Succinct Vector, Polynomial, and Functional Commitments from Lattices

Hoeteck Wee
NTT Research and ENS, Paris
wee@di.ens.fr

David J. Wu*
UT Austin
dwu4@cs.utexas.edu

Abstract

Vector commitment schemes allow a user to commit to a vector of values $\mathbf{x} \in \{0,1\}^\ell$ and later, open up the commitment to a specific set of positions. Both the size of the commitment and the size of the opening should be succinct (i.e., polylogarithmic in the length ℓ of the vector). Vector commitments and their generalizations to polynomial commitments and functional commitments are key building blocks for many cryptographic protocols.

We introduce a new framework for constructing non-interactive lattice-based vector commitments and their generalizations. A simple instantiation of our framework yields a new vector commitment scheme from the standard short integer solution (SIS) assumption that supports *private* openings and large messages. We then show how to use our framework to obtain the first succinct *functional* commitment scheme that supports openings with respect to arbitrary bounded-depth Boolean circuits. In this scheme, a user commits to a vector $\mathbf{x} \in \{0,1\}^\ell$, and later on, open the commitment to any function $f(\mathbf{x})$. Both the commitment *and* the opening are non-interactive and succinct: namely, they have size $\mathsf{poly}(\lambda, d, \log \ell)$, where λ is the security parameter and d is the *depth* of the Boolean circuit computing f. Previous constructions of functional commitments could only support constant-degree polynomials, or require a trusted *online* authority, or rely on non-falsifiable assumptions. The security of our functional commitment scheme is based on a new falsifiable family of "basis-augmented" SIS assumptions (BASIS) we introduce in this work.

We also show how to use our vector commitment framework to obtain (1) a polynomial commitment scheme where the user can commit to a polynomial $f \in \mathbb{Z}_q[x]$ and subsequently open the commitment to an evaluation $f(x) \in \mathbb{Z}_q$; and (2) an aggregatable vector (resp., functional) commitment where a user can take a set of openings to multiple indices (resp., function evaluations) and aggregate them into a *single* short opening. Both of these extensions rely on the same BASIS assumption we use to obtain our succinct functional commitment scheme.

1 Introduction

Vector commitment schemes [Mer87, CFM08, LY10, CF13] allow a user to commit to a vector of values $\mathbf{x} \in \{0,1\}^\ell$ and subsequently, open up the commitment to a specific set of positions. Both the commitment and the openings should be *succinct* (i.e., have size that scales *polylogarithmically* with the vector length ℓ) and *non-interactive*.[1] There has recently been tremendous interest and progress in the design and application of vector commitments, and even a "Vector Commitment Research Day" [Res22]. Starting from the classic vector commitment scheme of Merkle [Mer87] based on collision-resistant hash functions, we now have a broad range of algebraic constructions from pairing-based assumptions [LY10, KZG10, CF13, LRY16, LM19, TAB+20, GRWZ20] as well as assumptions over groups of unknown order (e.g., RSA groups or class groups) [CF13, LM19, CFG+20, AR20, TXN20]. We refer to [Nit21] for a survey of recent schemes. As a primitive, vector commitment schemes have found numerous applications to verifiable outsourced databases [BGV11, CF13], cryptographic accumulators [CF13], pseudonymous credentials [KZG10], and to blockchain protocols [RMCI17, CPSZ18, BBF19]. Moreover, the generalization of vector commitments to polynomial commitments [KZG10] has emerged as a key building block in many recent (random-oracle) constructions of succinct

*Part of this work was done while visiting NTT Research.

[1] We discuss interactive commitments (as well as constructions in the random oracle model) in Section 1.3.

non-interactive arguments of knowledge (SNARKs) [MBKM19, CHM+20, GWC19, BDFG21, BFS20, COS20] having various appealing properties (e.g., universal or transparent setup, recursive composability, and more).

In this work, we focus on two themes in the study of vector commitments where progress has been more limited: (1) post-quantum constructions based on lattices [PSTY13, LLNW16, PPS21, ACL+22, FSZ22]; and (2) functional commitments, a generalization of vector commitments that supports openings to various functions on the committed values [LRY16, LP20, PPS21, BNO21, ACL+22]. There are good technical reasons for the limited progress on these two fronts. First, many of the techniques developed for vector commitments crucially exploit algebraic structure in pairing and RSA/class groups that do not naturally extend to the lattice setting. Second, pairing and RSA/class groups only support limited homomorphic capabilities.

1.1 Our Results

In this work, we introduce a general framework for constructing lattice-based vector commitments that simultaneously encapsulates recent lattice-based vector commitment schemes [PPS21, ACL+22] and enables us to achieve stronger functionality and security properties. As we describe below, our framework readily generalizes to also yield *polynomial commitments*, *functional commitments*, and *aggregatable commitments* from (falsifiable) lattice-based assumptions.

A new family of SIS assumptions. The security of our schemes relies on a new "basis-augmented" family of short integer solution (SIS) assumptions we introduce in this work. We refer to our basis-augmented SIS assumption as the BASIS assumption (Assumption 3.2). The basic version of our assumption (denoted $\mathsf{BASIS}_{\mathsf{rand}}$) suffices for constructing standard vector commitments and is implied by the *standard* SIS assumption. The structured version of the assumption (denoted $\mathsf{BASIS}_{\mathsf{struct}}$) we need for our extensions has a similar flavor as the k-SIS-like assumptions introduced in [ACL+22] for constructing lattice-based succinct arguments (c.f., Section 6). While the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption is not a standard lattice-based assumption, it is a *falsifiable* assumption [Nao03]. We view our assumption as a "q-type" lattice assumption and at a conceptual level, it shares a similar flavor as the q-type assumptions used in the pairing-based world for constructing vector commitments [CF13] and polynomial commitments [KZG10].

Vector commitments with private opening. An immediate consequence of our framework is a vector commitment scheme that supports *private* openings. In this setting, a user can commit to a vector $\mathbf{x} \in \{0,1\}^\ell$ with a short commitment σ and then open σ to an index-value pair (i, x_i) with a short opening π_i. We say the vector commitment scheme supports private openings if the commitment σ and any collection of openings $\{(i, x_i, \pi_i)\}_{i \in S}$ reveal no additional information about x_j for any $j \notin S$. Notably and in contrast to previous lattice-based vector commitment schemes [PPS21, ACL+22], our scheme also does *not* impose any restrictions on the magnitude of the entries of $\mathbf{x}$ (the vectors can be arbitrary elements of $\mathbb{Z}_q^\ell$ and the commitment as well as the opening are vectors over $\mathbb{Z}_q$). Previous lattice-based schemes [PPS21, ACL+22] require that the components of $\mathbf{x}$ be small and this property was essential for *both* correctness and security.

Our vector commitment scheme has the same efficiency properties as the earlier scheme of Peikert et al. [PPS21] which did not support private openings and was limited to a small message space. Our scheme provides the same functionality (e.g., support for "stateless updates") and like the scheme of [PPS21], security can be based on the *standard* SIS assumption. Thus, relative to [PPS21], our framework achieves private openings and supports a large message space with essentially no overhead.

We could alternatively obtain a lattice-based vector commitment by instantiating Merkle's classic construction [Mer87] with a lattice-based collision-resistant hash function (e.g., Ajtai's hash function from SIS [Ajt96, GGH96]). Our vector commitment scheme improves upon this generic approach in two main ways: (1) we support (bounded) stateless updates like [PPS21] (where a user can update a commitment to a vector $\mathbf{x}$ into a commitment to a vector $\mathbf{x}'$ given only knowledge of the difference $\mathbf{x}' - \mathbf{x}$ and not the entirety of $\mathbf{x}$ or $\mathbf{x}'$); and (2) we can support private openings directly. It is possible to extend Merkle hashing to support private openings via zero-knowledge proofs, but this would either need non-black-box use of cryptography or require interaction, random oracles, or correlation-intractable hash functions [CCH+19, PS19]. More broadly, as we illustrate below, our algebraic scheme serves as a stepping stone for realizing polynomial and functional commitment schemes (for which we crucially exploit *algebraic* structure).

Functional commitments. A functional commitment [GVW15, LRY16] is a generalization of a vector commitment with the property that given a commitment to an input $\mathbf{x} \in \{0,1\}^\ell$, one can then construct an opening π_f to $y = f(\mathbf{x})$, for some function f. The basic binding property of the commitment scheme says that the adversary cannot come up with openings π_f and π'_f that open σ to different values $y \neq y'$ with respect to the same function f. The efficiency requirements are that the size of the commitment and the opening should be sublinear in both the size of the function f and the length of the input $\mathbf{x}$. Previously, Peikert et al. [PPS21] showed how to construct functional commitments for bounded-depth Boolean circuits in an *online* model where a central *trusted* authority issues opening keys for functions f, with security based on the standard SIS assumption. Albrecht et al. [ACL+22] subsequently showed how to construct functional commitments for constant-degree polynomials from new variants of the SIS assumption in the standard setting without an online authority. Earlier pairing-based functional commitments could only support linear functions [LRY16] or sparse polynomials [LP20]. Functional commitments can also be obtained generically by combining a vanilla vector commitment (e.g., a Merkle tree [Mer87]) with a succinct non-interactive argument of knowledge (for NP). However, existing constructions of SNARKs (for NP) either rely on making non-falsifiable assumptions [GW11] or working in idealized models.

Our vector commitment framework directly yields a succinct functional commitment scheme for all bounded-depth Boolean circuits in the standard offline model without an authority and from falsifiable assumptions. The size of the commitment and the openings are $\mathsf{poly}(\lambda, d, \log \ell)$, where λ is a security parameter, d is the *depth* of the Boolean circuit computing $f\colon \{0,1\}^\ell \to \{0,1\}$, and ℓ is the length of the input. Security relies on the new non-standard, but falsifiable, $\mathsf{BASIS}_{\mathsf{struct}}$ assumption we introduce in this work (with a *sub-exponential* noise bound). Notably, this is the first succinct functional commitment scheme for general circuits from a falsifiable assumption, and answers an open question posed by Peikert et al. [PPS21]. Our construction can be viewed as a *succinct* analog of the homomorphic commitments and signatures introduced by [GSW13, GVW15].[2]

Polynomial commitments. In a polynomial commitment [KZG10], a user can commit to a polynomial $f \in \mathbb{Z}_q[x]$ over $\mathbb{Z}_q$ and later open to an evaluation $f(x)$ at any point $x \in \mathbb{Z}_q$. A polynomial commitment can be viewed as a succinct commitment to the vector of evaluations of f on all inputs $x \in \mathbb{Z}_q$. While a polynomial commitment can be built from a succinct functional commitment for Boolean circuits, this incurs a $\mathsf{poly}(\log q)$ overhead to encode the polynomial evaluation over $\mathbb{Z}_q$ as a Boolean circuit and also relies on the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with a *sub-exponential* noise bound. In this work, we show that a simple adaptation of our succinct functional commitments to the setting of linear functions directly gives a polynomial commitment over $\mathbb{Z}_q$. Notably, this construction can be based on our $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with only a *polynomial* noise bound. This is the first polynomial commitment scheme from lattices based on falsifiable assumptions.

An important feature of our framework that enables the direct construction of polynomial commitments is that it natively supports values over $\mathbb{Z}_q$. Previous lattice-based vector commitments [PPS21, ACL+22] required that the committed value x and the opened value $f(x)$ be "small," and moreover, that the modulus q scale with the norm of the output (i.e., $f(x)$) when computed over the *integers*. This is not suitable when constructing polynomial commitments directly, as the size of $f(x)$ computed over the integers scales with the degree of f. Correspondingly, if the modulus q scales linearly with the degree of f, then the resulting scheme is no longer succinct. The ability to directly work over the entirety of $\mathbb{Z}_q$ is an appealing property of our new framework.

Aggregatable commitments. A simple modification to our basic vector commitment scheme yields a scheme that supports *aggregation*. We say a vector commitment scheme is aggregatable [BBF19, CFG+20] if given a commitment σ along with a set of openings $\pi_1, \ldots, \pi_t$ to indices $i_1, \ldots, i_t \in [\ell]$ and values $x_{i_1}, \ldots, x_{i_t}$, there is an efficient aggregation algorithm that outputs a short aggregate opening π that validates the full set of values $\{(i_j, x_{i_j})\}_{j \in [t]}$. The requirement is that the size of π scale sublinearly, or better yet, *polylogarithmically* with t. Aggregatable commitments immediately imply subvector commitments [LM19] (i.e., a vector commitment scheme that supports batch openings to a set of indices $S \subseteq [\ell]$). Our framework yields an aggregatable commitment scheme for short messages from the same falsifiable $\mathsf{BASIS}_{\mathsf{struct}}$ assumption used to construct succinct functional commitments. This is the first aggregatable

[2] The homomorphic commitments from [GSW13, GVW15] are *non-succinct*; in particular, the size of the commitment scales *linearly* with the input length ℓ.

commitment scheme from lattice assumptions *without* relying on general-purpose succinct arguments [ACL+22] or batch arguments [CJJ21, DGKV22], and answers another open question posed by Peikert et al. [PPS21].

A limitation of our aggregatable commitment is that it only satisfies *same-set binding*, which guarantees that for every subset of indices $S \subseteq [\ell]$, the adversary can only open to a single set of values. However, there is still the possibility that an adversary could open the commitment to different sets S and T that are *inconsistent* (i.e., $x_i = 0$ with respect to S while $x_i = 1$ with respect to T).[3] Constructing aggregatable commitments that satisfy the stronger notion of *different-set* binding directly from falsifiable lattice-based assumptions is an interesting open problem.

The same techniques we use to construct aggregatable vector commitments also applies to our succinct functional commitment scheme, and we obtain an aggregatable functional commitment scheme from the same underlying hardness assumption. In this setting, a user can take openings $\pi_1, \ldots, \pi_t$ for function-value pairs $(f_1, y_1), \ldots, (f_t, y_t)$ and aggregate the openings into a single short opening π that validates all t function-value pairs and where the size of the aggregated opening scales polylogarithmically with t.

Summary. Similar to previous lattice-based vector commitments [PPS21, ACL+22], we rely on a *structured* reference string in all of our constructions. We refer to the structured reference string as a common reference string (CRS). To summarize, our new lattice-based vector commitment framework yields the following constructions:

- A vector commitment scheme with private openings based on the *standard* SIS assumption with polynomial noise bound (Corollary 3.17). For vectors of dimension ℓ, the size of the commitment is $O(\lambda(\log \lambda + \log \ell))$ and the size of an opening is $O(\lambda(\log^2 \lambda + \log^2 \ell))$.[4] The size of the CRS is $\ell^2 \cdot \mathsf{poly}(\lambda, \log \ell)$.
- A succinct functional commitment scheme supporting all bounded-depth Boolean circuits from the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with a sub-exponential noise bound (Corollary 4.8). A variant of this construction supports private openings under a weaker notion of target binding (Corollary 4.33). For both constructions, to support functions on ℓ-bit inputs and computable by Boolean circuits of depth d, the sizes of the commitment and openings are $\mathsf{poly}(\lambda, d, \log \ell)$. The size of the CRS is $\ell^2 \cdot \mathsf{poly}(\lambda, d, \log \ell)$.
- A polynomial commitment (for polynomials of a priori bounded degree) under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with a polynomial noise bound (Corollary 4.21). To support polynomials of degree up to d over $\mathbb{Z}_q$ (where $q = \mathsf{poly}(\lambda)$), the sizes of the commitment and openings are $\mathsf{poly}(\lambda, \log d)$. The size of the CRS is $d^2 \cdot \mathsf{poly}(\lambda, \log d)$.
- An aggregatable vector commitment scheme (over a small message space) based on the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with polynomial noise bound (Corollary 5.11). The sizes of the commitment, openings, and CRS match those above for our vanilla vector commitment.
- An aggregatable functional commitment scheme for all bounded-depth Boolean circuits from the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption used to obtain succinct functional commitments (Corollary 5.23). To support aggregating T openings for functions on ℓ-bit inputs and computable by Boolean circuits of depth d, the sizes of the commitment and opening are $\mathsf{poly}(\lambda, d, \log \ell, \log T)$. The size of the CRS is $(\ell^2 + T) \cdot \mathsf{poly}(\lambda, d, \log \ell, \log T)$. In the random oracle model, we can reduce the CRS size to $\ell^2 \cdot \mathsf{poly}(\lambda, d, \log \ell)$ and support an *arbitrary* polynomial number of aggregations.

1.2 Technical Overview

In this section, we provide a general overview of our new framework for constructing vector commitments from lattices as well as the family of basis-augmented SIS assumptions (BASIS) we use to prove hardness. In the following description, for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a target vector $\mathbf{t} \in \mathbb{Z}_q^n$, we write $\mathbf{A}^{-1}(\mathbf{t})$ to denote a random variable $\mathbf{x} \in \mathbb{Z}_q^m$ whose entries are distributed according to a discrete Gaussian conditioned on $\mathbf{Ax} = \mathbf{t}$. Sampling $\mathbf{x} \leftarrow \mathbf{A}^{-1}(\mathbf{t})$ can be done efficiently given a trapdoor for $\mathbf{A}$ (see Section 2.1). Here, we will use the Micciancio-Peikert gadget trapdoors [MP12]; namely, a matrix $\mathbf{R}$ is a gadget trapdoor for $\mathbf{A}$ if $\mathbf{R}$ is short and $\mathbf{AR} = \mathbf{G}$, where $\mathbf{G} = \mathbf{I}_n \otimes \mathbf{g}^\intercal$ is the gadget matrix and $\mathbf{g}^\intercal = [1, 2, \ldots, 2^{\lfloor \log q \rfloor}]$.

[3] Note though that if the commitment is honestly-generated, then same-set binding implies different-set binding; see Remark 5.13.

[4] We note that these bounds match the base construction of Peikert et al. [PPS21]. While [PPS21, Figure 1] reports that their scheme has $O(\log \ell)$-size openings (ignoring the security parameter λ), the construction itself [PPS21, Construction 1] has $O(\log^2 \ell)$-size openings.

A general framework for constructing vector commitments. We begin by describing a general framework for constructing lattice-based vector commitments that encapsulates the recent schemes from [PPS21, ACL+22]:

- The common reference string (CRS) specifies a collection of ℓ matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times m}$ and ℓ vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell \in \mathbb{Z}_q^n$ along with some auxiliary input $\mathsf{aux}_\ell := \{\mathbf{A}_i^{-1}(\mathbf{t}_j)\}_{i \neq j}$.
- The commitment to a vector $\mathbf{x} = (x_1, \ldots, x_\ell) \in \{0,1\}^\ell$ is a vector $\mathbf{c} \leftarrow \sum_{i \in [\ell]} x_i \mathbf{t}_i \in \mathbb{Z}_q^n$.
- An opening to index $i \in [\ell]$ and value $x_i \in \{0,1\}$ is a short vector $\mathbf{v}_i \in \mathbb{Z}_q^m$ such that

$$\mathbf{c} = \mathbf{A}_i \mathbf{v}_i + x_i \mathbf{t}_i. \tag{1.1}$$

  The honest opening is computed as $\mathbf{v}_i \leftarrow \sum_{j \neq i} x_j \mathbf{A}_i^{-1}(\mathbf{t}_j)$.

Correctness follows by inspection:

$$\mathbf{A}_i \mathbf{v}_i + x_i \mathbf{t}_i = \sum_{j \neq i} x_j \mathbf{A}_i \cdot \mathbf{A}_i^{-1}(\mathbf{t}_j) + x_i \mathbf{t}_i = \sum_{i \in [\ell]} x_i \mathbf{t}_i = \mathbf{c}.$$

For binding, we require that it is hard to find a short vector $\mathbf{z} \in \mathbb{Z}_q^m$ such that $\mathbf{A}_i \mathbf{z} = \mathbf{t}_i$ for any $i \in [\ell]$ given the components in the CRS. Next, we explain how the schemes PPS_1 from [PPS21][5] and MatrixACLMT from [ACL+22][6] fall into this framework.

- In PPS_1, the matrices $\mathbf{A}_i \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ and vectors $\mathbf{t}_i \xleftarrow{R} \mathbb{Z}_q^n$ are independent and uniformly random for all $i \in [\ell]$. Binding in turn is based on the standard SIS assumption.
- In MatrixACLMT, they sample uniformly random vectors $\mathbf{u}_i \xleftarrow{R} \mathbb{Z}_q^n$, a matrix $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, and invertible matrices $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n \times n}$ for each $i \in [\ell]$. Then, they set $\mathbf{A}_i \leftarrow \mathbf{W}_i \mathbf{A}$, $\mathbf{t}_i \leftarrow \mathbf{W}_i \mathbf{u}_i$. In this case, $\mathbf{A}_i^{-1}(\mathbf{t}_j) = \mathbf{A}^{-1}(\mathbf{W}_i^{-1} \mathbf{W}_j \mathbf{u}_j)$. Binding is based on a new assumption which stipulates that it is hard to find a short vector $\mathbf{z} \in \mathbb{Z}_q^m$ where $\mathbf{A}\mathbf{z} = \mathbf{u}_i$ for any $i \in [\ell]$ given the CRS. The authors of [ACL+22] then show how to leverage the extra structure arising from the correlated $\mathbf{A}_i$'s to obtain a functional commitment scheme for constant-degree polynomials as well as a preprocessing succinct non-interactive argument (SNARG) for NP.

Before describing our approach, we describe two limitations of these instantiations:

- **Small message space:** In both the PPS_1 and the MatrixACLMT instantiations of this framework, both correctness *and* security require that the entries of the vector $\mathbf{x} = [x_1, \ldots, x_\ell]$ be small. This is because the verification relation is checking that the opening $\mathbf{v}_i = \sum_{j \neq i} x_j \mathbf{A}_i^{-1}(\mathbf{t}_j)$ is small. Thus, correctness requires that each x_j be small. Moreover, in the proof of binding, the reduction algorithm takes a commitment $\mathbf{c}$ along with two openings $(x_i, \mathbf{v}_i)$, $(x_i', \mathbf{v}_i')$ to derive a solution to SIS or a related problem. The existing reductions require that the difference $(x_i - x_i')$ be small (in order to derive a *short* solution).
- **Uniform target vectors.** In both the PPS_1 and MatrixACLMT constructions, the target vectors $\mathbf{t}_i$ are essentially random vectors. This is important for ensuring that $\mathbf{A}_i^{-1}(\mathbf{t}_j)$ does not leak a trapdoor for $\mathbf{A}_i$, which would immediately break binding. Using structured target vectors could enable additional functionality. For instance, in Remark 6.1, we show that instantiating MatrixACLMT with structured targets can be used to support functional openings. Unfortunately, this instantiation *also* leaks a trapdoor for $\mathbf{A}_i$, and is insecure.

The approach we take in this work avoids these limitations and allows us to construct vector commitments with a large message space as well as support new capabilities like polynomial and functional openings.

[5] By PPS_1, we refer to the the base scheme from [PPS21, Construction 1]; they also present a second tree-based scheme that uses PPS_1 as a building block.

[6] The authors of [ACL+22] describe their scheme in the ring setting. We write MatrixACLMT to denote one possible translation from the ring setting to the integer setting. Note that there are other ways to translate their scheme to the integer setting such as sampling $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{m \times m}$ and then setting $\mathbf{A}_i \leftarrow \mathbf{A}\mathbf{W}_i$.

Our approach. We consider the same verification relation $\mathbf{c} = \mathbf{A}_i\mathbf{v}_i + x_i\mathbf{t}_i$ from Eq. (1.1), but take a completely different approach for computing the commitment $\mathbf{c}$ and the openings $\mathbf{v}_i$: we sample a *random* tuple $(\mathbf{v}_1, \ldots, \mathbf{v}_\ell, \mathbf{c})$ that simultaneously satisfies the verification relation for all $i \in [\ell]$. As in the previous verification relation, we require that the openings $\mathbf{v}_1, \ldots, \mathbf{v}_\ell$ are short. The commitment $\mathbf{c}$ can have large entries. In our particular setting, we write $\mathbf{c}$ as $\mathbf{c} = \mathbf{G}\hat{\mathbf{c}}$ where $\hat{\mathbf{c}} \in \mathbb{Z}_q^m$ is a short vector. Using the gadget matrix $\mathbf{G}$ will be important in the security analysis. Then, Eq. (1.1) corresponds to the relation $\mathbf{G}\hat{\mathbf{c}} = \mathbf{A}_i\mathbf{v}_i + x_i\mathbf{t}_i$, or equivalently, $\mathbf{A}_i\mathbf{v}_i - \mathbf{G}\hat{\mathbf{c}} = -x_i\mathbf{t}_i$ for all $i \in [\ell]$. We can express these ℓ relations as a linear system:

$$\underbrace{\left[\begin{array}{ccc|c} \mathbf{A}_1 & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & -\mathbf{G} \end{array}\right]}_{\mathbf{B}_\ell} \cdot \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} = \begin{bmatrix} -x_1\mathbf{t}_1 \\ \vdots \\ -x_\ell\mathbf{t}_\ell \end{bmatrix}. \tag{1.2}$$

Our goal now is to sample a random *short* tuple $(\mathbf{v}_1, \ldots, \mathbf{v}_\ell, \hat{\mathbf{c}})$ that satisfies Eq. (1.2). This can be done by giving out a random trapdoor for the matrix $\mathbf{B}_\ell$:

$$\mathbf{B}_\ell := \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & -\mathbf{G} \end{array}\right]. \tag{1.3}$$

Using $\mathbf{B}_\ell$, we can sample a random short preimage $(\mathbf{v}_1, \ldots, \mathbf{v}_\ell, \hat{\mathbf{c}})$ satisfying Eq. (1.2). This yields the commitment $\mathbf{c} = \mathbf{G}\hat{\mathbf{c}}$ and the openings $\mathbf{v}_1, \ldots, \mathbf{v}_\ell$. In our construction, we set the target vector $\mathbf{t}_i$ to the first basis vector $\mathbf{e}_1 = [1, 0, \ldots, 0]^\intercal$ for all $i \in [\ell]$. We now make the following observations:

- **Binding:** To argue that the scheme is binding, we require that it is hard to find a short vector $\mathbf{z}$ where $\underline{\mathbf{A}_i}\mathbf{z} = \mathbf{0}$ for any $i \in [\ell]$ even given the (related) matrix $\mathbf{B}_\ell$ and a trapdoor for $\mathbf{B}_\ell$. Here, $\underline{\mathbf{A}_i}$ denotes $\mathbf{A}_i$ with the first row removed. We refer to assumptions of this type as "basis-augmented SIS" (BASIS) assumptions (Assumption 3.2). As we sketch below (and show formally in Theorem 3.4), when $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \xleftarrow{R} \mathbb{Z}_q^{n\times m}$ are *random*, this instantiation of the BASIS assumption holds under the *standard* SIS assumption. We refer to this instance of the BASIS assumption with random matrices as $\mathsf{BASIS}_{\mathsf{rand}}$. Now, to argue binding, we observe that an adversary that breaks binding is able to come up with an index $i \in [\ell]$, short vectors $\mathbf{v}, \mathbf{v}' \in \mathbb{Z}_q^m$ and values $x, x' \in \mathbb{Z}_q$ such that $\mathbf{c} = \mathbf{A}_i\mathbf{v} + x\mathbf{e}_1 = \mathbf{A}_i\mathbf{v}' + x'\mathbf{e}_1$. This means that

  $$\mathbf{A}_i(\mathbf{v} - \mathbf{v}') = (x' - x)\mathbf{e}_1.$$

  As long as $x' - x \neq 0$, $\mathbf{v} - \mathbf{v}' \neq \mathbf{0}$, and so $\mathbf{v} - \mathbf{v}'$ is a SIS solution to $\underline{\mathbf{A}_i}$. Observe that this analysis does not impose *any* restriction on the magnitude of $x' - x$. This means our construction naturally supports committing to arbitrary vectors over $\mathbb{Z}_q$ as opposed to vectors with small entries.[7] We give the formal reduction to the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption in Theorem 3.11.

- **Private openings.** A vector commitment scheme supports private openings if the commitment $\mathbf{c}$ and any collections of openings $\{(i, x_i, \mathbf{v}_i)\}_{i\in S}$ completely hide the values x_j for $j \notin S$. Since we sample the commitment $\mathbf{c}$ and the openings $\mathbf{v}_i$ *jointly* in our approach, it is straightforward to argue (by appealing to properties of discrete Gaussians; see Corollary 2.11) that the commitment $\mathbf{c}$ is statistically close to uniform over $\mathbb{Z}_q^n$ and each opening $\mathbf{v}_i$ is statistically close to the distribution $\mathbf{A}_i^{-1}(\mathbf{c} - x_i\mathbf{t}_i)$. Thus our scheme provides *statistically* private openings out of the box.

Taken together, this yields a vector commitment from standard SIS that supports statistically private openings and commitments to arbitrary vectors over $\mathbb{Z}_q^\ell$. We give the full description and analysis in Section 3.

[7] As discussed earlier, previous vector commitments [PPS21, ACL+22] based on SIS or its generalizations needed to assume small inputs for *both* correctness and security.

Reducing $\mathsf{BASIS}_{\mathsf{rand}}$ to standard SIS. As described above, the binding property of our vector commitment relies on the BASIS assumption, which says that SIS with respect to $\underline{\mathbf{A}_i}$ (i.e., $\mathbf{A}_i$ with the first row removed) is hard even given the related matrix $\mathbf{B}_\ell$ from Eq. (1.3) and a trapdoor for $\overline{\mathbf{B}_\ell}$. As we show in Theorem 3.4, when the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ are uniform and independent, this assumption ($\mathsf{BASIS}_{\mathsf{rand}}$) reduces to the *standard* SIS assumption in a straightforward way. Here, we provide a sketch of the reduction. For ease of exposition, we show that SIS with respect to $\mathbf{A}_i$ (as opposed to $\underline{\mathbf{A}_i}$) is hard given a trapdoor for $\mathbf{B}_\ell$. We also describe the approach for the case $i = 1$, and refer to Theorem 3.4 for the full analysis.

The idea is simple: we set $\mathbf{A}_1$ to be the SIS challenge and sample matrices $\mathbf{A}_2, \ldots, \mathbf{A}_\ell$ together with trapdoors $\mathbf{R}_2, \ldots, \mathbf{R}_\ell$ (i.e., $\mathbf{A}_i\mathbf{R}_i = \mathbf{G}$). Let $\tilde{\mathbf{B}}_\ell$ be $\mathbf{B}_\ell$ with the first column block removed (i.e., the column block containing $\mathbf{A}_1$). Then, using $\mathbf{R}_2, \ldots, \mathbf{R}_\ell$ we can construct a trapdoor $\tilde{\mathbf{R}}_\ell$ for $\tilde{\mathbf{B}}_\ell$ (i.e., $\tilde{\mathbf{B}}_\ell\tilde{\mathbf{R}}_\ell = \mathbf{G}_{n\ell} = \mathbf{I}_{n\ell} \otimes \mathbf{g}^\intercal$):

$$\tilde{\mathbf{B}}_\ell = \left[\begin{array}{ccc|c} \mathbf{0} & \cdots & \mathbf{0} & -\mathbf{G} \\ \mathbf{A}_2 & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & -\mathbf{G} \end{array}\right] \quad \text{and} \quad \tilde{\mathbf{R}}_\ell = \begin{bmatrix} -\mathbf{R}_2 & \mathbf{R}_2 & & \\ \vdots & & \ddots & \\ -\mathbf{R}_\ell & & & \mathbf{R}_\ell \\ -\mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}$$

Using standard trapdoor extension techniques [ABB10a, ABB10b, CHKP10, MP12], we can *extend* $\tilde{\mathbf{R}}_\ell$ to obtain a trapdoor $\mathbf{R}_\ell$ for $\mathbf{B}_\ell$. This yields a $\mathsf{BASIS}_{\mathsf{rand}}$ instance (i.e., comprised of the matrix $\mathbf{A}_1$, the matrix $\mathbf{B}_\ell$, and the trapdoor $\mathbf{R}_\ell$). Thus, an adversary that breaks the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption implies an adversary that breaks SIS (with comparable parameters). We give the formal analysis in Theorem 3.4.

Functional commitments using structured $\mathbf{A}_i$. Instantiating our framework with uniform $\mathbf{A}_i \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ (as in PPS_1), we obtain a vector commitment scheme with private openings and supporting large messages from the standard SIS assumption. If we instead use a structured set of matrices $\mathbf{A}_i$ as in MatrixACLMT, we obtain functional commitments, polynomial commitments, and aggregatable commitments.

We start by describing our functional commitment scheme. Our starting point is to consider the main verification relation from Eq. (1.1) and generalize it in two ways: (1) we replace the matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times m}$ with *structured* matrices; and (2) we consider a *matrix* extension of the verification relation. In particular, we first sample $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$. Then, for each $i \in [\ell]$, we sample an invertible matrix $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n \times n}$ and set $\mathbf{A}_i \leftarrow \mathbf{W}_i\mathbf{A}$. We now consider a matrix analog of the verification relation from Eq. (1.1) where each target vector $\mathbf{t}_i$ is replaced with the matrix $\mathbf{W}_i\mathbf{G}$ (this choice will be helpful for supporting *functional* openings). Our matrix verification relation is now

$$\mathbf{C} = \mathbf{A}_i\mathbf{V}_i + x_i\mathbf{W}_i\mathbf{G}. \tag{1.4}$$

Our goal now is to sample a tuple $(\mathbf{V}_1, \ldots, \mathbf{V}_\ell, \mathbf{C})$ that satisfy Eq. (1.4) for all $i \in [\ell]$ and where $\mathbf{V}_1, \ldots, \mathbf{V}_\ell \in \mathbb{Z}_q^{m \times m}$ are short. As before, the commitment $\mathbf{C}$ can be large and we specifically define it to be $\mathbf{C} = \mathbf{G}\hat{\mathbf{C}}$, where $\hat{\mathbf{C}} \in \mathbb{Z}_q^{m \times m}$ is short. This way, we can sample $\hat{\mathbf{C}}$ using an analogous trapdoor sampling procedure as before. Specifically, the trapdoor for the same matrix $\mathbf{B}_\ell$ from Eq. (1.3) allows us to jointly sample short openings $\mathbf{V}_1, \ldots, \mathbf{V}_\ell$ along with a matrix $\hat{\mathbf{C}}$ that satisfy Eq. (1.4):

$$\mathbf{B}_\ell \begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_\ell \\ \hat{\mathbf{C}} \end{bmatrix} = \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & -\mathbf{G} \end{array}\right] \cdot \begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_\ell \\ \hat{\mathbf{C}} \end{bmatrix} = \begin{bmatrix} -x_1\mathbf{W}_1\mathbf{G} \\ \vdots \\ -x_\ell\mathbf{W}_\ell\mathbf{G} \end{bmatrix}. \tag{1.5}$$

By construction, for all $i \in [\ell]$, we have that $\mathbf{A}_i\mathbf{V}_i - \mathbf{G}\hat{\mathbf{C}} = -x_i\mathbf{W}_i\mathbf{G}$, or equivalently, $\mathbf{C} = \mathbf{G}\hat{\mathbf{C}} = \mathbf{A}_i\mathbf{V}_i + x_i\mathbf{W}_i\mathbf{G}$ and Eq. (1.4) holds. We now show that this directly extends to yield a succinct functional commitment. Since $\mathbf{A}_i = \mathbf{W}_i\mathbf{A}$ and $\mathbf{W}_i$ is invertible, we can rewrite Eq. (1.4) as

$$\mathbf{W}_i^{-1}\mathbf{C} = \mathbf{A}\mathbf{V}_i + x_i\mathbf{G},$$

where $\mathbf{V}_i$ is *short*. Readers familiar with the homomorphic encryption scheme of Gentry et al. [GSW13] or the homomorphic signature scheme of Gorbunov et al. [GVW15] may recognize that $\mathbf{W}_i^{-1}\mathbf{C}$ is an encryption of x_i under

randomness $\mathbf{V}_i$ or that $\mathbf{V}_i$ is a signature on x_i under the verification key $\mathbf{W}_i^{-1}\mathbf{C}$. Thus, we can use the same lattice-based homomorphic evaluation machinery [GSW13, BGG+14] to homomorphically compute an opening to $f(\mathbf{x})$ for an arbitrary Boolean circuit $f\colon \{0,1\}^\ell \to \{0,1\}$.

In slightly more detail, let $\tilde{\mathbf{C}} = [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$ and $\tilde{\mathbf{V}} = [\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell]$. Then,

$$\mathbf{A}\tilde{\mathbf{V}} = \mathbf{A}[\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell] = [\mathbf{W}_1^{-1}\mathbf{C} - x_1\mathbf{G} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C} - x_\ell\mathbf{G}] = \tilde{\mathbf{C}} - \mathbf{x}^\intercal \otimes \mathbf{G}.$$

Using the homomorphic evaluation techniques from [GSW13, BGG+14], there exists a short matrix $\mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}}$ that depends on $\tilde{\mathbf{C}}$, f, and $\mathbf{x}$ such that

$$(\tilde{\mathbf{C}} - \mathbf{x}^\intercal \otimes \mathbf{G}) \cdot \mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} = \tilde{\mathbf{C}}_f - f(\mathbf{x}) \cdot \mathbf{G}, \tag{1.6}$$

where $\tilde{\mathbf{C}}_f$ is a matrix that can be efficiently computed from $\tilde{\mathbf{C}}$ and f. To open $\mathbf{C}$ to a function f, the user computes $\tilde{\mathbf{V}}_f \leftarrow \tilde{\mathbf{V}} \cdot \mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}}$. To verify a candidate value $y \in \{0,1\}$ with respect to a function f and commitment $\mathbf{C}$, the verifier first computes $\tilde{\mathbf{C}}_f$ from $(\mathbf{C}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, f)$ and then checks that $\tilde{\mathbf{V}}_f$ is short and moreover,

$$\mathbf{A}\tilde{\mathbf{V}}_f = \tilde{\mathbf{C}}_f - y \cdot \mathbf{G}.$$

For correctness, observe that

$$\mathbf{A}\tilde{\mathbf{V}}_f = \mathbf{A}\tilde{\mathbf{V}}\mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} = (\tilde{\mathbf{C}} - \mathbf{x}^\intercal \otimes \mathbf{G}) \cdot \mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} = \tilde{\mathbf{C}}_f - f(\mathbf{x}) \cdot \mathbf{G}.$$

For binding, we require that SIS is hard with respect to $\mathbf{A}$ even given the matrix $\mathbf{B}_\ell$ and a trapdoor for $\mathbf{B}_\ell$. We refer to this instance of the BASIS assumption with structured $\mathbf{A}_i$'s as $\mathsf{BASIS}_{\mathsf{struct}}$. Since the matrices $\mathbf{A}_i$ that comprise $\mathbf{B}_\ell$ are now correlated, we do not know how to reduce $\mathsf{BASIS}_{\mathsf{struct}}$ to the standard SIS assumption. Nonetheless, $\mathsf{BASIS}_{\mathsf{struct}}$ is a falsifiable assumption under which we obtain a succinct functional commitment for all bounded-depth Boolean circuits. This is the first succinct functional commitment for general circuits from a falsifiable assumption. We provide the full description in Section 4 and a comparison to previous succinct functional commitments in Table 1.

Functional commitments with private openings. Using the approach from [GVW15] for constructing context-hiding homomorphic signatures [GVW15], we can easily extend our functional commitment scheme above to support *private* openings (i.e., where the commitment $\mathbf{C}$ and the opening $\tilde{\mathbf{V}}_f$ reveals nothing more about the input $\mathbf{x}$ other than the value $f(\mathbf{x})$). We sketch the approach here. Let $\mathbf{C}$ be a commitment to $\mathbf{x}$ and let $\tilde{\mathbf{V}}_f = \tilde{\mathbf{V}} \cdot \mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}}$ be the opening computed as described above. Then, define the matrix $\mathbf{D}_f$ to be

$$\mathbf{D}_f = [\mathbf{A} \mid \tilde{\mathbf{C}}_f + (f(\mathbf{x}) - 1) \cdot \mathbf{G}] = [\mathbf{A} \mid \mathbf{A}\tilde{\mathbf{V}}_f + (2f(\mathbf{x}) - 1) \cdot \mathbf{G}].$$

Since $\tilde{\mathbf{V}}_f$ is short and $2f(\mathbf{x}) - 1 \in \{-1, 1\}$, the matrix $\begin{bmatrix} \tilde{\mathbf{V}}_f \\ -\mathbf{I}_n \end{bmatrix}$ is a trapdoor for $\mathbf{D}_f$. We now include a random target $\mathbf{u} \in \mathbb{Z}_q^n$ as part of the CRS, and define the opening to be a *random* short vector $\mathbf{v}_f$ where $\mathbf{D}_f\mathbf{v}_f = \mathbf{u}$. The honest prover samples $\mathbf{v}_f$ using the trapdoor for $\mathbf{D}_f$ (derived from $\tilde{\mathbf{V}}_f$). To check an opening $\mathbf{v}_f$ is a valid opening to a value $y \in \{0,1\}$ with respect to a function f and commitment $\mathbf{C}$, the verifier computes $\tilde{\mathbf{C}}_f$ from $(\mathbf{C}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, f)$ as before, defines the matrix $\mathbf{D}_f = [\mathbf{A} \mid \tilde{\mathbf{C}}_f + (y - 1) \cdot \mathbf{G}]$, and finally, checks that $\mathbf{D}_f\mathbf{v}_f = \mathbf{u}$. To argue that $\mathbf{v}_f$ hides all information about $\mathbf{x}$ other than what is revealed by $f(\mathbf{x})$, observe that the matrix $\mathbf{D}_f$ depends only on $f(\mathbf{x})$ and *not* $\mathbf{x}$. Thus, given a trapdoor for $\mathbf{A}$ (which can be extended into a trapdoor for $\mathbf{D}_f$ for all f), and the value $f(\mathbf{x})$, we can sample a short $\mathbf{v}_f$ such that $\mathbf{D}_f\mathbf{v}_f = \mathbf{u}$ whose distribution is statistically close to the real opening. This latter procedure only depends on $f(\mathbf{x})$ and not $\mathbf{x}$, so hiding follows. While this construction is hiding, we do not know how to show that it is binding; however, it does satisfy the *weaker* notion of *target binding* where binding holds for all honestly-generated commitments. We provide the full details and analysis in Section 4.2.

Polynomial commitments. We can obtain a polynomial commitment over $\mathbb{Z}_q$ via a simple adaptation of our functional commitment. The starting point is to construct a functional commitment scheme for linear functions on

Scheme	CRS Size	Function Class	Assumption	Fast Verification
Folklore	$\mathrm{poly}(\lambda, \log \ell)$	Boolean circuits	CRHF + SNARK*	✓
[LRY16]	$O(\ell)$	linear functions	bilinear maps	✓
[PPS21]†	$s \cdot \mathrm{poly}(\lambda, d)$	depth d Boolean circuits‡	SIS	✗
[ACL+22]	$\ell^{2d} \cdot \mathrm{poly}(\lambda)$	degree d polynomials§	k-R-ISIS	✓
Construction 4.2	$\ell^2 \cdot \mathrm{poly}(\lambda, d, \log \ell)$	depth d Boolean circuits	$\mathsf{BASIS}_{\mathsf{struct}}$	✗
Construction 4.14	$\ell^2 \cdot \mathrm{poly}(\lambda, \log \ell)$	linear functions	$\mathsf{BASIS}_{\mathsf{struct}}$	✓

* Collision-resistant hash functions (CRHFs) together with a succinct non-interactive argument of knowledge (SNARK).
†This scheme is in a significantly weaker model that requires an *online trusted* authority to issue opening keys.
‡The Boolean circuit has size at most s.
§Only supports commitments and openings to *small* values.

Table 1: Summary of *succinct* functional commitments. For each scheme, we report the size of the CRS as a function of the security parameter λ and the input length ℓ. We say that a scheme supports "fast verification" if after an *input-independent* preprocessing step, the verification running time is sublinear in ℓ. In all schemes, the size of the commitment and the openings are polylogarithmic in the input length ℓ.

$\mathbb{Z}_q^\ell$ (as opposed to a function on the binary domain $\{0,1\}^\ell$). We first consider linear functions with small coefficients. Let $\mathbf{z} \in \{0,1\}^\ell$ be a vector and define the linear function $f_{\mathbf{z}}(\mathbf{x}) := \mathbf{z}^\intercal \mathbf{x}$. We use the same commitment and opening structure as in our functional commitment. Namely, a commitment $\mathbf{C}$ and the openings $\mathbf{V}_1, \ldots, \mathbf{V}_\ell$ for an input $\mathbf{x}$ satisfy $\mathbf{A}\mathbf{V}_i = \mathbf{W}_i^{-1}\mathbf{C} - x_i\mathbf{G}$, where $x_i \in \mathbb{Z}_q$ now. Observe that

$$\underbrace{\sum_{i\in[\ell]} z_i \mathbf{A}\mathbf{V}_i}_{\mathbf{A}\mathbf{V}_{\mathbf{z}}} = \sum_{i\in[\ell]} z_i \mathbf{W}_i^{-1}\mathbf{C} - \sum_{i\in[\ell]} z_i x_i \mathbf{G} = \underbrace{\sum_{i\in[\ell]} z_i \mathbf{W}_i^{-1}\mathbf{C}}_{\hat{\mathbf{c}}_{\mathbf{z}}} - \underbrace{(\mathbf{z}^\intercal \mathbf{x}) \cdot \mathbf{G}}_{f_{\mathbf{z}}(\mathbf{x})\cdot\mathbf{G}}.$$

Thus, $\mathbf{V}_{\mathbf{z}} = \sum_{i\in[\ell]} z_i \mathbf{V}_i$ is an opening to the function $f_{\mathbf{z}}$. Here, we need $\mathbf{z} \in \{0,1\}^\ell$ to be short so $\mathbf{V}_{\mathbf{z}}$ is short. To extend to arbitrary linear functions over $\mathbb{Z}_q^\ell$ (rather than short ones), we rely on standard binary decomposition and blow up the vector dimension by a factor of $O(\log q)$. Namely, to commit to a vector $\mathbf{x}$, the user now commits to $\mathbf{x} \otimes \mathbf{g}^\intercal$, and to open to a linear function $f_{\mathbf{z}}$ where $\mathbf{z} \in \mathbb{Z}_q^n$, the user constructs an opening with respect to $f_{\mathbf{g}^{-1}(\mathbf{z})}$. This yields a functional commitment scheme for linear functions over $\mathbb{Z}_q^\ell$.

As observed by Libert et al. [LRY16], a functional commitment scheme for linear functions over $\mathbb{Z}_q^\ell$ directly implies a polynomial commitment over $\mathbb{Z}_q$ for polynomials of degree up to $d = \ell - 1$. Namely, a commitment to a polynomial $f \in \mathbb{Z}_q[x]$ of degree d is a vector commitment to the coefficients of f. To open the commitment to a point $x \in \mathbb{Z}_q$, the user constructs a linear opening with respect to the evaluation vector $[1, x, x^2, \ldots, x^d]$. For this to work, it is critical that our functional commitment for linear functions over $\mathbb{Z}_q^\ell$ supports committing to and opening to *arbitrary* $\mathbb{Z}_q$ values (and not just *small* values). Thus, we obtain a polynomial commitment scheme where the commitment size and the opening size is $\mathrm{poly}(\lambda, \log d)$. We provide the full details in Section 4.1.

Aggregatable commitments. Another application of using structured matrices $\mathbf{A}_i$ is it immediately gives an aggregatable commitment. As before, we instantiate our framework with $\mathbf{A}_i = \mathbf{W}_i\mathbf{A}$. We also sample target vectors $\mathbf{u}_1, \ldots, \mathbf{u}_\ell \xleftarrow{R} \mathbb{Z}_q^n$ and include them as part of the CRS. To commit to an input $\mathbf{x} \in \mathbb{Z}_q^\ell$, we sample $(\mathbf{v}_1, \ldots, \mathbf{v}_\ell, \mathbf{c})$ where

$$\mathbf{B}_\ell \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} = \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & -\mathbf{G} \end{array}\right] \cdot \begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} = \begin{bmatrix} -x_1\mathbf{W}_1\mathbf{u}_1 \\ \vdots \\ -x_\ell\mathbf{W}_\ell\mathbf{u}_\ell \end{bmatrix}.$$

Let $\mathbf{c} = \mathbf{G}\hat{\mathbf{c}}$. Then, for all $i \in [\ell]$, $\mathbf{A}_i\mathbf{v}_i - \mathbf{c} = -x_i\mathbf{W}_i\mathbf{u}_i$, or equivalently, $\mathbf{W}_i^{-1}\mathbf{c} = \mathbf{A}\mathbf{v}_i + x_i\mathbf{u}_i$. Observe now that this scheme immediately supports aggregation: for any set $S \subseteq [\ell]$,

$$\sum_{i\in S} \mathbf{W}_i^{-1}\mathbf{c} = \mathbf{A}\sum_{i\in S}\mathbf{v}_i - \sum_{i\in S} x_i\mathbf{u}_i.$$

Thus, $\sum_{i\in S}\mathbf{v}_i$ is an opening to all of the indices in S. We show in Section 5 that under the same $\mathsf{BASIS}_{\mathsf{struct}}$ assumption (i.e., SIS is hard with respect to $\mathbf{A}$ given $\mathbf{B}_\ell$ and a trapdoor for $\mathbf{B}_\ell$), this scheme satisfies "same-set binding." This means no efficient adversary can open a commitment $\mathbf{c}$ to different sets of values $\{(i, x_i)\}_{i\in S}$ and $\{(i, x_i')\}_{i\in S}$ for the *same* set S. Unlike our vector commitment construction, the security of our aggregatable construction only holds when the input vector $\mathbf{x}$ is short (i.e., our reduction to the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption in Theorem 5.6 constructs an SIS solution whose norm scales with the magnitude of the opened values).

Our aggregatable vector commitment scheme does not satisfy the stronger notion of "different-set binding." This means an efficient adversary may be able to construct a commitment $\mathbf{c}$ along with valid openings $\{(i, x_i)\}_{i\in S}$ and $\{(i, x_i')\}_{i\in T}$ to (distinct) sets S and T, respectively, such that $x_i = 0$ and $x_i' = 1$. Indeed in Remark 5.12, we describe an explicit attack where an adversary can use the trapdoor for $\mathbf{B}_\ell$ to (heuristically) obtain a trapdoor for the matrix $[\mathbf{W}_S^{-1}\mathbf{A} \mid \mathbf{W}_T^{-1}\mathbf{A}]$ whenever $S \neq T$ and where $\mathbf{W}_S = \sum_{i\in S}\mathbf{W}_i^{-1}$ and $\mathbf{W}_T = \sum_{i\in T}\mathbf{W}_i^{-1}$. Knowledge of this trapdoor allows an adversary to construct a valid opening to $\{(i, x_i)\}_{i\in S}$ and $\{(i, x_i')\}_{i\in T}$ for any choice of x_i, x_i'.

Conceptually, our approach for constructing an aggregatable vector commitment scheme is to replace the *fixed* target value $x_i\mathbf{e}_1$ from our basic vector commitment with *random* linear combinations of $\{x_i\}_{i\in S}$ (where the coefficients of the random linear combination are the vectors $\{\mathbf{u}_i\}_{i\in S}$). A similar approach was used for aggregating pairing-based signatures in [BDN18] and for aggregating openings (to constant-degree polynomials) in [ACL+22].

Aggregating functional commitments. The same aggregation technique applies to our succinct functional commitment scheme. Recall the functional commitment verification relation from Eq. (1.6): $\mathbf{A}\tilde{\mathbf{V}}_f = \tilde{\mathbf{C}}_f - y\cdot\mathbf{G}$. Here $\tilde{\mathbf{V}}_f$ is the opening, $\tilde{\mathbf{C}}_f$ is a function of the commitment $\mathbf{C}$ and the function f, and y is the value. To support aggregating up to t openings, we include random vectors $\mathbf{u}_1, \ldots, \mathbf{u}_t \xleftarrow{R} \mathbb{Z}_q^n$ in the CRS. Then, given a collection of openings $(f_1, y_1, \tilde{\mathbf{V}}_1), \ldots, (f_t, y_t, \tilde{\mathbf{V}}_t)$ where the functions $f_1, \ldots, f_t$ are sorted in lexicographic order, we define the aggregate opening to be $\mathbf{v} = \sum_{i\in[t]}\tilde{\mathbf{V}}_i \cdot \mathbf{G}^{-1}(\mathbf{u}_i)$. The new verification relation is then

$$\sum_{i\in[t]} \tilde{\mathbf{C}}_{f_i}\cdot\mathbf{G}^{-1}(\mathbf{u}_i) = \mathbf{A}\mathbf{v} - \sum_{i\in[t]} y_i\mathbf{u}_i.$$

Similar to the case with aggregatable vector commitments, we can argue "same-function binding," where no efficient adversary can open a commitment $\mathbf{C}$ to two different sets of values $(y_1, \ldots, y_t) \neq (y_1', \ldots, y_t')$ with respect to the same set of functions $(f_1, \ldots, f_t)$. We provide the full analysis in Section 5.1.

Understanding the BASIS assumption. The BASIS assumptions we introduce in this work enable a number of new constructions of vector commitments and their generalizations. While the basic version $\mathsf{BASIS}_{\mathsf{rand}}$ that suffices for vector commitments can be reduced to the standard SIS assumption (Theorem 3.4), the more general version $\mathsf{BASIS}_{\mathsf{struct}}$ with structured matrices does not. Nonetheless, the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption is still falsifiable and thus, yields the first succinct functional commitments and polynomial commitments from falsifiable lattice assumptions. We invite cryptanalysis of our new family of SIS assumptions and are also optimistic that the assumption as well as our general methodology will be helpful for realizing new lattice-based cryptographic primitives.

In Section 6, we compare the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption to similar assumptions made in [ACL+22]. We show a close connection between the two families of assumptions. We can also view the BASIS assumptions as a new type of "q-type" assumption in the lattice-based setting (where the size of the assumption grows with the input dimension).

1.3 Related Work and Concurrent Work

In this section, we describe some closely-related notions of functional commitments as well as a comparison to two concurrent construction of functional commitments for Boolean circuits [BCFL22, dCP23].

Scheme	$\lvert\mathsf{crs}\rvert$	$\lvert\sigma\rvert$	$\lvert\pi\rvert$	T_{Commit}	T_{Open}	T_{Verify}	Assumption	FV	TR	AS
[dCP23]	$\lvert y\rvert$	1	$\lvert y\rvert$	$\lvert F\rvert$	$\lvert F\rvert$	$\lvert y\rvert$	SIS	✗	✓	✗*
[BCFL22]	w^5	1	1	$\lvert x\rvert$	$\lvert F\rvert$	$\lvert F\rvert$	twin-k-M-ISIS	✓	✗	✓
Construction 4.2†	$\lvert x\rvert^2$	1	1	$\lvert x\rvert$	$\lvert F\rvert$	$\lvert F\rvert$	$\mathsf{BASIS}_{\mathsf{struct}}$	✗	✗	✓

*Can be made adaptively-secure via complexity leveraging (and in some cases, relying on sub-exponential hardness of SIS). The use of complexity leveraging increases all parameter sizes by $\lvert y\rvert^{\varepsilon}$ for a constant $\varepsilon > 0$.
†For comparison purposes, we consider a variant with a deterministic commitment and opening procedure where Commit runs in quasi-linear time. We refer to Remark 4.12 for more details.

Table 2: Comparison to concurrent works [BCFL22, dCP23] on lattice-based succinct functional commitments for general Boolean circuits. To establish a common basis for comparison between schemes, we fix a bivariate function $F(x, y)$ with a single output. We consider the setting where the user commits to input x and opens to the value $F(x, y)$. In [BCFL22] and in our construction, the user commits to x and opens to the function $F_y(\cdot) := F(\cdot, y)$. In the "dual" notion of [dCP23], the user would commit to the function $F_x(\cdot) := F(x, \cdot)$ and open to the value y. We write w to denote the width, d to denote the depth, and $\lvert F\rvert$ to denote the size of the Boolean circuit computing F. Our construction and [dCP23] impose an a prori bound on d whereas [BCFL22] imposes an a priori bound on w. For each scheme, we report the size of the common reference string crs, the size of the commitment σ, the size of the opening π, and the running times T_{Commit}, T_{Open}, T_{Verify} of the commit, opening, and verification algorithms, respectively. For simplicity, we suppress $\mathsf{poly}(\lambda, d)$ terms as well as polylogarithmic terms in $\lvert x\rvert$, $\lvert y\rvert$, and $\lvert F\rvert$. We say a scheme supports "fast verification" (**FV**) if after an input-independent preprocessing step, the verification time is *sublinear* in $\lvert x\rvert$, $\lvert y\rvert$, and $\lvert F\rvert$. We also indicate whether the scheme has a *transparent* setup (**TR**) and whether it is *adaptively secure* (**AS**). We say the commitment is adaptively secure if in the binding security game, the adversary can choose x, y *after* seeing the common reference string, and that it is selectively secure if the adversary has to choose either x or y beforehand.

Interactive functional commitments. Functional commitments have also been extensively studied in the *interactive* model. In these settings, there is typically an *interactive* opening procedure between the committer and the verifier. Ishai et al. [IKO07] introduced interactive functional commitments for linear function, and subsequently, Bitansky and Chiesa [BC12] extended it to interactive functional commitments. In both cases, these were used to construct (interactive) succinct arguments without relying on probabilistically-checkable proofs (PCPs). Alternatively, using PCPs or their generalization to interactive oracle proofs [BCS16], we can also construct a functional commitment from any collision-resistant hash function via Kilian's interactive succinct argument [Kil92], which can then be made non-interactive in the random oracle model [Mic00]. Our focus in this work is on *non-interactive* vector and functional commitments in the plain model (i.e., *without* random oracles).

Homomorphic commitments. Homomorphic commitments [GSW13, GVW15] also allow a user to commit to an input x and later on, open up the commitment to an arbitrary function evaluation $f(x)$. The key difference is that there are no succinctness requirements on a homomorphic commitment. In particular, the size of the commitment in previous constructions [GSW13, GVW15] scaled linearly with the input length. On the flip side, homomorphic commitments can be constructed from standard lattice assumptions.

Concurrent works. Recently, two concurrent works [dCP23, BCFL22] introduced new constructions of functional commitments. The authors of [dCP23] introduce a *vector commitment* with a short CRS (i.e., logarithmic in the input dimension) from the standard SIS assumption. In their construction, the CRS is a *uniform* random string rather than a structured reference string. They then generalize their construction to obtain a "dual" functional commitment[8] scheme for (bounded-depth) Boolean circuits from the standard SIS assumption. Their functional commitment does *not* support succinct openings. Namely, the size of the opening in their construction scales either with the length of

[8]In a "dual" functional commitment scheme, the commitment is to a function f and the opening is to a point x. In the version we are considering, the commitment is to an input x and the opening is to a function f. In the case of functional commitments for circuits, we can translate between these two notions using universal circuits, but this transformation will incur an a priori bound on the description length of the function.

the input x (when committing to a function f and opening to an input x) or the description length of the function f (when committing to an input x and opening to a function f). In this work, we consider the setting where *both* the commitment and the opening are succinct (i.e., polylogarithmic in both $|x|$ and $|f|$). However, we rely on a stronger (but still falsifiable) version of the SIS assumption.

The authors of [BCFL22] provide new pairing-based and lattice-based constructions of functional commitments (with succinct commitments *and* openings) for arithmetic circuits of *a priori* bounded width. Like our scheme, the size of the openings in their lattice-based instantiation grows with the depth of the circuit. Their scheme also relies on new, non-standard, but falsifiable generalizations of SIS (specifically, variants of the [ACL+22] family of assumptions used to obtain lattice-based preprocessing succinct arguments). The functional commitments we introduce in this work support *arbitrary* Boolean circuits (without a width constraint), and similarly, the sizes of the commitment and the openings scale with the *depth* of the computation. The structured version of our basis-augmented SIS assumption ($\mathsf{BASIS}_{\mathsf{struct}}$) is conceptually similar to the [ACL+22] family of assumptions (the latter imposes even more structure), and we provide a more detailed discussion in Section 6. We provide a comparison with the lattice-based functional commitments from [dCP23, BCFL22] in Table 2.

2 Preliminaries

We write λ to denote the security parameter. For a positive integer $n \in \mathbb{N}$, we write $[n]$ to denote the set $\{1, \ldots, n\}$. For integers $a, b \in \mathbb{N}$, we write $[a, b]$ to denote the set $\{a, a+1, \ldots, b\}$. For a positive integer $q \in \mathbb{N}$, we write $\mathbb{Z}_q$ to denote the integers modulo q. We use bold uppercase letters to denote matrices (e.g., $\mathbf{A}, \mathbf{B}$) and bold lowercase letters to denote vectors (e.g., $\mathbf{u}, \mathbf{v}$). We use non-boldface letters to refer to their components: $\mathbf{v} = (v_1, \ldots, v_n)$. For matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times m}$, we write $\mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \in \mathbb{Z}_q^{n\ell \times m\ell}$ to denote the block diagonal matrix with blocks $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ along the main diagonal (and 0 elsewhere).

We write $\mathsf{poly}(\lambda)$ to denote a fixed function that is $O(\lambda^c)$ for some $c \in \mathbb{N}$ and $\mathsf{negl}(\lambda)$ to denote a function that is $o(\lambda^{-c})$ for all $c \in \mathbb{N}$. For functions $f = f(\lambda), g = g(\lambda)$, we write $g \geq O(f)$ to denote that there exists a fixed function $f'(\lambda) = O(f)$ such that $g(\lambda) > f'(\lambda)$ for all $\lambda \in \mathbb{N}$. We say an event occurs with overwhelming probability if its complement occurs with negligible probability. An algorithm is efficient if it runs in probabilistic polynomial time in its input length. We say that two families of distributions $\mathcal{D}_1 = \{\mathcal{D}_{1,\lambda}\}_{\lambda \in \mathbb{N}}$ and $\mathcal{D}_2 = \{\mathcal{D}_{2,\lambda}\}_{\lambda \in \mathbb{N}}$ are computationally indistinguishable if no efficient algorithm can distinguish them with non-negligible probability, and we say they are statistically indistinguishable if the statistical distance $\Delta(\mathcal{D}_1, \mathcal{D}_2)$ is bounded by a negligible function $\mathsf{negl}(\lambda)$.

Min-entropy. We recall some basic definitions on min-entropy. Our definitions are taken from [DRS04, GVW15]. For a (discrete) random variable X, we write $\mathbf{H}_\infty(X) = -\log(\max_x \Pr[X = x])$ to denote its min-entropy. For two (possibly correlated) discrete random variables X and Y, we define the average min-entropy of X given Y to be $\mathbf{H}_\infty(X \mid Y) = -\log(\mathbb{E}_{y \leftarrow Y} \max_x \Pr[X = x \mid Y = y])$. The optimal probability of an unbounded adversary guessing X given the correlated value Y is $2^{-\mathbf{H}_\infty(X|Y)}$.

2.1 Lattice Preliminaries

In this section, we recall some standard notions from lattice-based cryptography. Unless otherwise noted, we use the ℓ_∞-norm for vectors and matrices. For a vector $\mathbf{u}$, we write $\|\mathbf{u}\| := \max_i |x_i|$, and for a matrix $\mathbf{A}$, we write $\|\mathbf{A}\| = \max_{i,j} |A_{i,j}|$. For $k \in \mathbb{N}$, we write $\mathbf{I}_k \in \mathbb{Z}_q^{k \times k}$ to denote the k-by-k identity matrix. We now recall the short integer solution (SIS) problem [Ajt96] as well as its inhomogeneous variant:

Assumption 2.1 (Short Integer Solution [Ajt96])**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, and $\beta = \beta(\lambda)$ be lattice parameters. We say the short integer solution problem $\mathsf{SIS}_{n,m,q,\beta}$ holds if for all efficient adversaries $\mathcal{A}$,

$$\Pr\left[\mathbf{A}\mathbf{x} = \mathbf{0} \text{ and } 0 < \|\mathbf{x}\| \leq \beta : \begin{array}{c} \mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}; \\ \mathbf{x} \leftarrow \mathcal{A}(1^\lambda, \mathbf{A}) \end{array}\right] = \mathsf{negl}(\lambda).$$

We also define the *inhomogeneous SIS* (ISIS) assumption where the target $\mathbf{0}$ in the above assumption is replaced by a uniform random vector $\mathbf{y} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$. When $m = \mathsf{poly}(n, \log q)$ and for sufficiently-large $q \geq \beta \cdot \mathsf{poly}(n)$, hardness of the

SIS and ISIS problems can be based on approximating certain worst-case lattice problems on n-dimensional lattices to within a $\beta \cdot \mathrm{poly}(n)$ factor [Ajt96, MR04, GPV08].

Lattices. Let $\mathbf{B} \in \mathbb{R}^{n \times n}$ be a full-rank matrix (over $\mathbb{R}$). Then, the n-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$ generated by $\mathbf{B}$ is $\Lambda = \mathbf{B} \cdot \mathbb{Z}^n = \{\mathbf{Bz} : \mathbf{z} \in \mathbb{Z}^n\}$. For a vector $\mathbf{c} \in \mathbb{R}^n$ and a lattice $\Lambda \subset \mathbb{R}^n$, we write $\mathbf{c} + \Lambda = \{\mathbf{c} + \mathbf{x} : \mathbf{x} \in \Lambda\}$ to denote the coset of Λ associated with $\mathbf{c}$. For a lattice Λ, we define the *dual lattice* $\Lambda^* = \{\mathbf{w} \in \mathbb{R}^n : \forall \mathbf{x} \in \Lambda, \mathbf{w}^\intercal \mathbf{x} \in \mathbb{Z}\}$. Finally, if $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ for integers n, m, q, we define the q-ary lattices

$$\Lambda^\perp(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{Ax} = \mathbf{0} \bmod q\}$$
$$\Lambda(\mathbf{A}) = \{\mathbf{y} \in \mathbb{Z}^m : \mathbf{y} = \mathbf{A}^\intercal \mathbf{x} \bmod q \text{ for some } \mathbf{x} \in \mathbb{Z}^n\}.$$

For all $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, it holds that $\Lambda(\mathbf{A}) = q \cdot (\Lambda^\perp(\mathbf{A}))^*$. For a lattice Λ, we write $\lambda_1^\infty(\Lambda) := \min_{\mathbf{0} \neq \mathbf{v} \in \Lambda} \|\mathbf{x}\|$ to denote ℓ_∞ norm of the shortest non-zero vector in Λ. We will use the simple bound shown (implicitly) in [GPV08, Lemma 5.3]:

Lemma 2.2 (Minimum Distance [GPV08, Lemma 5.3])**.** *Let n, m, q be lattice parameters where q is prime and $m \geq 2n \log q$. Then, for all but a $q^{-n} = \mathrm{negl}(n)$ fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\lambda_1^\infty(\Lambda(\mathbf{A})) \geq q/4$.*

Lemma 2.3 (Minimum Distance of $\Lambda(\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell))$)**.** *Take matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n \times m}$ where $\lambda_1^\infty(\Lambda(\mathbf{A}_i)) \geq t$ for all $i \in [\ell]$. Then, $\lambda_1^\infty(\Lambda(\mathbf{B})) \geq t$ where $\mathbf{B} = \mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell)$.*

Proof. This follows immediately from the definition. Let $\mathbf{B} = \mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell)$. Take any non-zero vector $\mathbf{y} \in \Lambda(\mathbf{B})$. Write $\mathbf{y} \in \mathbb{Z}^{\ell m}$ as the vertical concatenation of $\mathbf{y}_1, \ldots, \mathbf{y}_\ell \in \mathbb{Z}^m$. Since $\mathbf{y} \in \Lambda(\mathbf{B})$, there exists $\mathbf{x}_1, \ldots, \mathbf{x}_\ell \in \mathbb{Z}^n$ such that

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_\ell \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1^\intercal & & \\ & \ddots & \\ & & \mathbf{A}_\ell^\intercal \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_\ell \end{bmatrix} = \begin{bmatrix} \mathbf{A}_1^\intercal \mathbf{x}_1 \\ \vdots \\ \mathbf{A}_\ell^\intercal \mathbf{x}_\ell \end{bmatrix} \bmod q.$$

Since $\mathbf{y} \neq \mathbf{0}$, there exists some $i \in [\ell]$ where $\mathbf{y}_i \neq \mathbf{0}$. Since $\mathbf{y}_i = \mathbf{A}_i^\intercal \mathbf{x}_i \bmod q$, this means $\mathbf{y}_i \in \Lambda(\mathbf{A}_i)$. Since $\lambda_1^\infty(\Lambda(\mathbf{A}_i)) \geq t$, then $\|\mathbf{y}_i\| \geq t$. Finally, $\|\mathbf{y}\| \geq \|\mathbf{y}_i\| \geq t$ so we conclude that $\lambda_1^\infty(\Lambda(\mathbf{B})) \geq t$. □

Leftover hash lemma. We will also use the leftover hash lemma and a simple corollary of it:

Lemma 2.4 (Leftover Hash Lemma [HILL99])**.** *Let n, m, q be lattice parameters and suppose $m \geq 2n \log q$. Then, the statistical distance between the following distributions is at most 2^{-n}:*

$$\{(\mathbf{A}, \mathbf{Ax}) : \mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}, \mathbf{x} \xleftarrow{R} \{0,1\}^m\} \quad \text{and} \quad \{(\mathbf{A}, \mathbf{y}) : \mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}, \mathbf{y} \xleftarrow{R} \mathbb{Z}_q^n\}.$$

Corollary 2.5 (Column-Space of Random Matrix [GPV08, Lemma 5.1])**.** *Let n, q be lattice parameters and take any $m \geq 2n \log q$. Then, for all but a $q^{-n} = \mathrm{negl}(n)$ fraction of matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, the columns of $\mathbf{A}$ generate $\mathbb{Z}_q^n$.*

Discrete Gaussians over lattices. For a Gaussian width parameter $s \in \mathbb{R}^+$ and a dimension $n \in \mathbb{N}$, we write $\rho_s \colon \mathbb{R}^n \to \mathbb{R}$ to denote the Gaussian function $\rho_s(\mathbf{x}) := \exp(-\pi \|\mathbf{x}\|_2^2 / s^2)$. Typically, the input dimension will be *implicit*. For a lattice coset $\mathbf{c} + \Lambda$, we write $\rho_s(\mathbf{c} + \Lambda) = \sum_{\mathbf{x} \in \Lambda} \rho(\mathbf{c} + \mathbf{x})$ to denote the Gaussian mass associated with the coset. We write $D_{\mathbb{Z}^n, s} \equiv D_{\mathbb{Z}, s}^n$ to denote the discrete Gaussian distribution over $\mathbb{Z}^n$ with parameter $s \in \mathbb{R}^+$: namely, $D_{\mathbb{Z}^n, s}(\mathbf{x}) := \rho_s(\mathbf{x}) / \rho_s(\mathbb{Z}^n)$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times t}$, and a vector $\mathbf{v} \in \mathbb{Z}_q^n$, we write $\mathbf{A}_s^{-1}(\mathbf{v})$ to denote the random variable $\mathbf{x} \leftarrow D_{\mathbb{Z}, s}^m$ conditioned on $\mathbf{Ax} = \mathbf{v} \bmod q$. We extend $\mathbf{A}_s^{-1}$ to matrices by applying $\mathbf{A}_s^{-1}$ to each column of the input.

The smoothing parameter. We also recall the notion of the smoothing parameter introduced by Micciancio and Regev [MR04]. For an n-dimensional lattice Λ and a positive real number $\varepsilon > 0$, the smoothing parameter $\eta_\varepsilon(\Lambda)$ is the smallest real value $s > 0$ such that $\rho_{1/s}(\Lambda^*) \leq 1 + \varepsilon$. We now state some lemmas on the smoothing parameter:

Lemma 2.6 (Smoothing Parameter [MR04, Lemma 4.4, implicit])**.** *Let Λ be an n-dimensional lattice. Then, for all $\varepsilon \in (0,1)$, all $s \geq \eta_\varepsilon(\Lambda)$, and all cosets $\mathbf{c} + \Lambda$, $\rho_s(\mathbf{c} + \Lambda) \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right] \cdot \rho_s(\Lambda)$.*

Lemma 2.7 (Smoothing Parameter Bound [Pei07, Lemma 3.5]). *Let Λ be an n-dimensional lattice. Then, for every $\omega(\sqrt{\log n})$ function, there exists a negligible function $\varepsilon(n) = \mathsf{negl}(n)$ such that $\eta_\varepsilon(\Lambda) \le \omega(\sqrt{\log n})/\lambda_1^\infty(\Lambda^*)$.*

Corollary 2.8 (Smoothing Parameter of $\Lambda^\perp(\mathbf{A})$ [GPV08, Lemma 5.3]). *Let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and suppose $\lambda_1^\infty(\Lambda^\perp(\mathbf{A})) \ge t$. Then, for every $\omega(\sqrt{\log m})$ function, there exists a negligible function $\varepsilon(m) = \mathsf{negl}(m)$ such that $\eta_\varepsilon(\Lambda^\perp(\mathbf{A})) \le q/t \cdot \omega(\sqrt{\log m})$.*

Proof. Follows from the fact that $q \cdot (\Lambda^\perp(\mathbf{A}))^* = \Lambda(\mathbf{A})$ and Lemma 2.7:

$$\eta_\varepsilon(\Lambda^\perp(\mathbf{A})) \le \frac{\omega(\sqrt{\log m})}{\lambda_1^\infty((\Lambda^\perp(\mathbf{A}))^*)} = \frac{\omega(\sqrt{\log m})}{1/q \cdot \lambda_1^\infty(\Lambda(\mathbf{A}))} \le q/t \cdot \omega(\sqrt{\log m}). \qquad \square$$

Lemma 2.9 (Gaussian Tail Bound [MP12, Lemma 2.6, adapted]). *Let $\mathbf{B} \in \mathbb{R}^{n\times n}$ be a full-rank basis and let Λ be the n-dimensional lattice generated by $\mathbf{B}$. Suppose $s \ge \eta_\varepsilon(\Lambda)$ for some $\varepsilon \in (0,1)$. Then for all $\mathbf{c} \in \mathbb{R}^n$,*

$$\Pr\left[\, \|\mathbf{v}\| > \sqrt{n}s : \mathbf{v} \leftarrow D_{\mathbf{c}+\Lambda,s} \right] \le 2^{-n} \cdot \frac{1+\varepsilon}{1-\varepsilon}.$$

Our analysis will make use of the following lemma on the marginal distributions of Gaussian-distributed preimages. The proof of this is essentially implicit in the preimage sampling algorithm of Gentry et al. [GPV08], but we provide a full proof here for completeness:

Lemma 2.10 (Discrete Gaussian Preimages). *Let n, m, q be lattice parameters. Take matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{n\times \ell}$ where $\ell = \mathsf{poly}(n, \log q)$. Suppose the columns of $\mathbf{A}$ generate $\mathbb{Z}_q^n$. Let $\mathbf{C} = [\mathbf{A} \mid \mathbf{B}] \in \mathbb{Z}_q^{n\times(m+\ell)}$. Then for all target vectors $\mathbf{t} \in \mathbb{Z}_q^n$ and all width parameters $s \ge \eta_\varepsilon(\Lambda^\perp(\mathbf{A}))$ for some $\varepsilon(m) = \mathsf{negl}(m)$, the statistical distance between the following distributions is $\mathsf{negl}(m)$:*

$$\{\mathbf{v} : \mathbf{v} \leftarrow \mathbf{C}_s^{-1}(\mathbf{t})\} \quad \text{and} \quad \left\{ \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} : \mathbf{v}_2 \leftarrow D_{\mathbb{Z},s}^\ell, \mathbf{v}_1 \leftarrow \mathbf{A}_s^{-1}(\mathbf{t} - \mathbf{B}\mathbf{v}_2) \right\}.$$

In particular, when the columns of $\mathbf{A}$ generate $\mathbb{Z}_q^n$ and $\lambda_1^\infty(\Lambda(\mathbf{A})) \ge t$, it suffices to take $s \ge q/t \cdot \log m$.

Proof. Consider $\mathbf{v} \leftarrow \mathbf{C}_s^{-1}(\mathbf{t})$. In the following, we parse $\mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}$. Consider the distribution D of $\mathbf{v}$:

$$D(\mathbf{v}) = \frac{\rho_s(\mathbf{v})}{\sum_{\mathbf{v}':\mathbf{C}\mathbf{v}'=\mathbf{t}} \rho_s(\mathbf{v}')} = \frac{\rho_s(\mathbf{v}_1)\rho_s(\mathbf{v}_2)}{\sum_{\mathbf{v}_2'\in\mathbb{Z}_q^\ell} \sum_{\mathbf{v}_1':\mathbf{A}\mathbf{v}_1'=\mathbf{t}-\mathbf{B}\mathbf{v}_2'} \rho_s(\mathbf{v}_1')\rho_s(\mathbf{v}_2')} \tag{2.1}$$

Since the columns of $\mathbf{A}$ generate $\mathbb{Z}_q^n$, for every $\mathbf{v}_2' \in \mathbb{Z}_q^\ell$, there exists some $\mathbf{u}_{\mathbf{v}_2'} \in \mathbb{Z}_q^m$ such that $\mathbf{A}\mathbf{u}_{\mathbf{v}_2'} = \mathbf{t} - \mathbf{B}\mathbf{v}_2'$. Let Q be the denominator in Eq. (2.1). Then,

$$Q = \sum_{\mathbf{v}_2'\in\mathbb{Z}_q^\ell} \sum_{\mathbf{v}_1':\mathbf{A}\mathbf{v}_1'=\mathbf{t}-\mathbf{B}\mathbf{v}_2'} \rho_s(\mathbf{v}_1')\rho_s(\mathbf{v}_2') = \sum_{\mathbf{v}_2'\in\mathbb{Z}_q^\ell} \rho_s(\mathbf{v}_2')\rho_s\big(\mathbf{u}_{\mathbf{v}_2'} + \Lambda^\perp(\mathbf{A})\big).$$

Since $s \ge \eta_\varepsilon(\Lambda^\perp(\mathbf{A}))$, Lemma 2.6 says that for all $\mathbf{v}_2' \in \mathbb{Z}_q^\ell$,

$$\rho_s\big(\mathbf{u}_{\mathbf{v}_2'} + \Lambda^\perp(\mathbf{A})\big) \in \left[\tfrac{1-\varepsilon}{1+\varepsilon}, 1\right] \cdot \rho_s(\Lambda^\perp(\mathbf{A})).$$

Thus, we can write $Q = \delta_Q \cdot \rho_s(\mathbb{Z}_q^\ell)\rho_s(\Lambda^\perp(\mathbf{A}))$ for some $\delta_Q \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right]$.

- Consider first the marginal distribution D_2 of $\mathbf{v}_2$:

$$D_2(\mathbf{v}_2) = \sum_{\mathbf{v}_1:\mathbf{A}\mathbf{v}_1=\mathbf{t}-\mathbf{B}\mathbf{v}_2} D\left(\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix}\right) = \sum_{\mathbf{v}_1:\mathbf{A}\mathbf{v}_1=\mathbf{t}-\mathbf{B}\mathbf{v}_2} \frac{\rho_s(\mathbf{v}_1)\rho_s(\mathbf{v}_2)}{\delta_Q \cdot \rho_s(\mathbb{Z}_q^\ell)\rho_s(\Lambda^\perp(\mathbf{A}))} = \frac{\rho_s(\mathbf{u}_{\mathbf{v}_2} + \Lambda^\perp(\mathbf{A}))\rho_s(\mathbf{v}_2)}{\delta_Q \cdot \rho_s(\mathbb{Z}_q^\ell)\rho_s(\Lambda^\perp(\mathbf{A}))} = \frac{\delta}{\delta_Q} D_{\mathbb{Z}^\ell,s}(\mathbf{v}_2),$$

  where $\mathbf{u}_{\mathbf{v}_2} \in \mathbb{Z}_q^m$ is an arbitrary vector that satisfies $\mathbf{A}\mathbf{u}_{\mathbf{v}_2} = \mathbf{t} - \mathbf{B}\mathbf{v}_2$ and $\delta \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right]$ by Lemma 2.6. Since $\varepsilon = \mathsf{negl}(m)$, this means that $\delta/\delta_Q \in [1-\mathsf{negl}(m), 1+\mathsf{negl}(m)]$. Thus, for all $\mathbf{v}_2 \in \mathbb{Z}_q^\ell$, $D_2(\mathbf{v}_2) \in [1-\mathsf{negl}(m), 1+\mathsf{negl}(m)] \cdot D_{\mathbb{Z}^\ell,s}(\mathbf{v}_2)$, so the statistical distance between D_2 and $D_{\mathbb{Z}^\ell,s}$ is bounded by a negligible function $\mathsf{negl}(m)$.

- Consider now the conditional distribution D_1 of $\mathbf{v}_1$ given $\mathbf{v}_2 \leftarrow D_{\mathbb{Z}^\ell,s}$:

$$D_1(\mathbf{v}_1 \mid \mathbf{v}_2) = \frac{D\left(\left[\begin{smallmatrix}\mathbf{v}_1\\ \mathbf{v}_2\end{smallmatrix}\right]\right)}{D_{\mathbb{Z}^\ell,s}(\mathbf{v}_2)} = \frac{\rho_s(\mathbf{v}_1)\rho_s(\mathbf{v}_2)}{\delta_Q \cdot \rho_s(\mathbb{Z}_q^\ell)\rho_s(\Lambda^\perp(\mathbf{A}))} \cdot \frac{\rho_s(\mathbb{Z}_q^\ell)}{\rho_s(\mathbf{v}_2)} = \frac{\delta'}{\delta_Q}\frac{\rho_s(\mathbf{v}_1)}{\rho_s(\mathbf{u}_{\mathbf{v}_2} + \Lambda^\perp(\mathbf{A}))},$$

  where $\mathbf{u}_{\mathbf{v}_2} \in \mathbb{Z}_q^m$ is an arbitrary vector satisfying $\mathbf{A}\mathbf{u}_{\mathbf{v}_2} = \mathbf{t} - \mathbf{B}\mathbf{v}_2$ and $\delta' \in \left[\frac{1-\varepsilon}{1+\varepsilon}, 1\right]$ by Lemma 2.6. By the same argument as above, this is $\mathsf{negl}(m)$-close to the distribution $\mathbf{A}_s^{-1}(\mathbf{t} - \mathbf{B}\mathbf{v}_2)$ and the claim follows.

Finally, consider any matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ where the columns generate $\mathbb{Z}_q^n$ and $\lambda_1^\infty(\Lambda(\mathbf{A})) \geq t$. By Corollary 2.8, for every $\omega(\sqrt{\log m})$ function, there exists a negligible function $\varepsilon(m) = \mathsf{negl}(m)$ such that $\eta_\varepsilon(\Lambda^\perp(\mathbf{A})) \leq q/t \cdot \omega(\sqrt{\log m})$. Taking the $\omega(\sqrt{\log m})$ function to be $\log m$, we have $\eta_\varepsilon(\Lambda^\perp(\mathbf{A})) \leq q/t \log m$ and the claim follows. □

Corollary 2.11 (Discrete Gaussian Preimages of $[\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid \mathbf{B}]$)**.** *Let n, m, q, t be parameters where $m \geq n$. Take any $\ell, k = \mathsf{poly}(n, \log q)$, any collection of matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell \in \mathbb{Z}_q^{n\times m}$ where $\mathbf{A}_i$ is full rank and $\lambda_1^\infty(\Lambda(\mathbf{A}_i)) \geq t$ for all $i \in [\ell]$, any collection of matrices $\mathbf{B}_1, \ldots, \mathbf{B}_\ell \in \mathbb{Z}_q^{n\times k}$, and any target vector $\mathbf{t} \in \mathbb{Z}_q^{n\ell}$. Define the following matrices*

$$\mathbf{C} = \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & \mathbf{B}_1 \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & \mathbf{B}_\ell \end{array}\right] \quad \text{and} \quad \mathbf{t} = \begin{bmatrix}\mathbf{t}_1\\ \vdots \\ \mathbf{t}_\ell\end{bmatrix}.$$

Then, for all width parameters $s \geq q/t \cdot \log(\ell m)$, the statistical distance between the following distributions is $\mathsf{negl}(m)$:

$$\{\mathbf{v} : \mathbf{v} \leftarrow \mathbf{C}_s^{-1}(\mathbf{t})\} \quad \text{and} \quad \left\{ \begin{bmatrix}\mathbf{v}_1\\ \vdots \\ \mathbf{v}_\ell \\ \mathbf{v}_{\ell+1}\end{bmatrix} : \begin{array}{c} \mathbf{v}_{\ell+1} \leftarrow D_{\mathbb{Z},s}^k, \\ \mathbf{v}_i \leftarrow (\mathbf{A}_i)_s^{-1}(\mathbf{t}_i - \mathbf{B}_i\mathbf{u}). \end{array} \right\}. \tag{2.2}$$

Proof. By Lemma 2.3, $\lambda_1^\infty(\Lambda(\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell))) \geq t$. Moreover, since each $\mathbf{A}_i$ is full rank, it holds that $\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell)$ is also full rank. The claim now follows by Lemma 2.10. □

The gadget matrix. We recall the definition of the gadget matrix [MP12]. For positive integers $n, q \in \mathbb{N}$, let $\mathbf{G}_n = \mathbf{I}_n \otimes \mathbf{g}^\intercal \in \mathbb{Z}_q^{n\times m'}$ be the gadget matrix where $\mathbf{g}^\intercal = [1, 2, \ldots, 2^{\lfloor \log q \rfloor}]$ and $m' = n(\lfloor \log q \rfloor + 1)$. The inverse function $\mathbf{G}_n^{-1} : \mathbb{Z}_q^{n\times t} \to \mathbb{Z}_q^{m'\times t}$ expands each entry $x \in \mathbb{Z}_q$ into a column of size $\lfloor \log q \rfloor + 1$ consisting of the bits in the binary representation of x. By construction, for every matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times t}$, it follows that $\mathbf{G}_n \cdot \mathbf{G}_n^{-1}(\mathbf{A}) = \mathbf{A} \bmod q$. When the lattice dimension n is clear, we omit the subscript and simply write $\mathbf{G}$ and $\mathbf{G}^{-1}(\cdot)$ to denote $\mathbf{G}_n$ and $\mathbf{G}_n^{-1}(\cdot)$. We also write $\mathbf{g}^{-1}(\cdot)$ to denote the 1-dimensional operator $\mathbf{G}_1^{-1}(\cdot)$.

Gadget trapdoors. Our constructions will use the gadget trapdoors introduced by Micciancio and Peikert [MP12], which builds on a long sequence of works on constructing lattice trapdoors [Ajt96, GPV08, AP09, ABB10a, ABB10b, CHKP10].

Theorem 2.12 (Gadget Trapdoor [MP12, adapted])**.** *Let n, m, q be lattice parameters. Let $m' = n(\lfloor \log q \rfloor + 1)$. Then there exist efficient algorithms* (TrapGen, SamplePre) *with the following syntax:*

- $\mathsf{TrapGen}(1^n, q, m) \to (\mathbf{A}, \mathbf{R})$*: On input the lattice dimension n, the modulus q, and the number of samples m, the trapdoor-generation algorithm outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ together with a trapdoor $\mathbf{R} \in \mathbb{Z}_q^{m\times m'}$.*
- $\mathsf{SamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{v}, s) \to \mathbf{u}$*: On input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a trapdoor $\mathbf{R} \in \mathbb{Z}_q^{m\times m'}$, a target vector $\mathbf{v} \in \mathbb{Z}_q^n$, and a Gaussian width parameter s, the preimage-sampling algorithm outputs a vector $\mathbf{u} \in \mathbb{Z}_q^m$.*

Moreover, for all $m \geq O(n \log q)$, the above algorithms satisfy the following properties:

- **Trapdoor distribution:** *The matrix* $\mathbf{A}$ *output by* $\mathsf{TrapGen}(1^n, q, m)$ *is statistically close to uniform. Specifically, if* $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ *and* $\mathbf{A}' \xleftarrow{R} \mathbb{Z}_q^{n \times m}$*, then* $\Delta(\mathbf{A}, \mathbf{A}') \leq 2^{-n}$*. Moreover,* $\mathbf{AR} = \mathbf{G}$ *and* $\|\mathbf{R}\| = 1$*.*

- **Preimage sampling:** *For all matrices* $\mathbf{R} \in \mathbb{Z}_q^{m \times m'}$*, parameters* $s > 0$*, and all target vectors* $\mathbf{v} \in \mathbb{Z}_q^n$ *in the column span of* $\mathbf{A}$*, the output* $\mathbf{u} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{v}, s)$ *of* $\mathsf{SamplePre}$ *satisfies* $\mathbf{Au} = \mathbf{v}$*.*

- **Preimage distribution:** *Suppose* $\mathbf{R}$ *is a gadget trapdoor for* $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ *(i.e.,* $\mathbf{AR} = \mathbf{G}$*). Then, for all* $s \geq \sqrt{mm'}\,\|\mathbf{R}\| \cdot \omega(\sqrt{\log n})$*, and all target vectors* $\mathbf{v} \in \mathbb{Z}_q^n$*, the statistical distance between the following distributions is at most* 2^{-n}*:*

$$\{\mathbf{u} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{v}, s)\} \quad \text{and} \quad \{\mathbf{u} \leftarrow \mathbf{A}_s^{-1}(\mathbf{v})\}.$$

  More generally, the above properties hold if $\mathbf{AR} = \mathbf{HG}$ *for some* invertible *matrix* $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$*. In this case, we refer to* $\mathbf{H}$ *as the "tag."*

For a matrix $\mathbf{V} \in \mathbb{Z}_q^{n \times \ell}$*, we define* $\mathsf{SamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{V}, s)$ *to be the algorithm that outputs the matrix where the* i^{th} *column is* $\mathsf{SamplePre}(\mathbf{A}, \mathbf{R}, \mathbf{v}_i, s)$ *and* $\mathbf{v}_i$ *denotes the* i^{th} *column of* $\mathbf{V}$*.*

Ajtai trapdoors. When analyzing our new hardness assumptions, it will also be convenient to use the more traditional lattice trapdoors introduced by Ajtai [Ajt96] and subsequently expanded by a number of subsequent works [GPV08, AP09, ABB10b, ABB10a, CHKP10, LW15]. We recall the main property we need:

Definition 2.13 (Ajtai Trapdoor [Ajt96])**.** Let n, m, q be lattice parameters and $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ be a matrix. We say that a matrix $\mathbf{R} \in \mathbb{Z}^{m \times m}$ is an Ajtai-trapdoor[9] for $\mathbf{A}$ if $\mathbf{AR} = \mathbf{0} \bmod q$ and $\mathbf{R}$ is full rank over $\mathbb{R}$. We write $\tilde{\mathbf{R}} \in \mathbb{R}^{m \times m}$ to denote the Gram-Schmidt orthogonalization of the columns of $\mathbf{R}$ (from left to right).

Theorem 2.14 (Preimage Sampling [GPV08])**.** *There exists an efficient algorithm* $\mathsf{SamplePre}'(\mathbf{A}, \mathbf{R}, \mathbf{v}, s)$ *that takes as input a matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$*, an Ajtai-trapdoor* $\mathbf{R} \in \mathbb{Z}^{m \times m}$ *for* $\mathbf{A}$*, a target vector* $\mathbf{v} \in \mathbb{Z}_q^n$ *in the column-span of* $\mathbf{A}$*, and a Gaussian width parameter* s*, and outputs a vector* $\mathbf{u} \in \mathbb{Z}_q^m$ *such that* $\mathbf{Au} = \mathbf{v}$*. Moreover, if* $s \geq \|\tilde{\mathbf{R}}\| \cdot \omega\big(\sqrt{m \log m}\big)$*, then the statistical distance between the following distributions is* $\mathsf{negl}(n)$*:*

$$\{\mathbf{u} \leftarrow \mathsf{SamplePre}'(\mathbf{A}, \mathbf{R}, \mathbf{v}, s)\} \quad \text{and} \quad \{\mathbf{u} \leftarrow \mathbf{A}_s^{-1}(\mathbf{v})\}$$

Homomorphic evaluation. Our construction of succinct functional commitments will rely on the lattice homomorphic evaluation procedure developed in [GSW13, BGG+14]. Our presentation is adapted from that in [BV15, BCTW16, BTVW17].

Theorem 2.15 (Homomorphic Encodings [GSW13, BGG+14])**.** *Let* λ *be a security parameter and* $n = n(\lambda)$*,* $m = m(\lambda)$*,* $q = q(\lambda)$ *be lattice parameters. Let* $m' = n(\lfloor \log q \rfloor + 1)$*. Let* $\ell = \ell(\lambda)$ *be an input length. Let* $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ *be a family of functions* $f : \{0,1\}^\ell \to \{0,1\}$ *that can be computed by a Boolean circuit of depth at most* $d = d(\lambda)$*. Then, there exist a pair of efficient algorithms* $(\mathsf{EvalF}, \mathsf{EvalFX})$ *with the following properties:*

- $\mathsf{EvalF}(\mathbf{A}, f) \to \mathbf{A}_f$*: On input a matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times \ell m'}$ *and a function* $f \in \mathcal{F}$*, the input-independent evaluation algorithm outputs a matrix* $\mathbf{A}_f \in \mathbb{Z}_q^{n \times m'}$*.*

- $\mathsf{EvalFX}(\mathbf{A}, f, \mathbf{x}) \to \mathbf{H}_{\mathbf{A},f,\mathbf{x}}$*: On input a matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times \ell m'}$*, a function* $f \in \mathcal{F}$*, and an input* $\mathbf{x} \in \{0,1\}^\ell$*, the input-dependent evaluation algorithm outputs a matrix* $\mathbf{H}_{\mathbf{A},f,\mathbf{x}} \in \mathbb{Z}_q^{\ell m' \times m'}$*.*

Moreover for all security parameters $\lambda \in \mathbb{N}$*, matrices* $\mathbf{A} \in \mathbb{Z}_q^{n \times \ell m'}$*, all functions* $f \in \mathcal{F}$*, and all inputs* $\mathbf{x} \in \{0,1\}^\ell$*, the matrices* $\mathbf{A}_f \leftarrow \mathsf{EvalF}(\mathbf{A}, f)$ *and* $\mathbf{H}_{\mathbf{A},f,\mathbf{x}} \leftarrow \mathsf{EvalFX}(\mathbf{A}, f, \mathbf{x})$ *satisfy the following properties:*

- $\|\mathbf{H}_{\mathbf{A},f,\mathbf{x}}\| \leq (n \log q)^{O(d)}$*.*

- $(\mathbf{A} - \mathbf{x}^\intercal \otimes \mathbf{G}) \cdot \mathbf{H}_{\mathbf{A},f,\mathbf{x}} = \mathbf{A}_f - f(\mathbf{x}) \cdot \mathbf{G}$*.*

[9] Technically, $\mathbf{R}$ is a trapdoor for the lattice $\Lambda^\perp(\mathbf{A})$, but for simplicity of description, we simply refer to $\mathbf{R}$ as a trapdoor for the matrix $\mathbf{A}$.

3 Vector Commitments with Private Opening from SIS

In this section, we show how to construct a vector commitment with private openings from the standard SIS assumption. We start by recalling the definition of a vector commitment:

Definition 3.1 (Vector Commitment)**.** A vector commitment scheme with succinct local openings over a message space $\mathcal{M}$ consists of a tuple of efficient algorithms $\Pi_{\mathsf{VC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ with the following properties:

- $\mathsf{Setup}(1^\lambda, 1^\ell) \to \mathsf{crs}$: On input the security parameter λ and the vector length ℓ, the setup algorithm outputs a common reference string crs.
- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x}) \to (\sigma, \mathsf{st})$: On input the common reference string crs and a vector $\mathbf{x} \in \mathcal{M}^\ell$, the commit algorithm outputs a commitment σ and a state st.
- $\mathsf{Open}(\mathsf{st}, i) \to \pi$: On input a commitment state st and an index $i \in [\ell]$, the open algorithm outputs an opening π.
- $\mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i, \pi) \to \{0,1\}$: On input the common reference string crs, a commitment σ, an index i, a message $x_i \in \mathcal{M}$, and an opening π, the verification algorithm outputs a bit $b \in \{0,1\}$.

We now define several standard properties on vector commitment schemes:

- **Correctness:** For all security parameters λ, vector dimensions ℓ, and inputs $\mathbf{x} = (x_1, \ldots, x_\ell) \in \mathcal{M}^\ell$,

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, \sigma, i, m_i, \pi) = 1 : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell); \\ (\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x}); \\ \pi \leftarrow \mathsf{Open}(\mathsf{st}, i) \end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

- **Succinctness:** The vector commitment scheme is succinct if there exists a universal polynomial $\mathsf{poly}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\sigma| = \mathsf{poly}(\lambda, \log \ell)$ and $|\pi| = \mathsf{poly}(\lambda, \log \ell)$ in the correctness definition.
- **Binding:** We say the commitment scheme is statistically binding (resp., computationally binding) if for all polynomials $\ell = \ell(\lambda)$ and all adversaries $\mathcal{A}$ (resp., efficient adversaries $\mathcal{A}$),

$$\Pr\left[\begin{array}{c} \mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i, \pi) = 1 \\ \text{and } x_i \neq x_i' \text{ and} \\ \mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i', \pi') = 1 \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell); \\ \left(\sigma, i, (x_i, \pi), (x_i', \pi')\right) \leftarrow \mathcal{A}(1^\lambda, 1^\ell, \mathsf{crs}) \end{array}\right] = \mathsf{negl}(\lambda).$$

- **Private openings:** For a vector dimension ℓ, an adversary $\mathcal{A}$, and a simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$, we define two distributions $\mathsf{Real}_{\mathcal{A}}(1^\lambda, 1^\ell)$ and $\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda, 1^\ell)$ as follows:

$\mathsf{Real}_{\mathcal{A}}(1^\lambda)$:	$\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda)$:
1. Give $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ to $\mathcal{A}$. 2. Algorithm $\mathcal{A}$ outputs an input $\mathbf{x} \in \mathcal{M}^\ell$. 3. Compute $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ and give σ to $\mathcal{A}$. 4. Algorithm $\mathcal{A}$ outputs an index $i \in [\ell]$. 5. Give $\pi_i \leftarrow \mathsf{Open}(\mathsf{st}, i)$ to $\mathcal{A}$. 6. Algorithm $\mathcal{A}$ outputs a bit $b \in \{0,1\}$ which is the output of the experiment.	1. Sample $(\mathsf{crs}, \sigma, \mathsf{st}) \leftarrow \mathcal{S}_0(1^\lambda, 1^\ell)$ and give crs to $\mathcal{A}$. 2. Algorithm $\mathcal{A}$ outputs an input $\mathbf{x} \in \mathcal{M}^\ell$. 3. Give σ to $\mathcal{A}$. 4. Algorithm $\mathcal{A}$ outputs an index $i \in [\ell]$. 5. Compute $\pi_i \leftarrow \mathcal{S}_1(\mathsf{st}, i, x_i)$ and give π_i to $\mathcal{A}$. 6. Algorithm $\mathcal{A}$ outputs a bit $b \in \{0,1\}$ which is the output of the experiment.

  We say that the vector commitment scheme has statistically (resp., computationally) private openings if for all polynomials $\ell = \ell(\lambda)$ and adversaries $\mathcal{A}$ (resp., efficient adversaries $\mathcal{A}$), there exists an efficient simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ such that $\mathsf{Real}_{\mathcal{A}}(1^\lambda, 1^\ell)$ and $\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda, 1^\ell)$ are statistically (resp., computationally) indistinguishable.

3.1 The Basis-Augmented SIS (BASIS) Assumption

In this section, we introduce the family of SIS assumptions that we use to build our vector commitment schemes. At a high level, our assumptions assert that the SIS problem is hard with respect to a random matrix $\mathbf{A}$ even given a trapdoor for a matrix $\mathbf{B}$ that is correlated with $\mathbf{A}$. We refer to our family of assumptions as the "basis-augmented SIS" (BASIS) assumption. As we discuss below (Theorem 3.4), some versions of the BASIS assumption can be reduced to the standard SIS assumption. For instance, our first construction of a vector commitments with *private* openings (Construction 3.9) relies on a version that reduces to the standard SIS assumption.

Assumption 3.2 (BASIS Assumption)**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, and $\beta = \beta(\lambda)$ be lattice parameters. Let $s = s(\lambda)$ be a Gaussian width parameter. Let Samp be an efficient sampling algorithm that takes as input a security parameter λ and a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and outputs a matrix $\mathbf{B} \in \mathbb{Z}_q^{n' \times m'}$ along with auxiliary input aux. We say that the basis-augmented SIS (BASIS) assumption holds with respect to Samp if for all efficient adversaries $\mathcal{A}$,

$$\Pr\left[\mathbf{A}\mathbf{x} = \mathbf{0} \text{ and } 0 < \|\mathbf{x}\| \leq \beta : \begin{array}{c} \mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}; \\ (\mathbf{B}, \text{aux}) \leftarrow \text{Samp}(1^\lambda, \mathbf{A}), \mathbf{T} \leftarrow \mathbf{B}_s^{-1}(\mathbf{G}_{n'}); \\ \mathbf{x} \leftarrow \mathcal{A}(1^\lambda, \mathbf{A}, \mathbf{B}, \mathbf{T}, \text{aux}) \end{array}\right] = \text{negl}(\lambda).$$

In other words, we require that SIS is hard with respect to $\mathbf{A}$ even given a trapdoor $\mathbf{T}$ for the related matrix $\mathbf{B}$.

Assumption 3.3 (BASIS Assumption Instantiations)**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, and $\beta = \beta(\lambda)$ be lattice parameters. Let $s = s(\lambda)$ be a Gaussian width parameter and $\ell = \ell(\lambda)$ be a dimension. We consider two concrete instantiations of the BASIS assumption:

- $\text{BASIS}_{\text{rand}}$: The sampling algorithm $\text{Samp}(1^\lambda, \mathbf{A})$ samples $i^* \xleftarrow{R} [\ell]$, $\mathbf{A}_i \xleftarrow{R} \mathbb{Z}_q^{(n+1) \times m}$ for all $i \neq i^*$, $\mathbf{a} \xleftarrow{R} \mathbb{Z}_q^m$, sets $\mathbf{A}_{i^*} \leftarrow \left[\begin{smallmatrix} \mathbf{a}^T \\ \mathbf{A} \end{smallmatrix}\right]$, and outputs

  $$\mathbf{B}_\ell = \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & -\mathbf{G}_{n+1} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & -\mathbf{G}_{n+1} \end{array}\right] \quad \text{and} \quad \text{aux} = i^*.$$

  We refer to this assumption as "the BASIS assumption with random matrices."

- $\text{BASIS}_{\text{struct}}$: The sampling algorithm $\text{Samp}(1^\lambda, \mathbf{A})$ samples $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n \times n}$ for all $i \in [\ell]$ and outputs

  $$\mathbf{B}_\ell = \left[\begin{array}{ccc|c} \mathbf{W}_1\mathbf{A} & & & -\mathbf{G}_n \\ & \ddots & & \vdots \\ & & \mathbf{W}_\ell\mathbf{A} & -\mathbf{G}_n \end{array}\right] \quad \text{and} \quad \text{aux} = (\mathbf{W}_1, \ldots, \mathbf{W}_\ell). \tag{3.1}$$

  This is essentially $\text{BASIS}_{\text{rand}}$ with *structured* matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$. We refer to this assumption as "the BASIS assumption with structured matrices."

Each of the above assumptions is parameterized by the tuple of parameters $(n, m, q, \beta, s, \ell)$. Strictly speaking, in both cases above, the auxiliary information aux can be efficiently computed directly from $\mathbf{A}$ and $\mathbf{B}_\ell$, and thus, can be safely omitted. We include them here for notational convenience.

Hardness of $\text{BASIS}_{\text{rand}}$**.** We start by showing that the BASIS assumption with random matrices (i.e., $\text{BASIS}_{\text{rand}}$ in Assumption 3.3) holds under the *standard* SIS assumption.

Theorem 3.4 (Hardness of $\text{BASIS}_{\text{rand}}$)**.** *Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, $q = q(\lambda)$, $\beta = \beta(\lambda)$ be lattice parameters. Take any polynomial $\ell = \ell(\lambda)$ and suppose $n \geq \lambda$, $m \geq O(n \log q)$, and $s \geq O(\ell m \log(n\ell))$. Then, under the $\text{SIS}_{n,m,q,\beta}$ assumption, the $\text{BASIS}_{\text{rand}}$ assumption with parameters $(n, m, q, \beta, s, \ell)$ holds.*

Proof. Take any polynomial $\ell = \ell(\lambda)$. We proceed via a hybrid argument:

- Hyb_0: This is the $\mathsf{BASIS}_{\mathsf{rand}}$ security game:
  - The challenger samples $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ and $(\mathbf{B}_\ell, \mathsf{aux}) \leftarrow \mathsf{Samp}(1^\lambda, \mathbf{A})$. In this case, the sampling algorithm samples $i^* \xleftarrow{R} [\ell]$. It then samples $\mathbf{a} \xleftarrow{R} \mathbb{Z}_q^m$, $\mathbf{A}_{i^*} \leftarrow \left[\begin{smallmatrix} \mathbf{a}^\intercal \\ \mathbf{A} \end{smallmatrix}\right]$, and $\mathbf{A}_i \xleftarrow{R} \mathbb{Z}_q^{(n+1) \times m}$ for each $i \neq i^*$.
  - The challenger gives $\mathbf{A}$, $\mathbf{B}_\ell = [\mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid -\mathbf{1}^\ell \otimes \mathbf{G}_{n+1}] \in \mathbb{Z}_q^{\ell(n+1) \times \ell m + m'}$, $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_s^{-1}(\mathbf{G}_{(n+1)\ell})$ and $\mathsf{aux} = i^*$ to $\mathcal{A}$, where $m' = (n+1)(\lfloor \log q \rfloor + 1)$
  - The output of the experiment is 1 if $0 < \|\mathbf{x}\| \leq \beta$ and $\mathbf{A}\mathbf{x} = \mathbf{0}$.
- Hyb_1: Same as Hyb_0 except the challenger samples $(\mathbf{A}_i, \mathbf{R}_i) \leftarrow \mathsf{TrapGen}(1^{n+1}, q, m)$ for each $i \neq i^*$.
- Hyb_2: Same as Hyb_1 except after constructing $\mathbf{B}_\ell$, the challenger constructs a matrix $\mathbf{R} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$ as follows:

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_1 & & & -\mathbf{R}_1 & & & \\ & \ddots & & \vdots & & & \\ & & \mathbf{R}_{i^*-1} & -\mathbf{R}_{i^*-1} & & & \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ & & & -\mathbf{R}_{i^*+1} & \mathbf{R}_{i^*+1} & & \\ & & & \vdots & & \ddots & \\ & & & -\mathbf{R}_\ell & & & \mathbf{R}_\ell \\ \mathbf{0} & \cdots & \mathbf{0} & -\mathbf{I}_{m'} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}. \tag{3.2}$$

  It then samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{R}, \mathbf{G}_{(n+1)\ell}, s)$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output distribution of an execution of experiment Hyb_i with adversary $\mathcal{A}$. We now analyze each pair of adjacent hybrids.

Lemma 3.5. *Suppose $n \geq \lambda$ and $m \geq O(n \log q)$. Then, for all adversaries $\mathcal{A}$, it holds that $\mathsf{Hyb}_0(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A})$.*

Proof. The only difference between Hyb_0 and Hyb_1 is the distribution of $\mathbf{A}_i$ for $i \neq i^*$. In Hyb_0, $\mathbf{A}_i \xleftarrow{R} \mathbb{Z}_q^{(n+1) \times m}$ while in Hyb_1, it is sampled using $\mathsf{TrapGen}(1^{n+1}, q, m)$. By Theorem 2.12, these two distributions are statistically indistinguishable when $m \geq O((n+1)\log q) = O(n \log q)$. Since $\ell = \mathsf{poly}(\lambda)$, the lemma follows by a hybrid argument. □

Lemma 3.6. *Suppose $n \geq \lambda$, $m \geq O(n \log q)$, and $s \geq O(\ell m \log(n\ell))$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_1(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

Proof. The only difference between Hyb_1 and Hyb_2 is the distribution of the trapdoor $\mathbf{T}$. Consider the distribution in Hyb_2. By Theorem 2.12, we have that $\mathbf{A}_i \mathbf{R}_i = \mathbf{G}_{n+1} \in \mathbb{Z}_q^{(n+1) \times m'}$ for all $i \neq i^*$ and moreover $\|\mathbf{R}_i\| = 1$. Thus, by construction, $\mathbf{B}_\ell \mathbf{R} = \mathbf{G}_{\ell(n+1)}$, and $\|\mathbf{R}\| = 1$. Suppose $s \geq \sqrt{(\ell m + m')\ell m'}\, \|\mathbf{R}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell m \log(n\ell))$ for $m \geq m' = O(n \log q)$. Again by Theorem 2.12, the distribution of $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{R}, \mathbf{G}_{(n+1)\ell}, s)$ is statistically close $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_s^{-1}(\mathbf{G}_{(n+1)\ell})$, which is the distribution in Hyb_1. □

Lemma 3.7. *Under the $\mathsf{SIS}_{n,m,q,\beta}$ assumption, for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists an efficient adversary $\mathcal{A}$ where $\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] = \varepsilon$ for some non-negligible ε. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the SIS assumption:

1. At the beginning of the game, algorithm $\mathcal{B}$ obtains a SIS challenge $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$.
2. Algorithm $\mathcal{B}$ samples an index $i^* \xleftarrow{R} [\ell]$. It also samples a row $\mathbf{a} \xleftarrow{R} \mathbb{Z}_q^m$ and sets $\mathbf{A}_{i^*} \leftarrow \left[\begin{smallmatrix} \mathbf{a}^\intercal \\ \mathbf{A} \end{smallmatrix}\right]$. For all $i \neq i^*$, it samples $(\mathbf{A}_i, \mathbf{R}_i) \leftarrow \mathsf{TrapGen}(1^{n+1}, m, q)$. It constructs the matrix $\mathbf{B}_\ell = [\mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid -\mathbf{1}^\ell \otimes \mathbf{G}_{n+1}]$, and the trapdoor $\mathbf{R}$ according to Eq. (3.2). Finally, algorithm $\mathcal{B}$ computes $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{R}, \mathbf{G}_{(n+1)\ell}, s)$ and gives $\mathbf{A}$, $\mathbf{B}_\ell$, $\mathbf{T}$, and $\mathsf{aux} = i^*$ to $\mathcal{A}$.

3. After algorithm $\mathcal{A}$ outputs a vector $\mathbf{x} \in \mathbb{Z}_q^m$, algorithm $\mathcal{B}$ outputs the same vector $\mathbf{x}$.

By construction, algorithm $\mathcal{B}$ perfectly simulates an execution of Hyb_2 for algorithm $\mathcal{A}$, so with probability ε, algorithm $\mathcal{A}$ outputs $\mathbf{x}$ where $\mathbf{A}\mathbf{x} = \mathbf{0}$ and $0 < \|\mathbf{x}\| \leq \beta$. Thus is a valid SIS solution, so algorithm $\mathcal{B}$ breaks SIS with the same advantage ε. □

Combining Lemmas 3.5 to 3.7, we have that for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$. □

Hardness of $\mathsf{BASIS}_{\mathsf{struct}}$. While we are able to reduce the hardness of the BASIS assumption with random matrices ($\mathsf{BASIS}_{\mathsf{rand}}$) to the standard SIS assumption, we do not know of an analogous reduction for the BASIS assumption with structured matrices ($\mathsf{BASIS}_{\mathsf{struct}}$). In Section 6, we compare our $\mathsf{BASIS}_{\mathsf{struct}}$ assumption to the conceptually-similar k-R-SIS assumptions introduced in [ACL+22] for constructing lattice-based vector commitment and preprocessing SNARGs. In addition, we also describe an alternative view of the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption, and suggest a set of candidate parameter instantiations (Remark 6.2).

Remark 3.8 (An Alternative Formulation of $\mathsf{BASIS}_{\mathsf{struct}}$)**.** When the matrices $\mathbf{W}_1, \ldots, \mathbf{W}_\ell$ in the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption are invertible (e.g., this holds with overwhelming probability when q is prime), we can alternatively interpret the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption as saying that SIS is hard with respect to $\mathbf{A}$ given a trapdoor for the matrix

$$\tilde{\mathbf{B}}_\ell = \left[\mathbf{I}_\ell \otimes \mathbf{A} \mid \tilde{\mathbf{W}}\mathbf{G}\right] \quad \text{where} \quad \tilde{\mathbf{W}} = \begin{bmatrix} \tilde{\mathbf{W}}_1 \\ \vdots \\ \tilde{\mathbf{W}}_\ell \end{bmatrix} \in \mathbb{Z}_q^{n\ell \times n} \quad \text{and} \quad \tilde{\mathbf{W}}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times n}.$$

When $\mathbf{W}_1, \ldots, \mathbf{W}_\ell$ are random invertible matrices, we can set $\tilde{\mathbf{W}}_i := -\mathbf{W}_i^{-1}$. In this case, we can write $\mathbf{B}_\ell = (\tilde{\mathbf{W}}')^{-1}\tilde{\mathbf{B}}_\ell$, where $\mathbf{B}_\ell$ is the matrix in Eq. (3.1) of the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption and $\tilde{\mathbf{W}}' = \mathrm{diag}(\tilde{\mathbf{W}}_1, \ldots, \tilde{\mathbf{W}}_\ell)$. Then, if $\mathbf{T}$ is a trapdoor for the matrix $\mathbf{B}_\ell$ (i.e., $\mathbf{B}_\ell\mathbf{T} = \mathbf{G}_{n\ell}$), then $\mathbf{T}$ is also a trapdoor for matrix $\tilde{\mathbf{B}}_\ell$ with tag $\tilde{\mathbf{W}}'$ since $\tilde{\mathbf{B}}_\ell\mathbf{T} = \tilde{\mathbf{W}}'\mathbf{B}_\ell\mathbf{T} = \tilde{\mathbf{W}}'\mathbf{G}_{n\ell}$.

3.2 Vector Commitments with Private Opening from SIS

We now show how to construct a vector commitment scheme with statistically private openings from the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption. By Theorem 3.4, we can in turn base hardness on the standard SIS assumption (with polynomial modulus).

Construction 3.9 (Vector Commitments from SIS)**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, and $q = q(\lambda)$ be lattice parameters. Let $m' = n(\lceil \log q \rceil + 1)$ and $B = B(\lambda)$ be a bound. Let $s_0 = s_0(\lambda)$, $s_1 = s_1(\lambda)$ be Gaussian width parameters. Let ℓ be the vector dimension. We construct a vector commitment scheme $\Pi_{\mathsf{VC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ for $\mathbb{Z}_q^\ell$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^\ell)$: On input the security parameter λ and the vector dimension ℓ, the setup algorithm samples $(\mathbf{A}_i, \mathbf{R}_i) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ for each $i \in [\ell]$. Then, it constructs matrices $\mathbf{B}_\ell$ and $\mathbf{R}$ where

$$\mathbf{B}_\ell = \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_\ell & -\mathbf{G} \end{array}\right] \in \mathbb{Z}_q^{n\ell \times (\ell m + m')} \quad \text{and} \quad \mathbf{R} = \left[\begin{array}{c} \mathrm{diag}(\mathbf{R}_1, \ldots, \mathbf{R}_\ell) \\ \hline \mathbf{0}^{m' \times \ell m'} \end{array}\right] \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}. \tag{3.3}$$

  Finally, the setup algorithm samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{R}, \mathbf{G}_{n\ell}, s_0)$ and outputs the common reference string $\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$.

- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x})$: On input the common reference string $\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$ and a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$, the commit algorithm constructs $\mathbf{B}_\ell$ from $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ according to Eq. (3.3) and then uses $\mathbf{T}$ to sample

$$\begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{T}, -\mathbf{x} \otimes \mathbf{e}_1, s_1), \tag{3.4}$$

where $\mathbf{e}_1 = [1, 0, \ldots, 0]^\intercal \in \mathbb{Z}_q^m$ is the first standard basis vector. It computes $\mathbf{c} \leftarrow \mathbf{G}\hat{\mathbf{c}} \in \mathbb{Z}_q^n$ and outputs the commitment $\sigma = \mathbf{c}$ and the state $\mathsf{st} = (\mathbf{v}_1, \ldots, \mathbf{v}_\ell)$.

- $\mathsf{Open}(\mathsf{st}, i)$: On input the state $\mathsf{st} = (\mathbf{v}_1, \ldots, \mathbf{v}_\ell)$ and the index $i \in [\ell]$, the opening algorithm outputs $\mathbf{v}_i$.
- $\mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i, \pi)$: On input the common reference string $\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$, a commitment $\sigma = \mathbf{c} \in \mathbb{Z}_q^n$, an index $i \in [\ell]$, a message $x_i \in \mathbb{Z}_q$, and an opening $\pi = \mathbf{v}_i$, the verification algorithm outputs 1 if

$$\|\mathbf{v}_i\| \le B \quad \text{and} \quad \mathbf{c} = \mathbf{A}_i \mathbf{v}_i + x_i \mathbf{e}_1.$$

Theorem 3.10 (Correctness). *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, q is prime, $s_0 \ge O(\ell m \log(n\ell))$, $s_1 \ge O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$, and $B \ge \sqrt{\ell m + m'} \cdot s_1$. Then Construction 3.9 is correct.*

Proof. Take any polynomial $\ell = \ell(\lambda)$, any vector $\mathbf{x} \in \mathbb{Z}_q^\ell$, and any index $i \in [\ell]$. Let $\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$. Let $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ where $\sigma = \mathbf{c} = \mathbf{G}\hat{\mathbf{c}} \in \mathbb{Z}_q^n$ and $\mathsf{st} = (\mathbf{v}_1, \ldots, \mathbf{v}_\ell)$. Let $\pi = \mathbf{v}_i \leftarrow \mathsf{Open}(\mathsf{st}, i)$, and consider $\mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i, \pi)$:

- Let $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$ and $\mathbf{R} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$ be the matrices from Eq. (3.3). By construction, $\mathbf{B}_\ell \mathbf{R} = \mathbf{G}_{n\ell}$ and $\|\mathbf{R}\| = 1$. Suppose $m \ge m' = O(n \log q)$ and

$$s_0 \ge \sqrt{(\ell m + m')\ell m'}\, \|\mathbf{R}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell m \log(n\ell)).$$

  By Theorem 2.12, this means $\mathbf{B}_\ell \mathbf{T} = \mathbf{G}_{n\ell}$, and moreover, by Corollaries 2.8 and 2.11 and Lemmas 2.2 and 2.9, $\|\mathbf{T}\| \le \sqrt{\ell m + m'} \cdot s_0$ with probability $1 - \mathsf{negl}(n) = 1 - \mathsf{negl}(\lambda)$.
- Suppose $s_1 \ge \sqrt{(\ell m + m')\ell m'}\, \|\mathbf{T}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$. Then, by Theorem 2.12 and by construction of $\mathbf{v}_1, \ldots, \mathbf{v}_\ell, \hat{\mathbf{c}}$ (see Eq. (3.4)), we have that $\mathbf{A}_i \mathbf{v}_i - \mathbf{c} = \mathbf{A}_i \mathbf{v}_i - \mathbf{G}\hat{\mathbf{c}} = -x_i \mathbf{e}_1$. In addition, by Corollaries 2.8 and 2.11 and Lemmas 2.2 and 2.9, we have that $\|\mathbf{v}_i\| \le \sqrt{\ell m + m'} s_1 \le B$, and the verification algorithm accepts with overwhelming probability. □

Theorem 3.11 (Binding). *Take any polynomial $\ell = \ell(\lambda)$ and suppose $n \ge \lambda$, $m \ge O(n \log q)$, and $s_0 \ge O(\ell m \log(n\ell))$. Then, under the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption with parameters $(n-1, m, q, 2B, s_0, \ell)$. Construction 3.9 is computationally binding.*

Proof. Take any polynomial $\ell = \ell(\lambda)$. We now define a sequence of hybrid experiments.

- Hyb_0: This is the real binding experiment:
  - The challenger starts by sampling $(\mathbf{A}_i, \mathbf{R}_i) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ for each $i \in [\ell]$. It then constructs $\mathbf{R}$ according to Eq. (3.3) and sets $\mathbf{B}_\ell = [\mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid -\mathbf{1}^\ell \otimes \mathbf{G}]$. It samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{R}, \mathbf{G}_{n\ell}, s_0)$. Then, the challenger gives $\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$ to $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ then outputs a commitment $\mathbf{c} \in \mathbb{Z}_q^n$, an index $i \in [\ell]$, and openings $(x, \mathbf{v})$, $(x', \mathbf{v}')$.
  - The output of the experiment is 1 if $x \ne x'$ and

$$\|\mathbf{v}\| \le B \quad \text{and} \quad \|\mathbf{v}'\| \le B \quad \text{and} \quad \mathbf{c} = \mathbf{A}_i \mathbf{v} + x \mathbf{e}_1 \quad \text{and} \quad \mathbf{c} = \mathbf{A}_i \mathbf{v}' + x' \mathbf{e}_1.$$

- Hyb_1: Same as Hyb_0 except at the beginning of the game, the challenger samples an index $i^* \xleftarrow{R} [\ell]$. The output of the experiment is 1 if the conditions in Hyb_0 hold *and* $i = i^*$.
- Hyb_2: Same as Hyb_1 except the challenger samples $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_{s_0}^{-1}(\mathbf{G}_{n\ell})$.
- Hyb_3: Same as Hyb_2 except the challenger samples $\mathbf{A}_i \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ for all $i \in [\ell]$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output distribution of an execution of experiment Hyb_i with adversary $\mathcal{A}$. We now analyze each pair of adjacent hybrids.

Lemma 3.12. *For all adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] = \ell \cdot \Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1]$.*

Proof. This follows by definition. The index i^* is sampled independently of the adversary's view, so $\Pr[i^* = i] = 1/\ell$, and the claim follows. □

Lemma 3.13. *Suppose $n \geq \lambda$, $m \geq O(n \log q)$, $s_0 \geq O(\ell m \log(n\ell))$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_1(\mathcal{A}) \stackrel{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

Proof. The only difference between Hyb_1 and Hyb_2 is the distribution of $\mathbf{T}$. Consider the distribution of $\mathbf{T}$ in Hyb_1. Here, $\mathbf{B}_\ell \mathbf{R} = \mathbf{G}_{n\ell}$ by construction and $\|\mathbf{R}\| = 1$ since $\|\mathbf{R}_i\| = 1$ for all $i \in [\ell]$ by Theorem 2.12. Suppose $m \geq m' = O(n \log q)$ and $s_0 \geq \sqrt{(\ell m + m')\ell m'}\,\|\mathbf{R}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell m \log(n\ell))$. Again by Theorem 2.12, the distribution of $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{R}, \mathbf{G}_{n\ell}, s_0)$ is statistically close to the distribution of $(\mathbf{B}_\ell)_{s_0}^{-1}(\mathbf{G}_{n\ell})$. This is the distribution of $\mathbf{T}$ in Hyb_2 and the claim follows. □

Lemma 3.14. *Suppose $n \geq \lambda$ and $m \geq O(n \log q)$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_2(\mathcal{A}) \stackrel{s}{\approx} \mathsf{Hyb}_3(\mathcal{A})$.*

Proof. The only difference between Hyb_2 and Hyb_3 is the distribution of $\mathbf{A}_i$. In Hyb_2, the challenger samples $(\mathbf{A}_i, \mathbf{R}_i) \stackrel{\mathrm{R}}{\leftarrow} \mathsf{TrapGen}(1^n, q, m)$. By Theorem 2.12, for sufficiently large $m \geq O(n \log q)$, the distribution of $\mathbf{A}_i$ is statistically close to uniform over $\mathbb{Z}_q^{n \times m}$. This coincides with the distribution of $\mathbf{A}_i$ in Hyb_3 and the claim follows by a hybrid argument since $\ell = \mathsf{poly}(\lambda)$. □

Lemma 3.15. *Under the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption with parameters $(n-1, m, q, 2B, s_0, \ell)$, for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists an efficient adversary $\mathcal{A}$ where $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \varepsilon$ for some non-negligible ε. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption:

1. At the beginning of the game, algorithm $\mathcal{B}$ obtains the challenge $\mathbf{A} \in \mathbb{Z}_q^{(n-1)\times m}$, $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$, $\mathbf{T} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$ and $\mathsf{aux} = i^*$.
2. Algorithm $\mathcal{B}$ parses $\mathbf{B}_\ell = [\mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid -\mathbf{1}^\ell \otimes \mathbf{G}]$ and defines $\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$. Algorithm $\mathcal{B}$ gives $\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$ to algorithm $\mathcal{A}$.
3. After algorithm $\mathcal{A}$ outputs a commitment $\hat{\mathbf{c}} \in \mathbb{Z}_q^n$, the index $i \in [\ell]$, and openings $(x, \mathbf{v})$, $(x', \mathbf{v}')$, algorithm $\mathcal{B}$ first checks that $i = i^*$. If so, then it outputs $\mathbf{v} - \mathbf{v}'$. Otherwise, it outputs $\perp$.

By construction, algorithm $\mathcal{B}$ perfectly simulates an execution of Hyb_3 for algorithm $\mathcal{A}$, so with probability ε, algorithm $\mathcal{A}$ outputs $\mathbf{c}, i, (x, \mathbf{v}), (x', \mathbf{v}')$ where

$$i = i^* \quad \text{and} \quad x \neq x' \quad \text{and} \quad \|\mathbf{v}\| \leq B \quad \text{and} \quad \|\mathbf{v}'\| \leq B \quad \text{and} \quad \mathbf{c} = \mathbf{A}_{i^*}\mathbf{v} + x\mathbf{e}_1 \quad \text{and} \quad \mathbf{c} = \mathbf{A}_{i^*}\mathbf{v}' + x'\mathbf{e}_1.$$

This means that $\|\mathbf{v} - \mathbf{v}'\| \leq 2B$, and moreover, $\mathbf{A}_{i^*}(\mathbf{v} - \mathbf{v}') = (x' - x)\mathbf{e}_1$. Since $x \neq x'$, this means that $\mathbf{v} - \mathbf{v}' \neq \mathbf{0}$. Next, by definition of $\mathbf{A}_{i^*}$ in the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption, we have

$$\mathbf{A}_{i^*}(\mathbf{v} - \mathbf{v}') = \begin{bmatrix} \mathbf{a}^\intercal \\ \mathbf{A} \end{bmatrix} (\mathbf{v} - \mathbf{v}') = \begin{bmatrix} x' - x \\ \mathbf{0}^{n-1} \end{bmatrix}.$$

Thus, $\mathbf{A}(\mathbf{v} - \mathbf{v}') = \mathbf{0}$ and $\mathbf{v} - \mathbf{v}'$ is a valid solution to the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption. Correspondingly, algorithm $\mathcal{B}$ breaks the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption with advantage ε. □

By Lemmas 3.12 to 3.14, we have that for all adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] \leq \ell \cdot (\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] + \mathsf{negl}(\lambda))$. Since $\ell = \mathsf{poly}(\lambda)$, we can conclude via Lemma 3.15 that for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] \leq \mathsf{negl}(\lambda)$. □

Theorem 3.16 (Hiding)**.** *Suppose $n \geq \lambda$, $m \geq O(n \log q)$, q is prime, $s_0 \geq O(\ell m \log(n\ell))$, and $s_1 \geq O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$. Then, Construction 3.9 has statistically private openings.*

Proof. We construct an efficient simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ as follows:

- $\mathcal{S}_0(1^\lambda, 1^\ell)$: On input the security parameter λ and the vector dimension ℓ, the simulator samples $(\mathbf{A}_i, \mathbf{R}_i) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ for each $i \in [\ell]$. Let $\bar{\mathbf{A}} = \mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell)$. Then, construct the matrix $\mathbf{B}_\ell = [\bar{\mathbf{A}} \mid -\mathbf{1}^\ell \otimes \mathbf{G}]$ along with the trapdoor $\mathbf{R}$ according to Eq. (3.3). It then computes $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{R}, \mathbf{G}_{n\ell}, s_0)$ and sets the common reference string to be $\mathsf{crs} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$. Next, it samples $\hat{\mathbf{c}} \leftarrow D^m_{\mathbb{Z}, s_1}$ and outputs crs, $\sigma = \mathbf{c} = \mathbf{G}\hat{\mathbf{c}}$, and $\mathsf{st}_{\mathcal{S}} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{R}_1, \ldots, \mathbf{R}_\ell, \mathbf{c})$.
- $\mathcal{S}_1(\mathsf{st}_{\mathcal{S}}, i, x_i)$: On input the simulation state $\mathsf{st}_{\mathcal{S}} = (\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{R}_1, \ldots, \mathbf{R}_\ell, \mathbf{c})$, an index $i \in [\ell]$, and a value $x_i \in \mathbb{Z}_q$, the simulator samples $\mathbf{v}_i \leftarrow \mathsf{SamplePre}(\mathbf{A}_i, \mathbf{R}_i, -x_i\mathbf{e}_1 + \mathbf{c}, s_1)$. It outputs the opening $\pi_i = \mathbf{v}_i$.

To complete the proof, it suffices to argue that the real distribution and the simulated distributions are statistically indistinguishable. First, the simulator samples crs using the same procedure as Setup, so it suffices to analyze the commitment and the opening. Take any vector $\mathbf{x} \in \mathbb{Z}_q^m$ and let $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ where $\sigma = \mathbf{c} \in \mathbb{Z}_q^n$ is the commitment and $\mathsf{st} = (\mathbf{v}_1, \ldots, \mathbf{v}_\ell)$ are the openings. We consider their joint distribution in the real distribution:

- Let $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$ be the matrix from Eq. (3.3). Let $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{R}, \mathbf{G}_{n\ell}, s_0)$ be the trapdoor sampled in the real scheme. As in the proof of Theorem 3.10, we can appeal to Theorem 2.12 to conclude that with overwhelming probability, $\mathbf{B}_\ell\mathbf{T} = \mathbf{G}_{n\ell}$ and $\|\mathbf{T}\| \le \sqrt{\ell m + m'}\, s_0$.
- In the real scheme, the vector $\hat{\mathbf{c}}$ and the openings $\mathbf{v}_1, \ldots, \mathbf{v}_\ell$ are sampled using $\mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{T}, \mathbf{x} \otimes \mathbf{e}_1, s_1)$ according to Eq. (3.4). Suppose $m \ge m' = O(n \log q)$ and

$$s_1 \ge \sqrt{(\ell m + m')\ell m'}\, \|\mathbf{T}\| \cdot \omega(\sqrt{\log n}) = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0).$$

  By Theorem 2.12, the joint distribution of $(\mathbf{v}_1, \ldots, \mathbf{v}_\ell, \hat{\mathbf{c}})$ is statistically close to $(\mathbf{B}_\ell)^{-1}_{s_1}(\mathbf{x} \otimes \mathbf{e}_1)$.
- Let $\bar{\mathbf{A}} = \mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell)$. Then, $\mathbf{B}_\ell = [\bar{\mathbf{A}} \mid -\mathbf{1}^\ell \otimes \mathbf{G}]$. With overwhelming probability over the choice of $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$, the columns of $\bar{\mathbf{A}}$ generate $\mathbb{Z}_q^{n\ell}$ and $\lambda_1^\infty(\Lambda(\mathbf{A}_i)) \ge q/4$ by Lemma 2.2 and Corollary 2.5. Since $s_1 \ge 4\log(\ell m')$, by Corollary 2.11, the distribution of $(\mathbf{v}_1, \ldots, \mathbf{v}_\ell, \hat{\mathbf{c}}) \leftarrow (\mathbf{B}_\ell)^{-1}_{s_1}(\mathbf{x} \otimes \mathbf{e})$ is statistically close to the distribution

$$\left\{\hat{\mathbf{c}} \leftarrow D^{m'}_{\mathbb{Z}, s_1},\ (\mathbf{v}_1, \ldots, \mathbf{v}_\ell) \leftarrow \bar{\mathbf{A}}^{-1}_{s_1}\big(-(\mathbf{x} \otimes \mathbf{e}_1) + (\mathbf{1}^\ell \otimes \mathbf{G}\hat{\mathbf{c}})\big)\right\}.$$

- Since $\bar{\mathbf{A}} = \mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell)$, this means that each $\mathbf{v}_i$ is distributed according to $(\mathbf{A}_i)^{-1}_{s_1}(-x_i\mathbf{e}_1 + \mathbf{G}\hat{\mathbf{c}})$. By Theorem 2.12, this is statistically close to the distribution of $\mathsf{SamplePre}(\mathbf{A}_i, \mathbf{R}_i, -x_i\mathbf{e}_1 + \mathbf{G}\hat{\mathbf{c}}, s_1)$ (since $s_1 \ge \sqrt{mm'}\, \|\mathbf{R}\| \cdot \omega(\sqrt{\log n}) = O(m \log n)$), which precisely coincides with the simulated distribution. □

Parameter instantiation. Let λ be a security parameter and ℓ be the vector dimension. We can instantiate the lattice parameters in Construction 3.9 to satisfy Theorems 3.10, 3.11, and 3.16:

- We set the lattice dimension $n = \lambda$ and $m = O(n \log q)$.
- We set $s_0 = O(\ell m \log(n\ell))$ and $s_1 = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0) = O(\ell^{5/2} m^{5/2} \log^2(n\ell))$.
- We set the bound $B = \sqrt{\ell m + m'} \cdot s_1 = O(\ell^3 m^3 \log^2(n\ell)) = O(\ell^3 n^3 \log^2(n\ell) \log^3 q)$.
- Finally, we set the modulus $q = B \cdot \mathsf{poly}(n)$ so that the $\mathsf{SIS}_{n-1,m,q,2B}$ assumption holds. By Theorem 3.4, this implies the $\mathsf{BASIS}_{\mathsf{rand}}$ with parameters $(n-1, m, q, 2B, s_0, \ell)$. In this case, $\log q = O(\log\lambda + \log\ell)$.

With this setting of parameters, we obtain a commitment scheme over $\mathbb{Z}_q^\ell$ with the following parameter sizes:

- **Commitment size:** A commitment σ to a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$ consists of a vector $\sigma = \mathbf{c} \in \mathbb{Z}_q^n$, so

$$|\sigma| = O(n \log q) = O(\lambda \cdot (\log\lambda + \log\ell)).$$

- **Opening size:** An opening π to index i consists of a vector $\pi = \mathbf{v}_i \in \mathbb{Z}_q^m$, where $\|\mathbf{v}_i\| \le B$. Thus, the length of $\mathbf{v}_i$ is bounded by $O(m \log B)$, or equivalently, $|\pi| = O(\lambda \cdot (\log^2\lambda + \log^2\ell))$.

- **CRS size:** The CRS consists of $(\mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{T})$, where $\mathbf{A}_i \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{T} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$. Thus, the total size of the CRS is
$$|\mathsf{crs}| = \ell n m \log q + (\ell m + m')(\ell m') \log q = \ell^2 \cdot \mathsf{poly}(\lambda, \log \ell).$$

Thus, Construction 3.9 is a succinct vector commitment scheme, and we obtain the following corollary:

Corollary 3.17 (Vector Commitments with Private Openings from SIS)**.** *Let λ be a security parameter. Then, for all polynomials $\ell = \ell(\lambda)$, under the SIS assumption with a polynomial norm bound $\beta = \mathsf{poly}(\lambda, \ell)$ and a polynomial modulus $q = \mathsf{poly}(\lambda, \ell)$, there exists a vector commitment scheme over $\mathbb{Z}_q^\ell$ that is computationally binding and has statistically private openings. The size of a commitment to a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$ has size $O(\lambda(\log \lambda + \log \ell))$ and the openings have size $O(\lambda(\log^2 \lambda + \log^2 \ell))$. The size of the CRS is $\ell^2 \cdot \mathsf{poly}(\lambda, \log \ell)$.*

Extensions: linear homomorphism and updatability. Similar to the non-private scheme of Peikert et al. [PPS21], our vector commitment scheme is linearly homomorphic and supports stateless updates. We provide more details below:

Remark 3.18 (Linear Homomorphism)**.** Like the vector commitment scheme of Peikert et al. [PPS21], Construction 3.9 has a linear verification procedure. This means that if $\mathbf{c}$ is a commitment to a vector $\mathbf{x}$ (with openings $\mathbf{v}_1, \ldots, \mathbf{v}_\ell$) and $\mathbf{c}'$ is a commitment to a vector $\mathbf{x}'$ (with opening $\mathbf{v}'_1, \ldots, \mathbf{v}'_\ell$), then $\mathbf{c} + \mathbf{c}'$ is a commitment to the vector $\mathbf{x} + \mathbf{x}' \in \mathbb{Z}_q^\ell$ with openings $\mathbf{v}_1 + \mathbf{v}'_1, \ldots, \mathbf{v}_\ell + \mathbf{v}'_\ell$. In particular, observe that for every $i \in [\ell]$,
$$\mathbf{c} + \mathbf{c}' = \mathbf{A}_i(\mathbf{v}_i + \mathbf{v}'_i) + (x_i + x'_i)\mathbf{e}_1.$$
Note though that the norm of the openings $\|\mathbf{v}_i + \mathbf{v}'_i\| \leq \|\mathbf{v}_i\| + \|\mathbf{v}'_i\| \leq 2B$ can now be as high as $2B$ rather than B. More generally, given a collection of t vector commitments $\mathbf{c}_1, \ldots, \mathbf{c}_t$ to vectors $\mathbf{x}_1, \ldots, \mathbf{x}_t$, and a set of *small* coefficients $\alpha_1, \ldots, \alpha_t$, then $\sum_{i \in [t]} \alpha_i \mathbf{c}_i$ is a commitment to $\sum_{i \in [t]} \alpha_i \mathbf{x}_i$, with openings of size at most $\sum_{i \in [t]} |\alpha_i| \cdot B$. By setting the verification bound accordingly, the scheme supports a bounded number of linear operations on committed vectors. Note that we can also set the modulus q and the norm bound B to be super-polynomial in λ (e.g., $2^{\omega(\log \lambda)}$); then, the scheme supports an arbitrary polynomial number of homomorphic operations (with *small* coefficients). Security in this case relies on solving worst-case lattice problems to a *super-polynomial* approximation factor.

Remark 3.19 (Stateless Updates)**.** Let σ be a commitment to a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$ (with openings $\pi_1, \ldots, \pi_\ell$). Suppose we want to update σ to a commitment σ' of a vector $\mathbf{x}' \in \mathbb{Z}_q^\ell$ that differs from $\mathbf{x}$ at just a single index $i \in [\ell]$ (i.e., $x_j = x'_j$ for all $j \neq i$). A vector commitment scheme supports stateless updates [CPSZ18, PPS21] if it is possible to compute the new commitment σ' from σ given only the original commitment σ, the index i, the original value x_i, and the new value x'_i (or even just the difference $x'_i - x_i \in \mathbb{Z}_q$). Notably, the update algorithm does *not* know x_j for $j \neq i$. The update algorithm outputs a new commitment σ' along with a set of opening updates $\delta'_1, \ldots, \delta'_\ell$. For all indices $i \in [\ell]$, it should be possible to compute an opening π'_i to σ' given the original opening π_i and the opening update δ'_i.

We can leverage the linear verification property (Remark 3.18) to support stateless updates. Suppose $\mathbf{c}$ is a vector commitment to a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$ (with openings $\mathbf{v}_1, \ldots, \mathbf{v}_\ell$). To update $\mathbf{c}$ to a commitment on $\mathbf{x}'$, we first compute a commitment $\mathbf{c}'$ to the difference $\mathbf{x}' - \mathbf{x}$. Let $\mathbf{v}'_1, \ldots, \mathbf{v}'_\ell$ be the openings to $x'_1 - x_1, \ldots, x'_\ell - x_\ell$. By linearity, $\mathbf{c} + \mathbf{c}'$ is a commitment to $\mathbf{x} + (\mathbf{x}' - \mathbf{x}) = \mathbf{x}'$ with openings given by $\mathbf{v}_1 + \mathbf{v}'_1, \ldots, \mathbf{v}_\ell + \mathbf{v}'_\ell$. Observe that the update procedure only requires knowledge of the *difference* $\mathbf{x}' - \mathbf{x}$. Since Construction 3.9 supports a bounded number of homomorphic operations, it also supports a bounded number of updates. Namely, to support up to k updates, we set the norm bound on the opening to be kB, and correspondingly, the size of the commitment and the openings scale with $\mathsf{poly}(\log k)$. Similar to Remark 3.18, we can also set the norm bound and the modulus to be super-polynomial in order to support an arbitrary polynomial number of updates.

4 Succinct Functional Commitments for Circuits

In this section, we show how to obtain a succinct functional commitment for general circuits from the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption. We consider schemes where the parameters scale with the *depth* of the Boolean circuit. We start with the formal definition:

Definition 4.1 (Succinct Functional Commitment)**.** Let λ be a security parameter. Let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of functions $f\colon \{0,1\}^\ell \to \{0,1\}$ on inputs of length $\ell = \ell(\lambda)$ and which can be computed by Boolean circuits of depth at most $d = d(\lambda)$. A succinct functional commitment for $\mathcal{F}$ is a tuple of efficient algorithms $\Pi_{\mathsf{FC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Eval}, \mathsf{Verify})$ with the following properties:

- $\mathsf{Setup}(1^\lambda, 1^\ell, 1^d) \to \mathsf{crs}$: On input the security parameter λ, the input length ℓ, and the bound on the circuit depth d, the setup algorithm outputs a common reference string crs.
- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x}) \to (\sigma, \mathsf{st})$: On input the common reference string crs and an input $\mathbf{x} \in \{0,1\}^\ell$, the commitment algorithm outputs a commitment σ and a state st.
- $\mathsf{Eval}(\mathsf{st}, f) \to \pi_f$: On input a commitment state st and a function $f \in \mathcal{F}$, the evaluation algorithm outputs an opening π_f.
- $\mathsf{Verify}(\mathsf{crs}, \sigma, f, y, \pi) \to \{0,1\}$: On input the common reference string crs, a commitment σ, a function $f \in \mathcal{F}$, a value $y \in \{0,1\}$, and an opening π, the verification algorithm outputs a bit $b \in \{0,1\}$.

We now define several correctness and security properties on the functional commitment scheme:

- **Correctness:** For all security parameters λ, all functions $f \in \mathcal{F}$, and all inputs $\mathbf{x} \in \{0,1\}^\ell$,

$$\Pr\left[\mathsf{Verify}\big(\mathsf{crs}, \sigma, f, f(\mathbf{x}), \pi_f\big) = 1 : \begin{array}{c}\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d);\\ (\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x});\\ \pi_f \leftarrow \mathsf{Eval}(\mathsf{st}, f)\end{array}\right] = 1 - \mathsf{negl}(\lambda).$$

- **Succinctness:** The functional commitment scheme is succinct if there exists a universal polynomial $\mathsf{poly}(\cdot,\cdot,\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\sigma| = \mathsf{poly}(\lambda, d, \log \ell)$ and $|\pi_f| = \mathsf{poly}(\lambda, d, \log \ell)$ in the correctness definition.[10]
- **Binding:** We say Π_{FC} satisfies statistical (resp., computational) binding if for all adversaries $\mathcal{A}$ (resp., efficient adversaries $\mathcal{A}$),

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, \sigma, f, 0, \pi_0) = 1 = \mathsf{Verify}(\mathsf{crs}, \sigma, f, 1, \pi_1) : \begin{array}{c}\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d);\\ (\sigma, f, \pi_0, \pi_1) \leftarrow \mathcal{A}(1^\lambda, 1^\ell, 1^d, \mathsf{crs})\end{array}\right] = \mathsf{negl}(\lambda).$$

- **Private openings:** For an adversary $\mathcal{A}$ and a simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$, we start by defining two distributions $\mathsf{Real}_{\mathcal{A}}(1^\lambda, 1^\ell, 1^d)$ and $\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda, 1^\ell, 1^d)$:

$\mathsf{Real}_{\mathcal{A}}(1^\lambda, 1^\ell, 1^d)$:	$\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda, 1^\ell, 1^d)$:
1. Give $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d)$ to $\mathcal{A}$. 2. Algorithm $\mathcal{A}$ outputs an input $\mathbf{x} \in \{0,1\}^\ell$. 3. Compute $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ and give σ to $\mathcal{A}$. 4. Algorithm $\mathcal{A}$ outputs a function $f \in \mathcal{F}_\lambda$. 5. Give $\pi_f \leftarrow \mathsf{Eval}(\mathsf{st}, f)$ to $\mathcal{A}$. 6. Algorithm $\mathcal{A}$ outputs a bit $b \in \{0,1\}$ which is the output of the experiment.	1. Sample $(\mathsf{crs}, \sigma, \mathsf{st}) \leftarrow \mathcal{S}_0(1^\lambda, 1^\ell, 1^d)$ and give crs to $\mathcal{A}$. 2. Algorithm $\mathcal{A}$ outputs an input $\mathbf{x} \in \{0,1\}^\ell$. 3. Give σ to $\mathcal{A}$. 4. Algorithm $\mathcal{A}$ outputs a function $f \in \mathcal{F}_\lambda$. 5. Compute $\pi_f \leftarrow \mathcal{S}_1(\mathsf{st}, f, f(\mathbf{x}))$ and give π_f to $\mathcal{A}$. 6. Algorithm $\mathcal{A}$ outputs a bit $b \in \{0,1\}$ which is the output of the experiment.

  We say that Π_{FC} has statistical (resp., computational) private openings if for all adversaries $\mathcal{A}$ (resp., efficient adversaries $\mathcal{A}$), there exists an efficient simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ such that $\mathsf{Real}_{\mathcal{A}}(1^\lambda, 1^\ell, 1^d)$ and $\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}(1^\lambda, 1^\ell, 1^d)$ are statistically (resp., computationally) indistinguishable.

Construction 4.2 (Succinct Functional Commitment)**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, and $q = q(\lambda)$ be lattice parameters where q is prime. Let $m' = n(\lceil \log q \rceil + 1)$ and $B = B(\lambda)$ be a bound. Let $s_0 = s_0(\lambda)$, $s_1 = s_1(\lambda)$ be Gaussian width parameters. Let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of Boolean valued functions $f\colon \{0,1\}^\ell \to \{0,1\}$

[10] We could consider an even stronger notion of succinctness where the size of the commitment and the opening depends polylogarithmically on the *size* of the Boolean circuits computing $\mathcal{F}$. However, like existing (non-succinct) lattice-based homomorphic commitments and signatures [GVW15], the size of the commitment and openings in our construction scale with the *depth* of the computation.

where each function $f\colon \{0,1\}^\ell \to \{0,1\}$ is a function on inputs of length $\ell = \ell(\lambda)$ and which can be computed by a Boolean circuit of depth at most $d = d(\lambda)$. We construct a functional commitment $\Pi_{\mathsf{VC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ for $\mathcal{F}$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^\ell, 1^d)$: On input the security parameter λ, the input length ℓ, and the bound d on the circuit depth, the setup algorithm samples $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ and for each $i \in [\ell]$, samples an *invertible* matrix $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n\times n}$. Next, it computes $\tilde{\mathbf{R}}_i \leftarrow \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G}) \in \mathbb{Z}_q^{m\times m'}$ for each $i \in [\ell]$ and constructs matrices $\mathbf{B}_\ell$ and $\mathbf{R}$ as follows:

$$\mathbf{B}_\ell = \left[\begin{array}{ccc|c} \mathbf{W}_1\mathbf{A} & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{W}_\ell\mathbf{A} & -\mathbf{G} \end{array}\right] \in \mathbb{Z}_q^{n\ell\times(\ell m+m')} \quad \text{and} \quad \tilde{\mathbf{R}} = \left[\begin{array}{c} \mathsf{diag}(\tilde{\mathbf{R}}_1, \ldots, \tilde{\mathbf{R}}_\ell) \\ \hline \mathbf{0}^{m'\times\ell m'} \end{array}\right] \in \mathbb{Z}_q^{(\ell m+m')\times\ell m'}. \tag{4.1}$$

  Finally, the setup algorithm samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \mathbf{G}_{n\ell}, s_0)$ and outputs the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$.

- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x})$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$ and a vector $\mathbf{x} \in \{0,1\}^\ell$, the commit algorithm constructs $\mathbf{B}_\ell$ from $\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell$ according to Eq. (4.1). It then constructs a target matrix

$$\mathbf{U}_\mathbf{x} = \begin{bmatrix} -x_1\mathbf{W}_1\mathbf{G} \\ \vdots \\ -x_\ell\mathbf{W}_\ell\mathbf{G} \end{bmatrix} \in \mathbb{Z}_q^{n\ell\times m'}. \tag{4.2}$$

  It then uses $\mathbf{T}$ to sample a preimage

$$\begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_\ell \\ \hat{\mathbf{C}} \end{bmatrix} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{T}, \mathbf{U}_\mathbf{x}, s_1). \tag{4.3}$$

  It outputs the commitment $\sigma = \mathbf{C} = \mathbf{G}\hat{\mathbf{C}} \in \mathbb{Z}_q^{n\times m'}$ and the state $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$.

- $\mathsf{Eval}(\mathsf{crs}, \mathsf{st}, f)$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$, a commitment state $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$, and a function $f\colon \{0,1\}^\ell \to \{0,1\}$, the evaluation algorithm sets $\tilde{\mathbf{C}} \leftarrow [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$, computes $\mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} \leftarrow \mathsf{EvalFX}(\tilde{\mathbf{C}}, f, \mathbf{x})$, and outputs the opening $\pi_f = \mathbf{V}_f \leftarrow [\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell] \cdot \mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}}$.

- $\mathsf{Verify}(\mathsf{crs}, \sigma, f, y, \pi)$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$, a commitment $\sigma = \mathbf{C} \in \mathbb{Z}_q^{n\times m'}$, a function $f\colon \{0,1\}^\ell \to \{0,1\}$, a value $y \in \{0,1\}$, and an opening $\pi = \mathbf{V}_f \in \mathbb{Z}_q^{m\times m'}$, the verification algorithm sets $\tilde{\mathbf{C}} \leftarrow [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$, computes $\tilde{\mathbf{C}}_f \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$ and outputs 1 if

$$\left\|\mathbf{V}_f\right\| \le B \quad \text{and} \quad \mathbf{A}\mathbf{V}_f = \tilde{\mathbf{C}}_f - y\mathbf{G}. \tag{4.4}$$

Theorem 4.3 (Correctness)**.** *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, $s_0 \ge O(\ell m^2 \log(n\ell))$, $s_1 \ge O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$ and $B \ge \sqrt{\ell m + m'} \cdot (n \log q)^{O(d)} \cdot s_1$. Then, Construction 4.2 is correct.*

Proof. Take a security parameter λ, a function $f \in \mathcal{F}_\lambda$, and an input $\mathbf{x} \in \{0,1\}^\ell$. Let $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d)$ and $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ where $\sigma = \mathbf{C} = \mathbf{G}\hat{\mathbf{C}} = \mathbb{Z}_q^{n\times m'}$ and $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$. Let $\pi = \mathbf{V}_f \leftarrow \mathsf{Eval}(\mathsf{crs}, \mathsf{st}, f)$ and consider $\mathsf{Verify}(\mathsf{crs}, \sigma, f, f(\mathbf{x}), \pi)$:

- Let $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell\times(\ell m+m')}$ and $\tilde{\mathbf{R}} \in \mathbb{Z}_q^{(\ell m+m')\times\ell m'}$ be the matrices from Eq. (4.1). For each $i \in [\ell]$, we have that $\tilde{\mathbf{R}}_i = \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$, so

$$\mathbf{W}_i\mathbf{A}\tilde{\mathbf{R}}_i = \mathbf{W}_i\mathbf{A}\mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G}) = \mathbf{W}_i\mathbf{G}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G}) = \mathbf{G}.$$

Thus, $\mathbf{B}_\ell \tilde{\mathbf{R}} = \mathbf{G}_{n\ell}$ and $\|\tilde{\mathbf{R}}\| = \max_{i\in[\ell]} \|\tilde{\mathbf{R}}_i\| \le m' = O(n \log q)$ since $\|\mathbf{R}\| = 1 = \|\mathbf{G}^{-1}(\mathbf{W}_i^{-1}(\mathbf{G}))\|$ for all $i \in [\ell]$. Suppose also that $m \ge m' = O(n \log q)$. By Theorem 2.12 and Lemmas 2.6 and 2.9, for sufficiently-large $m \ge O(n \log q)$,

$$s_0 \ge \sqrt{(\ell m + m')\ell m'} \|\tilde{\mathbf{R}}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell m^2 \log(n\ell)),$$

then $\mathbf{B}_\ell \mathbf{T} = \mathbf{G}_{n\ell}$ and $\|\mathbf{T}\| \le \sqrt{\ell m + m'} s_0$ with overwhelming probability.

- Suppose $s_1 \ge \sqrt{(\ell m + m')\ell m'}\, \|\mathbf{T}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$. By Theorem 2.12 and by construction of $(\mathbf{V}_1, \ldots, \mathbf{V}_\ell, \mathbf{C})$ in Eq. (4.3), we have

  $$\mathbf{W}_i \mathbf{A}\mathbf{V}_i - \mathbf{C} = \mathbf{W}_i \mathbf{A}\mathbf{V}_i - \mathbf{G}\hat{\mathbf{C}} = -x_i \mathbf{W}_i \mathbf{G}.$$

  Rearranging, this means $\mathbf{A}\mathbf{V}_i = \mathbf{W}_i^{-1}\mathbf{C} - x_i \mathbf{G}$. Let $\tilde{\mathbf{C}} = [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$ and $\tilde{\mathbf{V}} = [\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell]$. Then,

  $$\tilde{\mathbf{C}} - \mathbf{x}^\intercal \otimes \mathbf{G} = \mathbf{A}[\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell] = \mathbf{A}\tilde{\mathbf{V}}. \tag{4.5}$$

  Let $B_0 = \sqrt{\ell m + m'} \cdot s_1$ be the "initial" noise bound. Since $\mathbf{A}$ is statistically close to uniform over $\mathbb{Z}_q^{n\times m}$ and $\mathbf{W}_i \in \mathbb{Z}_q^{n\times n}$ is invertible, $\mathbf{W}_i\mathbf{A}$ is also statistically close to uniform over $\mathbb{Z}_q^{n\times m}$. By Lemma 2.2 and Corollary 2.5, with overwhelming probability, each $\mathbf{W}_i\mathbf{A}$ is full rank and $\lambda_1^\infty(\Lambda(\mathbf{W}_i\mathbf{A})) \ge q/4$. Then, we can invoke Lemma 2.9 and Corollary 2.11 and conclude that $\|\mathbf{V}_i\| \le \sqrt{\ell m + m'} s_1 = B_0$ and so $\|\tilde{\mathbf{V}}\| \le B_0$.

- By construction of Eval, we have that $\mathbf{V}_f = \tilde{\mathbf{V}} \cdot \mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}}$ where $\mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} \leftarrow \mathsf{EvalFX}(\tilde{\mathbf{C}}, f, \mathbf{x})$. By Theorem 2.15, $\left\|\mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}}\right\| \le (n \log q)^{O(d)}$, so $\|\mathbf{V}_f\| \le m' \cdot B_0 \cdot (n \log q)^{O(d)} \le s_1 \cdot \sqrt{\ell m + m'} \cdot (n \log q)^{O(d)}$.

- Again appealing to Theorem 2.15 and Eq. (4.5), we can write

  $$\mathbf{A}\mathbf{V}_f = \mathbf{A}\tilde{\mathbf{V}}\mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} = (\tilde{\mathbf{C}} - \mathbf{x}^\intercal \otimes \mathbf{G}) \cdot \mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} = \tilde{\mathbf{C}}_f - f(\mathbf{x}) \cdot \mathbf{G},$$

  where $\tilde{\mathbf{C}} = \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$. Correspondingly, $\mathsf{Verify}(\mathsf{crs}, \sigma, f, f(\mathbf{x}), \pi)$ outputs 1. □

Theorem 4.4 (Binding). *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, $s_0 \ge O(\ell m^2 \log(n\ell))$, and $\beta \ge 2Bm\sqrt{m'} \log n$. Then, under the* $\mathsf{BASIS}_{\mathsf{struct}}$ *assumption with parameters $(n, m, q, \beta, s_0, \ell)$, Construction 4.2 is computationally binding.*

Proof. We proceed via a hybrid argument:

- Hyb_0: This is the real binding experiment:
  - The challenger starts by sampling $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ and $\mathbf{W}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times n}$ for each $i \in [\ell]$. It sets $\tilde{\mathbf{R}}_i \leftarrow \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$ and constructs $\mathbf{B}_\ell$ and $\tilde{\mathbf{R}}$ according to Eq. (4.1). It samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \mathbf{G}_{n\ell}, s_0)$ and gives $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$ to the adversary $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{C} \in \mathbb{Z}_q^{n\times m'}$, a function $f \in \mathcal{F}_\lambda$, and openings $\mathbf{V}_0, \mathbf{V}_1 \in \mathbb{Z}_q^{m\times m'}$.
  - The output of the experiment is 1 if $\|\mathbf{V}_0\|, \|\mathbf{V}_1\| \le B$, $\mathbf{A}\mathbf{V}_0 = \tilde{\mathbf{C}}_f$, and $\mathbf{A}\mathbf{V}_1 = \tilde{\mathbf{C}}_f - \mathbf{G}$, where $\tilde{\mathbf{C}}_f \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$ and $\tilde{\mathbf{C}} = [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$. Otherwise, the experiments outputs 0.
- Hyb_1: Same as Hyb_0 except after constructing the matrix $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell\times(\ell m+m')}$ according to Eq. (4.1), the challenger samples $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_{s_0}^{-1}(\mathbf{G}_{n\ell})$ *without* using the trapdoor $\tilde{\mathbf{R}}$. The CRS is now sampled independently of $\mathbf{R}$.
- Hyb_2: Same as Hyb_1 except the challenger samples $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n\times m}$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output of an execution of Hyb_i with adversary $\mathcal{A}$. We now show that each adjacent pair of experiments are computationally indistinguishable.

Lemma 4.5. *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, and $s_0 \ge O(\ell m^2 \log(n\ell))$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_0(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A})$.*

Proof. The only difference between Hyb_0 and Hyb_1 is the distribution of $\mathbf{T}$. In Hyb_0, $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \mathbf{G}_{n\ell}, s_0)$. As shown in the proof of Theorem 4.3, $\mathbf{B}_\ell \tilde{\mathbf{R}} = \mathbf{G}_{n\ell}$ and $\|\tilde{\mathbf{R}}\| \le m' = O(n \log q)$. Suppose $m \ge m' = O(n \log q)$ and

$$s_0 \ge \sqrt{(\ell m + m')\ell m'} \|\tilde{\mathbf{R}}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell m^2 \log(n\ell)).$$

By Theorem 2.12 the distribution of $\mathbf{T}$ is statistically close to $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_{s_0}^{-1}(\mathbf{G}_{n\ell})$, which is the distribution in Hyb_1. □

Lemma 4.6. *Suppose $n \ge \lambda$ and $m \ge O(n \log q)$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_1(\mathcal{A}) \stackrel{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

Proof. The only difference between Hyb_1 and Hyb_2 is the distribution of $\mathbf{A}$. In Hyb_1, the challenger samples $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$. By Theorem 2.12, the distribution of $\mathbf{A}$ is statistically close to $\mathbf{A} \stackrel{\mathrm{R}}{\leftarrow} \mathbb{Z}_q^{n \times m}$. □

Lemma 4.7. *Suppose $\beta \ge 2Bm\sqrt{m'} \log n$. Under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with parameters $(n, m, q, \beta, s_0, \ell)$, for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists an efficient adversary $\mathcal{A}$ where $\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] = \varepsilon$ for some non-negligible ε. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption:

1. Algorithm $\mathcal{B}$ receives a challenge $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$, $\mathbf{T} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$, and $\mathsf{aux} = (\mathbf{W}_1, \ldots, \mathbf{W}_\ell)$. Algorithm $\mathcal{B}$ gives $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$ to $\mathcal{A}$.
2. Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{C} \in \mathbb{Z}_q^{n \times m'}$, a function $f \in \mathcal{F}_\lambda$, and openings $\mathbf{V}_0, \mathbf{V}_1 \in \mathbb{Z}_q^{m \times m'}$.
3. Algorithm $\mathcal{B}$ outputs $\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{V}_0 - \mathbf{V}_1, \mathbf{0}, s')$ where $s' = 2B\sqrt{mm'} \log n$.

By construction, algorithm $\mathcal{B}$ perfectly simulates the common reference string according to the specification of Hyb_2. Thus, with probability ε, $\|\mathbf{V}_0\|, \|\mathbf{V}_1\| \le B$, $\mathbf{A}\mathbf{V}_0 = \tilde{\mathbf{C}}_f$, $\mathbf{A}\mathbf{V}_1 = \tilde{\mathbf{C}}_f - \mathbf{G}$. This means that $\mathbf{A}(\mathbf{V}_0 - \mathbf{V}_1) = \mathbf{G}$, so $\mathbf{V}_0 - \mathbf{V}_1$ is a trapdoor for $\mathbf{A}$. By Theorem 2.12, the distribution of $\mathbf{x}$ is statistically close to $\mathbf{A}_{s'}^{-1}(\mathbf{0})$. Thus, $\mathbf{x}$ is non-zero with probability $1 - \mathsf{negl}(n)$. Finally, by Lemma 2.9, $\|\mathbf{x}\| \le \sqrt{m} s' = \beta$, and the claim holds. □

Combining Lemmas 4.5 to 4.7, the functional commitment scheme is computationally binding. □

Parameter instantiation. Let λ be a security parameter and $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of functions $f : \{0,1\}^\ell \to \{0,1\}$ on inputs of length $\ell = \ell(\lambda)$ and which can be computed by Boolean circuits of depth at most $d = d(\lambda)$. We instantiate the lattice parameters in Construction 4.2 as follows to satisfy Theorems 4.3 and 4.4:

- Let $\varepsilon > 0$ be a constant. We set the lattice dimension $n = d^{1/\varepsilon} \cdot \mathsf{poly}(\lambda)$ and $m = O(n \log q)$.
- We set $s_0 = O(\ell m^2 \log(n\ell))$ and

$$s_1 = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0) = O(\ell^{5/2} m^{7/2} \log^2(n\ell)) = O(\ell^{5/2} n^{7/2} \log^2(n\ell) \log^{7/2} q).$$

- We set the bound $B = s_1 \cdot \sqrt{\ell m + m'} \cdot (n \log q)^{O(d)} = \ell^3 \log^2 \ell \cdot (n \log q)^{O(d)}$.
- We set the modulus q so that the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption holds with parameters $(n, m, q, \beta, s_0, \ell)$, where

$$\beta = 2Bm\sqrt{m'} \log n = \ell^3 \log^2 \ell \cdot (n \log q)^{O(d)} = 2^{\tilde{O}(d)} = 2^{\tilde{O}(n^\varepsilon)},$$

  where we write $\tilde{O}(\cdot)$ to suppress polylogarithmic factors in λ, d, ℓ. Note that this also requires that $\mathsf{SIS}_{n,m,q,\beta}$ hold. For instance, we set $q = \beta \cdot \mathsf{poly}(n)$. Then, $\log q = \mathsf{poly}(d, \log \lambda, \log \ell)$. Note that the underlying SIS assumption now relies on a *sub-exponential* noise bound.

With this setting of parameters, we obtain a functional commitment scheme for $\mathcal{F}$ with the following parameter sizes:

- **Commitment size:** A commitment σ to an input $\mathbf{x} \in \{0,1\}^\ell$ consists of a matrix $\sigma = \mathbf{C} \in \mathbb{Z}_q^{n \times m'}$ so

$$|\sigma| = nm' \log q = O(n^2 \log^2 q) = \mathsf{poly}(\lambda, d, \log \ell).$$

- **Opening size:** An opening π to a function f consists of a matrix $\pi = \mathbf{V}_f \in \mathbb{Z}_q^{m \times m'}$. Then,

$$|\pi| = mm' \log q = \mathsf{poly}(\lambda, d, \log \ell).$$

  In Remark 4.9, we describe a simple approach to compress the opening to a *vector* instead of a matrix.

- **CRS size:** The CRS consists of $(\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$, where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{W}_i \in \mathbb{Z}_q^{n \times n}$, and $\mathbf{T} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$. Thus, the total size of the CRS is

$$|\mathsf{crs}| = nm \log q + \ell n^2 \log q + (\ell m + m')(\ell m') \log q = \ell^2 \cdot \mathsf{poly}(\lambda, d, \log \ell).$$

Thus, Construction 4.2 is a succinct functional commitment scheme for bounded-depth circuits. We summarize the instantiation in the following corollary:

Corollary 4.8 (Succinct Vector Commitment from $\mathsf{BASIS}_{\mathsf{struct}}$)**.** *Let λ be a security parameter, and let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of functions $f \colon \{0,1\}^\ell \to \{0,1\}$ on inputs of length $\ell = \ell(\lambda)$ and which can be computed by Boolean circuits of depth at most $d = d(\lambda)$. Under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with a norm bound $\beta = 2^{\tilde{O}(d)}$ and modulus $q = 2^{\tilde{O}(d)}$, there exists a computationally-binding succinct functional commitment scheme for $\mathcal{F}$. Both the size of the commitment and the opening are $\mathsf{poly}(\lambda, d, \log \ell)$, and the CRS has size $\ell^2 \cdot \mathsf{poly}(\lambda, d, \log \ell)$. Here, $\tilde{O}(\cdot)$ suppresses polylogarithmic factors in λ, d, and ℓ.*

Remark 4.9 (Reducing the Opening Size)**.** An opening π_f to a function f in Construction 4.2 consists of a matrix $\pi_f = \mathbf{V}_f \in \mathbb{Z}_q^{m \times m'}$ where $m, m' = O(n \log q)$. It is easy to adapt Construction 4.2 to obtain slightly shorter openings (i.e., $\pi_f = \mathbf{v}_f \in \mathbb{Z}_q^m$). The idea is simple: we publish a random target vector $\mathbf{u} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ in the CRS and define the new opening to be $\mathbf{v}_f \leftarrow \mathbf{V}_f \mathbf{G}^{-1}(\mathbf{u})$, where $\mathbf{V}_f$ is the original opening from Construction 4.2. The updated verification relation then checks that $\|\mathbf{v}_f\|$ is small and that $\mathbf{A}\mathbf{v}_f = \tilde{\mathbf{C}}\mathbf{G}^{-1}(\mathbf{u}) - y \cdot \mathbf{u}$. We also use this approach to aggregate openings in Section 5.1 (see Construction 5.16). However, giving out the "matrix" opening is convenient when specializing our construction to obtain polynomial commitments (Section 4.1).

Remark 4.10 (Comparison with [ACL+22])**.** The authors of [ACL+22] showed how to construct a functional commitment for constant-degree polynomials where the size of the CRS scales exponentially with the *degree* of the polynomial. Our functional commitment scheme (Construction 4.2) supports arbitrary Boolean circuits of bounded depth, and the size of our CRS scales polynomially with the depth of the circuit family. Moreover, security of our construction can be reduced to a function-independent assumption (the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption) whereas the scheme in [ACL+22] relied on a function-dependent assumption. We compare the two assumptions in more detail in Section 6.

An advantage of the [ACL+22] construction is that if supports *fast* verification with preprocessing. Namely, in their scheme, the verifier can precompute a verification key for a function f, and subsequently, verify openings with respect to f in time that is *polylogarithmic* in the running time of f. In contrast, with our scheme, the verifier has to first homomorphically compute f on the commitment in order to verify. In Section 4.1, we show that for the special case of linear functions, we can adapt Construction 4.2 to support fast verification in the preprocessing model.

Remark 4.11 (A Candidate SNARG with Expensive Verification)**.** The authors of [ACL+22] show how to boost their functional commitment for quadratic polynomials to a preprocessing SNARG for NP as follows:

1. First, [ACL+22] applying "sparsification" to their functional commitment scheme. Over the integers, one analog of sparsification is to require the adversary to output a short $\tilde{\mathbf{V}}$ such that $\tilde{\mathbf{A}}\tilde{\mathbf{V}} = \mathbf{D}\mathbf{C}$, where $\mathbf{D} \in \mathbb{Z}_q^{2m \times n}$, where $\tilde{\mathbf{A}} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{2m \times 2m \log q}$ is a sparsification matrix. The CRS includes short preimages of $\tilde{\mathbf{A}}$ to enable sampling of $\tilde{\mathbf{V}}$.

2. Next, [ACL+22] introduce a knowledge assumption that says that the only way an adversary can produce $\mathbf{C}$ and $\tilde{\mathbf{V}}$ is by computing a short linear combination of the preimages in the CRS (where the coefficients correspond to the committed vector $\mathbf{x}$).

3. To support openings to multiple quadratic polynomials with a succinct opening (i.e., sublinear in the number of openings), [ACL+22] introduces a novel SIS-based technique.

Taken together, [ACL+22] show how to obtain an "extractable" commitment scheme, a notion that is equivalent to a succinct argument of knowledge for satisfiability of quadratic systems. This yields a publicly-verifiable preprocessing SNARG for NP since satisfiability of degree-2 polynomials is NP-complete. Specifically, a proof for a statement x consists of a commitment σ to a satisfying witness w and an opening π of σ to a satisfying assignment to the quadratic constraint system representing the NP relation. By relying on preprocessing (Remark 4.10), the [ACL+22] SNARG has short proofs and fast verification.

We can apply an analogous approach to our functional commitment scheme (Construction 4.2) to obtain a candidate SNARG for NP; our SNARG would have short proofs but an *expensive* verification step (since our functional commitment does not support fast verification in the preprocessing model). We also note that even without sparsification, our construction is still a candidate SNARG: we do not know how to prove soundness of our construction, but at the same time, are not aware of any attacks either. An attack on our candidate SNARG (without sparsification) would be interesting, and we invite cryptanalysis of our candidate.

Remark 4.12 (Deterministic Commitments)**.** Here, we describe a variant of Construction 4.2 with a *deterministic* commitment algorithm. The advantage of this approach is that computing the commitment $\sigma = \mathbf{C}$ now runs in time $\ell \cdot \mathsf{poly}(\lambda, d, \log \ell)$ time; with preimage sampling, the same algorithm requires time that scales quadratically with the dimension ℓ. The drawback of using a deterministic commitment procedure is that the scheme no longer extends to support private openings (see Section 4.2). To support deterministic commitments, we proceed as follows:

- In Setup, construct the matrix $\mathbf{B}_\ell$ exactly as in Eq. (4.1), and define the target matrix

$$\tilde{\mathbf{U}} = \begin{bmatrix} -\mathbf{W}_1\mathbf{G} & & \\ & \ddots & \\ & & -\mathbf{W}_\ell\mathbf{G} \end{bmatrix} \in \mathbb{Z}_q^{n\ell \times \ell m'}.$$

  Then, the Setup algorithm constructs the matrix $\mathbf{T} = \begin{bmatrix} \mathbf{T}_{\mathsf{open}} \\ \mathbf{T}_{\mathsf{com}} \end{bmatrix}$ as

$$\begin{bmatrix} \mathbf{T}_{\mathsf{open}} \\ \mathbf{T}_{\mathsf{com}} \end{bmatrix} \leftarrow \mathsf{SamplePre}\left(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \tilde{\mathbf{U}}, s_1\right),$$

  where $\mathbf{T}_{\mathsf{open}} \in \mathbb{Z}_q^{n\ell \times \ell m'}$ and $\mathbf{T}_{\mathsf{com}} \in \mathbb{Z}_q^{n \times \ell m'}$.

- To commit to a vector $\mathbf{x} \in \{0,1\}^\ell$, the Commit algorithm computes $\hat{\mathbf{C}} = \mathbf{T}_{\mathsf{com}}(\mathbf{x} \otimes \mathbf{I}_{m'})$ and sets $\sigma = \mathbf{C} = \mathbf{G}\hat{\mathbf{C}}$.

- To evaluate a function f on a commitment to $\mathbf{x}$, the Eval algorithm first computes

$$\begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_\ell \end{bmatrix} \leftarrow \mathbf{T}_{\mathsf{open}}(\mathbf{x} \otimes \mathbf{I}_{m'})$$

  and then proceeds exactly as in Construction 4.2.

To see that this preserves correctness, observe that

$$\mathbf{B}_\ell \cdot \begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_\ell \\ \hat{\mathbf{C}} \end{bmatrix} = \mathbf{B}_\ell \mathbf{T}(\mathbf{x} \otimes \mathbf{I}_{m'}) = \tilde{\mathbf{U}}(\mathbf{x} \otimes \mathbf{I}_{m'}) = \begin{bmatrix} -x_1\mathbf{W}_1\mathbf{G} \\ \vdots \\ -x_\ell\mathbf{W}_\ell\mathbf{G} \end{bmatrix} = \mathbf{U}_{\mathbf{x}}.$$

Thus, $\mathbf{V}_1, \ldots, \mathbf{V}_\ell, \hat{\mathbf{C}}$ is a short solution to the same linear system as in Eq. (4.3). Correctness follows by the same analysis as in Theorem 4.3 (with slightly larger bounds). Security follows similarly to the proof of Theorem 4.4 (since $\mathbf{T}_{\mathsf{com}}, \mathbf{T}_{\mathsf{open}}$ can be derived from the CRS components in Construction 4.2). Observe that with this deterministic commitment and opening procedures and using the same parameter instantiation as in Corollary 4.8, the running time Commit is $\ell \cdot \mathsf{poly}(\lambda, d, \log \ell)$ and the running time of Eval (on function f) is $|f| \cdot \mathsf{poly}(\lambda, d, \log \ell)$.

Remark 4.13 (An Alternative Formulation of Construction 4.2)**.** We can obtain an alternative formulation of Construction 4.2 that does not require explicit matrix inverses by starting with the alternative version of $\mathsf{BASIS}_{\mathsf{struct}}$ in Remark 3.8. Specifically, we define the matrix

$$\tilde{\mathbf{B}}_\ell = [\mathbf{I}_\ell \otimes \mathbf{A} \mid \tilde{\mathbf{W}}\mathbf{G}] \quad \text{where} \quad \tilde{\mathbf{W}} = \begin{bmatrix} \tilde{\mathbf{W}}_1 \\ \vdots \\ \tilde{\mathbf{W}}_\ell \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{W}}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times n}.$$

Let $\mathbf{T}$ be a trapdoor for $\tilde{\mathbf{B}}_\ell$. The CRS is then $(\mathbf{A}, \tilde{\mathbf{W}}_1, \ldots, \tilde{\mathbf{W}}_\ell, \mathbf{T})$. To commit to an input $\mathbf{x}$, the prover now computes

$$\begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_\ell \\ \hat{\mathbf{C}} \end{bmatrix} \leftarrow \mathsf{SamplePre}(\tilde{\mathbf{B}}_\ell, \mathbf{T}, -\mathbf{x} \otimes \mathbf{G}, s_1).$$

By construction of $\tilde{\mathbf{B}}_\ell$ and again setting $\mathbf{C} = \mathbf{G}\hat{\mathbf{C}}$, we have for all $i \in [\ell]$,

$$\mathbf{A}\mathbf{V}_i + \tilde{\mathbf{W}}_i\mathbf{C} = -x_i\mathbf{G},$$

or equivalently, $\mathbf{A}\mathbf{V}_i = -\tilde{\mathbf{W}}_i\mathbf{C} - x_i\mathbf{G}$. Letting $\tilde{\mathbf{C}} = [-\tilde{\mathbf{W}}_1\mathbf{C} \mid \cdots \mid -\tilde{\mathbf{W}}_\ell\mathbf{C}]$ and $\tilde{\mathbf{V}} = [\tilde{\mathbf{V}}_1 \mid \cdots \mid \tilde{\mathbf{V}}_\ell]$, we can again write

$$\mathbf{A}\tilde{\mathbf{V}} = \tilde{\mathbf{C}} - \mathbf{x}^\intercal \otimes \mathbf{G},$$

from which correctness follows as in the proof of Theorem 4.3. The binding analysis follows the same structure as the proof of Theorem 4.4, except we rely on the variant of $\mathsf{BASIS}_{\mathsf{struct}}$ where $\mathbf{T}$ is a trapdoor for $\tilde{\mathbf{B}}_\ell$. For the particular case where q is prime (i.e., the setting where $\tilde{\mathbf{W}}_1, \ldots, \tilde{\mathbf{W}}_\ell$ are invertible with overwhelming probability), these two variants of $\mathsf{BASIS}_{\mathsf{struct}}$ are equivalent (see Remark 3.8).

4.1 Opening to Linear Functions and Applications to Polynomial Commitments

An appealing property of the succinct functional commitment by Albrecht et al. [ACL+22] and the non-succinct construction of Gorbunov et al. [GVW15] is they support *fast* verification after an initial slow preprocessing step. In these schemes, there is a preprocessing algorithm that takes the CRS and the description of a function f, and outputs a short verification key vk_f. Later on, given a commitment σ, an opening π_f, and the precomputed verification key vk_f, the verification algorithm runs in time that is sublinear in the time it takes to compute f. In contrast, our succinct functional commitment (Construction 4.2) has short commitments and openings, but the verification algorithm runs in time proportional to f (in order to homomorphically compute $\tilde{\mathbf{C}}_f$ from $\tilde{\mathbf{C}}$).

Here, we describe a simple adaptation of Construction 4.2 for the setting of *linear* functions that supports fast verification in the preprocessing model. Our construction naturally supports linear functions over $\mathbb{Z}_q^\ell$ (as opposed to $\{0,1\}^\ell$) and generalizes to yield a *polynomial commitment* [KZG10]. Unlike [ACL+22], we do *not* require the values in the committed vector or the output of the linear function to be small. Supporting *large* values is necessary for obtaining a succinct polynomial commitment. We start by describing a construction that supports linear functions with *small* coefficients, and then show how to extend the construction to arbitrary linear functions over $\mathbb{Z}_q^\ell$.

Construction 4.14 (Succinct Functional Commitment for Linear Functions)**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, and $q = q(\lambda)$ be lattice parameters. Let $m' = n(\lceil \log q \rceil + 1)$ and $B = B(\lambda)$ be a bound. Let $s_0 = s_0(\lambda)$, $s_1 = s_1(\lambda)$ be Gaussian width parameters. Let $\ell = \ell(\lambda)$ be an input length. For a vector $\mathbf{z} \in \{0,1\}^\ell$, let $f_{\mathbf{z}} \colon \mathbb{Z}_q^\ell \to \mathbb{Z}_q$ be the linear function $\mathbf{x} \mapsto \mathbf{z}^\intercal\mathbf{x}$. Let $\mathcal{F}_\lambda = \{f_{\mathbf{z}} \mid \mathbf{z} \in \{0,1\}^\ell\}$. We construct a functional commitment $\Pi_{\mathsf{FC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Eval}, \mathsf{Verify})$ for $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^\ell)$: Same as Construction 4.2.
- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x})$: Same as Construction 4.2 (with $\mathbf{x} \in \mathbb{Z}_q^\ell$).

- $\mathsf{Eval}(\mathsf{crs}, \mathsf{st}, f_{\mathbf{z}})$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$, a commitment state $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$, and a function $f_{\mathbf{z}}$ where $\mathbf{z} \in \{0,1\}^\ell$, the evaluation algorithm outputs $\mathbf{V}_{\mathbf{z}} = \sum_{i \in [\ell]} z_i \mathbf{V}_i$.
- $\mathsf{Verify}(\mathsf{crs}, \sigma, f_{\mathbf{z}}, y, \pi)$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$, a commitment $\sigma = \mathbf{C} \in \mathbb{Z}_q^{n \times m'}$, a function $f_{\mathbf{z}} \colon \mathbb{Z}_q^\ell \to \mathbb{Z}_q$ where $\mathbf{z} \in \{0,1\}^\ell$, a value $y \in \mathbb{Z}_q$, and an opening $\pi = \mathbf{V}_{\mathbf{z}} \in \mathbb{Z}_q^{m \times m'}$, the verification algorithm outputs 1 if

$$\|\mathbf{V}_{\mathbf{z}}\| \le B \quad \text{and} \quad \mathbf{A}\mathbf{V}_{\mathbf{z}} = \sum_{i \in [\ell]} z_i \mathbf{W}_i^{-1}\mathbf{C} - y\mathbf{G}.$$

Remark 4.15 (Verification in the Preprocessing Model)**.** As described, the running time of the verification algorithm Construction 4.14 is linear in the vector dimension ℓ. However, observe that the verifier can *precompute* the matrix $\mathbf{W}_{\mathbf{z}} \coloneqq \sum_{i \in [\ell]} z_i \mathbf{W}_i^{-1}$, which only depends on the CRS and the function $\mathbf{z} \in \{0,1\}^\ell$. Given the precomputed verification key $\mathbf{W}_{\mathbf{z}}$, the verification relation becomes:

$$\|\mathbf{V}_{\mathbf{z}}\| \le B \quad \text{and} \quad \mathbf{A}\mathbf{V}_{\mathbf{z}} = \mathbf{W}_{\mathbf{z}}\mathbf{C} - y\mathbf{G}.$$

This yields a verification algorithm whose running time is $\mathsf{poly}(\lambda, \log \ell)$.

Theorem 4.16 (Correctness)**.** *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, $s_0 \ge O(\ell m^2 \log(n\ell))$, $s_1 \ge O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$ and $B \ge \ell\sqrt{\ell m + m'} \cdot s_1$. Then, Construction 4.14 is correct.*

Proof. This follows by a similar argument as the proof of Theorem 4.3. Take a security parameter λ, a vector $\mathbf{z} \in \{0,1\}^\ell$, and an input $\mathbf{x} \in \mathbb{Z}_q^\ell$. Let $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ where $\sigma = \mathbf{C} = \mathbf{G}\hat{\mathbf{C}} = \mathbb{Z}_q^{n \times m'}$ and $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$. Let $\pi = \mathbf{V}_{\mathbf{z}} \leftarrow \mathsf{Eval}(\mathsf{crs}, \mathsf{st}, f_{\mathbf{z}})$ and consider $\mathsf{Verify}(\mathsf{crs}, \sigma, f_{\mathbf{z}}, \mathbf{z}^\intercal \mathbf{x}, \pi)$:

- By the analysis in the proof of Theorem 4.3, we have $\mathbf{A}\mathbf{V}_i = \mathbf{W}_i^{-1}\mathbf{C} - x_i\mathbf{G}$ where $\|\mathbf{V}_i\| \le \sqrt{\ell m + m'} s_1$.
- By construction of Eval, we have $\mathbf{V}_{\mathbf{z}} = \sum_{i \in [\ell]} z_i \mathbf{V}_i$. Since $z_i \in \{0,1\}$, this means $\|\mathbf{V}_{\mathbf{z}}\| \le \ell\sqrt{\ell m + m'} s_1 \le B$. Moreover,

$$\mathbf{A}\mathbf{V}_z = \sum_{i \in [\ell]} z_i \mathbf{A}\mathbf{V}_i = \sum_{i \in [\ell]} z_i \mathbf{W}_i^{-1}\mathbf{C} - \sum_{i \in [\ell]} z_i x_i \mathbf{G} = \sum_{i \in [\ell]} z_i \mathbf{W}_i^{-1}\mathbf{C} - (\mathbf{z}^\intercal \mathbf{x}) \cdot \mathbf{G},$$

  and the verification algorithm accepts. □

Theorem 4.17 (Binding)**.** *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, $s_0 \ge O(\ell m^2 \log(n\ell))$, q is prime, and $\beta \ge 2Bm\sqrt{m'} \log n$. Then, under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with parameters $(n, m, q, \beta, s_0, \ell)$, Construction 4.14 is computationally binding.*

Proof. We use the same sequence of hybrid experiments as in the proof of Theorem 4.4.

- Hyb_0: This is the real binding experiment.
- Hyb_1: Same as Hyb_0 except the challenger samples $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_{s_0}^{-1}(\mathbf{G}_{n\ell})$ when setting up the CRS.
- Hyb_2: Same as Hyb_1 except the challenger samples $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}$ in the CRS.

For the given choice of parameters, and for all adversaries $\mathcal{A}$, we have that $\mathsf{Hyb}_0(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A})$ and $\mathsf{Hyb}_1(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$ using the same argument as in the proofs of Lemmas 4.5 and 4.6. It suffices to consider the output distribution in Hyb_2:

Lemma 4.18. *Suppose q is prime and $\beta \ge 2Bm\sqrt{m'} \log n$. Under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with parameters $(n, m, q, \beta, s_0, \ell)$, for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists an efficient algorithm $\mathcal{A}$ where $\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] = \varepsilon$ for some non-negligible ε. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption in a similar manner as the proof of Lemma 4.7:

1. Algorithm $\mathcal{B}$ receives a challenge $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$, $\mathbf{T} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$, and $\mathsf{aux} = (\mathbf{W}_1, \ldots, \mathbf{W}_\ell)$. Algorithm $\mathcal{B}$ gives $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T})$ to $\mathcal{A}$.

2. Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{C} \in \mathbb{Z}_q^{n \times m'}$, a function $f_{\mathbf{z}} \colon \mathbb{Z}_q^n \to \mathbb{Z}_q$, where $\mathbf{z} \in \{0,1\}^\ell$, and openings $(y_1, \mathbf{V}_1)$, $(y_2, \mathbf{V}_2)$, where $y_1 \neq y_2 \in \mathbb{Z}_q$ and $\mathbf{V}_1, \mathbf{V}_2 \in \mathbb{Z}_q^{m \times m'}$.

3. Algorithm $\mathcal{B}$ outputs $\mathbf{x} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{V}_1 - \mathbf{V}_2, \mathbf{0}, s')$, where $s' = 2B\sqrt{mm'}\log n$

By construction, algorithm $\mathcal{B}$ perfectly simulates the common reference string according to the specification of Hyb_2. Thus, with probability ε, the following conditions hold: $\|\mathbf{V}_1\|, \|\mathbf{V}_2\| \leq B$, and $\mathbf{A}(\mathbf{V}_1 - \mathbf{V}_2) = (y_2 - y_1)\mathbf{G}$. Thus, $\mathbf{V}_1 - \mathbf{V}_2$ is a trapdoor for $\mathbf{A}$ with tag $(y_2 - y_1)\mathbf{I}_n$. Since q is prime and $y_2 - y_1 \neq 0$, the matrix $(y_2 - y_1)\mathbf{I}_n$ is invertible. Thus, by Theorem 2.12, the distribution of $\mathbf{x}$ is statistically close to $\mathbf{A}_{s'}^{-1}(\mathbf{0})$. Thus, $\mathbf{x}$ is non-zero with overwhelming probability. Finally, by Lemma 2.9 $\|\mathbf{x}\| \leq \sqrt{m}s' = \beta$, and the claim holds. □

Computational binding now follows by a hybrid argument. □

Remark 4.19 (Extending to General $\mathbb{Z}_q$-Linear Functions)**.** While Construction 4.14 supports commitments to arbitrary vectors over $\mathbb{Z}_q$, we still require that the functions $f_{\mathbf{z}}$ have small coefficients. We can support arbitrary linear functions on $\mathbb{Z}_q^\ell$ by blowing up the vector dimension by a factor $k = \lfloor \log q \rfloor + 1 = O(\log q)$ factor:

- The CRS is a CRS for a vector commitment scheme on dimension $\ell k = O(\ell \log q)$: $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell k})$.
- A commitment to a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$ is a commitment to the vector $\mathbf{x} \otimes \mathbf{g} \in \mathbb{Z}_q^{\ell k}$ in the underlying scheme: $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x} \otimes \mathbf{g})$.
- An opening to the linear function $f_{\mathbf{z}} \colon \mathbb{Z}_q^\ell \to \mathbb{Z}_q$ (for an *arbitrary* $\mathbf{z} \in \mathbb{Z}_q^\ell$) is an opening with respect to the linear function $f_{\mathbf{g}^{-1}(\mathbf{z})}$ in the underlying scheme: $\pi \leftarrow \mathsf{Eval}(\mathsf{crs}, \mathsf{st}, f_{\mathbf{g}^{-1}(\mathbf{z})})$.

By construction, $\mathbf{g}^{-1}(\mathbf{z}) \in \{0,1\}^{\ell k}$. For correctness, observe that $(\mathbf{x} \otimes \mathbf{g})^\intercal \mathbf{g}^{-1}(\mathbf{z}) = \mathbf{x}^\intercal \mathbf{z} = \mathbf{z}^\intercal \mathbf{x}$. Thus, with $O(\log q)$ overhead, Construction 4.14 gives a vector commitment scheme for arbitrary linear functions over $\mathbb{Z}_q^\ell$. We no longer impose any size restrictions on the coefficients of the linear function or on the vector values.

Remark 4.20 (Polynomial Commitments)**.** In a polynomial commitment [KZG10], a committer can commit to a polynomial $h \in \mathbb{Z}_q[x]$ over $\mathbb{Z}_q$ and subsequently open to evaluations $h(x)$ of the polynomial on arbitrary points $x \in \mathbb{Z}_q$ of the committer's choosing. The succinctness requirement is that the commitment and the openings are both short (sublinear in the degree of the polynomial). By the approach of Libert et al. [LRY16], our functional commitment for linear functions over $\mathbb{Z}_q$ directly implies a polynomial commitment. Namely, to commit to a degree-d polynomial $h(x) = \sum_{i \in [0,d]} h_i x^i$, the committer constructs a commitment to the vector of coefficients $\mathbf{h} = [h_0, \ldots, h_d]$. To open to a point $x \in \mathbb{Z}_q$, the committer constructs an opening to the function $f_{\mathbf{z}}$ where $\mathbf{x} = [1, x, x^2, \ldots, x^d]$. By construction $\mathbf{x}^\intercal \mathbf{h} = h(x)$. When instantiated with our vector commitment scheme for linear functions (Construction 4.14 and Remark 4.19), the size of the commitment and the size of the opening are both $\mathsf{poly}(\lambda, \log d)$, which satisfies the efficiency requirements for a polynomial commitment. Note though that this construction does not provide hiding.

We also note that a similar transformation from vector commitments supporting linear functions to polynomial commitments does not apply if we start with the scheme of Albrecht et al. [ACL+22]. The [ACL+22] scheme only supports committing to vectors with small values and opening to linear functions with small coefficients (with magnitude smaller than q). Our transformation relies on opening to a linear function with coefficients $(1, x, \ldots, x^d)$. If $q > x^d$, then the bit-length of q scales with the degree of the polynomial and the resulting scheme is no longer succinct. Note that the binary-decomposition approach from Remark 4.19 is not applicable in this case since it scales up the magnitude of the vector components by a factor proportional to x^d.

Parameter instantiation. Let λ be a security parameter, $q = q(\lambda)$ be the modulus, $\ell = \ell(\lambda)$ be a vector dimension, and $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of linear functions over $\mathbb{Z}_q^\ell$. We instantiate the lattice parameters in Construction 4.14 as follows to satisfy Theorems 4.16 and 4.17:

- We set the lattice dimension $n = \mathsf{poly}(\lambda)$ and $m = O(n \log q)$.
- We set $s_0 = O(\ell m^2 \log(n\ell))$ and $s_1 = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0) = O(\ell^{5/2} m^{7/2} \log^2(n\ell))$.
- We set the bound $B = \ell\sqrt{\ell m + m'} s_1 = O(\ell^4 m^4 \log^2(n\ell)) = O(\ell^4 n^4 \log^2(n\ell) \log^4 q)$.
- We set the modulus q so that the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption holds with parameters $(n, m, q, \beta, s_0, \ell)$, where $\beta \geq 2Bm\sqrt{m'} \log n = \mathsf{poly}(\lambda, \ell)$. Note that this also requires that $\mathsf{SIS}_{n,m,q,\beta}$ hold. For instance, we set $q = \beta \cdot \mathsf{poly}(n)$. Then, $\log q = \mathsf{poly}(\log\lambda, \log\ell)$.

Taken together and in conjunction with Remarks 4.15, 4.19, and 4.20, we obtain the following corollary:

Corollary 4.21 (Polynomial Commitments from $\mathsf{BASIS}_{\mathsf{struct}}$)**.** *Let λ be a security parameter. Then, for all polynomials $\ell = \ell(\lambda)$, and under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with a norm bound $\beta = \mathsf{poly}(\lambda, \ell)$ and modulus $q = \mathsf{poly}(\lambda, \ell)$, we have the following vector commitment schemes:*

- *A vector commitment scheme over $\mathbb{Z}_q^\ell$ that supports opening to arbitrary linear functions $\mathbf{x} \mapsto \mathbf{z}^\intercal\mathbf{x}$ for all $\mathbf{z} \in \mathbb{Z}_q^\ell$ with commitment and opening size $\mathsf{poly}(\lambda, \log\ell)$.*
- *A polynomial commitment scheme over $\mathbb{Z}_q$ for polynomials of degree up to $\ell - 1$ with commitment and opening size $\mathsf{poly}(\lambda, \log\ell)$.*

Both schemes are computationally binding and support fast verification in the preprocessing *model where the linear function f or the point $x \in \mathbb{Z}_q$ are known in advance (i.e., with preprocessing, the running time of* Verify *is $\mathsf{poly}(\lambda, \log\ell)$). Without preprocessing, the running time of* Verify *is $\mathsf{poly}(\lambda, \ell)$. The size of the CRS in both constructions is $\ell^2 \cdot \mathsf{poly}(\lambda, \log\ell)$.*

4.2 Supporting Private Openings

In this section, we show how to extend our functional commitment scheme (Construction 4.2) to support (statistically) *private* openings. Recall from Definition 4.1 that a functional commitment supports private opening if the commitment σ to an input $\mathbf{x}$ together with an opening π_f with respect to a function f leaks no additional information about $\mathbf{x}$ other than the value $f(\mathbf{x})$. In the context of homomorphic signatures [GVW15], this property is called *context hiding*. We show that the same approach used to achieve context hiding in the setting of homomorphic signatures applies to our setting and yields a succinct functional commitment that supports private openings. However, the transformation does not preserve the binding property on the functional commitments scheme. Nonetheless, we can still show that the scheme satisfies a *weaker* notion of binding called *target binding*, which says that any *honestly-generated* commitment on $\mathbf{x}$ can only be opened to $f(\mathbf{x})$ for any function f. We start by defining this notion and then give our construction below.

Definition 4.22 (Target Binding)**.** Let λ be a security parameter and let $\Pi_{\mathsf{FC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Eval}, \mathsf{Verify})$ be a succinct functional commitment for a family of functions $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ on inputs of length $\ell = \ell(\lambda)$ and computable by Boolean circuits of depth at most $d = d(\lambda)$. For a security parameter λ, we define the target binding game between an adversary $\mathcal{A}$ and a challenger:

1. The challenger starts by sampling $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d)$ and gives crs to $\mathcal{A}$.
2. Algorithm $\mathcal{A}$ outputs an input $\mathbf{x} \in \{0,1\}^\ell$.
3. Algorithm $\mathcal{B}$ computes $\sigma \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ and gives σ to $\mathcal{A}$.
4. Algorithm $\mathcal{A}$ outputs a function $f \in \mathcal{F}_\lambda$ and an opening π.

We say that Π_{FC} satisfies statistical (resp., computational) target binding if for all adversaries $\mathcal{A}$ (resp., efficient adversaries $\mathcal{A}$),

$$\Pr[\mathsf{Verify}(\mathsf{crs}, \sigma, f, 1 - f(\mathbf{x}), \pi) = 1] = \mathsf{negl}(\lambda)$$

in the target binding game.

Construction 4.23 (Succinct Functional Commitment with Private Opening)**.** Let λ be a security parameter and $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of functions $f\colon \{0,1\}^\ell \to \{0,1\}$ where each function $f\colon \{0,1\}^\ell \to \{0,1\}$ is a function on inputs of length $\ell = \ell(\lambda)$ and which can be computed by a Boolean circuit of depth at most $d = d(\lambda)$. Let $\Pi_{\mathsf{FC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Eval}, \mathsf{Verify})$ be the succinct functional commitment scheme from Construction 4.2 for $\mathcal{F}$. Let n, m, q, m', B, s_0, s_1 be the scheme parameters from Construction 4.2. Let $s_2 = s_2(\lambda)$ be an additional Gaussian width parameter. To construct a functional commitment scheme with *private* openings for $\mathcal{F}$, we modify the Setup, Eval, and Verify algorithms in Construction 4.2 as follows (the Commit algorithm is unchanged):

- $\mathsf{Setup}(1^\lambda, 1^\ell, 1^d)$: Sample $\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}$ according to the specification of $\mathsf{Setup}(1^\lambda, 1^\ell, 1^d)$ in Construction 4.2. Then, sample $\mathbf{u} \xleftarrow{R} \mathbb{Z}_q^n$, and output the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u})$.
- $\mathsf{Eval}(\mathsf{crs}, \mathsf{st}, f)$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u})$, a commitment state $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$, a function $f\colon \{0,1\}^\ell \to \{0,1\}$, the evaluation algorithm proceeds as follow:
  1. Compute $\tilde{\mathbf{C}} \leftarrow [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}] \in \mathbb{Z}_q^{n \times \ell m'}$, $\mathbf{H}_{\tilde{\mathbf{C}}, f, \mathbf{x}} \leftarrow \mathsf{EvalFX}(\tilde{\mathbf{C}}, f, \mathbf{x})$, and $\mathbf{V}_f \leftarrow [\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell] \cdot \mathbf{H}_{\tilde{\mathbf{C}}, f, \mathbf{x}}$.
  2. Let $\tilde{\mathbf{C}}_f \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$ and define the matrix $\mathbf{D}_f = [\mathbf{A} \mid \tilde{\mathbf{C}}_f + (f(\mathbf{x}) - 1) \cdot \mathbf{G}] \in \mathbb{Z}_q^{n \times (m+m')}$. Let $\mathbf{R}_f = \begin{bmatrix} -\mathbf{V}_f \\ \mathbf{I}_{m'} \end{bmatrix}$. Sample $\mathbf{v}_f \leftarrow \mathsf{SamplePre}(\mathbf{D}_f, \mathbf{R}_f, \mathbf{u}, s_2)$ and output the opening $\pi = \mathbf{v}_f$.
- $\mathsf{Verify}(\mathsf{crs}, \sigma, f, y, \pi)$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u})$, a commitment $\sigma = \mathbf{C} \in \mathbb{Z}_q^{n \times m'}$, a function $f\colon \{0,1\}^\ell \to \{0,1\}$, a value $y \in \{0,1\}$, and an opening $\pi = \mathbf{v}_f \in \mathbb{Z}_q^{m+m'}$, the verification algorithm computes $\tilde{\mathbf{C}} \leftarrow [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}] \in \mathbb{Z}_q^{n \times \ell m'}$. Let $\tilde{\mathbf{C}}_f \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$ and output 1 if

$$\|\mathbf{v}_f\| \le B \quad \text{and} \quad [\mathbf{A} \mid \tilde{\mathbf{C}}_f + (y - 1) \cdot \mathbf{G}]\mathbf{v}_f = \mathbf{u}.$$

Theorem 4.24 (Correctness)**.** *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, q is prime, $s_0 \ge O(\ell m^2 \log(n\ell))$, $s_1 \ge O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$, $s_2 \ge s_1 \cdot m^{3/2} \ell^{1/2} \cdot (n \log q)^{O(d)}$, and $B \ge s_2 \cdot \sqrt{m + m'}$. Then, Construction 4.23 is correct.*

Proof. Take a security parameter λ, a function $f \in \mathcal{F}_\lambda$, and an input $\mathbf{x} \in \{0,1\}^\ell$. Let $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d)$ and $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ where $\sigma = \mathbf{C} = \mathbf{G}\hat{\mathbf{C}} = \mathbb{Z}_q^{n \times m'}$ and $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$. Let $\pi = \mathbf{v}_f \leftarrow \mathsf{Eval}(\mathsf{crs}, \mathsf{st}, f)$ and consider $\mathsf{Verify}(\mathsf{crs}, \sigma, f, f(\mathbf{x}), \pi)$. By the same argument as in the proof of Theorem 4.3, the following hold:

- Let $\tilde{\mathbf{C}} = [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$ and $\tilde{\mathbf{V}} = [\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell]$. Then, $\tilde{\mathbf{C}} - \mathbf{x}^\intercal \otimes \mathbf{G} = \mathbf{A}\tilde{\mathbf{V}}$, and $\|\tilde{\mathbf{V}}\| \le B_0 = \sqrt{\ell m + m'} s_1$.
- Let $\mathbf{H}_{\tilde{\mathbf{C}}, f, \mathbf{x}} \leftarrow \mathsf{EvalFX}(\tilde{\mathbf{C}}, f, \mathbf{x})$ and $\mathbf{V}_f \leftarrow \tilde{\mathbf{V}}\mathbf{H}_{\tilde{\mathbf{C}}, f, \mathbf{x}}$. By Theorem 2.15, $\|\mathbf{V}_f\| \le s_1 \cdot \sqrt{\ell m} \cdot (n \log q)^{O(d)}$.

$$\mathbf{A}\mathbf{V}_f = \mathbf{A}\tilde{\mathbf{V}}\mathbf{H}_{\tilde{\mathbf{C}}, f, \mathbf{x}} = \tilde{\mathbf{C}}_f - f(\mathbf{x}) \cdot \mathbf{G},$$

  where $\tilde{\mathbf{C}}_f \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$.

Let $\mathbf{D}_f = [\mathbf{A} \mid \tilde{\mathbf{C}}_f + (f(\mathbf{x}) - 1) \cdot \mathbf{G}] \in \mathbb{Z}_q^{n \times (m+m')}$ and $\mathbf{R}_f = \begin{bmatrix} -\mathbf{V}_f \\ \mathbf{I}_{m'} \end{bmatrix} \in \mathbb{Z}_q^{(m+m') \times m'}$. Then, $\|\mathbf{R}_f\| = \|\mathbf{V}_f\| \le s_1 \cdot \sqrt{\ell m} \cdot (n \log q)^{O(d)}$ and

$$\mathbf{D}_f\mathbf{R}_f = -\mathbf{A}\mathbf{V}_f + \tilde{\mathbf{C}}_f + (f(\mathbf{x}) - 1) \cdot \mathbf{G} = (2f(\mathbf{x}) - 1) \cdot \mathbf{G} \in \{\mathbf{G}, -\mathbf{G}\}.$$

Thus, $\mathbf{R}_f$ is a gadget trapdoor for $\mathbf{D}_f$ (with tag $\mathbf{I}_n$ or $-\mathbf{I}_n$, depending on the value of $f(\mathbf{x}) \in \{0,1\}$). Suppose that $m \ge m' = O(n \log q)$ and

$$s_2 \ge \sqrt{(m + m')m'} \cdot \|\mathbf{R}_f\| \cdot \omega(\sqrt{\log n}) = s_1 \cdot m^{3/2} \ell^{1/2} \cdot (n \log q)^{O(d)}.$$

Since $\mathbf{v}_f \leftarrow \mathsf{SamplePre}(\mathbf{D}_f, \mathbf{R}_f, \mathbf{u}, s_2)$ we appeal to Theorem 2.12 to conclude that $\mathbf{D}_f \mathbf{v}_f = \mathbf{u}$. Since the distribution of $\mathbf{A}$ is statistically close to uniform over $\mathbb{Z}_q^{n \times m}$, we appeal to Lemmas 2.2, 2.9, and 2.10 and Corollary 2.8 and conclude that $\|\mathbf{v}_f\| \le \sqrt{m + m'} \cdot s_2 \le B$. The verification algorithm accepts. □

Theorem 4.25 (Target Binding)**.** *Suppose the conditions on n, m, s_0, s_1 in Theorem 4.24 hold ($n \ge \lambda$, $m \ge O(n \log q)$, $s_0 \ge O(\ell m^2 \log(n\ell))$, $s_1 \ge O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$). Then, under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with parameters $(n, m, q, \beta, s_0, \ell)$, where $\beta = s_1 \cdot m^{3/2} \ell^{1/2} \cdot B \cdot (n \log q)^{O(d)}$, Construction 4.23 satisfies computational target binding.*

Proof. We use a similar hybrid structure as in the proof of Theorem 4.4.

- Hyb_0: This is the real targeted binding experiment:
  - The challenger starts by sampling $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ and $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n \times n}$ for each $i \in [\ell]$. It sets $\tilde{\mathbf{R}}_i \leftarrow \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$ and constructs $\mathbf{B}_\ell$ and $\tilde{\mathbf{R}}$ according to Eq. (4.1). It samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \mathbf{G}_{n\ell}, s_0)$ and $\mathbf{u} \xleftarrow{R} \mathbb{Z}_q^n$. It givens $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u})$ to $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ chooses an input vector $\mathbf{x} \in \{0,1\}^\ell$.
  - The challenger gives $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ to $\mathcal{A}$ where $\sigma = \mathbf{C} \in \mathbb{Z}_q^{n \times m'}$ and $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$.
  - At the end of the experiment, the adversary outputs a function $f \in \mathcal{F}_\lambda$ and an opening $\pi = \mathbf{v}_f$. The output of the experiment is 1 if $\|\mathbf{v}_f\| \le B$ and $[\mathbf{A} \mid \tilde{\mathbf{C}}_f - f(\mathbf{x}) \cdot \mathbf{G}]\mathbf{v}_f = \mathbf{u}$ where $\tilde{\mathbf{C}}_f \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$ and $\tilde{\mathbf{C}} = [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$. Otherwise, the experiments outputs 0.
- Hyb_1: Same as Hyb_0 except after constructing the matrix $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$ according to Eq. (5.1), the challenger samples $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_{s_0}^{-1}(\mathbf{G}_{n\ell})$ *without* using the trapdoor $\tilde{\mathbf{R}}$. The CRS is now sampled independently of $\mathbf{R}$.
- Hyb_2: Same as Hyb_1 except the challenger samples $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$.
- Hyb_3: Same as Hyb_2 except the challenger samples $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$ where $\mathbf{r} \xleftarrow{R} \{0,1\}^m$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output of an execution of $\mathsf{Hyb}_i(\mathcal{A})$ with adversary $\mathcal{A}$. We now show that each adjacent pair of experiments are computationally indistinguishable.

Lemma 4.26. *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, and $s_0 \ge O(\ell m^2 \log(n\ell))$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_0(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A})$.*

Proof. Identical to the proof of Lemma 4.5. □

Lemma 4.27. *Suppose $n \ge \lambda$ and $m = O(n \log q)$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_1(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

Proof. Identical to the proof of Lemma 4.6. □

Lemma 4.28. *Suppose $n \ge \lambda$ and $m \ge 2n \log q$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_2(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_3(\mathcal{A})$.*

Proof. The only difference between Hyb_2 and Hyb_3 is the distribution of $\mathbf{u}$. In Hyb_2, $\mathbf{u} \xleftarrow{R} \mathbb{Z}_q^n$ while in Hyb_3, $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$ where $\mathbf{r} \xleftarrow{R} \{0,1\}^m$. Since $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$ and $m \ge 2n \log q$, the claim follows by the leftover hash lemma (Lemma 2.4). □

Lemma 4.29. *Suppose the conditions on n, m, s_0, s_1 in Theorem 4.24 hold and $m \ge n \log q + \lambda$. Let $\beta = s_1 \cdot m^{3/2} \ell^{1/2} \cdot B \cdot (n \log q)^{O(d)}$. Then, under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with parameters $(n, m, q, \beta, s_0, \ell)$, for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists an efficient adversary $\mathcal{A}$ where $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \varepsilon$ for some non-negligible ε. We use $\mathcal{A}$ to construct an efficient adversary $\mathcal{B}$ for the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption:

1. Algorithm $\mathcal{B}$ receives a challenge $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$, and $\mathbf{T} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$, and $\mathsf{aux} = (\mathbf{W}_1, \ldots, \mathbf{W}_\ell)$. Algorithm $\mathcal{B}$ samples $\mathbf{r} \xleftarrow{R} \{0,1\}^m$, computes $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}$, and gives $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u})$ to $\mathcal{A}$.

2. Algorithm $\mathcal{A}$ outputs a vector $\mathbf{x} \in \mathbb{Z}_q^\ell$.

3. Algorithm $\mathcal{B}$ constructs the matrix $\mathbf{U}_\mathbf{x} \in \mathbb{Z}_q^{n\ell \times m'}$ according to Eq. (4.2) and samples $(\mathbf{V}_1, \ldots, \mathbf{V}_\ell, \hat{\mathbf{C}})$ according to Eq. (4.3). It gives $\sigma = \mathbf{C} = \mathbf{G}\hat{\mathbf{C}} \in \mathbb{Z}_q^{n \times m'}$ and the state $\mathsf{st} = (\mathbf{x}, \mathbf{C}, \mathbf{V}_1, \ldots, \mathbf{V}_\ell)$ to $\mathcal{A}$.

4. Algorithm $\mathcal{A}$ outputs a function $f \in \mathcal{F}_\lambda$ and an opening $\mathbf{v}_f \in \mathbb{Z}_q^{m+m'}$.

5. Algorithm $\mathcal{B}$ outputs the vector $\mathbf{z} = [\mathbf{I}_n \mid \mathbf{V}_f]\mathbf{v}_f - \mathbf{r}$, where $\mathbf{V}_f = [\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell] \cdot \mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}}$, $\mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} \leftarrow \mathsf{EvalFX}(\tilde{\mathbf{C}}, f, \mathbf{x})$, and $\tilde{\mathbf{C}} = [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$.

By construction, algorithm $\mathcal{B}$ perfectly simulates the common reference string crs and the commitment (σ, st) according to the specification of Hyb_3. Thus, with probability ε, algorithm $\mathcal{A}$ outputs $(f, \mathbf{v}_f)$ where $\|\mathbf{v}_f\| \le B$ and $[\mathbf{A} \mid \tilde{\mathbf{C}}_f - f(\mathbf{x}) \cdot \mathbf{G}]\mathbf{v}_f = \mathbf{u}$. Since $(\mathbf{V}_1, \ldots, \mathbf{V}_\ell, \hat{\mathbf{C}})$ are sampled using the Commit algorithm, we can appeal to the analysis of Theorem 4.24 and conclude that with overwhelming probability, $\|\mathbf{V}_f\| \le s_1 \cdot (n \log q)^{O(d)}$ and $\mathbf{A}\mathbf{V}_f = \tilde{\mathbf{C}}_f - f(\mathbf{x}) \cdot \mathbf{G}$ where $\tilde{\mathbf{C}} \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$. This means that

$$\mathbf{u} = [\mathbf{A} \mid \tilde{\mathbf{C}}_f - f(\mathbf{x}) \cdot \mathbf{G}]\mathbf{v}_f = [\mathbf{A} \mid \mathbf{A}\mathbf{V}_f]\mathbf{v}_f = \mathbf{A}[\mathbf{I}_n \mid \mathbf{V}_f]\mathbf{v}_f.$$

Since $\mathbf{u} = \mathbf{A}\mathbf{r}$, this means that $\mathbf{A}\mathbf{z} = \mathbf{A}\big([\mathbf{I}_n \mid \mathbf{V}_f]\mathbf{v}_f - \mathbf{r}\big) = \mathbf{0}$. To conclude the proof, we argue that $0 < \|\mathbf{z}\| \le \beta$:

- Since $\|\mathbf{V}_f\| \le s_1 \cdot \sqrt{\ell m} \cdot (n \log q)^{O(d)}$, $\|\mathbf{v}_f\| \le B$, $\|\mathbf{r}\| = 1$, and $m > m'$, we have that

$$\|\mathbf{z}\| \le s_1 \cdot \sqrt{\ell m} \cdot (n \log q)^{O(d)} B(m + m') + 1 \le s_1 \cdot m^{3/2} \ell^{1/2} \cdot B \cdot (n \log q)^{O(d)} = \beta.$$

- It suffices to show that $\mathbf{z} \neq \mathbf{0}$, or equivalently, that $\mathbf{r} \neq [\mathbf{I}_n \mid \mathbf{V}_f]\mathbf{v}_f$. Here, we can appeal to the same entropy argument as in [GVW15, Theorem 3.1]. By construction, $\mathbf{V}_f$ and $\mathbf{v}_f$ are functions of $\mathbf{u} \in \mathbb{Z}_q^n$ (and other quantities that are independent of $\mathbf{r}$). By construction, $\mathbf{u}$ contains at most $n \log q$ bits of information about $\mathbf{r}$. This means that

$$\mathbf{H}_\infty(\mathbf{r} \mid \mathbf{V}_f, \mathbf{v}_f) \ge \mathbf{H}_\infty(\mathbf{r} \mid \mathbf{u}) \ge m - n \log q \ge \lambda.$$

  This means that $\Pr[\mathbf{r} = [\mathbf{I}_n \mid \mathbf{V}_f]\mathbf{v}_f] \le 2^{-\lambda}$, and so $\mathcal{B}$ breaks the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with advantage $\varepsilon - 2^{-\lambda}$. □

Combining Lemmas 4.26 to 4.29, Construction 4.23 satisfies computational target binding. □

Theorem 4.30 (Private Opening)**.** *Suppose the conditions of Theorem 4.24 hold: namely, $n \ge \lambda$, $m \ge O(n \log q)$, $s_0 \ge O(\ell m^2 \log(n\ell))$, $s_1 \ge O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$, and $s_2 \ge s_1 \cdot m^{3/2} \ell^{1/2} \cdot (n \log q)^{O(d)}$. Suppose also that q is prime. Then, Construction 4.23 provides statistical private openings.*

Proof. We construct an efficient simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ as follows:

- $\mathcal{S}_0(1^\lambda, 1^\ell, 1^d)$: On input the security parameter λ, the input dimension ℓ and the depth bound d, the simulator samples $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ and an invertible matrix $\mathbf{W}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times n}$ for each $i \in [\ell]$. Then, it constructs the matrix $\mathbf{B}_\ell$ and trapdoor $\tilde{\mathbf{R}}$ according to Eq. (4.1). Next, it samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \mathbf{G}_{n\ell}, s_0)$ and $\mathbf{u} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$. It samples $\hat{\mathbf{C}} \xleftarrow{\mathrm{R}} D_{s_1}^{m' \times m'}$, computes $\mathbf{C} \leftarrow \mathbf{G}\hat{\mathbf{C}}$, and outputs $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u})$, the commitment $\sigma = \mathbf{C}$, and the state $\mathsf{st}_\mathcal{S} = (\mathbf{A}, \mathbf{R}, \mathbf{C}, \mathbf{u})$.

- $\mathcal{S}_1(\mathsf{st}_\mathcal{S}, f, y)$: On input the simulation state $\mathsf{st} = (\mathbf{A}, \mathbf{R}, \mathbf{C}, \mathbf{u})$, a function $f \in \mathcal{F}_\lambda$, and a value $y \in \{0, 1\}$, the simulator first computes $\tilde{\mathbf{C}} \leftarrow [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$ and $\tilde{\mathbf{C}}_f \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$. It defines the matrix $\mathbf{D}_f = [\mathbf{A} \mid \tilde{\mathbf{C}}_f + (y-1)\mathbf{G}]$ and $\mathbf{R}_f = \left[\begin{smallmatrix} \mathbf{R} \\ \mathbf{0} \end{smallmatrix}\right]$ and outputs $\mathbf{v}_f \leftarrow \mathsf{SamplePre}(\mathbf{D}_f, \mathbf{R}_f, \mathbf{u}, s_2)$.

To complete the proof, it suffices to show that the real distribution and the simulated distribution are statistically indistinguishable. We proceed with a hybrid argument:

- Hyb_0: This is the real distribution:

1. The challenger starts by sampling $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ and $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n \times n}$ for each $i \in [\ell]$. It sets $\tilde{\mathbf{R}}_i \leftarrow \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$ and constructs $\mathbf{B}_\ell$ and $\tilde{\mathbf{R}}$ according to Eq. (4.1). It samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \mathbf{G}_{n\ell}, s_0)$ and $\mathbf{u} \xleftarrow{R} \mathbb{Z}_q^n$. It givens $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u})$ to $\mathcal{A}$.
2. Algorithm $\mathcal{A}$ chooses an input vector $\mathbf{x} \in \{0,1\}^\ell$.
3. The challenger constructs the target matrix $\mathbf{U}_\mathbf{x}$ according to Eq. (4.2) and samples $(\mathbf{V}_1, \ldots, \mathbf{V}_\ell, \hat{\mathbf{C}})$ using $\mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{T}, \mathbf{U}_\mathbf{x}, s_1)$. It gives the commitment $\sigma = \mathbf{C} = \mathbf{G}\hat{\mathbf{C}} \in \mathbb{Z}_q^{n \times m'}$ to $\mathcal{A}$.
4. Adversary $\mathcal{A}$ outputs a function $f \in \mathcal{F}_\lambda$.
5. The challenger now computes $\tilde{\mathbf{C}} \leftarrow [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$, $\mathbf{H}_{\tilde{\mathbf{C}},f,\mathbf{x}} \leftarrow \mathsf{EvalFX}(\tilde{\mathbf{C}}, f, \mathbf{x})$, and $\mathbf{V}_f \leftarrow [\mathbf{V}_1 \mid \cdots \mid \mathbf{V}_\ell]$. Next, it computes $\tilde{\mathbf{C}}_f \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f)$, $\mathbf{D}_f = [\mathbf{A} \mid \tilde{\mathbf{C}}_f + (f(\mathbf{x}) - 1) \cdot \mathbf{G}]$, $\mathbf{R}_f = \begin{bmatrix} -\mathbf{V}_f \\ \mathbf{I}_{m'} \end{bmatrix}$, and samples $\mathbf{v}_f \leftarrow \mathsf{SamplePre}(\mathbf{D}_f, \mathbf{R}_f, \mathbf{u}, s_2)$. It replies to $\mathcal{A}$ with $\mathbf{v}_f$.
6. Finally, algorithm $\mathcal{A}$ outputs a bit $b \in \{0,1\}$, which is also the output of the experiment.

- Hyb_1: Same as Hyb_0, except when simulating the opening, the challenger instead sets $\mathbf{R}_f = \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}$ and samples $\mathbf{v}_f \leftarrow \mathsf{SamplePre}(\mathbf{D}_f, \mathbf{R}_f, \mathbf{u}, s_2)$.
- Hyb_2: Same as Hyb_1 except the challenger samples $\hat{\mathbf{C}} \leftarrow D_{\mathbb{Z}, s_1}^{m' \times m'}$. This is the simulated distribution.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output of an execution of $\mathsf{Hyb}_i(\mathcal{A})$ with adversary $\mathcal{A}$. We now show that each adjacent pair of experiments are statistically indistinguishable.

Lemma 4.31. *Suppose the conditions of Theorem 4.24 hold. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_0(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A})$.*

Proof. The only difference between the Hyb_0 and Hyb_1 is the distribution of $\mathbf{v}_f$:

- Consider the distribution of $\mathbf{v}_f$ in Hyb_0. By the same analysis as in the proof of Theorem 4.24, $\mathbf{D}_f\mathbf{R}_f \in \{\mathbf{G}, -\mathbf{G}\}$ where $\|\mathbf{R}_f\| \le s_1 \cdot \sqrt{\ell m} \cdot (n \log q)^{O(d)}$. Moreover, if $s_2 \ge \sqrt{(m+m')m'}\|\mathbf{R}_f\| \cdot \omega(\sqrt{\log n}) = s_1 \cdot m^{3/2}\ell^{1/2} \cdot (n \log q)^{O(d)}$, then by Theorem 2.12, the distribution of $\mathbf{v}_f \leftarrow \mathsf{SamplePre}(\mathbf{D}_f, \mathbf{R}_f, \mathbf{u}, s_2)$ is statistically close to the distribution $\mathbf{v}_f \leftarrow (\mathbf{D}_f)_{s_2}^{-1}(\mathbf{u})$.
- In Hyb_1, $\mathbf{D}_f\mathbf{R}_f = \mathbf{A}\mathbf{R} = \mathbf{G}$ and $\|\mathbf{R}_f\| = \|\mathbf{R}\| = 1$. As long as $s_2 \ge \sqrt{(m+m')m'}\|\mathbf{R}_f\| \cdot \omega(\sqrt{\log n}) = O(m \log n)$, the distribution of $\mathbf{v}_f \leftarrow \mathsf{SamplePre}(\mathbf{D}_f, \mathbf{R}_f, \mathbf{u}, s_2)$ is statistically close to the distribution of $\mathbf{v}_f \leftarrow (\mathbf{D}_f)_{s_2}^{-1}(\mathbf{u})$

In both cases, the distribution of $\mathbf{v}_f$ is statistically close to $(\mathbf{D}_f)_{s_2}^{-1}(\mathbf{u})$ and the claim holds. □

Lemma 4.32. *Suppose the conditions of Theorem 4.24 hold. Then for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_1(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

Proof. The only difference between Hyb_1 and Hyb_2 is the distribution of $\hat{\mathbf{C}}$:

- In Hyb_1, the challenger samples

$$\begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_\ell \\ \hat{\mathbf{C}} \end{bmatrix} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{T}, \mathbf{U}_\mathbf{x}, s_1),$$

  where $\mathbf{B}_\ell$ is the matrix from Eq. (4.1). By the same analysis as in the proof of Theorem 4.3, we have that $\mathbf{B}_\ell\mathbf{T} = \mathbf{G}$ and $\|\mathbf{T}\| \le \sqrt{\ell m + m'} s_0$. If

$$s_1 \ge \sqrt{(\ell m + m')\ell m'}\,\|\mathbf{T}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0),$$

  we can appeal to Theorem 2.12 to conclude that the distribution of $(\mathbf{V}_1, \ldots, \mathbf{V}_\ell, \hat{\mathbf{C}})$ is statistically close to $(\mathbf{B}_\ell)_{s_1}^{-1}(\mathbf{U}_\mathbf{x})$.

Next, let $\bar{\mathbf{A}} = \mathsf{diag}(\mathbf{W}_1\mathbf{A}, \ldots, \mathbf{W}_\ell\mathbf{A})$. Then, $\mathbf{B}_\ell = [\bar{\mathbf{A}} \mid -\mathbf{1}^\ell \otimes \mathbf{G}]$. By Theorem 2.12, the distribution of $\mathbf{A}$ is statistically close to uniform over $\mathbb{Z}_q^{n\times m}$. For each $i \in [\ell]$, the matrix $\mathbf{W}_i$ is invertible so the distribution of each $\mathbf{W}_i\mathbf{A}$ remains statistically close to uniform over $\mathbb{Z}_q^{n\times m}$. By Lemma 2.2 and Corollary 2.5, this means that with overwhelming probability, $\mathbf{W}_i\mathbf{A}$ is full rank and $\lambda_1^\infty(\Lambda(\mathbf{W}_i\mathbf{A})) \geq q/4$. By a union bound over all $i \in [\ell]$, this holds for all $i \in [\ell]$. Since $s_1 \geq 4\log(\ell m)$, we can now appeal to Corollary 2.11 to conclude that the distribution of $(\mathbf{V}_1, \ldots, \mathbf{V}_\ell, \hat{\mathbf{C}}) \leftarrow (\mathbf{B}_\ell)_{s_1}^{-1}(\mathbf{U}_\mathbf{x})$ is statistically close to the distribution

$$\left\{\hat{\mathbf{C}} \leftarrow D_{\mathbb{Z},s_1}^{m'\times m'},\ (\mathbf{V}_1, \ldots, \mathbf{V}_\ell) \leftarrow \bar{\mathbf{A}}_{s_1}^{-1}(\mathbf{U}_\mathbf{x} + (\mathbf{1}^\ell \otimes \mathbf{G}\hat{\mathbf{C}}))\right\}.$$

- In Hyb_2, the challenger samples $\hat{\mathbf{C}} \leftarrow D_{\mathbb{Z},s_1}^{m'\times m'}$.

In both experiments, the marginal distribution of $\hat{\mathbf{C}}$ is statistically close to the distribution $D_{\mathbb{Z},s_1}^{m'\times m'}$. □

Statistical private openings now follows from Lemmas 4.31 and 4.32. □

Parameter instantiation. Let λ be a security parameter and $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of functions $f\colon \{0,1\}^\ell \to \{0,1\}$ on inputs of length $\ell = \ell(\lambda)$ and which can be computed by Boolean circuits of depth at most $d = d(\lambda)$. We instantiate the lattice parameters in Construction 4.23 as follows to satisfy Theorems 4.24, 4.25, and 4.30. The parameter instantiations essentially match those of the non-private construction (Construction 4.2):

- Let $\varepsilon > 0$ be a constant. We set the lattice dimension $n = d^{1/\varepsilon} \cdot \mathsf{poly}(\lambda)$ and $m = O(n\log q)$.
- We set $s_0 = O(\ell m^2 \log(n\ell))$, $s_1 = O(\ell^{3/2} m^{3/2}\log(n\ell) \cdot s_0) = O(\ell^{5/2} m^{7/2}\log^2(n\ell))$, and $s_2 \geq s_1 \cdot m^{3/2}\ell^{1/2} \cdot (n\log q)^{O(d)} = \ell^3 \log^2 \ell \cdot (n\log q)^{O(d)}$.
- We set the bound $B = s_2 \cdot \sqrt{m+m'} = \ell^3 \log^2 \ell \cdot (n\log q)^{O(d)}$.
- We choose the modulus q so that the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption holds with parameters $(n, m, q, \beta, s_0, \ell)$, where

$$\beta = s_1 \cdot m^{3/2}\ell^{1/2} \cdot B \cdot (n\log q)^{O(d)} = 2^{\tilde{O}(d)} = 2^{\tilde{O}(n^\varepsilon)},$$

  where we write $\tilde{O}(\cdot)$ to suppress polylogarithmic factors in λ, d, ℓ. Note that this also requires that $\mathsf{SIS}_{n,m,q,\beta}$ hold. For instance, we set $q = \beta \cdot \mathsf{poly}(n)$. Then, $\log q = \mathsf{poly}(d, \log\lambda, \log\ell)$. Note that the underlying SIS assumption relies on a *sub-exponential* noise bound.

With this setting of parameters, we obtain a functional commitment scheme with private openings for $\mathcal{F}$ with similar properties as the non-hiding variant (Corollary 4.8):

Corollary 4.33 (Succinct Functional Commitment with Private Opening from $\mathsf{BASIS}_{\mathsf{struct}}$)**.** *Let λ be a security parameter, and let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of functions $f\colon \{0,1\}^\ell \to \{0,1\}$ on inputs of length $\ell = \ell(\lambda)$ and which can be computed by Boolean circuits of depth at most $d = d(\lambda)$. Under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with a norm bound $\beta = 2^{\tilde{O}(d)}$ and modulus $q = 2^{\tilde{O}(d)}$, there exists a succinct functional commitment scheme for $\mathcal{F}$ that satisfies computational target binding and has statistically private openings. Both the size of the commitment and the opening are $\mathsf{poly}(\lambda, d, \log\ell)$ and the CRS has size $\ell^2 \cdot \mathsf{poly}(\lambda, d, \log\ell)$. Here, $\tilde{O}(\cdot)$ suppresses polylogarithmic factors in λ, d, and ℓ.*

5 Aggregatable Vector and Functional Commitments

In this section, we describe a variant of our SIS-based vector commitment (Construction 3.9) that supports aggregation. The same techniques also applies to our succinct functional commitment scheme (Construction 4.2) and yields an aggregatable functional commitment (see Section 5.1). In an aggregatable commitment, one can take a collection of openings $\{(i, \pi_i)\}_{i\in S}$ for a set of $S \subseteq [\ell]$ of indices and aggregate them into a *single* opening π (for the set S) whose size scales sublinearly with the size of S. Aggregatable commitments imply subvector commitments [LM19] which are vector commitments that allow for succinct openings to a set of indices (but do *not* necessarily support aggregating openings). We start by defining the notion of an aggregatable vector commitment:

Definition 5.1 (Aggregatable Vector Commitment)**.** An aggregatable vector commitment over a message space $\mathcal{M}$ is a tuple of efficient algorithms $\Pi_{\mathsf{AVC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify}, \mathsf{Aggregate}, \mathsf{VerifyAgg})$ where $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ is a standard vector commitment scheme and the additional functions $(\mathsf{Aggregate}, \mathsf{VerifyAgg})$ satisfy the following properties:

- $\mathsf{Aggregate}(\mathsf{crs}, \sigma, \{(i, x_i, \pi_i)\}_{i \in S}) \to \pi_S$: On input the common reference string crs, a commitment σ, a set of openings π_i to $x_i \in \mathcal{M}$ to indices $i \in S$, the aggregation algorithm outputs an aggregated opening π_S.
- $\mathsf{VerifyAgg}(\mathsf{crs}, \sigma, S, \{(i, x_i)\}_{i \in S}, \pi) \to \{0, 1\}$: On input the common reference string crs, a commitment σ, a set $S \subseteq [\ell]$, and values $x_i \in \mathcal{M}$ for $i \in S$, the aggregate verification algorithm outputs a bit $b \in \{0, 1\}$.

In addition to the basic properties of a vector commitment scheme, an aggregatable vector commitment scheme should satisfy the following additional properties:

- **Correctness of aggregation:** For all security parameters $\lambda \in \mathbb{N}$, vector lengths $\ell \in \mathbb{N}$, and sampling $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$, and for all sets $S \subseteq [\ell]$, commitments σ, and openings $\{(i, x_i, \pi_i)\}_{i \in S}$ where $\mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i, \pi_i) = 1$ for all $i \in S$, it holds that

$$\Pr[\mathsf{VerifyAgg}(\mathsf{crs}, \sigma, \{(i, x_i)\}_{i \in S}, \pi') : \pi' \leftarrow \mathsf{Aggregate}(\mathsf{crs}, \sigma, \{(i, x_i, \pi_i)\}_{i \in S})] = 1 - \mathsf{negl}(\lambda),$$

  where the probability is taken over the randomness of Setup.

- **Succinctness:** The aggregatable vector commitment scheme is succinct if there exists a universal polynomial $\mathsf{poly}(\cdot, \cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\pi'| = \mathsf{poly}(\lambda, \log \ell)$ in the correctness of aggregation definition.

- **Same-set binding:** We say Π_{AVC} satisfies statistical (resp., computational) same-set binding if for all polynomials $\ell = \ell(\lambda)$ and all adversaries $\mathcal{A}$ (resp., efficient adversaries $\mathcal{A}$),

$$\Pr\left[\begin{array}{c} \mathsf{VerifyAgg}(\mathsf{crs}, \sigma, S, \{(i, x_i)\}_{i \in S}, \pi) = 1 \\ \text{and } x_i \neq x_i' \text{ for some } i \in S \text{ and} \\ \mathsf{VerifyAgg}(\mathsf{crs}, \sigma, S, \{(i, x_i')\}_{i \in S}, \pi') = 1 \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell); \\ (\sigma, S, \{(i, x_i, x_i')\}_{i \in S}, \pi, \pi') \leftarrow \mathcal{A}(1^\lambda, 1^\ell, \mathsf{crs}) \end{array}\right] = \mathsf{negl}(\lambda).$$

Remark 5.2 (Different-Set Binding)**.** We can also define a stronger notion of security for aggregatable vector commitments where we allow the adversary to output two *different* sets S and T along with openings π_S and π_T to $\{(i, x_i)\}_{i \in S}$ and $\{(i, x_i')\}_{i \in T}$, respectively, in the binding game. The adversary wins if there exists some index $i \in S \cap T$ such that $x_i \neq x_i'$. Namely, we say an aggregatable vector commitment scheme $\Pi_{\mathsf{AVC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify}, \mathsf{Aggregate}, \mathsf{VerifyAgg})$ satisfies statistical (resp., computational) *different-set binding* if for all adversaries (resp., efficient adversaries) $\mathcal{A}$,

$$\Pr\left[\begin{array}{c} \mathsf{VerifyAgg}(\mathsf{crs}, \sigma, S, \{(i, x_i)\}_{i \in S}, \pi_S) = 1 \\ \text{and } x_i \neq x_i' \text{ for some } i \in S \cap T \text{ and} \\ \mathsf{VerifyAgg}(\mathsf{crs}, \sigma, T, \{(i, x_i')\}_{i \in T}, \pi_T) = 1 \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell); \\ (\sigma, S, T, \{(i, x_i)\}_{i \in S}, \{(i, x_i')\}_{i \in T}, \pi_S, \pi_T) \leftarrow \mathcal{A}(1^\lambda, 1^\ell, \mathsf{crs}) \end{array}\right] = \mathsf{negl}(\lambda).$$

Different-set binding effectively says that the commitment σ fully defines the value of x_i for all $i \in [\ell]$. In contrast, if the vector commitment scheme only satisfies same-set binding, then an adversary may be able to open a commitment σ to $x_i = 0$ with respect to a set S and to $x_i = 1$ with respect to a different set T. The vector commitments we construct in this work satisfy the weaker notion of same-set binding (Theorem 5.6), but do not satisfy different-set binding (Remark 5.12). Note though that same-set binding implies different-set binding in the setting where the commitments are generated honestly (Remark 5.13). It is an interesting open problem to construct a lattice-based vector commitment scheme that satisfies the stronger notion of different-set binding.

Construction 5.3 (Aggregatable Vector Commitments)**.** Let λ be a security parameter and $n = n(\lambda)$, $m = m(\lambda)$, and $q = q(\lambda)$ be lattice parameters. Let $m' = n(\lceil \log q \rceil + 1)$ and $B = B(\lambda)$ be a bound. Let $s_0 = s_0(\lambda)$, $s_1 = s_1(\lambda)$ be Gaussian width parameters. Let $\ell = \ell(\lambda)$ be the input dimension. Let $p = p(\lambda)$ be the message-space modulus. We construct an aggregatable vector commitment scheme $\Pi_{\mathsf{VC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify})$ for $\mathbb{Z}_p^\ell$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^\ell)$: On input the security parameter λ and the input length ℓ, the setup algorithm samples $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$. Then, for each $i \in [\ell]$, it samples an invertible matrix $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n\times n}$ and a target vector $\mathbf{u}_i \xleftarrow{R} \mathbb{Z}_q^n$. Next, it computes $\tilde{\mathbf{R}}_i \leftarrow \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G}) \in \mathbb{Z}_q^{m\times m'}$ for each $i \in [\ell]$ and constructs $\mathbf{B}_\ell$ and $\mathbf{R}$ as follows:

$$\mathbf{B}_\ell = \left[\begin{array}{ccc|c} \mathbf{W}_1\mathbf{A} & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{W}_\ell\mathbf{A} & -\mathbf{G} \end{array}\right] \in \mathbb{Z}_q^{n\ell\times(\ell m+m')} \quad \text{and} \quad \tilde{\mathbf{R}} = \left[\begin{array}{c} \mathrm{diag}(\tilde{\mathbf{R}}_1, \ldots, \tilde{\mathbf{R}}_\ell) \\ \hline \mathbf{0}^{n\times\ell m'} \end{array}\right] \in \mathbb{Z}_q^{(\ell m+m')\times\ell m'}. \tag{5.1}$$

  Finally, the setup algorithm samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \mathbf{G}_{n\ell}, s_0)$ and outputs the common reference string $\mathsf{crs} = \big(\mathbf{A}, \{\mathbf{W}_i\}_{i\in[\ell]}, \{\mathbf{u}_i\}_{i\in[\ell]}, \mathbf{T}\big)$.

- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x})$: On input the common reference string $\mathsf{crs} = \big(\mathbf{A}, \{\mathbf{W}_i\}_{i\in[\ell]}, \{\mathbf{u}_i\}_{i\in[\ell]}, \mathbf{T}\big)$ and a vector $\mathbf{x} \in \mathbb{Z}_p^\ell$, the commit algorithm constructs $\mathbf{B}_\ell$ from $\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{A}_\ell$ according to Eq. (5.1). It then constructs the target vector

$$\hat{\mathbf{u}} = \begin{bmatrix} -x_1\mathbf{W}_1\mathbf{u}_1 \\ \vdots \\ -x_\ell\mathbf{W}_\ell\mathbf{u}_\ell \end{bmatrix}$$

  and uses $\mathbf{T}$ to sample a preimage

$$\begin{bmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_\ell \\ \hat{\mathbf{c}} \end{bmatrix} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{T}, \hat{\mathbf{u}}, s_1). \tag{5.2}$$

  It computes $\mathbf{c} \leftarrow \mathbf{G}\hat{\mathbf{c}} \in \mathbb{Z}_q^n$ and outputs the commitment $\sigma = \mathbf{c}$ and the state $\mathsf{st} = (\mathbf{v}_1, \ldots, \mathbf{v}_\ell)$.

- $\mathsf{Open}(\mathsf{st}, i)$: On input the state $\mathsf{st} = (\mathbf{v}_1, \ldots, \mathbf{v}_\ell)$ and an index $i \in [\ell]$, the opening algorithm outputs $\pi = \mathbf{v}_i$.

- $\mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i, \pi)$: On input the common reference string $\mathsf{crs} = \big(\mathbf{A}, \{\mathbf{W}_i\}_{i\in[\ell]}, \{\mathbf{u}_i\}_{i\in[\ell]}, \mathbf{T}\big)$, a commitment $\sigma = \mathbf{c}$, an index $i \in [\ell]$, a value $x_i \in \mathbb{Z}_p$, and an opening $\pi = \mathbf{v}$, the verification algorithm outputs 1 if

$$\|\mathbf{v}\| \le B \quad \text{and} \quad \mathbf{W}_i^{-1}\mathbf{c} = \mathbf{A}\mathbf{v} + x_i\mathbf{u}_i.$$

We now define the aggregation algorithm for openings and the verification algorithm for a set of indices $S \subseteq [\ell]$:

- $\mathsf{Aggregate}\big(\mathsf{crs}, \sigma, \{(i, x_i, \pi_i)\}_{i\in S}\big)$: On input the common reference string $\mathsf{crs} = \big(\mathbf{A}, \{\mathbf{W}_i\}_{i\in[\ell]}, \{\mathbf{u}_i\}_{i\in[\ell]}, \mathbf{T}\big)$, a commitment $\sigma = \mathbf{c}$, and a set S of openings (i, x_i, π_i), where $\pi_i = \mathbf{v}_i \in \mathbb{Z}_q^m$, the aggregation algorithm outputs $\pi = \mathbf{v} = \sum_{i\in S} \mathbf{v}_i$.

- $\mathsf{VerifyAgg}(\mathsf{crs}, \sigma, S, \{(i, x_i)\}_{i\in S}, \pi)$: On input the common reference string $\mathsf{crs} = \big(\mathbf{A}, \{\mathbf{W}_i\}_{i\in[\ell]}, \{\mathbf{u}_i\}_{i\in[\ell]}, \mathbf{T}\big)$, a commitment $\sigma = \mathbf{c}$, a set of indices $S \subseteq [\ell]$, values $x_i \in \mathbb{Z}_p$ for each $i \in S$, and an opening $\pi = \mathbf{v}$, the aggregate verification algorithm outputs 1 if

$$\|\mathbf{v}\| \le |S| \cdot B \quad \text{and} \quad \sum_{i\in S} \mathbf{W}_i^{-1}\mathbf{c} = \mathbf{A}\mathbf{v} + \sum_{i\in S} x_i\mathbf{u}_i.$$

Theorem 5.4 (Correctness). *Suppose $n \ge \lambda$, $m \ge O(n \log q)$, $s_0 \ge O(\ell m^2 \log(n\ell))$, $s_1 \ge O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0)$, and $B \ge \sqrt{\ell m + m'} \cdot s_1$. Then, Construction 5.3 is correct.*

Proof. The proof is nearly identical to that of the proof of Theorem 3.10. Take any polynomial $\ell = \ell(\lambda)$, any vector $\mathbf{x} \in \mathbb{Z}_p^\ell$, and any index $i \in [\ell]$. Let $\mathsf{crs} = \big(\mathbf{A}, \{\mathbf{W}_i\}_{i\in[\ell]}, \{\mathbf{u}_i\}_{i\in[\ell]}, \mathbf{T}\big) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$. Let $(\sigma, \mathsf{st}) \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$ where $\sigma = \mathbf{c} = \mathbf{G}\hat{\mathbf{c}} \in \mathbb{Z}_q^n$ and $\mathsf{st} = (\mathbf{v}_1, \ldots, \mathbf{v}_\ell)$. Let $\pi = \mathbf{v}_i \leftarrow \mathsf{Open}(\mathsf{st}, i)$ and consider $\mathsf{Verify}(\mathsf{crs}, \sigma, i, x_i, \pi)$:

- Let $\mathbf{B}_\ell \in \mathbb{Z}_q^{(\ell m+m')\times \ell m'}$ and $\tilde{\mathbf{R}} \in \mathbb{Z}_q^{(\ell m+m')\times \ell m'}$ be the matrices from Eq. (5.1). For each $i \in [\ell]$, we have that $\tilde{\mathbf{R}}_i = \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$, so
$$\mathbf{W}_i\mathbf{A}\tilde{\mathbf{R}}_i = \mathbf{W}_i\mathbf{A}\mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G}) = \mathbf{W}_i\mathbf{G}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G}) = \mathbf{G}.$$
Thus, $\mathbf{B}_\ell\tilde{\mathbf{R}} = \mathbf{G}_{n\ell}$ and $\|\tilde{\mathbf{R}}\| = \max_{i\in[\ell]}\|\tilde{\mathbf{R}}_i\| \le m' = O(n\log q)$ since $\|\mathbf{R}\| = 1 = \|\mathbf{G}^{-1}(\mathbf{W}_i^{-1}(\mathbf{G}))\|$ for all $i \in [\ell]$. Suppose also that $m \ge m' = O(n\log q)$. By Theorem 2.12 and Lemmas 2.6 and 2.9, as long as
$$s_0 \ge \sqrt{(\ell m+m')\ell m'}\|\tilde{\mathbf{R}}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell m^2\log(n\ell)),$$
then $\mathbf{B}_\ell\mathbf{T} = \mathbf{G}_{n\ell}$ and $\|\mathbf{T}\| \le \sqrt{\ell m+m'}s_0$ with overwhelming probability.
- Suppose $s_1 \ge \sqrt{(\ell m+m')\ell m'}\,\|\mathbf{T}\| \cdot \omega(\sqrt{\log(n\ell)}) = O(\ell^{3/2}m^{3/2}\log(n\ell)\cdot s_0)$. Then, by Theorem 2.12 and by construction of $(\mathbf{v}_1,\ldots,\mathbf{v}_\ell,\hat{\mathbf{c}})$ in Eq. (5.2), $\mathbf{W}_i\mathbf{A}\mathbf{v}_i - \mathbf{G}\hat{\mathbf{c}} = -x_i\mathbf{W}_i\mathbf{u}_i$. Since $\mathbf{c} = \mathbf{G}\hat{\mathbf{c}}$, this means that $\mathbf{W}_i^{-1}\mathbf{c} = \mathbf{A}\mathbf{v}_i + x_i\mathbf{u}_i$. Moreover, by Lemmas 2.6 and 2.9, $\|\mathbf{v}_i\| \le \sqrt{\ell m+m'}s_1 \le B$. Thus, $\mathsf{Verify}(\mathsf{crs},\sigma,i,x_i,\pi)$ outputs 1. □

Theorem 5.5 (Correctness of Aggregation)**.** *Construction 5.3 satisfies correctness of aggregation.*

Proof. Take any polynomial $\ell = \ell(\lambda)$ and let $\mathsf{crs} = (\mathbf{A},\{\mathbf{W}_i\}_{i\in[\ell]},\{\mathbf{u}_i\}_{i\in[\ell]},\mathbf{T}) \leftarrow \mathsf{Setup}(1^\lambda,1^\ell)$. Take any set $S \subseteq [\ell]$, commitment $\sigma = \mathbf{c} \in \mathbb{Z}_q^n$, and openings $\{(i,x_i,\pi_i)\}_{i\in S}$ where $\pi_i = \mathbf{v}_i$ and $\mathsf{Verify}(\mathsf{crs},\sigma,i,x_i,\pi_i) = 1$. Let $\mathbf{v} \leftarrow \mathsf{Aggregate}(\mathsf{crs},\sigma,\{(i,x_i,\pi_i)\}_{i\in S})$. Since π_i is a valid opening to σ on index i, we have that $\|\mathbf{v}_i\| \le B$ and moreover, $\mathbf{W}_i^{-1}\mathbf{c} = \mathbf{A}\mathbf{v}_i + x_i\mathbf{u}_i$. By construction, $\mathbf{v} = \sum_{i\in S}\mathbf{v}_i$, so $\|\mathbf{v}\| \le |S|\cdot B$. Moreover,
$$\mathbf{A}\mathbf{v} = \sum_{i\in S}\mathbf{A}\mathbf{v}_i = \sum_{i\in S}\mathbf{W}_i^{-1}\mathbf{c} - \sum_{i\in S}x_i\mathbf{u}_i,$$
and the aggregate verification algorithm accepts. □

Theorem 5.6 (Computational Same-Set Binding)**.** *Let ℓ be the vector dimension. Suppose $n \ge \lambda$, $m \ge O(n\log q)$, q is prime, and $s_0 \ge O(\ell m^2\log(n\ell))$. Then, under the* $\mathsf{BASIS}_{\mathsf{struct}}$ *assumption with parameters $(n,m,q,\ell(2B+p),s_0,\ell)$, Construction 5.3 satisfies computational same-set binding.*

Proof. Let $\ell = \ell(\lambda)$ be a polynomial. We use a nearly identical set of hybrid experiments as in the proof of Theorem 4.4:

- Hyb_0: This is the real same-set binding experiment:
  - The challenger starts by sampling $(\mathbf{A},\mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n,q,m)$, $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n\times n}$, and $\mathbf{u}_i \xleftarrow{R} \mathbb{Z}_q^n$ for each $i \in [\ell]$. It sets $\tilde{\mathbf{R}}_i \leftarrow \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$ and constructs $\mathbf{B}_\ell$ and $\tilde{\mathbf{R}}$ according to Eq. (5.1). It samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell,\tilde{\mathbf{R}},\mathbf{G}_{n\ell},s_0)$ and gives $\mathsf{crs} = (\mathbf{A},\{\mathbf{W}_i\}_{i\in[\ell]},\{\mathbf{u}_i\}_{i\in[\ell]},\mathbf{T})$ to the adversary $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{c} \in \mathbb{Z}_q^n$ a set $S \subseteq [\ell]$, values $\{(i,x_i,x_i')\}_{i\in S}$, and openings $\mathbf{v},\mathbf{v}'$.
  - The output of the experiment is 1 if there exists $i \in S$ such that $x_i \ne x_i'$, $\|\mathbf{v}\|,\|\mathbf{v}'\| \le |S|\cdot B$, $\mathbf{A}\mathbf{v} = \sum_{i\in S}\mathbf{W}_i^{-1}\mathbf{c} - \sum_{i\in S}x_i\mathbf{u}_i$, and $\mathbf{A}\mathbf{v}' = \sum_{i\in S}\mathbf{W}_i^{-1}\mathbf{c} - \sum_{i\in S}x_i'\mathbf{u}_i$.
- Hyb_1: Same as Hyb_0 except after constructing the matrix $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell\times(\ell m+m')}$ according to Eq. (5.1), the challenger samples $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_{s_0}^{-1}(\mathbf{G}_{n\ell})$ *without* using the trapdoor $\tilde{\mathbf{R}}$. The CRS is now sampled independently of $\mathbf{R}$.
- Hyb_2: Same as Hyb_1 except the challenger samples $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n\times m}$.
- Hyb_3: Same as Hyb_2 except for each $i \in [\ell]$, it samples $\mathbf{r}_i \xleftarrow{R} \{0,1\}^m$ and sets $\mathbf{u}_i \leftarrow \mathbf{A}\mathbf{r}_i$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output of an execution of Hyb_i with adversary $\mathcal{A}$. We now show that each adjacent pair of experiments are computationally indistinguishable.

Lemma 5.7. *Suppose $n \ge \lambda$, $m \ge O(n\log q)$, and $s_0 \ge O(\ell m^2\log(n\ell))$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_0(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A})$.*

Proof. Identical to the proof of Lemma 4.5 (since the structure of the CRS in Construction 5.3 is identical to that in Construction 4.2 except for the additional random vectors $\mathbf{u}_i$). □

Lemma 5.8. *Suppose $n \geq \lambda$ and $m \geq O(n \log q)$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_1(\mathcal{A}) \stackrel{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

Proof. Identical to the proof of Lemma 4.6. □

Lemma 5.9. *Suppose $n \geq \lambda$ and $m \geq 2n \log q$. Then, for all adversaries $\mathcal{A}$ and all polynomials $\ell = \ell(\lambda)$, $\mathsf{Hyb}_2(\mathcal{A}) \stackrel{s}{\approx} \mathsf{Hyb}_3(\mathcal{A})$.*

Proof. The only difference between Hyb_2 and Hyb_3 is the distribution of $\mathbf{u}_i$. In Hyb_2, $\mathbf{u}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ while in Hyb_3, $\mathbf{u} \leftarrow \mathbf{A}\mathbf{r}_i$ where $\mathbf{r}_i \xleftarrow{\mathrm{R}} \{0,1\}^m$. Since $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}$ and $m \geq 2n \log q$, the distribution of $\mathbf{u}_i$ in the two distributions are statistically indistinguishable by the leftover hash lemma (Lemma 2.4). The claim now follows by a hybrid argument (since $\ell = \ell(\lambda)$ is polynomially-bounded). □

Lemma 5.10. *Suppose q is prime and $m \geq n \log q + \lambda$. Under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with parameters $(n, m, q, \ell(2B + p), s_0, \ell)$, for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists an efficient algorithm $\mathcal{A}$ where $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \varepsilon$ for some non-negligible ε. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption.

1. At the beginning of the game, algorithm $\mathcal{B}$ receives a challenge $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$, $\mathbf{T} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$, and $\mathsf{aux} = (\mathbf{W}_1, \ldots, \mathbf{W}_\ell)$.
2. For each $i \in [\ell]$, algorithm $\mathcal{B}$ samples $\mathbf{r}_i \xleftarrow{\mathrm{R}} \{0,1\}^m$ and sets $\mathbf{u}_i \leftarrow \mathbf{A}\mathbf{r}_i$. It gives $\mathsf{crs} = \left(\mathbf{A}, \{\mathbf{W}_i\}_{i \in [\ell]}, \{\mathbf{u}_i\}_{i \in [\ell]}, \mathbf{T}\right)$ to $\mathcal{A}$.
3. Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{c} \in \mathbb{Z}_q^n$, a set $S \subseteq [\ell]$, openings $\pi = \mathbf{v}$, $\pi' = \mathbf{v}'$, and values $\{(i, x_i, x_i')\}_{i \in S}$.
4. Algorithm $\mathcal{B}$ outputs $\mathbf{z} = \mathbf{v} - \mathbf{v}' + \sum_{i \in S}(x_i - x_i')\mathbf{r}_i$.

By construction, algorithm $\mathcal{B}$ perfectly simulates the common reference string according to the specification of Hyb_3. Thus, with probability ε, $\|\mathbf{v}\|, \|\mathbf{v}'\| \leq |S|B \leq \ell B$ and moreover, there exists $i \in S$ where $x_i \neq x_i'$ and

$$\sum_{i \in S} \mathbf{W}_i^{-1}\mathbf{c} = \mathbf{A}\mathbf{v} + \sum_{i \in S} x_i \mathbf{u}_i = \mathbf{A}\mathbf{v}' + \sum_{i \in S} x_i' \mathbf{u}_i.$$

We can thus write

$$\mathbf{0} = \mathbf{A}(\mathbf{v} - \mathbf{v}') + \sum_{i \in S}(x_i - x_i')\mathbf{u}_i = \mathbf{A}(\mathbf{v} - \mathbf{v}') + \sum_{i \in S}(x_i - x_i')\mathbf{A}\mathbf{r}_i = \mathbf{A}\mathbf{z}.$$

Moreover, $\|\mathbf{z}\| \leq 2\ell B + p|S| \leq \ell(2B + p)$. It suffices to show that $\mathbf{z} \neq \mathbf{0}$. Here, we use a min-entropy entropy argument similar to the proof of Lemma 4.29:

- If adversary $\mathcal{A}$ is successful, there exists some $i \in S$ where $x_i \neq x_i'$.
- Let $\mathbf{z}' = (x_i - x_i')^{-1}\left(\mathbf{v}' - \mathbf{v} - \sum_{j \in S \setminus \{i\}}(x_j - x_j')\mathbf{r}_j\right) \in \mathbb{Z}_q^m$. If $\mathbf{z} = \mathbf{0}$, then $\mathbf{r}_i = \mathbf{z}' \in \mathbb{Z}_q^m$. By construction, $\mathbf{v}'$, $\mathbf{v}$, $\{x_i\}_{i \in S}$, $\{x_i'\}_{i \in S}$ are functions of $\mathbf{u}_i \in \mathbb{Z}_q^n$ (and other quantities that are independent of $\mathbf{r}_i$). By construction, each $\mathbf{u}_i$ contains at most $n \log q$ bits of information about $\mathbf{r}_i$. Moreover, $\mathbf{r}_i$ is sampled independently of $\mathbf{r}_j$ for all $j \neq i$. This means that
$$\mathbf{H}_\infty(\mathbf{r}_i \mid \mathbf{z}') \geq \mathbf{H}_\infty(\mathbf{r}_i \mid \mathbf{u}_i) \geq m - n \log q \geq \lambda.$$
This means that $\Pr[\mathbf{r}_i = \mathbf{z}'] \leq 2^{-\lambda}$, so with overwhelming probability, $\mathbf{r}_i \neq \mathbf{z}'$, and so $\mathbf{z} \neq \mathbf{0}$. In this case, algorithm $\mathcal{B}$ breaks the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with advantage at least $\varepsilon - 2^{-\lambda}$. □

Combining Lemmas 5.7 to 5.10, same-set binding holds. □

Parameter instantiation. Let λ be a security parameter and ℓ be the vector dimension. We can instantiate the lattice parameters in Construction 5.3 to satisfy Theorems 5.4 and 5.6:

- We set the lattice dimension $n = \lambda$ and $m = O(n \log q)$.
- We set $s_0 = O(\ell m^2 \log(n\ell))$ and $s_1 = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0) = O(\ell^{5/2} m^{7/2} \log^2(n\ell))$
- We set the bound $B = \sqrt{\ell m + m'} s_1 = O(\ell^3 m^4 \log^2(n\ell)) = O(\ell^3 n^4 \log^2(n\ell) \log^4 q)$.
- We choose the modulus q so that the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption holds with parameters $(n, m, q, \beta, s_0, \ell)$ where

$$\beta = \ell(2B + p) = O(\ell^4 n^4 \log^2(n\ell) \log^4 q + \ell p).$$

  Note that this also requires that $\mathsf{SIS}_{n,m,q,\beta}$ hold. For instance, we can set $q = \beta \cdot \mathsf{poly}(n)$. In this case, $\log q = O(\log \lambda + \log \ell + \log p)$.

With this setting of parameters, we obtain an aggregatable commitment scheme over $\mathbb{Z}_p^\ell$ with the following succinctness properties:

- **Commitment size:** A commitment σ to a vector $\mathbf{x} \in \mathbb{Z}_p^\ell$ consists of a vector $\sigma = \mathbf{c} \in \mathbb{Z}_q^n$, so

$$|\sigma| = O(n \log q) = O\big(\lambda \cdot (\log \lambda + \log \ell + \log p)\big).$$

- **Opening size:** An opening π to a set S consists of a vector $\pi = \mathbf{v}_S \in \mathbb{Z}_q^m$. Thus, the length of $\mathbf{v}_i$ is bounded by $O(m \log q)$, or equivalently, $|\pi| = O(\lambda \cdot (\log^2 \lambda + \log^2 \ell + \log^2 p))$.
- **CRS size:** The CRS consists of $(\mathbf{A}, \{\mathbf{W}_i\}_{i\in[\ell]}, \{\mathbf{u}_i\}_{i\in[\ell]}, \mathbf{T})$, where $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{W}_i \in \mathbb{Z}_q^{n\times n}$, $\mathbf{u}_i \in \mathbb{Z}_q^n$, and $\mathbf{T} \in \mathbb{Z}_q^{(\ell m+m')\times \ell m'}$. Thus, the total size of the CRS is

$$|\mathsf{crs}| = nm \log q + \ell(n^2 + n) \log q + (\ell m + m')(\ell m') \log q = \ell^2 \cdot \mathsf{poly}(\lambda, \log \ell, \log p).$$

Thus, Construction 5.3 is a succinct aggregatable vector commitment scheme that satisfies same-set binding:

Corollary 5.11 (Aggregatable Vector Commitment from $\mathsf{BASIS}_{\mathsf{struct}}$)**.** *Let λ be a security parameter. Then, for all polynomials $\ell = \ell(\lambda)$ and $p = p(\lambda)$, under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with a norm bound $\beta = \mathsf{poly}(\lambda, \ell, p)$ and modulus $q = \mathsf{poly}(\lambda, \ell, p)$, there exists an aggregatable vector commitment scheme over $\mathbb{Z}_p^\ell$ that satisfies computational same-set binding. A commitment to a vector $\mathbf{x} \in \mathbb{Z}_p^\ell$ has size $O(\lambda(\log \lambda + \log \ell + \log p))$ and the opening to any set of indices $S \subseteq [\ell]$ has size $O(\lambda(\log^2 \lambda + \log^2 \ell + \log^2 p))$. The CRS has size $\ell^2 \cdot \mathsf{poly}(\lambda, \log \ell, \log p)$.*

Different-set binding. While Construction 5.3 is an aggregatable vector commitment scheme satisfying same-set binding, it does not satisfy different-set binding. We describe an attack in Remark 5.12. Then, in Remark 5.13, we show that in the setting where the commitment is guaranteed to be *honestly-generated*, then same-set binding implies different-set binding.

Remark 5.12 (Attack on Different-Set Binding)**.** While Construction 5.3 satisfies same-set binding, it does not satisfy the stronger notion of different-set binding (Remark 5.2). Here, we describe a general attack strategy:

- Let $\mathsf{crs} = \big(\mathbf{A}, \{\mathbf{W}_i\}_{i\in[\ell]}, \{\mathbf{u}_i\}_{i\in[\ell]}, \mathbf{T}\big) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$.
- Take any two sets $S, T \subseteq [\ell]$ where $S \neq T$. Let $\mathbf{W}_S = \sum_{i\in S} \mathbf{W}_i^{-1}$ and $\mathbf{W}_T = \sum_{i\in T} \mathbf{W}_i^{-1}$. The adversary's goal is to compute a commitment $\mathbf{c} \in \mathbb{Z}_q^n$ and short openings $\mathbf{v}_S \in \mathbb{Z}_q^m$, $\mathbf{v}_T \in \mathbb{Z}_q^m$ such that

$$\mathbf{c} = \mathbf{W}_S^{-1}\mathbf{A}\mathbf{v}_S + \sum_{i\in S} x_i \mathbf{W}_S^{-1}\mathbf{u}_i = \mathbf{W}_T^{-1}\mathbf{A}\mathbf{v}_T + \sum_{i\in T} y_i \mathbf{W}_T^{-1}\mathbf{u}_i, \tag{5.3}$$

and there exists $i \in S \cap T$ where $x_i \neq y_i$. We can rewrite this as

$$\left[\mathbf{W}_S^{-1}\mathbf{A} \mid -\mathbf{W}_T^{-1}\mathbf{A}\right]\begin{bmatrix}\mathbf{v}_S\\ \mathbf{v}_T\end{bmatrix} = \sum_{i\in T} y_i \mathbf{W}_T^{-1}\mathbf{u}_i - \sum_{i\in S} x_i \mathbf{W}_S^{-1}\mathbf{u}_i. \tag{5.4}$$

If the adversary has a trapdoor for $\left[\mathbf{W}_S^{-1}\mathbf{A} \mid -\mathbf{W}_T^{-1}\mathbf{A}\right]$, then it can sample a short $\mathbf{v}_S, \mathbf{v}_T$ to satisfy Eq. (5.4) for *any* choice of x_i and y_i. Given $\mathbf{v}_S, \mathbf{v}_T$, the adversary can then compute the commitment $\mathbf{c}$ according to Eq. (5.3), which breaks different-set binding.

It suffices now to show how to obtain a trapdoor for $\left[\mathbf{W}_S^{-1}\mathbf{A} \mid -\mathbf{W}_T^{-1}\mathbf{A}\right]$. Suppose the adversary obtains many short vectors $\mathbf{v}_{j,i} \in \mathbb{Z}_q^m$ that satisfy $\mathbf{A}\mathbf{v}_{j,i} = \mathbf{W}_i^{-1}\mathbf{t}_j$ for some fixed set of vectors $\mathbf{t}_j$. Then,

$$\mathbf{W}_S^{-1}\mathbf{A}\sum_{i\in S}\mathbf{v}_{j,i} = \mathbf{W}_S^{-1}\sum_{i\in S}\mathbf{W}_i^{-1}\mathbf{t}_j = \mathbf{W}_S^{-1}\mathbf{W}_S\mathbf{t}_j = \mathbf{t}_j$$
$$\mathbf{W}_T^{-1}\mathbf{A}\sum_{i\in T}\mathbf{v}_{j,i} = \mathbf{W}_T^{-1}\sum_{i\in T}\mathbf{W}_i^{-1}\mathbf{t}_j = \mathbf{W}_T^{-1}\mathbf{W}_T\mathbf{t}_j = \mathbf{t}_j.$$

This means that

$$\left[\mathbf{W}_S^{-1}\mathbf{A} \mid -\mathbf{W}_T^{-1}\mathbf{A}\right]\underbrace{\begin{bmatrix}\sum_{i\in S}\mathbf{v}_{j,i}\\ \sum_{i\in T}\mathbf{v}_{j,i}\end{bmatrix}}_{\mathbf{z}_j} = \mathbf{t}_j - \mathbf{t}_j = \mathbf{0}.$$

If the adversary can construct $2m$ vectors $\mathbf{z}_j \in \mathbb{Z}^{2m}$ that are linearly independent, this yields an Ajtai-trapdoor (Definition 2.13) for $\left[\mathbf{W}_S^{-1}\mathbf{A} \mid -\mathbf{W}_T^{-1}\mathbf{A}\right]$, which suffices to break the scheme via the above blueprint. We now show that we can use the trapdoor $\mathbf{T}$ in the common reference string of Construction 5.3 to obtain the short vectors $\mathbf{z}_j$:

- Let $\mathbf{B}_\ell \in \mathbb{Z}_q^{(\ell m+m')\times \ell m'}$ be the matrix from Eq. (5.1). By construction, $\mathbf{T}$ in the CRS is a trapdoor for $\mathbf{B}_\ell$.
- For each $j \in [2m]$, we can sample a (short) preimage in the kernel of $\mathbf{B}_\ell$:

$$\begin{bmatrix}\mathbf{v}_{j,1}\\ \vdots\\ \mathbf{v}_{j,\ell}\\ \hat{\mathbf{c}}_j\end{bmatrix} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \mathbf{T}, \mathbf{0}, s_1).$$

  By construction of $\mathbf{B}_\ell$, we have that for all $i \in [\ell]$, $\mathbf{W}_i\mathbf{A}\mathbf{v}_{j,i} + \mathbf{G}\hat{\mathbf{c}}_j = \mathbf{0}$, or equivalently, $\mathbf{A}\mathbf{v}_{j,i} = -\mathbf{W}_i^{-1}\mathbf{G}\hat{\mathbf{c}}_j$. We can set $\mathbf{t}_j = -\mathbf{G}\hat{\mathbf{c}}_j$.

- By Theorem 2.12, the distribution of $\mathbf{A}$ is statistically close to uniform over $\mathbb{Z}_q^{n\times m}$. For each $i \in [\ell]$, the matrix $\mathbf{W}_i$ is invertible so the distribution of each $\mathbf{W}_i\mathbf{A}$ remains statistically close to uniform over $\mathbb{Z}_q^{n\times m}$. By Lemma 2.2 and Corollary 2.5, this means that with overwhelming probability, $\mathbf{W}_i\mathbf{A}$ is full rank and $\lambda_1^\infty(\Lambda(\mathbf{W}_i\mathbf{A})) \geq q/4$. By a union bound over all $i \in [\ell]$, this holds for all $i \in [\ell]$. Since $s_1 \geq 4\log(\ell m)$, we can now appeal to Corollary 2.11 to conclude that the distribution of $\hat{\mathbf{c}}_j$ is statistically close to $D_{\mathbb{Z},s_1}^m$, and each $\mathbf{v}_{j,i}$ is independent and distributed according to $\mathbf{A}_{s_1}^{-1}(\mathbf{W}_i^{-1}\mathbf{G}\hat{\mathbf{c}}_j)$. The marginal distribution of each $\mathbf{v}_{j,i}$ is then distributed according to $D_{\mathbb{Z},s_1}^m$. Moreover, the vectors $\mathbf{v}_{j,i}$ and $\mathbf{v}_{j',i}$ for $j \neq j'$ are independent.

- Heuristically, the vectors $\mathbf{z}_j = \begin{bmatrix}\sum_{i\in S}\mathbf{v}_{j,i}\\ \sum_{i\in T}\mathbf{v}_{j,i}\end{bmatrix} \in \mathbb{Z}^{2m}$ are linearly independent for different j, and thus, can be used to construct a trapdoor for $\left[\mathbf{W}_S^{-1}\mathbf{A} \mid -\mathbf{W}_T^{-1}\mathbf{A}\right]$. Note that this step requires that $S \neq T$ (otherwise, the vectors $\{\mathbf{z}_1, \ldots, \mathbf{z}_{2m}\}$ lie in the column span of $\begin{bmatrix}\mathbf{I}_m\\ \mathbf{I}_m\end{bmatrix}$ and span a vector space of rank at most m).

We note that our attack strategy described above is general and having a trapdoor $\mathbf{T}$ for $\mathbf{B}_\ell$ is *not* essential to the attack. For instance, we can also consider a variant of the construction using an [ACL+22]-style CRS distribution which contains vectors of the form $\mathbf{v}_{j,i} = \mathbf{A}^{-1}(\mathbf{W}^{-1}\mathbf{t}_j)$ for $i \neq j \in [\ell]$ (see Sections 1.2 and 6). Our general attack strategy above still applies in this setting, so long as $\ell - |S \cup T| \geq 2m$ (i.e., the CRS components allows the adversary to compute $\mathbf{z}_j = \begin{bmatrix}\sum_{i\in S}\mathbf{v}_{j,i}\\ \sum_{i\in T}\mathbf{v}_{j,i}\end{bmatrix}$ for every $j \notin S \cup T$.

Remark 5.13 (Different-Set Binding for Honestly-Generated Commitments)**.** While Construction 5.3 does not satisfy different-set binding for maliciously-chosen commitments, it does satisfy different-set binding in the setting where the commitment is *honestly-generated*; this is analogous to the notion of target binding in the setting of functional commitments (Definition 4.22). In this setting, the adversary in the different-set binding game receives the CRS and chooses a vector $\mathbf{x} \in \mathbb{Z}_p^\ell$. The challenger replies with a commitment $\sigma \leftarrow \mathsf{Commit}(\mathsf{crs}, \mathbf{x})$. At the end of the game, the adversary $\mathcal{A}$ wins if it outputs two sets $S, T \subseteq [\ell]$ along with openings π_S and π_T to values $\{(i, x_i)\}_{i \in S}$ and $\{(i, x'_i)\}_{i \in T}$, respectively. The adversary wins if π_S and π_T are valid openings and moreover, there exists $i \in S \cap T$ where $x_i \neq x'_i$. This binding notion is called *weak binding* by Gorbunov et al. [GRWZ20] and suffices for applications where the commitment is guaranteed to be well-formed (say, due to external protocol constraints or consensus mechanisms).

It is easy to see that same-set binding implies different-set binding for honestly-generated commitments. Suppose an adversary is able to open σ to $\{(i, x_i)\}_{i \in S}$ with a proof π_S and to $\{(i, x'_i)\}_{i \in T}$ with a proof π_T such that there exists $i \in S \cap T$ where $x_i \neq x'_i$. Let π'_S and π'_T be the honestly-computed openings of σ to S and T, respectively. Since $x_i \neq x'_i$, either (π_S, π'_S) is a pair of inconsistent openings of σ to S, or (π_T, π'_T) is a pair of inconsistent openings of σ to T. This breaks same-set binding of the underlying commitment scheme.

Remark 5.14 (Aggregatable Commitments with an [ACL+22]-Style CRS)**.** We can also obtain a variant of Construction 5.3 using an [ACL+22]-style CRS that does not include an explicit trapdoor for the matrix $\mathbf{B}_\ell$, and instead, just contains a collection of short preimages. We provide a more detailed comparison of the two approaches in Section 6.

- The CRS consists of matrices $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n \times m}$, $\mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n \times n}$ for each $i \in [\ell]$, vectors $\mathbf{u}_i \xleftarrow{R} \mathbb{Z}_q^n$ for each $i \in [\ell]$, and short vectors $\mathbf{z}_{j,i} \in \mathbb{Z}_q^m$ for all $i \neq j$ where $\mathbf{A}\mathbf{z}_{j,i} = \mathbf{W}_i^{-1}\mathbf{W}_j\mathbf{u}_j$.
- A commitment to a vector $\mathbf{x} \in \mathbb{Z}_p^\ell$ is a vector $\mathbf{c} = \sum_{i \in [\ell]} x_i \mathbf{W}_i \mathbf{u}_i \in \mathbb{Z}_q^n$. The opening to an index i is the vector $\mathbf{v}_i = \sum_{j \neq i} x_j \mathbf{z}_{j,i}$. The the verification relation checks that $\|\mathbf{v}_i\|$ is small and that

$$\mathbf{W}_i^{-1}\mathbf{c} = x_i\mathbf{u}_i + \mathbf{A}\mathbf{v}_i. \tag{5.5}$$

  Correctness follows from the fact that

$$\mathbf{W}_i^{-1}\mathbf{c} = \mathbf{W}_i^{-1} \sum_{i \in [\ell]} x_i \mathbf{W}_i \mathbf{u}_i = x_i\mathbf{u}_i + \sum_{i \neq j} x_j \mathbf{W}_i^{-1}\mathbf{W}_j\mathbf{u}_j = x_i\mathbf{u}_i + \sum_{i \neq j} x_j \mathbf{A}\mathbf{z}_{j,i} = x_i\mathbf{u}_i + \mathbf{A}\mathbf{v}_i.$$

- Observe that Eq. (5.5) is the same verification relation from Construction 5.3. Thus, the same approach for aggregation applies.

Computational same-set binding in this case would rely on a variant of the ISIS assumption that asserts that given $\left(\mathbf{A}, \{\mathbf{W}_i\}_{i \in [\ell]}, \{\mathbf{u}_i\}_{i \in [\ell]}, \{\mathbf{z}_{j,i}\}_{i \neq j}\right)$, where $\mathbf{z}_{j,i}$ are short vectors that satisfy $\mathbf{A}\mathbf{z}_{j,i} = \mathbf{W}_i^{-1}\mathbf{W}_j\mathbf{u}_j$, no efficient adversary can compute a short $\mathbf{v}_i$ where $\mathbf{A}\mathbf{v}_i = \mathbf{W}_i^{-1}\mathbf{W}_i\mathbf{u}_i = \mathbf{u}_i$. Note that this version of the construction still does *not* satisfy different-set binding. The vectors $\mathbf{z}_{j,i}$ that are included as part of the CRS (which are *necessary* for correctness) still suffice to carry out the attack in Remark 5.12.

5.1 Aggregatable Functional Commitments

Our techniques for aggregating vector commitments also applies to aggregating openings for the functional commitment scheme from Section 4 (Construction 4.2). Recall that in the case of a functional commitment, an opening π is defined with respect to a function f and a value y. Much like the setting with vector commitments, the goal here is to take a collection of t function-value-opening tuples $\{(f_i, y_i, \pi_i)\}_{i \in [t]}$ and aggregate the openings into a *single* opening π whose size scales sublinearly with t. We start by extending Definition 5.1 to the setting of functional commitments:

Definition 5.15 (Aggregatable Functional Commitment)**.** Let λ be a security parameter, $T = T(\lambda)$ be an aggregation bound, and $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda \in \mathbb{N}}$ be a family of functions $f\colon \{0,1\}^\ell \to \{0,1\}$ on inputs of length $\ell = \ell(\lambda)$ and depth at most $d = d(\lambda)$. A T-aggregatable functional commitment for function family $\mathcal{F}$ is a tuple of efficient algorithms $\Pi_{\mathsf{AFC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Eval}, \mathsf{Verify}, \mathsf{Aggregate}, \mathsf{VerifyAgg})$ where $(\mathsf{Setup}, \mathsf{Commit}, \mathsf{Eval}, \mathsf{Verify})$ is a functional commitment scheme over $\mathcal{F}$ and the additional functions $(\mathsf{Aggregate}, \mathsf{VerifyAgg})$ satisfy the following properties:

- $\mathsf{Aggregate}(\mathsf{crs}, \sigma, \{(f_i, y_i, \pi_i)\}_{i\in[t]}) \to \pi$: On input the common reference string crs, a commitment σ, a set of openings π_i for functions $f_i \in \mathcal{F}$ and values y_i, the aggregation algorithm outputs an aggregated opening π.
- $\mathsf{VerifyAgg}(\mathsf{crs}, \sigma, \{(f_i, y_i)\}_{i\in[t]}, \pi) \to \{0,1\}$: On input the common reference string crs, a commitment σ, a set of functions $f_i \in \mathcal{F}$ and values y_i, the aggregate verification algorithm outputs a bit $b \in \{0,1\}$.

In addition, we allow the Setup algorithm to take as input the aggregation bound parameter 1^T (encoded in *unary*). Next, in addition to the basic properties of a functional commitment scheme, an aggregatable functional commitment scheme should satisfy the following additional properties:

- **Correctness of aggregation:** For all aggregation bound parameters $T = T(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all security parameters $\lambda \in \mathbb{N}$, and sampling $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d, 1^T)$, and for all $t \le T$, commitments σ, and openings $\{(f_i, y_i, \pi_i)\}_{i\in[t]}$ where $\mathsf{Verify}(\mathsf{crs}, \sigma, f_i, y_i, \pi_i) = 1$ for all $i \in [t]$, it holds that

$$\Pr[\mathsf{VerifyAgg}(\mathsf{crs}, \sigma, \{(f_i, y_i)\}_{i\in[t]}, \pi') : \pi' \leftarrow \mathsf{Aggregate}(\mathsf{crs}, \sigma, \{(f_i, y_i, \pi_i)\}_{i\in[t]})] = 1 - \mathsf{negl}(\lambda),$$

  where the probability is taken over the randomness of Setup.
- **Succinctness:** The aggregatable functional commitment scheme is succinct if there exists a universal polynomial $\mathsf{poly}(\cdot,\cdot,\cdot,\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\pi'| = \mathsf{poly}(\lambda, d, \log \ell, \log T)$ in the correctness of aggregation definition.
- **Same-function binding:** We say Π_{AVC} satisfies statistical (resp., computational) same-function binding if for all polynomials $T = T(\lambda)$ and all adversaries $\mathcal{A}$ (resp., efficient adversaries $\mathcal{A}$),

$$\Pr\left[\begin{array}{c} \mathsf{VerifyAgg}(\mathsf{crs}, \sigma, \{(f_i, y_i)\}_{i\in[t]}, \pi) = 1 \\ \text{and } y_i \neq y_i' \text{ for some } i \in [t] \text{ and} \\ \mathsf{VerifyAgg}(\mathsf{crs}, \sigma, \{(f_i, y_i')\}_{i\in[t]}, \pi') = 1 \end{array} : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d, 1^T); \\ (\sigma, \{(f_i, y_i, y_i')\}_{i\in[t]}, \pi, \pi') \leftarrow \mathcal{A}(1^\lambda, 1^\ell, 1^d, 1^T, \mathsf{crs}) \end{array}\right] = \mathsf{negl}(\lambda).$$

Aggregatable functional commitments. We now show how to adapt Construction 4.2 to support opening aggregation. Our construction can be viewed as a combination of Construction 4.2 and Construction 5.3:

Construction 5.16 (Aggregatable Functional Commitment)**.** Let λ be a security parameter and let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of functions $f\colon \{0,1\}^\ell \to \{0,1\}$ where each $f\colon \{0,1\}^\ell \to \{0,1\}$ is a function on inputs of length $\ell = \ell(\lambda)$ and which can be computed by a Boolean circuit of depth at most $d = d(\lambda)$. Let n, m, q, m', B, s_0, s_1 be the scheme parameters from Construction 4.2. Let $T = T(\lambda)$ be an arbitrary aggregation bound and $B_{\mathsf{agg}} = B_{\mathsf{agg}}(\lambda)$ be an aggregation norm bound. We now extend Construction 4.2 to obtain an aggregatable functional commitment $\Pi_{\mathsf{AVC}} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Open}, \mathsf{Verify}, \mathsf{Aggregate}, \mathsf{VerifyAgg})$ for $\mathcal{F}$:

- $\mathsf{Setup}(1^\lambda, 1^\ell, 1^d, 1^T)$: Sample $\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}$ according to the specification of $\mathsf{Setup}(1^\lambda, 1^\ell, 1^d)$ in Construction 4.2. Then, for each $i \in [T]$, sample $\mathbf{u}_i \xleftarrow{R} \mathbb{Z}_q^n$, and output $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u}_1, \ldots, \mathbf{u}_T)$.
- $\mathsf{Commit}(\mathsf{crs}, \mathbf{x})$: Same as in Construction 4.2.
- $\mathsf{Eval}(\mathsf{crs}, \mathsf{st}, f)$: Same as in Construction 4.2.
- $\mathsf{Verify}(\mathsf{crs}, \sigma, f, y, \pi)$: Same as in Construction 4.2.
- $\mathsf{Aggregate}\big(\mathsf{crs}, \sigma, \{(f_i, y_i, \pi_i)\}_{i\in[t]}\big)$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u}_1, \ldots, \mathbf{u}_T)$, a commitment $\sigma = \mathbf{C} \in \mathbb{Z}_q^{n\times m'}$, a collection of functions $f_1, \ldots, f_t \in \mathcal{F}$ (arranged in lexicographic order),[11] values $y_i \in \{0,1\}$, and openings $\pi_i = \mathbf{V}_i \in \mathbb{Z}_q^{m\times m'}$, the aggregation algorithm outputs $\pi = \mathbf{v} \leftarrow \sum_{i\in[t]} \mathbf{V}_i \cdot \mathbf{G}^{-1}(\mathbf{u}_i) \in \mathbb{Z}_q^m$.
- $\mathsf{VerifyAgg}\big(\mathsf{crs}, \sigma, \{(f_i, y_i)\}_{i\in[t]}, \pi\big)$: On input the common reference string $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u}_1, \ldots, \mathbf{u}_T)$, a commitment $\sigma = \mathbf{C} \in \mathbb{Z}_q^{n\times m'}$, a collection of functions $f_1, \ldots, f_t \in \mathcal{F}$ (arranged in lexicographic order), values $y_i \in \{0,1\}$, and an opening $\pi = \mathbf{v} \in \mathbb{Z}_q^m$, the verification algorithm first sets $\tilde{\mathbf{C}} \leftarrow [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$. Then, for each $i \in [t]$, it computes $\tilde{\mathbf{C}}_i \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f_i)$. Finally, it outputs 1 if

$$\|\mathbf{v}\| \le B_{\mathsf{agg}} \quad \text{and} \quad \mathbf{A}\mathbf{v} = \sum_{i\in[t]} \tilde{\mathbf{C}}_i \mathbf{G}^{-1}(\mathbf{u}_i) - \sum_{i\in[t]} y_i \mathbf{u}_i.$$

[11] This ensures that there is a *canonical* ordering for all tuples of functions $(f_1, \ldots, f_t)$.

Theorem 5.17 (Correctness of Aggregation). *Suppose $B_{\mathsf{agg}} \geq Tm'B$. Then, Construction 5.16 satisfies correctness of aggregation.*

Proof. Take any $T = T(\lambda)$ and let $\mathsf{crs} = (\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u}_1, \ldots, \mathbf{u}_T) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell, 1^d, 1^T)$. Take any $t \leq T$, any commitment $\sigma = \mathbf{C} \in \mathbb{Z}_q^{n \times m'}$, and set of openings (f_i, y_i, π_i) for $i \in [t]$, where $f_1, \ldots, f_t \in \mathcal{F}$ are in lexicographic order, and $\pi_i = \mathbf{V}_i \in \mathbb{Z}_q^{m \times m'}$. Let $\tilde{\mathbf{C}} \leftarrow [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$ and $\tilde{\mathbf{C}}_i \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f_i)$ for each $i \in [t]$. Suppose $\mathsf{Verify}(\mathsf{crs}, \sigma, f_i, y_i, \pi_i) = 1$. By definition this means that

$$\|\mathbf{V}_i\| \leq B \quad \text{and} \quad \mathbf{A}\mathbf{V}_i = \tilde{\mathbf{C}}_i - y_i\mathbf{G}.$$

Suppose $\mathbf{v} \leftarrow \mathsf{Aggregate}\big(\mathsf{crs}, \mathbf{C}, \{(f_i, y_i, \pi_i)\}_{i \in [t]}\big)$ and consider $\mathsf{VerifyAgg}(\mathsf{crs}, \mathbf{C}, \{(f_i, y_i)\}_{i \in [t]}, \mathbf{v})$. By definition, $\mathbf{v} = \sum_{i \in [t]} \mathbf{V}_i \mathbf{G}^{-1}(\mathbf{u}_i)$. Since $\|\mathbf{V}_i\| \leq B$, this means that $\|\mathbf{v}\| \leq tm'B \leq Tm'B = B_{\mathsf{agg}}$. Moreover,

$$\mathbf{A}\mathbf{v} = \sum_{i \in [t]} \mathbf{A}\mathbf{V}_i \mathbf{G}^{-1}(\mathbf{u}_i) = \sum_{i \in [t]} \tilde{\mathbf{C}}_i \mathbf{G}^{-1}(\mathbf{u}_i) - \sum_{i \in [t]} y_i \mathbf{u}_i,$$

and the aggregate verification algorithm accepts. □

Theorem 5.18 (Computational Same-Function Binding). *Suppose $n \geq \lambda$, $m \geq O(n \log q)$, and $s_0 \geq O(\ell m^2 \log(n\ell))$. Then, under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with parameters $(n, m, q, 2B_{\mathsf{agg}} + T, s_0, \ell)$, Construction 5.16 satisfies same-function binding.*

Proof. The proof follows by a combination of the proofs of Theorem 4.4 and Theorem 5.6. Specifically, let $T = T(\lambda)$ be a polynomial. We now define a sequence of hybrid experiments:

- Hyb_0: This is the same-function binding experiment:
  - The challenger starts by sampling $(\mathbf{A}, \mathbf{R}) \leftarrow \mathsf{TrapGen}(1^n, q, m)$ and $\mathbf{W}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times n}$ for each $i \in [\ell]$. It sets $\tilde{\mathbf{R}}_i \leftarrow \mathbf{R}\mathbf{G}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$ and constructs $\mathbf{B}_\ell$ and $\tilde{\mathbf{R}}$ according to Eq. (4.1). It samples $\mathbf{T} \leftarrow \mathsf{SamplePre}(\mathbf{B}_\ell, \tilde{\mathbf{R}}, \mathbf{G}_{n\ell}, s_0)$, $\mathbf{u}_i \xleftarrow{\mathrm{R}} \mathbb{Z}_q^n$ for each $i \in [T]$, and gives $\mathsf{crs} = \big(\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u}_1, \ldots, \mathbf{u}_T\big)$ to the adversary $\mathcal{A}$.
  - Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{C} \in \mathbb{Z}_q^{n \times m'}$, a set of functions $f_1, \ldots, f_t$ (arranged in lexicographic order), a set of values $y_1, \ldots, y_t \in \{0,1\}$, $y_1', \ldots, y_t' \in \{0,1\}$, with $t \leq T$, and openings $\mathbf{v}, \mathbf{v}' \in \mathbb{Z}_q^m$.
  - The output of the experiment is 1 if $\|\mathbf{v}\|, \|\mathbf{v}'\| \leq B_{\mathsf{agg}}$ and

$$\mathbf{A}\mathbf{v} = \sum_{i \in [t]} \tilde{\mathbf{C}}_i \mathbf{G}^{-1}(\mathbf{u}_i) - \sum_{i \in [t]} y_i \mathbf{u}_i \quad \text{and} \quad \mathbf{A}\mathbf{v}' = \sum_{i \in [t]} \tilde{\mathbf{C}}_i \mathbf{G}^{-1}(\mathbf{u}_i) - \sum_{i \in [t]} y_i' \mathbf{u}_i,$$

    where $\tilde{\mathbf{C}}_i \leftarrow \mathsf{EvalF}(\tilde{\mathbf{C}}, f_i)$ and $\tilde{\mathbf{C}} = [\mathbf{W}_1^{-1}\mathbf{C} \mid \cdots \mid \mathbf{W}_\ell^{-1}\mathbf{C}]$. Otherwise, the experiments outputs 0.
- Hyb_1: Same as Hyb_0 except after constructing the matrix $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$ according to Eq. (4.1), the challenger samples $\mathbf{T} \leftarrow (\mathbf{B}_\ell)_{s_0}^{-1}(\mathbf{G}_{n\ell})$ *without* using the trapdoor $\tilde{\mathbf{R}}$. The CRS is now sampled independently of $\mathbf{R}$.
- Hyb_2: Same as Hyb_1 except the challenger samples $\mathbf{A} \xleftarrow{\mathrm{R}} \mathbb{Z}_q^{n \times m}$.
- Hyb_3: Same as Hyb_2 except for each $i \in [\ell]$, it samples $\mathbf{r}_i \xleftarrow{\mathrm{R}} \{0,1\}^m$ and sets $\mathbf{u}_i \leftarrow \mathbf{A}\mathbf{r}_i$.

For an adversary $\mathcal{A}$, we write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output of an execution of Hyb_i with adversary $\mathcal{A}$. We now show that each adjacent pair of experiments are computationally indistinguishable.

Lemma 5.19. *Suppose $n \geq \lambda$, $m \geq O(n \log q)$, and $s_0 \geq O(\ell m^2 \log(n\ell))$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_0(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_1(\mathcal{A})$.*

Proof. Identical to the proof of Lemma 4.5. □

Lemma 5.20. *Suppose $n \geq \lambda$ and $m \geq O(n \log q)$. Then, for all adversaries $\mathcal{A}$, $\mathsf{Hyb}_1(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_2(\mathcal{A})$.*

Proof. Identical to the proof of Lemma 4.6. □

Lemma 5.21. *Suppose $n \geq \lambda$ and $m \geq 2n \log q$. Then for all adversaries $\mathcal{A}$ and all polynomials $T = T(\lambda)$, $\mathsf{Hyb}_2(\mathcal{A}) \overset{s}{\approx} \mathsf{Hyb}_3(\mathcal{A})$.*

Proof. Identical to the proof of Lemma 5.9 (except using the fact that T is polynomially-bounded). □

Lemma 5.22. *Suppose $m \geq n \log q + \lambda$. Under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with parameters $(n, m, q, 2B_{\mathsf{agg}} + T, s_0, \ell)$, for all efficient adversaries $\mathcal{A}$, $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \mathsf{negl}(\lambda)$.*

Proof. Suppose there exists an efficient algorithm $\mathcal{A}$ where $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \varepsilon$ for some non-negligible ε. We use $\mathcal{A}$ to construct an efficient adversary $\mathcal{B}$ for the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption.

1. At the beginning of the game, algorithm $\mathcal{B}$ receives a challenge $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{B}_\ell \in \mathbb{Z}_q^{n\ell \times (\ell m + m')}$, $\mathbf{T} \in \mathbb{Z}_q^{(\ell m + m') \times \ell m'}$, and $\mathsf{aux} = (\mathbf{W}_1, \ldots, \mathbf{W}_\ell)$.
2. For each $i \in [T]$, algorithm $\mathcal{B}$ samples $\mathbf{r}_i \xleftarrow{\mathrm{R}} \{0,1\}^m$ and sets $\mathbf{u}_i \leftarrow \mathbf{A}\mathbf{r}_i$. It gives $\mathsf{crs} = \left(\mathbf{A}, \mathbf{W}_1, \ldots, \mathbf{W}_\ell, \mathbf{T}, \mathbf{u}_1, \ldots, \mathbf{u}_T\right)$ to $\mathcal{A}$.
3. Algorithm $\mathcal{A}$ outputs a commitment $\mathbf{C} \in \mathbb{Z}_q^{n \times m'}$, a set of functions $f_1, \ldots, f_t$ (arranged in lexicographic order), a set of values $y_1, \ldots, y_t \in \{0,1\}$, $y'_1, \ldots, y'_t \in \{0,1\}$, with $t \leq T$, and openings $\mathbf{v}, \mathbf{v}' \in \mathbb{Z}_q^m$.
4. Algorithm $\mathcal{B}$ outputs $\mathbf{z} = \mathbf{v} - \mathbf{v}' + \sum_{i \in [t]} (y_i - y'_i)\mathbf{r}_i$.

By construction, algorithm $\mathcal{B}$ perfectly simulates the common reference string according to the specification of Hyb_3. Thus, with probability ε, $\|\mathbf{v}\|, \|\mathbf{v}'\| \leq B_{\mathsf{agg}}$, $t \leq T$, and moreover, there exists $i \in [t]$ where $y_i \neq y'_i$ and

$$\sum_{i \in [t]} \tilde{\mathbf{C}}_i \mathbf{G}^{-1}(\mathbf{u}_i) = \mathbf{A}\mathbf{v} + \sum_{i \in [t]} y_i \mathbf{u}_i = \mathbf{A}\mathbf{v}' + \sum_{i \in [t]} y'_i \mathbf{u}_i,$$

Rearranging terms, we now have

$$\mathbf{0} = \mathbf{A}(\mathbf{v} - \mathbf{v}') + \sum_{i \in [t]} (y_i - y'_i)\mathbf{u}_i = \mathbf{A}(\mathbf{v} - \mathbf{v}') + \sum_{i \in [t]} (y_i - y'_i)\mathbf{A}\mathbf{r}_i = \mathbf{A}\mathbf{z}.$$

Moreover, $\|\mathbf{z}\| \leq 2B_{\mathsf{agg}} + t \leq 2B_{\mathsf{agg}} + T$. It suffices to show that $\mathbf{z} \neq \mathbf{0}$. Here, we use a min-entropy argument similar to the proof of Lemma 5.10:

- From above, there exists some $i \in [t]$ where $y_i \neq y'_i$ (and $y_i, y'_i \in \{0,1\}$ so $y_i - y'_i \in \{-1, 1\}$).
- Let $\mathbf{z}' = (y_i - y'_i)^{-1}\left(\mathbf{v}' - \mathbf{v} - \sum_{j \neq i} (y_j - y'_j)\mathbf{r}_j\right) \in \mathbb{Z}_q^m$. If $\mathbf{z} = \mathbf{0}$, then $\mathbf{r}_i = \mathbf{z}' \in \mathbb{Z}_q^m$. By construction, $\mathbf{v}'$, $\mathbf{v}$, $\{y_i\}_{i \in [t]}$, $\{y'_i\}_{i \in [t]}$ are functions of $\mathbf{u}_i \in \mathbb{Z}_q^n$ (and other quantities that are independent of $\mathbf{r}_i$). By construction, each $\mathbf{u}_i$ contains at most $n \log q$ bits of information about $\mathbf{r}_i$. Moreover, $\mathbf{r}_i$ is sampled independently of $\mathbf{r}_j$ for all $j \neq i$. This means that
$$\mathbf{H}_\infty(\mathbf{r}_i \mid \mathbf{z}') \geq \mathbf{H}_\infty(\mathbf{r}_i \mid \mathbf{u}_i) \geq m - n \log q \geq \lambda.$$
This means that $\Pr[\mathbf{r}_i = \mathbf{z}'] \leq 2^{-\lambda}$, so with overwhelming probability $\mathbf{z} \neq \mathbf{0}$, and $\mathcal{B}$ breaks the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with advantage at least $\varepsilon - 2^{-\lambda}$. □

Same-function binding now follows by combining Lemmas 5.19 to 5.22. □

Parameter instantiation. Let λ be a security parameter and $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of functions $f\colon \{0,1\}^\ell \to \{0,1\}$ on inputs of length $\ell = \ell(\lambda)$ and which can be computed by Boolean circuits of depth at most $d = d(\lambda)$. Let $T = T(\lambda)$ be an aggregation bound. We can instantiate the lattice parameters in Construction 5.16 similar to those for Construction 4.2. We summarize the instantiation below:

- Let $\varepsilon > 0$ be a constant. We set the lattice dimension $n = d^{1/\varepsilon} \cdot \mathsf{poly}(\lambda)$ and $m = O(n \log q)$.
- We set $s_0 = O(\ell m^2 \log(n\ell))$ and

$$s_1 = O(\ell^{3/2} m^{3/2} \log(n\ell) \cdot s_0) = O(\ell^{5/2} m^{7/2} \log^2(n\ell)) = O(\ell^{5/2} n^{7/2} \log^2(n\ell) \log^{7/2} q).$$

- We set the bound $B = s_1 \cdot \sqrt{\ell m + m'} \cdot (n \log q)^{O(d)} = \ell^3 \log^2 \ell \cdot (n \log q)^{O(d)}$ and the aggregation bound $B_{\mathsf{agg}} = Tm'B = T\ell^3 \log^2 \ell \cdot (n \log q)^{O(d)}$.
- We set the modulus q so that the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption holds with parameters $(n, m, q, \beta, s_0, \ell)$, where

$$\beta = 2B_{\mathsf{agg}} + T = T\ell^3 \log^2 \ell \cdot (n \log q)^{O(d)} = 2^{\tilde{O}(d)} = 2^{\tilde{O}(n^\varepsilon)},$$

  where we write $\tilde{O}(\cdot)$ to suppress polylogarithmic factors in λ, d, ℓ, T. Note that this also requires that $\mathsf{SIS}_{n,m,q,\beta}$ hold. For instance, we set $q = \beta \cdot \mathsf{poly}(n)$. Then, $\log q = \mathsf{poly}(d, \log\lambda, \log\ell, \log T)$. Note that the underlying SIS assumption relies on a *sub-exponential* noise bound.

With this setting of parameters, we obtain an aggregatable functional commitment scheme for $\mathcal{F}$ with the following properties:

Corollary 5.23 (Aggregatable Functional Commitment from $\mathsf{BASIS}_{\mathsf{struct}}$)**.** *Let λ be a security parameter, and let $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of functions $f\colon \{0,1\}^\ell \to \{0,1\}$ on inputs of length $\ell = \ell(\lambda)$ and which can be computed by Boolean circuits of depth at most $d = d(\lambda)$. Let $T = T(\lambda)$ be an arbitrary polynomial. Under the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption with a norm bound $\beta = 2^{\tilde{O}(d)}$ and modulus $q = 2^{\tilde{O}(d)}$, there exists a computationally-binding T-aggregatable functional commitment scheme for $\mathcal{F}$. The size of the commitment is $\mathsf{poly}(\lambda, d, \log\ell, \log T)$, and an aggregate opening to a set of up to T functions has size $\mathsf{poly}(\lambda, d, \log\ell, \log T)$. The size of the CRS is $(\ell^2 + T) \cdot \mathsf{poly}(\lambda, d, \log\ell, \log T)$. Here, $\tilde{O}(\cdot)$ suppresses polylogarithmic factors in λ, d, ℓ, and T.*

Remark 5.24 (Reducing the Public Parameter Size)**.** As described, the size of the CRS in Construction 5.16 scales with the aggregation bound T since we need to publish T target vectors $\mathbf{u}_1, \ldots, \mathbf{u}_T \in \mathbb{Z}_q^n$. We can reduce the size of the CRS by deriving $\mathbf{u}_i$ from a random oracle. Here, we include the description of a hash function $\mathsf{H}\colon [T] \to \mathbb{Z}_q^n$ in the CRS, and define $\mathbf{u}_i \leftarrow \mathsf{H}(i)$. Security follows if we model H as a random oracle. This also yields a scheme that supports aggregating an arbitrary polynomial number of commitments (i.e., since the scheme parameters scale with $\log T$, we can instantiate the scheme with $T = 2^\lambda$).

6 New SIS Assumptions: Relations and Discussion

In this section, we compare our approach of publishing a full trapdoor in the common reference string with the approach of Albrecht et al. [ACL+22] of publishing short preimages in the CRS. While Albrecht et al. formulate their assumption over polynomial rings, their ideas apply equally well in the integer setting. We describe everything over the integers to enable a more direct comparison. We start by recalling the general paradigm for constructing vector commitments from Section 1.2 common to our approach and their approach:

- The CRS consists of ℓ matrices $\mathbf{A}_i \in \mathbb{Z}_q^{n\times m}$ and a set of target vectors $\mathbf{t}_i \in \mathbb{Z}_q^n$ for $i \in [\ell]$. The CRS also contains some auxiliary information aux_ℓ that is used to construct commitments and openings.
- An opening $\mathbf{v}_i$ to value x_i at index i with respect to a commitment $\mathbf{c} \in \mathbb{Z}_q^n$ is a short vector $\mathbf{v}_i$ that satisfies $\mathbf{c} = \mathbf{A}_i\mathbf{v}_i - x_i\mathbf{t}_i$.

We now compare the two types of auxiliary information aux_ℓ in our approach (based on the BASIS assumption) and the Albrecht et al. approach (based on variants of the k-ISIS assumption):

(I) **Our approach:** In our approach based on the BASIS assumption, $\mathsf{aux}_\ell = \mathbf{T}$ is a trapdoor $\mathbf{T} \leftarrow \mathbf{B}_\ell^{-1}(\mathbf{G}_{n\ell})$ for the matrix $\mathbf{B}_\ell = [\mathrm{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_\ell) \mid -\mathbf{1}^\ell \otimes \mathbf{G}]$. As shown in Sections 3 to 5, the trapdoor $\mathbf{T}$ suffices to *jointly* sample commitments $\mathbf{c}$ and openings $\mathbf{v}_1, \ldots, \mathbf{v}_\ell$ that satisfy the verification relation.

(II) **The Albrecht et al. approach:** In the Albrecht et al. [ACL+22] approach, the auxiliary information $\mathsf{aux}_\ell = \{\mathbf{z}_{j,i}\}_{i \neq j}$ consists of a collection of short vectors $\mathbf{z}_{j,i} \leftarrow \mathbf{A}_i^{-1}(\mathbf{t}_j)$. The commitment to a vector $\mathbf{x} \in \{0,1\}^\ell$ is the vector $\mathbf{c} = \sum_{i \in [\ell]} -x_i \mathbf{t}_i$ and the openings are $\mathbf{v}_i = \sum_{j \neq i} -x_j \mathbf{z}_{j,i}$.

We now compare the relative power of these two types of auxiliary information. We refer to the above auxiliary data as "Type I" auxiliary data and "Type II" auxiliary data, respectively.

- When the target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell$ are uniform, we can simulate a CRS with Type II auxiliary data from a CRS with Type I auxiliary data. Namely, given (random) matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ and the trapdoor $\mathsf{aux}_\ell = \mathbf{T}$ for $\mathbf{B}$, we can sample
$$\begin{bmatrix} \mathbf{z}_{j,1} \\ \vdots \\ \mathbf{z}_{j,\ell} \\ \hat{\mathbf{c}}_j \end{bmatrix} \leftarrow \mathbf{B}_\ell^{-1}(\mathbf{0}),$$
for each $j \in [\ell]$. By construction of $\mathbf{B}_\ell$, for all $j \in [\ell]$, $\mathbf{z}_{j,i} \in \mathbb{Z}_q^m$ is a short vector satisfying $\mathbf{A}_i \mathbf{z}_{j,i} = \mathbf{G}\hat{\mathbf{c}}_j$. When the $\mathbf{A}_i$ are sampled randomly, the precondition of Corollary 2.11 holds (see Lemma 2.2 and Corollary 2.5), and so the marginal distribution of each $\hat{\mathbf{c}}_j$ is a discrete Gaussian, and $\mathbf{G}\hat{\mathbf{c}}_j$ is uniform over $\mathbb{Z}_q^n$. Thus, we obtain a Type II CRS with matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$, target vectors $\mathbf{t}_1 = \mathbf{G}\hat{\mathbf{c}}_1, \ldots, \mathbf{t}_\ell = \mathbf{G}\hat{\mathbf{c}}_\ell$, and auxiliary data $\mathsf{aux}_\ell = \{\mathbf{z}_{j,i}\}_{i \neq j}$.

- Next, we show that we can also use Type II auxiliary data to obtain a trapdoor for *sub-matrices* of $\mathbf{B}$. We illustrate this with a concrete example. Suppose we want to obtain a trapdoor for the matrix $\mathbf{B}_k$ (where $k < \ell/m$):
$$\mathbf{B}_k = \left[\begin{array}{ccc|c} \mathbf{A}_1 & & & -\mathbf{G} \\ & \ddots & & \vdots \\ & & \mathbf{A}_k & -\mathbf{G} \end{array}\right] \in \mathbb{Z}_q^{kn \times (km+m')},$$
where $m' = n(\lfloor \log q \rfloor + 1)$. For $j \neq i$, let $\mathbf{z}_{j,i} \in \mathbb{Z}_q^m$ be short vectors where $\mathbf{A}_i \mathbf{z}_{j,i} = \mathbf{t}_j$ be the vectors in the Type II auxiliary data. For any $j > k$, consider the vector
$$\mathbf{v}_j = \begin{bmatrix} \mathbf{z}_{j,1} \\ \vdots \\ \mathbf{z}_{j,k} \\ \mathbf{G}^{-1}(\mathbf{t}_j) \end{bmatrix} \in \mathbb{Z}_q^{km+m'},$$
Observe that $\mathbf{v}_j$ is short, and moreover $\mathbf{B}_k \mathbf{v}_j = \mathbf{0}$. If $\ell - k > km + m'$, then we can collect $km + m'$ such vectors $\mathbf{v}_j$. Heuristically, if these vectors are linearly independent (over the integers), then this yields a Ajtai-basis for $\mathbf{B}_k$ (Definition 2.13). Thus Type II auxiliary data implies Type I auxiliary data for a slightly smaller dimension $k \approx \ell/m$.

While asking for security given a full trapdoor for the related matrix $\mathbf{B}_\ell$ might seem like a stronger assumption than giving our many short preimages under $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$, the above analysis shows that these these two types of auxiliary data have comparable power. Hardness of SIS/ISIS with respect to one type of auxiliary data is comparable to hardness with respect to the other (up to an $O(n \log q)$ loss in the vector dimension ℓ). In fact, the above analysis shows that Type II auxiliary data (with essentially arbitrary target vectors $\mathbf{u}_i$) is already sufficient to construct a trapdoor that yields a Type I auxiliary data for a smaller input dimension. However, the converse is not true, as the trapdoor for $\mathbf{B}_\ell$ seem to only allow sampling preimages of $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ with respect to random target vectors $\mathbf{t}_1, \ldots, \mathbf{t}_\ell$.

This distinction is important, and as we discuss in Remark 6.1, Type I auxiliary data seem essential to realizing the functional commitment scheme from Section 4 (as well as its aggregatable analog in Section 5.1).[12] Other advantages to using a Type I auxiliary data include supporting private openings (Construction 3.9) and commitments to large inputs (Constructions 3.9 and 4.14).

Remark 6.1 (Structured Targets and Functional Commitments)**.** The main verification relation of our functional commitment scheme (Construction 4.2) is $\mathbf{C} = \mathbf{A}_i\mathbf{V}_i + x_i\mathbf{G}$ where $\mathbf{A}_i = \mathbf{W}_i\mathbf{A}$. If we consider a Type II auxiliary data for this verification relation, the auxiliary data would contain $\mathbf{A}_i^{-1}(\mathbf{G})$, or equivalently, $\mathbf{A}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$. However, $\mathbf{A}^{-1}(\mathbf{W}_i^{-1}\mathbf{G})$ is a trapdoor for $\mathbf{A}$ (with tag $\mathbf{W}_i^{-1}$), which trivially breaks security. In contrast, using Type I auxiliary data does not appear to yield a trapdoor for $\mathbf{A}$, and plausibly yields a succinct functional commitment scheme.

6.1 Another View of the $\mathsf{BASIS}_{\mathsf{struct}}$ Assumption

To facilitate cryptanalysis of our new assumption, we provide an equivalent formulation of the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption (Assumption 3.3) underlying our functional, polynomial, and aggregatable commitments. Consider a variant of the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption where $\mathbf{T}$ is an Ajtai trapdoor (Definition 2.13) for $\mathbf{B}$ (i.e., $\mathbf{T} \leftarrow \mathbf{B}_s^{-1}(\mathbf{0}^{m\times 2m})$). Note that we can efficiently convert between gadget trapdoors and Ajtai trapdoors, up to small polynomial losses in the quality of the trapdoor. It is easy to see that we can re-express $\mathbf{B}^{-1}(\mathbf{0}^{m\times 2m})$ as $\mathbf{A}^{-1}(\mathbf{W}_i^{-1}\mathbf{R})$ for all $i \in [\ell]$, and $\mathbf{R} \xleftarrow{R} \mathbb{Z}_q^{n\times 2m}$. Therefore, the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption is equivalent to:

$$\textit{SIS is hard with respect to } \mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n\times m} \textit{ given } \mathbf{A}^{-1}(\mathbf{W}_i^{-1}\mathbf{R}) \textit{ for all } i \in [\ell], \textit{ where } \mathbf{W}_i \xleftarrow{R} \mathbb{Z}_q^{n\times n} \textit{ and } \mathbf{R} \xleftarrow{R} \mathbb{Z}_q^{n\times 2m}.$$

The assumption in the LWE setting. One way to understand this assumption is to consider the analogous assumption in the learning with errors (LWE) setting [Reg05]. Recall that the vanilla LWE assumption says that the distribution $(\mathbf{A}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}^\intercal)$ is pseudorandom when $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, and $\mathbf{e} \leftarrow \chi^m$, where χ is an error distribution. It is easy to see that if LWE holds with public matrix $\mathbf{A}$, then SIS is also hard with respect to $\mathbf{A}$. Correspondingly, we can then ask whether LWE is hard with respect to $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n\times m}$ given $\mathbf{A}^{-1}(\mathbf{W}_i^{-1}\mathbf{R})$ for each $i \in [\ell]$. One way to reason about this is to appeal to the "evasive LWE" assumption [Wee22, Tsa22], which roughly says that if

$$(\mathbf{A}, \mathbf{B}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}_1^\intercal, \mathbf{s}^\intercal\mathbf{B} + \mathbf{e}_2^\intercal) \overset{c}{\approx} (\mathbf{A}, \mathbf{B}, \mathbf{u}_1, \mathbf{u}_2),$$

where $\mathbf{A} \xleftarrow{R} \mathbb{Z}_q^{n\times m}$, $\mathbf{B} \in \mathbb{Z}_q^{n\times m'}$, $\mathbf{s} \xleftarrow{R} \mathbb{Z}_q^n$, $\mathbf{e}_1 \leftarrow \chi^m$, $\mathbf{e}_2 \leftarrow \chi^{m'}$, $\mathbf{u}_1 \xleftarrow{R} \mathbb{Z}_q^m$, and $\mathbf{u}_2 \xleftarrow{R} \mathbb{Z}_q^{m'}$, then

$$(\mathbf{A}, \mathbf{B}, \mathbf{s}^\intercal\mathbf{A} + \mathbf{e}_1^\intercal, \mathbf{A}^{-1}(\mathbf{B})) \overset{c}{\approx} (\mathbf{A}, \mathbf{B}, \mathbf{u}_1, \mathbf{A}^{-1}(\mathbf{B})).$$

Thus, under the evasive LWE assumption, if $\mathbf{s}^\intercal\mathbf{W}_i^{-1}\mathbf{R} + \mathbf{e}_i^\intercal$ is pseudorandom for all $i \in [\ell]$, then LWE (and correspondingly, SIS) is hard with respect to $\mathbf{A}$ even given $\mathbf{A}^{-1}(\mathbf{W}_i^{-1}\mathbf{R})$. This question itself is closely related to the question of building simpler PRFs from lattices (c.f., the discussion in [BPR12, §1.2, 1.3]). To the best of our knowledge, there are no known attacks on pseudorandomness of the above distribution.

Remark 6.2 (Parameter Choices for the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption)**.** While hardness of the $\mathsf{BASIS}_{\mathsf{rand}}$ assumption can be based on the hardness of the standard SIS assumption, we do not know of an analogous reduction for the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption. When setting parameters for the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption, we use Theorem 3.4 as a guide and consider instantiations where $n \geq \lambda$, $m \geq O(n \log q)$ and $s \geq O(\ell m \log n) = \mathsf{poly}(\lambda, \ell)$. Note that this means the quality of the basis *decreases* with the dimension. For this parameter setting, we are not aware of any concrete attacks on the $\mathsf{BASIS}_{\mathsf{struct}}$ assumption and conjecture that its security is comparable with the hardness of $\mathsf{SIS}_{n,m,q,\beta}$, with a noise bound bound $\beta = \mathsf{poly}(\lambda, \ell)$ that scales with the dimension of the vector (as in Theorem 3.4). In particular, the hardness of the SIS instance decreases with the dimension ℓ (similar to the case with q-type assumptions over groups [Che06]).

[12] In contrast, when the target vectors in the verification relation are uniform, it is straightforward to translate between the notions. For example, we refer to Remark 5.14 for an analog of the aggregatable commitment scheme (Construction 5.3) using Type II auxiliary data. In this case, both schemes satisfy same-set binding (assuming hardness of SIS/ISIS even given the auxiliary data), and neither satisfy different-set binding.

Acknowledgments

We thank Jonathan Bootle and Brent Waters for helpful discussions about this work. We thank Dario Fiore and Chris Peikert for helpful discussions on the concurrent works. D. J. Wu is supported by NSF CNS-2151131, CNS-2140975, a Microsoft Research Faculty Fellowship, and a Google Research Scholar award.

References

[ABB10a] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT*, 2010.

[ABB10b] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In *CRYPTO*, 2010.

[ACL+22] Martin R. Albrecht, Valerio Cini, Russell W. F. Lai, Giulio Malavolta, and Sri AravindaKrishnan Thyagarajan. Lattice-based SNARKs: Publicly verifiable, preprocessing, and recursively composable. In *CRYPTO*, 2022.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, 1996.

[AP09] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. In *STACS*, 2009.

[AR20] Shashank Agrawal and Srinivasan Raghuraman. Kvac: Key-value commitments for blockchains and beyond. In *ASIACRYPT*, 2020.

[BBF19] Dan Boneh, Benedikt Bünz, and Ben Fisch. Batching techniques for accumulators with applications to iops and stateless blockchains. In *CRYPTO*, 2019.

[BC12] Nir Bitansky and Alessandro Chiesa. Succinct arguments from multi-prover interactive proofs and their efficiency benefits. In *CRYPTO*, 2012.

[BCFL22] David Balbás, Dario Catalano, Dario Fiore, and Russell W. F. Lai. Functional commitments for circuits from falsifiable assumptions. *IACR Cryptol. ePrint Arch.*, 2022.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In *TCC*, 2016.

[BCTW16] Zvika Brakerski, David Cash, Rotem Tsabary, and Hoeteck Wee. Targeted homomorphic attribute-based encryption. In *TCC*, 2016.

[BDFG21] Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Halo infinite: Proof-carrying data from additive polynomial commitments. In *CRYPTO*, 2021.

[BDN18] Dan Boneh, Manu Drijvers, and Gregory Neven. Compact multi-signatures for smaller blockchains. In *ASIACRYPT*, 2018.

[BFS20] Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent snarks from DARK compilers. In *EUROCRYPT*, 2020.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT*, 2014.

[BGV11] Siavosh Benabbas, Rosario Gennaro, and Yevgeniy Vahlis. Verifiable delegation of computation over large datasets. In *CRYPTO*, 2011.

[BNO21] Dan Boneh, Wilson Nguyen, and Alex Ozdemir. Efficient functional commitments: How to commit to private functions. *IACR Cryptol. ePrint Arch.*, 2021.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, 2012.

[BTVW17] Zvika Brakerski, Rotem Tsabary, Vinod Vaikuntanathan, and Hoeteck Wee. Private constrained prfs (and more) from LWE. In *TCC*, 2017.

[BV15] Zvika Brakerski and Vinod Vaikuntanathan. Constrained key-homomorphic prfs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In *TCC*, 2015.

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-shamir: from practice to theory. In *STOC*, pages 1082–1090, 2019.

[CF13] Dario Catalano and Dario Fiore. Vector commitments and their applications. In *PKC*, 2013.

[CFG+20] Matteo Campanelli, Dario Fiore, Nicola Greco, Dimitris Kolonelos, and Luca Nizzardo. Incrementally aggregatable vector commitments and applications to verifiable decentralized storage. In *ASIACRYPT*, 2020.

[CFM08] Dario Catalano, Dario Fiore, and Mariagrazia Messina. Zero-knowledge sets with short proofs. In *EUROCRYPT*, 2008.

[Che06] Jung Hee Cheon. Security analysis of the strong diffie-hellman problem. In *EUROCRYPT*, 2006.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, 2010.

[CHM+20] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas P. Ward. Marlin: Preprocessing zksnarks with universal and updatable SRS. In *EUROCRYPT*, 2020.

[CJJ21] Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. SNARGs for $\mathcal{P}$ from LWE. In *FOCS*, 2021.

[COS20] Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. In *EUROCRYPT*, 2020.

[CPSZ18] Alexander Chepurnoy, Charalampos Papamanthou, Shravan Srinivasan, and Yupeng Zhang. Edrax: A cryptocurrency with stateless transaction validation. *IACR Cryptol. ePrint Arch.*, 2018.

[dCP23] Leo de Castro and Chris Peikert. Functional commitments for all functions, with transparent setup. In *EUROCRYPT*, 2023.

[DGKV22] Lalita Devadas, Rishab Goyal, Yael Kalai, and Vinod Vaikuntanathan. Rate-1 non-interactive arguments for batch-np and applications. *IACR Cryptol. ePrint Arch.*, 2022.

[DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT*, 2004.

[FSZ22] Nils Fleischhacker, Mark Simkin, and Zhenfei Zhang. Squirrel: Efficient synchronized multi-signatures from lattices. In *ACM CCS*, 2022.

[GGH96] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Collision-free hashing from lattice problems. *IACR Cryptol. ePrint Arch.*, page 9, 1996.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, 2008.

[GRWZ20] Sergey Gorbunov, Leonid Reyzin, Hoeteck Wee, and Zhenfei Zhang. Pointproofs: Aggregating proofs for multiple vector commitments. In *ACM CCS*, 2020.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO*, 2013.

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In *STOC*, 2015.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *STOC*, 2011.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. PLONK: permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. *IACR Cryptol. ePrint Arch.*, 2019.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4), 1999.

[IKO07] Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. Efficient arguments without short PCPs. In *CCC*, 2007.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *STOC*, 1992.

[KZG10] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *ASIACRYPT*, 2010.

[LLNW16] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *EUROCRYPT*, 2016.

[LM19] Russell W. F. Lai and Giulio Malavolta. Subvector commitments with application to succinct arguments. In *CRYPTO*, 2019.

[LP20] Helger Lipmaa and Kateryna Pavlyk. Succinct functional commitment for a large class of arithmetic circuits. In *Advances in Cryptology - ASIACRYPT 2020, Part III*, 2020.

[LRY16] Benoît Libert, Somindu C. Ramanna, and Moti Yung. Functional commitment schemes: From polynomial commitments to pairing-based accumulators from simple assumptions. In *ICALP*, 2016.

[LW15] Vadim Lyubashevsky and Daniel Wichs. Simple lattice trapdoor sampling from a broad class of distributions. In *PKC*, 2015.

[LY10] Benoît Libert and Moti Yung. Concise mercurial vector commitments and independent zero-knowledge sets with short proofs. In *TCC*, 2010.

[MBKM19] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge snarks from linear-size universal and updatable structured reference strings. In *ACM CCS*, 2019.

[Mer87] Ralph C. Merkle. A digital signature based on a conventional encryption function. In *CRYPTO*, 1987.

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4), 2000.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, 2012.

[MR04] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. In *FOCS*, 2004.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In *CRYPTO*, 2003.

[Nit21] Anca Nitulescu. SoK: Vector commitments, 2021. https://www.di.ens.fr/~nitulesc/files/vc-sok.pdf.

[Pei07] Chris Peikert. Limits on the hardness of lattice problems in ℓ_p norms. In *CCC*, 2007.

[PPS21] Chris Peikert, Zachary Pepin, and Chad Sharp. Vector and functional commitments from lattices. In *TCC*, 2021.

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In *CRYPTO*, pages 89–114, 2019.

[PSTY13] Charalampos Papamanthou, Elaine Shi, Roberto Tamassia, and Ke Yi. Streaming authenticated data structures. In *EUROCRYPT*, 2013.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, 2005.

[Res22] Protocol Labs Research. Vector commitment research day, 2022. https://cryptonet.vercel.app/.

[RMCI17] Leonid Reyzin, Dmitry Meshkov, Alexander Chepurnoy, and Sasha Ivanov. Improving authenticated dynamic dictionaries, with applications to cryptocurrencies. In *FC*, 2017.

[TAB+20] Alin Tomescu, Ittai Abraham, Vitalik Buterin, Justin Drake, Dankrad Feist, and Dmitry Khovratovich. Aggregatable subvector commitments for stateless cryptocurrencies. In *SCN*, 2020.

[Tsa22] Rotem Tsabary. Candidate witness encryption from lattice techniques. In *CRYPTO*, 2022.

[TXN20] Alin Tomescu, Yu Xia, and Zachary Newman. Authenticated dictionaries with cross-incremental proof (dis)aggregation. *IACR Cryptol. ePrint Arch.*, 2020.

[Wee22] Hoeteck Wee. Optimal broadcast encryption and CP-ABE from evasive lattice assumptions. In *EUROCRYPT*, 2022.