 0$ only with probability $1/q$. Hence, $\vec{\gamma}$ will become a challenge generated from the verifier. Using Lemma 2.1, we rewrite the inner product,

$$\begin{aligned}\langle A\vec{s} - \vec{u}, \vec{\gamma} \rangle &= \langle A\vec{s}, \vec{\gamma} \rangle - \langle \vec{u}, \vec{\gamma} \rangle = \langle \vec{s}, A^T\vec{\gamma} \rangle - \langle \vec{u}, \vec{\gamma} \rangle \\ &= \sum_{j \in \mathbb{Z}_{2d}^\times} \boldsymbol{s}(\zeta^j) \left(\mathsf{NTT}^{-1}(A^T\vec{\gamma})\right)(\zeta^j) - \langle \vec{u}, \vec{\gamma} \rangle \\ &= \frac{1}{d} \sum_{j \in \mathbb{Z}_{2d}^\times} \boldsymbol{f}(\zeta^j) = f_0\end{aligned}$$

where $\boldsymbol{f} \in \mathcal{R}_q$ is the polynomial defined by $\boldsymbol{f} := \mathsf{NTT}^{-1}(dA^T\vec{\gamma})\check{\boldsymbol{s}} - \langle \vec{u}, \vec{\gamma} \rangle$ and $f_0 \in \mathbb{Z}_q$ is the constant coefficient of $\boldsymbol{f}$. So by utilizing the polynomial product in $\mathcal{R}_q$ it is possible to compute a scalar product over $\mathbb{Z}_q$ with a vector encoded in the NTT representation of the polynomial. We observe that the verifier can compute a commitment to $\boldsymbol{f}$. Indeed, note that

$$\mathsf{NTT}^{-1}(dA^T\vec{\gamma})\boldsymbol{t}_1 - \langle \vec{u}, \vec{\gamma} \rangle = \langle \mathsf{NTT}^{-1}(dA^T\vec{\gamma})\vec{\boldsymbol{b}_1}, \vec{\boldsymbol{r}} \rangle + \boldsymbol{f}.$$

Hence, $\mathcal{V}$ computes

$$\boldsymbol{\tau} = \mathsf{NTT}^{-1}(dA^T\vec{\gamma})\boldsymbol{t}_1 - \langle \vec{u}, \vec{\gamma} \rangle. \tag{5}$$

Now, $\mathcal{P}$ needs to prove that $\boldsymbol{f}$ has a zero constant coefficient. The idea is to first send a commitment $\boldsymbol{t}_2$ to a random polynomial $\boldsymbol{g}$ with a zero constant coefficient before $\vec{\gamma}$ is generated. Intuitively, $\boldsymbol{g}$ is introduced to mask $\boldsymbol{f}$. After getting $\vec{\gamma}$, $\mathcal{P}$ sends $\boldsymbol{h} = \boldsymbol{f} + \boldsymbol{g}$ and the verifier can check that $h_0 = 0$. Note that by knowing $\boldsymbol{\tau}, \boldsymbol{t}_2$ and $\boldsymbol{h}$, the verifier can compute a commitment $\boldsymbol{\tau} + \boldsymbol{t}_2 - \boldsymbol{h}$ to the zero polynomial $\boldsymbol{0}$. Hence, in the final stage $\mathcal{P}$ needs to prove that this polynomial is indeed a commitment to $\boldsymbol{0}$ in the usual way.

We present the full protocol in Figure 4. Firstly, prover $\mathcal{P}$ generates a random polynomial $\boldsymbol{g} \in \mathcal{R}_q$ with zero constant coefficient and computes a commitment to $\boldsymbol{g}$ defined as $\boldsymbol{t}_2 = \langle \vec{\boldsymbol{b}_2}, \vec{\boldsymbol{r}} \rangle + \boldsymbol{g}$. The prover also starts the opening proof with soundness error $1/q$ for the commitments and samples a vector of small polynomials $\vec{\boldsymbol{y}}$ and computes the commitment $\vec{\boldsymbol{w}} = \boldsymbol{B}_0\vec{\boldsymbol{y}}$. Then $\mathcal{P}$ sends $\boldsymbol{t}_2$ and $\vec{\boldsymbol{w}}$ to the verifier. Next, $\mathcal{V}$ generates and sends a uniformly random vector $\vec{\gamma} \in \mathbb{Z}_q^m$. Prover $\mathcal{P}$ can then compute the polynomial $\boldsymbol{f}$ defined above and $\boldsymbol{h} = \boldsymbol{f} + \boldsymbol{g}$. Furthermore, it sets $\boldsymbol{v} = \langle \mathsf{NTT}^{-1}(dA^T\vec{\gamma})\vec{\boldsymbol{b}_1} + \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{y}} \rangle$ and sends $\boldsymbol{h}, \boldsymbol{v}$ to $\mathcal{V}$. Then, the verifier generates a challenge $\boldsymbol{c} \xleftarrow{\$} C$ and sends it to the prover. Eventually, $\mathcal{P}$ sends a response $\vec{\boldsymbol{z}} = \vec{\boldsymbol{y}} + \boldsymbol{c}\vec{\boldsymbol{r}}$.

The verifier $\mathcal{V}$ first checks if $\vec{\boldsymbol{z}}$ consists of small polynomials and if $\boldsymbol{h}$ has constant coefficient equal to 0. Also, $\mathcal{V}$ checks that $\boldsymbol{B}_0\vec{\boldsymbol{z}} = \vec{\boldsymbol{w}} + \boldsymbol{c}\vec{\boldsymbol{t}_0}$ and

$$\langle \mathsf{NTT}^{-1}(dA^T\vec{\gamma})\vec{\boldsymbol{b}_1} + \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{z}} \rangle = \boldsymbol{v} + \boldsymbol{c}\left(\boldsymbol{\tau} + \boldsymbol{t}_2 - \boldsymbol{h}\right)$$

where $\boldsymbol{\tau}$ is computed as in Equation (5).

One observes that if $A\vec{s} \neq \vec{u}$ then the constant coefficient of $\boldsymbol{f}$ becomes a uniformly random element of $\mathbb{Z}_q$ not under the control of the prover. Thus, also the constant coefficient of $\boldsymbol{h} = \boldsymbol{f} + \boldsymbol{g}$ will be uniformly random because the constant coefficient of $\boldsymbol{g}$ is independent from the constant coefficient of $\boldsymbol{f}$. In particular, it will not be zero with probability $1 - 1/q$ – this can be detected by the verifier. Therefore, the probability that a malicious prover manages to cheat is equal to $1/q$.

3.2 Boosting Soundness by Mapping Down

More abstractly, in the above protocol we checked $\langle A\vec{s} - \vec{u}, \vec{\gamma}\rangle = 0$ by investigating whether $\mathsf{L}(\vec{\gamma})$ has a zero constant coefficient where $\mathsf{L} : \mathbb{Z}_q^m \to \mathcal{R}_q$ is defined as

$$\mathsf{L}(\vec{\gamma}) := \mathsf{NTT}^{-1}(dA^T\vec{\gamma})\check{\boldsymbol{s}} - \langle \vec{u}, \vec{\gamma}\rangle. \tag{6}$$

As we observed earlier, the constant coefficient of $\mathsf{L}(\vec{\gamma})$ is indeed $\langle A\vec{s} - \vec{u}, \vec{\gamma}\rangle$.

Now, suppose we can define k functions $\mathsf{L}_0, \ldots, \mathsf{L}_{k-1}$ with the following property. For any $0 \le \mu < k$ and $\vec{\gamma}_\mu \in \mathbb{Z}_q^m$, $\boldsymbol{p} = L_\mu(\vec{\gamma}_\mu) \in \mathcal{R}_q$ is a polynomial such that $p_0 = \ldots = p_{\mu-1} = p_{\mu+1} = \ldots = p_{k-1} = 0$ and $p_\mu = \langle A\vec{s} - \vec{u}, \vec{\gamma}_\mu\rangle$. This would mean that for $0 \le \mu < k$, the μ-th coefficient related to X^μ of the polynomial

$$\boldsymbol{f} = \mathsf{L}_0(\vec{\gamma}_0) + \mathsf{L}_1(\vec{\gamma}_1) + \ldots + \mathsf{L}_{k-1}(\vec{\gamma}_{k-1})$$

is equal to $\langle A\vec{s} - \vec{u}, \vec{\gamma_\mu}\rangle$. In particular, if $A\vec{s} = \vec{u}$ then $f_0 = f_1 = \ldots = f_{k-1} = 0$. Thus, in order to decrease the soundness error we can let the verifier $\mathcal{V}$ send k independently uniformly random vectors $\vec{\gamma}_0, \ldots, \vec{\gamma}_{k-1}$ and then $\mathcal{P}$ proves that $\boldsymbol{f} \in \mathcal{R}_q$ has the first k coefficients equal to zero. Note that we still need to find a way for $\mathcal{V}$ to compute a commitment to $\boldsymbol{f}$ from $\vec{\boldsymbol{t}}_1$ and $\vec{\gamma}_0, \ldots, \vec{\gamma}_{k-1}$.

Constructing L_μ. Let $\mathcal{S}_q$ be the $\mathbb{Z}_q$-submodule of $\mathcal{R}_q$ generated by X^k, i.e.

$$\mathcal{S}_q = \{p_0 + p_1X^k + \cdots + p_{d/k-1}X^{d-k} \in \mathcal{R}_q\} \subset \mathcal{R}_q.$$

We have $\mathcal{S}_q \cong \mathbb{Z}_q[X]/(X^{d/k} + 1)$. From Galois theory there is a corresponding subgroup H of $\mathsf{Aut}(\mathcal{R}_q)(\mathcal{R}_q)$ of order k such that $\sigma(\boldsymbol{p}) = \boldsymbol{p}$ for all $\sigma \in H$ if and only if $\boldsymbol{p} \in \mathcal{S}_q$. It is easy to see that this group is generated by $\sigma = \sigma_{2d/k+1} \in \mathsf{Aut}(\mathcal{R}_q)(\mathcal{R}_q)$, which is the same automorphism that we use in the automorphism opening proof. In fact this follows from the fact that $\mathrm{ord}(\sigma) = k$ and $\sigma(X^k) = X^{k(2d/k+1)} = X^k$.

We have the trace map $\mathsf{Tr}\colon \mathcal{R}_q \to \mathcal{S}_q$ given by

$$\mathsf{Tr}(\boldsymbol{p}) = \sum_{\nu=0}^{k-1} \sigma^\nu(\boldsymbol{p}).$$

Notice that the constant coefficient of $\mathsf{Tr}(\boldsymbol{p})$ is given by kp_0. Now define L_μ by

$$\mathsf{L}_\mu(\vec{\gamma}) = \frac{1}{k}X^\mu\mathsf{Tr}(\mathsf{L}(\vec{\gamma})) = \frac{1}{k}X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu\left(\mathsf{NTT}^{-1}(dA^T\vec{\gamma})\check{\boldsymbol{s}} - \langle \vec{u}, \vec{\gamma}\rangle\right).$$

If $\boldsymbol{p} = \mathsf{L}_\mu(\vec{\gamma})$, then $\boldsymbol{p}$ is of the form

$$\boldsymbol{p} = p_\mu X^\mu + p_{k+\mu}X^{k+\mu} + \cdots + p_{d-k+\mu}X^{d-k+\mu}$$

and thus has the property that the first k coefficients except the μ-th coefficient are zero. Moreover, it is clear from above that $p_\mu = \langle A\vec{s} - \vec{u}, \vec{\gamma}\rangle$.

Finally, given the commitment $\boldsymbol{t}_1$ to $\boldsymbol{s}$, the verifier can compute a commitment to $\boldsymbol{f} = \mathsf{L}_0(\vec{\gamma}_0) + \cdots + \mathsf{L}_{k-1}(\vec{\gamma}_{k-1})$ via

$$\begin{aligned}\boldsymbol{\tau} &= \sum_{\mu=0}^{k-1}\frac{1}{k}X^\mu \sum_{\nu=0}^{k-1}\sigma^\nu\left(\mathsf{NTT}^{-1}(dA^T\vec{\gamma}_\mu)\boldsymbol{t}_1 - \langle \vec{u}, \vec{\gamma}_\mu\rangle\right)\\ &= \sum_{\mu=0}^{k-1}\frac{1}{k}X^\mu \sum_{\nu=0}^{k-1}\sigma^\nu\left(\langle\mathsf{NTT}^{-1}(dA^T\vec{\gamma}_\mu)\vec{\boldsymbol{b}}_1, \vec{\boldsymbol{r}}\rangle\right) + \boldsymbol{f}.\end{aligned} \tag{7}$$

The Protocol. We present the protocol in Figure 2 with the verification algorithm given in Figure 3. Prover $\mathcal{P}$ starts by generating a uniformly random polynomial $\boldsymbol{g}$ satisfying $g_0 = \ldots = g_{k-1} = 0$ and computing a commitment $\boldsymbol{t}_2 = \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{r}} \rangle + \boldsymbol{g}$. Now the prover needs to start an opening proof with soundness $1/q^k$. Also it is going to prove a relation which involves the k automorphisms σ^i. Therefore it uses the automorphism opening proof from [ALS20] and samples vectors $\vec{\boldsymbol{y}}_0, \ldots, \vec{\boldsymbol{y}}_{k-1}$ of short polynomials that are going to be used to mask $\vec{\boldsymbol{r}}$ k times with challenges of the form $\sigma^i(\boldsymbol{c})$. Also, $\mathcal{P}$ computes $\vec{\boldsymbol{w}}_i = \boldsymbol{B}_0 \vec{\boldsymbol{y}}_i$. The prover sends $\boldsymbol{t}_2$ and $\vec{\boldsymbol{w}}_i$ to $\mathcal{V}$.

Next, the verifier selects uniformly random vectors $\vec{\gamma}_0, \ldots, \vec{\gamma}_{k-1} \in \mathbb{Z}_q^m$ and sends them to $\mathcal{P}$. Then, the prover computes

$$\boldsymbol{f} = \sum_{\mu=0}^{k-1} \mathsf{L}_\mu(\vec{\gamma}_\mu) = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\mathsf{NTT}^{-1}(dA^T \vec{\gamma}_\mu) \check{\boldsymbol{s}} - \langle \vec{u}, \vec{\gamma}_\mu \rangle \right).$$

By construction, $f_0 = \ldots = f_{k-1} = 0$. Note that $\mathcal{V}$ can compute a commitment $\boldsymbol{\tau}$ to $\boldsymbol{f}$ as explained above. Now the prover sets $\boldsymbol{h} = \boldsymbol{f} + \boldsymbol{g}$ and computes for $i = 0, \ldots, k-1$,

$$\boldsymbol{v}_i = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\langle \mathsf{NTT}^{-1}(dA^T \vec{\gamma}_\mu) \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{y}}_{i-\nu \bmod k} \rangle \right) + \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{y}}_i \rangle.$$

It sends $\boldsymbol{h}$ and $\boldsymbol{v}_0, \ldots, \boldsymbol{v}_{k-1}$. The verifier sends a random challenge polynomial $\boldsymbol{c} \xleftarrow{\$} C$. Eventually, $\mathcal{P}$ computes $\vec{\boldsymbol{z}}_i = \vec{\boldsymbol{y}}_i + \sigma^i(\boldsymbol{c}) \vec{\boldsymbol{r}}$ for $i = 0, \ldots, k-1$ and sends $\vec{\boldsymbol{z}}_0, \ldots, \vec{\boldsymbol{z}}_{k-1}$.

Verifier $\mathcal{V}$ first checks if for all $i = 0, \ldots, k-1$, $\vec{\boldsymbol{z}}_i$ is short, and

$$\boldsymbol{B}_0 \vec{\boldsymbol{z}}_i \stackrel{?}{=} \vec{\boldsymbol{w}}_i + \sigma^i(\boldsymbol{c}) \vec{\boldsymbol{t}}_0.$$

Then, $\mathcal{V}$ checks that $h_0, \ldots, h_{k-1}$ are all equal to zero and computes $\boldsymbol{\tau}$ as in (7). Finally, the verifier checks whether for all $i = 0, \ldots, k-1$,

$$\begin{aligned} &\sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\langle \mathsf{NTT}^{-1}(dA^T \vec{\gamma}_\mu) \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{z}}_{i-\nu \bmod k} \rangle \right) + \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{z}}_i \rangle \\ &= \boldsymbol{v}_i + \sigma^i(\boldsymbol{c})(\boldsymbol{\tau} + \boldsymbol{t}_2 - \boldsymbol{h}) \end{aligned}$$

to test whether $\boldsymbol{\tau} + \boldsymbol{t}_2 - \boldsymbol{h}$ really is a commitment to zero.

Security Analysis.

Theorem 3.1. *The protocol in Figure 2 is complete, computational honest verifier zero-knowledge under the Module-LWE assumption and computational special sound under the Module-SIS assumption. More precisely, let p be the maximum probability over $\mathbb{Z}_q$ of the coefficients of $\boldsymbol{c} \bmod X^k - \zeta^k$ as in Lemma 2.2.*

Then, for completeness, unless the honest prover $\mathcal{P}$ aborts due to the rejection sampling, it convinces the honest verifier $\mathcal{V}$ with overwhelming probability.

For zero-knowledge, there exists a simulator $\mathcal{S}$, that, without access to secret information, outputs a simulation of a non-aborting transcript of the protocol between $\mathcal{P}$ and $\mathcal{V}$. Then for every algorithm $\mathcal{A}$ that has advantage ε in distinguishing the simulated transcript from an actual transcript, there is an algorithm $\mathcal{A}'$ with the same running time that has advantage $\varepsilon - 2^{-100}$ in distinguishing $\mathsf{MLWE}_{\lambda,\chi}$.

For soundness, there is an extractor $\mathcal{E}$ with the following properties. When given rewindable black-box access to a deterministic prover $\mathcal{P}^*$ that convinces $\mathcal{V}$ with probability $\varepsilon \geq q^{-k} + p^k$, $\mathcal{E}$ either outputs a weak opening for the commitment $\vec{\boldsymbol{t}}$ with message $\check{\boldsymbol{s}}^*$, such that $A\mathsf{NTT}(\check{\boldsymbol{s}}^*) = \vec{u}$, or a $\mathsf{MSIS}_{\kappa,8d\beta}$ solution for $\boldsymbol{B}_0$ in expected time at most $1/\varepsilon + (d/k)(\varepsilon - p^k)^{-1}$ when running $\mathcal{P}^*$ once is assumed to take unit time.

Proof. Completeness. The distributions of the vectors $\vec{\boldsymbol{z}}_i$ sent by $\mathcal{P}$ are independent and have statistical distance at most 2^{-100} from $D_{\mathfrak{s}}^{(\lambda+\kappa+3)d}$ by Lemma 2.6. Lemma 2.7 implies that the bounds $\|\vec{\boldsymbol{z}}_i\|_2 \leq \beta =$

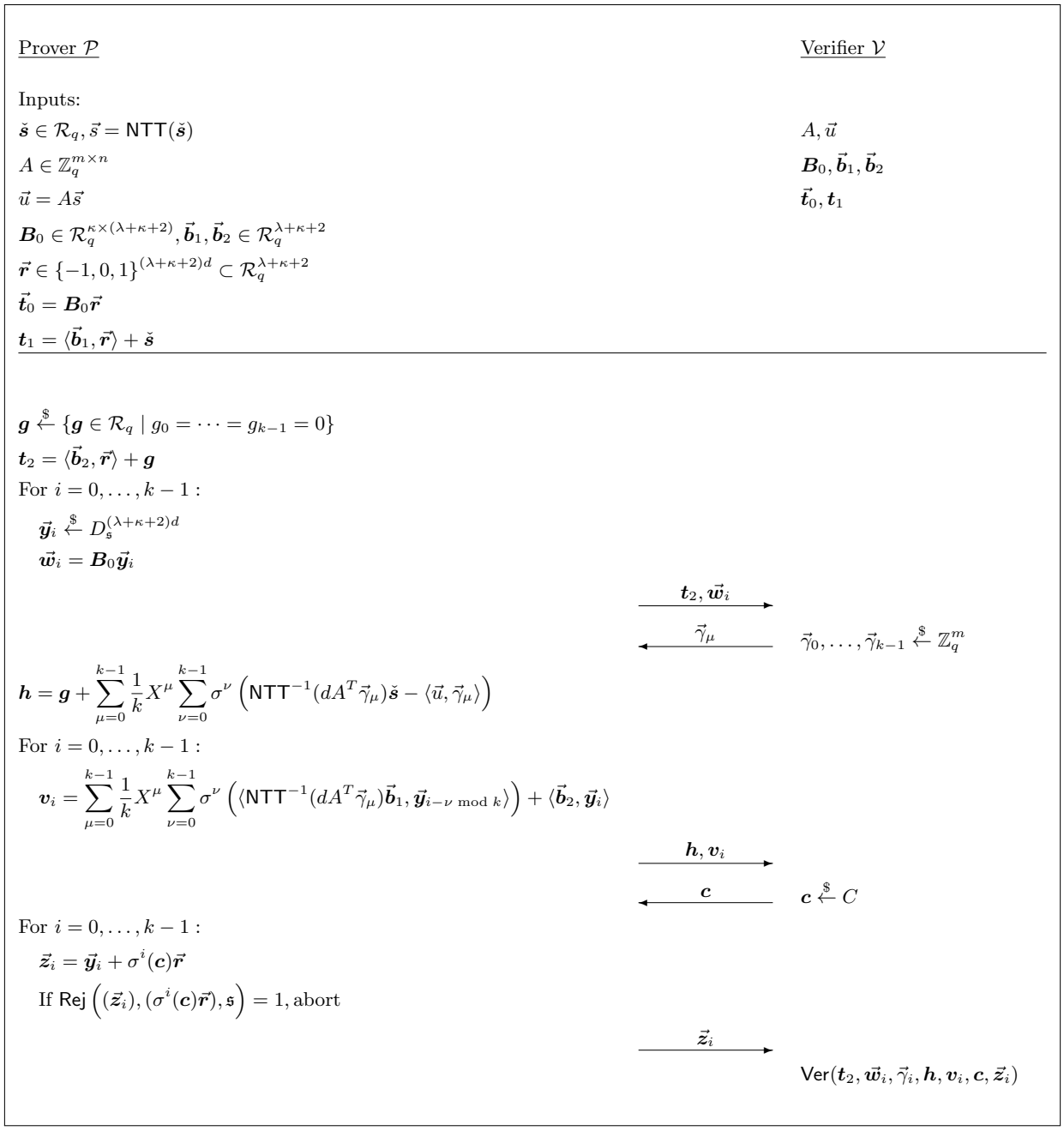

Fig. 2. Automorphism proof of knowledge of a solution to an unstructured linear equation over $\mathbb{Z}_q$. Verification equations are described in Figure 3.

$\mathfrak{s}\sqrt{2(\lambda+\kappa+3)d}$ are true with overwhelming probability. It follows by careful inspection that the other verification equations are always true for the messages sent by $\mathcal{P}$.

Zero-Knowledge. We can simulate a non-aborting transcript between the honest prover and the honest verifier in the following way. First, in a non-aborting transcript the vectors $\vec{\boldsymbol{z}}_i$ are statistically close to $D_{\mathfrak{s}}^{(\lambda+\kappa+2)d}$ by Lemma 2.6. So the simulator can just sample $\vec{\boldsymbol{z}}_i \xleftarrow{\$} D_{\mathfrak{s}}^{(\lambda+\kappa+2)d}$. Next, again by Lemma 2.6, we know that $\sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}$ is independent from $\vec{\boldsymbol{z}}_i$, and hence $\boldsymbol{c}$ is independent from the $\vec{\boldsymbol{z}}_i$. So, the simulator picks

$\mathsf{Ver}(\boldsymbol{t}_2, \vec{\boldsymbol{w}}_i, \vec{\gamma}_i, \boldsymbol{h}, \boldsymbol{v}_i, \boldsymbol{c}, \vec{\boldsymbol{z}}_i)$

01 For $i = 0, \ldots, k-1$:

02 $\quad \|\vec{\boldsymbol{z}}_i\|_2 \overset{?}{\leq} \beta = \mathfrak{s}\sqrt{2(\lambda+\kappa+2)d}$

03 $\quad \boldsymbol{B}_0\vec{\boldsymbol{z}}_i \overset{?}{=} \vec{\boldsymbol{w}}_i + \sigma^i(\boldsymbol{c})\vec{\boldsymbol{t}}_0$

04 $h_0 \overset{?}{=} \ldots \overset{?}{=} h_{k-1} \overset{?}{=} 0$

05 $\boldsymbol{\tau} = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\mathsf{NTT}^{-1}(dA^T\vec{\gamma}_\mu)\boldsymbol{t}_1 - \langle \vec{u}, \vec{\gamma}_\mu\rangle\right)$

06 For $i = 0, \ldots, k-1$:

07 $\quad \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\langle \mathsf{NTT}^{-1}(dA^T\vec{\gamma}_\mu)\boldsymbol{b}_1, \vec{\boldsymbol{z}}_{i-\nu \bmod k}\rangle\right) + \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{z}}_i\rangle \overset{?}{=} \boldsymbol{v}_i + \sigma^i(\boldsymbol{c})(\boldsymbol{\tau} + \boldsymbol{t}_2 - \boldsymbol{h})$

Fig. 3. Verification equations for Figure 2.

$\boldsymbol{c} \overset{\$}{\leftarrow} C$ like the honest verifier. The polynomial $\boldsymbol{h}$ is such that $h_0 = \cdots = h_{k-1} = 0$ in honest transcripts and the other coefficients are uniformly random because of the additive term $\boldsymbol{g}$. Hence, the simulator samples $\boldsymbol{h} \overset{\$}{\leftarrow} \{\boldsymbol{h} \in \mathcal{R}_q \mid h_0 = \cdots = k_{k-1} = 0\}$. Then, the challenges $\vec{\gamma}_\mu \in \mathbb{Z}_q^m$ are independently uniformly random and the simulator samples them in this way. Next, the commitment $\boldsymbol{t}_2$ is computationally indistinguishable from a uniformly random polynomial if MLWE_λ is hard. So the simulator can just take a uniformly random $\boldsymbol{t}_2 \in \mathcal{R}_q$. Now, in an honest transcript, the remaining messages $\vec{\boldsymbol{w}}_i$ and $\boldsymbol{v}_i$ are all uniquely determined by the verification equations because of completeness. We see that if the simulator computes these messages so that the verification equations become true, then the resulting transcript is indistinguishable from the honest transcript.

Soundness. First the extractor opens the commitments $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$. From [ALS20][Theorem 4.4], unless $\mathcal{E}$ finds a $\mathsf{MSIS}_{\kappa,8d\beta}$ solution, the extractor can compute vectors $\vec{\boldsymbol{y}}^*$ and $\vec{\boldsymbol{r}}^*$ such that for every accepting transcript with first messages $\boldsymbol{t}_2$ and $\vec{\boldsymbol{w}}_i$,

$$\boldsymbol{z}_i = \vec{\boldsymbol{y}}_i^* + \sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}^*.$$

The expected runtime for this equals the runtime in the theorem statement. Then let $\check{\boldsymbol{s}}^* \in \mathcal{R}_q$ and $\boldsymbol{g}^* \in \mathcal{R}_q$ be the extracted messages, which are defined by

$$\begin{aligned}\boldsymbol{t}_1 &= \langle \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{r}}^*\rangle + \check{\boldsymbol{s}}^*,\\ \boldsymbol{t}_2 &= \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{r}}^*\rangle + \boldsymbol{g}^*.\end{aligned}$$

Now substituting these expressions into τ gives

$$\boldsymbol{\tau} = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\langle \mathsf{NTT}^{-1}(dA^T\vec{\gamma}_\mu)\vec{\boldsymbol{b}}_1, \vec{\boldsymbol{r}}^*\rangle\right) + \boldsymbol{f}^*,$$

where

$$\boldsymbol{f}^* = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\mathsf{NTT}^{-1}(dA^T\vec{\gamma}_\mu)\check{\boldsymbol{s}}^* - \langle \vec{u}, \vec{\gamma}_\mu\rangle\right).$$

From the discussion in this section we know that $f_\mu^* = \langle A\vec{s}^* - \vec{u}, \vec{\gamma}_\mu\rangle$ for $\mu = 0, \ldots, k-1$, $\vec{s}^* = \mathsf{NTT}(\check{\boldsymbol{s}}^*)$. Next we find from the last verification equations,

$$\begin{aligned}&\left(\sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\langle \mathsf{NTT}^{-1}(dA^T\vec{\gamma}_\mu)\vec{\boldsymbol{b}}_1, \vec{\boldsymbol{y}}^*_{i-\nu \bmod k}\rangle\right) + \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{y}}^*\rangle - \boldsymbol{v}_i\right)\\ &= \sigma^i(\boldsymbol{c})\left(\boldsymbol{f}^* + \boldsymbol{g}^* - \boldsymbol{h}\right). \qquad (8)\end{aligned}$$

for all $i = 0, \ldots, k-1$. The coefficients of these linear polynomials in $\sigma^i(\boldsymbol{c})$ are independent from $\boldsymbol{c}$. We bound the success probability ε of the prover under the assumption $A\vec{s}^* \neq \vec{u}$. In this case the coefficients

f^*_μ for $\mu = 0, \ldots, k-1$ are uniformly random elements in $\mathbb{Z}_q$ in a random transcript. Hence, $f^*_\mu + g^*_\mu$ is uniformly random since $\boldsymbol{g}^*$ is independent from the $\vec{\gamma}_\mu$. Also we know that $h_\mu = 0$ in every accepting transcript. So, suppose $f^*_\mu + g^*_\mu - h^*_\mu = f^*_\mu + g^*_\mu \neq 0$ for some μ. Then there exists some $j \in \mathbb{Z}^\times_{2d}$ with $\boldsymbol{f}^* + \boldsymbol{g}^* - \boldsymbol{h} \bmod (X - \zeta^j) \neq 0$. Therefore, there is only one possible value modulo $(X^k - \zeta^{jk})$ for the challenge in such a transcript, otherwise Equation 8 can not be true for all i. Since the maximum probability of every coefficient of $\boldsymbol{c} \bmod (X^k - \zeta^{jk})$ is less than p we see that the success probability is bounded by

$$\begin{aligned}\varepsilon = \Pr\left[\text{accepting}\right] &< \left(\frac{1}{q}\right)^k + \Pr\left[\text{accepting} \,\middle|\, f^*_\mu + g^*_\mu \neq 0 \text{ for some } \mu\right] \\ &\leq \left(\frac{1}{q}\right)^k + p^k.\end{aligned}$$

This is in contradiction to the bound in the theorem statement and thus it must hold $A\vec{s}^* = \vec{u}$. □

3.3 General Case

Previously, we assumed that $n = d$ so that $\vec{s} = \mathsf{NTT}(\check{\boldsymbol{s}}) = \mathsf{NTT}(\check{\boldsymbol{s}}_1)$. When $n > d$, we slightly modify our approach. We have $\vec{s} = \mathsf{NTT}(\check{\boldsymbol{s}}_1) \parallel \cdots \parallel \mathsf{NTT}(\check{\boldsymbol{s}}_{n/d})$ and now also define polynomials $\boldsymbol{\psi}_j$ such that

$$A^T\vec{\gamma} = \mathsf{NTT}(\boldsymbol{\psi}_1) \parallel \cdots \parallel \mathsf{NTT}(\boldsymbol{\psi}_{n/d}).$$

Then the inner product $\langle A\vec{s}, \vec{\gamma}\rangle = \langle \vec{s}, A^T\vec{\gamma}\rangle$ can be written as a sum of smaller inner products. We find

$$\begin{aligned}\langle A\vec{s} - \vec{u}, \vec{\gamma}\rangle &= \sum_{j=1}^{n/d} \langle \mathsf{NTT}(\check{\boldsymbol{s}}_j), \mathsf{NTT}(\boldsymbol{\psi}_j)\rangle - \langle \vec{u}, \vec{\gamma}\rangle \\ &= \sum_{j=1}^{n/d} \sum_{i \in \mathbb{Z}^\times_{2d}} \check{\boldsymbol{s}}_j(\zeta^i)\boldsymbol{\psi}_j(\zeta^i) - \langle \vec{u}, \vec{\gamma}\rangle = \frac{1}{d} \sum_{i \in \mathbb{Z}^\times_{2d}} \left(\sum_{j=1}^{n/d} d\check{\boldsymbol{s}}_j \boldsymbol{\psi}_j - \langle \vec{u}, \vec{\gamma}\rangle \right)(\zeta^i).\end{aligned}$$

Next, similarly as before, we incorporate more challenges. So, for $\vec{\gamma}_0, \ldots, \vec{\gamma}_{k-1} \in \mathbb{Z}_q^m$ we write

$$A^T\vec{\gamma}_\mu = \mathsf{NTT}(\boldsymbol{\psi}_1^{(\mu)}) \| \cdots \| \mathsf{NTT}(\boldsymbol{\psi}_{n/d}^{(\mu)})$$

and then set

$$\boldsymbol{f} = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\sum_{j=1}^{n/d} d\boldsymbol{\psi}_j^{(\mu)} \boldsymbol{s}_j - \langle \vec{u}, \vec{\gamma}_\mu \rangle \right).$$

It holds that for $\mu = 0, \ldots, k-1$, $f_\mu = \langle A\vec{s} - \vec{u}, \vec{\gamma}_\mu\rangle$. Now, note that $\boldsymbol{\tau}$ defined as

$$\begin{aligned}\boldsymbol{\tau} &= \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\sum_{j=1}^{n/d} d\boldsymbol{\psi}_j^{(\mu)} \boldsymbol{t}_j - \langle \vec{u}, \vec{\gamma}_\mu \rangle \right) \\ &= \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\sum_{j=1}^{n/d} \langle d\boldsymbol{\psi}_j^{(\mu)} \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{r}} \rangle + d\boldsymbol{\psi}_j^{(\mu)} \check{\boldsymbol{s}}_j - \langle \vec{u}, \vec{\gamma} \rangle \right) \\ &= \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\left\langle \sum_{j=1}^{n/d} d\boldsymbol{\psi}_j^{(\mu)} \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{r}} \right\rangle \right) + \boldsymbol{f}\end{aligned}$$

is indeed a commitment to $\boldsymbol{f}$ and can be computed by the verifier.

3.4 Boosting Soundness by Going Up

We now present the second method to decrease the soundness error of the protocol from Section 3.1. This method is efficient if there are fewer secret coefficients than the ring dimension, i.e. if $n < d$. For example $n = 32$ and $d = 128$. Then it is better not to choose a completely splitting prime q so that the opening proof has negligible soundness error with only one repetition ($k = 1$). So assume $q-1 \equiv 2l \pmod{4l}$ with $l < d$, and $n = l$. In this case the analysis of the basic protocol from Section 3.1 does not apply directly and we can not use automorphisms to boost soundness by mapping down to a smaller ring. Instead we go the other direction. The prime q splits completely in the subring $\mathcal{S}_q = \{p_0 + p_1 X^{d/l} + \cdots + p_{l-1} X^{d-d/l} \in \mathcal{R}_q\} \cong \mathbb{Z}_q[X]/(X^l + 1)$ of $\mathcal{R}_q$. So we choose the secret polynomial $\check{\boldsymbol{s}}$ such that it lies in $\mathcal{S}_q$, which is the case if and only if the NTT vector $\mathsf{NTT}(\check{\boldsymbol{s}})$ lies in the subvector space $\mathbb{Z}_q^l$ of $(\mathbb{F}_{q^{d/l}})^l$. Then $\check{\boldsymbol{s}}$ encodes the l coefficients of $\vec{s}$. Our protocol assumes that there is a proof for this property. This can for example be part of the shortness proof since $\check{\boldsymbol{s}}(\check{\boldsymbol{s}} - \mathbf{1})(\check{\boldsymbol{s}} + \mathbf{1}) = \mathbf{0}$ shows that $\mathsf{NTT}(\check{\boldsymbol{s}})$ even lies in $\{-1, 0, 1\}^l \subset \mathbb{Z}_q^l \subset (\mathbb{F}_{q^{d/l}})^l$. With this setup the basic protocol using $\vec{\gamma} \in \mathbb{Z}_q^m$ proves the linear relation $A\vec{s} = \vec{u}$ with soundness error $1/q$. But now we can let $\vec{\gamma}$ be uniformly random over $\mathbb{F}_{q^{d/l}}$ and directly get negligible soundness error. Indeed, note that by Lemma 2.1,

$$\begin{aligned}
\langle A\vec{s} - \vec{u}, \vec{\gamma} \rangle_{\mathbb{F}_{q^{d/l}}} &= \langle \vec{s}, A^T\vec{\gamma} \rangle_{\mathbb{F}_{q^{d/l}}} - \langle \vec{u}, \vec{\gamma} \rangle_{\mathbb{F}_{q^{d/l}}} \\
&= \sum_{j \in \mathbb{Z}_{2l}^\times} \left(\check{\boldsymbol{s}} \mathsf{NTT}^{-1}(A^T\vec{\gamma}) \bmod (X^{d/l} - \zeta^j) \right) - \langle \vec{u}, \vec{\gamma} \rangle_{\mathbb{F}_{q^{d/l}}} \\
&= \frac{1}{l} \sum_{j \in \mathbb{Z}_{2l}^\times} (\boldsymbol{f} \bmod (X^{d/l} - \zeta^j)) = f_0 + f_1 X + \cdots + f_{d/l-1} X^{d/l-1},
\end{aligned}$$

where the scalar product is over the finite field $\mathbb{F}_{q^{d/l}}$ and the polynomial $\boldsymbol{f} \in \mathcal{R}_q$ is defined by $\boldsymbol{f} = \check{\boldsymbol{s}} \mathsf{NTT}^{-1}(lA^T\vec{\gamma}) - \langle \vec{u}, \vec{\gamma} \rangle$. The protocol is given in Figure 4.

4 Main Protocol

In this section we present our main protocol for proving knowledge of a ternary solution $\vec{s} \in \{-1, 0, 1\}^n$ to an arbitrary linear equation $A\vec{s} = \vec{u}$ over $\mathbb{Z}_q$. The protocol is essentially an amalgamation of the linear proof from Section 3 and the product proof from [ALS20]. We use a fully splitting prime q and automorphisms to boost the soundness. So, at a high level the prover commits to n/d polynomials $\check{\boldsymbol{s}}_j$ whose NTT coefficients are the coefficients of $\vec{s}$. That is,

$$\vec{s} = \begin{pmatrix} \mathsf{NTT}(\check{\boldsymbol{s}}_1) \\ \vdots \\ \mathsf{NTT}(\check{\boldsymbol{s}}_{n/d}) \end{pmatrix}.$$

Then the prover uses the obvious generalization of the product proof to many cubic relations to show that

$$\check{\boldsymbol{s}}_j(\check{\boldsymbol{s}}_j + \mathbf{1})(\check{\boldsymbol{s}}_j - \mathbf{1}) = \mathbf{0}$$

for all j. This shows that $\mathsf{NTT}(\check{\boldsymbol{s}}_j) \in \{-1, 0, 1\}^d$ since the polynomial product in $\mathcal{R}_q$ is coefficient-wise in the NTT representation. This is the technique that was used in [BLS19].

In parallel, the prover uses the linear proof for the general case from Section 3.3, to show that the polynomials $\check{\boldsymbol{s}}_j$ really give a solution to the linear equation. The complete protocol is given in Figure 5 and it is proven secure in Theorem 4.1.

4.1 Security Analysis

Theorem 4.1. *The protocol in Figure 5 is complete, computational honest verifier zero-knowledge under the Module-LWE assumption and computational special sound under the Module-SIS assumption. More precisely, let p be the maximum probability over $\mathbb{Z}_q$ of the coefficients of $\boldsymbol{c} \bmod X^k - \zeta^k$ as in Lemma 2.2.*

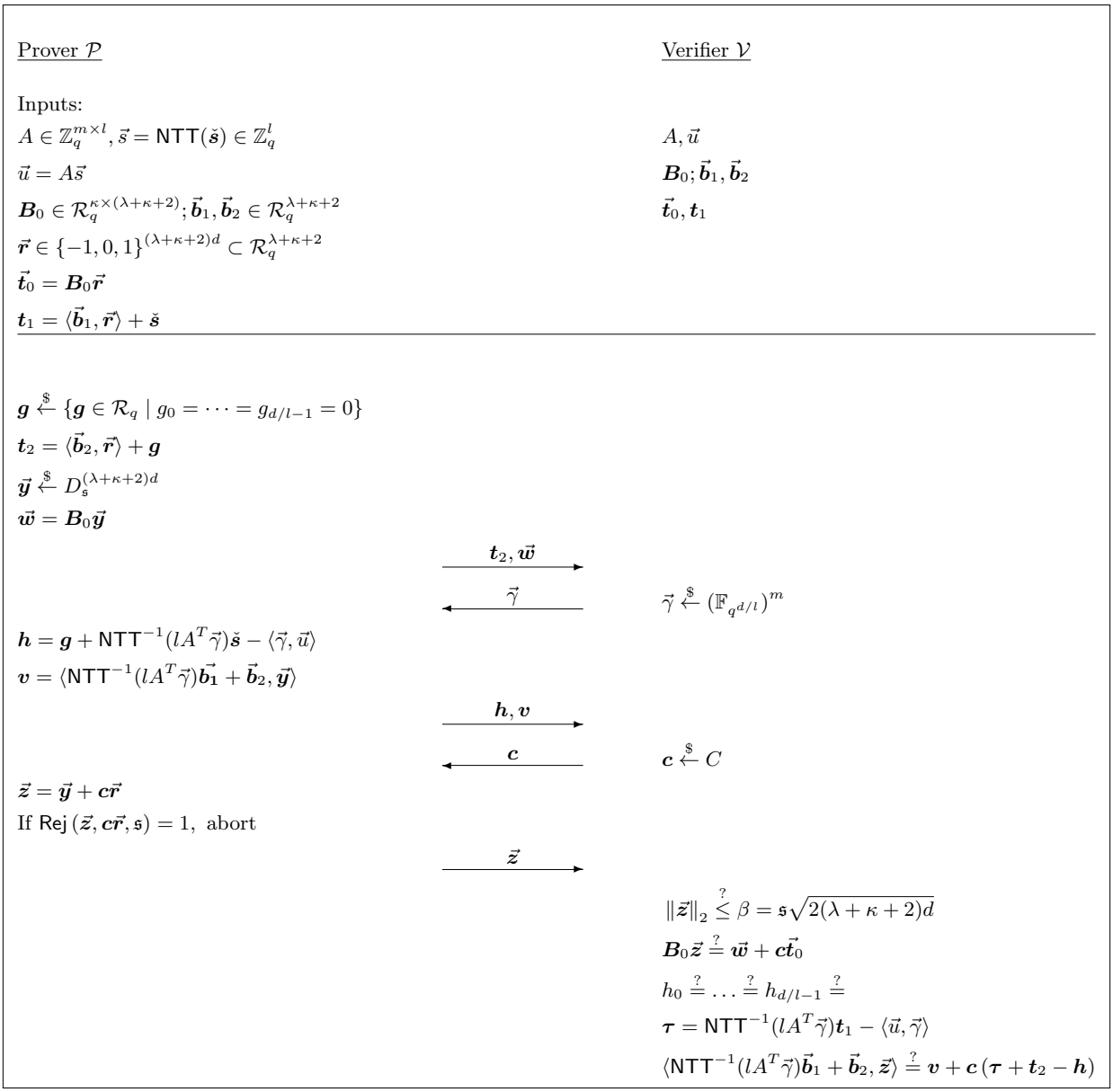

Fig. 4. Simple proof of unstructured linear relations among $l \mid d$ committed integers. The prime q is such that $q \equiv 2l \pmod{4l}$ and hence splits into l prime ideals in the ring $\mathcal{R}$.

Then, for completeness, in case the honest prover $\mathcal{P}$ does not abort due to rejection sampling, it convinces the honest verifier $\mathcal{V}$ with overwhelming probability.

For zero-knowledge, there exists a simulator $\mathcal{S}$, that, without access to secret information, outputs a simulation of a non-aborting transcript of the protocol between $\mathcal{P}$ and $\mathcal{V}$. Then for every algorithm $\mathcal{A}$ that has advantage ε in distinguishing the simulated transcript from an actual transcript, there is an algorithm $\mathcal{A}'$ with the same running time that has advantage $\varepsilon - 2^{-100}$ in distinguishing $\mathsf{MLWE}_{\lambda,\chi}$.

For soundness, there is an extractor $\mathcal{E}$ with the following properties. When given rewindable black-box access to a deterministic prover $\mathcal{P}^*$ that convinces $\mathcal{V}$ with probability $\varepsilon > (3p)^k$, $\mathcal{E}$ either outputs a solution

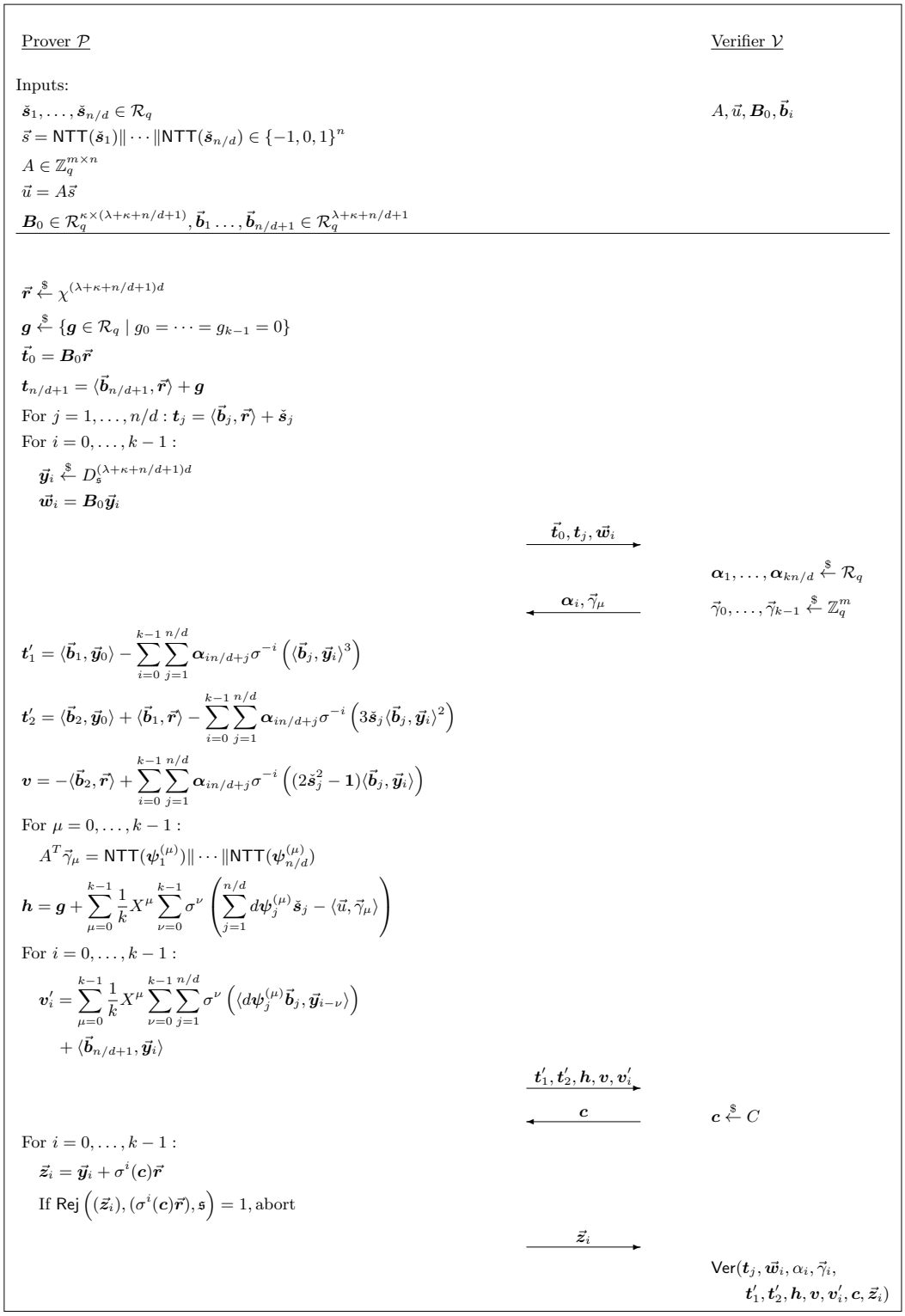

Fig. 5. Proof of knowledge of a ternary solution to an unstructured linear equation over $\mathbb{Z}_q$. Verification equations are defined in Figure 3.

$\mathsf{Ver}(\boldsymbol{t}_j, \vec{\boldsymbol{w}}_i, \boldsymbol{\alpha}_i, \vec{\gamma}_i, \boldsymbol{t}'_1, \boldsymbol{t}'_2, \boldsymbol{h}, \boldsymbol{v}, \boldsymbol{v}'_i, \boldsymbol{c}, \vec{\boldsymbol{z}}_i)$

For $i = 0, \dots, k-1$:

$\quad \|\vec{\boldsymbol{z}}_i\|_2 \overset{?}{\leq} \beta = \mathfrak{s}\sqrt{2(\lambda + \kappa + n/d + 1)d}$

$\quad \boldsymbol{B}_0 \vec{\boldsymbol{z}}_i \overset{?}{=} \vec{\boldsymbol{w}}_i + \sigma^i(\boldsymbol{c})\vec{\boldsymbol{t}}_0$

For $i = 0, \dots, k-1$:

$\quad$ For $j = 1, \dots, n/d$:

$\quad\quad \boldsymbol{f}_j^{(i)} = \langle \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{z}}_i \rangle - \sigma^i(\boldsymbol{c})\boldsymbol{t}_j$

$\boldsymbol{f}'_1 = \langle \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{z}}_0 \rangle - \boldsymbol{t}'_1$

$\boldsymbol{f}'_2 = \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{z}}_0 \rangle - \boldsymbol{t}'_2$

$\sum_{i=0}^{k-1} \sum_{j=1}^{n/d} \boldsymbol{\alpha}_{in/d+j} \sigma^{-i} \left(\boldsymbol{f}_j^{(i)} (\boldsymbol{f}_j^{(i)} + \sigma^i(\boldsymbol{c}))(\boldsymbol{f}_j^{(i)} - \sigma^i(\boldsymbol{c})) \right) - \boldsymbol{f}'_1 - \boldsymbol{c}\boldsymbol{f}'_2 \overset{?}{=} \boldsymbol{c}^2 \boldsymbol{v}$

For $\mu = 0, \dots, k-1$:

$\quad h_\mu \overset{?}{=} 0$

$\quad A^T \vec{\gamma}_\mu = \mathsf{NTT}(\boldsymbol{\psi}_1^{(\mu)}) \| \cdots \| \mathsf{NTT}(\boldsymbol{\psi}_{n/d}^{(\mu)})$

$\boldsymbol{\tau} = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\sum_{j=1}^{n/d} d\boldsymbol{\psi}_j^{(\mu)} \boldsymbol{t}_j - \langle \vec{u}, \vec{\gamma}_\mu \rangle \right)$

For $i = 0, \dots, k-1$:

$\quad \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sum_{j=1}^{n/d} \sigma^\nu \left(d\boldsymbol{\psi}_j^{(\mu)} \langle \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{z}}_{i-\nu \bmod k} \rangle \right) + \langle \vec{\boldsymbol{b}}_{n/d+1}, \vec{\boldsymbol{z}}_i \rangle$

$\quad\quad \overset{?}{=} \boldsymbol{v}'_i + \sigma^i(\boldsymbol{c})(\boldsymbol{\tau} + \boldsymbol{t}_{n/d+1} - \boldsymbol{h})$

Fig. 6. Verification equations for Figure 5.

$\vec{s}^* \in \{-1, 0, 1\}^n$ *to* $A\vec{s}^* = \vec{u}$*, or a* $\mathrm{MSIS}_{\kappa, 8d\beta}$ *solution for* $\boldsymbol{B}_0$ *in expected time at most* $1/\varepsilon + (\varepsilon - p^k)^{-1}$ *when running* $\mathcal{P}^*$ *once is assumed to take unit time.*

Proof. Completeness. The distributions of the vectors $\vec{\boldsymbol{z}}_i$ sent by $\mathcal{P}$ are independent and have statistical distance at most 2^{-100} from $D_{\mathfrak{s}}^{(\lambda+\kappa+n/d+1)d}$ by Lemma 2.6. Lemma 2.7 implies that the bounds $\|\vec{\boldsymbol{z}}_i\|_2 \leq \beta = \mathfrak{s}\sqrt{2(\lambda + \kappa + n/d + 1)d}$ are true with overwhelming probability. It follows by careful inspection that the other verification equations are always true for the messages sent by $\mathcal{P}$.

Zero-Knowledge. We can simulate a non-aborting transcript between the honest prover and the honest verifier in the following way. First, in a non-aborting transcript the vectors $\vec{\boldsymbol{z}}_i$ are statistically close to $D_{\mathfrak{s}}^{(\lambda+\kappa+n/d+1)d}$ by Lemma 2.6. So the simulator can just sample $\vec{\boldsymbol{z}}_i \overset{\$}{\leftarrow} D_{\mathfrak{s}}^{(\lambda+\kappa+n/d+1)d}$. Next, again by Lemma 2.6, we know that $\sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}$ is independent from $\vec{\boldsymbol{z}}_i$, and hence $\boldsymbol{c}$ is independent from the $\vec{\boldsymbol{z}}_i$. So, the simulator picks $\boldsymbol{c} \overset{\$}{\leftarrow} C$ like the honest verifier. The polynomial $\boldsymbol{h}$ is such that $h_0 = \cdots = h_{k-1} = 0$ in honest transcripts and the other coefficients are uniformly random because of the additive term $\boldsymbol{g}$. Hence, the simulator samples $\boldsymbol{h} \overset{\$}{\leftarrow} \{\boldsymbol{h} \in \mathcal{R}_q \mid h_0 = \cdots = k_{k-1} = 0\}$. Then, the challenges $\boldsymbol{\alpha}_i \in \mathcal{R}_q$ and $\vec{\gamma}_\mu \in \mathbb{Z}_q^m$ are independently uniformly random and the simulator samples them in this way. Next, the commitments $\boldsymbol{t}'_1$, $\boldsymbol{t}'_2$ and $\boldsymbol{t}_j$ are computationally indistinguishable from uniformly random polynomials if MLWE_λ is hard. In fact, they include independent Module-LWE samples. So the simulator can just take uniformly random $\boldsymbol{t}'_1, \boldsymbol{t}'_2, \boldsymbol{t}_j \in \mathcal{R}_q$. Now, in an honest transcript, the remaining messages $\vec{\boldsymbol{w}}_i$, $\boldsymbol{v}$, $\boldsymbol{v}'_i$ are all uniquely determined by the verification equations because of completeness. We see that if the simulator computes these messages so that the verification equations become true, then the resulting transcript is indistinguishable from an honest transcript.

Soundness. First the extractor opens the commitments $\boldsymbol{t}_j$ and $\boldsymbol{t}'_1$, $\boldsymbol{t}'_2$. From [ALS20][Theorem 4.4], unless $\mathcal{E}$ has found a $\mathsf{MSIS}_{\kappa, 8d\beta}$ solution, the extractor can compute vectors $\vec{\boldsymbol{y}}_i^*$ and $\vec{\boldsymbol{r}}^*$ such that for every accepting transcript with first messages $\boldsymbol{t}_j$, $\vec{\boldsymbol{w}}_i$,

$$\boldsymbol{z}_i = \vec{\boldsymbol{y}}_i^* + \sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}^*.$$

The expected runtime for this is equal to the runtime given in the theorem statement. Then let $\check{\boldsymbol{s}}_j^*$, $\boldsymbol{g}^*$, $\boldsymbol{m}_1^{*\prime}$ and $\boldsymbol{m}_2^{*\prime}$ be the extracted messages, which are such that

$$\begin{aligned}
\boldsymbol{t}_j &= \langle \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{r}}^* \rangle + \check{\boldsymbol{s}}_j^* \quad \text{for } j = 1, \ldots, n/d, \\
\boldsymbol{t}_{\frac{n}{d}+1} &= \langle \vec{\boldsymbol{b}}_{\frac{n}{d}+1}, \vec{\boldsymbol{r}}^* \rangle + \boldsymbol{g}^*, \\
\boldsymbol{t}_1' &= \langle \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{y}}_0^* \rangle + \boldsymbol{m}_1^{*\prime}, \\
\boldsymbol{t}_2' &= \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{y}}_0^* \rangle + \boldsymbol{m}_2^{*\prime}.
\end{aligned}$$

Now substituting these expressions into $\boldsymbol{f}_j^{(i)}$, $\boldsymbol{f}_1'$, $\boldsymbol{f}_2'$ as computed in the verification algorithm gives

$$\begin{aligned}
\boldsymbol{f}_j^{(i)} &= \langle \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{y}}_i^* \rangle - \sigma^i(\boldsymbol{c}) \check{\boldsymbol{s}}_j^*, \\
\boldsymbol{f}_1' &= \langle \vec{\boldsymbol{b}}_1, \boldsymbol{c}\vec{\boldsymbol{r}}^* \rangle - \boldsymbol{m}_1^{*\prime}, \\
\boldsymbol{f}_2' &= \langle \vec{\boldsymbol{b}}_2, \boldsymbol{c}\vec{\boldsymbol{r}}^* \rangle - \boldsymbol{m}_2^{*\prime}.
\end{aligned}$$

Next, the verification equation in Line 9 of the verification algorithm reads

$$\begin{aligned}
&\left(\sum_{i=0}^{k-1} \sum_{j=1}^{n/d} \boldsymbol{\alpha}_{in/d+j} \sigma^{-i} \left(\langle \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{y}}_i^* \rangle^3 \right) + \boldsymbol{m}_1^{*\prime} \right) \\
&+ \boldsymbol{c} \left(\sum_{i=0}^{k-1} \sum_{j=1}^{n/d} \boldsymbol{\alpha}_{in/d+j} \sigma^{-i} \left(3 \langle \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{y}}_i \rangle^2 \check{\boldsymbol{s}}_j^* \right) - \langle \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{r}}^* \rangle + \boldsymbol{m}_2^{*\prime} \right) \\
&+ \boldsymbol{c}^2 \left(\sum_{i=0}^{k-1} \sum_{j=1}^{n/d} \boldsymbol{\alpha}_{in/d+j} \sigma^{-i} \left(\langle \vec{\boldsymbol{b}}_j, \vec{\boldsymbol{y}}_i^* \rangle (2(\check{\boldsymbol{s}}_j^*)^2 - \mathbf{1}) \right) - \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{r}}^* \rangle - \boldsymbol{v} \right) \\
&+ \boldsymbol{c}^3 \left(\sum_{i=0}^{k-1} \sum_{j=1}^{n/d} \boldsymbol{\alpha}_{in/d+j} \sigma^{-i} \left(\check{\boldsymbol{s}}_j^* (\check{\boldsymbol{s}}_j^* - \mathbf{1})(\check{\boldsymbol{s}}_j^* + \mathbf{1}) \right) \right) \\
&= \mathbf{0}.
\end{aligned}$$

If we assume that $\check{\boldsymbol{s}}_j^*(\check{\boldsymbol{s}}_j^* - \mathbf{1})(\check{\boldsymbol{s}}_j^* + \mathbf{1}) \neq 0$ for some j, then following the same argument as in [ALS20, Theorem 5.1], the success probability of the prover must be bounded by

$$\varepsilon \leq \sum_{i=0}^{k} \binom{k}{i} \left(\frac{1}{q} \right)^i \left(1 - \frac{1}{q} \right)^{k-i} 2^{k-i} q^i p^k < (3p)^k.$$

This is not the case and therefore $\vec{s}_j^* = \mathsf{NTT}(\check{\boldsymbol{s}}_j^*) \in \{-1, 0, 1\}^d$ for all j.

Now substituting $\boldsymbol{t}_j$ into τ gives

$$\boldsymbol{\tau} = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\sum_{j=1}^{n/d} \langle d\boldsymbol{\psi}_j^{(\mu)} \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{r}}^* \rangle \right) + \boldsymbol{f}^*.$$

where

$$\boldsymbol{f}^* = \sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sigma^\nu \left(\sum_{j=1}^{n/d} d\boldsymbol{\psi}_j^{(\mu)} \check{\boldsymbol{s}}_j^* - \langle \vec{u}, \vec{\gamma}_\mu \rangle \right)$$

We know that $f^*_\mu = \langle A\vec{s}^* - \vec{u}, \vec{\gamma}_\mu \rangle$ for $\mu = 0, \ldots, k-1$ and $\vec{s}^* = \vec{s}^*_j \| \cdots \| \vec{s}^*_j$. Next we find from the last verification equations,

$$\left(\sum_{\mu=0}^{k-1} \frac{1}{k} X^\mu \sum_{\nu=0}^{k-1} \sum_{j=1}^{n/d} \sigma^\nu \left(\langle d\boldsymbol{\psi}_j^{(\mu)} \vec{\boldsymbol{b}}_1, \vec{\boldsymbol{y}}^*_{i-\nu \bmod k} \rangle \right) + \langle \vec{\boldsymbol{b}}_2, \vec{\boldsymbol{y}}^* \rangle - \boldsymbol{v}'_i \right) \tag{9}$$

$$= \sigma^i(\boldsymbol{c}) \left(\boldsymbol{f}^* + \boldsymbol{g}^* - \boldsymbol{h} \right). \tag{10}$$

for all $i = 0, \ldots, k-1$. The coefficients of these linear polynomials in $\sigma^i(\boldsymbol{c})$ are independent from $\boldsymbol{c}$. With the same reasoning as in the proof of Theorem 3.1 it follows that if $A\vec{s}^* \neq \vec{u}$, then

$$\varepsilon < \left(\frac{1}{q} \right)^k + p^k$$

in contradiction to the bound in the statement. Hence $A\vec{s}^* = \vec{u}$.

4.2 Proof Size

We now look at the size of the non-interactive proof outputs created by the protocol in Figure 2. First, note that for the non-interactive proof $\boldsymbol{w}_i$'s, $\boldsymbol{v}$ and $\boldsymbol{v}'_i s$ need not be included in the output as they are uniquely determined by the remaining components. Further, the challenges can be generated from a small seed of 256 bits, which itself is generated as the hash of some components. Therefore, the contribution of the challenges to the total proof length is extremely small and thus we neglect it.

As "full-sized" elements of $\mathcal{R}_q$, we have $\boldsymbol{t}_j$'s, $\boldsymbol{t}'_1, \boldsymbol{t}'_2$, and $\boldsymbol{h}$ (in fact, $\boldsymbol{h}$ is missing k coefficients, but that is a negligible consideration). Therefore, we have in total

$$n/d + 1 + \kappa + 3$$

full-sized elements of $\mathcal{R}_q$, which altogether costs

$$(n/d + \kappa + 4) \cdot d \log q \quad \text{bits.}$$

Now, the only remaining part is $\vec{\boldsymbol{z}}_i$'s. Due to rejection sampling, each coefficient of $\vec{\boldsymbol{z}}_i$ follows a Gaussian distribution with standard deviation $\mathfrak{s}$ since each coefficient of $\vec{\boldsymbol{y}}_i$ is sampled from $D_{\mathfrak{s}}$. Therefore, similar to prior works, e.g., [BLS19], we can bound each coefficient in absolute value by $6\mathfrak{s}$. If we then take into account the total number of coefficients in $\vec{\boldsymbol{z}}_i$ and an additional sign bit for each coefficient, then we get

$$k \cdot ((\lambda + \kappa + 1) \cdot d + n) \cdot \log(12\mathfrak{s})$$

bits of communication required for all $\vec{\boldsymbol{z}}_i$'s together.

For the rejection sampling, we set $\mathfrak{s} = 2T$, where T is a bound on the Euclidean norm of the concatenated vector $\left(\sigma^0(\boldsymbol{c})\vec{\boldsymbol{r}}, \ldots, \sigma^{k-1}(\boldsymbol{c})\vec{\boldsymbol{r}}\right)$. Therefore, we assume here that there is a single rejection sampling step done on all k $\vec{\boldsymbol{z}}_i$'s together. It is easy to see that no coefficient of the product $\sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}$ can exceed d for any $0 \leq i \leq k-1$. Therefore, we have the following theoretical bound

$$\left\| \left(\sigma^0(\boldsymbol{c})\vec{\boldsymbol{r}}, \ldots, \sigma^{k-1}(\boldsymbol{c})\vec{\boldsymbol{r}}\right) \right\|_2 \leq d \cdot \sqrt{k \cdot ((\lambda + \kappa + 1) \cdot d + n)} =: T. \tag{11}$$

Note that by setting T dependant on k, the average number of iterations in the protocol remains the same for varying k.

In conclusion, the overall proof length is about

$$(n/d + \kappa + 4) \cdot d \log q \;+\; k \cdot ((\lambda + \kappa + 1) \cdot d + n) \cdot \log(12\mathfrak{s}) \quad \text{bits,} \tag{12}$$

for $\mathfrak{s} = 2 \cdot d\sqrt{k \cdot ((\lambda + \kappa + 1) \cdot d + n)}$.

An important advantage of our proof system is that the proof length (i.e., the communication size) is *independent* of the height, m, of the matrix A.

Proof length optimizations. The proof length calculation in (12) does not take into account the fact that a (truncated) discrete Gaussian with known standard deviation has less entropy than the uniform distribution. Therefore, for concrete sizes to be described in the applications, we compute the entropy of a discrete Gaussian coefficient from $D_{\mathfrak{s}}$ and use the corresponding value instead of $\log(12\mathfrak{s})$ above in (12). One can encode $\vec{\boldsymbol{z}}_i$'s using, for example, a Huffman code to realize this.

Another optimization we employ is in the calculation of a maximum absolute coefficient in $\sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}$. In our applications, we aim to minimize d and set $d = 128$. Now in this case, a coefficient of $\sigma^i(\boldsymbol{c})\vec{\boldsymbol{r}}$ is the sum of 128 coefficients with i.i.d. $P(-1) = P(1) = 5/32$ and $P(0) = 22/32$.[5] If we calculate the convolution of this distribution, we find that a coefficient is bigger than 78 in absolute value with probability less than 2^{-114}. Hence, by a union bound the probability that any of the coefficients in $\left(\sigma^0(\boldsymbol{c})\vec{\boldsymbol{r}}, \ldots, \sigma^{k-1}(\boldsymbol{c})\vec{\boldsymbol{r}}\right)$ is bigger than 78 will still be negligibly small. Therefore, we can set $\mathfrak{s} = 2 \cdot 78\sqrt{k \cdot ((\lambda + \kappa + 1) \cdot d + n)}$ instead (when $d = 128$).

5 Applications

5.1 Proving Knowledge of LWE Secrets

As also considered in [BLS19], the first application of our proofs is to prove knowledge of secrets in LWE samples. For a fair comparison, we consider the same setting as in [BLS19]. That is, for $n = 2048$, we want to prove knowledge of a ternary vector $\vec{s} \in \{-1, 0, 1\}^n$ such that

$$\vec{u} = (A' \,\|\, I_m) \cdot \vec{s} \pmod{q},$$

where I_m is the m-dimensional identity matrix, $A' \in \mathbb{Z}_q^{m \times (n-m)}$ is a public matrix chosen uniformly at random and q is a modulus of about 32 bits (i.e., $\log q = 32$). Note that $\vec{s}$ here corresponds to the concatenation of a secret vector and an error vector of 1024 dimension each in the usual LWE setting. Let us denote $A = (A' \,\|\, I_m)$. This setting is now precisely the one of the protocol in Figure 2 with $\vec{u} = A\vec{s} \bmod q$, $n = 2048$ and $q \approx 2^{32}$.

First, to reach 128-bit security level we set $k = 128/\log q = 4$. Then, to optimize the proof length, we need to set $d = \dim(\mathcal{R}_q)$ as small as possible. This is due to the fact that regardless of what level of security is desired, each "garbage term", namely $\boldsymbol{t}'_1, \boldsymbol{t}'_2, \boldsymbol{h}$, requires $d \log q$ bits of storage. Using Lemma 2.2, the smallest possible d we can choose is 128, thus we set $d = 128$. The remaining task is to choose λ and κ to make M-LWE and M-SIS hard in practice against known attacks.

As in prior works (cf. [ESS+19, ESLL19, BLS19]), we measure the hardness of these problems in terms of root Hermite factor δ, aim for $\delta \approx 1.0045$ and follow an estimation strategy as in the recent works [ESS+19, ESLL19], where the authors also aimed for about 128-bit security. Particularly, using the "LWE estimator" in [APS15] and the SIS practical security estimation methodology in [MR09], we can reach the desired security level with $\lambda = 10$ and $\kappa = 9$ for a root Hermite factor (for both SIS and LWE) of $\delta \approx 1.0045$. An advantage of our construction here over [BLS19] is that setting of overall SIS/LWE dimension (which needs to be a multiple of d) is more flexible as we can use a relatively small d of 128.

Plugging in this parameter setting into (12) (with the described optimizations) yields a proof length of 51.44 KB. As a result, we achieve an improvement of almost 7.5× over the proposal in [BLS19] in terms of proof length. There are further results in [YAZ+19] similar to [BLS19], but the concrete proof length is not provided for this particular scenario. We refer to Appendix A for more on applications.

References

ALS20. Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments., 2020. https://eprint.iacr.org/2020/?

[5] Recall that a coefficient of $\boldsymbol{c}$ is zero with probability 1/2 and a coefficient of $\vec{\boldsymbol{r}}$ is zero with probability 6/16. The probabilities of ± 1 are always equal to each other.

APS15. Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

Ban93. Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1):625–635, Dec 1993.

BBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046, 2018. `https://eprint.iacr.org/2018/046`.

BCR+19. Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 103–128. Springer, Heidelberg, May 2019.

BDK+18. Joppe W. 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 *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 *SCN*, pages 368–385, 2018.

BLS19. Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 176–202. Springer, Heidelberg, August 2019.

Cv91. David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer, Heidelberg, April 1991.

DKL+18. Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR TCHES*, 2018(1):238–268, 2018. `https://tches.iacr.org/index.php/TCHES/article/view/839`.

dLNS17. Rafaël del Pino, Vadim Lyubashevsky, Gregory Neven, and Gregor Seiler. Practical quantum-safe voting from lattices. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1565–1581. ACM Press, October / November 2017.

dLS18. Rafaël del Pino, Vadim Lyubashevsky, and Gregor Seiler. Lattice-based group signatures and zero-knowledge proofs of automorphism stability. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 574–591. ACM Press, October 2018.

ESLL19. Muhammed F. Esgin, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 115–146. Springer, Heidelberg, August 2019.

ESS+19. Muhammed F. Esgin, Ron Steinfeld, Amin Sakzad, Joseph K. Liu, and Dongxi Liu. Short lattice-based one-out-of-many proofs and applications to ring signatures. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 67–88. Springer, Heidelberg, June 2019.

EZS+19. Muhammed F. Esgin, Raymond K. Zhao, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. MatRiCT: Efficient, scalable and post-quantum blockchain confidential transactions protocol. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 567–584. ACM Press, November 2019.

Kil92. Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *24th ACM STOC*, pages 723–732. ACM Press, May 1992.

KTX08. Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In Josef Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 372–389. Springer, Heidelberg, December 2008.

LM06. Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In *ICALP (2)*, pages 144–155, 2006.

LN17. Vadim Lyubashevsky and Gregory Neven. One-shot verifiable encryption from lattices. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 293–323. Springer, Heidelberg, April / May 2017.

LNSW13. San Ling, Khoa Nguyen, Damien Stehlé, and Huaxiong Wang. Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 107–124. Springer, Heidelberg, February / March 2013.

LPR10. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, pages 1–23, 2010.

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

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

Lyu09. Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

Lyu12. Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, Heidelberg, April 2012.

MR09. Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer, 2009.

PR06. Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, pages 145–166, 2006.

RST01. Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 552–565. Springer, Heidelberg, December 2001.

Sch90. Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, Heidelberg, August 1990.

Ste94. Jacques Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 13–21. Springer, Heidelberg, August 1994.

YAZ+19. Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 147–175. Springer, Heidelberg, August 2019.

Auxiliary Supporting Material

A More on Applications

A.1 Proof of Plaintext Knowledge

In a proof of plaintext knowledge (also called a verifiable encryption), the goal is to produce a ciphertext and a zero-knowledge proof such that the decryption of a valid ciphertext is guaranteed to yield a plaintext known by the prover.

The only lattice-based verifiable encryption scheme with a satisfactory practical efficiency is presented in [LN17]. Although this proposal is very efficient in practice, it has some undesirable properties. First, the guarantee on the message $\vec{\boldsymbol{m}}'$ decrypted from a valid ciphertext is *relaxed* in a way that $\vec{\boldsymbol{m}}'$ only satisfies an "approximate" lattice relation. Second, the running time of the decryption algorithm is dependant on the running time of the prover and only the *expected* number of decryption tries is theoretically investigated.

Our proofs from previous sections can help mitigate these drawbacks at the cost of larger proofs. However, unlike the other previous approaches such as [YAZ+19] that can provide an *exact* proof of plaintext knowledge, we believe our results are of practical relevance.

Let us first recall a Module-LWE encryption scheme similar to Kyber [BDK+18] for a message $\boldsymbol{m} \in \mathcal{R}_p$ for $p \in \mathbb{Z}^+$. The secret keys are sampled as $\vec{\boldsymbol{s}}_1, \vec{\boldsymbol{s}}_2 \xleftarrow{\$} S_1^\ell$, where S_1 is the set of polynomials in $\mathcal{R}_q$ with infinity norm at most 1, and the public keys are $\boldsymbol{A} \xleftarrow{\$} \mathcal{R}_q^{\ell \times \ell}$ and $\vec{\boldsymbol{t}} = \boldsymbol{A}\vec{\boldsymbol{s}}_1 + \vec{\boldsymbol{s}}_2$. An encryption $(\vec{\boldsymbol{v}}, \boldsymbol{w})$ of a plaintext $\boldsymbol{m} \in \mathcal{R}_p$ satisfies

$$\begin{pmatrix} \vec{\boldsymbol{v}} \\ \boldsymbol{w} \end{pmatrix} = \begin{pmatrix} p\boldsymbol{A}^\top & p\boldsymbol{I}_\ell & 0\ 0 \\ p\vec{\boldsymbol{t}}^\top & 0^{\ell \times \ell} & p\ 1 \end{pmatrix} \cdot \begin{pmatrix} \vec{\boldsymbol{r}} \\ \vec{\boldsymbol{e}} \\ \boldsymbol{e}' \\ \boldsymbol{m} \end{pmatrix} \mod q, \tag{13}$$

where $\vec{\boldsymbol{r}}, \vec{\boldsymbol{e}} \xleftarrow{\$} S_1^\ell$, $\boldsymbol{e}' \xleftarrow{\$} S_1$ and $\boldsymbol{I}_\ell$ is the $\ell \times \ell$ identity matrix over $\mathcal{R}$. The decryption in this case works by computing

$$\boldsymbol{m} = \boldsymbol{w} - \vec{\boldsymbol{s}}_1^\top \vec{\boldsymbol{v}} \mod q \mod p.$$

For a successful decryption, we require

$$q/2 > \left\| p(\vec{\boldsymbol{s}}_2^\top \vec{\boldsymbol{r}} + \boldsymbol{e}' - \vec{\boldsymbol{s}}_1^\top \vec{\boldsymbol{e}}) + \boldsymbol{m} \right\|_\infty . \tag{14}$$

For simplicity, we consider $\|\boldsymbol{m}\|_\infty = 1$, i.e., $p = 3$. It is easy to adjust also to the setting where $\boldsymbol{m}$ is a binary polynomial.

What we need now is to construct a non-interactive protocol that proves knowledge of $(\vec{\boldsymbol{r}}, \vec{\boldsymbol{e}}, \boldsymbol{e}', \boldsymbol{m})$ with $\|\vec{\boldsymbol{r}}\|_\infty = \|\vec{\boldsymbol{e}}\|_\infty = \|\boldsymbol{e}'\|_\infty = \|\boldsymbol{m}\|_\infty = 1$ that satisfies the relation in (13).

If we expand the matrix in the middle of the relation (13) to its representative matrix over $\mathbb{Z}_q$ and denote it by A, and denote the concatenated coefficient vector of $(\vec{\boldsymbol{r}}, \vec{\boldsymbol{e}}, \boldsymbol{e}', \boldsymbol{m})$ by $\vec{s}$, then we again end up with a relation suitable for the protocol in Figure 2. In this case $\vec{s} \in \mathbb{Z}_q^{(2\ell+2)d}$, i.e., $n = (2\ell+2)d$. As in the previous application, let us consider the setting of $q \approx 2^{32}$ (i.e., $\log q = 32$ and $k = 4$) and $d = 128$. From the previous section, we know that a module rank of $\ell = 10$ for the M-LWE encryption would be sufficient with $q \approx 2^{32}$ and $d = 128$. As a result, we get $n = 2816$, which is close to the value of $n = 2048$ in the previous section. The same module ranks of $\lambda = 10$ and $\kappa = 9$ suffice for the zero-knowledge proof with $\delta \approx 1.0045$ in this case as well.

For this parameter setting, correctness of decryption (i.e., the inequality in (14)) is easily satisfied. Plugging in this parameter setting into (12) (with the described optimizations), we end up with a proof length of 60.89 KB.

A.2 Other Applications

The two applications from Section 5 show how effective our new techniques are. There are actually various other applications, where our unstructured linear equation proof and our techniques can be useful. Some examples include group signatures [Cv91], ring signatures [RST01], *exact* range proofs, cryptographic accumulators and proof of message-signature pairs. These examples are all studied in [YAZ+19], where each of them build mainly on a zero-knowledge proof of a relation similar to that of our unstructured linear equation proof. Since this core proof can be realized more efficiently in practice using our novel techniques, we expect more efficient applications to follow.

Particularly, we believe that our zero-knowledge proofs and techniques can be useful in more efficient group signatures that do not rely on *relaxed* zero-knowledge proofs as in [dLS18, EZS+19]. Although these two works [dLS18, EZS+19] offer relatively efficient constructions in practice, they have certain drawbacks. More specifically, the opening algorithm in [dLS18] relies on the decryption algorithm of the verifiable encryption in [LN17], and therefore its worst-case running time for adversarially-generated group signatures is not clear. This case of opening adversarially-generated signatures is not at all supported in [EZS+19], and the group public key length in [EZS+19] grows linearly in the group size, rendering the scheme unsuitable for large groups. Therefore, extending of our techniques here to build a group signature seems to be an interesting future research direction.