Orthus: Practical Sublinear Batch-Verification of Lattice Relations from Standard Assumptions

Madalina Bolboceanu[1,2], Jonathan Bootle[1], Vadim Lyubashevsky[1], Antonio Merino-Gallardo[1,2] and Gregor Seiler[1]

[1] IBM Research Europe, Zurich, Switzerland
madalina.bolboceanu@ibm.com, jbt@zurich.ibm.com, vad@zurich.ibm.com, antonio@m-g.es, gseiler@posteo.net
[2] University of Potsdam, Potsdam, Germany

Abstract. The past several years have seen a rapid rise in practical lattice-based proof systems with linear-sized zero-knowledge proofs forming the foundation of many of the most efficient quantum-safe privacy protocols, and succinct proofs rapidly catching up and surpassing other quantum-safe alternatives in many metrics. A recent comparison of aggregate signatures (Ethereum Foundation, 2025) involving the hash-based aggregate signature scheme Plonky3 and the instantiation of aggregate signatures from Falcon from the LaZer lattice library (Lyubashevsky, Seiler, Steuer, CCS 2024) using LaBRADOR (Beullens, Seiler, Crypto 2023), showed that lattice-based constructions have an advantage in terms of proof size and prover time, but are around an order of magnitude slower with regards to verification time. In general, it appears that slower verification times are the main obstacle to the adoption of succinct lattice-based proof systems.
In this work, we introduce and implement Orthus, a proof system with sub-linear verification designed for relations that naturally arise in lattice-based constructions. Asymptotically, the verification time grows with the square root of the witness size, and for a concrete example of aggregating Falcon signatures our implementation reduces the verifier running time by a factor of 6X when aggregating 2^{16} signatures.

Keywords: Lattices, succinct proofs, succinct verification

Table of Contents

1 Introduction . . . 1
1.1 Techniques and Results. . . . 2
2 Preliminaries . . . 6
2.1 The Linear Proof . . . 6
2.2 LaBRADOR . . . 7
3 The Left-Right Protocol . . . 7
3.1 The Left-Right Relation . . . 8
3.2 The Left Part . . . 8
3.3 The Right Part . . . 10
3.4 Efficiency . . . 10
3.5 Security Analysis . . . 10
4 From the Left-Right Protocol to Orthus . . . 12
4.1 Proving the Same Constraint for Each Row Block . . . 12
4.2 Proving the Same Set of Constraints for Each Row Block . . . 16
4.3 Orthus . . . 18
5 Reducing the Communication Cost . . . 22
5.1 Two-Step Commitment . . . 22
5.2 Proving Knowledge of Orthus Output with LaBRADOR . . . 23
6 Application: Aggregating Falcon signatures . . . 23
7 Implementation . . . 24
7.1 Orthus Statements . . . 25
7.2 The Column NTT . . . 25
7.3 LaBRADOR . . . 26
References . . . 26
Appendix . . . 29
A Deferred preliminaries . . . 29
A.1 The Modular Johnson-Lindenstrauss Lemma . . . 29
A.2 Lattice Assumptions . . . 30
A.3 Commitment Schemes . . . 31
A.4 Interactive Arguments . . . 32
B Deferred Proofs . . . 32
B.1 Proof of Lemma 1 . . . 32
B.2 Proof of Lemma 2 . . . 33
B.3 Proof of Lemma 3 . . . 33
B.4 Proof of Theorem 6 . . . 34
C Details of (some) LaBRADOR Steps . . . 35
C.1 Project . . . 36
C.2 Aggregate with Integer Challenges . . . 36
C.3 Aggregate with Polynomial Challenges . . . 37
D Particular Constraints . . . 37
E Parameter Selection . . . 38
F More details on aggregating Falcon signatures . . . 40
G Figures for protocols . . . 43
H Proving Claim for Johnson-Lindenstrauss Lemma (Theorem 6) . . . 45
H.1 Some preliminary lemmas . . . 45
H.2 Proof of the Claim . . . 48

1 Introduction

Succinct proof systems are at the heart of many cryptographic primitives, ranging from digital currency applications, to aggregate signatures, to blockchain state verification. With the new quantum-safe NIST standards being published last year and quantum computing research progressing more rapidly, one needs to also begin preparing for the transition of advanced cryptographic primitives, including proof systems, to their quantum-safe variants.

There has been a lot of research on building practical hash-based proof systems, which are naturally quantum-safe. Descending from celebrated results in complexity theory such as the PCP-theorem, today, there are many hash-based proof systems with schemes such as Ligero+ [12], Blaze [15], BaseFold [39], STIR [6] and WHIR [7]. These schemes have very efficient verifiers, some on the order of milliseconds, reasonably-efficient provers, and outputs on the order of several hundred kilobytes to a few megabytes. These proof systems rely on collision-resistant hash functions for security, which is considered a minimal cryptographic assumption. On the other hand, many cryptographic assumptions, such as those based on lattices, offer rich algebraic structure, and exploiting this structure has led to improved concrete efficiency, particularly in terms of proof size.

Lattice-based succinct proof systems have a much shorter history. The first asymptotically-succinct proof system was proposed in [9], and the first practical scheme was only proposed a few years ago. The LaBRADOR proof system [11] was the first concretely practical lattice-based succinct proof system, and is currently the quantum-safe scheme with the smallest proof output size. It achieves proof sizes of around 60-70KB for systems with up to 2^{30} constraints and its prover is also quite efficient in practice. The main negative aspect of the scheme is that the verification time is linear in the size of the witness and it is concretely more than an order of magnitude slower than for other quantum-safe succinct proofs.

Several lines of research worked on improving the verification time of lattice-based schemes. The works of [25, 26] used a non-standard lattice-like assumption in which the random matrix used in (Ring)-SIS and (Ring)-LWE was replaced with a matrix consisting of tensors of small matrices, which allowed natural recursion which resulted in asymptotic verification logarithmic in the witness size. Other recent works in the LatticeFold line [13, 14] use a rather different technique for proving smallness of committed vectors by directly utilizing the sumcheck protocol used elsewhere in proof system constructions. The design results in an asymptotically very efficient verifier, but at this point the protocol works with norms that are not optimally compatible with proving lattice relations that underlie other lattice schemes. It is therefore not yet clear how concretely efficient it would be, both in prover and verifier time, when fed a large witness corresponding to natural lattice statements. The recent Greyhound scheme [36] achieves square root verification time for polynomial commitments. To assess its effectiveness for proving lattice relations, one would first need a compiler going from lattice statements to polynomial commitments, and it is unclear how concretely efficient this will be. In short, designing a proof system based on standard assumptions with a concretely efficient, sub-linear verifier was still an open problem.

Our Contribution. The sub-linear verification Orthus proof system designed in this work continues along the lines of the proof systems of [11,29] which take inputs that correspond directly to statements that one would want to prove in lattice protocols. This proof system is furthermore based on standard lattice assumptions (Module-SIS and Module-LWE) and is concretely efficient in practice.

Orthus is tailored to proving batched lattice statements that arise in many applications, such as aggregate signatures, verifiable secret sharing, blockchain consistency, etc. In particular, our proof system is designed to work on a set of N independent statement blocks labeled by ℓ for $1 \leq \ell \leq N$, each committing to polynomial

vectors $\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}$ and satisfying[3]

$$f(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}) \coloneqq \sum_{i,j \in [K]} a_{ij} \langle \tilde{\boldsymbol{s}}_i^{(\ell)}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \sum_{i \in [K]} \langle \boldsymbol{\varphi}_i, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle - b = 0, \tag{1}$$

$$\|\tilde{\boldsymbol{s}}_1^{(\ell)}\| \leq \beta_1, \ldots, \|\tilde{\boldsymbol{s}}_K^{(\ell)}\| \leq \beta_K. \tag{2}$$

If the total witness size is N, then the asymptotic running time of the prover is $\tilde{O}(N)$ and the verification time is roughly $\tilde{O}(K^2\sqrt{N})$. Thus in the typical scenarios where the number of polynomial equations in each statement block is a small constant, the verification run-time is square root in the witness length.

At the heart of the protocol is a simple technique, which we call the *Left-Right* proof, which asymptotically reduces the input by a factor of square root in the witness size, and whose verifier also runs in square root time. We then transform the shrunken output of this procedure into a form that can be efficiently fed into LaBRADOR, which then outputs a short proof of knowledge of the full statement.

Thus one can view our proof system as a generalization of LaBRADOR, which (when combined together with its Dachshund front-end from [33]) allows to prove the above set of equations, but would have a (quasi)-linear time verifier.

1.1 Techniques and Results.

We believe that the most natural way to explain our result and technique is by first understanding the reason why LaBRADOR did not have efficient verification. One can then think of our Left-Right proof as "fixing" this problem in LaBRADOR. So before explaining our result, we will briefly sketch the relevant part of LaBRADOR where the main roadblock appears.

The LaBRADOR Proof System. Suppose that we would like to do something as simple as possible – commit to a long vector $\boldsymbol{s}$ and prove that $\langle \boldsymbol{s}, \boldsymbol{s} \rangle = v$. What one does is break up the long $\boldsymbol{s}$ into a few parts $\boldsymbol{s}_i$ and create compressing Ajtai commitments [2] to them as

$$\boldsymbol{B}\boldsymbol{s}_i = \boldsymbol{t}_i \tag{3}$$

and send them to the verifier (the prover will actually send them in committed form, but this is not important here). The verifier creates random low-norm challenge polynomials c_i, sends them to the prover, and the prover *amortizes* the witnesses as

$$\boldsymbol{z} = \sum_i \boldsymbol{s}_i c_i. \tag{4}$$

The verifier checks that $\|\boldsymbol{z}\|$ is short and that $\boldsymbol{B}\boldsymbol{z} = \sum_i \boldsymbol{t}_i c_i$, which proves that the openings to the $\boldsymbol{t}_i$ are unique and (with some additional work) that $\boldsymbol{z}$ is of the form in (4). The verifier is also able to compute the useful equation

$$\langle \boldsymbol{z}, \boldsymbol{z} \rangle = \sum_{i,j} \langle \boldsymbol{s}_i, \boldsymbol{s}_j \rangle c_i c_j. \tag{5}$$

If, at the same time as sending the $\boldsymbol{t}_i$ in (3), the prover also sends all the $v_{i,j} = \langle \boldsymbol{s}_i, \boldsymbol{s}_j \rangle$ before seeing the challenges, then by the Schwartz-Zippel lemma, it implies that the $v_{i,j}$ really are $\langle \boldsymbol{s}_i, \boldsymbol{s}_j \rangle$, and so one can prove any statement about their linear combinations using standard lattice proofs for linear relations. Just like for the vectors $\boldsymbol{t}_i$, the prover sends commitments to these polynomials. This procedure is then repeated

[3] More generally, the system can also prove knowledge of the constant coefficient of the first statement and also prove the norm bounds in the second statement with a small gap, which makes the proofs more concretely efficient (though the same in an asymptotic sense) in those scenarios.

recursively by also committing to the $\boldsymbol{z}$ and proving $\langle \boldsymbol{z}, \boldsymbol{z} \rangle = \sum_{i,j} v_{i,j} c_i c_j$. Even though we can prove an arbitrary statement in the linear combination of $v_{i,j}$, very often we do not need to do this. In the statement that we are trying to prove, we only care about the $v_{i,i}$, of which there are only linearly-many (where linear refers to the number of chunks into which we broke up $\boldsymbol{s}$), and so the rest of the quadratic number of $v_{i,j}$ that we need to commit to are just auxiliary "garbage" polynomials.

To see the main inefficiency in LaBRADOR, we look at (3), which commits to "chunks" of $\boldsymbol{s}$. If the number of $\boldsymbol{s}_i$ is small, then each $\boldsymbol{s}_i$ is going to be long and we need the matrix $\boldsymbol{B}$ to also be long.[4] On the other hand, if one breaks $\boldsymbol{s}$ into many chunks, then the number of terms $v_{i,j}$ grows quadratically in the number of chunks, becoming the cause for inefficiency. The authors of [11] set the parameters to optimize the "chunk-splitting" in order to achieve optimal concrete proof sizes, but in practice, this may not be optimal for the prover or verifier time due to the fact that the witness size grows quadratically in the number of chunks. Furthermore, the verification time is (at least) linear in the witness size, and in practice is only about 2X faster than the prover's, whereas in hash-based proof systems, the verifier time can be more than 50X faster (c.f. [34]). The main property of our Left-Right proof system is that it allows the witness size to grow linearly in the number of chunks, rather than quadratically, which allows us to break $\boldsymbol{s}$ up into a larger (i.e. square root in the dimension of $\boldsymbol{s}$) number of parts.

Trying to reduce the number of garbage polynomials. For soundness of LaBRADOR, it was crucial that the $\boldsymbol{z}$ in (4) had small norm, which in turn required that the c_i did too. By having distinct, small challenges c_i, we then ended up with a quadratic number of $v_{i,j}$ when computing $\langle \boldsymbol{z}, \boldsymbol{z} \rangle$. If, for a second, we ignore the need for the c_i to be small, we could think of doing the following: instead of sending many challenges c_i, the verifier just sends one challenge c and the prover outputs

$$\boldsymbol{z} = \sum_{i=1}^{k} \boldsymbol{s}_i c^i, \quad \boldsymbol{z}' = \sum_{i=1}^{k} \boldsymbol{s}_i c^{-i}. \tag{6}$$

Then

$$\langle \boldsymbol{z}, \boldsymbol{z}' \rangle = \sum_{i=1}^{k} \langle \boldsymbol{s}_i, \boldsymbol{s}_i \rangle + \sum_{-k<i<k, i \neq 0} v_i c^i, \tag{7}$$

for some v_i which are combinations of the $\boldsymbol{s}_i$. If we only care about proving $\sum \langle \boldsymbol{s}_i, \boldsymbol{s}_i \rangle$ as before, then we only need to commit to $2(k-1)$ auxiliary terms v_i before the protocol starts (instead of $O(k^2)$ as in LaBRADOR) to use the Schwartz-Zippel lemma. The problem is that this proof is not sound because the polynomials c^i have large coefficients and thus the verifier is not convinced that $\boldsymbol{z}$ and $\boldsymbol{z}'$ are actually constructed as in (6). The way to obtain soundness is to do a proof on "both sides" of the $\boldsymbol{s}_i$, hence the name "Left-Right" proof.

The Left-Right Proof. Let us write $\boldsymbol{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k]$ where the $\boldsymbol{s}_i$ are the columns of $\boldsymbol{S}$ and $\tilde{\boldsymbol{s}}_i$ are the rows, and suppose that we would like to commit to $\boldsymbol{S}$ and prove knowledge that $\sum \langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_i \rangle = v$.[5] We will later show how to prove more general relations, but let us stick with this simple one for now. For a polynomial $\theta \in \mathcal{R}_q$, we define the vector $\boldsymbol{\theta}_+ \in \mathcal{R}_q^k$ to be the row vector $[\theta, \theta^2, \theta^3, \ldots, \theta^k]$ and $\boldsymbol{\theta}_-$ to be $[\theta^{-1}, \theta^{-2}, \theta^{-3}, \ldots, \theta^{-k}]$.

The protocol begins with the prover sending $\boldsymbol{t}_i$ as in (3), thus $\boldsymbol{BS} = \boldsymbol{T}$, and also some auxiliary "garbage" polynomials v_i, which will be defined in (9). The verifier then creates a random (non-zero) challenge $\theta \in \mathcal{R}_q$, and the prover computes

$$\tilde{\boldsymbol{z}}_+ = \boldsymbol{\theta}_+ \cdot \boldsymbol{S}, \quad \tilde{\boldsymbol{z}}_- = \boldsymbol{\theta}_- \cdot \boldsymbol{S}. \tag{8}$$

In the outline of the protocol (c.f. Figure 1), we call this amortization over the rows.

[4] This is where having $\boldsymbol{B}$ being a tensor of small matrices, as in [25], overcomes the need for $\boldsymbol{B}$ to be long.

[5] Instead of proving statements over the columns, we prove them over the rows, but this is just notation.

Observe that the inner product of the $\tilde{\boldsymbol{z}}_\pm$ is

$$\langle \tilde{\boldsymbol{z}}_+, \tilde{\boldsymbol{z}}_- \rangle = \sum_{i=1}^{k} \langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_i \rangle + \sum_{-k<i<k, i\neq 0} v_i \theta^i, \tag{9}$$

where v_i's are some linear combinations of the inner-products $\langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_j \rangle$ that we committed to earlier. Thus if we knew that $\tilde{\boldsymbol{z}}_+ = \boldsymbol{\theta}_+ \cdot \boldsymbol{S}$ and $\tilde{\boldsymbol{z}}_- = \boldsymbol{\theta}_- \cdot \boldsymbol{S}$, then by the Schwartz-Zippel lemma, we would know that the polynomial[6] equality

$$\left\langle \sum_{i=1}^{k} \tilde{\boldsymbol{s}}_i Y^i, \sum_{i=1}^{k} \tilde{\boldsymbol{s}}_i Y^{-i} \right\rangle = v + \sum_{-k<i<k, i\neq 0} v_i Y^i \tag{10}$$

holds (since checking (9) checks the above equality evaluated at a random θ) with high probability, and therefore $\sum \langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_i \rangle = v$. In other words, proving knowledge of $\boldsymbol{S}$ satisfying

$$\begin{pmatrix} \boldsymbol{B} \\ \boldsymbol{\theta}_+ \\ \boldsymbol{\theta}_- \end{pmatrix} \cdot \boldsymbol{S} = \begin{pmatrix} \boldsymbol{T} \\ \tilde{\boldsymbol{z}}_+ \\ \tilde{\boldsymbol{z}}_- \end{pmatrix} \tag{11}$$

would conclude the proof.

To prove this statement, the verifier sends a challenge vector $\boldsymbol{c}$ consisting of small polynomials, and the prover simply outputs $\boldsymbol{z} = \boldsymbol{S}\boldsymbol{c}$. The verifier checks that $\|\boldsymbol{z}\|$ is small and that

$$\begin{pmatrix} \boldsymbol{B} \\ \boldsymbol{\theta}_+ \\ \boldsymbol{\theta}_- \end{pmatrix} \cdot \boldsymbol{z} = \begin{pmatrix} \boldsymbol{T} \\ \tilde{\boldsymbol{z}}_+ \\ \tilde{\boldsymbol{z}}_- \end{pmatrix} \cdot \boldsymbol{c}. \tag{12}$$

The above is just a classical linear proof system for committed values (e.g. [10]) and is in fact exactly what is used by LaBRADOR to prove (4), where instead of the $\boldsymbol{B}$ and $\boldsymbol{T} = [\boldsymbol{t}_1, \ldots, \boldsymbol{t}_k]$ in (4), we have the matrices with more rows from (11). We call this part of the overall proof (Figure Figure 1) amortization of the columns. This completes the high-level description of the Left-Right proof technique.

Observe that if $\boldsymbol{S} \in \mathcal{R}_q^{k\times k}$, then it contains k^2 polynomials, but everything which the verifier handles – $\boldsymbol{T}, v_i, \boldsymbol{z}, \tilde{\boldsymbol{z}}_\pm, \boldsymbol{\theta}_\pm$, are all $O(k)$ in size. The running time of the verifier (as well as the proof output) is thus asymptotically square root in the size of the instance.

From the Left-Right Proof to Orthus. Above, we described how to compute the simple relation $\sum_i \langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_i \rangle$, but we would like to give proofs for general relations as in (1), (2). This is performed via a series of protocols in Sections 3, 4.1, 4.2, 4.3.

In Section 3, we show how one can, instead of proving one quadratic relation over the entire secret, prove relations of the form $\langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_i \rangle = u_i$, and a more generalized version where the previous are also multiplied by some public values.

In Section 4.1, we begin to consider "row blocks". That is, instead of only allowing inner products of the form $\langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_i \rangle$, we also allow proving a constraint involving inner products of rows in the same block. For example, if we have rows $\tilde{\boldsymbol{s}}_1, \ldots, \tilde{\boldsymbol{s}}_6$, and we break them up into two row blocks, then we can prove that, say, $\langle \tilde{\boldsymbol{s}}_1, \tilde{\boldsymbol{s}}_2 + 2\tilde{\boldsymbol{s}}_3 \rangle = u_1$ and $\langle \tilde{\boldsymbol{s}}_4, \tilde{\boldsymbol{s}}_5 + 2\tilde{\boldsymbol{s}}_6 \rangle = u_2$. The reduction from this proof to the Left-Right protocol is given in Section 4.1. If each row-block consists of K rows, then one may need to create up to $O(K^2)$ garbage terms for the same reason as we needed a quadratic number of garbage terms in LaBRADOR (c.f. the discussion in Section 1.1). In Section 4.2, we consider the more general version where one wishes to prove several constraints per row block, and then finally in Section 4.3, we show how one can put many constraints of the form as in (1), (2) into one row block, which is the final step needed to create a protocol with an $O(K^2\sqrt{N})$ time verifier.

[6] Laurent polynomial would be more precise, since it has negative powers. But the Schwartz-Zippel lemma still applies if we just multiply the Laurent polynomial by Y^k to make all the powers positive.

In Section 5, we describe some practical optimizations and explain how the output of the Orthus proof system can be fed into LaBRADOR. The latter part is particularly important for concrete efficiency because, even though the input to LaBRADOR is smaller than the original input into Orthus, if the verification of Orthus requires a lot of inner-product between various parts, then the number of garbage terms produced by LaBRADOR, which can grow quadratically (see Section 1.1) can negate all the efficiency gains. In Section 7, we finally describe our full implementation.

A Polynomial Commitment Perspective on our Results. One way to view our results is as implicitly providing a compiler from lattice relations to batched polynomial evaluation statements related to those used in polynomial commitment schemes. For example, the left-right proof provides a way to demonstrate that the batched polynomial evaluations described in Equation (8) hold, while Orthus efficiently reduces lattice statements to this form.

Concrete Results. The way one would use Orthus in practice is to pass its shrunken output to a proof system that is not necessarily fast when it comes to verification, but produces small outputs, like LaBRADOR. We implemented and tested such a construction by using it to create an aggregate signature scheme by aggregating Falcon signatures [37]. The results we obtain are described in Table 1, which compares the aggregate signature implemented using LaBRADOR in the LaZer library [33] with our current version combining Orthus and LaBRADOR.

The verification times are between 2.3X and 6X faster with this combination than with just LaBRADOR alone, with the improvement naturally getting larger as the instance size increases. [7]

The proof size we obtain for all the sizes of aggregate signatures from Table 1 is around $110 - 115$KB, compared to the $\sim$ 75KB from LaZer [33], which can be attributed to the use of higher-degree rings in our LaBRADOR implementation, but this is not inherent to the protocol. [8] We leave the reduction of the proof size for future work.

Table 1. Performance in seconds of Orthus on aggregating Falcon signatures and comparing to the LaBRADOR implementation in LaZer. The numbers in parentheses for the Orthus implementation are the times needed for the "Orthus part", which shrinks the input down to square root size before feeding it into LaBRADOR. The remaining time is the running time of LaBRADOR on the shrunken input. All timings were obtained on a single core of an Intel Core i7-11850H at about 4.5GHz.

Number of Signatures	LaZer Prover Time	LaZer Verifier Time	Orthus Prover Time	Orthus Verifier Time
2^{13}	5.56	3.2	5.1 (2.4)	1.4 (.2)
2^{14}	9.8	5.6	9.1 (5.2)	2 (.3)
2^{15}	21.7	12.8	17.5 (11.1)	3 (.4)
2^{16}	46.7	27.1	33.1 (23)	4.4 (.7)

[7] An earlier version of this paper also reported results for 2^{17} signatures, achieving a 9X speedup in verification time compared with LaBRADOR alone. These results are no longer valid due to the increased slack derived by the revised Johnson-Lindenstrauss lemma. In particular, the 38-bit modulus used in our implementation is insufficient to support the 2^{17}-signature setting under the revised analysis. Nevertheless, we expect that an implementation of the proof system using a slightly larger modulus would achieve performance comparable to that reported in the earlier version.

[8] An earlier version of this paper reported 100KB of proof size. The increase has to do with the revised output size of the outer commitments (see Section 5.1) affecting both Orthus and the LaBRADOR with outer commitments that we run after it.

2 Preliminaries

Notation. Let q be an odd prime. We denote by $\mathbb{Z}_q$ the ring of integers mod q. We write $[n] = \{1, 2, \ldots, n\}$. We say $x \leftarrow S$ when $x \in S$ is sampled uniformly at random from a finite set S and similarly $x \leftarrow \chi$ when x is sampled according to a distribution χ. We denote vectors $\mathbf{v} \in \mathbb{Z}^n$ and matrices $\mathbf{A} \in \mathbb{Z}^{r \times m}$ with bold letters. Let d be a power of two, and let $\mathcal{R}, \mathcal{R}_q$ be the rings $\mathbb{Z}[X]/(X^d + 1)$ and $\mathbb{Z}_q[X]/(X^d + 1)$ respectively, where q, d are such that $X^d + 1$ splits in two irreducible factors mod q. For a polynomial $f = a_0 + a_1 X + \ldots + a_{d-1} X^{d-1} \in \mathcal{R}$, we denote by $\mathtt{ct}(f)$ its constant-term, i.e. $\mathtt{ct}(f) = a_0$. We denote the vectors $\boldsymbol{v} \in \mathcal{R}^n$ by italic-bold lower-case and matrices $\boldsymbol{A} \in \mathcal{R}^{r \times m}$ by italic-bold capital letters. We denote by $\mathbf{1}_r$ the column vector containing all entries equal to (the constant polynomial) 1, and by $\mathbf{0}_r$ the one containing all entries equal to (the constant polynomial) 0. If $\boldsymbol{a} \in \mathcal{R}_q^{n_a}$ and $\boldsymbol{b} \in \mathcal{R}_q^{n_b}$ are vectors, we denote by $(\boldsymbol{a}, \boldsymbol{b}) \in \mathcal{R}_q^{n_a + n_b}$ the vector obtained by concatenating them. A function $f : \mathbb{N} \to \mathbb{R}$ is said to be negligible if for all $c \in \mathbb{N}$ there exists some $n_c \in \mathbb{N}$ such that for all $n > n_c$, $|f(n)| < \frac{1}{n^c}$. In the context of probabilities, we say $a \approx b$ if $a - b$ is negligible. The rest of the preliminaries can be found in Appendix A.

2.1 The Linear Proof

This is an interactive argument of knowledge for Ajtai commitment openings that satisfy a linear relation, similar to the one described in [9].

Our witness vectors $(\boldsymbol{s}_i)_{i \in [m]}, \boldsymbol{s}_i \in \mathcal{R}^r$ form a witness matrix $\boldsymbol{S} \in \mathcal{R}^{r \times m}$, and each vector has bounded Euclidean norm $\|\boldsymbol{s}_i\| \leq B_i$. A commitment of the witness matrix is given, $\boldsymbol{AS} = \boldsymbol{T}$, for $\boldsymbol{A} \in \mathcal{R}_q^{t \times r}$, and the witness satisfies a certain linear relation, $\boldsymbol{BS} = \boldsymbol{U}$, for $\boldsymbol{B} \in \mathcal{R}_q^{t' \times r}$. Overall, we describe an argument for

$$R_{\mathtt{lin}} = \left\{ ((\boldsymbol{T}, (B_i)_i, \boldsymbol{B}, \boldsymbol{U}), \boldsymbol{S}) \;\middle|\; \begin{matrix} \|\boldsymbol{s}_i\| \leq B_i, \forall i \in [m] \\ \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{B} \end{pmatrix} \cdot \boldsymbol{S} = \begin{pmatrix} \boldsymbol{T} \\ \boldsymbol{U} \end{pmatrix} \end{matrix} \right\}.$$

The protocol for proving the linear relation goes as follows. Upon receiving random challenges $\boldsymbol{c}_i \in \mathcal{C} \subseteq \mathcal{R}_q$ from the verifier, the prover computes and sends $\boldsymbol{z} = \boldsymbol{Sc}$, called an amortization of the columns. Then the verifier simply checks whether $\|\boldsymbol{z}\| \leq B$ and

$$\begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{B} \end{pmatrix} \cdot \boldsymbol{z} = \begin{pmatrix} \boldsymbol{T} \\ \boldsymbol{U} \end{pmatrix} \cdot \boldsymbol{c}. \tag{13}$$

A summary of the protocol can be found in Figure 2. The challenge space $\mathcal{C} \subseteq \mathcal{R}_q$ is chosen such that $c - c'$ is invertible for any pair of distinct $c, c' \in \mathcal{C}$ and such that $\|c\| \leq \tau$ and $\|c\|_{\mathrm{op}} \leq T$, for any $c \in \mathcal{C}$, for some constants $\tau, T \in \mathbb{R}$, where $\|c\|_{\mathrm{op}} \coloneqq \sup_{r \in \mathcal{R} \setminus \{0\}} \frac{\|cr\|}{\|r\|}$. A popular choice [11] is to sample c with small $\|c\| = \tau$ by taking a fixed number of short non-zero entries (e.g. $\pm 1, \pm 2$) that ensures a large-enough size of $\mathcal{C}$, while the invertibility is guaranteed by their shortness as proved in [32, Cor.1.2]. The bound B depends on the bounds $(B_i)_i$ and $\mathcal{C}$. It could be set as a worst-case bound derived from T to ensure perfect completeness. However, if one is willing to accept a small completeness error, then we can use probabilistic arguments to choose a smaller B and improve the proof size. For this, we follow the analysis of [11, 35, 36].

Lemma 1. *Let $\boldsymbol{S} \in \mathcal{R}_q^{r \times m}$ be a matrix with bounded columns $\|\boldsymbol{s}_j\| \leq B_j$. Let $\mathcal{C} \subseteq \mathcal{R}_q$ be a challenge space of ring elements with ℓ_2 norm equal to τ. Then, the $\mathbb{Z}_q$ coefficients of $\boldsymbol{z} = \boldsymbol{Sc} \in \mathcal{R}_q^r$, for $\boldsymbol{c} \in \mathcal{C}^m$ follow a Gaussian distribution of standard deviation $\sigma_{\boldsymbol{z}} \coloneqq \tau \sqrt{\sum_{j \in [m]} B_j^2 / rd}$ and so, for any $t > 0$, we have that $\|\boldsymbol{z}\| \leq t \sigma_{\boldsymbol{z}} \sqrt{rd}$, except with probability $2rd \exp(-\frac{t^2}{2})$.*

We defer the proof to Appendix B.

Security Analysis. Denote by $\bar{\mathcal{C}} \triangleq \mathcal{C} - \mathcal{C}$. The linear proof gives an argument of knowledge for a relaxed version of $R_{\mathtt{lin}}$ that we call $R'_{\mathtt{lin}}$ where the norm constraints are not proven exactly; instead we can guarantee that $\|\bar{\boldsymbol{c}}_i \boldsymbol{s}_i\| \leq 2B$ for random $\bar{\boldsymbol{c}} \in \bar{\mathcal{C}}^m$.

The proofs of the lemmas follow [9] and are deferred to Appendix B.

Lemma 2. *The protocol in Section 2.1 is complete with completeness error* $2rd\exp(-\frac{t^2}{2})$*, when setting* $B = t\sigma_{\boldsymbol{z}}\sqrt{rd}$*.*

The value of t can be chosen based on r and d and the desired completeness error.

Lemma 3. *The protocol in Section 2.1 is knowledge sound for* R'_{lin}*, defined as*

$$R'_{lin} = \left\{ ((\boldsymbol{T}, B, \boldsymbol{B}, \boldsymbol{U}), (\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})) \middle| \begin{matrix} \|\bar{\boldsymbol{c}}_i \bar{\boldsymbol{s}}_i\| \leq 2B, \forall i \in [m] \\ \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{B} \end{pmatrix} \cdot \bar{\boldsymbol{S}} = \begin{pmatrix} \boldsymbol{T} \\ \boldsymbol{U} \end{pmatrix} \end{matrix} \right\}$$

with knowledge error $\kappa_{lin} \coloneqq \frac{1}{|\mathcal{C}|}$*.*

2.2 LaBRADOR

The LaBRADOR proof system [11] is an iterative public coin proof system with concretely small communication size, whose security relies on the hardness of the Module Short Integer Solution problem. It is constructed from a base protocol that receives as input an instance $\mathbb{x}$ and proves the knowledge of a witness $\mathbb{w}$ such that $(\mathbb{x}, \mathbb{w})$ belongs to a particular relation $R_{\mathtt{L}}$. The design of the protocol is such that the last prover message dominates the communication size, while being smaller than the witness $\mathbb{w}$. Rather than sending it, this last message actually becomes the witness $\mathbb{w}'$ for another iteration of the protocol where the instance $\mathbb{x}'$ corresponds to the verifier's checks from the base protocol. More precisely, proving knowledge of a witness $\mathbb{w}$ such that $(\mathbb{x}, \mathbb{w})$ belongs to $R_{\mathtt{L}}$ is reduced to proving knowledge of a witness $\mathbb{w}'$ such that $(\mathbb{x}', \mathbb{w}')$ belongs to the same relation $R_{\mathtt{L}}$. This process continues iteratively, reducing the size of the witness with each iteration, until no more progress is made and the prover finally sends the last witness for the verifier to check directly.

We recall here the relation $R_{\mathtt{L}}$, parametrized by the rank $n \geq 1$, multiplicity $r \geq 1$ and norm bound $\beta > 0$. The statement $\mathbb{x}$ involves collections of quadratic dot product constraints $f : \mathcal{R}_q^n \times \ldots \times \mathcal{R}_q^n \to \mathcal{R}_q$ of the form

$$f(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_r) \triangleq \sum_{i,j \in [r]} a_{ij} \langle \boldsymbol{s}_i, \boldsymbol{s}_j \rangle + \sum_{i \in [r]} \langle \boldsymbol{\varphi}_i, \boldsymbol{s}_i \rangle - b, \tag{14}$$

where $a_{ij}, b \in \mathcal{R}_q$, $\boldsymbol{\varphi}_i \in \mathcal{R}_q^n$, such that $a_{ij} = a_{ji}$. For some constraints f', we are only interested in the constant term $\mathtt{ct}(\cdot)$ of their output, so the other coefficients of b are not included in the statement. We collect these *constant term* constraints in a family $\mathcal{F}'$, separated from the *full* constraints, that we collect in a family $\mathcal{F}$. A tuple $\mathbb{w} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_r) \in (\mathcal{R}^n)^r$ is a valid witness for the statement $\mathbb{x} = (\mathcal{F}, \mathcal{F}', \beta)$ if and only if

$$(f(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_r) = 0, \forall f \in \mathcal{F}) \wedge (\mathtt{ct}(f'(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_r)) = 0, \forall f' \in \mathcal{F}') \wedge (\sum_{i \in [r]} \|\boldsymbol{s}_i\|^2 \leq \beta^2).$$

We recall the inner and outer commitments from LaBRADOR used in [11,36], which are Ajtai commitments. Let $N, n, r, \mathtt{b} \in \mathbb{N}$ and $\mathtt{t} \coloneqq \lceil \log_{\mathtt{b}} q \rceil$. Let $(\boldsymbol{A}, \boldsymbol{B}) \in \mathcal{R}_q^{N \times n} \times \mathcal{R}_q^{N \times trN}$ be public parameters. Then the *inner* commitments are the r vectors $\boldsymbol{t}_i \coloneqq \boldsymbol{A}\boldsymbol{s}_i \in \mathcal{R}_q^N$, whereas the *outer* commitment is $\boldsymbol{u} \coloneqq \boldsymbol{B}\hat{\boldsymbol{t}}$, where $\hat{\boldsymbol{t}} = (\hat{\boldsymbol{t}}_i)_{i \in [r]}$, $\hat{\boldsymbol{t}}_i \coloneqq \boldsymbol{G}_{\mathtt{b},N}^{-1}(\boldsymbol{t}_i)$.

3 The Left-Right Protocol

In this section we present the Left-Right Protocol. This protocol acts as the core subroutine for our main protocol in Section 4. Given Ajtai commitments to the *columns* $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m$ of a witness matrix $\boldsymbol{S} \in \mathcal{R}^{r \times m}$, the Left-Right protocol proves n quadratic relations between the *rows* $\tilde{\boldsymbol{s}}_1, \ldots, \tilde{\boldsymbol{s}}_r$ of $\boldsymbol{S}$. The communication complexity and verification runtime of the Left-Right Protocol are $O(n + m + r)$, which is sublinear in the size of $\boldsymbol{S}$.

The protocol consists of two parts, which we call the Left and the Right parts, as they correspond to left and right multiplications of the witness matrix $\boldsymbol{S}$. The Left part takes a structured linear combination of the rows $\tilde{\boldsymbol{s}}_i$ of the witness matrix $\boldsymbol{S}$ using consecutive powers of a random challenge θ, corresponding to a left multiplication of $\boldsymbol{S}$. These combinations are useful for proving the given quadratic relations. The Right part is simply the classic linear proof of knowledge from Section 2.1, which takes random linear combinations of the columns, corresponding to a right multiplication of $\boldsymbol{S}$. Consistency checks between the left and right multiplications prove that the structured linear combinations of rows are correctly computed.

The linear combinations used in the Left-Right protocol come from two different challenge spaces $\hat{\mathcal{C}} \subseteq \mathcal{R}_q$ and $\mathcal{C} \subseteq \mathcal{R}_q$, one for the Left and one for the Right part. The challenge θ which generates linear combinations for the left part is sampled from $\hat{\mathcal{C}}$. There are no restrictions on $\hat{\mathcal{C}}$, and in fact one can take $\hat{\mathcal{C}} = \mathcal{R}_q$, the entire ring. We use $\hat{\mathcal{C}} = \mathcal{R}_q$ for simplicity here, although using a challenge space made up of lower-norm elements provides some implementation speed-ups. The linear combinations for the Right part are sampled from $\mathcal{C}$, and defined as in Section 2.1.

3.1 The Left-Right Relation

We now give a detailed description of the relation proved by the Left-Right protocol. Let $\boldsymbol{T} \coloneqq \boldsymbol{A}\boldsymbol{S} \in \mathcal{R}_q^{t\times m}$ be the Ajtai commitment to the columns of $\boldsymbol{S} \in \mathcal{R}^{r\times m}$. Each column $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m$ satisfies a norm constraint $\|\boldsymbol{s}_i\| \leq B_i$. The relation we want to prove involves n linear constraints characterized by n row vectors $\tilde{\boldsymbol{r}}_i \in \mathcal{R}_q^r$, collected into a matrix $\boldsymbol{R} \in \mathcal{R}_q^{n\times r}$, n quadratic constraints characterized by a bilinear constraint $\boldsymbol{D} \in \mathcal{R}_q^{m\times m}$, along with the vectors $\tilde{\boldsymbol{r}}_i$ and n pairs of targets $(u_i, w_i) \in \mathcal{R}_q^2$. Later on in Sections 4.1 to 4.3 the matrices $\boldsymbol{R}$ and $\boldsymbol{D}$ will be instantiated with different choices, when proving correct computation of a wide variety of different auxiliary terms and functions of the witness $\boldsymbol{S}$, while giving proofs relative to the same commitment $\boldsymbol{T}$ to $\boldsymbol{S}$.

Concretely, the prover wants to prove knowledge of $\boldsymbol{S}$ satisfying the following constraints:

$$\langle \tilde{\boldsymbol{r}}_i \boldsymbol{S}\boldsymbol{D}, \tilde{\boldsymbol{r}}_i \boldsymbol{S}\rangle = u_i,\ \forall i \in [n], \tag{15}$$

$$\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_i \boldsymbol{S}\rangle = w_i,\ \forall i \in [n]. \tag{16}$$

Equation (15) encodes quadratic constraints, while Equation (16) encodes linear constraints, which we summarize in the following relation:

$$R_{\mathtt{LR}} = \left\{ ((\boldsymbol{T}, (B_i)_i, \boldsymbol{R}, \boldsymbol{D}, \boldsymbol{\varphi}, (u_i, w_i)_i), \boldsymbol{S}) \middle| \begin{array}{c} \boldsymbol{A}\boldsymbol{S} = \boldsymbol{T},\ \|\boldsymbol{s}_i\| \leq B_i,\ i \in [m] \\ \langle \tilde{\boldsymbol{r}}_i \boldsymbol{S}\boldsymbol{D}, \tilde{\boldsymbol{r}}_i \boldsymbol{S}\rangle = u_i,\ i \in [n] \\ \langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_i \boldsymbol{S}\rangle = w_i,\ i \in [n] \end{array} \right\}.$$

We now describe the two parts of the Left-Right protocol. A full algorithmic description of the protocol can be found in Figure 3.

3.2 The Left Part

The Left Part of the protocol begins with the verifier sending random challenges $\beta_1, \ldots, \beta_n \leftarrow \mathcal{R}_q$ to the prover. These are used to take a linear combination of Equation (15) over $i \in [n]$, reducing n quadratic equations to a single one:

$$\sum_{i\in[n]} \beta_i \langle \tilde{\boldsymbol{r}}_i \boldsymbol{S}\boldsymbol{D}, \tilde{\boldsymbol{r}}_i \boldsymbol{S}\rangle = \sum_{i\in[n]} \beta_i u_i. \tag{17}$$

In this part, the verifier will check Equation (17) using a strategy employed in many works including [23], and later in lattice-based protocols such as [17, 25]. Namely, the verifier will use two structured linear combinations of the rows of $\boldsymbol{R}\boldsymbol{S}$ and of $\boldsymbol{R}\boldsymbol{S}\boldsymbol{D}$, embedding the rows into polynomials $\sum_{i\in[n]} \beta_i \tilde{\boldsymbol{r}}_i \boldsymbol{S}\boldsymbol{D} Y^i$, $\sum_{i\in[n]} \tilde{\boldsymbol{r}}_i \boldsymbol{S} Y^{-i} \in \mathcal{R}_q^m[Y, Y^{-1}]$.

Note that if Equation (17) is indeed satisfied, then the constant coefficient of Y in $\left\langle \sum_{i\in[n]} \beta_i \tilde{\boldsymbol{r}}_i \boldsymbol{S}\boldsymbol{D}Y^i, \sum_{i\in[n]} \tilde{\boldsymbol{r}}_i \boldsymbol{S}Y^{-i} \right\rangle$ is in fact equal to $\sum_{i\in[n]} \beta_i u_i$. More precisely, we have

$$\left\langle \sum_{i\in[n]} \beta_i \tilde{\boldsymbol{r}}_i \boldsymbol{S}\boldsymbol{D}Y^i, \sum_{i\in[n]} \tilde{\boldsymbol{r}}_i \boldsymbol{S}Y^{-i} \right\rangle = \sum_{i\in[n]} \beta_i u_i + \sum_{-n<i<n, i\neq 0} v_i Y^i. \tag{18}$$

To check Equation (17), the verifier must check that Equation (18) holds. To achieve this, the prover sends all of the other coefficients $v_i \in \mathcal{R}_q$ to the verifier computing them as

$$v_i = \sum_{j,k\in[n], j-k=i} \beta_j \langle \tilde{\boldsymbol{r}}_j \boldsymbol{S}\boldsymbol{D}, \tilde{\boldsymbol{r}}_k \boldsymbol{S} \rangle, \text{ for } -n < i < n,\ i \neq 0. \tag{19}$$

We call these coefficients v_i "garbage" coefficients, because they are artefacts created to help the verifier check Equation (18), but not directly relevant to our original aim of checking Equation (17).

Now, we have the verifier check Equation (18) by evaluating it at a random challenge θ from $\hat{\mathcal{C}}$. After the prover has sent the garbage terms v_i from Equation (17) to the verifier, the verifier sends θ to the prover. To avoid sending $\boldsymbol{S}$ to the verifier directly, as this would incur high communication costs, the prover helps the verifier to evaluate Equation (18) by computing and sending the evaluations of the polynomials in Y on the left hand side of Equation (18) at θ. Namely, the prover computes

$$\tilde{\boldsymbol{z}}_+ \coloneqq \sum_{i\in[n]} \beta_i \theta^i \tilde{\boldsymbol{r}}_i \boldsymbol{S} \in \mathcal{R}_q^m, \quad \tilde{\boldsymbol{z}}_- \coloneqq \sum_{i\in[n]} \theta^{-i} \tilde{\boldsymbol{r}}_i \boldsymbol{S} \in \mathcal{R}_q^m, \tag{20}$$

and sends them to the verifier, who can simply check the following equation:

$$\langle \tilde{\boldsymbol{z}}_+ \cdot \boldsymbol{D}, \tilde{\boldsymbol{z}}_- \rangle = \sum_{i\in[n]} \beta_i u_i + \sum_{-n<i<n, i\neq 0} v_i \theta^i. \tag{21}$$

The verifier can check Equation (16) using a similar strategy with the w_i:

$$\langle \boldsymbol{\varphi}, \sum_{i\in[n]} \tilde{\boldsymbol{r}}_i \boldsymbol{S}Y^{-i} \rangle = \sum_{i\in[n]} w_i Y^{-i}. \tag{22}$$

Eventually, this means checking the following equation:

$$\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{z}}_- \rangle = \sum_{i\in[n]} w_i \theta^{-i}. \tag{23}$$

Provided that $\tilde{\boldsymbol{z}}_\pm$ were computed correctly as in Equation (24), checking (21),(23) guarantees to the verifier that (18) and (22) hold, except with a negligible soundness error. The Right Part of the Left Right protocol will ensure that a cheating prover who does not compute $\tilde{\boldsymbol{z}}_\pm$ correctly will be caught by the verifier.

As mentioned earlier in this section, the vectors $\tilde{\boldsymbol{z}}_\pm$ are linear combinations of the rows of the witness matrix $\boldsymbol{S}$, obtained by multiplying from the left:

$$\tilde{\boldsymbol{z}}_+ = [\theta\beta_1, \ldots, \theta^n \beta_n] \boldsymbol{R} \cdot \boldsymbol{S}, \quad \tilde{\boldsymbol{z}}_- = [\theta^{-1}, \ldots, \theta^{-n}] \boldsymbol{R} \cdot \boldsymbol{S}, \tag{24}$$

As such, we refer to $\tilde{\boldsymbol{z}}_\pm$ as amortizations of rows.

Note that we do not incorporate $\boldsymbol{D}$ into $\tilde{\boldsymbol{z}}_+$ in Equation (21) because this would involve multiplying $\boldsymbol{S}$ from the *right* hand side, and make it more costly to check that $\tilde{\boldsymbol{z}}_+$ was computed correctly. To achieve this, we would have to check that the row amortizations of both $\boldsymbol{S}$ and $\boldsymbol{S}\boldsymbol{D}$ were computed correctly, requiring extra elements to be sent in the Right Part of the protocol, which we would like to avoid.

3.3 The Right Part

The Right Part completes the Left-Right protocol, by convincing the verifier that the row amortizations $\tilde{\boldsymbol{z}}_\pm$ were correctly computed from the witness matrix $\boldsymbol{S}$ whose columns are committed in $\boldsymbol{T}$. It comes down to running the linear protocol from Section 2.1 to prove that

$$\begin{pmatrix} \boldsymbol{A} \\ [\theta\beta_1, \ldots, \theta^n\beta_n]\boldsymbol{R} \\ [\theta^{-1}, \ldots, \theta^{-n}]\boldsymbol{R} \end{pmatrix} \cdot \boldsymbol{S} = \begin{pmatrix} \boldsymbol{T} \\ \tilde{\boldsymbol{z}}_+ \\ \tilde{\boldsymbol{z}}_- \end{pmatrix}. \tag{25}$$

The linear proof involves sending vector $\boldsymbol{z} = \boldsymbol{S}\boldsymbol{c}$, a linear combination of the columns of the witness matrix $\boldsymbol{S}$, obtained by multiplying from the right.

3.4 Efficiency

The communication complexity is $O(n+m+r)$, which is sublinear in the size of $\boldsymbol{S}$, and the verifier runtime is also $O(n+m+r)$.

3.5 Security Analysis

Completeness. If the quadratic described in Equation (15) holds for all $i \in [n]$, then taking linear combinations immediately yields Equation (17). For v_i computed as in Equation (19), the polynomial equality in Y from Equation (18) is satisfied, and hence also satisfied when evaluated at θ. Therefore, the verifier check from Equation (21) passes. Similarly, assuming that Equation (16) holds, the polynomial equality in Y from Equation (22) also holds, and so does its evaluation at θ. This means that the verifier check from Equation (23) passes. Thus, the completeness of the Left-Right protocol follows from the completeness of the Linear Proof from Section 2.1.

Knowledge Soundness. The Left-Right protocol is knowledge sound under the Module-SIS assumption. As in Section 2.1, this protocol is knowledge sound for a relaxed relation $R'_{\mathtt{LR}}$ with looser norm bounds than $R_{\mathtt{LR}}$ (see Lemma 3).

Theorem 1. *Let $\hat{\mathcal{C}} = \mathcal{R}_q$, and $\mathcal{C} \subseteq \mathcal{R}_q$ be the challenge space from Section 2.1 of polynomials of operator norm T and ℓ_2 norm τ. Suppose that M-SIS$_{t,8TB}$ is hard, where B is as in Section 2.1. Then the protocol in Section 3.1 is a knowledge-sound proof for the relation R'_{LR} defined as*

$$R'_{LR} = \left\{ ((\boldsymbol{T}, B, \boldsymbol{R}, \boldsymbol{D}, \boldsymbol{\varphi}, (u_i, w_i)_i), (\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})) \;\middle|\; \begin{array}{c} \boldsymbol{A}\bar{\boldsymbol{S}} = \boldsymbol{T},\ \|\bar{\boldsymbol{c}}_i\bar{\boldsymbol{s}}_i\| \le 2B,\ i \in [m] \\ \langle \tilde{\boldsymbol{r}}_i\bar{\boldsymbol{S}}\boldsymbol{D}, \tilde{\boldsymbol{r}}_i\bar{\boldsymbol{S}}\rangle = u_i,\ i \in [n] \\ \langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_i\bar{\boldsymbol{S}}\rangle = w_i,\ i \in [n] \end{array} \right\}$$

with knowledge error $\kappa_{LR} \coloneqq \frac{2n-1}{q^{d/2}} + \kappa_{lin}$.

Proof. Let $\mathcal{A}$ be an adversary for the protocol in Figure 3. Suppose that $\mathcal{A}$ convinces the verifier $\mathcal{V}$ to accept with probability ϵ. We describe a knowledge extractor $\mathcal{E}_{\mathtt{LR}}$ for the protocol. On input public parameters $\boldsymbol{A}$ and instance $(\boldsymbol{T}, B, \boldsymbol{R}, \boldsymbol{D}, \boldsymbol{\varphi}, (u_i, w_i)_i)$ the extractor does as follows:

1. The extractor samples $(\beta_i)_i, \theta \leftarrow \mathcal{R}_q$, computes $\boldsymbol{B} \coloneqq \begin{pmatrix} [\theta\beta_1, \ldots, \theta^n\beta_n]\boldsymbol{R} \\ [\theta^{-1}, \ldots, \theta^{-n}]\boldsymbol{R} \end{pmatrix} \in \mathcal{R}_q^{2\times r}$ and $\boldsymbol{U} \coloneqq \begin{pmatrix} \tilde{\boldsymbol{z}}_+ \\ \tilde{\boldsymbol{z}}_- \end{pmatrix} \in \mathcal{R}_q^{2\times m}$, and runs $\mathcal{E}_{\mathtt{lin}}$ (from Appendix B.3) on input $\boldsymbol{A}$ and instance $(\boldsymbol{T}, (B_i)_i, \boldsymbol{B}, \boldsymbol{U})$ which then calls $\mathcal{P}^*_{\mathtt{lin}}((\beta_i)_i, \theta)$ in a black-box way.
2. If the tuple $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ is not a valid witness for $R'_{\mathtt{lin}}$, then the extractor aborts.
3. If the extractor succeeds at Step 2, it checks whether $((\boldsymbol{T}, B, \boldsymbol{R}, \boldsymbol{D}, \boldsymbol{\varphi}, (u_i, w_i)_i), (\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})) \in R'_{\mathtt{LR}}$ and if so, outputs $\bar{\boldsymbol{c}} \coloneqq (\bar{\boldsymbol{c}}_i)_i \in \bar{\mathcal{C}}^m$ and $\bar{\boldsymbol{S}} \coloneqq [\bar{\boldsymbol{s}}_1 | \ldots | \bar{\boldsymbol{s}}_m] \in \mathcal{R}_q^{r\times m}$ and terminates.

4. If the extractor succeeds at Step 2, but it does not produce a valid witness according to Step 3, the extractor samples $(\beta'_i)_i, \theta' \leftarrow \mathcal{R}_q$ and proceeds as in Step 1 by using the extractor $\mathcal{E}_{\mathtt{lin}}$, (which then calls the malicious prover $\mathcal{P}^*_{\mathtt{lin}}((\beta'_i)_i, \theta')$), until it outputs a new tuple $(\bar{\boldsymbol{S}}', \bar{\boldsymbol{c}}')$, a valid witness for the relation $R'_{\mathtt{lin}}$.
5. If $\bar{\boldsymbol{S}} = \bar{\boldsymbol{S}}'$, then the extractor aborts. Else, the extractor outputs a Module-SIS solution $\bar{\boldsymbol{c}}'_j \bar{\boldsymbol{c}}_j \bar{\boldsymbol{s}}_j - \bar{\boldsymbol{c}}_j \bar{\boldsymbol{c}}'_j \bar{\boldsymbol{s}}'_j$ with respect to $\boldsymbol{A}$ and terminates.

Correctness of the Extractor. The extractor checks whether its output at Step 3 is indeed a valid witness. If not, then when outputting at Step 5, the extractor obtains two tuples, $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ and $(\bar{\boldsymbol{S}}', \bar{\boldsymbol{c}}')$, with $\bar{\boldsymbol{S}} \neq \bar{\boldsymbol{S}}'$, such that $\boldsymbol{A}\bar{\boldsymbol{S}} = \boldsymbol{A}\bar{\boldsymbol{S}}' = \boldsymbol{T}$, and $\max \|\bar{\boldsymbol{c}}_j \bar{\boldsymbol{s}}_j\|, \|\bar{\boldsymbol{c}'}_j \bar{\boldsymbol{s}}'_j\| \leq 2B$, as they are valid witnesses for $R'_{\mathtt{LR}}$. Observe that for some $j \in [m]$ such that $\bar{\boldsymbol{s}}_j \neq \bar{\boldsymbol{s}}'_j$, $\bar{\boldsymbol{c}}'_j \bar{\boldsymbol{c}}_j \bar{\boldsymbol{s}}_j - \bar{\boldsymbol{c}}_j \bar{\boldsymbol{c}}'_j \bar{\boldsymbol{s}}'_j$ is an M-SIS solution, as $\bar{\boldsymbol{c}}'_j \bar{\boldsymbol{c}}_j \boldsymbol{A}\bar{\boldsymbol{s}}_j = \bar{\boldsymbol{c}}_j \bar{\boldsymbol{c}}'_j \boldsymbol{A}\bar{\boldsymbol{s}}'_j = \bar{\boldsymbol{c}}'_j \bar{\boldsymbol{c}}_j \boldsymbol{t}_j$, and its norm is bounded, using the triangle inequality and the differences of challenges having operator norm bounded by $2T$, by the choice of the challenge space $\mathcal{C}$ in Section 2.1. We have $\|\bar{\boldsymbol{c}}'_j \bar{\boldsymbol{c}}_j \bar{\boldsymbol{s}}_j - \bar{\boldsymbol{c}}_j \bar{\boldsymbol{c}}'_j \bar{\boldsymbol{s}}'_j\| \leq 2T \cdot 2B + 2T \cdot 2B = 8TB$.

Expected Extractor Running Time. Assume $\mathcal{E}_{\mathtt{lin}}$ runs in time $T_{\mathtt{lin}}$. Then our extractor $\mathcal{E}_{\mathtt{LR}}$ runs Step 1 in time $T_{\mathtt{lin}}$. Moreover, the success probability of $\mathcal{E}_{\mathtt{lin}}$ is, according to Lemma 3, $\epsilon - \kappa_{\mathtt{lin}} = \epsilon - \frac{1}{|\mathcal{C}|}$. As the expected number of reruns at Step 4 is $\frac{1}{\epsilon - \kappa_{\mathtt{lin}}}$, each running in $T_{\mathtt{lin}}$ time, we thus obtain the expected running time of our extractor as

$$T_{\mathtt{lin}} + (\epsilon - \kappa_{\mathtt{lin}}) \cdot \frac{T_{\mathtt{lin}}}{\epsilon - \kappa_{\mathtt{lin}}} = 2T_{\mathtt{lin}}.$$

According to Lemma 3, as $\mathcal{E}_{\mathtt{lin}} = 2Tm$, where T is the running time of $\mathcal{P}^*_{\mathtt{lin}}$, our extractor runs in expected polynomial time in T and m.

Success Probability of the Extractor. To argue knowledge soundness according to the definition in Appendix A.4, we show that in fact, when $\bar{\boldsymbol{S}}$ is not a valid witness, then the extractor will output an M-SIS solution.

Whenever $\mathcal{E}_{\mathtt{lin}}$, and hence $\mathcal{E}_{\mathtt{LR}}$, do not abort at Step 1, the output $\bar{\boldsymbol{S}}$ satisfies

$$\begin{pmatrix} \boldsymbol{A} \\ [\theta\beta_1, \ldots, \theta^n \beta_n]\boldsymbol{R} \\ [\theta^{-1}, \ldots, \theta^{-n}]\boldsymbol{R} \end{pmatrix} \cdot \bar{\boldsymbol{S}} = \begin{pmatrix} \boldsymbol{T} \\ \tilde{\boldsymbol{z}}_+ \\ \tilde{\boldsymbol{z}}_- \end{pmatrix}. \tag{26}$$

If $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ produced at Step 1 is nevertheless not a witness for $R'_{\mathtt{LR}}$, then at least one of the following conditions must hold:

$$\langle \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \boldsymbol{D}, \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \rangle \neq u_i, \text{for some } i \in [n] \text{ or } \langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \rangle \neq w_i, \text{ for some } i \in [n]. \tag{27}$$

Second Case. Assume the second case i.e. $\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \rangle \neq w_i$ for some $i \in [n]$. Taking a linear combination using powers of θ', using Lemma 5, and scaling by $(\theta')^n$, we have that

$$\sum_{i \in [n]} \langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \rangle (\theta')^{-i} \neq \sum_{i \in [n]} w_i (\theta')^{-i}, \tag{28}$$

except with probability at most $(n-1)/q^{d/2}$, for the uniformly random $\theta' \leftarrow \hat{\mathcal{C}} = \mathcal{R}_q$ when re-running the protocol in Step 4. Then, later during Step 4, the malicious prover sends $\tilde{\boldsymbol{z}}'_-$ to the verifier. Either $\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{z}}'_- \rangle \neq \sum_{i \in [n]} w_i (\theta')^{-i}$, in which case, the verifier will definitely reject (as Equation (23) does not hold), or $\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{z}}'_- \rangle = \sum_{i \in [n]} w_i (\theta')^{-i}$, in which case, Equation (28) implies that $\tilde{\boldsymbol{z}}'_- \neq \sum_{i \in [n]} (\theta')^{-i} \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}}$.

If $\mathcal{E}_{\mathtt{lin}}$ extracts a valid witness $\bar{\boldsymbol{S}}'$ at Step 4, then similarly to $\bar{\boldsymbol{S}}$ and Equation (26), its output $\bar{\boldsymbol{S}}'$ satisfies $\tilde{\boldsymbol{z}}'_- = \sum_{i \in [n]} (\theta')^{-i} \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}}'$. This means that we cannot have $\bar{\boldsymbol{S}} = \bar{\boldsymbol{S}}'$, and so $\mathcal{E}_{\mathtt{LR}}$ outputs an M-SIS solution.

First Case. Now, assume the first case i.e. $\langle \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \boldsymbol{D}, \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \rangle \neq u_i$ for some $i \in [n]$. Taking a linear combination using the β'_i, and using Lemma 5, we have that

$$\sum_{i \in [n]} \beta'_i \langle \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \boldsymbol{D}, \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \rangle \neq \sum_{i \in [n]} \beta'_i u_i, \tag{29}$$

except with probability at most $1/q^{d/2}$, for the uniformly random $\beta'_i \leftarrow \mathcal{R}_q$ when re-running the protocol in Step 4. Further, for $\bar{v}_i := \sum_{k,j\in[n]:k-j=i} \beta'_k \langle \tilde{\boldsymbol{r}}_k \bar{\boldsymbol{S}} \boldsymbol{D}, \tilde{\boldsymbol{r}}_j \bar{\boldsymbol{S}} \rangle$ and any $\theta' \in \mathcal{C}$, we have

$$\left\langle \Big(\sum_{i\in[n]} \beta'_i (\theta')^i \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \Big) \boldsymbol{D}, \sum_{i\in[n]} (\theta')^{-i} \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \right\rangle = \sum_{i\in[n]} \beta'_i \langle \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \boldsymbol{D}, \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \rangle + \sum_{-n<i<n, i\neq 0} \bar{v}_i (\theta')^i. \tag{30}$$

By Equations (29) and (30) and Lemma 5, we have that for any choice of prover messages v'_i in Step 4,

$$\left\langle \Big(\sum_{i\in[n]} \beta'_i (\theta')^i \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \Big) \boldsymbol{D}, \sum_{i\in[n]} (\theta')^{-i} \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}} \right\rangle \neq \sum_{i\in[n]} \beta'_i u_i + \sum_{-n<i<n, i\neq 0} v'_i (\theta')^i. \tag{31}$$

except with probability at most $(2n-2)/q^{d/2}$, for the uniformly random $\theta' \in \mathcal{C}$ drawn in Step 4.

Later during Step 4, the malicious prover sends $\tilde{\boldsymbol{z}}'_+$ and $\tilde{\boldsymbol{z}}'_-$ to the verifier. Either $\langle \tilde{\boldsymbol{z}}'_+ \cdot \boldsymbol{D}, \tilde{\boldsymbol{z}}'_- \rangle \neq \sum_{i\in[n]} \beta'_i u_i + \sum_{i\neq 0} v'_i (\theta')^i$, in which case, the verifier will definitely reject (as Equation (21) does not hold), or $\langle \tilde{\boldsymbol{z}}'_+ \cdot \boldsymbol{D}, \tilde{\boldsymbol{z}}'_- \rangle = \sum_{i\in[n]} \beta'_i u_i + \sum_{i\neq 0} v'_i (\theta')^i$, in which case, Equation (31) implies that either $\tilde{\boldsymbol{z}}'_+ \neq \sum_{i\in[n]} \beta'_i (\theta')^i \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}}$ or $\tilde{\boldsymbol{z}}'_- \neq \sum_{i\in[n]} (\theta')^{-i} \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}}$.

Similarly to previous reasoning, if $\mathcal{E}_{\mathtt{lin}}$ does not abort at Step 4, then its output $\bar{\boldsymbol{S}}'$ satisfies $\tilde{\boldsymbol{z}}_+ = \sum_{i\in[n]} \beta'_i (\theta')^i \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}}'$ and $\tilde{\boldsymbol{z}}'_- = \sum_{i\in[n]} (\theta')^{-i} \tilde{\boldsymbol{r}}_i \bar{\boldsymbol{S}}'$. Again, this means that we cannot have $\bar{\boldsymbol{S}} = \bar{\boldsymbol{S}}'$, and $\mathcal{E}_{\mathtt{LR}}$ outputs an M-SIS solution.

The Final Knowledge Error. Note that our extractor succeeds if it outputs a witness for $R'_{\mathtt{LR}}$ in Step 3 or it outputs the SIS solution in Step 5. As our extractor gets the matrix $\bar{\boldsymbol{S}}$ with probability at least $\epsilon - \kappa_{\mathtt{lin}}$, then, according to the two cases above, the second matrix $\bar{\boldsymbol{S}}'$ could coincide with the first one with probability at most $\max\left((n-1)/q^{d/2}, (n-1)/q^{d/2}\right) = (n-1)/q^{d/2}$, we conclude that the success probability of our extractor should be at least $\epsilon - \kappa_{\mathtt{lin}} - (n-1)/q^{d/2}$. Thus, its knowledge soundness is $\kappa_{\mathtt{LR}} := (2n-1)/q^{d/2} + \kappa_{\mathtt{lin}}$.

4 From the Left-Right Protocol to Orthus

In this section we show how to leverage the Left-Right protocol from Section 3 to prove that the rows $\tilde{\boldsymbol{s}}_i$ of a witness matrix $\boldsymbol{S} \in \mathcal{R}^{nK\times m}$ satisfy more general constraints.

Given Ajtai commitments to the columns of $\boldsymbol{S}$, the Left-Right protocol proves quadratic relations involving inner products $\langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_i \rangle$ between rows at the same index. The first generalization partitions $\boldsymbol{S}$ into n groups of K consecutive rows, which we call "row blocks", and allows *mixed* inner-product constraints $\langle \tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_j \rangle$ between two rows $\tilde{\boldsymbol{s}}_i, \tilde{\boldsymbol{s}}_j$ from the same row block. In Section 4.1, we explain how to prove one such constraint per row block, and in Section 4.2 we cover the case of several such constraints per row block, along with approximate shortness proofs. This means that we can support arbitrary LaBRADOR dot-product constraints as in Equation (14), within each row block.

The second generalization leads to our main protocol, Orthus, which we describe in Section 4.3. Here, we further partition the row blocks into groups of consecutive columns that we call simply "blocks", and support LaBRADOR dot-product constraints within each block. We refer the reader to Figure 1 for an overview for the main protocol Orthus.

4.1 Proving the Same Constraint for Each Row Block

We provide a high-level description of a protocol for proving that every row block satisfies the same constraint. A full algorithmic description of the protocol can be found in Figure 4. As previously described, we partition the witness matrix $\boldsymbol{S} \in \mathcal{R}^{nK\times m}$ into n groups of K consecutive rows, which we call row blocks. For $\ell \in [n]$, we denote the rows of the ℓth row-block by $\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)} \in \mathcal{R}^m$. The columns $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m \in \mathcal{R}^{nK}$ of $\boldsymbol{S}$ have bounded norms $\|\boldsymbol{s}_i\| \leq B_i$ and are committed using Ajtai commitment $\boldsymbol{T} = \boldsymbol{A}\boldsymbol{S} \in \mathcal{R}_q^{t\times m}$, for a matrix $\boldsymbol{A} \in \mathcal{R}_q^{t\times nK}$.

The protocol proves that for each $\ell \in [n]$, the ℓ-th row block satisfies a dot-product constraint $F^{(\ell)} : \mathcal{R}_q^m \times \ldots \times \mathcal{R}_q^m \to \mathcal{R}_q$ of the following form:

$$F^{(\ell)}(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}) \triangleq \sum_{i,j \in [K]} a_{ij} \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \sum_{i \in [K]} \langle \boldsymbol{\varphi}_i, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle - b^{(\ell)} = 0. \tag{32}$$

Here $a_{ij}, b^{(\ell)} \in \mathcal{R}_q$, $\boldsymbol{\varphi}_i \in \mathcal{R}_q^m$, with $a_{ij} = a_{ji}$, and $\boldsymbol{D} \in \mathcal{R}_q^{m \times m}$ such that $\boldsymbol{D}^T = \boldsymbol{D}$. This constraint has a slightly different form from the LaBRADOR-type constraint of Equation (14). We allow the rows $\tilde{\boldsymbol{s}}_i^{(\ell)}$ from the quadratic parts to be modified by a public matrix $\boldsymbol{D}$, which will be later instantiated in the main protocol (Section 4.3). The symmetry of this matrix is required in the soundness proof (see Theorem 2). Note that the constant terms $b^{(\ell)}$ are parametrized by ℓ, whereas the coefficients a_{ij} and the vectors $\boldsymbol{\varphi}_i$ are independent of ℓ. We allow for the terms $b^{(\ell)}$ to be different as we will need this in the reduction from Section 4.2. More concretely, we provide a protocol for the relation

$$R_{\texttt{row-block-1}} = \left\{ ((\boldsymbol{T}, (B_i)_i, (F^{(\ell)}), \boldsymbol{D}), \boldsymbol{S}) \middle| \begin{array}{l} \boldsymbol{AS} = \boldsymbol{T}, \|\boldsymbol{s}_i\| \leq B_i,\ i \in [m] \\ F^{(\ell)}(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots \tilde{\boldsymbol{s}}_K^{(\ell)}) = 0,\ \ell \in [n] \end{array} \right\}.$$

The protocol reduces $R_{\texttt{row-block-1}}$ to $R_{\texttt{LR}}$ and relies on the Left-Right protocol from Section 3 to prove the latter, as we now describe.

Reducing Row Block Constraints to a Left-Right Instance. To prove that the row blocks satisfy certain constraints, we "collapse" each row block into a linear combination $\boldsymbol{z} \in \mathcal{R}_q^m$ of the rows within the row block, using random ring elements from a challenge space C. Similarly to LaBRADOR [11], we reduce the original constraints to constraints on the linear combinations $\boldsymbol{z}$ using auxiliary elements sent by the prover to the verifier before having received the random challenges. We then prove the constraints on $\boldsymbol{z}$ using the Left-Right protocol from Section 3.1. Note that while in [11] the challenge space used when computing the random linear combinations had to consist of short ring elements, here we can take $C = \mathcal{R}_q$, the entire ring.

The prover starts by sending the following auxiliary elements in $\mathcal{R}_q$

$$\boldsymbol{g}_{ij}^{(\ell)} \coloneqq \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle,\ \boldsymbol{h}_{ij}^{(\ell)} \coloneqq \frac{1}{2}(\langle \boldsymbol{\varphi}_i, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \langle \boldsymbol{\varphi}_j, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle), \tag{33}$$

for $i, j \in [K]$ and $\ell \in [n]$. These elements are also called "garbage", as they are used to help the verifier check Equation (32). Notice that, for any ℓ, the matrix $(\boldsymbol{h}_{ij}^{(\ell)})$ is symmetric. The other matrix $(\boldsymbol{g}_{ij}^{(\ell)})$ is also symmetric, as, since $\boldsymbol{D}^T = \boldsymbol{D}$, $\langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle = \langle \tilde{\boldsymbol{s}}_j^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle$. Therefore, the prover can send only $n \cdot K(K+1)$ garbage elements $\boldsymbol{g}_{ij}^{(\ell)}$ and $\boldsymbol{h}_{ij}^{(\ell)}$.

Upon receiving the random elements $c_1, \ldots, c_K \in C$ from the verifier, the prover collapses each row block of index $\ell \in [n]$, by computing the row vector

$$\boldsymbol{z}^{(\ell)} \coloneqq c_1 \tilde{\boldsymbol{s}}_1^{(\ell)} + \ldots + c_K \tilde{\boldsymbol{s}}_K^{(\ell)} \in \mathcal{R}_q^m.$$

Proving that Equation (32) is satisfied comes down to showing

$$\langle \boldsymbol{z}^{(\ell)} \boldsymbol{D}, \boldsymbol{z}^{(\ell)} \rangle = \sum_{i,j \in [K]} \boldsymbol{g}_{ij}^{(\ell)} c_i c_j,\ \sum_{i \in [K]} \langle \boldsymbol{\varphi}_i, \boldsymbol{z}^{(\ell)} \rangle c_i = \sum_{i,j \in [K]} \boldsymbol{h}_{ij}^{(\ell)} c_i c_j,\ \ell \in [n], \tag{34}$$

$$\sum_{i,j \in [K]} a_{ij} \boldsymbol{g}_{ij}^{(\ell)} + \sum_{i \in [K]} \boldsymbol{h}_{ii}^{(\ell)} - b^{(\ell)} = 0,\ \ell \in [n]. \tag{35}$$

Equation (34) assures the well-formedness of the garbage terms $\boldsymbol{g}_{ij}^{(\ell)}, \boldsymbol{h}_{ij}^{(\ell)}$. Since the vectors $\boldsymbol{z}^{(\ell)}$ are linear combinations of the rows of $\boldsymbol{S}$, we can prove that Equation (34) is satisfied using the Left-Right protocol, defining $\boldsymbol{R}$ in terms of the challenges c_i. As for Equation (35), it can be checked directly by the verifier.

We now describe the concrete instance to prove with the Left-Right protocol. Notice that the $\boldsymbol{z}^{(\ell)}$'s are linear combinations of the rows of $\boldsymbol{S}$:

$$\boldsymbol{z}^{(\ell)} = (c_1 \dots c_K) \begin{pmatrix} \tilde{\boldsymbol{s}}_1^{(\ell)} \\ \dots \\ \tilde{\boldsymbol{s}}_K^{(\ell)} \end{pmatrix} = \tilde{\boldsymbol{r}}_\ell \cdot \boldsymbol{S} \in \mathcal{R}_q^m, \tag{36}$$

where $\tilde{\boldsymbol{r}}_\ell \coloneqq [\boldsymbol{0}_K | \dots | c_1, \dots, c_K | \dots | \boldsymbol{0}_K] \in \mathcal{R}^{nK}$, with the challenges c_i's on the ℓ-th position, for any $\ell \in [n]$. Let $\boldsymbol{\varphi} \coloneqq \sum_{i\in[K]} c_i \boldsymbol{\varphi}_i$. Rewriting Equation (34), one could finish by using the Left-Right protocol in Section 3.1 to prove

$$\begin{aligned} &\boldsymbol{A}\boldsymbol{S} = \boldsymbol{T}, \\ &\langle \tilde{\boldsymbol{r}}_\ell \boldsymbol{S}\boldsymbol{D}, \tilde{\boldsymbol{r}}_\ell \boldsymbol{S} \rangle = \sum_{i,j\in[K]} \boldsymbol{g}_{ij}^{(\ell)} c_i c_j, \ \ell \in [n], \\ &\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_\ell \boldsymbol{S} \rangle = \sum_{i,j\in[K]} \boldsymbol{h}_{ij}^{(\ell)} c_i c_j, \ \ell \in [n]. \end{aligned} \tag{37}$$

Efficiency. The communication complexity is $O(nK^2 + m)$ and the verifier runtime is $O(nK^2 + mK)$.

Security Analysis. The completeness of the protocol easily follows from the completeness of the Left-Right protocol. Here we only state the knowledge soundness. Similarly to Sections 2.1 and 3.1, knowledge soundness is satisfied for a relaxed relation $R'_{\texttt{row-block-1}}$ with looser norm bounds than $R_{\texttt{row-block-1}}$.

Theorem 2. *Let $\hat{\mathcal{C}} = C = \mathcal{R}_q$ and $\mathcal{C} \subseteq \mathcal{R}_q$ be the challenge space from Section 2.1 of polynomials of operator norm T and ℓ_2 norm τ. Suppose that M-SIS$_{t,8TB}$ is hard, where B is as in Section 2.1. Then the protocol in Section 4.1 is a knowledge-sound proof for the relation $R'_{\texttt{row-block-1}}$ defined as*

$$R'_{\texttt{row-block-1}} = \left\{ ((\boldsymbol{T}, B, (F^{(\ell)}), \boldsymbol{D}), (\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})) \middle| \begin{matrix} \boldsymbol{A}\bar{\boldsymbol{S}} = \boldsymbol{T}, \|\bar{\boldsymbol{c}}_i \bar{\boldsymbol{s}}_i\| \le 2B, \ i \in [m] \\ F^{(\ell)}(\bar{\boldsymbol{s}}_1^{(\ell)}, \dots, \bar{\boldsymbol{s}}_K^{(\ell)}) = 0, \ \ell \in [n] \end{matrix} \right\}$$

with knowledge error $\kappa_{\texttt{row-block-1}} \coloneqq \frac{2}{q^{d/2}} + \kappa_{LR}$.

Proof. Let $\mathcal{A}$ be an adversary for the protocol in Figure 4 and suppose that it convinces the verifier $\mathcal{V}$ to accept with probability ϵ. Without loss of generality, we can assume that the adversary is deterministic. Hence, the prover's first message, made by $\boldsymbol{g}_{ij}^{(\ell)}, \boldsymbol{h}_{ij}^{(\ell)}$, is the same in any (accepting) transcript. We describe a knowledge extractor $\mathcal{E}_{\text{row-block-1}}$ for the protocol, similarly as in the proof of Theorem 1. On input public parameters $\boldsymbol{A}$, instance $(\boldsymbol{T}, B, (F^{(\ell)}), \boldsymbol{D})$ the extractor $\mathcal{E}_{\text{row-block-1}}$ proceeds as follows.

1. The extractor samples $(c_i)_i \leftarrow \mathcal{R}_q$, computes $\boldsymbol{R} \in \mathcal{R}_q^{n\times nK}$ of rows $\tilde{\boldsymbol{r}}^\ell \coloneqq [\boldsymbol{0}_K | \dots | c_1, \dots, c_K | \dots \boldsymbol{0}_K] \in \mathcal{R}_q^{nK}$, for any $\ell \in [n]$, $\boldsymbol{\varphi} \coloneqq \sum_{i\in[K]} c_i \varphi_i \in \mathcal{R}_q^m$, $(u_\ell, w_\ell) = (\sum_{i,j\in[K]} \boldsymbol{g}_{ij}^{(\ell)} c_i c_j, \sum_{i,j\in[K]} \boldsymbol{h}_{ij}^{(\ell)} c_i c_j) \in \mathcal{R}_q \times \mathcal{R}_q$ for any $\ell \in [n]$, and runs $\mathcal{E}_{\texttt{LR}}$ (from the proof of Theorem 1) on input $\boldsymbol{A}$ and instance $(\boldsymbol{T}, (B_i)_i, \boldsymbol{R}, \boldsymbol{D}, \boldsymbol{\varphi}, (u_\ell, w_\ell)_\ell)$, which then calls $\mathcal{P}^*_{\texttt{LR}}((c_i)_i)$ in a black-box way.
2. If the tuple $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ is not a valid witness for the relation $R'_{\texttt{LR}}$, then the extractor aborts.
3. If the extractor succeeds at Step 2, it checks whether $((\boldsymbol{T}, B, (F^{(\ell)}), \boldsymbol{D}), (\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})) \in R'_{\texttt{row-block-1}}$ and if so, outputs $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ and terminates.
4. If the extractor succeeds at Step 2, but it does not produce a valid witness at Step 3, the extractor samples $(c_i)_i \leftarrow \mathcal{R}_q$ and proceeds as in Step 1 (which then calls the malicious prover $\mathcal{P}^*_{\texttt{LR}}((c_i)_i)$, until it outputs a new tuple $(\bar{\boldsymbol{S}}', \bar{\boldsymbol{c}}')$ witness for the relation $R'_{\texttt{LR}}$.
5. If $\bar{\boldsymbol{S}} = \bar{\boldsymbol{S}}'$, the extractor aborts. Else, the extractor outputs a Module-SIS solution and terminates.

Following the proof of Theorem 1, the correctness of the extractor holds, and, moreover, the expected running time of the extractor is double that of $\mathcal{E}_{\texttt{LR}}$, i.e. $8Tm$.

Success Probability of the Extractor. We focus on the probability that $\mathcal{E}_{\texttt{row-block-1}}$ fails to produce a witness or a valid M-SIS solution, which can happen when this extractor aborts. As discussed in the proof of Theorem 1, when $\bar{\boldsymbol{S}}$ is not a valid witness, the extractor will output an M-SIS solution.

Whenever $\mathcal{E}_{\texttt{row-block-1}}$, and hence $\mathcal{E}_{\texttt{LR}}$, do not abort at Step 1, the output $\bar{\boldsymbol{S}}$ satisfies

$$\boldsymbol{A}\bar{\boldsymbol{S}} = \boldsymbol{T},$$

$$\langle \tilde{\boldsymbol{r}}_\ell \bar{\boldsymbol{S}} \boldsymbol{D}, \tilde{\boldsymbol{r}}_\ell \bar{\boldsymbol{S}} \rangle = \sum_{i,j\in[K]} \boldsymbol{g}_{ij}^{(\ell)} c_i c_j, \text{ for any } \ell \in [n]. \tag{38}$$

$$\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_\ell \bar{\boldsymbol{S}} \rangle = \sum_{i,j\in[K]} \boldsymbol{h}_{ij}^{(\ell)} c_i c_j, \text{ for any } \ell \in [n]. \tag{39}$$

If $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ produced at Step 1 is not a valid witness for $R'_{\text{row-block-1}}$, then

$$\sum_{i,j\in[K]} a_{ij} \langle \bar{\boldsymbol{s}}_i^{(\ell_0)} \boldsymbol{D}, \bar{\boldsymbol{s}}_j^{(\ell_0)} \rangle + \sum_{i\in[K]} \langle \boldsymbol{\varphi}_i, \bar{\boldsymbol{s}}_i^{(\ell_0)} \rangle - b^{(\ell_0)} \neq 0, \text{ for some } \ell_0 \in [n]. \tag{40}$$

Equation (40) implies that the garbage terms are maliciously formed, i.e either $\boldsymbol{g}_{i_0 j_0}^{(\ell_0)} \neq \langle \bar{\boldsymbol{s}}_{i_0}^{(\ell_0)} \boldsymbol{D}, \bar{\boldsymbol{s}}_{j_0}^{(\ell_0)} \rangle$, for some $i_0, j_0 \in [K]$, or $\boldsymbol{h}_{i_0 i_0}^{(\ell_0)} \neq \langle \boldsymbol{\varphi}_{i_0}, \bar{\boldsymbol{s}}_{i_0}^{(\ell_0)} \rangle$, for some $i_0 \in [K]$, since Equation (35) holds.

Second Case. Assume $\boldsymbol{h}_{i_0 i_0}^{(\ell_0)} \neq \langle \boldsymbol{\varphi}_{i_0}, \bar{\boldsymbol{s}}_{i_0}^{(\ell_0)} \rangle$, for some $i_0 \in [K]$. Then, for $\boldsymbol{\varphi}' \coloneqq \sum_{i\in[K]} c'_i \boldsymbol{\varphi}_i$ and $\tilde{\boldsymbol{r}}'_{\ell_0} \coloneqq [\boldsymbol{0}_K \dots | c'_1 \dots c'_K | \dots \boldsymbol{0}_K]$, writing $\tilde{\boldsymbol{r}}'_{\ell_0} \bar{\boldsymbol{S}} = c'_{i_0} \bar{\boldsymbol{s}}_{i_0}^{(\ell_0)} + \boldsymbol{y}$ with $\boldsymbol{y}$ depending only on $\bar{\boldsymbol{S}}$ and c'_i's for $i \neq i_0$, and using Lemma 5, we have that

$$\langle \boldsymbol{\varphi}', \tilde{\boldsymbol{r}}'_{\ell_0} \bar{\boldsymbol{S}} \rangle \neq \sum_{i,j\in[K]} \boldsymbol{h}_{ij}^{(\ell_0)} c'_i c'_j \tag{41}$$

except with probability at most $2/q^{d/2}$, for uniformly random $c'_{i_0} \leftarrow \mathcal{R}_q$, since otherwise one could rewrite Equation (41) as a non-zero polynomial expression of total degree 2 in c'_{i_0} of leading coefficient $\boldsymbol{h}_{i_0 i_0}^{(\ell_0)} - \langle \boldsymbol{\varphi}_{i_0}, \bar{\boldsymbol{s}}_{i_0}^{(\ell_0)} \rangle$. Then, later, during Step 4, the malicious prover sends $\boldsymbol{z}^{'(\ell)}$ to the verifier. Therefore, as part of a proper instance of the Left-Right Protocol (Section 3.1), $\sum_{i\in[K]} \langle \boldsymbol{\varphi}_i, \boldsymbol{z}^{'(\ell_0)} \rangle c'_i = \sum_{i,j\in[K]} \boldsymbol{h}_{ij}^{(\ell_0)} c'_i c'_j$, and by Equation (41), this implies that $\tilde{\boldsymbol{z}}^{'(\ell_0)} \neq \sum_{i\in[K]} c'_i \bar{\boldsymbol{s}}_i{}^{(\ell_0)}$. If $\mathcal{E}_{\texttt{LR}}$ does not abort at Step 4, (unless with probability $\kappa_{\texttt{LR}}$, then similarly to $\bar{\boldsymbol{S}}$ and Equation (39), we have a new witness $\bar{\boldsymbol{S}}'$ satisfying, in particular, $\langle \boldsymbol{\varphi}', \tilde{\boldsymbol{r}}'_{\ell_0} \bar{\boldsymbol{S}}' \rangle = \sum_{i,j\in[K]} \boldsymbol{h}_{ij}^{(\ell_0)} c'_i c'_j$. Clearly, $\bar{\boldsymbol{S}} \neq \bar{\boldsymbol{S}}'$, which, as in the proof of Theorem 1, this leads to a SIS-break.

First Case. Assume that $\boldsymbol{g}_{i_0 j_0}^{(\ell_0)} \neq \langle \bar{\boldsymbol{s}}_{i_0}^{(\ell_0)} \boldsymbol{D}, \bar{\boldsymbol{s}}_{j_0}^{(\ell_0)} \rangle$, for some $i_0, j_0 \in [K]$. If $i_0 = j_0$, then we proceed exactly as before. If $i_0 \neq j_0$, then, for $\tilde{\boldsymbol{r}}'_{\ell_0}$ defined as in the second case, writing $\tilde{\boldsymbol{r}}'_{\ell_0} \bar{\boldsymbol{S}} = c'_{i_0} \bar{\boldsymbol{s}}_{i_0}^{(\ell_0)} + c'_{j_0} \bar{\boldsymbol{s}}_{j_0}^{(\ell_0)} + \boldsymbol{y}$ with $\boldsymbol{y}$ depending only on $\bar{\boldsymbol{S}}$ and c'_i's for $i \neq i_0, j_0$, and using Lemma 5, we have that

$$\langle \tilde{\boldsymbol{r}}'_{\ell_0} \bar{\boldsymbol{S}} \boldsymbol{D}, \tilde{\boldsymbol{r}}'_{\ell_0} \bar{\boldsymbol{S}} \rangle \neq \sum_{i,j\in[K]} \boldsymbol{g}_{ij}^{(\ell_0)} c'_i c'_j \tag{42}$$

except with probability at most $2/q^{d/2}$, for uniformly random $c'_{i_0}, c'_{j_0} \leftarrow \mathcal{R}_q$, since otherwise one could rewrite Equation (42) as a non-zero polynomial expression of total degree 2 in c'_{i_0} and c'_{j_0} with the coefficient of $c'_{i_0} c'_{j_0}$ as $2\langle \bar{\boldsymbol{s}}_{i_0}^{(\ell_0)} \boldsymbol{D}, \bar{\boldsymbol{s}}_{j_0}^{(\ell_0)} \rangle - 2\boldsymbol{g}_{i_0 j_0}^{(\ell_0)}$. Here we used $\boldsymbol{D}^T = \boldsymbol{D}$. Then, later, during Step 4, the malicious prover sends $\boldsymbol{z}^{'(\ell)}$ to the verifier. As in the second case, $\sum_{i\in[K]} \langle \boldsymbol{z}^{'(\ell_0)} \boldsymbol{D}, \boldsymbol{z}^{'(\ell_0)} \rangle c'_i = \sum_{i,j\in[K]} \boldsymbol{g}_{ij}^{(\ell_0)} c'_i c'_j$, and by Equation (42), this implies that $\tilde{\boldsymbol{z}}^{'(\ell_0)} \neq \sum_{i\in[K]} c'_i \bar{\boldsymbol{s}}_i{}^{(\ell_0)}$. If $\mathcal{E}_{\texttt{LR}}$ does not abort at Step 4, then similarly to $\bar{\boldsymbol{S}}$ and Equation (38), we have a new witness $\bar{\boldsymbol{S}}'$ satisfying, in particular, $\langle \tilde{\boldsymbol{r}}'_{\ell_0} \bar{\boldsymbol{S}}' \boldsymbol{D}, \tilde{\boldsymbol{r}}'_{\ell_0} \bar{\boldsymbol{S}}' \rangle = \sum_{i,j\in[K]} \boldsymbol{g}_{ij}^{(\ell_0)} c'_i c'_j$. Clearly, $\bar{\boldsymbol{S}} \neq \bar{\boldsymbol{S}}'$, which, as in the proof of Theorem 1, this leads to a SIS-break.

The Final Knowledge Error. Note that our extractor succeeds if it outputs a witness for $R'_{\texttt{row-block-1}}$ in Step 3 or it outputs the SIS solution in Step 5. As our extractor gets the matrix $\bar{\boldsymbol{S}}$ with probability at least $\epsilon - \kappa_{\texttt{LR}}$, then, according to the two cases above, the second matrix $\bar{\boldsymbol{S}}'$ could coincide with the first one with probability at most $\max\{2/q^{d/2}, 2/q^{d/2}\}$ we conclude that the success probability of our extractor should be at least $\epsilon - \kappa_{\texttt{LR}} - 2/q^{d/2}$. Thus, its knowledge soundness is $\kappa_{\texttt{row-block-1}} \coloneqq 2/q^{d/2} + \kappa_{\texttt{LR}}$.

4.2 Proving the Same Set of Constraints for Each Row Block

We now consider the task of proving that each of the n row blocks of a witness matrix $\boldsymbol{S} \in \mathcal{R}^{nK \times m}$ satisfies the same LaBRADOR-like constraints (32) grouped as full ring constraints $f \in \mathcal{F}$ and constant-term ones $f' \in \mathcal{F}'$, as well as norm bounds $\texttt{B}_i$ over the rows. As in Section 4.1, in all the constraints from $\mathcal{F}$ and $\mathcal{F}'$ the K rows $\tilde{\boldsymbol{s}}_i^{(\ell)}$ are affected by the same public symmetric matrix $\boldsymbol{D}$, though the constant terms $b^{(\ell)}$ are now independent of ℓ. We again assume that the columns $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m \in \mathcal{R}^{nK}$ of $\boldsymbol{S}$ have bounded norms $\|\boldsymbol{s}_i\| \leq B_i$ and are committed using an Ajtai commitment $\boldsymbol{T} = \boldsymbol{A}\boldsymbol{S} \in \mathcal{R}_q^{t \times m}$, for a matrix $\boldsymbol{A} \in \mathcal{R}_q^{t \times nK}$. More precisely, we consider the relation

$$R_{\texttt{row-block-many}} = \left\{ ((\boldsymbol{T}, (B_i)_i, \mathcal{F}, \mathcal{F}', (\texttt{B}_i)_i, \boldsymbol{D}), \boldsymbol{S}) \;\middle|\; \begin{array}{c} \boldsymbol{A}\boldsymbol{S} = \boldsymbol{T}, \|\boldsymbol{s}_i\| \leq B_i,\ i \in [m] \\ f(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}) = 0, f \in \mathcal{F} \\ \texttt{ct}(f'(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)})) = 0, f' \in \mathcal{F}' \\ \|\tilde{\boldsymbol{s}}_1^{(\ell)}\| \leq \texttt{B}_1, \ldots, \|\tilde{\boldsymbol{s}}_K^{(\ell)}\| \leq \texttt{B}_K \end{array} \right\}.$$

We provide a reduction from $R_{\texttt{row-block-many}}$ to $R_{\texttt{row-block-1}}$, by performing standard techniques, as used e.g. in LaBRADOR [11], and then rely on the protocol from Section 4.1. Here, we present the high level ideas behind the techniques, and defer the full details to Appendix C. A full algorithmic description of the protocol can be found in Figure 5.The prover starts by sending projections of the rows, using a matrix randomly sampled by the verifier. By checking the norm of these projections, the verifier is convinced that the norms of the rows hold, up to a slack factor of $\sqrt{337/30}$.

To prove the correctness of the projections, the prover encodes them as constant-term constraints, which are then linearly combined with integer challenges, along with all $f' \in \mathcal{F}'$, into $\lambda \coloneqq \lceil 128/\log q \rceil$ full ring constraints f''. Subsequently, the prover proves the correctness of these new constraints f'' by further linearly combining them, along with all $f \in \mathcal{F}$, with random ring elements sent by the verifier. In this way, one obtains a single final constraint $F^{(\ell)}$ to prove for each row block of index $\ell \in [n]$, which resembles Equation (32). All its coefficients are the same for each row block, except for the constant term $b^{(\ell)}$. Finally, we can rely on the protocol from Section 4.1. These additional steps come with new prover messages, namely the projections of each row per row block ($256Kn$ elements from $\mathbb{Z}_q$) and the "lifting" polynomials $b''^{(\ell)}_k \in \mathcal{R}_q$, obtained when aggregating with integer challenges λ times per row block, (λn ring elements in total). We also get additional verifier checks on the norms of the projections and on the constant-term of the liftings.

Efficiency. The communication and verifier cost of proving $R_{\texttt{row-block-many}}$ is dominated by the call to the protocol from Section 4.1 that proves $R_{\texttt{row-block-1}}$, so the asymptotics remain the same. In particular, the communication complexity is $O(nK^2 + m)$ and the verifier runtime is $O(nK^2 + mK)$.

Security Analysis. The completeness follows easily from the completeness of the additional steps and from the one of the protocol in Section 4.1. Here we only state the knowledge soundness.

Theorem 3. *Let $\hat{\mathcal{C}} = C = \mathcal{R}_q$ and $\mathcal{C} \subseteq \mathcal{R}_q$ be the challenge space from Section 2.1 of polynomials of operator norm T and ℓ_2 norm τ. Suppose that M-SIS$_{t,8TB}$ is hard, where B is as in Section 2.1. Further, suppose that $\texttt{B}_i \leq \sqrt{30/337}q/125$, for any $i \in [K]$. Then the protocol in Section 4.2 is a knowledge-sound proof for*

the relation $R'_{\texttt{row-block-many}}$ *defined as*

$$R'_{\texttt{row-block-many}} = \left\{ \begin{array}{c|c} ((\boldsymbol{T}, B, \mathcal{F}, \mathcal{F}', (\mathtt{B}_i)_i, \boldsymbol{D}), & \boldsymbol{A}\bar{\boldsymbol{S}} = \boldsymbol{T}, \|\bar{\boldsymbol{c}}_i \bar{\boldsymbol{s}}_i\| \leq 2B,\ i \in [m] \\ (\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})) & f(\bar{\boldsymbol{s}}_1^{(\ell)}, \ldots, \bar{\boldsymbol{s}}_K^{(\ell)}) = 0,\ f \in \mathcal{F} \\ & \boldsymbol{ct}(f'(\bar{\boldsymbol{s}}_1^{(\ell)}, \ldots, \bar{\boldsymbol{s}}_K^{(\ell)})) = 0,\ f \in \mathcal{F}' \\ & \|\bar{\boldsymbol{s}}_1^{(\ell)}\| \leq \sqrt{337/30} B_1, \ldots, \\ & \|\bar{\boldsymbol{s}}_K^{(\ell)}\| \leq \sqrt{337/30} B_K \end{array} \right\}$$

with knowledge error $\kappa_{\texttt{row-block-many}} := \frac{K}{2^{128}} + \left(\frac{1}{q}\right)^{\lambda} + \frac{1}{q^{d/2}} + \kappa_{\texttt{row-block-1}}$.

Proof. Let $\mathcal{A}$ be an adversary for the protocol in Figure 4 and suppose that it convinces the verifier $\mathcal{V}$ to accept with probability ϵ. We describe a knowledge extractor $\mathcal{E}_{\texttt{row-block-many}}$ for the protocol in Figure 5. On input public parameters $\boldsymbol{A}$, instance $(\boldsymbol{T}, (B_i), \mathcal{F}, \mathcal{F}', (\mathtt{B}_i), \boldsymbol{D})$, the extractor $\mathcal{E}_{\texttt{row-block-many}}$ proceeds as follows.

1. The extractor samples $(\boldsymbol{\Pi}, \psi_i^{(k)}, (\omega_i^{(k)}), (\mu_i), (\nu_j))$, computes the constraints $F^{(\ell)}$ described by $a_{ij} \in \mathcal{R}_q$, $\boldsymbol{\varphi}_i \in \mathcal{R}_q^m$, $b^{(\ell)} \in \mathcal{R}_q$ as in (56),(57),(58), for any $\ell \in [n]$, and runs $\mathcal{E}_{\texttt{row-block-1}}$ (from the proof of Theorem 2) on input $\boldsymbol{A}$ and instance $(\boldsymbol{T}, (B_i)_i, (F^{(\ell)}), \boldsymbol{D})$, which then calls the malicious prover $\mathcal{P}^*_{\texttt{row-block-1}}(\boldsymbol{\Pi}, \psi_i^{(k)}, (\omega_i^{(k)}), (\mu_i), (\nu_j))$, producing $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}}) \in \mathcal{R}_q^{nK \times m} \times \bar{\mathcal{C}}^m$.
2. If the tuple $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ is not a valid witness for the relation $R'_{\texttt{row-block-1}}$, then the extractor aborts.
3. The extractor checks whether $((\boldsymbol{T}, (B_i), \mathcal{F}, \mathcal{F}', (\mathtt{B}_i), \boldsymbol{D}), (\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})) \in R'_{\texttt{row-block-many}}$ and, if so, outputs $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ and terminates.
4. If the extractor succeeds at Step 2, but it does not produce a valid witness at Step 3, the extractor samples $(\boldsymbol{\Pi}', \psi_i^{'(k)}, (\omega_i^{'(k)}), (\mu_i'), (\nu_j'))$ and proceeds as in Step 1 (which then calls the malicious prover $\mathcal{P}^*_{\texttt{row-block-1}}((\boldsymbol{\Pi}', \psi_i^{'(k)}, (\omega_i^{'(k)}), (\mu_i'), (\nu_j'))$, until it outputs a new tuple $(\bar{\boldsymbol{S}}', \bar{\boldsymbol{c}}')$, witness for the relation $R'_{\texttt{row-block-1}}$.
5. If $\bar{\boldsymbol{S}} = \bar{\boldsymbol{S}}'$, the extractor aborts. Else, the extractor outputs a Module-SIS solution and terminates.

Following the proof of Theorem 1, the extractor is correct and runs in expected time double as $\mathcal{E}_{\texttt{row-block-1}}$.

Success Probability of the Extractor. Whenever $\mathcal{E}_{\texttt{row-block-1}}$, and hence $\mathcal{E}_{\texttt{row-block-many}}$ do not abort at Step 1, the output $\bar{\boldsymbol{S}} \in \mathcal{R}_q^{nK \times m}$ made by n row blocks of rows $\bar{\boldsymbol{s}}_1^{(\ell)}, \ldots, \bar{\boldsymbol{s}}_K^{(\ell)}$, for $\ell \in [n]$, satisfies

$$\begin{aligned} &\boldsymbol{A}\bar{\boldsymbol{S}} = \boldsymbol{T}, \\ &F^{(\ell)}(\bar{\boldsymbol{s}}_1^{(\ell)}, \ldots, \bar{\boldsymbol{s}}_K^{(\ell)}) \triangleq \sum_{i,j \in [K]} a_{ij} \langle \bar{\boldsymbol{s}}_i^{(\ell} \boldsymbol{D}, \bar{\boldsymbol{s}}_j^{(\ell)} \rangle + \sum_{i \in [K]} \langle \boldsymbol{\varphi}_i, \bar{\boldsymbol{s}}_i^{(\ell)} \rangle - b^{(\ell)} = 0,\ \forall \ell \in [n], \end{aligned} \tag{43}$$

where $F^{(\ell)}$ is the final constraint satisfied by each row block, obtained in the instantiation of the protocol in Section 4.1. If $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ produced at Step 1 does not satisfy the relation $R'_{\texttt{row-block-many}}$, it must be the case that one of the following events X, Y or Z holds, where:

- X is the event that one of the constraints $f \in \mathcal{F}$ is not satisfied by (some) row blocks of $\bar{\boldsymbol{S}}$;
- Y is the event that one of the constant-term constraints $f' \in \mathcal{F}'$ is not satisfied by (some) row blocks of $\bar{\boldsymbol{S}}$; and
- Z is the event that (some) rows of $\bar{\boldsymbol{S}}$ do not satisfy the norm bound, i.e. $\|\bar{\boldsymbol{s}}_i^{(\ell)}\| > \sqrt{337/30}\mathtt{B}_i$, for some $i \in [K]$ and $\ell \in [n]$.

If X happens, then by Lemma 5, we know that $\bar{\boldsymbol{S}}$ does not satisfy the aggregation of the constraints f and f'' with random polynomial challenges $\mu_i', \nu_i' \leftarrow \mathcal{R}_q$, except with probability at most $1/q^{d/2}$. This aggregation actually leads to the expression of $F^{(\ell)}$ from Equation (43). Therefore, when re-running Step 4, the extractor gets a new matrix $\bar{\boldsymbol{S}}'$, thanks to $\mathcal{E}_{\texttt{row-block-1}}$ (unless with probability $\kappa_{\texttt{row-block-1}}$) satisfying relation $R'_{\texttt{row-block-1}}$ and in particular this aggregation (i.e. Equation (43)), and clearly being different from $\bar{\boldsymbol{S}}$, and thus leading to a SIS solution.

If Y happens and the liftings $b_k''^{(\ell)} \in \mathcal{R}_q$ are correctly formed, then by the same reasoning, the earlier aggregation is satisfied by $\bar{\boldsymbol{S}}$ with probability at most $1/q^{d/2}$, over the randomness of $\mu_i', \nu_i' \leftarrow \mathcal{R}_q$, by Lemma 5. If Y happens but the liftings are not correctly formed, then the checks on their constant terms pass with probability at most $(1/q)^\lambda$ over the randomness of $(\psi_i'^{(k)}, (\omega_i'^{(k)}) \leftarrow \mathbb{Z}_q$, as there are λ liftings. Again, when re-running Step 4, if $\mathcal{E}_{\texttt{row-block-1}}$ does not abort, he extracts a new matrix $\bar{\boldsymbol{S}}'$ satisfying this constant-term check from an accepting transcript and clearly being different from $\bar{\boldsymbol{S}}$, and thus leading to a SIS solution.

If Z happens, then in the case when the corresponding projection of the long extracted vector is correctly formed, the verifier's norm check on this projection holds with probability at most $K/2^{128}$ over the randomness of Π' by Theorem 5 (and union bound) and the assumption on the bounds $\mathtt{B}_i$. Otherwise, the prover can cheat with probability at most $(1/q)^\lambda + 1/q^{d/2}$, as before. As before, when re-running Step 4, the $\mathcal{E}_{\texttt{row-block-1}}$ extracts a new matrix $\bar{\boldsymbol{S}}'$ satisfying in particular the norm check from an accepting transcript and clearly being different from $\bar{\boldsymbol{S}}$, and thus leading to a SIS solution.

The Final Knowledge Error. Note that our extractor succeeds if it outputs a witness for $R'_{\texttt{row-block-many}}$ in Step 3 or it outputs the SIS solution in Step 5. As our extractor gets the matrix $\bar{\boldsymbol{S}}$ with probability at least $\epsilon - \kappa_{\texttt{row-block-1}}$, then, according to the cases above, the second matrix $\bar{\boldsymbol{S}}'$ could coincide with the first one with probability at most $\max\{1/q^{d/2}, 1/q^{d/2} + (1/q)^\lambda, 1/q^{d/2} + (1/q)^\lambda + K/2^{128}\}$ and we conclude that the success probability of our extractor should be at least $\epsilon - \kappa_{\texttt{row-block-1}} - 1/q^{d/2} - (1/q)^\lambda - K/2^{128}$. Thus, its knowledge soundness is $\kappa_{\texttt{row-block-many}} \coloneqq 1/q^{d/2} + (1/q)^\lambda + K/2^{128} + \kappa_{\texttt{row-block-1}}$.

4.3 Orthus

In this section, we describe our main protocol Orthus for proving that each "block" of a witness matrix satisfies the same set of LaBRADOR dot-product constraints, along with norm bounds.

More precisely, the witness matrix $\boldsymbol{S} \in \mathcal{R}^{nK \times mN}$ is partitioned into n row blocks of K consecutive rows, which are further partitioned into m groups of N consecutive columns, called blocks. Thus, we can think of $\boldsymbol{S}$ as a collection of nm witness blocks, each of dimensions $K \times N$. We consider a scenario where the prover wants to prove knowledge of these nm witness blocks, whose rows we denote by $\tilde{\boldsymbol{u}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{u}}_K^{(\ell)} \in \mathcal{R}^N$, that satisfy the same LaBRADOR constraints (14) collected separately, as full constraints $f \in \mathcal{F}$ and constant-term ones $f' \in \mathcal{F}'$, and norm bounds $\mathtt{B}_i'$. These constraints are no longer affected by a public matrix $\boldsymbol{D}$, as opposed to the ones considered in Sections 4.1 and 4.2, but instead they are exactly like the ones from LaBRADOR, described in Section 2.2. In other words, our main protocol targets a batched version of LaBRADOR's relation, described by the following relation

$$R_{\mathtt{0}}(\gamma) = \left\{ ((\mathcal{F}, \mathcal{F}', (\mathtt{B}_i')_i), \boldsymbol{S}) \middle| \begin{array}{c} f(\tilde{\boldsymbol{u}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{u}}_K^{(\ell)}) = 0,\ f \in \mathcal{F} \\ \mathtt{ct}(f'(\tilde{\boldsymbol{u}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{u}}_K^{(\ell)})) = 0,\ f' \in \mathcal{F}' \\ \|\tilde{\boldsymbol{u}}_1^{(\ell)}\| \leq \gamma \mathtt{B}_1', \ldots, \|\tilde{\boldsymbol{u}}_K^{(\ell)}\| \leq \gamma \mathtt{B}_K' \end{array} \right\}.$$

We provide an interactive proof for this relation. While for completeness the norm constraints need to hold for $\gamma = 1$, the protocol we describe only proves knowledge of a solution with norm bounds multiplied by a slack factor of $\gamma = \sqrt{337m/30}$. We note that there are standard techniques to extend the protocol to support proving exact norm bounds (see Appendix D). Here, we provide a high-level description of Orthus, and a full algorithmic description of the protocol can be found in Figure 6. Our proof first extends the blocks with additional rows, in order to convert the constant-term constraints into full ones. Then we reduce Orthus to proving row block constraints as in the relation $R_{\texttt{row-block-many}}$, by aggregating the constraints per blocks with random challenges from the ring, and then make use of the protocol from Section 4.2.

Adding Auxiliary Rows. In order to linearly combine the blocks and their constraints more easily, we first convert the constant-term constraints $f' \in \mathcal{F}$ into full constraints, by extending each block with additional rows made by their evaluations $f'(\tilde{\boldsymbol{u}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{u}}_K^{(\ell)})$ and possibly padding with 0's. [9] Here we assume

[9] We could do this just for the quadratic constant-term constraints, i.e. the ones with some $a_{ij}' \neq 0$. For the linear ones, we can simply rewrite them in terms of the whole row of the row block, see Equation (48).

for simplicity that $|\mathcal{F}'| \leq N$, so we only need to add one auxiliary row per block, denoted by $\tilde{\boldsymbol{u}}_{K+1}^{(\ell)}$. After constructing these extra rows, we send Ajtai commitments to the now extended columns of $\boldsymbol{S}$. Each constraint $f' \in \mathcal{F}'$ is thus reformulated as a full constraint $\bar{f}' \in \mathcal{F}$, along with a constant-term one, only linear in the additional row:

$$\bar{f}'(\tilde{\boldsymbol{u}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{u}}_{K+1}^{(\ell)}) \triangleq \sum_{i,j\in[K]} a'_{ij} \langle \tilde{\boldsymbol{u}}_i^{(\ell)}, \tilde{\boldsymbol{u}}_j^{(\ell)} \rangle + \sum_{i\in[K+1]} \langle \boldsymbol{\varphi}'_i, \tilde{\boldsymbol{u}}_i^{(\ell)} \rangle - b' = 0 \tag{44}$$

$$\mathtt{ct}(\langle \boldsymbol{\varphi}'_{K+1}, \tilde{\boldsymbol{u}}_{K+1}^{(\ell)} \rangle) = 0, \tag{45}$$

where $\boldsymbol{\varphi}'_{K+1}$ is a unit vector whose entries are all 0, except for a single -1 entry in the right position. Let $\bar{\mathcal{F}}$ denote the expanded set of full constraints to prove. From now on, we denote the extended matrix by $\boldsymbol{S}$ and its Ajtai commitment by $\boldsymbol{T} = \boldsymbol{A}\boldsymbol{S}$, for a matrix $\boldsymbol{A} \in \mathcal{R}_q^{t \times nK}$. We note that we do not need to prove any norm bound on the additional row $\tilde{\boldsymbol{u}}_{K+1}^{(\ell)} \in \mathcal{R}_q^N$.

Reducing Orthus to Proving Row Block Constraints. To reduce to row block constraints, first note that each group of m consecutive blocks, of indices $\ell'+1, \ldots, \ell'+m$, for $\ell' \in \{0, m, \ldots, (n-1)m\}$, make a row block, made by $K+1$ consecutive rows of $\boldsymbol{S}$ of index $\ell \coloneqq \ell'/m + 1$,

$$\begin{pmatrix} \tilde{\boldsymbol{s}}_1^{(\ell)} \\ \vdots \\ \tilde{\boldsymbol{s}}_{K+1}^{(\ell)} \end{pmatrix} \triangleq \begin{pmatrix} \tilde{\boldsymbol{u}}_1^{(\ell'+1)} & \ldots & \tilde{\boldsymbol{u}}_1^{(\ell'+m)} \\ \vdots & \ddots & \vdots \\ \tilde{\boldsymbol{u}}_{K+1}^{(\ell'+1)} & \ldots & \tilde{\boldsymbol{u}}_{K+1}^{(\ell'+m)} \end{pmatrix} \in \mathcal{R}_q^{(K+1)\times mN}.$$

Instead of individually proving the constraints for m consecutive blocks, one could just linearly combine the constraints with random polynomial challenges $\alpha_1, \ldots, \alpha_m$ from the verifier to prove, in one shot, a linear combination of the constraints satisfied by all the m blocks, and thus, by their whole row block. Namely, as any full constraint f satisfied by a block could be written as in Equation (44) (if $f \in \bar{\mathcal{F}}$, $\boldsymbol{\varphi}_{K+1} = \boldsymbol{0}$ and if $f = \bar{f}' \in \bar{\mathcal{F}}$, $\boldsymbol{\varphi}_{K+1}$ is a unit vector), for any $\ell' \in \{0, m, \ldots, (n-1)m\}$, one could prove:

$$\sum_{k\in[m]} \alpha_k \sum_{i,j\in[K]} a_{ij} \langle \tilde{\boldsymbol{u}}_i^{(\ell'+k)}, \tilde{\boldsymbol{u}}_j^{(\ell'+k)} \rangle + \sum_{k\in[m]} \alpha_k \sum_{i\in[K+1]} \langle \boldsymbol{\varphi}'_i, \tilde{\boldsymbol{u}}_i^{(\ell'+k)} \rangle - \sum_{k\in[m]} \alpha_k b' = 0, \tag{46}$$

which is the same as

$$\sum_{i,j\in[K]} a_{ij} \sum_{k\in[m]} \langle \alpha_k \tilde{\boldsymbol{u}}_i^{(\ell'+k)}, \tilde{\boldsymbol{u}}_j^{(\ell'+k)} \rangle + \sum_{i\in[K+1]} \sum_{k\in[m]} \langle \alpha_k \boldsymbol{\varphi}'_i, \tilde{\boldsymbol{u}}_i^{(\ell'+k)} \rangle - \sum_{k\in[m]} \alpha_k b' = 0.$$

Setting $\boldsymbol{D} \coloneqq \mathrm{diag}(\alpha_1 \boldsymbol{I}_N, \ldots, \alpha_m \boldsymbol{I}_N) \in \mathcal{R}_q^{mN \times mN}$, $\bar{\boldsymbol{\varphi}}_i \coloneqq [\alpha_1 \boldsymbol{\varphi}'_i, \ldots, \alpha_K \boldsymbol{\varphi}'_i] \in \mathcal{R}_q^{mN}$ and $\bar{b} \coloneqq \sum_{k\in[m]} \alpha_k b' \in \mathcal{R}_q$, one could then rewrite Equation (46) as

$$\sum_{i,j\in[K]} a_{ij} \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \sum_{i\in[K+1]} \langle \bar{\boldsymbol{\varphi}}_i, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle - \bar{b} = 0, \ \ell \in [n], \tag{47}$$

which looks like the constraints satisfied by row blocks in the protocol in Section 4.2, where the matrix $\boldsymbol{D}$ is defined in terms of the challenges α_i. Let f_α be the aggregated constraint obtained, satisfying Equation (47) and $\bar{\mathcal{F}}_{\boldsymbol{\alpha}}$ be the set of the f_α's. Further, Equation (45) satisfied by each of the m blocks could be written as m constant-term constraints satisfied by the whole row $\tilde{\boldsymbol{s}}_{K+1}^{(\ell)} = [\tilde{\boldsymbol{u}}_{K+1}^{(\ell'+1)}, \ldots, \tilde{\boldsymbol{u}}_{K+1}^{(\ell'+m)}]$, by expanding the vector $\boldsymbol{\varphi}'_{K+1} \in \mathcal{R}_q^N$ to one from $\mathcal{R}_q^{mN}$, by placing it on any i-th position, $i \in [m]$, and adding 0's elsewhere:

$$\mathtt{ct}(\langle [0, \ldots 0, \boldsymbol{\varphi}'_{K+1}, 0 \ldots, 0], [\tilde{\boldsymbol{u}}_{K+1}^{(\ell'+1)}, \ldots, \tilde{\boldsymbol{u}}_{K+1}^{(\ell'+m)}] \rangle) = \mathtt{ct}(\langle \boldsymbol{\varphi}'_{K+1}, \tilde{\boldsymbol{u}}_{K+1}^{(\ell'+i)} \rangle). \tag{48}$$

For consistency, let $\bar{\mathcal{F}}'_{\boldsymbol{\alpha}}$ be the set of such constant term constraints. Moreover, due to the triangle inequality, we could consider (approximate) norm bounds on each $\tilde{\boldsymbol{s}}_i^{(\ell)}$, $i \in [K]$, keeping in mind that this is made by m

rows of blocks, all of norms less than $\mathtt{B}'_i$. Then, one could finish, by using the protocol from Section 4.2, to prove that for each row block, of index $\ell \in [n]$ and of rows $\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_{K+1}^{(\ell)}$, satisfies the constraints from $\bar{\mathcal{F}}_{\boldsymbol{\alpha}}$ and $\bar{\mathcal{F}}'_{\boldsymbol{\alpha}}$ and the norm bounds $\|\tilde{\boldsymbol{s}}_i^{(\ell)}\| \leq \mathtt{B}'_i\sqrt{m}$, for any $i \in [K]$. We do not give any approximate norm proof for $\tilde{\boldsymbol{s}}_{K+1}^{(\ell)}$. Although the norms of the columns $\boldsymbol{s}_i$ of the extended witness matrix $\boldsymbol{S}$ are not public, the verifier can still compute the bound B from Section 2.1 by using the public norm bounds on the rows, as this bound is a function of $\sum_{i\in[mN]} \|\boldsymbol{s}_i\|^2 = \sum_{\ell\in[n],i\in[K+1]} \|\tilde{\boldsymbol{s}}_i^{(\ell)}\|^2$. We note that the bound on the evaluations from the additional row can be computed based on the public coefficients from the equations f' (Equation (44)) and the bounds on the witness. Whenever this bound is too large, the prover would actually decompose these liftings into an appropriate base. The bound can then be computed based on the decomposition base, and the public $\boldsymbol{\varphi}'_{K+1}$ from Equation (45) would need to be adjusted with the appropriate gadget vector.

Efficiency. Due to the call to the protocol from Section 4.2, the communication complexity is $O(nK^2+mN)$ and the verifier runtime is $O(nK^2 + mNK)$.

Security Analysis. The completeness follows easily as Eqs. (46) and (48) hold, due to Equation (47), and from the completeness of the protocol from Section 4.2. Here we only state the knowledge soundness.

Theorem 4. *Let $\hat{\mathcal{C}} = C = \mathcal{R}_q$ and $\mathcal{C} \subseteq \mathcal{R}_q$ be the challenge space from Section 2.1 of polynomials of operator norm T and ℓ_2 norm τ. Further, suppose that $\mathtt{B}'_i \leq \sqrt{30/(337m)}q/125$, for any $i \in [K]$. Suppose that M-SIS$_{t,8TB}$ is hard, where B is defined as in Section 2.1. Then the protocol in Section 4.3 is a knowledge-sound proof for the relation $R_{\mathtt{0}}(\sqrt{337m/30})$ with knowledge error $\kappa_{\texttt{block}} \coloneqq \frac{1}{q^{d/2}} + \kappa_{\texttt{row-block-many}}$.*

Proof. Let $\mathcal{A}$ be an adversary for the protocol in Figure 4 and suppose that it convinces the verifier $\mathcal{V}$ to accept with probability ϵ. We describe a knowledge extractor $\mathcal{E}_{\mathtt{0}}$ for the protocol in Figure 6. On input public parameters $\boldsymbol{A}$, instance $((B_i), \mathcal{F}, \mathcal{F}', (\mathtt{B}'_i))$, the extractor $\mathcal{E}_{\mathtt{0}}$ proceeds as follows.

1. The extractor samples $(\alpha_i)_i \leftarrow \mathcal{R}_q$, computes $\boldsymbol{D} \coloneqq \mathrm{diag}(\alpha_1\boldsymbol{I}_N, \ldots, \alpha_m\boldsymbol{I}_N) \in \mathcal{R}_q^{mN\times mN}$, the norm bound B as explained above, and the families of constraints $\bar{\mathcal{F}}_\alpha$ as described by (46),(47) and $\bar{\mathcal{F}'}_\alpha$ as described by (48), and runs $\mathcal{E}_{\texttt{row-block-many}}$ (from the proof of Theorem 3) on input $\boldsymbol{A}$ and instance $(\boldsymbol{T}, B, \bar{\mathcal{F}}_\alpha, \bar{\mathcal{F}'}_\alpha, (\mathtt{B}'_i\sqrt{m}), \boldsymbol{D})$ which then calls the malicious prover $\mathcal{P}^*_{\texttt{row-block-many}}((\alpha_i))$, producing $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}}) \in \mathcal{R}_q^{nK\times mN} \times \bar{\mathcal{C}}^{mN}$.
2. If the tuple $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ is not a valid witness for the relation $R'_{\texttt{row-block-many}}$, then the extractor aborts.
3. The extractor checks whether $(B, \mathcal{F}, \mathcal{F}', (\mathtt{B}'_i), (\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})) \in R_{\mathtt{0}}(\sqrt{337m/30})$ and if so, outputs $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ and terminates.
4. If the extractor succeeds at Step 2, but it does not produce a valid witness at Step 3, the extractor samples $(\alpha'_i)_i \leftarrow \mathcal{R}_q$, and proceeds as in Step 1 (which then calls the malicious prover $\mathcal{P}^*_{\texttt{row-block-many}}((\alpha'_i))$), until it outputs a new tuple $(\bar{\boldsymbol{S}}', \bar{\boldsymbol{c}}')$ witness for the relation $R'_{\texttt{row-block-many}}$.
5. If $\bar{\boldsymbol{S}} = \bar{\boldsymbol{S}}'$, the extractor aborts. Else, the extractor outputs a Module-SIS solution and terminates.

Following the proof of Theorem 1, the extractor is correct and runs in expected time double as $\mathcal{E}_{\texttt{row-block-many}}$, i.e. $32Tm$.

Success Probability of the Extractor. Whenever $\mathcal{E}_{\texttt{row-block-many}}$, and hence $\mathcal{E}_{\mathtt{0}}$ do not abort at Step 1, the output $\bar{\boldsymbol{S}} \in \mathcal{R}_q^{nK\times mN}$ made by n row blocks of rows $\bar{\boldsymbol{s}}_1^{(\ell)}, \ldots, \bar{\boldsymbol{s}}_K^{(\ell)}$, for $\ell \in [n]$, satisfies the relation $R_{\texttt{row-block-many}}$, i.e. the constraints from $\bar{\mathcal{F}}_{\boldsymbol{\alpha}}$ and $\bar{\mathcal{F}}'_{\boldsymbol{\alpha}}$ and the norm bounds on the rows of each row block being less than $\sqrt{337/30}\mathtt{B}'_i\sqrt{m}$. If $(\bar{\boldsymbol{S}}, \bar{\boldsymbol{c}})$ produced at Step 1 does not satisfy the relation $R_{\mathtt{0}}(\sqrt{337m/30})$ it must be the case that either one of the following events X and Y holds, where:

- X is the event when one of the constraints $f \in \mathcal{F}$ is not satisfied by (some of) the mn blocks that make the extracted matrix $\bar{\boldsymbol{S}}$;

– Y is the event when one of the constant-term constraints $f' \in \mathcal{F}'$ is not satisfied by (some of) the mn blocks of the same matrix.

Note that, from the extraction, the norm bounds on the rows of the blocks are already satisfied, i.e. they are less than $\sqrt{337m/30}\mathsf{B}'_i$.

If X happens, i.e. if some $f \in \mathcal{F}$ is not satisfied by some block, then a malicious prover could make its linear aggregation $f_\alpha \in \bar{\mathcal{F}}_{\boldsymbol{\alpha}}$ with other $m-1$ constraints with $\alpha'_i \in \mathcal{R}_q$'s be satisfied by $\bar{\boldsymbol{S}}$ with probability at most $1/q^{d/2}$, by Lemma 5, over uniformly random $\alpha'_i \leftarrow \mathcal{R}_q$. (See Equations (46) and (47)). As Equation (47) is part of the instantiation of the protocol from Section 4.2, when re-running the prover at Step 4, the extractor gets a new matrix $\bar{\boldsymbol{S}}'$, thanks to $\mathcal{E}_{\texttt{row-block-many}}$ (unless it aborts with probability $\kappa_{\texttt{row-block-many}}$), satisfying $R'_{\texttt{row-block-many}}$ and in particular this equation, and being different from $\bar{\boldsymbol{S}}$, which leads to a SIS solution.

If Y happens, i.e. if some $f' \in \mathcal{F}'$ is not satisfied by some block, then it must be the case that the malicious prover crafted its evaluation in the additional row, as this row satisfies the constant-term check, Equation (48) in the protocol from Section 4.2. Therefore, this block cannot satisfy the corresponding constraint $\bar{f}' \in \bar{\mathcal{F}}$ and as before, the prover could make its aggregation $\bar{f}'_\alpha \in \bar{\mathcal{F}}_{\boldsymbol{\alpha}}$ with new α'_i's happen with probability at most $1/q^{d/2}$ for uniformly random $\alpha'_i \leftarrow \mathcal{R}_q$, by Lemma 5. As before, when re-running Step 4, the extractor obtains a different matrix $\bar{\boldsymbol{S}}'$, leading in the end to a SIS solution.

The Final Knowledge Error. Note that our extractor succeeds if it outputs a witness for $R'_{\texttt{row-block-many}}$ in Step 3 or it outputs the SIS solution in Step 5. As our extractor gets the matrix $\bar{\boldsymbol{S}}$ with probability at least $\epsilon - \kappa_{\texttt{row-block-many}}$, and according to the discussion above, the second matrix $\bar{\boldsymbol{S}}'$ could coincide with the first one with probability at most $1/q^{d/2}$, we conclude that the success probability of our extractor should be at least $\epsilon - \kappa_{\texttt{row-block-many}} - 1/q^{d/2}$. Thus, its knowledge soundness is $\kappa_0 := 1/q^{d/2} + \kappa_{\texttt{row-block-many}}$.

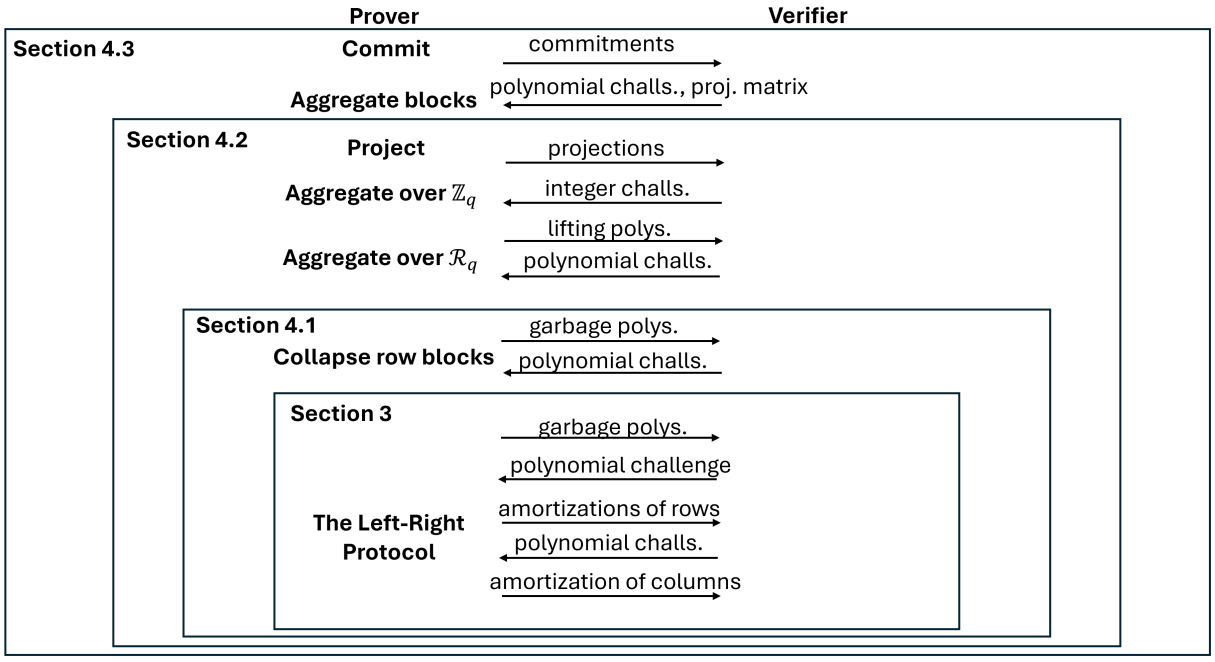

Fig. 1. Orthus

5 Reducing the Communication Cost

The natural approach to reduce the communication cost of the main protocol described in Section 4.3 involves two steps. First, all the prover's messages are moved to the last round, while in the previous rounds the prover sends instead a compressed commitment of the respective messages. This means that the verifier also has to check that the messages sent in the last round are indeed openings to the commitments sent throughout the protocol. Second, instead of *sending* the last message and having the verifier check all the relations, the prover *proves knowledge* of the last message using a succinct proof system like LaBRADOR, i.e., the last message becomes the witness to LaBRADOR. As a consequence, the communication cost we obtain is the size of the compressed commitments plus the size of the succinct LaBRADOR proof. In this section we describe the optimizations we introduce when applying those transformations, as well as some challenges we overcome.

5.1 Two-Step Commitment

Committing to the prover messages is a standard technique used in protocols like LaBRADOR to reduce the proof size. The norm of the to-be-committed messages needs to be small if we want to use Ajtai commitments and we achieve this by decomposing the messages. The decomposition base is chosen so that it is small enough for security, but also not too small, as otherwise the resulting vectors (the witness for LaBRADOR) will have a large dimension. LaBRADOR can then be used to prove that the ℓ_2 norm of the decomposed vectors is indeed small, allowing us to set the rank of the commitments according to the expected hardness of M-SIS for the respective bounds and dimension. The resulting commitments are then much smaller than the messages we started with. We refer to these commitments as *middle* commitments, to differentiate them from the *inner* commitments computed in the first round of Orthus.

We go a step further and commit to the middle commitments, obtaining *outer* commitments. The decomposed middle commitments then become part of the output witness. Since the middle commitments are already quite small, we can decompose them in binary without causing a significant increase to the dimension of the output witness. The advantage of this additional layer is that, if we prove that those decomposed middle commitments are indeed binary, then an even smaller rank for the outer commitment suffices to rely on the hardness of M-SIS for infinity norm bound 1 (as the difference of the two binary vectors that an attacker would have to find to break the binding property of the commitment scheme is a ternary vector). Following the estimates of the lattice estimator [4] for 128 bits of soundness for a 38-bit modulus, we note that it suffices to set the outer commitment matrix to have only 64 rows instead of 256 when committing to middle commitments, so the size of each outer commitment would consist of only 64 integers. When it comes to the projections, we decompose them directly in binary and commit them with a 256-row Ajtai matrix. [10]

Adapting the Johnson-Lindenstrauss Lemma. The projections are no longer sent in the clear, so the verifier cannot directly check their ℓ_2 norm. Proving exact ℓ_2 norms for each individual projection with LaBRADOR would be relatively expensive because each of them would have to be a separate witness vector. Instead, we consider LaBRADOR proving that the projections are binary with a certain number of bits. From this we can derive a bound on the infinity norm of the projections. However, the Johnson-Lindenstrauss lemma (Theorem 5) is tailored to an ℓ_2 norm check on the projections. We overcome this issue by adapting the proof to show that as long as the infinity norm of the projection mod q is smaller than a bound, the witness has a small ℓ_2 norm bound with high probability (see Theorem 6). More precisely, in combination with Lemma 6, we can prove approximate norm bounds on the witness with a slack factor of $9.75/0.49 \approx 19.89$. For reference, the slack obtained when proving exact ℓ_2 norms on the projections is $\sqrt{337/30} \approx 3.37$. We believe that Theorem 6 can be of independent interest.

A Step Towards Zero-Knowledge. We note that the outer commitments can be upgraded so that they do not leak any information about the messages that they are a commitment of. This works by sampling

[10] An earlier version of this paper claimed that 32 integers in the output was sufficient for all cases. This claim has been revised, increasing the output to 64 integers when committing to middle commitments, and 256 integers when committing to projections. This revision was prompted by Quang Dao's report; see [18] for more details.

a uniformly random binary vector and concatenating it to the message before committing. If the length of this vector is large enough, the outer commitment is computationally indistinguishable from random based on the M-LWE assumption. One would then have to feed the output of Orthus to a zero-knowledge proof system like LNP [29] to obtain zero-knowledge for the composition.

5.2 Proving Knowledge of Orthus Output with LaBRADOR

The last message of the compressed version of Orthus, which includes all the prover's messages of the non-compressed version from Section 4.3, becomes the witness of LaBRADOR, and the verifier's checks of Orthus become the statement to be proven with LaBRADOR. We now describe how to construct a LaBRADOR statement from the output of Orthus.

Rephrasing Checks. The checks performed by the Orthus verifier (Equations (13), (23), (35) and (55)) are standard linear lattice relations that can be proven with LaBRADOR. Since the vectors in the witness are in decomposed form, those checks are simply rephrased with a gadget matrix that recomposes the vectors. The remaining check, Equation (21), requires extending the witness with additional vectors.

Adding New Witness Vectors and Checks. Equation (21) considers the inner product $\langle \tilde{\boldsymbol{z}}'_+, \tilde{\boldsymbol{z}}_- \rangle$, where $\tilde{\boldsymbol{z}}'_+ \coloneqq \tilde{\boldsymbol{z}}_+ \cdot \boldsymbol{D}$ for $\boldsymbol{D} = \mathrm{diag}(\alpha_1 \boldsymbol{I}_N, \ldots, \alpha_m \boldsymbol{I}_N) \in \mathcal{R}_q^{mN \times mN}$, as instantiated by the protocol in Section 4.3. To make this constraint compatible with LaBRADOR, we need to include $\tilde{\boldsymbol{z}}'_+$ as part of the witness, as well as the following linear relations that ensure its correctness:

$$\tilde{\boldsymbol{z}}'_+ = \alpha_j (\tilde{\boldsymbol{z}}_+)_j, \ j \in [m], \tag{49}$$

where $\tilde{\boldsymbol{z}}'_+, \tilde{\boldsymbol{z}}_+$ are each split in n equal parts, $(\tilde{\boldsymbol{z}}'_+)_j, (\tilde{\boldsymbol{z}}_+)_j$.

Additionally, the two-step commitment compression technique requires more witness vectors and checks. In particular, the witness needs to be extended with the binary decomposition of the middle commitments. This way, the correctness of the middle and outer commitments can be encoded as linear relations going into LaBRADOR. We group the decompositions of the middle commitments and of the projections into a vector $\tilde{\boldsymbol{x}}$ for which we need to prove that it is binary. We do so following the standard approach of extending the witness with the automorphism $\sigma_{-1}(\tilde{\boldsymbol{x}})$ and adding the appropriate quadratic and linear relations. See Appendix D for more details.

Overall, the LaBRADOR witness becomes

$$((b_i''^{(\ell)}), (\boldsymbol{g}_{ij}^{(\ell)}), (\boldsymbol{h}_{ij}^{(\ell)}), (v_i), \tilde{\boldsymbol{z}}_+, \tilde{\boldsymbol{z}}_-, \tilde{\boldsymbol{z}}'_+, \boldsymbol{z}, \tilde{\boldsymbol{x}}, \sigma_{-1}(\tilde{\boldsymbol{x}})),$$

satisfying Equations (13), (21), (23), (35), (49), (55), (59) and (61), where the vectors are decomposed into appropriate bases. The decomposition bases are chosen to be the largest possible, provided that the coefficients are small enough for M-SIS to be hard. In this way, we keep the dimension of the witness under control, as well as the number of individual vectors in the witness, which determine the runtime efficiency of LaBRADOR. Finally, we also impose ℓ_2 norm bounds on the vectors, which we derive from the number of coefficients and their size (given by the choice of decomposition bases). See Appendix E for more details.

Note on Knowledge Soundness. The knowledge soundness proof for the compressed version of Orthus follows the same lines as for the simplified version (Theorem 4), except that the compressed version proves norm bounds approximately up to a new slack factor of $9.75/0.49 \cdot \sqrt{m}$, coming from Theorem 6, and relies on the hardness of additional M-SIS problems to ensure the binding property of the middle and outer commitments. Finally, the composition result from [11, Lem.3.7] can be used to ensure the knowledge soundness of the compressed Orthus with LaBRADOR as the underlying proof system.

6 Application: Aggregating Falcon signatures

The Falcon signature scheme [37] is an instantiation of the GPV framework [22] for lattice-based signature schemes over the NTRU [24] structured lattices. It works with a power-of-two cyclotomic ring

$\mathcal{R}' = \mathbb{Z}[X]/(X^{d_F}+1)$ of ring degree d_F, modulo q_F. To sign some target $t \in \mathcal{R}'_{q_F} \coloneqq \mathbb{Z}_{q_F}[X]/(X^{d_F}+1)$, the signer uses the secret key together with a trapdoor preimage sampler to sample a pair $(s_1, s_2) \in \mathcal{R}'^2$ from a Discrete Gaussian distribution such that $s_1 + hs_2 = t \bmod q_F$ and $\|(s_1, s_2)\| \leq \beta$, for a public key $h \in \mathcal{R}'_{q_F}$ generated according to the NTRU distribution.

The standard Falcon signing procedure computes the target $t \in \mathcal{R}'_{q_F}$ as a hash of the message and some salt, which is a fresh random value sampled during signing. We deviate from this procedure by considering a deterministic version of Falcon where the salt is a fixed constant. Provided that each message is signed at most once, the deterministic variant avoids the repeated-target sampling that the randomized salt in Falcon is designed to prevent. The advantage of this approach is that when different users sign the same message, they will use the same salt and arrive at the same target t. We note that this deterministic version has already been considered for practical deployments, e.g., by Algorand [38].

We work with the Falcon-512 parameters, i.e., $d_F = 512$, $q_F = 12289$ and $\lfloor \beta^2 \rfloor = 34034726$. In order to set the witness and statement for Orthus from the Falcon signatures, there are some challenges we overcome.

Change of modulus. We lift the Falcon verification equation to $\mathcal{R}'$. More precisely, we prove, for a lifting polynomial $v \in \mathcal{R}'$,

$$s_1 + hs_2 + q_F v = t \in \mathcal{R}'. \tag{50}$$

Additionally, we prove that the norm of v is small enough to ensure there is no wrap-around when proving the equation modulo q.

Change of ring. As Orthus works with $\mathcal{R} = \mathbb{Z}[X]/(X^d+1)$, of $d = 256$, we reformulate Equation (50) into one compatible with Orthus constraint system. As in [30, Sec. 2.8], we consider the embedding of $\mathcal{R}$ into the Falcon ring $\mathcal{R}'$ via $X \to X^2$ which allows the norm-preserving representation of any $f \in \mathcal{R}'$ as a tuple $(f^{(e)}, f^{(o)}) \in \mathcal{R}^2$ via their linear combination with respect to the basis $\{1, X\}$:

$$f(X) = f^{(e)}(X^2) + Xf^{(o)}(X^2).$$

Therefore, using that representation for s_1, s_2, t, h and v we can express Equation (50) as a system of two equations over $\mathcal{R}$:

$$\begin{aligned} s_1^{(e)} + h^{(e)}s_2^{(e)} + Xh^{(o)}s_2^{(o)} + q_F v^{(e)} &= t^{(e)}, \\ s_1^{(o)} + h^{(o)}s_2^{(e)} + h^{(e)}s_2^{(o)} + q_F v^{(o)} &= t^{(o)}. \end{aligned} \tag{51}$$

We consider nm Falcon signatures, (s_1, s_2), along with their corresponding public keys h, each signing to the same public target t. Therefore, we can represent a Falcon signature as a 4×4 block of ring elements in $\mathcal{R}_q$:

$$\begin{pmatrix} s_1^{(e)} & s_1^{(o)} & s_2^{(e)} & s_2^{(o)} \\ 1 & 0 & h^{(e)} & Xh^{(o)} \\ 0 & 1 & h^{(o)} & h^{(e)} \\ v^{(e)} & v^{(o)} & 0 & 0 \end{pmatrix} \in \mathcal{R}_q^{4\times 4},$$

and thus construct a $4n \times 4m$ witness matrix for Orthus.

We assume that there is a preprocessing procedure that computes commitments to the (rows of) public keys, and both prover and verifier receive these commitments as input.

The input constraints for Orthus are derived from Equation (51) proven mod q. Additionally, we prove the correctness of the preprocessed commitments. We also provide an approximate norm proof for the last row (i.e. for the liftings) and an exact norm proof for the first row (i.e. for the signatures). More details can be found in Appendix F.

7 Implementation

We have implemented Orthus with the optimizations from Section 5 starting from the polynomial arithmetic of the C implementation of LaBRADOR. We have improved and extended that arithmetic, and rewritten the

implementation of LaBRADOR in a generic way that makes it easier to compose with other protocols, while including further optimizations. This composability allows us to effectively feed the output of Orthus into LaBRADOR, and we have taken particular care in preparing an instance that the latter can efficiently prove. We support general Orthus statements, and concretely test the implementation on the aggregate signature instantiation from Section 6. The results can be found in Table 1. The implementation can be accessed at:

https://github.com/lazer-crypto/labrador

7.1 Orthus Statements

The interface to the Orthus proof system we implement works as follows. The user can introduce the LaBRADOR-like dot-products constraints that should hold for a block of vectors. Additionally, the user can indicate for each vector whether it requires a binary proof, an exact ℓ_2 norm proof or whether it suffices to prove an approximate ℓ_2 norm with the slack coming from the projections. Besides, there is the option of denoting some vectors as preprocessed, so that both the prover and verifier will receive commitments to them as input.

Having described the statement for a block, the user indicates how many blocks to prove. The proof system automatically decides how to arrange those blocks into a matrix such that the output witness for LaBRADOR is as small as possible. We implement the verifier so that it leverages this structured nature of a statement that repeats for many blocks, resulting in a very fast aggregation of the constraints, even when dealing with a large number of blocks (below 1 second for 2^{16} blocks, see Table 1).

A challenge that arises when supporting exact ℓ_2 norms in the statement is that we first need to prove using random projections that the norm is sufficiently small, so that there is no wrap-around in the quadratic constraint that proves the norm exactly (see Appendix D). In theory, one can increase the modulus as much as needed for the guarantee provided by the projections of rows to be good enough. In practice, we work with a bounded-size modulus (on 38 bits). We resolve this limitation by computing as many projections as needed in different parts of each row, so that the projections guarantee a tight enough norm for each vector in the row.

7.2 The Column NTT

The computation of the garbage terms v_i from Section 3 involves the product between column vectors represented as polynomials in $\mathcal{R}_q[Y]$ (see Equations (18) and (19)). To this end, we have implemented cyclic NTTs $\mathcal{R}_p[Y]/(Y^n-1) \mapsto R_p^n$ for the 16-bit moduli p that are also used in the multimodular arithmetic implementation of LaBRADOR. LaBRADOR's arithmetic implementation for $\mathcal{R}_q$ works by lifting to $\mathcal{R}$, then using NTT-based operations modulo several 16-bit p, and finally applying the explicit CRT modulo q. Since the cyclic Y^n-1 has as roots all the roots of unity of order n in $\mathbb{Z}_p$, and not just the primitive roots of the cyclotomic Y^n+1, our implementation can work with n up to $2d=512$. But since we cannot have any wrap-around modulo Y^n-1 when multiplying column polynomials, we only multiply polynomials of degree less than 256 and hence our implementation supports up to 256 rows in the witness matrices. Alternatively to the roots of unity in $\mathbb{Z}_p$ we could also use the powers of the generator X of $\mathcal{R}=\mathbb{Z}[X]/(X^d+1)$. So here we would compute the radix-2 splitting

$$\mathcal{R}_p[Y]/(Y^n-1) = \mathcal{R}_p[Y]/(Y^n-X^{2d}) \mapsto \mathcal{R}_p[Y]/(Y^{n/2}-X^d) \times \mathcal{R}_p[Y]/(Y^{n/2}+X^d)$$

in the cyclic NTT, and continue recursively. This would result in a rotating NTT where all computed products are just multiplications by powers of X, i.e. rotations in the coefficient representation of $\mathcal{R}_p$. We implemented this rotating NTT but are not using it in our Orthus code. The reason is that then we would need to compute cyclotomic NTTs after the cyclic NTT; that is we would need to compute the chain

$$\mathcal{R}_p[Y]/(Y^n-1) \mapsto \mathcal{R}_p^n = \left(\mathbb{Z}_p[X]/(X^d+1)\right)^n \mapsto (\mathbb{Z}_p)^{dn}.$$

Instead, we compute the two NTTs in the other order

$$\mathcal{R}_p[Y]/(Y^n-1) \mapsto (\mathbb{Z}_p^d)[Y]/(Y^n-1) \mapsto (\mathbb{Z}_p)^{dn}.$$

In this way we can reuse the cyclotomic NTTs that we also need for the $\mathcal{R}_q$-arithmetic. So, in other words, our cyclic NTT for the column products operates on the cyclotomic NTT representation of the witness vectors. It is a vectorized implementation that computes 32 interleaved NTTs in parallel and achieves a very high speed of about 122000 cycles for 256 length-512 NTTs on a single core of the Intel Willow Cove microarchitecture implemented in a 2021 Tigerlake mobile CPU. This amounts to less than 500 cycles per length-512 NTT. On the other hand it is only very slightly faster than our non-interleaved cyclotomic NTT would be when extrapolated to length 512.

7.3 LaBRADOR

Orthus performs a large reduction in the size of the input witness while enjoying a fast verification time. We further reduce the size of the witness by feeding the output of Orthus into LaBRADOR as described in Section 5.2. To this end, we rewrite the implementation of LaBRADOR so that it can be more easily composed with other protocols. We do this by defining a set of quadratic and linear constraints that can efficiently encode the verifier's checks of different protocols, and providing a unified set of functions that can deal with their aggregation. In particular, the verifier's checks of Orthus can be encoded with these constraints and LaBRADOR is tailored to proving them. We also make sure that the number of individual vectors in the output of Orthus is small enough, as the runtime of LaBRADOR increases quadratically with the number of input vectors.

Additionally, we extend LaBRADOR to use the two-step commitments described in Section 5.1.[11] One important benefit of this technique is that the projections are no longer sent in the clear. This means that we can afford to compute projections for individual vectors and reuse the same projection matrix, rather than doing a single projection of the full witness. As a consequence, the projection matrix that the verifier needs to sample and aggregate is significantly smaller, and hence the verifier runtime is improved.

Another difference is that we run LaBRADOR on 256-degree rings, as opposed to the original 64-degree implementation. The reason for this is to keep it consistent with the rings used in Orthus, where the larger degree gives the best performance due to the column NTT from Section 7.2. This larger degree means that the elements outputted in the last round of LaBRADOR are larger, contributing to the larger proof sizes of our implementation. Concretely, the proof size we obtain for all the sizes of aggregate signatures from Table 1 is around $110-115$KB, compared to the ~ 75KB from LaZer [33]. We note that if one is concerned about this increase in proof size, one could switch to a 64-degree implementation in the last rounds of the protocol.

Acknowledgments. We would like to thank Beatrice Biasioli for discussions on the Johnson-Lindenstrauss lemma (Theorem 6). We would also like to thank Quang Dao and Alberto Centelles for pointing out issues [16, 18] in an earlier version of this paper regarding the size of the outer commitments and the Johnson-Lindenstrauss lemma respectively. We thank Ward Beullens for suggesting a fix for the proof of the aforementioned lemma (Theorem 6). We also thank the anonymous CRYPTO 2026 reviewers for their useful comments. This work was supported by the EU H2020 ERC Project 101002845 PLAZA and the SNSF Grant PretZel (10008425).

References

1. Aardal, M.A., Aranha, D.F., Boudgoust, K., Kolby, S., Takahashi, A.: Aggregating falcon signatures with LaBRADOR. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part I. LNCS, vol. 14920, pp. 71–106. Springer, Cham (Aug 2024). https://doi.org/10.1007/978-3-031-68376-3_3
2. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: 28th ACM STOC. pp. 99–108. ACM Press (May 1996). https://doi.org/10.1145/237814.237838

[11] We call this variant LaBRADOS.

3. Ajtai, M.: Generating hard instances of the short basis problem. In: Wiedermann, J., van Emde Boas, P., Nielsen, M. (eds.) ICALP 99. LNCS, vol. 1644, pp. 1–9. Springer, Berlin, Heidelberg (Jul 1999). https://doi.org/10.1007/3-540-48523-6_1
4. Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. Journal of Mathematical Cryptology **9**(3), 169–203 (Oct 2015). https://doi.org/10.1515/jmc-2015-0016
5. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange - A new hope. In: Holz, T., Savage, S. (eds.) USENIX Security 2016. pp. 327–343. USENIX Association (Aug 2016), https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/alkim
6. Arnon, G., Chiesa, A., Fenzi, G., Yogev, E.: STIR: reed-solomon proximity testing with fewer queries. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part X. pp. 380–413. LNCS, Springer (2024), https://doi.org/10.1007/978-3-031-68403-6_12
7. Arnon, G., Chiesa, A., Fenzi, G., Yogev, E.: WHIR: reed-solomon proximity testing with super-fast verification. In: Fehr, S., Fouque, P. (eds.) EUROCRYPT 2025, Part IV. pp. 214–243. LNCS, Springer (2025), https://doi.org/10.1007/978-3-031-91134-7_8
8. Attema, T., Cramer, R., Kohl, L.: A compressed Σ-protocol theory for lattices. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part II. LNCS, vol. 12826, pp. 549–579. Springer, Cham, Virtual Event (Aug 2021). https://doi.org/10.1007/978-3-030-84245-1_19
9. Baum, C., Bootle, J., Cerulli, A., del Pino, R., Groth, J., Lyubashevsky, V.: Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 669–699. Springer, Cham (Aug 2018). https://doi.org/10.1007/978-3-319-96881-0_23
10. Baum, C., Damgård, I., Lyubashevsky, V., Oechsner, S., Peikert, C.: More efficient commitments from structured lattice assumptions. In: Catalano, D., De Prisco, R. (eds.) SCN 18. LNCS, vol. 11035, pp. 368–385. Springer, Cham (Sep 2018). https://doi.org/10.1007/978-3-319-98113-0_20
11. Beullens, W., Seiler, G.: LaBRADOR: Compact proofs for R1CS from module-SIS. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part V. LNCS, vol. 14085, pp. 518–548. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38554-4_17
12. Bhadauria, R., Fang, Z., Hazay, C., Venkitasubramaniam, M., Xie, T., Zhang, Y.: Ligero++: A new optimized sublinear IOP. In: Ligatti, J., Ou, X., Katz, J., Vigna, G. (eds.) CCS 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual Event, USA. pp. 2025–2038. ACM (2020), https://doi.org/10.1145/3372297.3417893
13. Boneh, D., Chen, B.: Latticefold: A lattice-based folding scheme and its applications to succinct proof systems. In: Hanaoka, G., Yang, B. (eds.) ASIACRYPT 2025, Part III. pp. 330–362. LNCS, Springer (2025), https://doi.org/10.1007/978-981-95-5099-9_11
14. Boneh, D., Chen, B.: Latticefold+: Faster, simpler, shorter lattice-based folding for succinct proof systems. In: Kalai, Y.T., Kamara, S.F. (eds.) CRYPTO 2025, Part VII. pp. 327–361. LNCS, Springer (2025), https://doi.org/10.1007/978-3-032-01907-3_11
15. Brehm, M., Chen, B., Fisch, B., Resch, N., Rothblum, R.D., Zeilberger, H.: Blaze: Fast snarks from interleaved RAA codes. In: Fehr, S., Fouque, P. (eds.) EUROCRYPT 2025, Part IV. pp. 123–152. LNCS, Springer (2025), https://doi.org/10.1007/978-3-031-91134-7_5
16. Centelles, A.: Theorem 6 (modular l-infinity Johnson–Lindenstrauss) is false as stated: explicit counterexample accepted at 2^{-49.3}. https://github.com/lazer-crypto/labrador/issues/2 (Aug 2026), GitHub issue, opened August 20, 2026
17. Cini, V., Lai, R.W.F., Malavolta, G.: Lattice-based succinct arguments from vanishing polynomials - (extended abstract). In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part II. LNCS, vol. 14082, pp. 72–105. Springer, Cham (Aug 2023). https://doi.org/10.1007/978-3-031-38545-2_3
18. Dao, Q.: Security: outer commitment sizing omits the input width. https://github.com/lazer-crypto/labrador/issues/1 (Aug 2026), GitHub issue, opened August 17, 2026
19. Dodis, Y., Shrimpton, T. (eds.): CRYPTO 2022, Part II, LNCS, vol. 13508. Springer, Cham (Aug 2022)
20. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-Dilithium: A lattice-based digital signature scheme. IACR TCHES **2018**(1), 238–268 (2018). https://doi.org/10.13154/tches.v2018.i1.238-268, https://tches.iacr.org/index.php/TCHES/article/view/839
21. Gentry, C., Halevi, S., Lyubashevsky, V.: Practical non-interactive publicly verifiable secret sharing with thousands of parties. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part I. LNCS, vol. 13275, pp. 458–487. Springer, Cham (May / Jun 2022). https://doi.org/10.1007/978-3-031-06944-4_16
22. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008). https://doi.org/10.1145/1374376.1374407

23. Groth, J.: Linear algebra with sub-linear zero-knowledge arguments. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 192–208. Springer, Berlin, Heidelberg (Aug 2009). https://doi.org/10.1007/978-3-642-03356-8_12
24. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: Third Algorithmic Number Theory Symposium (ANTS). Lecture Notes in Computer Science, vol. 1423, pp. 267–288. Springer (June 1998). https://doi.org/10.1007/BFb0054868
25. Klooß, M., Lai, R.W.F., Nguyen, N.K., Osadnik, M.: RoK, paper, SISsors toolkit for lattice-based succinct arguments - (extended abstract). In: Chung, K.M., Sasaki, Y. (eds.) ASIACRYPT 2024, Part V. LNCS, vol. 15488, pp. 203–235. Springer, Singapore (Dec 2024). https://doi.org/10.1007/978-981-96-0935-2_7
26. Klooß, M., Lai, R.W.F., Nguyen, N.K., Osadnik, M.: Rok and roll - verifier-efficient random projection for $o\tilde{}(\lambda)$-size lattice arguments - (extended abstract). In: Hanaoka, G., Yang, B. (eds.) ASIACRYPT 2025, Part III. pp. 297–329. LNCS, Springer (2025), https://doi.org/10.1007/978-981-95-5099-9_10
27. Langlois, A., Stehlé, D.: Worst-case to average-case reductions for module lattices. DCC **75**(3), 565–599 (2015). https://doi.org/10.1007/s10623-014-9938-4
28. Lyubashevsky, V.: Lattice signatures without trapdoors. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 738–755. Springer, Berlin, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_43
29. Lyubashevsky, V., Nguyen, N.K., Plançon, M.: Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In: Dodis and Shrimpton [19], pp. 71–101. https://doi.org/10.1007/978-3-031-15979-4_3
30. Lyubashevsky, V., Nguyen, N.K., Plançon, M., Seiler, G.: Shorter lattice-based group signatures via "almost free" encryption and other optimizations. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part IV. LNCS, vol. 13093, pp. 218–248. Springer, Cham (Dec 2021). https://doi.org/10.1007/978-3-030-92068-5_8
31. Lyubashevsky, V., Nguyen, N.K., Seiler, G.: Shorter lattice-based zero-knowledge proofs via one-time commitments. In: Garay, J. (ed.) PKC 2021, Part I. LNCS, vol. 12710, pp. 215–241. Springer, Cham (May 2021). https://doi.org/10.1007/978-3-030-75245-3_9
32. Lyubashevsky, V., Seiler, G.: Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 204–224. Springer, Cham (Apr / May 2018). https://doi.org/10.1007/978-3-319-78381-9_8
33. Lyubashevsky, V., Seiler, G., Steuer, P.: The LaZer library: Lattice-based zero knowledge and succinct proofs for quantum-safe privacy. In: Luo, B., Liao, X., Xu, J., Kirda, E., Lie, D. (eds.) ACM CCS 2024. pp. 3125–3137. ACM Press (Oct 2024). https://doi.org/10.1145/3658644.3690330
34. Nevado, D., Kim, D., Stopar, M.: Lattice-based signature aggregation (2025), https://ethresear.ch/t/lattice-based-signature-aggregation/22282
35. Nguyen, N.K., Seiler, G.: Practical sublinear proofs for R1CS from lattices. In: Dodis and Shrimpton [19], pp. 133–162. https://doi.org/10.1007/978-3-031-15979-4_5
36. Nguyen, N.K., Seiler, G.: Greyhound: Fast polynomial commitments from lattices. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part X. LNCS, vol. 14929, pp. 243–275. Springer, Cham (Aug 2024). https://doi.org/10.1007/978-3-031-68403-6_8
37. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: FALCON. Tech. rep., National Institute of Standards and Technology (2017), available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-1-submissions
38. Young, L.: Technical brief: Quantum-resistant transactions on algorand with falcon signatures (2025), https://algorand.co/blog/technical-brief-quantum-resistant-transactions-on-algorand-with-falcon-signatures
39. Zeilberger, H., Chen, B., Fisch, B.: Basefold: Efficient field-agnostic polynomial commitment schemes from foldable codes. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part X. pp. 138–169. LNCS, Springer (2024), https://doi.org/10.1007/978-3-031-68403-6_5

Appendix

A Deferred preliminaries

Given $d \in \mathbb{N}$ which is a power of 2, and an odd prime modulus $q \in \mathbb{N}$, let $\mathcal{R} \coloneqq \frac{\mathbb{Z}[X]}{\langle X^d+1\rangle}$, and $\mathcal{R}_q \coloneqq \frac{\mathbb{Z}_q[X]}{\langle X^d+1\rangle}$. In this paper we work with q and d such that $X^d + 1$ splits in two irreducible factors mod q.

For any polynomial $a \in \mathcal{R}$, we denote its coefficient vector by $\mathtt{coef}(a) \in \mathbb{Z}^d$. For $a = \sum_i a_i X^i$, we denote by $\mathtt{ct}(a) \coloneqq a_0$ the constant term of a. We will consider the norm of elements in $\mathcal{R}$ to be $\|a\|_p = |a|$ if $a \in \mathbb{Z}$ and $\|a\|_p = \sqrt[p]{\sum a_i^p}$ if $a = \sum a_i X^i \in \mathcal{R}$, for $p > 1$. We extend the notation to vectors $\|\boldsymbol{a}\|_p = \sqrt{\sum_i \|\boldsymbol{a}_i\|_p^2}$. We remove the subscript of $\|\cdot\|_p$ when $p = 2$. In the quotient ring, the norm of an element in $\mathcal{R}_q$ is the norm of its unique representative $\mathcal{R}$ with coefficients in $[-\frac{q-1}{2}, \frac{q-1}{2}]$.

In power-of-two cyclotomic rings, working over subrings $\mathcal{S}$ of $\mathcal{R}$ is possible, as detailed out in [30, Section 2.8.]. Namely, if $\mathcal{S} = \mathbb{Z}[X]/(X^{d'} + 1)$ a subring of $\mathcal{R}$, for $d'|d$, then, there is a norm-preserving bijection $\Phi : \mathcal{R} \to \mathcal{S}^{d/d'}$, using the embedding of $\mathcal{S}$ in $\mathcal{R}$ via $X \to X^{d/d'}$. This bijection can be naturally extended to vectors over $\mathcal{S}$ and $\mathcal{R}$ and over their quotients mod q.

We recall the result of [32] which says that short elements in $\mathcal{R}_q$ are invertible.

Lemma 4 (Cor.1.2, [32]). *Let $q \equiv 5 \mod 8$ be a prime. Then, any $f \in \mathcal{R}_q$ which satisfies either $0 \leq \|f\|_\infty < \frac{1}{2}q^{1/2}$ or $0 < \|f\| < q^{1/2}$ is invertible in $\mathcal{R}_q$.*

Gadget matrices. Let $\mathsf{b}, r \in \mathbb{N}$. We define the gadget vector as $\mathbf{g}_\mathsf{b}^T \coloneqq \begin{pmatrix}1 & \mathsf{b} & \mathsf{b}^2 & \dots & \mathsf{b}^{\mathsf{t}-1}\end{pmatrix}$ where $\mathsf{t} \coloneqq \lceil \log_\mathsf{b} q \rceil$. Then we define the gadget matrix $\boldsymbol{G}_{\mathsf{b},r} = \mathbf{I}_r \otimes \mathbf{g}_\mathsf{b}^T$. We also define its inverse function $\boldsymbol{G}_{\mathsf{b},r}^{-1} : \mathcal{R}_q^{r\times m} \to \mathcal{R}_q^{\mathsf{t}r\times m}$, which decomposes each entry with respect to base b. Therefore, for any $\boldsymbol{t} \in \mathcal{R}^r$, we have

$$\boldsymbol{G}_{\mathsf{b},r}\boldsymbol{G}_{\mathsf{b},r}^{-1}(\boldsymbol{t}) = \boldsymbol{t} \text{ and } \|\boldsymbol{G}_{\mathsf{b},r}^{-1}(\boldsymbol{t})\|_\infty \leq \frac{\mathsf{b}}{2}.$$

The automorphism $\boldsymbol{\sigma_{-1}}$. The map $\sigma_{-1} \colon \mathcal{R}_q \to \mathcal{R}_q$ which sends $X \mapsto X^{-1}$ defines an automorphism of $\mathcal{R}_q$. We extend the definition of σ_{-1} to vectors and matrices over $\mathcal{R}_q$ by applying σ_{-1} entry-wise. We will use the observation made in [29] that for power-of-two cyclotomic rings, σ_{-1} relates the inner products over $\mathcal{R}_q$ with inner products of their coefficient vectors over $\mathbb{Z}_q$, namely for any $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{R}_q^n$,

$$\langle \mathtt{coef}(\boldsymbol{a}), \mathtt{coef}(\boldsymbol{b})\rangle = \mathtt{ct}(\langle \sigma_{-1}(\boldsymbol{a}), \boldsymbol{b}\rangle).$$

In particular, $\|\boldsymbol{a}\|^2 = \mathtt{ct}(\langle \sigma_{-1}(\boldsymbol{a}), \boldsymbol{a}\rangle)$. Moreover, we can recover any k-th coefficient of any polynomial $p = \sum_{i=0}^{d-1} p_i \in \mathcal{R}$ as $p_k = \mathtt{ct}(\sigma_{-1}(X^k)p)$, since multiplication by X^{-k} shifts via a cyclic rotation the k-th coefficient to the first position.

The Schwartz-Zippel lemma. We state (a consequence of) the Schwartz-Zippel lemma, used in proving equality of polynomials.

Lemma 5. *Let $p \in \mathcal{R}_q[X_1, \dots, X_n]$ be a nonzero polynomial of total degree m over $\mathcal{R}_q$. Assume $\mathcal{R}_q$ splits as a product of two subfields, each of size $q^{d/2}$. Let $r_1, \dots, r_n$ be sampled uniformly from $\mathcal{R}_q$. Then*

$$\Pr[p(r_1, \dots, r_n) = 0] \leq \frac{m}{q^{d/2}}.$$

A.1 The Modular Johnson-Lindenstrauss Lemma

Proving knowledge of long vectors $\mathbf{w} \in \mathbb{Z}^d$ with small ℓ_2 norm, $\|\mathbf{w}\| \leq b$, can be achieved by applying random linear projections to reduce dimensionality [11, 21, 29]. More precisely, the verifier samples a random linear map $\boldsymbol{\Pi} : \mathbb{Z}^d \to \mathbb{Z}^{256}$ with independent entries sampled from $\mathtt{Bin}_1$ distribution, i.e. these entries take values equal to -1, 0, or 1 with probabilities 1/4, 1/2 and 1/4 respectively. Then, the prover sends $\boldsymbol{\Pi}\mathbf{w}$ and proves it is correct, but mod q. Notice that the average of $\|\boldsymbol{\Pi}\mathbf{w}\|$ is $\sqrt{128}\|\mathbf{w}\|$.

The technical result behind this is called the Johnson Lindenstrauss lemma, introduced in [21], which intuitively says that this projection preserves the ℓ_2 norm up to some slack factor:

Lemma 6 (Cor.3.2, [21]). *For every vector* $\mathbf{w} \in \mathbb{Z}^d$,

$$\Pr_{\pi \leftarrow Bin_1^d}[|\langle \pi, \mathbf{w} \rangle| > 9.75\|\mathbf{w}\|] < 2^{-141},$$

$$\Pr_{\mathbf{\Pi} \leftarrow Bin_1^{256 \times d}}[\|\mathbf{\Pi w}\|_\infty > 9.75\|\mathbf{w}\|] < 2^{-133},$$

$$\Pr_{\mathbf{\Pi} \leftarrow Bin_1^{256 \times d}}[\|\mathbf{\Pi w}\| > \sqrt{337}\|\mathbf{w}\|] < 2^{-128}.$$

The prover can then send $\mathbf{\Pi w}$ and prove that it is computed correctly. [21, Cor.3.3] says that if the verifier checks that $\|\mathbf{\Pi w}\| \leq b\sqrt{30}$, on the condition $b < q/(45d)$, then the verifier is convinced that $|\mathbf{w}\| \leq b$ with high probability, in spite of a potential reduction mod q when computing the projection. [11, Le.4.2] strengthened this result, for a better bound on b independent of the degree of the ring d. This is stated for security parameter $\lambda = 128$. A later work [1] generalized the lemma for arbitrary λ and projection matrices from $\mathtt{Bin}_1$ distribution of 2λ rows.

Theorem 5 (Lem.4.2, [11]). *For every vector* $\mathbf{w} \in [\pm q/2]^d$ *with* $\|\mathbf{w}\| \geq b$, *for some bound* $b \leq q/125$, *we have*

$$\Pr_{\mathbf{\Pi} \leftarrow Bin_1^{256 \times d}}[\|\mathbf{\Pi w} \ mod\ q\| < \sqrt{30}b] < 2^{-128}.$$

If the verifier checks that $\|\mathbf{\Pi w}\| \leq \sqrt{337}\beta$, for β upper bounding $\|\mathbf{w}\|$, then he is convinced that $\|\mathbf{w}\| \leq \sqrt{337/30}\beta$ with high probability. Therefore we can give an approximate norm proof with *slack factor* of $\sqrt{337/30} \approx 3.37$.

In Section 5 we consider commitments of the decompositions of the projections. When opening, the verifier reads their infinity norm from the length of their decomposition, i.e. $\|\mathbf{\Pi w} \mod q\|_\infty$, so he needs to be convinced about an upper bound on $\|\mathbf{w}\|$. The following lemma is a variant of Johnson Lindenstrauss lemma, adapted for the infinity norm. We defer its proof to Appendix B.4.

Theorem 6. *For every vector* $\mathbf{w} \in [\pm q/2]^d$ *with* $\|\mathbf{w}\| \geq b$, *for some bound* $b \leq q/61$, *we have*

$$\Pr_{\mathbf{\Pi} \leftarrow Bin_1^{256 \times d}}[\|\mathbf{\Pi w} \ mod\ q\|_\infty < 0.49b] < 2^{-128}.$$

If the verifier checks that $\|\mathbf{\Pi w}\|_\infty \leq 9.75\beta$, for β upper bounding $\|\mathbf{w}\|$ then he is convinced that $\|\mathbf{w}\| \leq 9.75\beta/0.49$ with high probability. Therefore, we can give an approximate norm proof with a slack factor of $9.75/0.49 \approx 19.89$. For comparison, if one were to rely on an ℓ_2-norm Johnson Lindenstrauss result, one could use $\|\mathbf{x}\| \leq \sqrt{256} \cdot \|\mathbf{x}\|_\infty$, for any $\mathbf{x} \in \mathbb{Z}^{256}$ to derive an ℓ_2 norm bound on the projection, namely $\|\mathbf{\Pi w}\| \leq \sqrt{256} \cdot \|\mathbf{\Pi w}\|_\infty \leq \sqrt{256} \cdot 9.75\|\mathbf{w}\| \leq \sqrt{256} \cdot 9.75\beta$, with high probability, using Lemma 6. Using [11, Lem.4.2.], this would lead to a slack factor of $\sqrt{256} \cdot 9.75/\sqrt{30} \approx 28.48$.

A.2 Lattice Assumptions

Here we recall the computational lattice problems Module-SIS (M-SIS) and Module-LWE (M-LWE). [20,27].

Definition 1 (Module-SIS). *The* ***Module Short Integer Solution*** *(M-SIS) problem with parameters* λ, $N = N(\lambda)$, $M = M(\lambda)$, $q = q(\lambda)$ *and* $\beta = \beta(\lambda) < q$ *all natural numbers, asks the adversary* $\mathcal{A}$ *to find* $\boldsymbol{v} \in \mathcal{R}_q^M$ *such that* $\boldsymbol{A}\boldsymbol{v} = 0$ *over* $\mathcal{R}_q$ *and* $0 < \|\boldsymbol{v}\| \leq \beta$. *We denote the advantage of an adversary* $\mathcal{A}$ *in solving M-SIS*$_{N,M,\beta}$

$$Adv^{SIS}_{N,M,\beta}(\mathcal{A}) \coloneqq \Pr\left[\begin{array}{c} \boldsymbol{v} \in \mathcal{R}_q^M \setminus \{0\} \\ \wedge\, \boldsymbol{A}\boldsymbol{v} = 0 \wedge \|\boldsymbol{v}\| \leq \beta \end{array} \middle| \begin{array}{c} \boldsymbol{A} \leftarrow \mathcal{R}_q^{N \times M} \\ \boldsymbol{v} \leftarrow \mathcal{A}(\boldsymbol{A}) \end{array}\right].$$

We say the problem M-SIS$_{N,M,\beta}$ *is hard if for all PPT algorithms* $\mathcal{A}$, $Adv^{SIS}_{N,M,\beta}(\mathcal{A})$ *is negligible in* λ.

Definition 2 (Module-LWE). *The **Module-LWE** (M-LWE) problem with parameters λ, $M' = M'(\lambda)$, $N' = N'(\lambda)$, $q = q(\lambda)$ natural numbers and χ a distribution over $\mathcal{R}$, asks the adversary $\mathcal{A}$ to distinguish between the two distributions*

- $D_0 \coloneqq \{(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e}) : \boldsymbol{A} \leftarrow \mathcal{R}_q^{M' \times N'}, \boldsymbol{s} \leftarrow \chi^{N'}, \boldsymbol{e} \leftarrow \chi^{M'}\}$
- $D_1 \coloneqq \{(\boldsymbol{A}, \boldsymbol{b}) : \boldsymbol{A} \leftarrow \mathcal{R}_q^{M' \times N'}, \boldsymbol{b} \leftarrow \mathcal{R}_q^{M'}\}$.

We denote the advantage of an adversary $\mathcal{A}$ in solving M-LWE$_{M',N',\chi}$ as

$$Adv^{MLWE}_{M',N',\boldsymbol{e}}(\mathcal{A}) \coloneqq \left| \Pr\left[b = 1 \middle| \begin{matrix} (\boldsymbol{A}, \boldsymbol{u}) \leftarrow D_0, \\ b \leftarrow \mathcal{A}(\boldsymbol{A}, \boldsymbol{u}) \end{matrix} \right] - \Pr\left[b = 1 \middle| \begin{matrix} (\boldsymbol{A}, \boldsymbol{u}) \leftarrow D_1, \\ b \leftarrow \mathcal{A}(\boldsymbol{A}, \boldsymbol{u}) \end{matrix} \right] \right|.$$

We say the problem M-LWE$_{M',N',\chi}$ is hard if for all PPT algorithms $\mathcal{A}$, the advantage $Adv^{MLWE}_{M',N',\chi}(\mathcal{A})$ is negligible in λ.

A.3 Commitment Schemes

A commitment scheme allows a sender to create commitments to secret values which he decides to reveal later. Such a scheme satisfies *hiding* and *binding* properties. Hiding assures that the commitments do not leak information about the committed values, whereas binding that the given commitment cannot open to a different value than the one committed.

Definition 3. *A commitment scheme $\mathtt{Com}$ is a tuple of algorithms $(\mathtt{Gen}, \mathtt{Com}, \mathtt{Vfy})$ defined as follows:*

- $\mathtt{Gen}()$*: a PPT algorithm that generates public parameters $pp = (S_M, S_R, S_C, \chi)$ describing a message space S_M, a randomness space S_R, a commitment space S_C and an efficiently sampleable probability distribution χ over S_C.*
- $\mathtt{Com}(pp, m, r)$*: a PPT algorithm that given public parameters pp, a message $m \in S_M$ and randomness $r \leftarrow \chi$, it produces a commitment $c \in S_C$.*
- $\mathtt{Vfy}(pp, c, m, r)$*: a deterministic PT algorithm that given public parameters pp, commitment $c \in S_C$, message $m \in S_M$, randomness $r \in S_R$, it produces a bit $b \in \{0, 1\}$, which is equal to 1 if "accept", or 0 else.*

For simplicity, we will drop pp when working with the commitment scheme algorithms, but keep in mind they always receive as input pp.

Definition 4 (Correctness). *A commitment scheme $(\mathtt{Gen}, \mathtt{Com}, \mathtt{Vfy})$ is correct if for any algorithm $\mathcal{A}$,*

$$\Pr[\mathtt{Vfy}(c, m, r) = 1 | pp \leftarrow \mathtt{Gen}()\ m \leftarrow \mathcal{A}(pp)\ c \leftarrow \mathtt{Com}(pp, m, r)] = 1.$$

Definition 5 (Hiding). *A commitment scheme $(\mathtt{Gen}, \mathtt{Com}, \mathtt{Vfy})$ is computationally hiding if for any PPT algorithm $\mathcal{A}$,*

$$\left| \Pr\left[b = b' \middle| \begin{matrix} pp \leftarrow \mathtt{Gen}()\ m_0, m_1 \leftarrow \mathcal{A}(pp)\ b \leftarrow \{0,1\} \\ c \leftarrow \mathtt{Com}(pp, m_b, r), b' \leftarrow \mathcal{A}(c) \end{matrix} \right] - \frac{1}{2} \right| \approx 0.$$

Definition 6 (Binding). *A commitment scheme $(\mathtt{Gen}, \mathtt{Com}, \mathtt{Vfy})$ is computationally binding if for any PPT algorithm $\mathcal{A}$,*

$$\Pr\left[\begin{matrix} m_0 \neq m_1 \text{ and} \\ \mathtt{Vfy}(c, m_0, r_0) = \mathtt{Vfy}(c, m_1, r_1) = 1 \end{matrix} \middle| \begin{matrix} pp \leftarrow \mathtt{Gen}() \\ (m_0, m_1, c, r_0, r_1) \leftarrow \mathcal{A}(pp) \end{matrix} \right] \approx 0.$$

We describe the following commitment scheme already implicit in Ajtai's seminal work [3]. Its hiding property is based on the hardness of Module-LWE, whereas its binding property on the hardness of Module-SIS.

Definition 7 (Ajtai commitments). *The Ajtai commitment scheme is defined as $(\mathtt{Gen}^*, \mathtt{Com}^*, \mathtt{Vfy}^*)$, where*

– $\texttt{Gen}^*()$: *it outputs pp. Set* $B \coloneqq \sqrt{dM + dN'}$. *Define* (S_M, S_C, S_R), *where*

$$S_M = \{\boldsymbol{m} \in \mathcal{R}_q^M | \texttt{coef}(\boldsymbol{m}) \in \{0,1\}^{dM}\}, S_R = \{\boldsymbol{r} \in \mathcal{R}_q^{N'} | \texttt{coef}(\boldsymbol{r}) \in \{0,1\}^{dN'}\}),$$

$$\chi = \mathit{Unif}(\{\boldsymbol{r} \in \mathcal{R}_q^{N'} | \texttt{coef}(\boldsymbol{r}) \in \{0,1\}^{dN'}\}), S_C = \mathcal{R}_q^N.$$

Sample $\boldsymbol{A}_1 \in \mathcal{R}_q^{N \times M}, \boldsymbol{A}_2 \in \mathcal{R}_q^{N \times N'}$. *Output* $pp = (N, M, B, \boldsymbol{A}_1, \boldsymbol{A}_2)$.

– $\texttt{Com}^*(pp, \boldsymbol{m}, \boldsymbol{r})$: *output* $\boldsymbol{t} \coloneqq \boldsymbol{A}_1\boldsymbol{m} + \boldsymbol{A}_2\boldsymbol{r}$.

– $\texttt{Vfy}^*(\boldsymbol{t}, \boldsymbol{m}, \boldsymbol{r})$: *check if*

$$\|(\boldsymbol{m}, \boldsymbol{r})\| \leq B \text{ and } \boldsymbol{t} = \boldsymbol{A}_1\boldsymbol{m} + \boldsymbol{A}_2\boldsymbol{r}.$$

If one is not interested in the hiding property, the random vector $\boldsymbol{r}$ can be omitted.

A.4 Interactive Arguments

Definition 8. *A* ***relation*** $R(pp)$ *parametrized by* $pp \in \{0,1\}^*$ *is a set of tuples* $(\mathbb{x}, \mathbb{w})$ *where* $\mathbb{x} \in \{0,1\}^*$ *is the instance and* $\mathbb{w} \in \{0,1\}^*$ *the witness.*

An *interactive argument of knowledge* for a relation $R(pp)$ is a triple $\mathsf{ARG} = (\mathcal{G}, \mathcal{P}, \mathcal{V})$ of probabilistic polynomial time algorithms, where the *generator* algorithm $\mathcal{G}$ produces public parameters pp for the argument, and the prover $\mathcal{P}$ and $\mathcal{V}$ are stateful and interactive. The notation "$b \xleftarrow{tr} \langle \mathcal{P}(pp, \mathbb{x}, \mathbb{w}), \mathcal{V}(pp, \mathbb{x}) \rangle$" denotes an interaction between $\mathcal{P}$ and $\mathcal{V}$, producing a transcript tr of prover and verifier messages, where the verifier's final output is b. We say the protocol between $\mathcal{P}$ and $\mathcal{V}$ is public-coin if the verifier's challenges are uniformly sampled, independently of the prover's messages.

The argument ARG satisfies the following completeness and knowledge soundness properties.

– **Completeness.** ARG is *complete* with completeness error $\epsilon \colon \mathbb{N} \times \mathbb{N} \to [0,1)$ if for all $\lambda, n \in \mathbb{N}$, and all adversaries $\mathcal{A}$,

$$\Pr\left[\begin{array}{c|r} (\mathbb{x}, \mathbb{w}) \in R(pp) \text{ and } & pp \leftarrow \mathcal{G}(1^\lambda, n) \\ b = 0 & (\mathbb{x}, \mathbb{w}) \leftarrow \mathcal{A}(pp) \\ & b \leftarrow \langle \mathcal{P}(pp, \mathbb{x}, \mathbb{w}), \mathcal{V}(pp, \mathbb{x}) \rangle \end{array}\right] \leq \epsilon(\lambda, n),$$

where b is the verifier's output at the end of this interaction.

– **Knowledge soundness.** ARG has *witness extended emulation* with knowledge error $\kappa \colon \mathbb{N} \times \mathbb{N} \to [0,1)$ if there exists an expected probabilistic polynomial time extractor $\mathcal{E}$ such that for all $\lambda, n \in \mathbb{N}$, and all probabilistic polynomial time adversaries $\mathcal{A}$,

$$\left|\Pr\left[\mathcal{A}(st, tr) = 1 \,\middle|\, \begin{array}{r} pp \leftarrow \mathcal{G}(1^\lambda) \\ (\mathbb{x}, st) \leftarrow \mathcal{A}(pp) \\ tr \leftarrow \langle \mathcal{A}(st), \mathcal{V}(pp, \mathbb{x}) \rangle \end{array}\right]\right.$$
$$\left. - \Pr\left[\begin{array}{c|r} \mathcal{A}(st, tr) = 1 & pp \leftarrow \mathcal{G}(1^\lambda, n) \\ \text{and} & (\mathbb{x}, st) \leftarrow \mathcal{A}(pp) \\ \text{if } tr \text{ is accepting then } (\mathbb{x}, \mathbb{w}) \in R & (tr, \mathbb{w}) \leftarrow \mathcal{E}^{\mathcal{A}(\cdot, st)}(pp, \mathbb{x}) \end{array}\right]\right| \leq \kappa(\lambda, n),$$

where the extractor $\mathcal{E}$ has black-box oracle access to (the next-message functions of) $\mathcal{A}$.

B Deferred Proofs

B.1 Proof of Lemma 1

Proof. As in [11], we assume that the columns $\boldsymbol{s}_j \in \mathcal{R}_q^r$ have $\mathbb{Z}_q$ coefficients following independent Gaussian distributions of standard deviation $\sigma_j \coloneqq \frac{B_j}{\sqrt{rd}}$. Notice that $\boldsymbol{z}$ has r polynomial entries and each entry of index $i \in [r]$ is a sum of the form $\boldsymbol{c}_1\boldsymbol{S}_{i1} + \ldots + \boldsymbol{c}_m\boldsymbol{S}_{im} \in \mathcal{R}_q$. Each $\mathbb{Z}_q$ coefficient of it is a sum of m $\mathbb{Z}_q$

coefficients of the respective index, each coming from a term $\boldsymbol{c}_j\boldsymbol{S}_{ij}$. Each $\mathbb{Z}_q$ coefficient of one term $\boldsymbol{c}_j\boldsymbol{S}_{ij}$ is a sum of d products obtained when multiplying all $\mathbb{Z}_q$ coefficients of $\boldsymbol{S}_{ij}$ by a challenge $\mathbb{Z}_q$ coefficient of $\boldsymbol{c}_j$. Each such product follows a distribution of mean 0 and standard deviation $|x|\,\sigma_j$, depending on the nonzero challenge $\mathbb{Z}_q$ coefficient x, so then the distribution of any $\mathbb{Z}_q$ coefficient of $\boldsymbol{c}_j\boldsymbol{S}_{ij}$ has variance $\tau^2 \cdot \sigma_j^2$, where $\tau = \|c_j\|$. Therefore, the distribution of each $\mathbb{Z}_q$ coefficient of $\boldsymbol{c}_1\boldsymbol{S}_{i1} + \ldots + \boldsymbol{c}_m\boldsymbol{S}_{im}$, call it γ_m, follows a centered distribution of standard deviation τ_m, where $\tau_m^2 = \tau^2 \sum_{j\in[m]} \sigma_j^2$. By the central limit theorem, the standardization $\frac{\gamma_m}{\tau_m}$ converges to the standard normal distribution for $m \to \infty$. The same happens also for the random variable γ'_m distributed according to the discrete Gaussian distribution D_{σ_m} with the same standard deviation as γ_m. Therefore, for any $x \in \mathbb{Z}$

$$\lim_{m\to\infty} \Pr[\gamma_m \leq x\sigma_m] - \Pr[\gamma'_m \leq x\sigma_m] = 0,$$

so we can model the distribution of the $\mathbb{Z}_q$ coefficients of each polynomial entry of $\boldsymbol{z}$ by the discrete Gaussian distribution of standard deviation $\sigma_{\boldsymbol{z}} = \tau_m$. For deriving the norm bound B of $\boldsymbol{z}$, notice that its $\mathbb{Z}_q$ coefficients are not independent and have each absolute value at most $t\sigma_{\boldsymbol{z}}$ with probability at least $1 - 2\exp(-\frac{t^2}{2})$. ([28, Le.4.4]) Then, by union bound, $\|\boldsymbol{z}\| \leq t\sigma_{\boldsymbol{z}}\sqrt{rd}$ with probability $1 - 2rd\exp(-\frac{t^2}{2})$.

B.2 Proof of Lemma 2

Proof. We show that the honest prover's messages always satisfy the verifier's checks. Since $\boldsymbol{z} = \boldsymbol{S}\boldsymbol{c}$, it is easy to see that the linear check (13) is satisfied. Finally, thanks to Lemma 1, the norm check $\|\boldsymbol{z}\| \leq t\sigma_{\boldsymbol{z}}\sqrt{rd}$ holds, unless with probability $2rd\exp(-\frac{t^2}{2})$.

B.3 Proof of Lemma 3

Proof. Let $\mathcal{A}$ be an adversary for the protocol in Section 2.1. We describe a knowledge extractor $\mathcal{E}_{\text{lin}}$ for the protocol following the approach from [9]. On input public parameters $\boldsymbol{A}$, instance $(\boldsymbol{T}, (B_i)_i, \boldsymbol{B}, \boldsymbol{U})$ and transcript $tr = (\boldsymbol{c}_0, \boldsymbol{z})$, the extractor $\mathcal{E}_{\text{lin}}$ proceeds as follows.

1. If the transcript is not accepting, i.e. one of the conditions $\|\boldsymbol{z}\| \leq B$, $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{T}\boldsymbol{c}_0$ or $\boldsymbol{B}\boldsymbol{z} = \boldsymbol{U}\boldsymbol{c}_0$ is not satisfied, then the extractor aborts.
2. For each $i \in [m]$:
   (a) Sample $c_i \leftarrow \mathcal{C}$ and set $\boldsymbol{c}_i = \boldsymbol{c}_0$, except that $(c_i)_i = c_i$.
   (b) Run $\mathcal{A}$ on the challenge $\boldsymbol{c}_i$ and get $\boldsymbol{z}_i$. If $(\boldsymbol{c}_i, \boldsymbol{z}_i)$ is not an accepting transcript, and all possible choices of c_i have not yet been exhausted, go back to Step 2a, and otherwise, abort.
   (c) Compute $\bar{\boldsymbol{c}}_i = (\boldsymbol{c}_0)_i - (\boldsymbol{c}_i)_i \in \bar{\mathcal{C}}$ and $\bar{\boldsymbol{s}}_i \coloneqq (\boldsymbol{z}_0 - \boldsymbol{z}_i)/\bar{\boldsymbol{c}}_i \in \mathcal{R}_q^m$.
3. Output $\bar{\boldsymbol{c}} \coloneqq (\bar{\boldsymbol{c}}_i)_i \in \bar{\mathcal{C}}^m$ and $\bar{\boldsymbol{S}} \coloneqq [\bar{\boldsymbol{s}}_1 | \ldots | \bar{\boldsymbol{s}}_m] \in \mathcal{R}_q^{r\times m}$.

Another way to look at the extractor is by considering, for a fixed $i \in [m]$ a binary matrix $H_i \in \{0,1\}^{|\mathcal{C}|^{m-1}\times|\mathcal{C}|}$, where the rows correspond to the values of $(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{i-1}, \boldsymbol{c}_{i+1}, \ldots, \boldsymbol{c}_m)$ and the columns correspond to the values of $\boldsymbol{c}_i$. The matrix contains a 1 if the transcript corresponding to that entry is accepting and otherwise contains a 0. At Step 1, $\mathcal{E}_{\text{lin}}$ finds a 1-entry in a fixed row corresponding to $((\boldsymbol{c}_0)_1, \ldots, (\boldsymbol{c}_0)_{i-1}, (\boldsymbol{c}_0)_{i+1}, \ldots, (\boldsymbol{c}_0)_m)$ and in a column corresponding to $(\boldsymbol{c}_0)_i$, and in Step 2a, the extractor must find a second 1-entry in the same row, but in a different column, to extract the vector $\bar{\boldsymbol{s}}_i$.

Correctness of the Extractor. The extractor produces accepting transcripts with messages $(\boldsymbol{c}_i, \boldsymbol{z}_i)$ for $0 \leq i \leq m$, which satisfy $\boldsymbol{A}\boldsymbol{z}_i = \boldsymbol{T}\boldsymbol{c}_i$, $\boldsymbol{B}\boldsymbol{z}_i = \boldsymbol{U}\boldsymbol{c}_i$ and $\|\boldsymbol{z}_i\| \leq B$.

Let $\boldsymbol{T} = [\boldsymbol{t}_1 \mid \ldots \mid \boldsymbol{t}_m]$ and $\boldsymbol{U} = [\boldsymbol{u}_1 \mid \ldots \mid \boldsymbol{u}_m]$. Subtracting and dividing by $\bar{c}_i$ shows that for $1 \leq i \leq m$, we have $\boldsymbol{A}\bar{\boldsymbol{s}}_i = \boldsymbol{t}_i$ and $\boldsymbol{B}\bar{\boldsymbol{s}}_i = \boldsymbol{u}_i$. Concatenating horizontally and rewriting in terms of $\boldsymbol{S}$ gives $\boldsymbol{A}\bar{\boldsymbol{S}} = \boldsymbol{T}$, $\boldsymbol{B}\bar{\boldsymbol{S}} = \boldsymbol{U}$. Finally, by the triangle inequality, $\|\bar{c}_i\bar{\boldsymbol{s}}_i\| \leq 2B$.

Extractor Running Time. Suppose that $\mathcal{A}$ runs in time T. Then the extractor can check the H_i-entries in time at most T. By [8, Lemma 4], the expected number of H_i-entries queried is at most 2, for any $i \in [m]$, and hence the expected running time of $\mathcal{E}_{lin}$ is at most $2Tm$.

Extractor Success Probability. To argue knowledge soundness according to the definition in Appendix A.4, we must consider the probability that the transcript produced by $\mathcal{A}$ in the definition is accepting and that the extractor fails to produce a witness. Based on the correctness of the extractor, this can only happen if the extractor aborts at Step 2b.

Suppose that $\mathcal{A}$ convinces $\mathcal{V}$ to accept with probability ϵ. The probability that the transcript produced by $\mathcal{A}$ in the knowledge soundness definition is accepting and that the extractor does *not* abort is captured by [8, Lemma 4], and is at most $\epsilon - 1/|\mathcal{C}|$. Therefore, the probability that the extractor aborts at Step 2b is at most $1/|\mathcal{C}|$, so the knowledge error is $\kappa_{lin} = 1/|\mathcal{C}|$.

B.4 Proof of Theorem 6

Proof. We will adapt the proof of [11, Lemma 4.2] to the infinity-norm case. We will denote by $\mathbf{\Pi}_i$ the i-th row of the projection matrix, for any $i \in [256]$.
Case 1. Assume $\|\mathbf{w}\| < q/10$. Then since $\|\mathbf{\Pi w} \bmod q\|_\infty < 0.49b$, then either a) $\|\mathbf{\Pi w}\|_\infty < 0.49b$ or b) there exists some $i \in [256]$ such that $\langle \mathbf{\Pi}_i, \mathbf{w} \rangle \in [kq - 0.49b, kq + 0.49b]$, for some nonzero integer k. Then the probability that event a) happens is at most $\Pr[a)] = \prod_{i\in[256]} \Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| < 0.49\|\mathbf{w}\|] \leq (\frac{1}{\sqrt{2}})^{256} = 2^{-128}$. Here we use $\Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| < 0.49\|\mathbf{w}\|] \leq \frac{1}{\sqrt{2}}$ (proven in Appendix H). The probability that event b) happens is at most

$$\begin{aligned}
\Pr[b)] &\leq \sum_{i\in[256]} \Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| \geq q - 0.49b] \\
&\leq 256 \Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| \geq q - \frac{0.49q}{61}] \\
&\leq 256 \Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| \geq q \cdot 0.99] \\
&\leq 256 \Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| \geq 9.75\|\mathbf{w}\|] \leq 256 \cdot 2^{-141} = 2^{-133}.
\end{aligned}$$

where for the last inequality we have used Lemma 6. Therefore $\Pr[a) \cup b)] \leq \Pr[a)] + \Pr[b)] \approx 2^{-128}$.
Case 2. Assume $\|\mathbf{w}\|_\infty > q/61$. This leads to, for any $i \in [256]$, $\Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| \leq q/122] \leq \Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| \leq 1/2\|\mathbf{w}\|_\infty] \leq 1/2$, where for the last part we use Lemma A.1. from [31]. Using this and the fact that $0.49b < 0.49q/61 < q/122$, we get

$$\begin{aligned}
\Pr[\|\mathbf{\Pi w} \bmod q\|_\infty < 0.49b] &\leq \Pr[\|\mathbf{\Pi w} \bmod q\|_\infty \leq q/122] \\
&\leq \Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle| \leq q/122]^{256} \leq 2^{-256}
\end{aligned}$$

Case 3. Assume $\|\mathbf{w}\| \geq q/10$ and $\|\mathbf{w}\|_\infty \leq q/61$. We construct a vector $\mathbf{v} \in \mathbb{Z}^d$ from $\mathbf{w}$ for which we set its entries $\mathbf{v}_i$ as 0 as least as possible and elsewhere equal to $\mathbf{w}_i$ such that $6q/61 \leq \|\mathbf{v}\| \leq q/10$. Indeed, if we denote by $\mathbf{v}'$ the previous vector before we set one entry of $\mathbf{w}$ as 0 to get $\mathbf{v}$, then $\|\mathbf{v}'\| \geq q/10$ and $\|\mathbf{v}\| = \sqrt{\|\mathbf{v}'\|^2 - |\mathbf{w}_{j_0}|^2} \geq \sqrt{(q/10)^2 - (q/61)^2} \geq 6q/61$, since $|\mathbf{w}_{j_0}| \leq \|\mathbf{w}\|_\infty \leq q/61$. By construction, $\mathbf{v}$ and $\mathbf{w} - \mathbf{v}$ have disjoint supports. We also have, for an arbitrary fixed row $\mathbf{\Pi}_i$, $i \in [256]$, $\langle \mathbf{\Pi}_i, \mathbf{w}\rangle = \langle \mathbf{\Pi}_i, \mathbf{v}\rangle + \langle \mathbf{\Pi}_i, \mathbf{w} - \mathbf{v}\rangle$. Moreover, $\|\mathbf{v}\|_\infty \leq \|\mathbf{w}\|_\infty \leq q/61 < \|\mathbf{v}\|/6$, which tells that the ℓ_2 norm of $\mathbf{v}$ is not concentrated too much in a few entries, so intuitively, $\langle \mathbf{\Pi}_i, \mathbf{v}\rangle$ is a sum of small independent variables and its distribution should therefore be close to the normal distribution. To argue this fact, we will make use of the Berry-Esseen Theorem, which says that if $X = X_1 + \ldots + X_k$ is a sum of independent variables of mean $E[X_i] = 0$ and variances $E[X_i^2] = \sigma_i^2$, and third moments $E[|X_i|^3] = \rho_i$ and if Y is a normal distribution with zero mean and variance $\sigma^2 = \sum \sigma_i^2$, then

$$\sup_x |\Pr[X \leq x] - \Pr[Y \leq x]| \leq 0.52 \frac{\max_{i\in[k]} \rho_i/\sigma_i^2}{\sqrt{\sum_{i\in[k]} \sigma_i^2}}.$$

We apply this theorem for $\langle \mathbf{\Pi}_i, \mathbf{v}\rangle = \sum_{j\in[d]} \mathbf{\Pi}_{ij}\mathbf{v}_j$. Since the projection matrix has entries from $\mathtt{Bin}_1$ distribution, any term $X_j = \mathbf{\Pi}_{ij}\mathbf{v}_j$ from the sum has $E[X_j] = 0$, $E[X_j^2] = \mathbf{v}_j^2/2$ and $E[|X_j|^3] = |\mathbf{v}_j|^3/2$.

Therefore $\max_{j\in[d]} \rho_j/\sigma_j^2 = \max_{j\in[d]} |\mathbf{v}_j| = \|\mathbf{v}\|_\infty$ and $\sqrt{\sum_{i\in[k]} \sigma_i^2} = \|\mathbf{v}\|/\sqrt{2}$. Applying the theorem leads us to

$$\sup_x |\Pr[\langle \mathbf{\Pi}_i, \mathbf{v}\rangle \le x] - \Pr[Y \le x]| \le 0.52 \frac{\|\mathbf{v}\|_\infty}{\|\mathbf{v}\|/\sqrt{2}} \le 0.52 \frac{1}{6/\sqrt{2}} < 0.13 := \delta,$$

where Y is a normal random variable of standard deviation $\sigma = \|\mathbf{v}\|/\sqrt{2}$. Let a be a fixed value in $\{-q/2, \ldots, q/2\}$. Then

$$\Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle \bmod q| < 0.49b | \langle \mathbf{\Pi}_i, \mathbf{w} - \mathbf{v}\rangle = a] \le p_1 + p_2,$$

where $p_1 = \Pr[|\langle \mathbf{\Pi}_i, \mathbf{v}\rangle| > q/2]$ and $p_2 = \Pr[\langle \mathbf{\Pi}_i, \mathbf{v}\rangle \in G]$, where we define the set G as $G = \{x \,|\, |x + a \bmod q| < 0.49b\} \cap [-q/2, q/2]$. Since $q/2 > 5\|\mathbf{v}\|$, we have $p_1 \le \Pr[|\langle \mathbf{\Pi}_i, \mathbf{v}\rangle| > 5\|\mathbf{v}\|] = 2\Pr[\langle \mathbf{\Pi}_i, \mathbf{v}\rangle > 5\|v\|]$, where equality follows by symmetry. As for a variable $r \leftarrow \mathtt{Bin}_1$, $E[\exp(\lambda \cdot r)] = \frac{1}{4}\exp(-\lambda) + \frac{1}{2} + \frac{1}{4}\exp(\lambda) = \cosh^2(\frac{\lambda}{2})$, using independence of the entries of $\mathbf{\Pi}_i$ and $\cosh(x) \le \exp(\frac{x^2}{2})$, we get $E[\exp(\lambda \cdot \langle \mathbf{\Pi}_i, \mathbf{v}\rangle)] = \prod_i \cosh^2(\frac{\lambda \mathbf{v}_i^2}{2}) \le \exp(\frac{\lambda^2\|\mathbf{v}\|^2}{4})$. By Markov's inequality,

$$\Pr[\langle \mathbf{\Pi}_i, \mathbf{v}\rangle > t] \le \Pr[\exp(\lambda \cdot \langle \mathbf{\Pi}_i, \mathbf{v}\rangle) > \exp(\lambda \cdot t)] \le \frac{E[\exp(\lambda \cdot \langle \mathbf{\Pi}_i, \mathbf{v}\rangle)]}{\exp(\lambda \cdot t)} \le \exp(-\lambda \cdot t + \frac{\lambda^2\|\mathbf{v}\|^2}{4}).$$

Letting $t = 5\|\mathbf{v}\|$ and $\lambda = \frac{2t}{\|\mathbf{v}\|^2}$, we get

$$p_1 \le 2\Pr[\langle \mathbf{\Pi}_i, \mathbf{v}\rangle > t] \le 2\exp(-\frac{t^2}{\|\mathbf{v}\|^2}) = 2\exp(-25) \approx 2^{-35.1}.$$

So we can upper bound $p_1 < 2^{-35}$. [12] To upper bound p_2, we "approximate" $\langle \mathbf{\Pi}_i, \mathbf{v}\rangle$ with the random variable $Y \simeq N(0, \sigma)$, where $\sigma = \|\mathbf{v}\|/\sqrt{2}$. Since the probability that Y falls in G is at most the measure of the set G, but divided by $\sigma\sqrt{2\pi}$, and the measure of G is at most $2 \cdot 0.49b$, then the probability that Y falls in G is at most $\frac{2 \cdot 0.49b}{\|\mathbf{v}\|\sqrt{2}\cdot\sqrt{2\pi}}$. Further, either the set G or $[-\frac{q}{2}, \frac{q}{2}] \setminus G$ is an interval, so it follows using the Berry-Esseen theorem that $|\langle \mathbf{\Pi}_i, \mathbf{v}\rangle| \in G$ with probability p_2 at most

$$p_2 \le \frac{2 \cdot 0.49b}{\|\mathbf{v}\|/\sqrt{2} \cdot \sqrt{2\pi}} + 2\delta \le \frac{2 \cdot 0.49 \cdot q/61}{6q/61 \cdot \sqrt{\pi}} + 2\delta < 0.1 + 0.26 = 0.36.$$

Therefore,

$$\Pr[\|\mathbf{\Pi}\mathbf{w} \bmod q\|_\infty < 0.49b] \le (\Pr[|\langle \mathbf{\Pi}_i, \mathbf{w}\rangle \bmod q| < 0.49b])^{256} < (\frac{1}{2^{35}} + 0.36)^{256} < \left(\frac{1}{\sqrt{2}}\right)^{256} < \frac{1}{2^{128}}.$$

C Details of (some) LaBRADOR Steps

We detail here the LaBRADOR [11] steps used in the Protocol from Section 4.2, performed per each row block. We assume that each row block of index $\ell \in [n]$ and of rows $\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}$ satisfies for any $f_k \in \mathcal{F}$ and $f'_l \in \mathcal{F}'$

$$f_k(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}) \triangleq \sum_{i,j\in[K]} a_{ij}^{(k)} \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)}\rangle + \sum_{i\in[K]} \langle \boldsymbol{\varphi}_i^{(k)}, \tilde{\boldsymbol{s}}_i^{(\ell)}\rangle - b_k = 0,$$

$$\mathtt{ct}(f'_l(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)})) \triangleq \mathtt{ct}(\sum_{i,j\in[K]} a_{ij}^{\prime(l)} \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)}\rangle + \sum_{i\in[K]} \langle \boldsymbol{\varphi}_i^{\prime(l)}, \tilde{\boldsymbol{s}}_i^{(\ell)}\rangle - b'_{0,l}) = 0,$$

$$\|\tilde{\boldsymbol{s}}_1^{(\ell)}\| \le \mathtt{B}_1, \ldots, \|\tilde{\boldsymbol{s}}_K^{(\ell)}\| \le \mathtt{B}_K.$$

where $a_{ij}^{(k)}, a_{ij}^{\prime(l)} \in \mathcal{R}_q$, $\boldsymbol{\varphi}_i^{(k)}, \boldsymbol{\varphi}_i^{\prime(l)} \in \mathcal{R}_q^m$, $b_k \in \mathcal{R}_q$, $b'_{0,l} \in \mathbb{Z}_q$.

[12] Alternatively, using the heuristic from [21, Sec 3.2], that all the entries from the $\mathbf{\Pi}_i$ follow the normal distribution of standard deviation $1/\sqrt{2}$, one could find $p_1 \le \Pr[|\langle \mathbf{\Pi}_i, \mathbf{v}\rangle| > 5\|\mathbf{v}\|] < 2^{-39}$.

C.1 Project

In this step, we replace the norm statements on the rows $\tilde{\boldsymbol{s}}_i^{(\ell)} \in \mathcal{R}_q^m$, $\|\tilde{\boldsymbol{s}}_i^{(\ell)}\| \leq \mathtt{B}_i$, $i \in [K]$, with modular Johnson-Lindenstrauss projections, as in [11, 21, 29], which allows us to prove looser bounds than the actual norm bound $\mathtt{B}_i$, up to some slack factor α. Namely, upon receiving the projection matrix $\boldsymbol{\Pi} \leftarrow \mathtt{Bin}_1^{256\times dm}$, the prover answers with the projection of the coefficient vector $s_i^{(\ell)} \in \mathbb{Z}^{dm}$, namely $\boldsymbol{\Pi} s_i^{(\ell)}$, for any $\ell \in [n]$ and $i \in [K]$, and lets the verifier check if $\|\boldsymbol{\Pi} s_i^{(\ell)}\| \leq \sqrt{337}\mathtt{B}_i$. By Lemma 6, this happens unless with probability $1/2^{128}$. One could also consider a verifier check $\|\boldsymbol{\Pi} s_i^{(\ell)}\| \leq \sqrt{128}\mathtt{B}_i$, but this happens with a bigger probability, $1/2$, which would lead to higher probability of all norm checks to not hold simultaneously. Note that the same projection matrix could be used, for any row $\tilde{\boldsymbol{s}}_i^{(\ell)}$. If one of these checks does not hold, the prover can request other projection matrix until they do. If all these K checks hold, then we get that $\|\tilde{\boldsymbol{s}}_i^{(\ell)}\| \leq \sqrt{337/30}\mathtt{B}_i$, unless with probability $K/2^{128}$ (see Theorem 5), hence we introduce the slack factor $\alpha = \sqrt{337/30} \approx 3.37$.

We also need to prove that the projections are well formed. As in [11, 21, 29], we rewrite the projection entries as constant-term constraints satisfied by the rows $\tilde{\boldsymbol{s}}_i^{(\ell)}$. Let us denote $p_i^{(\ell)} \coloneqq \boldsymbol{\Pi} s_i^{(\ell)} \in \mathbb{Z}^{256}$, $p_{i,j}^{(\ell)}$ the j-th entry of this projection, $\tilde{\boldsymbol{\pi}}^{(j)}$ be the j-th row of the projection matrix, and $\boldsymbol{\pi}^{(j)}$ be its corresponding polynomial vector via this encoding, for $j \in [256]$. Then each $\tilde{\boldsymbol{s}}_i^{(\ell)}$ satisfies 256 constraints, of the form

$$\mathtt{ct}(\langle \sigma_{-1}(\boldsymbol{\pi}^{(j)}), \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle - p_{i,j}^{(\ell)}) = 0.$$

As there are K rows $\tilde{\boldsymbol{s}}_i^{(\ell)}$, we get in total $256K$ projection constraints per each row block, all linear in the rows. We can identify each projection with a ring element in $\mathcal{R}$, as the ring degree we are using is 256. Notice that, per row block, these constraints are identical up to their constant coefficient $p_{i,j}^{(\ell)} \in \mathbb{Z}$, which highly depends on the rows of the row block. We denote these constraints by $g_i^{(\ell)}$, $i \in [256K]$, $\ell \in [n]$.

C.2 Aggregate with Integer Challenges

In this step, all the input constant-term constraints $f' \in \mathcal{F}'$ are linearly combined (or aggregated) as in [11, 29], within the row block, together with the projection constraints from Appendix C.1, with uniformly random integer challenges from $\mathbb{Z}_q$ from the verifier. Note that this aggregation preserves the zero constant coefficient. Note that, if one of the constant-term constraints is not satisfied, i.e. its constant coefficient is not 0, then this aggregation has constant coefficient 0 with probability at most $1/q$ over the randomness of the integer challenges. One can linearly combine the constraints $\lambda \coloneqq \lceil 128/\log q \rceil$ many times, so then the cheating probability becomes $(1/q)^\lambda \approx 1/2^{128}$. Then, upon receiving from the verifier λ sets of challenges $\psi_1^{(k)}, \ldots, \psi_{|\mathcal{F}'|}^{(k)} \leftarrow \mathbb{Z}_q$ to aggregate f''s and of $\omega_1^{(k)}, \ldots, \omega_{256K}^{(k)} \leftarrow \mathbb{Z}_q$, where $\omega_{256(i-1)+1}^{(k)}, \ldots, \omega_{256i}^{(k)}$ aggregate each group of 256 projection constraints per row i, the prover computes for any $k \in [\lambda]$,

$$\begin{aligned} f_k''^{(\ell)}(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}) &= \sum_{j\in[|\mathcal{F}'|]} \psi_j^{(k)} f_j'(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}) + \sum_{l\in[256K]} \omega_l^{(k)} g_l^{(\ell)}(\tilde{\boldsymbol{s}}_1^{(\ell)}, \ldots, \tilde{\boldsymbol{s}}_K^{(\ell)}) \\ &= \sum_{i,j\in[K]} a_{ij}''^{(k)} \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \sum_{i\in[K']} \langle \boldsymbol{\varphi}_i''^{(k)}, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle - b_{0,k}''^{(\ell)} \end{aligned}$$

where $a_{ij}''^{(k)} \in \mathcal{R}_q, \boldsymbol{\varphi}_i''^{(k)} \in \mathcal{R}_q^m, b_{0,k}''^{(\ell)} \in \mathbb{Z}$ are set accordingly.

$$a_{ij}''^{(k)} = \sum_{l\in[|\mathcal{F}'|]} \psi_l^{(k)} a_{ij}'^{(l)} \tag{52}$$

$$\boldsymbol{\varphi}_i''^{(k)} = \sum_{l\in[|\mathcal{F}'|]} \psi_l^{(k)} \boldsymbol{\varphi}_i'^{(l)} + \sum_{j\in[256]} \omega_{256(i-1)+j}^{(k)} \sigma_{-1}(\boldsymbol{\pi}^{(j)}) \tag{53}$$

$$b_{0,k}''^{(\ell)} = \sum_{l\in[|\mathcal{F}'|]} \psi_l^{(k)} b_{0,l}' + \sum_{i\in[K],j\in[256]} \omega_{256(i-1)+j}^{(k)} p_{i,j}^{(\ell)} \tag{54}$$

Except for the constant term $b''^{(\ell)}_{0,k}$, the constraints $f''^{(\ell)}$ have the same coefficients, for any row block of index ℓ. These linear combinations $f''^{(\ell)}_k$ are now extended to full constraints of the type of $\mathcal{F}$, for proving their well formedness. Namely, the prover extends the integers $b''^{(\ell)}_{0,k}$ to full elements $b''^{(\ell)}_k$ in $\mathcal{R}_q$ called "lifting" polynomials, such that $f''^{(\ell)}_k(\tilde{\boldsymbol{s}}^{(\ell)}_1, \ldots, \tilde{\boldsymbol{s}}^{(\ell)}_K) = 0$. Note that the lifting polynomials depend on the rows $\tilde{\boldsymbol{s}}^{(\ell)}_1, \ldots, \tilde{\boldsymbol{s}}^{(\ell)}_K$ of the row block of index ℓ. The prover then sends these lifting polynomials $b''^{(\ell)}_k$, λ many per each row block, leading to λn ring elements in total. The verifier has to check

$$\mathtt{ct}(b''^{(\ell)}_k) = b''^{(\ell)}_{0,k}, \text{ for } k \in [\lambda], \ \ell \in [n]. \tag{55}$$

C.3 Aggregate with Polynomial Challenges

In this step, all the input full constraints $f \in \mathcal{F}$ are linearly combined (or aggregated) as in [11, 29], within the row block, together with the constraints f'' from Appendix C.2, with uniformly random challenges over $\mathcal{R}_q$ from the verifier. Namely, the verifier sends $\mu_1, \ldots, \mu_{|\mathcal{F}|} \leftarrow \mathcal{R}_q$ and $\nu_1, \ldots, \nu_\lambda \leftarrow \mathcal{R}_q$, which define

$$\begin{aligned} F^{(\ell)}(\tilde{\boldsymbol{s}}^{(\ell)}_1, \ldots, \tilde{\boldsymbol{s}}^{(\ell)}_K) &= \sum_{j \in [|\mathcal{F}|]} \mu_j f_j(\tilde{\boldsymbol{s}}^{(\ell)}_1, \ldots, \tilde{\boldsymbol{s}}^{(\ell)}_K) + \sum_{k \in [t]} \nu_k f''^{(\ell)}_k(\tilde{\boldsymbol{s}}^{(\ell)}_1, \ldots, \tilde{\boldsymbol{s}}^{(\ell)}_K) \\ &= \sum_{i,j \in [K]} a_{ij} \langle \tilde{\boldsymbol{s}}^{(\ell)}_i \boldsymbol{D}, \tilde{\boldsymbol{s}}^{(\ell)}_j \rangle + \sum_{i \in [K]} \langle \boldsymbol{\varphi}_i, \tilde{\boldsymbol{s}}^{(\ell)}_i \rangle - b^{(\ell)} = 0, \end{aligned}$$

where $a_{ij} \in \mathcal{R}_q$, $\boldsymbol{\varphi}_i \in \mathcal{R}^m_q, b^{(\ell)} \in \mathcal{R}_q$ are set accordingly:

$$a_{ij} = \sum_{l \in [|\mathcal{F}|]} \mu_l a^{(l)}_{ij} + \sum_{k \in [\lambda]} \nu_k a''^{(k)}_{ij} \tag{56}$$

$$\boldsymbol{\varphi}_i = \sum_{l \in [|\mathcal{F}|]} \mu_l \boldsymbol{\varphi}^{(l)}_i + \sum_{k \in [\lambda]} \nu_k \boldsymbol{\varphi}''^{(k)}_i \tag{57}$$

$$b^{(\ell)} = \sum_{l \in [|\mathcal{F}|]} \mu_l b_l + \sum_{k \in [\lambda]} \nu_j b''^{(\ell)}_k \tag{58}$$

Except for the constant term $b^{(\ell)}$, the constraints $F^{(\ell)}$ have the same coefficients per each row block of index ℓ. Note that, if one of these constraints is not satisfied, then this aggregation is vanished with probability at most $1/q^{d/2}$ over the randomness of the challenges over $\mathcal{R}_q$, assuming q splits in two factors.

D Particular Constraints

We would be interested in proving approximate norm proofs, binary proofs, or exact norm proofs, using lattice techniques as in prior works [11, 29].

Binary proof. For proving a vector $\tilde{\boldsymbol{x}} \in \mathcal{R}^m_q$ is made only of ring elements with binary coefficients, or equivalently, the coefficients of any entry of this vector are binary vectors, we make use of the observation [29, Lem 5.2] saying that an integer vector $\mathbf{x} \in \mathbb{Z}^{md}$ is binary if and only if $\langle \mathbf{x}, \mathbf{x} - \mathbf{1}_{md} \rangle = 0$ over the integers, where $\mathbf{1}_n$ is the integer vector with all entries equal to 1. This translates as showing that the corresponding polynomial vector $\tilde{\boldsymbol{x}}$ over $\mathcal{R}$ satisfies

$$\mathtt{ct}(\langle \sigma_{-1}(\tilde{\boldsymbol{x}} - \mathbf{1}), \tilde{\boldsymbol{x}} \rangle) = 0, \tag{59}$$

For proving this, we need to add an auxiliary witness vector, $\sigma_{-1}(\tilde{\boldsymbol{x}})$. However, one could prove Equation (59) only mod q. In order to prove this holds over $\mathbb{Z}$, we would need to also prove that the vector $\tilde{\boldsymbol{x}}$ is small, i.e. $\|\tilde{\boldsymbol{x}}\| \leq \sqrt{md}$, by projecting it as before, and thus prove its norm, up to some slack factor α. As the left hand side of Equation (59) is $|\langle \mathbf{x}, \mathbf{x} - \mathbf{1}_{md} \rangle| \leq 2\|\mathbf{x}\|^2$, for any $\mathbf{x} \in \mathbb{Z}^{md}$ integer vector, by letting $2 \cdot \alpha^2 \cdot md < q$, we guarantee that Equation (59) holds over the integers.

Exact norm proof. For proving a vector $\tilde{\boldsymbol{x}} \in \mathcal{R}_q^m$ has exact norm, say $\|\tilde{\boldsymbol{x}}\| \leq B$, one could prove that $B^2 - \|\tilde{\boldsymbol{x}}\|^2$ is a positive integer, by taking its binary representation of length $\log B^2 \leq d$, as in [29]. More precisely, one could prove that for some binary polynomial $b \in \mathcal{R}$, both $\tilde{\boldsymbol{x}}$ and d satisfy

$$\mathtt{ct}(\langle \tilde{\boldsymbol{x}}, \sigma_{-1}(\tilde{\boldsymbol{x}}) \rangle + b \cdot e - B^2) = 0,$$

where $e = \sigma_{-1}(\sum_{i \in [\lceil \log (B^2) \rceil]} 2^{i-1} X^{i-1})$. This requires taking additional witness vectors $\tilde{\boldsymbol{x}}_2 \coloneqq \sigma_{-1}(\tilde{\boldsymbol{x}})$, $\tilde{\boldsymbol{x}}_3 \coloneqq [b, 0, \ldots, 0] \in \mathcal{R}_q^m$. Then, for proving exact norm, one proves the following satisfied by $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_2, \tilde{\boldsymbol{x}}_3$,

$$\mathtt{ct}(\langle \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_2 \rangle + \langle \tilde{\boldsymbol{x}}_3, e \cdot \tilde{\boldsymbol{e}}_1 \rangle - B^2) = 0, \tag{60}$$

where $\tilde{\boldsymbol{e}}_1$ is the unit vector $[1, 0, \ldots, 0] \in \mathcal{R}_q^m$. We also need to prove that $\tilde{\boldsymbol{x}}_3$ is a binary vector, as before. As in the binary case, one could prove Equation (60) only mod q. Therefore, one needs also to project the row $\tilde{\boldsymbol{x}}$, and thus prove that $\|\tilde{\boldsymbol{x}}\| \leq B$ up to some slack factor α. Since $\tilde{\boldsymbol{x}}_3$ is proven to be binary, the left hand side of Equation (60) is in absolute value less than $B^2 \cdot \alpha^2 + 2B^2 - 1$ and so, setting $B^2 \cdot \alpha^2 + 2B^2 - 1 < q$, we guarantee that Equation (60) holds over the integers.

Automorphism proof. As particular constraints, such as binary or exact norm proofs, involve adding new witness vectors when applying the automorphism σ_{-1}, one needs to prove their correctness. More precisely, as in [1, 11], we have to prove that the k-th coefficient of some polynomial $a \in \mathcal{R}$ matches the $d - k$-th coefficient of $\tilde{a} \coloneqq \sigma_{-1}(a)$, up to a sign, as in

$$\mathtt{ct}(X^{d-k} a - X^k \tilde{a}) = 0. \tag{61}$$

E Parameter Selection

We work with the ring $\mathcal{R} = \mathbb{Z}[X]/(X^{256} + 1)$. We consider the challenge space $\mathcal{C}$ as in Section 2.1 with ring elements having 22 coefficients equal to ± 1 and the rest of them equal to 0, and of bounded operator norm $T = 10$. In this case, the ℓ_2 norm of each challenge is $\tau = \sqrt{22}$.

We provide in Table 2 an overview of notation of parameters involved in our main protocol Orthus, as presented in Section 5.

Parameter	Explanation
q	SIS modulus
t	Height of the matrix $\boldsymbol{A}$
K	Height of a block
N	Width of a block
n	No. of row blocks
m	No. of blocks per row block
nK	No. of rows of witness matrix
mN	No. of columns of witness matrix
$\mathtt{t}_1$	Rank inner commit.
$\mathtt{t}_2$	Rank middle commit.

Table 2. Overview of parameters and notation

Proof size. The size of the non-interactive variant (via Fiat-Shamir) of Orthus is given by the size of each of the six outer commitments sent by the prover. As in Section 5.1, we can set its rank as only 64 to assure security of the M-SIS problem with ℓ_∞ norm bound 1 when committing to middle commitments,

and rank 256 when committing to the binary decompositions of the projections. As in total we have 6 outer commitments, we get 5 of them made of 64 elements in $\mathbb{Z}_q$ and one of them made of 256 elements in $\mathbb{Z}_q$. In total, we get a proof size of 576 elements in $\mathbb{Z}_q$.

Modulus q. Recall that the projections are decomposed in binary for creating the outer commitments, and hence, their decompositions inherently read the infinity norm of the projections. By the adapted Johnson-Lindenstrauss lemma for the infinity norm (Theorem 6) used in the soundness proof of this protocol, we have the following condition on the modulus:

$$\max_{i\in[K]} 61 \cdot \frac{9.75}{0.49} \cdot \mathtt{B}'_i\sqrt{m} \leq q.$$

More requirements can be made on the modulus q, if one wants to prove exact-norm or binary constraints, as is the case of, e.g., aggregating Falcon signatures.

We recall here all the $\mathcal{P} \to \mathcal{V}$ messages and describe the bases we use to decompose them, in order to derive the middle commitments and a better input to LaBRADOR. Except for the projections $p_i^{(\ell)}$ and the column amortization $\boldsymbol{z}$, we assume, as in [11], that each other prover message has uniform coefficients mod q and hence, it is decomposed with respect to a basis $\mathtt{b}_{\mathtt{mid}}$ in $\mathtt{t}_{\mathtt{mid}} = \lfloor \frac{\log q}{\mathtt{b}_{\mathtt{mid}}} \rceil$ parts. This basis is set in order to obtain a number of decomposition parts as small as possible and to assure binding via hardness of SIS as below. These decomposition parts are assumed, as in [11], to have uniform coefficients in $\{-\mathtt{b}_{\mathtt{mid}}/2, \ldots, \mathtt{b}_{\mathtt{mid}}/2\}$, so of standard deviation essentially $\mathtt{b}_{\mathtt{mid}}\sqrt{12}$. The projections are decomposed in binary decompositions, whereas $\boldsymbol{z}$ is decomposed with respect to a basis $\mathtt{b}$, which is set following the heuristic analysis of [11].

The prover's messages and their respective norm bounds involved in the SIS problems are the following:

- the inner commitments $\boldsymbol{T}$: it is made of mN commitments to the columns and each has $\mathtt{t}_1 = t$ elements in $\mathcal{R}_q$, so in total $\mathtt{n}_1 \coloneqq \mathtt{t}_1 mN$ ring elements. We decompose it into $\hat{\boldsymbol{T}}$ with respect to $\mathtt{b}_{\mathtt{mid}}$ to be committed in the middle layer. We expect its ℓ_2 norm to be
$$\|\hat{\boldsymbol{T}}\| \leq \beta''_1 \coloneqq \sqrt{\mathtt{n}_1\mathtt{t}_{\mathtt{mid}} d \cdot \frac{\mathtt{b}_{\mathtt{mid}}^2}{12}}.$$
- the projections $p_i^{(\ell)}$, $i \in [K]$, $\ell \in [n]$: each of them is just one element in the ring $\mathcal{R}$, as its degree is $d = 256$, so in total $\mathtt{n}_2 \coloneqq nK$ elements. Since $p_i^{(\ell)}$ is a projection of $\tilde{\boldsymbol{s}}_i^{(\ell)}$ of norm at most $\mathtt{B}'_i$, by Lemma 6, this has $\|p_i^{(\ell)}\|_\infty \leq 9.75\mathtt{B}'_i\sqrt{m}$. Therefore, its binary decomposition $\hat{p}_i^{(\ell)}$ is made of $\lceil \log 9.75\mathtt{B}'_i\sqrt{m} \rceil$ ring elements with binary coefficients to be committed in the outer layer. We get $\mathtt{n}'_2 \coloneqq n\sum_{i\in[K]} \lceil \log 9.75\mathtt{B}'_i\sqrt{m} \rceil$ ring elements with binary coefficients in total to be committed in the outer layer.
- the lifting polynomials $b_k''^{(\ell)}$, $k \in [\lambda]$, $\ell \in [K]$, where $\lambda = \lceil 128/\log q \rceil$: they are λ ring elements per row block, which leads to a total of $\mathtt{n}_3 \coloneqq \lambda n$ elements. We decompose them into $\hat{b}_k''^{(\ell)}$ with respect to basis $\mathtt{b}_{\mathtt{mid}}$ to be committed in the middle layer. We expect their ℓ_2 norm to be
$$\|(\hat{b}_k''^{(\ell)})\| \leq \beta''_3 \coloneqq \sqrt{\mathtt{n}_3\mathtt{t}_{\mathtt{mid}} d \cdot \frac{\mathtt{b}_{\mathtt{mid}}^2}{12}}.$$
- the garbage elements $\boldsymbol{g}_{ij}^{(\ell)}, \boldsymbol{h}_{ij}^{(\ell)}$: because of symmetry, we consider only $K(K+1)/2$ elements in $\mathcal{R}_q$ of each kind per row block, so we get $\mathtt{n}_4 \coloneqq nK(K+1)$ elements in $\mathcal{R}_q$. We decompose them into $(\hat{\boldsymbol{g}}_{ij}^{(\ell)}, \hat{\boldsymbol{h}}_{ij}^{(\ell)})$ with respect to basis $\mathtt{b}_{\mathtt{mid}}$ to be committed in the middle layer. We expect their ℓ_2 norm to be
$$\|(\hat{\boldsymbol{g}}_{ij}^{(\ell)}, \hat{\boldsymbol{h}}_{ij}^{(\ell)})\| \leq \beta''_4 \coloneqq \sqrt{\mathtt{n}_4\mathtt{t}_{\mathtt{mid}} d \cdot \frac{\mathtt{b}_{\mathtt{mid}}^2}{12}}.$$
- the garbage terms v_i: they are in total $\mathtt{n}_5 = 2n - 2$ ring elements as in Section 3. We decompose them into $(\hat{v}_i)$ with respect to basis $\mathtt{b}_{\mathtt{mid}}$ to be committed in the middle layer. We expect their ℓ_2 norm to be
$$\|(\hat{v}_i)\| \leq \beta''_5 \coloneqq \sqrt{\mathtt{n}_5\mathtt{t}_{\mathtt{mid}} d \cdot \frac{\mathtt{b}_{\mathtt{mid}}^2}{12}}.$$

– the row amortizations $\tilde{\boldsymbol{z}}_+, \tilde{\boldsymbol{z}}_-, \tilde{\boldsymbol{z}}'_+$ in Section 3 make together $\mathtt{n}_6 \coloneqq 3mN$ ring elements. We decompose them into $(\hat{\tilde{\boldsymbol{z}}}_+, \hat{\tilde{\boldsymbol{z}}}_-, \hat{\tilde{\boldsymbol{z}}}'_+)$ with respect to basis $\mathtt{b}_{\mathtt{mid}}$ to be committed in the middle layer. We expect their ℓ_2 norm to be

$$\|(\hat{\tilde{\boldsymbol{z}}}_+,\ \hat{\tilde{\boldsymbol{z}}}_-,\ \hat{\tilde{\boldsymbol{z}}}'_+)\| \leq \beta''_6 \coloneqq \sqrt{\mathtt{n}_6 \mathtt{t}_{\mathtt{mid}} d \cdot \frac{\mathtt{b}_{\mathtt{mid}}^2}{12}}.$$

– the column amortization $\boldsymbol{z}$: as in [11], we decompose it in two parts by reducing it modulo $\mathtt{b}$, as $\boldsymbol{z} = \boldsymbol{z}^{(0)} + \mathtt{b}\boldsymbol{z}^{(1)}$. This is to make sure that the coefficients of $\boldsymbol{z}$ do not blow up. We assume $\boldsymbol{z}^{(0)}$ has uniform coefficients modulo $\mathtt{b}$, so of standard deviation $\mathtt{b}/\sqrt{12}$, whereas $\boldsymbol{z}^{(1)}$ has coefficients following Gaussian distribution of standard deviation $\sigma_{\boldsymbol{z}}/\mathtt{b}$, where $\sigma_{\boldsymbol{z}}$ is as in Lemma 1. We impose $\mathtt{b}/\sqrt{12} \approx \sigma_{\boldsymbol{z}}/\mathtt{b}$, therefore we set $\mathtt{b} \coloneqq \lfloor \sqrt{\sigma_{\boldsymbol{z}}\sqrt{12}} \rceil$. We set the norm bounds on the parts of amortizations as $\|\boldsymbol{z}^{(i)}\| \leq B/\mathtt{b}$, hence $\|\boldsymbol{z}\| = \|\boldsymbol{z}^{(0)} + \mathtt{b}\boldsymbol{z}^{(1)}\| \leq \beta' \coloneqq (1+\mathtt{b})B/\mathtt{b}$, where B is set as in Section 2.1.

Commitment ranks $\mathtt{t}_1, \mathtt{t}_2$. For assuring the binding property of the inner commitment layer, we need to use hardness of M-SIS$_{\mathtt{t}_1, 8T\beta'}$. For assuring the binding property of the middle commitment layer, we need to use hardness of M-SIS$_{\mathtt{t}_2, 2\beta''_i}$, for $i \in [6]$, $i \neq 2$. The ranks are computed in the usual way, c.f. [5].

LaBRADOR modulus. For proving over $\mathbb{Z}$ the binary Equation (59) with LaBRADOR of $\tilde{\boldsymbol{x}}$, the concatenation of the binary decompositions of the middle commitments and of projections, as mentioned in Appendix D, we use the observation that $|\mathtt{ct}(\langle \sigma_{-1}(\tilde{\boldsymbol{x}} - \mathbf{1}), \tilde{\boldsymbol{x}} \rangle)| \leq 2\|\tilde{\boldsymbol{x}}\|^2 = \|(\tilde{\boldsymbol{x}}, \sigma_{-1}(\tilde{\boldsymbol{x}}))\|^2 \leq 2d\mathtt{n}_{\mathtt{bin}}$, where $\mathtt{n}_{\mathtt{bin}}$ is the length of $\tilde{\boldsymbol{x}}$ as a vector of ring elements in $\mathcal{R}$. Therefore, it suffices to set $d\mathtt{n}_{\mathtt{bin}} \leq q'/2$ so that Equation (59) does not wrap around modulo q', where q' is the LaBRADOR modulus. Recall from Section 5 that $\tilde{\boldsymbol{x}}$ is constructed from

– binary decompositions of the projections, in total $n \sum_{i \in [K]} \lceil \log (9.75\mathtt{B}'_i \sqrt{m}) \rceil$ elements in $\mathcal{R}_q$ with binary coefficients,
– binary decompositions of the six middle commitments, each being made of $\mathtt{t}_2 \lceil \log q \rceil$ ring elements with binary coefficients.

We get in total $\mathtt{n}_{\mathtt{bin}} \coloneqq n \sum_{i \in [K]} \lceil \log (9.75\mathtt{B}'_i \sqrt{m}) \rceil + 6\mathtt{t}_2 \lceil \log q \rceil$ such terms in $\mathcal{R}$. Since LaBRADOR provides approximate norm proofs up to the slack factor of $\sqrt{128/30}$ [11, Le.4.2], we need

$$2 \cdot d \cdot (n \sum_{i \in [K]} \lceil \log (9.75\mathtt{B}'_i \sqrt{m}) \rceil + 6\mathtt{t}_2 \lceil \log q \rceil) \cdot \frac{128}{30} \leq q'.$$

We notice that the additional outer commitment layer described in Section 5 can be also applied for the LaBRADOR proof system. This involves, in particular, decomposing the projections in binary and sending them in the outer commitment layer. As in Section 5, this version of LaBRADOR can provide approximate norm proof up to the slack factor of 9.75/0.49, (Theorem 6) so then in this case, we need

$$2 \cdot d \cdot (n \sum_{i \in [K]} \lceil \log (9.75\mathtt{B}'_i \sqrt{m}) \rceil + 6\mathtt{t}_2 \lceil \log q \rceil) \cdot \left(\frac{9.75}{0.49}\right)^2 \leq q'.$$

F More details on aggregating Falcon signatures

Recall $\mathcal{R}' = \mathbb{Z}[X]/(X^{d_F}+1)$ and $\mathcal{R}'_{q_F} = \mathbb{Z}_{q_F}[X]/(X^{d_F}+1)$, $d_F = 512$. As in Section 6, each of the nm Falcon signatures is represented as a 4×4 block of ring elements in $\mathcal{R}_q$. All these blocks will be arranged in a matrix form with n blocks per row and m blocks per column, leading to a big witness matrix of size $4n \times 4m$ over $\mathcal{R}_q$.

We recall here the Falcon block of index $\ell \in [nm]$:

$$\begin{pmatrix} \tilde{\boldsymbol{u}}_1^{(\ell)} \\ \tilde{\boldsymbol{u}}_2^{(\ell)} \\ \tilde{\boldsymbol{u}}_3^{(\ell)} \\ \tilde{\boldsymbol{u}}_4^{(\ell)} \end{pmatrix} \coloneqq \begin{pmatrix} s_1^{(e)} & s_1^{(o)} & s_2^{(e)} & s_2^{(o)} \\ 1 & 0 & h^{(e)} & Xh^{(o)} \\ 0 & 1 & h^{(o)} & h^{(e)} \\ v^{(e)} & v^{(o)} & 0 & 0 \end{pmatrix} \in \mathcal{R}_q^{4\times 4},$$

where $\|\tilde{\boldsymbol{u}}_1^{(\ell)}\|^2 \leq \lfloor \beta^2 \rfloor$. For simplicity of the exposition, we drop the notation ℓ when talking about the ℓ-th Falcon signature (s_1, s_2), public key h and the lifting v, but keep in mind they differ for any $\ell \in [nm]$.

We explain in detail below the constraints we need to feed into Orthus for aggregating the nm Falcon blocks.

For the Falcon public keys, we prove correctness of the preprocessed commitments $\boldsymbol{A}'\tilde{\boldsymbol{u}}_2^{(\ell)} = \boldsymbol{r} \in \mathcal{R}_q^{t'}$, for a public matrix $\boldsymbol{A}' \in \mathcal{R}_q^{t' \times N}$, as full constraints, by reading each row of the these commitments.

For the lifting element v, we provide an approximate norm on the last row, to make sure Equation (51) does not wrap around modulo q. Note that, from Equation (51), v can be computed as $v = \frac{1}{q_F}(t - s_1 - hs_2) \in \mathcal{R}'$, and has

$$\|v\| \leq \frac{1}{q_F}(\|t\| + \|s_1\| + \|hs_2\|) \tag{62}$$

$$\leq \beta_v \coloneqq \frac{1}{q_F}(\frac{\sqrt{d_F}q_F}{2} + \beta + \frac{d_F q_F}{2}\beta) \tag{63}$$

where we use $\|s_1\|, \|s_2\| \leq \beta$, $\|t\| \leq \|t\|_\infty \sqrt{d_F}$ and $\|hs_2\| \leq d_F \|h\|_\infty \|s_2\|$, and triangle inequality. Note that, since $t, h \in \mathcal{R}'_{q_F}$, $\|t\|_\infty, \|h\|_\infty \leq q_F/2$. We also prove correctness of the padding with 0's of the last row.

For the Falcon signature, we provide an exact norm on the first row. As in Appendix D, for this we need to consider the additional rows $\tilde{\boldsymbol{u}}_5^{(\ell)} \coloneqq [d^{(\ell)},\ 0,\ 0,\ 0]$, $\tilde{\boldsymbol{u}}_6^{(\ell)} \coloneqq \sigma_{-1}(\tilde{\boldsymbol{u}}_1^{(\ell)})$, where $d^{(\ell)} \in \mathcal{R}$ has the coefficients given by the binary decomposition of $\beta^2 - \|\tilde{\boldsymbol{u}}_1^{(\ell)}\|^2$. Here we use $\log(\beta^2) \leq d$, which is already satisfied by the choice of β and d. We also need to provide an approximate norm proof for $\tilde{\boldsymbol{u}}_1^{(\ell)}$.

For proving $d^{(\ell)}$ is a binary polynomial, we focus on proving the row $\tilde{\boldsymbol{u}}_5^{(\ell)}$ is a binary vector, and for this, as in Appendix D, we need to consider $\tilde{\boldsymbol{u}}_7^{(\ell)} \coloneqq \sigma_{-1}(\tilde{\boldsymbol{u}}_5^{(\ell)})$. We also prove correctness of the padding with 0's of this row and an approximate norm proof for $\tilde{\boldsymbol{u}}_5^{(\ell)}$, $\|\tilde{\boldsymbol{u}}_5^{(\ell)}\| \leq \sqrt{d}$.

For proving the binary and exact norm constraints (Equations (59) and (60)), which are constant-term type, we consider, as in Section 4.3 an additional row containing their liftings, in order to convert these constraints to full ones:

$$\tilde{\boldsymbol{u}}_8^{(\ell)} \coloneqq [\langle \tilde{\boldsymbol{u}}_6^{(\ell)}, \tilde{\boldsymbol{u}}_1^{(\ell)} \rangle + \sigma_{-1}(\sum_{0 \leq i \leq \lfloor \log \beta^2 \rfloor} 2^i X^i) d^{(\ell)} - \lfloor \beta^2 \rfloor, \langle \sigma_{-1}(\tilde{\boldsymbol{u}}_5^{(\ell)} - \mathbf{1}), \tilde{\boldsymbol{u}}_5^{(\ell)} \rangle, 0, 0].$$

We also have to prove the well formedness of the rows $\tilde{\boldsymbol{u}}_6^{(\ell)}, \tilde{\boldsymbol{u}}_7^{(\ell)}$ when applying the automorphism σ_{-1}, using Equation (61). This check is a constant-term type linear constraint, so no liftings are considered in additional rows.

Thus, we end up with a block of 8 rows. Rewriting Equations (51) and (59) to (61), the trusted commitments constraints and the ones for correctness of padding with 0's, the block satisfies the constraints:

$$
\begin{aligned}
&\langle \tilde{\boldsymbol{u}}_1^{(\ell)}, \tilde{\boldsymbol{u}}_2^{(\ell)} \rangle + \langle q_F \tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{u}}_4^{(\ell)} \rangle = t^{(e)},\\
&\langle \tilde{\boldsymbol{u}}_1^{(\ell)}, \tilde{\boldsymbol{u}}_3^{(\ell)} \rangle + \langle q_F \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{u}}_4^{(\ell)} \rangle = t^{(o)},\\
&\langle \boldsymbol{\varphi}_{2,i}, \tilde{\boldsymbol{u}}_2^{(\ell)} \rangle = \boldsymbol{r}_i,\ i \in [t'],\\
&\langle \tilde{\boldsymbol{u}}_6^{(\ell)}, \tilde{\boldsymbol{u}}_1^{(\ell)} \rangle + \langle \boldsymbol{\varphi}_5, \tilde{\boldsymbol{u}}_5^{(\ell)} \rangle - \lfloor \beta^2 \rfloor = \langle \tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{u}}_8^{(\ell)} \rangle,\\
&\langle \tilde{\boldsymbol{u}}_5^{(\ell)}, \tilde{\boldsymbol{u}}_7^{(\ell)} \rangle - \langle \boldsymbol{\varphi}'_5, \tilde{\boldsymbol{u}}_5^{(\ell)} \rangle = \langle \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{u}}_8^{(\ell)} \rangle,\\
&\langle \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{u}}_4^{(\ell)} \rangle = 0,\ i \in \{3,4\},\\
&\langle \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{u}}_5^{(\ell)} \rangle = 0,\ i \in \{2,3,4\},\\
&\mathtt{ct}(\langle \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{u}}_8^{(\ell)} \rangle) = 0,\ i \in \{1,2\},\\
&\mathtt{ct}(\langle X^{d-k} \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{u}}_1^{(\ell)} \rangle - \langle X^k \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{u}}_6^{(\ell)} \rangle) = 0,\ k \in [d],\ i \in [4],\\
&\mathtt{ct}(\langle X^{d-k} \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{u}}_5^{(\ell)} \rangle - \langle X^k \tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{u}}_7^{(\ell)} \rangle) = 0,\ k \in [d],\ i \in [4],\\
&\|\tilde{\boldsymbol{u}}_1^{(\ell)}\| \le \mathtt{B}'_1 \coloneqq \beta,\ \|\tilde{\boldsymbol{u}}_4^{(\ell)}\| \le \mathtt{B}'_4 \coloneqq \beta_v,\ \|\tilde{\boldsymbol{u}}_5^{(\ell)}\| \le \mathtt{B}'_5 \coloneqq \sqrt{d},
\end{aligned}
$$

where $\boldsymbol{\varphi}_{2,i}$ are the rows of the matrix $\boldsymbol{A}'$, $\boldsymbol{\varphi}_5 \coloneqq [\sigma_{-1}(\sum_{i\in[d]} 2^{i-1}X^{i-1}),\ 0,\ 0,\ 0]$, $\boldsymbol{\varphi}'_5 \coloneqq [\sigma_{-1}(\sum_{i\in[d]} X^{i-1}),\ 0,\ 0,\ 0]$, and $\tilde{\boldsymbol{e}}_i \in \mathcal{R}^4$ is the unit vector having the i-th entry equal to 1 and the others equal to 0, for any $i \in [4]$.

Modulus q. From Appendix E and Theorem 6, we have $\max_{i\in\{1,4,5\}} 61 \cdot \frac{9.75}{0.49}\mathtt{B}'_i\sqrt{m} \le q$. We get further requirements for the Orthus modulus q, as showing up from proving binary and exact-norm constraints (as in Appendix D) and from proving the Falcon verification Equation (50), all not wrapping around modulo q.

From Section 5, we prove with Orthus approximate norm bounds with a slack factor $\frac{9.75}{0.49}\sqrt{m}$, namely $\|(s_1,s_2)\| = \|\tilde{\boldsymbol{u}}_1^{(\ell)}\| \le \frac{9.75}{0.49}\cdot\beta\sqrt{m}$, $\|v\| = \|\tilde{\boldsymbol{u}}_4^{(\ell)}\| \le \frac{9.75}{0.49}\cdot\beta_v\sqrt{m}$, $\|d^{(\ell)}\| = \|\tilde{\boldsymbol{u}}_5^{(\ell)}\| \le \frac{9.75}{0.49}\cdot\sqrt{dm}$ and the following:

$$s_1^{(e)} + h^{(e)}s_2^{(e)} + Xh^{(o)}s_2^{(o)} + q_F v^{(e)} = t^{(e)}, \tag{64}$$

$$s_1^{(o)} + h^{(o)}s_2^{(e)} + h^{(e)}s_2^{(o)} + q_F v^{(o)} = t^{(o)}. \tag{65}$$

$$\mathtt{ct}(\langle \sigma_{-1}(s_1,s_2), (s_1,s_2)\rangle + \sigma_{-1}(\sum_{i\in[\lfloor \log \beta^2 \rfloor}} 2^{i-1}X^{i-1}) \cdot d^{(\ell)} - \lfloor \beta^2 \rfloor) = 0, \tag{66}$$

$$\mathtt{ct}(\sigma_{-1}(d^{(\ell)} - \sum_{i\in[d]} X^{i-1}) \cdot d^{(\ell)}) = 0. \tag{67}$$

For Equation (67) to hold over $\mathbb{Z}$, as in Appendix D, we impose $2\|d^{(\ell)}\|^2 \le q$. This leads to $2\left(\frac{9.75}{0.49}\right)^2 \cdot dm \le q$. This would prove indeed that the polynomial $d^{(\ell)}$ is binary.

Given that the polynomial $d^{(\ell)}$ is now binary, for Equation (66) to hold over $\mathbb{Z}$, as in Appendix D, we impose $\|\tilde{\boldsymbol{u}}_1^{(\ell)}\| + 2\beta^2 = \|(s_1,s_2)\|^2 + 2\beta^2 \le q$. If we set $\left(\frac{9.75}{0.49}\right)^2 \cdot \beta^2 m + 2\beta^2 \le q$, this shows that $\|(s_1,s_2)\| \le \beta$. In particular, we get $\|s_1\|, \|s_2\| \le \beta$.

Equations (64) and (65) lead to Equation (50). For this to hold over $\mathbb{Z}$, we can assume $\|t\|_\infty \le q_F/2 \le q/2$. Then we focus on the infinity norm of the $s_1 + hs_2 + q_F v - t$ from Equation (50) to be less than q. Since $\|s_1\|_\infty \le \|s_1\| \le \beta$, $\|hs_2\|_\infty \le \|h\|\|s_2\| \le q_F/2 \cdot \sqrt{d_F} \cdot \beta$, $\|v\|_\infty \le \|v\| \le \frac{9.75}{0.49}\beta_v\sqrt{m}$, using triangle inequality we get

$$\|s_1 + hs_2 + q_F v - t\|_\infty \le \beta + \frac{q_F}{2}\sqrt{d_F}\beta + q_F \cdot \frac{9.75}{0.49} \cdot \beta_v\sqrt{m} + \frac{q_F}{2}.$$

Therefore, it suffices to set $\beta + \frac{q_F}{2}\sqrt{d_F}\beta + q_F \cdot \frac{9.75}{0.49} \cdot \beta_v\sqrt{m} + \frac{q_F}{2} \le q$, to make sure the Falcon equation does not wrap around.

Note that the norm bound β_v to be proven on v is set in Equation (62) as a loose worst-case bound. In practice, we derive a tighter norm from the expected norms of s_1 and s_2, and check it experimentally.

G Figures for protocols

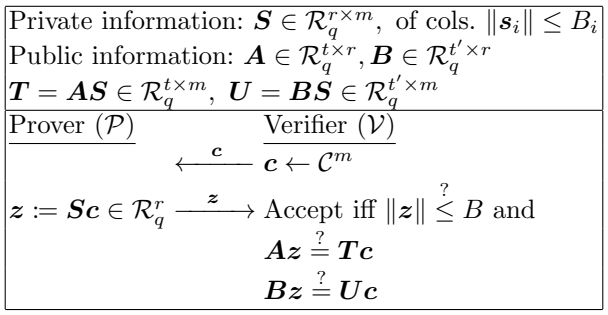

Fig. 2. The Linear Proof $\pi_{\mathtt{lin}}((\boldsymbol{T}, (B_i)_i, \boldsymbol{B}, \boldsymbol{U}), \boldsymbol{S})$: Proof of knowledge for relation $R_{\mathtt{lin}}$.

Private information: $\boldsymbol{S} \in \mathcal{R}_q^{r\times m}$, of cols. $\|\boldsymbol{s}_i\| \leq B_i$
Public information: $\boldsymbol{A} \in \mathcal{R}_q^{t\times r}, \boldsymbol{T} = \boldsymbol{AS}$
$\boldsymbol{R} \in \mathcal{R}_q^{n\times r}$ of rows $\tilde{\boldsymbol{r}}_i, \boldsymbol{D} \in \mathcal{R}_q^{m\times m}, \boldsymbol{\varphi} \in \mathcal{R}_q^m, (u_i, w_i)_i \in \mathcal{R}_q^{2n}$
$\langle \tilde{\boldsymbol{r}}_i \boldsymbol{SD}, \tilde{\boldsymbol{r}}_i \boldsymbol{S}\rangle = u_i,\ i \in [n]$,
$\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{r}}_i \boldsymbol{S}\rangle = w_i,\ i \in [n]$

Prover ($\mathcal{P}$)		Verifier ($\mathcal{V}$)
	$\xleftarrow{(\beta_i)}$	$(\beta_i) \leftarrow \mathcal{R}_q^n$
For $-n < i < n,\ i \neq 0$		
$v_i \coloneqq \sum_{\substack{k,j\in[n]\\ k-j=i}} \beta_k \langle \tilde{\boldsymbol{r}}_k \boldsymbol{SD}, \tilde{\boldsymbol{r}}_j \boldsymbol{S}\rangle$	$\xrightarrow{(v_i)}$	
$v_i \in \mathcal{R}_q$		
	$\xleftarrow{\theta}$	$\theta \leftarrow \hat{\mathcal{C}}$
$\tilde{\boldsymbol{z}}_+ \coloneqq \sum_{i\in[n]} \beta_i \theta^i \tilde{\boldsymbol{r}}_i \boldsymbol{S} \in \mathcal{R}_q^m$		
$\tilde{\boldsymbol{z}}_- \coloneqq \sum_{i\in[n]} \theta^{-i} \tilde{\boldsymbol{r}}_i \boldsymbol{S} \in \mathcal{R}_q^m$	$\xrightarrow{\tilde{\boldsymbol{z}}_+, \tilde{\boldsymbol{z}}_-}$	
Both $\mathcal{P}$ and $\mathcal{V}$ compute:		
$\boldsymbol{B} \coloneqq \begin{pmatrix} [\theta\beta_1, \ldots, \theta^n \beta_n]\boldsymbol{R} \\ [\theta^{-1}, \ldots, \theta^{-n}]\boldsymbol{R} \end{pmatrix}$		
$\boldsymbol{U} \coloneqq \begin{pmatrix} \tilde{\boldsymbol{z}}_+ \\ \tilde{\boldsymbol{z}}_- \end{pmatrix}$		
$\boldsymbol{B} \in \mathcal{R}_q^{2\times r},\ \boldsymbol{U} \in \mathcal{R}_q^{2\times m}$		
Both $\mathcal{P}$ and $\mathcal{V}$ run:		
$\pi \coloneqq \pi_{\mathtt{lin}}((\boldsymbol{T}, (B_i)_i, \boldsymbol{B}, \boldsymbol{U}), \boldsymbol{S})$		Accept iff π verifies and $\langle \tilde{\boldsymbol{z}}_+ \boldsymbol{D}, \tilde{\boldsymbol{z}}_-\rangle \stackrel{?}{=} \sum_{i\in[n]} \beta_i u_i + \sum_{\substack{-n<i<n\\ i\neq 0}} v_i \theta^i$ and $\langle \boldsymbol{\varphi}, \tilde{\boldsymbol{z}}_-\rangle \stackrel{?}{=} \sum_{i\in[n]} w_i \theta^{-i}$.

Fig. 3. The Left-Right protocol $\pi_{\mathtt{LR}}((\boldsymbol{T}, (B_i)_i, \boldsymbol{R}, \boldsymbol{D}, \boldsymbol{\varphi}, (u_i, w_i)_i), \boldsymbol{S})$: Proof of knowledge for relation $R_{\mathtt{LR}}$.

Private information: $\boldsymbol{S} \in \mathcal{R}_q^{nK \times m}$, of cols. $\|\boldsymbol{s}_j\| \le B_j$, and of rows $\tilde{\boldsymbol{s}}_i^{(\ell)}$, $i \in [K]$, $\ell \in [n]$ Public information: $\boldsymbol{A} \in \mathcal{R}_q^{t \times nK}$, $\boldsymbol{T} = \boldsymbol{AS}$ $a_{ij} \in \mathcal{R}_q, \boldsymbol{\varphi}_i \in \mathcal{R}_q^m, b^{(\ell)} \in \mathcal{R}_q$, $\boldsymbol{D} = \boldsymbol{D}^T \in \mathcal{R}_q^{m \times m}$ $b^{(\ell)} = \sum_{i,j \in [K]} a_{ij} \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \sum_{i \in [K]} \langle \boldsymbol{\varphi}_i, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle$		
Prover $(\mathcal{P})$		Verifier $(\mathcal{V})$
$\boldsymbol{g}_{ij}^{(\ell)} \coloneqq \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle \in \mathcal{R}_q$	$\xrightarrow{(\boldsymbol{g}_{ij}^{(\ell)})}$	
$\boldsymbol{h}_{ij}^{(\ell)} \coloneqq \frac{1}{2}(\langle \boldsymbol{\varphi}_i, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \langle \boldsymbol{\varphi}_j, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle) \in \mathcal{R}_q$	$\xrightarrow{(\boldsymbol{h}_{ij}^{(\ell)})}$	
Both $\mathcal{P}$ and $\mathcal{V}$ compute:	$\xleftarrow{(c_i)_i}$	$c_1, \ldots, c_K \leftarrow C$
$\boldsymbol{R}$ of rows $\tilde{\boldsymbol{r}}_\ell \in \mathcal{R}_q^{nK}$ $\tilde{\boldsymbol{r}}_\ell \coloneqq [\boldsymbol{0}_K \vert \ldots \vert c_1, \ldots, c_K \vert \ldots \vert \boldsymbol{0}_K]$ $\boldsymbol{\varphi} \coloneqq \sum_{i \in [K]} c_i \boldsymbol{\varphi}_i \in \mathcal{R}_q^m$ $u_\ell \coloneqq \sum_{i,j \in [K]} \boldsymbol{g}_{ij}^{(\ell)} c_i c_j \in \mathcal{R}_q$ $w_\ell \coloneqq \sum_{i,j \in [K]} \boldsymbol{h}_{ij}^{(\ell)} c_i c_j \in \mathcal{R}_q$		
Both $\mathcal{P}$ and $\mathcal{V}$ run: $\pi \coloneqq \pi_{\mathtt{LR}}((\boldsymbol{T}, (B_i)_i, \boldsymbol{R}, \boldsymbol{D}, \boldsymbol{\varphi}, (u_\ell, w_\ell)_\ell), \boldsymbol{S})$		Accept iff π verifies and $\forall \ell \in [n]$ $\sum_{i,j \in [K]} a_{ij} \boldsymbol{g}_{ij}^{(\ell)} + \sum_{i \in [K]} \boldsymbol{h}_{ii}^{(\ell)} \stackrel{?}{=} b^{(\ell)}$

Fig. 4. The protocol $\pi^1_{\mathtt{row}}((\boldsymbol{T}, (B_i)_i, (F^{(\ell)})_\ell, \boldsymbol{D}), \boldsymbol{S})$: Proof of knowledge for relation $R_{\mathtt{row\text{-}block\text{-}1}}$.

H Proving Claim for Johnson-Lindenstrauss Lemma (Theorem 6)

Claim. $\Pr_{\mathbf{r} \leftarrow \mathtt{Bin}_1}[|\langle \mathbf{r}, \mathbf{w} \rangle| < 0.49\|\mathbf{w}\|] \le 0.70 < \frac{1}{\sqrt{2}} = 0.70710678\ldots$, for any nonzero vector $\mathbf{w} \in \mathbb{R}^n$.

Throughout this section, $\mathbf{r} = (r_1, \ldots, r_n)$ has independent entries $r_i \leftarrow \mathtt{Bin}_1$, that is, $r_i = 0$ with probability $\frac{1}{2}$ and $r_i = \pm 1$ with probability $\frac{1}{4}$ each, and $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_n) \in \mathbb{R}^n$. We denote by $X \coloneqq \langle \mathbf{r}, \mathbf{w} \rangle = \sum_{i \in [n]} r_i \mathbf{w}_i$, $p(\mathbf{w}) \coloneqq \Pr[|X| < 0.49]$, and $\Phi_X(t) \coloneqq E[\cos(tX)]$.

Note that $|X|$ has the same distribution if the entries of $\mathbf{w}$ are permuted, since the r_i are independent and identically distributed. It is also the case when the sign of some $\mathbf{w}_i$ is flipped, as r_i and $-r_i$ have the same distribution. Moreover, by scaling, it suffices to treat unit vectors. Therefore, without loss of generality, we assume $\mathbf{w}_1 \ge \mathbf{w}_2 \ge \ldots \ge \mathbf{w}_n \ge 0$, and $\sum_{i \in [n]} \mathbf{w}_i^2 = 1$.

Proving the claim above follows this line of thought. We write the probability we are interested in as the expectation of an indicator function, we bound the indicator from above by a trigonometric function whose expectation can be computed or bounded explicitly, and we choose its coefficients by linear programming so that this expectation is below $1/\sqrt{2}$. We first collect the properties of these trigonometric functions.

H.1 Some preliminary lemmas

The next lemma computes Φ_X, and upper bounds it in terms of $\|\mathbf{w}\|_\infty$.

Lemma 7. *Let $X = \sum_{i \in [n]} r_i \mathbf{w}_i$, for r_i independently drawn from $\mathtt{Bin}_1$ and a vector $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_n) \in \mathbb{R}^n$. Then for any real number t, $E[\sin(tX)] = 0$ and $\Phi_X(t) = E[\cos(tX)] = \prod_{i \in [n]} \cos^2\left(\frac{t\mathbf{w}_i}{2}\right) \in [0, 1]$. Moreover,*

$$\Phi_X(t) \le \exp\Big(-\frac{t^2}{4} \sum_{i \in [n]} \mathbf{w}_i^2\Big) \qquad \text{for every } |t| \le \frac{\pi}{\max_{i \in [n]} |\mathbf{w}_i|};$$

Private information: $\boldsymbol{S} \in \mathcal{R}_q^{nK \times m}$, of cols. $\|\boldsymbol{s}_j\| \leq B_j$, rows $\tilde{\boldsymbol{s}}_i^{(\ell)}$

Public information: $\boldsymbol{A} \in \mathcal{R}_q^{t \times nK}$, $\boldsymbol{T} = \boldsymbol{A}\boldsymbol{S}$

$a_{ij}^{(k)}, a_{ij}^{\prime(l)} \in \mathcal{R}_q$, $\boldsymbol{\varphi}_i^{(k)}, \boldsymbol{\varphi}_i^{(l)} \in \mathcal{R}_q^m$, $b_k \in \mathcal{R}_q$, $b'_{0,l} \in \mathbb{Z}_q$, $\boldsymbol{D} = \boldsymbol{D}^T \in \mathcal{R}_q^{m \times m}$

$$f^{(k)} : b_k = \sum_{i,j \in [K]} a_{ij}^{(k)} \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \sum_{i \in [K]} \langle \boldsymbol{\varphi}_i^{(k)}, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle, \ f^{(k)} \in [|\mathcal{F}|]$$

$$f'^{(l)} : b'_{0,l} = \mathtt{ct}\Big(\sum_{i,j \in [K]} a_{ij}^{\prime(l)} \langle \tilde{\boldsymbol{s}}_i^{(\ell)} \boldsymbol{D}, \tilde{\boldsymbol{s}}_j^{(\ell)} \rangle + \sum_{i \in [K]} \langle \boldsymbol{\varphi}_i^{\prime(l)}, \tilde{\boldsymbol{s}}_i^{(\ell)} \rangle \Big), \ f'^{(l)} \in \mathcal{F}'$$

$\|\tilde{\boldsymbol{s}}_1^{(\ell)}\| \leq \mathtt{B}_1, \ldots, \|\tilde{\boldsymbol{s}}_K^{(\ell)}\| \leq \mathtt{B}_K$

Prover ($\mathcal{P}$)		Verifier ($\mathcal{V}$)
	$\xleftarrow{\boldsymbol{\Pi} = (\pi^{(j)})}$	$\boldsymbol{\Pi} \leftarrow \mathtt{Bin}_1^{256 \times m}$
$(p_{i,j})_j^{(\ell)} \coloneqq \boldsymbol{\Pi} s_i^{(\ell)} \in \mathbb{Z}_q^{256}$	$\xrightarrow{(p_i^{(\ell)})}$	$\|p_i^{(\ell)}\| \overset{?}{\leq} \sqrt{128} B_i$
		$(\psi_i^{(k)}) \leftarrow \mathbb{Z}_q^{\lvert\mathcal{F}'\rvert}$
Compute $a_{ij}^{\prime\prime(k)}, \boldsymbol{\varphi}_i^{\prime\prime(k)}, b_k^{\prime\prime(\ell)}$	$\xleftarrow{(\psi_i^{(k)}), (\omega_i^{(k)})}$	Compute $a_{ij}^{\prime\prime(k)}, \boldsymbol{\varphi}_i^{\prime\prime(k)}$
as in (52), (53), (54)	$\xrightarrow{(b_k^{\prime\prime(\ell)})}$	$\mathtt{ct}(b_k^{\prime\prime(\ell)}) \overset{?}{=} \sum_{l \in [\lvert\mathcal{F}'\rvert]} \psi_l^{(k)} b'_{0,l}$
$a_{ij}^{\prime\prime(k)}, b_k^{\prime\prime(\ell)} \in \mathcal{R}_q$, $\boldsymbol{\varphi}_i^{\prime\prime(k)} \in \mathcal{R}_q^m$		$+ \sum_{\substack{i \in [K] \\ j \in [256]}} \omega_{256(i-1)+j}^{(k)} p_{i,j}^{(\ell)}$
		$(\mu_i) \leftarrow \mathcal{R}_q^{\lvert\mathcal{F}\rvert}$
Both $\mathcal{P}$ and $\mathcal{V}$ compute	$\xleftarrow{(\mu_i), (\nu_j)}$	$(\nu_i) \leftarrow \mathcal{R}_q^{\lceil 128/\log q \rceil}$
$F^{(\ell)}$ described by $a_{ij}, \boldsymbol{\varphi}_i, b^{(\ell)}$		
as in (56), (57), (58)		
$a_{ij}, b^{(\ell)} \in \mathcal{R}_q$, $\boldsymbol{\varphi}_i \in \mathcal{R}_q^m$		
Both $\mathcal{P}$ and $\mathcal{V}$ run:		
$\pi \coloneqq \pi^1_{\mathtt{row}}((\boldsymbol{T}, (B_i)_i, (F^{(\ell)})_\ell, \boldsymbol{D}), \boldsymbol{S})$		Accept iff π verifies

Fig. 5. The Row-Block protocol $\pi^{\mathtt{many}}_{\mathtt{row}}((\boldsymbol{T}, (B_i)_i, \mathcal{F}, \mathcal{F}', (\mathtt{B}_i)_i, \boldsymbol{D}), \boldsymbol{S})$: Proof of knowledge for $R_{\mathtt{row\text{-}block\text{-}many}}$.

Private information: $\boldsymbol{S} \in \mathcal{R}_q^{nK \times mN}$, made of nm blocks of rows $\tilde{\boldsymbol{u}}_i^{(\ell)}$	
Public information: $\boldsymbol{A} \in \mathcal{R}_q^{t \times nK}$	
$f^{(k)}(\tilde{\boldsymbol{u}}_1^{(\ell)}, \dots, \tilde{\boldsymbol{u}}_K^{(\ell)}) = 0,\ f^{(k)} \in \mathcal{F}$	
$\mathtt{ct}(f'^{(l)}(\tilde{\boldsymbol{s}}_1^{(\ell)}, \dots, \tilde{\boldsymbol{s}}_K^{(\ell)})) = 0,\ f'^{(l)} \in \mathcal{F}'$	
$\|\tilde{\boldsymbol{u}}_1^{(\ell)}\| \le \mathtt{B}'_1, \dots, \|\tilde{\boldsymbol{u}}_K^{(\ell)}\| \le \mathtt{B}'_K$	
Prover ($\mathcal{P}$)	Verifier ($\mathcal{V}$)
Extend $\boldsymbol{S}$ with	
$\tilde{\boldsymbol{u}}_{K+1}^{(\ell)} \coloneqq [(f'(\tilde{\boldsymbol{u}}_1^{(\ell)}, \dots, \tilde{\boldsymbol{u}}_K^{(\ell)}))_{f' \in \mathcal{F}'}, 0, \dots, 0] \in \mathcal{R}_q^N$	
$\boldsymbol{T} \coloneqq \boldsymbol{A}\boldsymbol{S} \in \mathcal{R}_q^{t \times mN}$	$\xrightarrow{\boldsymbol{T}}$
Both $\mathcal{P}$ and $\mathcal{V}$ compute:	$\xleftarrow{(\boldsymbol{\alpha}_i)}$ $(\alpha_i) \leftarrow \mathcal{R}_q^m$
$\bar{\mathcal{F}}_\alpha$ as in (46), (47)	
$\bar{\mathcal{F}}'_{\boldsymbol{\alpha}}$ as in (48)	
$\boldsymbol{D} \coloneqq \mathrm{diag}(\alpha_1 \boldsymbol{I}_N, \dots, \alpha_m \boldsymbol{I}_N) \in \mathcal{R}_q^{mN \times mN}$	
B_i such that $\|\boldsymbol{s}_i\| \le B_i$	
Both $\mathcal{P}$ and $\mathcal{V}$ run:	
$\pi \coloneqq \pi^{\mathtt{many}}_{\mathtt{row}}((\boldsymbol{T}, (B_i)_i, \bar{\mathcal{F}}_\alpha, \bar{\mathcal{F}}'_\alpha, (\mathtt{B}'_i\sqrt{m})_i, \boldsymbol{D}), \boldsymbol{S})$	Accept iff π verifies

Fig. 6. The main protocol Orthus: Proof of knowledge for relation $R_{\mathtt{0}}$.

in particular, $\Phi_X(t) \le \Gamma(t) \coloneqq \exp(-t^2/4)$ *for such* t *whenever* $\mathbf{w}$ *is a unit vector.*

Proof. Since $r_i \leftarrow \mathtt{Bin}_1$, for any $i \in [n]$,

$$E[\cos(tr_i\mathbf{w}_i)] = \frac{1}{4}\cos(-t\mathbf{w}_i) + \frac{1}{2}\cos(0) + \frac{1}{4}\cos(t\mathbf{w}_i) = \frac{1}{2}\cos(t\mathbf{w}_i) + \frac{1}{2} = \cos^2\left(\frac{t\mathbf{w}_i}{2}\right), \tag{68}$$

$$E[\sin(tr_i\mathbf{w}_i)] = \frac{1}{4}\sin(-t\mathbf{w}_i) + \frac{1}{2}\sin(0) + \frac{1}{4}\sin(t\mathbf{w}_i) = 0. \tag{69}$$

Let $X_k \coloneqq \sum_{i\in[k]} r_i\mathbf{w}_i$ for $k \in [n]$, so that $X = X_n$. We prove by induction on k that $E[\cos(tX_k)] = \prod_{i\in[k]} \cos^2(\frac{t\mathbf{w}_i}{2})$ and $E[\sin(tX_k)] = 0$. The case $k = 1$ is Eqs. (68) and (69). For the inductive step, write $X_k = X_{k-1} + r_k\mathbf{w}_k$, where X_{k-1} and $r_k\mathbf{w}_k$ are independent, and expand

$$E[\cos(tX_k)] = E[\cos(tX_{k-1})] \cdot E[\cos(tr_k\mathbf{w}_k)] - E[\sin(tX_{k-1})] \cdot E[\sin(tr_k\mathbf{w}_k)] \tag{70}$$

$$E[\sin(tX_k)] = E[\sin(tX_{k-1})] \cdot E[\cos(tr_k\mathbf{w}_k)] + E[\cos(tX_{k-1})] \cdot E[\sin(tr_k\mathbf{w}_k)] \tag{71}$$

Equations (68) and (69) and the induction hypothesis close the induction. As $\Phi_X(t)$ is a product of squares, we get $\Phi_X(t) \in [0, 1]$.

For the second part, note that the function $f(x) = \ln(\cos(x)) + \frac{x^2}{2}$ is decreasing on $[0, \frac{\pi}{2})$ (its derivative is ≤ 0 on $[0, \frac{\pi}{2})$), and thus $f(x) \le f(0) = 0$. This shows that $\cos(x) \le \exp(-x^2/2)$, for $x \in [0, \frac{\pi}{2}]$, and as $\cos(\cdot)$ is an even function, we get $\cos^2(x) \le \exp(-x^2)$, for $|x| \le \frac{\pi}{2}$. If $|\frac{t\mathbf{w}_i}{2}| \le \frac{\pi}{2}$ for every $i \in [n]$, then $\Phi_X(t) = E[\cos(tX)] \le \prod_{i\in[n]} \exp(-(\frac{t\mathbf{w}_i}{2})^2) = \exp(-\frac{t^2}{4} \cdot \sum_{i\in[n]} \mathbf{w}_i^2)$.

We call $|t| \le \pi/\max_i|\mathbf{w}_i|$ the *usable range*; outside this range, we only use $\Phi_X(t) \le 1$. The next lemma introduces useful functions for proving the claim, whose expectation is also explicit.

Lemma 8. *Consider the function* $B(z) = 1 + \cos(2\pi(z-c)/L)$*, called a bump function, with spacing* $L > 0$ *and center* $c \in \mathbb{R}$*. Let* $X = \sum_{i\in[n]} r_i\mathbf{w}_i$*, for* r_i *independently drawn from* $\mathtt{Bin}_1$ *and a vector* $\mathbf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_n) \in \mathbb{R}^n$*. Then:*

1. $0 \le B(z) \le 2$ *for every* $z \in \mathbb{R}$*, with* $B(z) = 2$ *exactly at the* local maximum points $z = c + jL$*,* $j \in \mathbb{Z}$*, and* $B(z) = 0$ *exactly at the* zeros $z = c + (j + \frac{1}{2})L$*,* $j \in \mathbb{Z}$*, the midpoints between consecutive local maximum points;*

2. $E[B(X)] = 1 + \cos(2\pi c/L)\,\Phi_X\left(\frac{2\pi}{L}\right) \le 1 + \max(0,\ \cos(2\pi c/L))\,\Gamma$, *where* Γ *is any upper bound for* $\Phi_X(\frac{2\pi}{L})$*; by Lemma 7 one may take* $\Gamma = \exp(-\pi^2(\sum_{i\in[n]}\mathbf{w}_i^2)/L^2)$ *as soon as* $L \ge 2\max_{i\in[n]}|\mathbf{w}_i|$*, and* $\Gamma = 1$ *otherwise. We call the upper-bound of the expected value the* price *of the bump;*
3. *if an interval* $[p,q]$ *contains no zero of* B*, that is, if* $[p,q] \subseteq (c + jL - \frac{L}{2},\ c + jL + \frac{L}{2})$ *for some* $j \in \mathbb{Z}$*, and* $d \coloneqq \max(|p - c - jL|, |q - c - jL|)$*, then* $B(z) \ge 1 + \cos(2\pi d/L)$*, for all* $z \in [p,q]$.

Proof. Item 1 is immediate. For Item 2, write $\cos\left(\frac{2\pi(X-c)}{L}\right) = \cos\left(\frac{2\pi c}{L}\right)\cos\left(\frac{2\pi X}{L}\right) + \sin\left(\frac{2\pi c}{L}\right)\sin\left(\frac{2\pi X}{L}\right)$ and take expectations; the second term vanishes by Lemma 7, since $E[\sin(\frac{2\pi X}{L})] = 0$, which gives the identity. For the inequality, $\Phi_X(\frac{2\pi}{L}) \in [0,\Gamma]$ by Lemma 7: if $\cos(2\pi c/L) \ge 0$ we use the upper bound $\Phi_X \le \Gamma$, and if $\cos(2\pi c/L) < 0$ we use only $\Phi_X \ge 0$, which makes the term at most 0. The condition $L \ge 2\max_i|\mathbf{w}_i|$ is exactly the condition $\frac{2\pi}{L} \le \frac{\pi}{\max_i|\mathbf{w}_i|}$ that puts the frequency $\frac{2\pi}{L}$ inside the usable range of Lemma 7. For Item 3, B has period L, so $B(z) = 1 + \cos(2\pi(z - c - jL)/L)$; for $z \in [p,q]$ we have $|z - c - jL| \le d < \frac{L}{2}$, hence $\frac{2\pi|z-c-jL|}{L} \le \frac{2\pi d}{L} < \pi$, and since $\cos(\cdot)$ is even and decreasing on $[0,\pi]$ the claim follows.

We repeatedly use the fact that the probability $p(\mathbf{w})$ of the event $|X| < 0.49$ can be written as an expectation of the indicator of this event, namely,

$$p(\mathbf{w}) = \Pr[|X| < 0.49] = E\big[\mathbf{1}_{\{x:\ |x|<0.49\}}(X)\big],$$

where the indicator function $\mathbf{1}_A$ takes value 1 for any $x \in A$ and 0 otherwise, for a set $A \subseteq \mathbb{R}$. The following lemma makes this expression more precise by splitting X into two independent parts: a *head*, carrying the m largest coordinates of $\mathbf{w}$, and a *tail*, carrying the rest, and averaging over the head, which takes only 3^m values, leaving the expectation of an explicit function of the tail.

Lemma 9. *Fix* $m \le n$ *and split* $X = \sum_{i\in[n]} r_i\mathbf{w}_i$*, for* r_i *independently sampled from* $\mathtt{Bin}_1$ *and* $\mathbf{w} \in \mathbb{R}^n$ *as* $X = H + T$*, where* $H \coloneqq \sum_{i\le m} r_i\mathbf{w}_i$ *and* $T \coloneqq \sum_{j>m} r_j\mathbf{w}_j$*. For a pattern* $\mathbf{a} = (a_1,\dots,a_m) \in \{-1,0,1\}^m$ *write* $p_{\mathbf{a}} \coloneqq \Pr_{r_i\leftarrow\mathtt{Bin}_1}[(r_1,\dots,r_m) = \mathbf{a}]$ *and* $H_{\mathbf{a}} \coloneqq \sum_{i\le m} a_i\mathbf{w}_i$*. Then*

$$p(\mathbf{w}) = E[g(T)], \qquad g(y) \coloneqq \sum_{\mathbf{a}\in\{-1,0,1\}^m} p_{\mathbf{a}} \cdot \mathbf{1}_{\{h:\ |y+h|<0.49\}}(H_{\mathbf{a}}). \tag{72}$$

Proof. As the r_i's are independent, so are H and T. We rewrite $p(\mathbf{w})$ as follows

$$p(\mathbf{w}) = \Pr[|X| < 0.49] = E\big[\mathbf{1}_{\{x:|x|<0.49\}}(H+T)\big] = E_T\Big[\,E_H\big[\mathbf{1}_{\{h:\ |y+h|<0.49\}}(H)\big]\Big|_{y=T}\Big].$$

Let us identify the inner expectation. H is determined by $(r_1,\dots,r_m)$: on the event $\{(r_1,\dots,r_m) = \mathbf{a}\}$, it takes the value $H_{\mathbf{a}} = \sum_{i\in[m]} a_i\mathbf{w}_i$ with probability $p_{\mathbf{a}} = \prod_{i\in[m]}\Pr[r_i = a_i] = (1/2)^{m-|\mathrm{supp}(\mathbf{a})|}(1/4)^{|\mathrm{supp}(\mathbf{a})|}$, where $\mathrm{supp}(\mathbf{a}) \coloneqq \{i \in [m] : \mathbf{a}_i \ne 0\}$. Here we used that the r_i's are independent. The 3^m events partition the space, so for every fixed $y \in \mathbb{R}$, $E_H\big[\mathbf{1}_{\{h:\ |y+h|<0.49\}}(H)\big] = \sum_{\mathbf{a}\in\{-1,0,1\}^m} p_{\mathbf{a}} \cdot \mathbf{1}_{\{h:\ |y+h|<0.49\}}(H_{\mathbf{a}}) = g(y)$. Substituting $y = T$ and taking the outer expectation gives Equation (72).

H.2 Proof of the Claim

Proof. We consider two cases, depending on $\|\mathbf{w}\|_\infty$.

Case 1. Assume $\|\mathbf{w}\|_\infty \le 0.3$, i.e. $|\mathbf{w}_i| \le 0.3$ for every $i \in [n]$. Recall that $p(\mathbf{w}) = E[\mathbf{1}_{\{x:\ |x|<0.49\}}(X)]$. We bound the indicator $\mathbf{1}_{\{x:\ |x|<0.49\}}$ from above by some function $F_1(x) \coloneqq \alpha_0 + \sum_{k\in[M]} \alpha_k \cdot \cos(k\omega x)$, for a small number of terms $M = 12$, a *frequency* $\omega = 27/32$ and real coefficients α_k set by linear programming. Note that F_1 is periodic with period $L \coloneqq 2\pi/\omega$. If

$$F_1(x) \ge 1, \text{ for } |x| < 0.49, \tag{73}$$

$$F_1(x) \ge 0, \text{ for any } x. \tag{74}$$

then certainly F_1 upper bounds the indicator $\mathbf{1}_{\{x:\ |x|<0.49\}}$ and one can apply expectation:

$$p(\mathbf{w}) = E[\mathbf{1}_{\{x:\ |x|<0.49\}}(X)] \leq E[F_1(X)]. \tag{75}$$

Thus it suffices to construct F_1 such that $E[F_1(X)] \leq 1/\sqrt{2}$ to prove the claim in this case.

By definition of F_1 and Φ_X, $E[F_1(X)] = \alpha_0 + \sum_{k\in[M]} \alpha_k \Phi_X(k\omega)$. Let us look at $\alpha_k \Phi_X(k\omega)$. By Lemma 7, for $k \in [M]$, if $\alpha_k < 0$, then $\alpha_k \Phi_X(k\omega) \leq 0$, as $\Phi_X(k\omega) \geq 0$, whereas if $\alpha_k \geq 0$, then $\alpha_k \Phi_X(k\omega) \leq \alpha_k \exp(-\frac{(k\omega)^2}{4})$, using here $\sum_{i\in[n]} \mathbf{w}_i^2 = 1$ and assuming $|k\omega| \leq \frac{\pi}{0.3} = 10.47 \leq \frac{\pi}{\max_i \mathbf{w}_i}$. For $\omega = 27/32$ and $k \leq M = 12$, the latter inequality already holds, as $12\omega = 10.125$. Then, using Eq. (75),

$$p(\mathbf{w}) \leq E[F_1(X)] \leq \alpha_0 + \sum_{k\in[M]:\ \alpha_k>0} \alpha_k \exp(-\frac{(k\omega)^2}{4}) = \alpha_0 + \sum_{k\in[M]:\ \alpha_k>0} \alpha_k \Gamma(k\omega). \tag{76}$$

It suffices then to look for coefficients α_k such that the sum in Eq. (76) does not exceed $1/\sqrt{2}$. Note that the negative coefficients α_k disappeared, but they are chosen such that Equations (73) and (74) hold. The constant coefficient α_0 cannot be small, as it equals the mean value of F_1 over any period $[a, a+L]$, $a \in \mathbb{R}$: indeed, for any integer $k \geq 1$,

$$\frac{1}{L}\int_a^{a+L} \cos(k\omega x)\, dx = \frac{1}{L} \cdot \frac{\sin(k\omega(a+L)) - \sin(k\omega a)}{k\omega} = 0, \tag{77}$$

since $k\omega L = 2\pi k$. Thus, by Equations (73), (74) and (77), $\alpha_0 = \frac{1}{L}\int_a^{a+L} F_1(x)\, dx \geq \frac{1}{L}\int_a^{a+L} \mathbf{1}_{\{x:\ |x|<0.49\}}(x)\, dx$, and taking $a = -0.49$ (allowed since $L \geq 0.98$) gives $\alpha_0 \geq \frac{0.98}{L} = \frac{0.98\,\omega}{2\pi}$, that is, $\alpha_0 \geq 0.1316$ for $\omega = 27/32$ ($L = 7.4467$). A longer period allows a smaller α_0. The final bound in Equation (76) comes out as α_0 plus a few paid terms.

By $\cos((k+1)\theta) = 2\cos(\theta)\cos(k\theta) - \cos((k-1)\theta)$, $F_1 = \alpha_0 + \sum_{k\in[M]} \alpha_k \cdot \cos(k\omega x)$ becomes a polynomial P of degree M in $u = \cos(\omega x)$. The pair $(M, \omega) = (12, 27/32)$ was found by trial: $M\omega$ must stay below $\pi/0.3 = 10.47$, while a smaller ω gives a longer period L, hence a smaller lower bound on α_0.

The coefficients α_k minimise the bound in Equation (76) subject to Equations (73) and (74). This is a linear program once variables $y_k \geq \max(\alpha_k, 0)$ are introduced, and we solve it on a grid, asking for some slack, $F_1 \geq 1.002$ at the grid points of the window $(-0.49, 0.49)$, because the coefficients returned by the solver are rounded afterwards (see below). We call $(\alpha_0, \ldots, \alpha_{12})$ the *certificate* and the bound it gives in Eq. (76) its *value*, all presented in Table 3.

F_1 is an even function and of period L, so the interval $[0, L/2]$ completely describes it, where $L/2 = 3.72$: F_1 stays just above 1 on $[0, 0.49)$, drops steeply after 0.49, rises above 1 twice more, and vanishes only at the double roots $x \approx 2.726$ and 3.408:

x	0	0.40	0.49	0.50	0.70	0.90	1.60	2.38	2.73	3.07	3.41	3.72
$F_1(x)$	1.005	1.061	1.007	0.994	0.479	0.020	0.126	1.154	0.001	1.254	0.000	0.992

It remains to check Equations (73) and (74) for the α_k of Table 3, in terms of P and $u = \cos(\omega x) \in [-1, 1]$. If $|x| < 0.49$, then $|\omega x| < 0.41344 < \pi$, so $u > \cos(0.41344) = 0.91574\ldots > 0.915$. Equations (73) and (74) become

$$P(u) \geq 0, \text{ for } -1 \leq u \leq 1, \tag{78}$$

$$P(u) \geq 1, \text{ for } 0.915 \leq u \leq 1. \tag{79}$$

To check Equation (78), the strategy is to exhibit constants t_j and $s_j \geq 0$, $j \in [6]$, such that

$$P(u) = 415 \prod_{j\in[6]} ((u - t_j)^2 + s_j). \tag{80}$$

k	α_k	$k\omega$	$\Gamma(k\omega)$	contribution
0	+0.484494	—	1	0.48449
1	+0.000740	0.844	0.83696	0.00062
2	+0.296334	1.688	0.49070	0.14541
3	+0.319967	2.531	0.20153	0.06448
4	+0.048476	3.375	0.05798	0.00281
5	−0.001606	4.219	0.01168	0
6	−0.004468	5.062	0.00165	0
7	+0.007906	5.906	0.00016	0.00000
8	−0.175007	6.750	0.00001	0
9	−0.043648	7.594	0.00000	0
10	+0.146236	8.438	0.00000	0.00000
11	−0.277042	9.281	0.00000	0
12	+0.202637	10.125	0.00000	0.00000
total				**0.69783**

Table 3. The coefficients α_k and their contribution (last column) to Equation (76), with $\Gamma(k\omega) = \exp(-(k\omega)^2/4)$; the five negative ones contribute nothing. The α_k are exact rationals, shown to six decimals, obtained by expanding the rounded product Equation (80); in particular $\alpha_{12} = 415/2048$. The total is $0.697821\ldots$, rounded up.

Since $s_j \geq 0$, every factor is non-negative, so $P(u) \geq 0$ for every real u. The constant 415 is the leading coefficient of P, forced by $\cos(12\omega x) = 2^{11}\cos^{12}(\omega x) - \ldots$, so that $415 = 2^{11}\alpha_{12} = 2048 \cdot \frac{415}{2048}$, while the six factors are monic. The t_j, s_j are as follows:

j	1	2	3	4	5	6
t_j	−247/256	−341/512	−27/256	179/512	357/512	33/32
s_j	0	0	1/8192	1/4096	1/8192	31/8192

More precisely, P and the α_k are obtained in four steps.

1. *Linear program.* The linear program above returns decimal coefficients, and hence, via $F_1(x) = P(\cos(\omega x))$, a polynomial of degree 12 with decimal coefficients.
2. *Factoring.* This polynomial is strictly positive on $[-1, 1]$, with minimum about 10^{-9}, and has no real roots. Its twelve complex roots therefore come in six conjugate pairs $t_j \pm i\sqrt{s_j}$ with $s_j > 0$, and grouping each pair gives a factorisation $c\prod_{j\in[6]}((u - t_j)^2 + s_j)$, where c is the leading coefficient.
3. *Rounding.* Each t_j and s_j is rounded to a fraction with denominator dividing 8192, and c to 415, which gives the values in the table above. Two of the s_j round to 0.
4. *Reading off the α_k.* The *rounded* product Equation (80) is taken as the definition of P, and $F_1(x) \coloneqq P(\cos(\omega x))$. Expanding Equation (80) and writing each power $\cos^k(\omega x)$ back in terms of $\cos(\omega x)$,...$\cos(k\omega x)$ gives the exact α_k of Table 3, which differ slightly from those of the linear program.

Steps 1–3 only explain where the numbers come from, and the proof does not depend on them: from now on P and F_1 are the explicit functions of step 4. Rounding preserves Equation (78), since the factored form stays non-negative, but Equation (79) must be checked for the rounded P, which we do below; the slack 1.002 is what keeps it true after rounding. As $s_1 = s_2 = 0$, P has double roots at $u = -247/256$ and $u = -341/512$, both in $[-1, 1]$, so it is just non-negative there; these are the two zeros of F_1 seen above.

To check Equation (79), substitute $u = 0.915 + 0.085\,v$, which maps $v \in [0, 1]$ onto $u \in [0.915, 1]$, and let $Q(v) \coloneqq P(0.915 + 0.085\,v) - 1 = \sum_{j=0}^{12} q_j v^j$, a polynomial of degree 12 in v. We write Q in terms of the thirteen polynomials

$$(1-v)^{12}, \quad v(1-v)^{11}, \quad v^2(1-v)^{10}, \quad \ldots, \quad v^{11}(1-v), \quad v^{12},$$

that is, $v^i(1-v)^{12-i}$ for $i = 0, \ldots, 12$, all of which are non-negative for $v \in [0,1]$. Multiplying each monomial v^j by $1 = (v + (1-v))^{12-j}$ and expanding this gives

$$Q(v) = \sum_{j=0}^{12} q_j v^j \big(v + (1-v)\big)^{12-j} = \sum_{i=0}^{12} \lambda_i \, v^i (1-v)^{12-i}, \qquad \lambda_i \coloneqq \sum_{j=0}^{i} \binom{12-j}{i-j} q_j. \tag{81}$$

Some of the q_j are negative (e.g. $q_2 \approx -0.377$), but all thirteen λ_i are positive; rounded down to four decimals they are

i	0	1	2	3	4	5	6	7	8	9	10	11	12
λ_i	0.0041	0.3528	3.2277	13.6934	34.2312	55.1628	59.2372	42.2844	19.2856	5.0755	0.5896	0.0115	0.0050

(note $\lambda_0 = Q(0) = P(0.915) - 1$ and $\lambda_{12} = Q(1) = P(1) - 1 = F_1(0) - 1$). For $v \in [0,1]$ every term on the right-hand side of Equation (81) is then non-negative, hence $Q \geq 0$ on $[0,1]$, i.e. $P \geq 1$ on $[0.915, 1]$. The smallest coefficients are λ_0 and λ_{12}, both below 0.006; such small margins are the reason for asking the linear program for the slack 1.002.

Equations (80) and (81) prove Equations (78) and (79), hence Equations (73) and (74); with Equation (76) and Table 3 this gives $p(\mathbf{w}) = \Pr[|X| < 0.49] \leq 0.69783 < 1/\sqrt{2}$ for every $\mathbf{w}$ with $\|\mathbf{w}\|_\infty \leq 0.3$.

Case 2. Assume $\|\mathbf{w}\|_\infty > 0.3$, that is, $\mathbf{w}_1 > 0.3$. Fix a small number $m \leq 4$ of *head* entries of $\mathbf{w}$. By Lemma 9, $p(\mathbf{w}) = E[g(T)]$, with $T = \sum_{j>m} r_j \mathbf{w}_j$ and g as in Equation (72). We bound g from above by a function $F_2 \coloneqq \mu_0 + \sum_{k\in[K]} \mu_k B_k$, whose expectation can be bounded explicitly, where all μ_k's are non-negative and the functions B_k are the bump functions from Lemma 8. More precisely, $p(\mathbf{w}) \leq E[F_2(T)] = \mu_0 + \sum_{k\in[K]} \mu_k E[B_k(T)]$, which we can further bound using Lemma 8 (2). As in *Case 1*, the coefficients μ_k are found by linear programming.

In order to find F_2, note that g from Equation (72) is a sum of indicator functions, which are nonzero only over the interval $(-0.49 - H_{\mathbf{a}}, 0.49 - H_{\mathbf{a}})$, for a pattern $\mathbf{a} \in \{-1, 0, 1\}^m$. If we know where the m head entries $\mathbf{w}_i$ lie, we get lower and upper bounds on $H_{\mathbf{a}}$, and widening each interval accordingly gives an upper bound on g. The bounds on the head entries also bound the tail T, and hence the prices of the bumps making F_2. It is therefore natural to confine the head entries to a product of intervals, $(\mathbf{w}_1, \ldots, \mathbf{w}_m) \in [\ell_1, u_1] \times \ldots \times [\ell_m, u_m]$, called a *box*. The vectors $\mathbf{w} \in \mathbb{R}^n$ with $\|\mathbf{w}\|_\infty > 0.3$ from this case are thus split into a finite list of boxes narrow enough to keep the bound on $E[F_2(T)]$ below $0.70 < 1/\sqrt{2}$. Note that for any box considered, $\sum_{i\in[m]} \ell_i^2$ has to be below 1, as for unit vectors $\mathbf{w}$, $\sum_{i\in[m]} \ell_i^2 \leq \sum_{i\in[m]} \mathbf{w}_i^2 \leq 1$. We formalise this idea in the following lemma.

Lemma 10. *Fix $m \leq n$ and intervals $[\ell_1, u_1], \ldots, [\ell_m, u_m]$ together with a bound u_{m+1} on the next entry, with $\sum_{i\in[m]} \ell_i^2 \leq 1$. Set $E_{\min} \coloneqq \max\left(0,\ 1 - \sum_{i\in[m]} u_i^2\right), \theta \coloneqq \min\left(u_{m+1},\ \sqrt{1 - \sum_{i\in[m]} \ell_i^2}\right)$. Suppose we are given a constant $\mu_0 \geq 0$, spacings $L_1, \ldots, L_K > 0$, centers $c_1, \ldots, c_K \in \mathbb{R}$ and coefficients $\mu_1, \ldots, \mu_K \geq 0$, and write $B_k(z) \coloneqq 1 + \cos\left(2\pi(z - c_k)/L_k\right)$. For each pattern $\mathbf{a} \in \{-1, 0, 1\}^m$, let $H_{\mathbf{a}}^{\min}$ and $H_{\mathbf{a}}^{\max}$ be the smallest and the largest value of $\sum_{i\in[m]} a_i \mathbf{w}_i$ as each $\mathbf{w}_i$ ranges over $[\ell_i, u_i]$, and let $I_{\mathbf{a}} \coloneqq (-0.49 - H_{\mathbf{a}}^{\max},\ 0.49 - H_{\mathbf{a}}^{\min})$. Let $x_0 < x_1 < \ldots < x_N$ be points with $\bigcup_{\mathbf{a}} I_{\mathbf{a}} \subseteq [x_0, x_N]$, and suppose that on any interval $[x_{i-1}, x_i]$ (called cell), $i \in [N]$,*

$$\sum \{p_{\mathbf{a}} \ :\ I_{\mathbf{a}} \cap [x_{i-1}, x_i] \neq \emptyset\} \leq \mu_0 + \sum_{k\in[K]} \mu_k \beta_{k,i}, \tag{82}$$

where $\beta_{k,i}$ is the lower bound of Lemma 8 (3) for the bump B_k on the cell $[x_{i-1}, x_i]$, or 0 if the cell contains a zero of B_k. Then every unit vector $\mathbf{w} \in \mathbb{R}^n$ with $\mathbf{w}_1 \geq \ldots \geq \mathbf{w}_n \geq 0$, whose m largest entries satisfy $\ell_i \leq \mathbf{w}_i \leq u_i$ and whose next entry satisfies $\mathbf{w}_{m+1} \leq u_{m+1}$, has tail entries bounded by θ and satisfies

$$p(\mathbf{w}) \ \leq\ V \coloneqq \mu_0 + \sum_{k\in[K]} \mu_k \cdot (1 + \max(0,\ \cos(2\pi c_k/L_k))\, \gamma_k), \tag{83}$$

with $\gamma_k \coloneqq \exp\left(-\pi^2 E_{\min}/L_k^2\right)$ if $L_k \geq 2\theta$ and $\gamma_k \coloneqq 1$ otherwise.

Proof. Write $F_2 \coloneqq \mu_0 + \sum_{k\in[K]} \mu_k B_k$ and let $\mathbf{w}$ be such a vector. To get to Equation (83), we first upper bound the function g. For each pattern $\mathbf{a} \in \{-1,0,1\}^m$, $H_{\mathbf{a}} = \sum_{i\in[m]} a_i \mathbf{w}_i \in [H_{\mathbf{a}}^{\min}, H_{\mathbf{a}}^{\max}]$, so then $(-0.49 - H_{\mathbf{a}}, 0.49 - H_{\mathbf{a}}) \subseteq I_{\mathbf{a}}$. Hence $g(y) \le h(y) \coloneqq \sum_{\mathbf{a}} p_{\mathbf{a}} \mathbf{1}_{I_{\mathbf{a}}}(y)$ for every $y \in \mathbb{R}$.

Now we upper bound h by F_2. For this, let $y \in \mathbb{R}$. If $y \notin [x_0, x_N]$ then no $I_{\mathbf{a}}$ contains it, so $h(y) = 0$, while $F_2(y) \ge 0$ by Lemma 8 (1) and $\mu_i \ge 0$, for $i \ge 0$. Otherwise y lies in some cell $[x_{i-1}, x_i]$. For any $I_{\mathbf{a}}$ containing y, we have $I_{\mathbf{a}} \cap [x_{i-1}, x_i] \ne \emptyset$, so $h(y)$ is at most the left-hand side of Equation (82); and by Lemma 8 (3) each bump satisfies $B_k(y) \ge \beta_{k,i}$, so $F_2(y)$ is at least the right-hand side of Equation (82). Combining, $g(y) \le F_2(y)$ for every y.

Therefore, by Lemma 9 and linearity of expectation, $p(\mathbf{w}) = E[g(T)] \le E[F_2(T)] = \mu_0 + \sum_{k\in[K]} \mu_k E[B_k(T)]$. By Lemma 8 (2), $E[B_k(T)]$ is at most $1 + \max(0, \cos(2\pi c_k/L_k))\gamma'_k$, where $\gamma'_k \coloneqq \exp(-\pi^2 \sum_{j>m} \mathbf{w}_j^2 / L_k^2)$ if $L_k \ge 2\max_{j>m} |\mathbf{w}_j|$ or 1 otherwise, for $k \in [K]$. Moreover, as $\ell_i \le \mathbf{w}_i \le u_i$ for every $i \in [m]$, $E_{\min} \le \sum_{j>m} \mathbf{w}_j^2 = 1 - \sum_{i\in[m]} \mathbf{w}_i^2 \le 1 - \sum_{i\in[m]} \ell_i^2$, which also shows that for every $j > m$, $0 \le \mathbf{w}_j \le \sqrt{1 - \sum_{i\in[m]} \ell_i^2}$, and thus θ upper bounds $\mathbf{w}_j$, as $\mathbf{w}_j \le \mathbf{w}_{m+1} \le u_{m+1}$. So if $L_k \ge 2\theta$, we can upper bound $E[B_k(T)]$ using γ'_k and thus γ_k, as $\exp(-x)$ is decreasing; otherwise $\gamma_k = 1$.

Equation (82) and V in Equation (83) are linear in $\mu_0, \ldots, \mu_K$, so for a fixed box, with its cells and with the spacings and centers fixed in advance, minimising V is a linear program. By a *certificate* we mean the list consisting of μ_0 and the spacings, centers and coefficients μ_k of the bumps, and its *value* is V. The box determines the cells: their endpoints are the endpoints of the windows $I_{\mathbf{a}}$, refined until no cell is wider than $1/16$. The spacings and centers, and with them the number K of bumps, were chosen for each box by a heuristic computer search; in the resulting certificates, the spacings and centers are, with few exceptions, multiples of $1/16$, the spacings lie in $[0.125,\ 15.5625]$, almost all of them below 5, the centers lie in $[-2.5,\ 0.375]$, and $0 \le K \le 11$. Since Lemma 10 holds for any spacings and centers, how they were chosen plays no role in the proof. To make these checks rigorous, and not subject to floating-point error, every certificate is evaluated with the functions $1 + \cos(t)$, $\cos(t)$ and e^{-x} replaced by rational bounds, as follows.

Indeed, $1 + \cos(t) \ge 2\max(0, 1 - t^2/8)^2$ and $\cos(t) \le \min(1,\ 1 - t^2/2 + t^4/24)$ for every t; $e^{-x} \le 1/\sum_{i<7} x^i/i!$ for $x \ge 0$; and, to bound $\cos(2\pi c/L)$ from above, c/L is reduced modulo 1 to some s with $|s| \le \frac{1}{2}$, and $\cos(2\pi c/L) = 2\cos^2(\pi s) - 1$ is bounded using the upper bound on $\cos(\pi s)$ above. The lower bound $\pi^2 \ge 9.8695$ is used in the prices, i.e. the bounds on $E[B_k(T)]$ in Equation (83), where it can only increase $\exp(-\pi^2 E_{\min}/L_k^2)$; the upper bound $\pi^2 \le 9.8697$ is used in the lower bounds of B_k on $[x_{i-1}, x_i]$, $\beta_{k,i}$, from Equation (82). These $\beta_{k,i}$ are $\beta_{k,i} = 1 + \cos(2\pi d/L_k) \ge 2\max(0, 1 - \pi^2 d^2/(2L_k^2))^2$ according to Lemma 8 (3), where d is the number defined there for the cell $[x_{i-1}, x_i]$, in the case this cell has no zero of B_k. Since replacing $\beta_{k,i}$ by a smaller number only makes Equation (82) harder to satisfy, checking Equation (82) with these smaller values suffices. Note that if $1 - \pi^2 d^2/(2L_k^2) > 0$, then $d < \sqrt{2} L_k/\pi < L_k/2$, so the cell lies within $L_k/2$ of a local maximum point of B_k and thus the condition on the cell containing no zero is already assured. Apart from θ, all numbers entering Equations (82) and (83) are then rationals, so each cell inequality Equation (82) is an exact comparison of rational numbers, and the computed V is an upper bound for the exact value of Equation (83). The quantity θ, which may be irrational, enters only through the test $L_k \ge 2\theta$; as θ is a minimum, this test holds if and only if $L_k \ge 2u_{m+1}$ or $L_k^2 \ge 4\big(1 - \sum_{i\in[m]} \ell_i^2\big)$, again a comparison of rationals.

We need a list of boxes $[\ell_1, u_1] \times \ldots \times [\ell_m, u_m]$ and u_{m+1} covering all unit vectors $\mathbf{w}$ with $\mathbf{w}_1 \ge \ldots \ge \mathbf{w}_n \ge 0$ and $\mathbf{w}_1 > 0.3$, to apply Lemma 10 on each. We build it by repeated cuts, starting from the box $[0.3, 1]$ (i.e. $m = 1$, $\mathbf{w}_1 \in [0.3, 1]$ and $\mathbf{w}_2 \le 1$). If the linear program finds a certificate for the box with value $V < 0.70$, that box is done; otherwise, the box is cut in two boxes by a single comparison, of one of two kinds:

- *cutting an interval*: one entry $\mathbf{w}_i$ of the box is compared with a cut point, which is 0.49 or 0.3 when the interval (ℓ_i, u_i) contains one of them and its midpoint otherwise; for instance, cutting the box $[0.3, 1]$ for $m = 1$ at 0.49 gives the two boxes, both for $m = 1$: $[0.3, 0.49]$ and $[0.49, 1]$;
- *conditioning on one more entry*: the next entry $\mathbf{w}_{m+1}$, which the box bounds only by $\mathbf{w}_{m+1} \le u_{m+1}$, is compared with a threshold τ; the box with $\mathbf{w}_{m+1} \le \tau$ gets the smaller bound $u_{m+1} = \tau$ on the next

entry, and in the box with $\tau \le \mathbf{w}_{m+1} \le u_{m+1}$ the entry $\mathbf{w}_{m+1}$ is added to the conditioned entries, giving a box with $m+1$ entries, where $\mathbf{w}_{m+1} \in [\tau, u_{m+1}]$ and $\mathbf{w}_{m+2} \le u_{m+2} := u_{m+1}$. The thresholds τ were chosen heuristically; most of them are 0.3 or 0.49.

Every vector $\mathbf{w}$ of a box lies on one side of the comparison, so the two new boxes together cover the box they come from. The starting box covers all vectors $\mathbf{w}$ of *Case 2*, hence, by induction, the final list of boxes covers all these vectors. The process stops after 418 cuts, which leave 419 boxes: 9 of them are discarded, as they satisfy $\sum_{i\in[m]} \ell_i^2 > 1$ and contradict $\|\mathbf{w}\| = 1$, and for each of the remaining 410 boxes, a certificate was found by the linear program determined by the cell inequalities Equation (82); its coefficients were then replaced by nearby rationals, for which these inequalities were checked and the value V computed in exact rational arithmetic with the replacements above. Depending on m, these 410 boxes are 95 with $m = 1$, 136 with $m = 2$, 140 with $m = 3$ and 39 with $m = 4$. 33 of these boxes have no bumps ($K = 0$); for them Equation (82) constrains only μ_0, and $V = \mu_0$. The median value of V (Equation (83)) obtained when analysing all the 410 boxes is 0.65373 and the largest value of V is 0.69985, reached for the case when $(\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3)$ lies in the box $[0.4781, 0.4900] \times [0.4670, 0.4781] \times [0.4670, 0.4781]$, with $\mathbf{w}_4 \le 0.4781$. Its certificate has $\mu_0 = 0$ and $K = 4$ bumps; writing $B(L, c)(z) := 1 + \cos(2\pi(z - c)/L)$ for the bump with spacing L and center c, and rounding the coefficients to five decimals, it is

$$F_2 = 0.00354\, B(1, -1) + 0.22926\, B(3.125, 0) + 0.14666\, B(5, 0) + 0.01122\, B(5.125, 0).$$

So

$$\|\mathbf{w}\|_\infty > 0.3 \implies p(\mathbf{w}) = \Pr[|X| < 0.49] \le 0.69985, \tag{84}$$

which settles *Case 2*. Putting Equation (84) together with the bound of *Case 1*, for every unit vector $\mathbf{w} \in \mathbb{R}^n$,

$$\Pr[|\langle \mathbf{r}, \mathbf{w}\rangle| < 0.49] \le \max(0.69783,\ 0.69985) = 0.69985 \le 0.70 < \frac{1}{\sqrt{2}},$$

which proves the claim.