Beyond LWE: a Lattice Framework for Homomorphic Encryption

Alberto Leporati[1], Lorenzo Rovida[2,⋆], and Wessel van Woerden[3]

[1] University of Milano-Bicocca, DISCo, Milan, Italy
alberto.leporati@unimib.it
[2] Politecnico di Torino, Turin, Italy
lorenzo.rovida@polito.it
[3] PQShield, Amsterdam, the Netherlands
wessel.vanwoerden@pqshield.com

Abstract. We propose a generalization of homomorphic encryption (HE) schemes from a purely geometrical and lattice-based perspective. Most of the current reference HE schemes are based on the Learning with Errors (LWE) problem or structured versions thereof. In this proposal, we first investigate LWE-based cryptosystems from a lattice point of view and present a framework that allows to obtain the same result, in geometrical terms, from any lattice – as long as it contains a sufficiently short trapdoor vector. More precisely, we generalize the classical BGV (Brakerski, Gentry and Vaikuntanathan, ITCS '12) and GSW (Gentry, Sahai and Waters, CRYPTO '14) schemes to purely lattice-based variants, which we call Lattice-BGV and Lattice-GSW. By abstracting away the particular hardness assumption, our lattice framework allows to be instantiated with a broader range of lattice families and hardness assumptions. For example, LWE gives a natural trapdoor for random q-ary lattices, and when plugged into our framework one obtains the original BGV and GSW schemes, while in this work we will also consider an instantiation based on the Lattice Isomorphism Problem (LIP), leading to the first more advanced cryptographic scheme build from LIP [4]. Our framework also gives a geometrical and natural explanation of HE procedures and generalizes some properties, such as message packing, the ability to store many messages in a single ciphertext, without relying on any particular algebraic structure.

Keywords: Homomorphic encryption · Lattice-based cryptography · Lattice isomorphism problem

⋆ This work has been carried out while the author was a student at the University of Milan-Bicocca.

[4] In a concurrent work Branco, Malavolta and Maradni [BMM25] propose an alternative LIP-based FHE construction.

Contents

1 Introduction **3**

2 Preliminaries **6**
2.1 Lattices and hard problems 7
2.2 Public Key Encryption scheme 9
2.3 Original BGV cryptosystem 10
2.4 Original GSW cryptosystem 12

3 A lattice framework for HE **14**
3.1 Interpreting LWE from a lattice perspective 14
3.2 Generalizing key generation: a (trapdoor) lattice 16
3.3 The ambient space 18

4 Abstract HE cryptosystems from lattices **19**
4.1 Lattice-BGV 19
4.2 Concrete parameters and requirements 26
4.3 Lattice-GSW 27

5 An instantiation using LIP **29**

6 Further extensions and open questions **30**

A IND-CPA security proof **35**
A.1 NTRU and LWE-based instantiations 36
A.2 Security proof under LIP instantiation 36

B Ciphertext compression and small-secret LWE **37**
B.1 Compressed ambient space 37
B.2 Connections with short secret LWE 39
B.3 Connections and differences with structural lattice reduction 41

C Proofs of lemmas **42**
C.1 Proof of Lemma 4.2 42
C.2 Proof of Corollary 4.1 42
C.3 Proof of Lemma 4.3 43

D More details on key switching **43**
D.1 Original BGV key switching 43
D.2 Key switching in Lattice-BGV 44

E Connections with [BMM25] **46**

1 Introduction

Originally referred to as *privacy homomorphisms*, the theoretical concept of homomorphic encryption (HE) was introduced in 1978 by Rivest, Adleman and Dertouzos [RAD78]. A homomorphic encryption scheme allows for computations to be performed on encrypted data without having access to the secret key. The first proposal of an actual *fully homomorphic encryption* (FHE) scheme able to evaluate circuits of any depth, was presented by Gentry in 2009 in its breakthrough work [Gen09], where the *bootstrapping* operation was introduced. Years later, many FHE constructions followed and improved this blueprint.

Today it is possible to find at least two main approaches to instantiate a fully homomorphic encryption scheme, both with security reductions to hard lattice problems: the BGV-like [BGV12] (then evolved towards, e.g., BFV [Bra12,FV12] and CKKS [CKKS17]) and the DM/CGGI-like [DM15,CGGI16] (evolved towards TFHE [CGGI20]) approach, based on GSW [GSW13].

Both these approaches are based on the same hardness assumption, that is the Learning with Errors (LWE) problem [Reg05] or structured versions thereof. Indeed, most of the more popular FHE schemes share the same key generation procedure [BGGJ20]. This begs the question: is LWE required to instantiate a HE scheme or can we also use other lattice assumptions? Or related, what are the minimum requirements to instantiate it? For the sake of clarity, we will make no difference between HE and FHE as, for the appropriate parameters, our HE schemes can be naturally extended to FHE schemes by a bootstrapping procedure.

A lattice framework for (fully) homomorphic encryption. To answer these questions we go back to the core of these LWE-based HE schemes to better understand their workings. Since LWE can be seen as a lattice problem, we will take a lattice perspective to describe them. This allows to have some geometrical intuition on both the key generation, the encryption/decryption, and the other FHE specific procedures. In particular, it makes clear what *lattice properties and algorithms* these schemes are actually using, and what the requirements are for the lattices involved.

Following this perspective, we see that the LWE assumption is used in two ways for the underlying encryption scheme. Firstly, in the key generation, it is used to create a random q-ary lattice $q\mathbb{Z}^m \subset \mathcal{L} \subset \mathbb{Z}^m$ with a planted secret short vector $\boldsymbol{s} \in \mathcal{L}$. Secondly, for encryption, a message is encoded in a small error $\boldsymbol{e} \in \mathbb{Z}^m$ which is then hidden as a Bounded Distance Decoding (BDD) target $\boldsymbol{t} \in \boldsymbol{e} + q\mathcal{L}^*$ in the dual lattice $\mathcal{L}^*$ scaled up by q. By the LWE assumption, this BDD target is indistinguishable from a uniform target and thus the encoded message is securely hidden. To decrypt, one then uses the secret short vector $\boldsymbol{s} \in \mathcal{L}$ to partially decode the target, as $\langle \boldsymbol{t}, \boldsymbol{s} \rangle \equiv \langle \boldsymbol{e}, \boldsymbol{s} \rangle \bmod q$. Since $\langle \boldsymbol{e}, \boldsymbol{s} \rangle \in \mathbb{Z}$ is small, this value is recovered exactly, and we obtain information about the error $\boldsymbol{e}$ which, if well encoded, allows to recover the message.

It is folklore that the above strategy can also be generalized to construct a simple encryption scheme using different lattice families with different underlying

security assumptions. It turns out however that these minimal requirements are also sufficient for HE. To support all common HE primitives, one only requires the key generation to produce a *public lattice* $\mathcal{L}$, for which decoding in the dual lattice $\mathcal{L}^*$ is hard, along with a *secret short trapdoor vector* $\boldsymbol{s} \in \mathcal{L}$. We demonstrate this by presenting a *lattice framework* for HE. In particular, we present an explicit generalization of the classical BGV and GSW schemes using only this public lattice and the secret short trapdoor vector. This is a proper generalization: if instantiated with an LWE-based key generation, we obtain the original schemes.

A high-level view on the homomorphic operations. Recall that ciphertexts can be seen as BDD targets with respect to the lattice $q\mathcal{L}^*$. An important step in this generalization is the discretized ambient space $\mathcal{D} \supset q\mathcal{L}^*$ in which these BDD targets live. We show that it is sufficient that $\mathcal{L}^* \subset \mathcal{D} \subset q\mathcal{L}^*$ for some modulus q and such that we have a relatively good public basis D of $\mathcal{D}$. Note that these properties generalize the q-ary integer lattices and their orthogonal ambient space $\mathbb{Z}^m$. The former creates a ciphertext space $\mathcal{C} := \mathcal{D}/q\mathcal{L}^* \cong G \subset \mathbb{Z}^m/q\mathbb{Z}^m$ whose group structure is required to make the homomorphic operations succeed, while the good basis D is required to sample short errors $\boldsymbol{e} \in \mathcal{D}$ (discrete Gaussian sampling) and for modulus switching (decoding to the ambient space). One way to achieve this is to pick $\mathcal{D} := \mathcal{L}^*$ with D equal to its public basis and the modulus q sufficiently large (to make the basis D relatively good). We also demonstrate a more efficient construction, which further compresses the ambient space, which could be interpreted as a generalization of small-secret LWE.

With the ambient and ciphertext space settled we show that also in this more general setting, multiplications can naturally be performed by a tensor product $\boldsymbol{c}_1 \otimes \boldsymbol{c}_2 \in \mathcal{C} \otimes \mathcal{C}$ between ciphertexts or equivalently their errors $\boldsymbol{e}_1 \otimes \boldsymbol{e}_2$, just as for BGV. Given the right encoding of the message space $\mathcal{M}$ into errors this leads to a tensor product of the messages $\boldsymbol{m}_1 \otimes \boldsymbol{m}_2 \in \mathcal{M} \otimes \mathcal{M}$ (which, e.g., for the special case $\mathcal{M} \cong \mathbb{Z}/2\mathbb{Z}$ is a binary multiplication $\mu_1 \cdot \mu_2$). On the lattice side this means we move from a secret trapdoor vector $\boldsymbol{s} \in \mathcal{L}$ and the q-dual $q\mathcal{L}^*$ to the trapdoor vector $\boldsymbol{s} \otimes \boldsymbol{s} \in \mathcal{L} \times \mathcal{L}$ and the q-dual $q(\mathcal{L}^* \otimes \mathcal{L}^*)$.

We show how the key switching procedure allows us to move from this large-dimensional tensored space back to any new keypair $\boldsymbol{s}', \mathcal{L}'$ and ambient space $\mathcal{D}'$ using the same modulus, along with maintaining (part of) the message $\boldsymbol{m}_1 \otimes \boldsymbol{m}_2 \in \mathcal{M} \otimes \mathcal{M}$.

To decrease the absolute decoding errors we also demonstrate a general modulus switching technique from one ambient space $q_1\mathcal{L}^* \subset \mathcal{D}_1 \subset \mathcal{L}^*$ to a relatively sparser ambient space $q_1\mathcal{L}^* \subset (q_1/q_2)\mathcal{D}_2 \subset (q_1/q_2)\mathcal{L}^*$ where $q_2 \mid q_1$. We show that this procedure is precisely an (approximate) decoding problem from $\mathcal{D}_1$ to the sparser ambient space $(q_1/q_2)\mathcal{D}_2$, which we can perform using its good basis D_2. Some extra care has to be taken depending on the message encoding. After scaling down by q_1/q_2 the new error becomes $(\boldsymbol{e}+\boldsymbol{\delta})/(q_1/q_2)$ where $\boldsymbol{e}$ was the old ciphertext error, and $\boldsymbol{\delta}$ the ambient space rounding error. This also shows that

an ambient space with better decoding properties could lead to better modulus switching.

Together these operations form the basis of our Lattice-BGV framework. We also consider the GSW approach which uses a redundant encoding of the messages into multiple BGV ciphertexts at the benefit of a natural matrix-multiplication that avoids the large increase of dimension, leading to the Lattice-GSW framework.

The Lattice Isomorphism Problem Our lattice framework can directly be instantiated with lattices constructed from LWE, NTRU and related assumptions. However, a new assumption of interest that fits well within our framework is the Lattice Isomorphism Problem (LIP), which was recently introduced by Ducas and van Woerden [DvW22] and Bennett, Ganju, Peetathawatchai and Stephens-Davidowitz [BGPSD23]. Two lattices $\mathcal{L}_1, \mathcal{L}_2 \subset \mathbb{R}^n$ are called *isomorphic* if there exists a linear isometry mapping one to the other, i.e., if there exists an orthonormal transformation $O \in \mathcal{O}_m(\mathbb{R})$ such that $\mathcal{L}_2 = O \cdot \mathcal{L}_1 = \{O\boldsymbol{v} : \boldsymbol{v} \in \mathcal{L}_1\}$. LIP then asks, given two isomorphic lattices, to compute such an orthonormal transformation mapping one lattice to the other. The main idea is that one can use such an an isometry to hide the structure of a lattice.

Since the usage of LIP as a hardness assumption is relatively new, there are at this moment only LIP-based schemes for basic cryptographic primitives, like identification, encryption and signature schemes [DvW22,BGPSD23,DPPW22]. Given our lattice framework, we can however also construct a HE scheme based on LIP in a straightforward way. For this we can pick any lattice $\mathcal{L}'$ with some known sufficiently short vector $\boldsymbol{s}' \in \mathcal{L}'$. Then we hide the known structure by considering a bad public basis of a lattice $\mathcal{L} := O \cdot \mathcal{L}'$ isomorphic to $\mathcal{L}'$ by some *secret isometry* $O \in \mathcal{O}_m(\mathbb{R})$. Given the secret isometry O one also knows a secret short vector $\boldsymbol{s} = O \cdot \boldsymbol{s}' \in \mathcal{L}$. Informally, with only the bad description of $\mathcal{L}$, and being hard to recover the isometry, it is typically hard to decode a BDD instance in $q\mathcal{L}^*$ and therefore the scheme is secure. This can be formally reduced to a distinguishing-LIP assumption as in [DvW22], although we also present a more natural IND-CPA security proof based on two assumptions inspired by results from [vW24].

Related works. Highly related is the work by Gama, Izabachene, Nguyen and Xie [GINX16] on *structural lattice reduction*. This work makes the generalization of LWE from a decoding problem in $\mathbb{Z}^m/\mathcal{L} \cong \mathbb{Z}_q^n$, to one in $\mathbb{Z}^m/\mathcal{L} \cong G$ for any finite abelian group G. Furthermore, they show how to construct a GSW-like scheme based on GLWE a lá [AP14,DM15]. While this is a significant generalisation of LWE, and there is an overlap in our techniques, our framework works for more general lattice families and does not restrict itself to such integer lattices. We provide an extensive comparison of the techniques in Appendix B.3.

In a concurrent work Branco, Malavolta and Maradni [BMM25] propose an FHE scheme based on the hardness of LIP. While our work is aimed at a more

general framework, we do give a comparison between their results and a LIP-based instantiation of our framework in Appendix E. Overall, we conclude that the approach and goals are somewhat orthogonal.

Structure of the paper. The rest of the paper is structured as follows. Section 2 will contain all the required preliminaries from lattices and the BGV/GSW definitions. Section 3 will introduce the lattice perspective on the fundamental components of the previous schemes, namely the key generation and encryption/decryption procedures. Section 4 will introduce a pure lattice-based version of both BGV and GSW. Section 5 will present a possible LIP-based instantiation of previously introduced schemes. Eventually, in Section 6, we consider some further extensions and open questions.

Acknowledgements. A large part of this work was carried out while W. van Woerden was employed, and while L. Rovida was hosted, by IMB at the University of Bordeaux. There, they were both supported by the CHARM ANR-NSF grant (ANR-21-CE94-0003).

2 Preliminaries

In what follows, if not stated otherwise, lowercase letters are used to indicate numbers (e.g. a), bold-italic lowercase letters to indicate vectors (e.g. $\boldsymbol{v}$), and uppercase letters are used to indicate matrices (e.g. A).

Vectors are considered as column vectors, the i-th column of a matrix A is referred to as $\boldsymbol{a}_i$ and the i-th element of a vector $\boldsymbol{v}$ as v_i. The symbol $\mathbb{Z}_q$ refers to the cyclic group $\mathbb{Z}/q\mathbb{Z}$ of integers modulo q, which we identify with $[-q/2, \dots, q/2) \cap \mathbb{Z}$.

The general linear group of invertible matrices with integer (or real) entries of dimension m is referred to as $\mathrm{GL}_m(\mathbb{Z})$ (or $\mathrm{GL}_m(\mathbb{R})$, respectively). The group of orthonormal matrices of dimension m is referred to as $\mathcal{O}_m(\mathbb{R})$. Given a matrix B composed of vectors $\boldsymbol{b}_i$, we write $\|B\| := \max_i \|\boldsymbol{b}_i\|$, and will refer to its Gram-Schmidt orthogonalization as $\tilde{B}$ and to the Gram-Schmidt vectors as $\tilde{\boldsymbol{b}}_i$.

The notation $a \sim \chi$ indicates a value a sampled from the probability distribution χ. The notation $[a]_q : \mathbb{Z} \to \mathbb{Z}_q$ indicates the modulo reduction $a \bmod q$ in the interval $[-q/2, q/2)$. The notation $\lfloor a \rceil_p : \mathbb{Z}_q \to \{0, \dots, p-1\}$, indicates a function that returns the nearest integer to $a \cdot p/q$, namely $\lfloor a \rceil_p := \lfloor a \cdot p/q + 1/2 \rfloor \bmod p$.

We write $\log := \log_2$ for the logarithm function with base two, i.e., $\log(2^k) = k$. Let $\mathtt{Bits}(a) : \mathbb{Z}_q \to \{0,1\}^{\log(q)} \subset \mathbb{Z}_q^{\log(q)}$ be the bit decomposition of a, least to most significant bit, and let $\mathtt{Pow}(a) : \mathbb{Z}_q \to \mathbb{Z}_q^{\log(q)} = (a, 2a, 4a, \dots, 2^{\log(q)-1} \cdot a)$. By abuse of notation we allow both functions to be applied to vectors by evaluating the corresponding function independently on each entry, giving functions $\mathtt{Bits}, \mathtt{Pow} : \mathbb{Z}_q^n \to \mathbb{Z}_q^{n \log(q)}$.

2.1 Lattices and hard problems

For a basis $B \in \mathrm{GL}_m(\mathbb{R})$ we define the lattice $\mathcal{L}(B) \subset \mathbb{R}^m$ of dimension m generated by B as

$$\mathcal{L}(B) := B \cdot \mathbb{Z}^m = \{B\boldsymbol{x} : \boldsymbol{x} \in \mathbb{Z}^m\}.$$

Two bases $B, B' \in \mathrm{GL}_m(\mathbb{R})$ generate the same lattice if and only if there exists a unimodular matrix $U \in \mathrm{GL}_m(\mathbb{Z})$ such that $B' = B \cdot U$. The volume $\mathrm{vol}(\mathcal{L})$ of a lattice $\mathcal{L} = \mathcal{L}(B) \subset \mathbb{R}^m$ is given by $\mathrm{vol}(\mathbb{R}^m/\mathcal{L})$ and equals $|\det(B)|$ for any basis of the lattice. The first minimum is the length of the shortest non-zero vector of the lattice $\mathcal{L}$ and is denoted by $\lambda_1(\mathcal{L}) = \min \|\boldsymbol{v}\|$ for all nonzero $\boldsymbol{v} \in (\mathcal{L} \setminus \mathbf{0})$. A q-ary lattice is a lattice $\mathcal{L}$ that satisfies $q\mathbb{Z}^m \subset \mathcal{L} \subset \mathbb{Z}^m$ for some modulus $q \geq 2$. Given any (primal) lattice $\mathcal{L}$, it is possible to define its *dual* $\mathcal{L}^*$ as follows:

Definition 2.1 (Dual lattice). *Given a lattice $\mathcal{L}$, its dual is defined as the lattice $\mathcal{L}^*$ of all the vectors $\boldsymbol{x} \in \mathrm{span}_{\mathbb{R}}(\mathcal{L})$ such that $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \in \mathbb{Z}$ for all $\boldsymbol{y} \in \mathcal{L}$.*

For a lattice $\mathcal{L}(B)$ generated by a square basis $B \in \mathrm{GL}_m(\mathbb{R})$, its dual is generated by the basis B^{-t}, namely $\mathcal{L}(B)^* = \mathcal{L}(B^{-t})$. For a lattice $\mathcal{L}$ and any scalar $q \in \mathbb{R}_{>0}$ we will for convenience call $q\mathcal{L}^*$ the *q-dual* of $\mathcal{L}$. In particular, the q-dual of a q-ary lattice is again a q-ary lattice.

Two lattices $\mathcal{L}, \mathcal{L}'$ are said to be *isomorphic* if there exists an orthonormal transformation $O \in \mathcal{O}_m(\mathbb{R})$ such that $\mathcal{L}' = O \cdot \mathcal{L}$. This relation allows to introduce the *Lattice Isomorphism Problem* (LIP).

Definition 2.2 (Lattice Isomorphism Problem (LIP) - search version). *Given two isomorphic lattices $\mathcal{L}, \mathcal{L}' \subset \mathbb{R}^m$, find an orthonormal transformation $O \in \mathcal{O}_m(\mathbb{R})$ such that $\mathcal{L}' = O \cdot \mathcal{L}$.*

Since two bases generate the same lattice up to some unimodular transformation U, the Lattice Isomorphism Problem can also be stated in terms of the bases. Given that two bases $B', B \in \mathrm{GL}_m(\mathbb{R})$ generate the same lattice if there exists a unimodular transformation $U \in \mathrm{GL}_m(\mathbb{Z})$ such that $B' = B \cdot U$, LIP asks to find an orthonormal $O \in \mathcal{O}_m(\mathbb{R})$ and a unimodular $U \in \mathrm{GL}_m(\mathbb{Z})$ such that $B' = O \cdot B \cdot U$. The presence of both the orthonormal and the unimodular transformations is what makes this problem hard.

Lastly, we introduce two variants of the lattice decoding problem, along with the Learning with Errors (LWE) problem by Regev [Reg05].

Definition 2.3 (α-Bounded Distance Decoding (BDD_α)). *Let $0 \leq \alpha < \frac{1}{2}$, $\mathcal{L}$ a lattice, and $\boldsymbol{t} = \boldsymbol{v} + \boldsymbol{e}$ where $\boldsymbol{v} \in \mathcal{L}$ and $\|\boldsymbol{e}\| < \alpha\lambda_1(\mathcal{L})$. The BDD_α problem asks: given a basis B of $\mathcal{L}$ and the target $\boldsymbol{t} = \boldsymbol{v} + \boldsymbol{e}$, compute the closest lattice point $\boldsymbol{v}$ and the error $\boldsymbol{e}$.*

Throughout the paper we will ignore the particular promise factor α and simply talk about a BDD instance $(\boldsymbol{t}, \mathcal{L})$ whenever $\mathrm{dist}(\boldsymbol{t}, \mathcal{L}) < \frac{1}{2}\lambda_1(\mathcal{L})$.

Note that the error computed in the BDD problem is invariant under translation of the target by a lattice point, i.e., any $\boldsymbol{t} \in \boldsymbol{e} + \mathcal{L}$ will decode to the same

error. In certain cases one is only interested in this small error and it is sufficient to receive as input any *syndrome* $\boldsymbol{t} \equiv \boldsymbol{e} \in \text{span}(\mathcal{L})/\mathcal{L}$ and we call recovering $\boldsymbol{e}$ the *syndrome decoding problem* (SDP). Typically, when a (public) basis B is clear from the context we identify $\text{span}(\mathcal{L})/\mathcal{L}$ with the fundamental parallelepiped $\mathcal{P}(B) := B \cdot [-\frac{1}{2}, \frac{1}{2})^m$ and thereby assume a canonical representative as input. If the error is not necessarily small enough to allow for unique decoding one can also consider the *approximate syndrome decoding problem* (approxSDP) where the goal is merely to return a small enough error in the coset.

Definition 2.4 (Learning with Errors ($\text{LWE}_{n,m,q,\chi}$)). *For a security parameter λ, let $n = n(\lambda)$ be an integer dimension, let $q = q(\lambda) \geq 2$ be an integer, let $m = \mathcal{O}(n \log(q))$ be the number of samples, and let $\chi = \chi(\lambda)$ be a distribution over $\mathbb{Z}$. The $\text{LWE}_{n,m,q,\chi}$ problem is to distinguish with non-negligible probability (in λ) the following two distributions both returning $(A, \boldsymbol{b})$: in the first one draws $A, \boldsymbol{b}$ uniformly from $\mathbb{Z}_q^{m \times n}$ and $\mathbb{Z}_q^m$ respectively. In the second distribution one first draws $\boldsymbol{x} \in \mathbb{Z}_q^n$ uniformly, then samples A uniformly from $\mathbb{Z}_q^{m \times n}$ and $\boldsymbol{s} \in \mathbb{Z}_q^m$ with $s_i \sim \chi$, and sets $\boldsymbol{b} = A\boldsymbol{x} + \boldsymbol{s} = (\langle a_i, \boldsymbol{s} \rangle + s_i \bmod q)_i$. The $\text{LWE}_{n,q,m,\chi}$ assumption is that the $\text{LWE}_{n,m,q,\chi}$ problem is infeasible.*

We emphasize and stress that an LWE instance is defined using the notation introduced above, namely $\boldsymbol{b} = A\boldsymbol{x} + \boldsymbol{s} \bmod q$, where the "error" $\boldsymbol{s} \in \mathbb{Z}_q^m$ is a small vector from the distribution χ. Although not standard, this notation will allow to have a similar notation between the original LWE-based schemes and our generalizations. We will later see that recovering the $(\boldsymbol{x}, \boldsymbol{s})$ pair from an LWE instance is essentially a syndrome decoding problem in a random q-ary lattice.

Discrete Gaussian distributions. The Gaussian function $\rho_{\sigma,\boldsymbol{c}} : \mathbb{R}^m \to (0, 1]$ with width $\sigma \in \mathbb{R}$, centered at $\boldsymbol{c} \in \mathbb{R}^m$ (if not specified, $\boldsymbol{c} = \boldsymbol{0}$) is defined as $\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x}) := \exp(-\pi \|\boldsymbol{x} - \boldsymbol{c}\|^2 / \sigma^2)$. The Gaussian function can be used to define the continuous Gaussian distribution, however we restrict its domain to a lattice and obtain the *discrete Gaussian distribution.*

Definition 2.5 (Discrete Gaussian distribution). *For a lattice $\mathcal{L} \subset \mathbb{R}^m$, center $\boldsymbol{c} \in \mathbb{R}^m$, Gaussian width $\sigma > 0$, and for all $\boldsymbol{x} \in \mathcal{L}$ the discrete Gaussian distribution $\mathcal{G}_{\mathcal{L},\sigma,\boldsymbol{c}}$ is defined by:*

$$\Pr_{X \sim \mathcal{G}_{\mathcal{L},\sigma,\boldsymbol{c}}}[X = \boldsymbol{x}] := \frac{\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})}{\rho_{\sigma,\boldsymbol{c}}(\mathcal{L})}, \text{ where } \rho_{\sigma,\boldsymbol{c}}(\mathcal{L}) := \sum_{\boldsymbol{c} \in \mathcal{L}} \rho_{\sigma,\boldsymbol{c}}(\boldsymbol{c}).$$

We write $\mathcal{G}_{\mathcal{L},\sigma} := \mathcal{G}_{\mathcal{L},\sigma,\boldsymbol{0}}$ and $\mathcal{G}_{\sigma,\boldsymbol{c}} := \mathcal{G}_{\mathbb{Z},\sigma,\boldsymbol{c}}$. Given a basis B of a lattice $\mathcal{L}$ we can efficiently sample from the discrete Gaussian distribution over $\mathcal{L}$ for a Gaussian width σ that is slightly larger than the Gram-Schmidt norm $\|\tilde{B}\|$. In particular, $\|\tilde{B}\| \leq \|B\|$ so a short basis allows to sample somewhat short lattice vectors.

Lemma 2.1 (from [BLP+13, Lemma 2.3])**.** *There is a probabilistic polynomial time algorithm that, given a basis B of a lattice $\mathcal{L}(B)$, a center $\boldsymbol{c} \in \mathbb{R}^m$ and a parameter $\sigma \geq \|\tilde{B}\| \cdot \sqrt{\ln(2m + 4)/\pi}$, returns a sample distributed as $\mathcal{G}_{\mathcal{L}(B),\sigma,\boldsymbol{c}}$.*

2.2 Public Key Encryption scheme

While we consider Homomorphic Encryption (HE) schemes in this work, we will only consider the underlying plain Public Key Encryption (PKE) schemes for their security.

Definition 2.6 (PKE Scheme). *A* Public Key Encryption *scheme Π consists of a message space $\mathcal{M}$, a public key space $\mathcal{K}_p$, a secret key space $\mathcal{K}_s$, a ciphertext space $\mathcal{C}$, and three probabilistic polynomial-time (PPT) algorithms* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *such that* $\mathsf{Gen} : \{0,1\}^\lambda \to \mathcal{K}_p \times \mathcal{K}_s$ *(key generation),* $\mathsf{Enc} : \mathcal{K}_p \times \mathcal{M} \to \mathcal{C}$ *(encryption) and* $\mathsf{Dec} : \mathcal{K}_s \times \mathcal{C} \to \mathcal{M} \cup \{\perp\}$*, such that for any* $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$ *and* $m \in \mathcal{M}$ *it holds that* $\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m)) = m$ *with overwhelming probability in* λ *(correctness).*

For an HE scheme one has to add a (limited) evaluation function that can act on ciphertext with an optional evaluation key. In this work the evaluation key will have the form of encryptions of (an encoding of) the secret key. For simplicity we will assume that these encryptions are performed under a fresh keypair, so as not have to consider any circular security assumptions. As a result we can also simply consider the IND-CPA security of the underlying encryption scheme.

Definition 2.7 (IND-CPA Security). *A PKE scheme Π has* indistinguishable encryptions under a chosen-plaintext attack (IND-CPA) *if for all PPT adversaries $\mathcal{A}$, the advantage*

$$\mathbf{Adv}^{ind\text{-}cpa}_{\mathcal{A},\Pi}(\lambda) := \left|\Pr[PubK^{ind\text{-}cpa\text{-}1}_{\mathcal{A},\Pi}(\lambda) = 1] - \Pr[PubK^{ind\text{-}cpa\text{-}0}_{\mathcal{A},\Pi}(\lambda) = 1]\right|$$

is a negligible function of λ. The experiment $PubK^{ind\text{-}cpa\text{-}b}_{\mathcal{A},\Pi}(\lambda)$ is defined as:

Experiment *$PubK^{ind\text{-}cpa\text{-}b}_{\mathcal{A},\Pi}(\lambda)$ for $b \in \{0,1\}$*

1. $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$.
2. $\mathcal{A}(pk)$ *outputs a pair of messages* $(m_0, m_1) \in \mathcal{M} \times \mathcal{M}$.
3. $c^* \leftarrow \mathsf{Enc}_{pk}(m_b)$.
4. $b' \leftarrow \mathcal{A}(c^*)$.
5. *The experiment's output is* 1 *if* $b' = b$*, and* 0 *otherwise.*

We observe that, since Li and Micciancio attack [LM21], it is required for HE schemes to also answer to a more advanced security model, that is IND-CPA$^{\mathrm{D}}$. While discussing such definition is out of scope for this work – although we refer to, e.g., [CCP+24] – we believe that existing countermeasures for LWE-based schemes can easily be ported in our framework as well. For simplicity, we can assume that either the schemes are instantiated with sets of parameters such that decryption errors happen with negligible probability, or that decryption results are not shared with any possible attacker.

2.3 Original BGV cryptosystem

Started by the Brakerski-Gentry-Vaikuntanathan (BGV) cryptosystem [BGV12], the BGV-like type of schemes allows for practical homomorphic circuit evaluation based on (Ring-)LWE. The adopted "leveled" approach allows to evaluate circuits of a fixed multiplicative depth – or levels – depending on the selected set of parameters.

We describe the public-key version of the BGV cryptosystem in $\mathrm{LWE}_{n-1,m,q,\chi}$ form. Although the ring version based on Ring-LWE [LPR10] is more efficient, we focus on the plain variant to simplify the geometric interpretation we will give later – although similar arguments can be applied by considering RLWE as a particular case of LWE. Let $n, q \in \mathbb{N}$ be the LWE dimension and modulus respectively, and let $m = \mathcal{O}(n \log(q))$ be the number of LWE samples. We also define $t \ll q$ with $\gcd(t, q) = 1$ as the plaintext modulus (i.e., the message space $\mathcal{M} = \mathbb{Z}_t$) and χ as a small distribution from $\mathbb{Z}_q$. The ciphertext space $\mathcal{C}$ is identified with $\mathbb{Z}_q^n$. The starting modulus q is constructed as $q = \prod_{i=1}^{L+1} p_i$, where L is the multiplicative depth of the circuit to be evaluated and each p_i is set to be $p_i \equiv 1 \bmod t$.

- $\mathrm{Gen}(1^\lambda)$: pick $n = n(\lambda), m = \mathcal{O}(n \log(q))$, and let the secret $sk = \boldsymbol{x} := (1, -\boldsymbol{x}') \in \mathbb{Z}_q^n$, where $\boldsymbol{x}' \sim \mathbb{Z}_q^{n-1}$ is uniformly sampled. The public key is obtained by sampling a uniformly random $B \sim \mathbb{Z}_q^{m \times (n-1)}$ and by defining the LWE instance $\boldsymbol{b} := B\boldsymbol{x}' + \boldsymbol{s} \in \mathbb{Z}_q^m$, for some small $\boldsymbol{s} \sim \chi^m$. Then, generate the public key A by appending $\boldsymbol{b}$ as a column and B as

$$sk = \boldsymbol{x} := (1, -\boldsymbol{x}') \in \mathbb{Z}_q^n, \quad pk = A := \left[\, \boldsymbol{b} \mid B \,\right] \in \mathbb{Z}_q^{m \times n}. \tag{2.1}$$

  Note that $A \cdot \boldsymbol{x} = \boldsymbol{s} \in \mathbb{Z}_q^m$.
- $\mathrm{Enc}_{pk}(\mu)$: the encryption function under a fixed public key pk is defined as:

$$\mathrm{BGV.Enc}_{pk} : \mathcal{M} \to \mathcal{C}, \quad \mathrm{BGV.Enc}_{pk}(\mu) := [t \cdot A^t \cdot \boldsymbol{r} + \boldsymbol{m}]_q \in \mathcal{C}, \tag{2.2}$$

  where $\mu \in \mathcal{M}$, $\boldsymbol{m} := (\mu, 0, \ldots, 0)$ and $\boldsymbol{r} \sim \{0,1\}^m$ is uniformly sampled.
- $\mathrm{Dec}_{sk}(\boldsymbol{c})$: the decryption function under a fixed secret key sk is defined as

$$\mathrm{BGV.Dec}_{sk} : \mathcal{C} \to \mathcal{M}, \quad \mathrm{BGV.Dec}_{sk}(\boldsymbol{c}) := [[\langle \boldsymbol{c}, \boldsymbol{x} \rangle]_q]_t \in \mathcal{M}, \tag{2.3}$$

  correctness can be shown by observing that $pk \cdot sk = t\boldsymbol{s}$ is short.
- $\mathrm{Add}(\boldsymbol{c}, \boldsymbol{c}')$: to homomorphically add two underlying messages, evaluate an addition between their corresponding ciphertexts:

$$\mathrm{BGV.Add} : \mathcal{C} \times \mathcal{C} \to \mathcal{C}, \quad \mathrm{BGV.Add}(\boldsymbol{c}, \boldsymbol{c}') := \boldsymbol{c} + \boldsymbol{c}' \in \mathcal{C}. \tag{2.4}$$

- $\mathrm{Mul}(\boldsymbol{c}, \boldsymbol{c}')$: to homomorphically multiply two underlying messages, evaluate a tensor product between their corresponding ciphertexts:

$$\mathrm{BGV.Mul} : \mathcal{C} \times \mathcal{C} \to \mathcal{C} \otimes \mathcal{C}, \quad \mathrm{BGV.Mul}(\boldsymbol{c}, \boldsymbol{c}') := \boldsymbol{c} \otimes \boldsymbol{c}' \in \mathcal{C} \otimes \mathcal{C}. \tag{2.5}$$

Key switching. After performing a tensor product between two ciphertexts, in addition to the noise growth, the dimension of the resulting ciphertext increases from n to n^2. Given $\boldsymbol{c}^{\times} = \boldsymbol{c}_1 \otimes \boldsymbol{c}_2 \in \mathbb{Z}_q^{n^2}$, by the properties of the tensor product, $\boldsymbol{c}^{\times}$ can still be decrypted using a "tensored" version of the secret key, namely $\boldsymbol{x} \otimes \boldsymbol{x}$:

$$[\langle \boldsymbol{c}_1 \otimes \boldsymbol{c}_2, \boldsymbol{x} \otimes \boldsymbol{x} \rangle]_q = [\langle \boldsymbol{c}_1, \boldsymbol{x} \rangle \cdot \langle \boldsymbol{c}_2, \boldsymbol{x} \rangle]_q.$$

Nevertheless, this strategy is not sustainable in the long term, as the sizes of the ciphertexts and keys grow exponentially. A solution is to perform a *key switching* procedure. Given a ciphertext $\boldsymbol{c}$, encrypted under a secret key $\boldsymbol{x}_1$, the objective is to find a new ciphertext $\boldsymbol{c}'$, under a new secret key $\boldsymbol{x}_2$, such that $\mathrm{Dec}_{\boldsymbol{x}_1}(\boldsymbol{c}) = \mathrm{Dec}_{\boldsymbol{x}_2}(\boldsymbol{c}')$. In order to enable this procedure, the key generator must produce a *key switching key* $\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}$ that will be publicly shared.

Given two secret keys $\boldsymbol{x}_1, \boldsymbol{x}_2$, their respective dimensions n_1, n_2 and the current modulus q, the key generation procedure for key switching first generates a matrix $A_{\mathrm{ks}} \in \mathbb{Z}_q^{n_1 \times n_2}$ following the public key generation procedure with the secret key $\boldsymbol{x}_2$. While generated by the public key generation procedure, A_{ks} is not made public and should be seen as a matrix containing n_2 Regev encryptions [Reg05] of zero, one for each row. To conclude the key switching key generation phase, the first secret key $\boldsymbol{x}_1$ is added component-wise in the first column of A_{ks}.

We will use bit decomposition (as firstly suggested in [BV11]) to reduce the amount of noise generated by this procedure, although other approaches are possible (e.g., extending the modulus [GHS12]). To summarize, we have $\mathrm{BGV.KeySwitchGen}_{\boldsymbol{x}_2} : \mathbb{Z}_q^{n_1} \to \mathbb{Z}_q^{n_1 \log(q) \times n_2}$, given by

$$\mathrm{BGV.KeySwitchGen}(\boldsymbol{x}_1) = A_{\mathrm{ks}} + (\mathtt{Pow}(\boldsymbol{x}_1), 0, \ldots, 0) \in \mathbb{Z}_q^{n_1 \log(q) \times n_2}, \tag{2.6}$$

where $A_{\mathrm{ks}} \in \mathbb{Z}_q^{n_1 \log(q) \times n_2}$ consists of $n_1 \log(q)$ (secret-key) encryptions of 0 under the new secret $\boldsymbol{x}_2$. Secondly, we have $\mathrm{BGV.KeySwitch}_{\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}} : \mathbb{Z}_q^{n_1} \to \mathbb{Z}_q^{n_2}$, given by

$$\mathrm{BGV.KeySwitch}(\boldsymbol{c}) := \mathtt{Bits}(\boldsymbol{c})^t \cdot (\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}) \in \mathbb{Z}_q^{n_2}, \tag{2.7}$$

where $\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}$ is the output of $\mathrm{BGV.KeySwitchGen}(\boldsymbol{x}_1)$. The new secret and public key pair should be freshly generated by the Gen procedure. One can pick $n_1 = n^2$ and $n_2 = n$ to decrease the dimension of $\boldsymbol{c}^{\times} = \boldsymbol{c}_1 \otimes \boldsymbol{c}_2$ back to n. We refer to Appendix D.1 for further information about key switching.

Modulus switching. For a ciphertext $\boldsymbol{c} = [t \cdot A^t \cdot \boldsymbol{r} + \boldsymbol{m}]_q$ we call $\langle t \cdot A^t \cdot \boldsymbol{r} + \boldsymbol{m}, \boldsymbol{x} \rangle = t\langle \boldsymbol{r}, \boldsymbol{s} \rangle + \mu$ the *decryption error* for BGV, which for correctness should be less than $q/2$. Assuming the initial decryption error in a fresh ciphertext to be bounded by b, the output of an addition will be bounded by $2b$ and the multiplication by roughly b^2. By choosing $q = \Theta(b^{L+1})$, one would be allowed to approximately perform up to $\log(L)$ multiplications since the noise grows exponentially. A possible solution, firstly suggested in [BV11], is to perform a *modulus*

switching. The informal idea is that the error growth is mostly determined by the absolute size of the errors, and one can reduce the absolute value of the error in the ciphertext by "cutting" part of the current modulus after each multiplication. Although the relative error with respect to the modulus remains fixed, the absolute error is reduced and this enables to perform more computations with a smaller error growth.

Say that a fresh ciphertext is encrypted using modulus q_1 such that $q_1 \equiv 1 \bmod t$, and let q_2 be the new modulus such that $q_2 \mid q_1$ and $q_2 \ll q_1$. Given a ciphertext $\boldsymbol{c} \in \mathbb{Z}_{q_1}^n$, the goal is to create an equivalent ciphertext $\boldsymbol{c}' \in \mathbb{Z}_{q_1}^n$ with a smaller absolute decoding error. The idea is to adjust $\boldsymbol{c}$ by subtracting some $\boldsymbol{\delta_c}$ to a multiple of q_1/q_2, after which $\boldsymbol{c}' = (\boldsymbol{c} - \delta_{\boldsymbol{c}}) \cdot (q_2/q_1) \in \mathbb{Z}_{q_1}^n$. For correctness, and to avoid changing decoding noise too much, the vector $\boldsymbol{\delta_c} \in \mathbb{Z}_{q_1}^n$ must satisfy that $[\langle \boldsymbol{\delta_c}, \boldsymbol{x} \rangle]_{q_1}$ is small and equal to 0 modulo t. It turns out that if $A = [A_1; A_2]$ with $A_2 \in \mathbb{Z}_{q_1}^{n \times n}$ invertible over q, we can use[5]:

$$\boldsymbol{\delta_c} := t \cdot A_2^t \cdot [A_2^{-t} \cdot \boldsymbol{c} \cdot t^{-1}]_{q_1/q_2},$$

where t^{-1} and A_2^{-t} are the inverses of t and A_2^t modulo q_1/q_2. Finally, given the current and the new modulus q_1, q_2, respectively, and a ciphertext $\boldsymbol{c} \in \mathbb{Z}_{q_1}^n$, the modulus switching is performed as a "round and scale" procedure:

$$\begin{aligned} \text{BGV.ModSwitch} &: \mathcal{C} \times \mathbb{N} \times \mathbb{N} \to \tfrac{q_2}{q_1}\mathcal{C}, \\ \text{BGV.ModSwitch}(\boldsymbol{c}, q_1, q_2) &:= \left[\left((\boldsymbol{c} - \boldsymbol{\delta_c}) \cdot \frac{q_2}{q_1} \right) \right]_{q_2}. \end{aligned} \tag{2.8}$$

By applying this approach, the number of possible sequential multiplications, given a moduli chain $q_1 \approx b^{L+1}$, actually becomes roughly L, since after each multiplication the bound on the noise can be reduced from b^2 back to (approximately) b, while the modulus only decreases by a factor b, preventing an exponential growth [BGV12, Lemma 4].

2.4 Original GSW cryptosystem

One year after the introduction of the BGV cryptosystem, the Gentry-Sahai-Waters (GSW) [GSW13] scheme was proposed. Initially, it was presented as a "conceptually simple" alternative to the already available HE schemes. Today it is considered a fundamental building block to bootstrap advanced constructions such as TFHE. In particular, homomorphic operations are performed "naturally" between ciphertexts as they are encoded in matrices – note that we will follow the notation from [AP14, Section 3], where ciphertexts are not square matrices, but the two representations are semantically equivalent. As a consequence, multiplications are performed as matrix multiplications without requiring any key switching procedure. The noise growth is reduced by encoding the message to be encrypted in a so-called gadget matrix G; a multiplication between two ciphertexts C_1 and C_2 is indeed evaluated as $C_1 \cdot G^{-1}(C_2)$. The GSW cryptosystem is

[5] For small-secret LWE one equivalently has $A_2 = I_n$, which simplifies the expression.

defined by the LWE dimension and modulus n, q and by the number of samples $m = n\log(q)$. We assume q to be some power of two. Additionally, we define χ as some small distribution over the integers. A plaintext message is set to be in $\{0,1\} \subset \mathbb{Z}$.

The key generation phase is exactly the same as the one in the BGV cryptosystem, we thus have $\boldsymbol{x} = (1, -\boldsymbol{x}') \in \mathbb{Z}_q^n$ as the secret key and $A := \left[\, \boldsymbol{b} \mid B \,\right] \in \mathbb{Z}_q^{m \times n}$, with $\boldsymbol{b} := B\boldsymbol{x}' + \boldsymbol{s} \in \mathbb{Z}_q^m$. For simplicity, we will consider a binary message space $\mathcal{M} := \{0,1\}$ and ciphertext space $\mathcal{C} := \mathbb{Z}_q^{n \times m}$.

Gadget decompositions. Gadget matrices were firstly (implicitly) introduced by Ajtai [Ajt96] and formalized by Micciancio and Peikert [MP12] (although referred to as primitive matrices) as lattice trapdoors. Refer to [GMP19, Def. 4] for a precise definition; for our needs, we only require the basic $\mathbb{Z}_q$-gadget matrix of size $w = 2$, constructed as $\texttt{Pow}(1) \otimes I_n$, where $\texttt{Pow}(1) = (1, 2, 4, \ldots, 2^{\log(q)-1})$. As a result, we obtain the following block matrix G.

$$G = \begin{pmatrix} 1\ 2\ 4\ \ldots\ 2^{\log(q)-1} & & \\ & \ddots & \\ & & 1\ 2\ 4\ \ldots\ 2^{\log(q)-1} \end{pmatrix} \in \mathbb{Z}_q^{n \times m}. \tag{2.9}$$

By abuse of notation, we define $G^{-1} : \mathbb{Z}_q^{n \times m} \to \mathbb{Z}_2^{n\log(q) \times m}$ to be the application of $\texttt{Bits}$ over each entry of the input matrix. The main operations in GSW are defines as follows:

- $\mathrm{Enc}_{pk}(\mu)$: the encryption function under a fixed public key pk is defined as:

$$\mathrm{GSW.Enc}_{pk} : \mathcal{M} \to \mathcal{C}, \quad \mathrm{GSW.Enc}_{pk}(\mu) := [A^t R + G\mu]_q \in \mathcal{C}, \tag{2.10}$$

  where $\mu \in \mathcal{M}$ and $R \sim \{0,1\}^{m \times n}$ is uniformly sampled. Each column can be seen as a BGV encryption of zero with $t = 1$.
- $\mathrm{Dec}_{sk}(C)$: the decryption function under a fixed secret key sk is defined as

$$\mathrm{GSW.Dec}_{sk} : \mathcal{C} \to \mathcal{M}, \quad \mathrm{GSW.Dec}_{sk}(\boldsymbol{c}) := \lfloor [\langle \boldsymbol{c}_j, \boldsymbol{x} \rangle]_q \rceil_2 \in \mathcal{M}, \tag{2.11}$$

  where $\boldsymbol{c}_j$ is the j-th column of C, with $j = \log(q)$, where $\mu \cdot q/2$ has been added in the first component.
- $\mathrm{Add}(C, C)$: to homomorphically add two underlying messages, evaluate an addition between their corresponding ciphertexts:

$$\mathrm{GSW.Add} : \mathcal{C} \times \mathcal{C} \to \mathcal{C}, \quad \mathrm{GSW.Add}(C, C') := C + C' \in \mathcal{C}. \tag{2.12}$$

- $\mathrm{Mul}(C, C')$: to homomorphically multiply two underlying messages, evaluate the following:

$$\mathrm{GSW.Mul} : \mathcal{C} \times \mathcal{C} \to \mathcal{C}, \quad \mathrm{GSW.Mul}(C, C') := C \cdot G^{-1}(C') \in \mathcal{C}. \tag{2.13}$$

Encoding the message in the gadget matrix has the advantage of allowing for an asymmetric and quasi-additive error growth in multiplications, we refer to [AP14, Section 3] for further details.

3 A lattice framework for HE

Until now, we simply described the HE schemes without considering a lattice perspective. Now we provide a geometric intuition of the processes involved, in lattice terminology, when using an LWE-based cryptosystem. Then, we generalize these to any lattice.

3.1 Interpreting LWE from a lattice perspective

In LWE-based cryptosystems, the hardness assumption is used twice: for the key generation and for the encryption phases. Let us start from a geometric point of view: LWE has a reduction to hard problems over (random) q-ary lattices. More precisely, let $B \in \mathbb{Z}_q^{m\times n}$, we consider the q-ary lattice

$$\mathcal{L}_q(B) := \{\boldsymbol{y} \in \mathbb{Z}^m : \boldsymbol{y} \equiv B\boldsymbol{x} \bmod q, \text{ for some } \boldsymbol{x} \in \mathbb{Z}_q^n\}. \tag{3.1}$$

We call B a *q-ary basis* of $\mathcal{L}_q(B)$. Note that the lattice $\mathcal{L}_q(B)$ is generated by the columns of B and the *q-vectors* $q\boldsymbol{e}_i \in q\mathbb{Z}^m$ where $\boldsymbol{e}_i$ is a unit vector. The q-dual lattice of $\mathcal{L}_q(B)$, also known as a parity check or SIS lattice, is given by

$$\mathcal{L}_q^\perp(B) := \{\boldsymbol{y} \in \mathbb{Z}^m : B^t\boldsymbol{y} \equiv \boldsymbol{0} \bmod q\}. \tag{3.2}$$

Equivalently, for the the normal dual we thus have $\mathcal{L}_q(B)^* = {}^1\!/_q \cdot \mathcal{L}_q^\perp(B)$. While for $\mathcal{L}_q(B)$ we already mentioned a generating set of $m+n$ vectors, writing down a basis for these lattices can be a bit more involved. Typically, we can write $B^t = [B_1^t \,|\, B_2^t]$ where $B_2 \in \mathbb{Z}_q^{n\times n}$ is invertible, in this case $\mathcal{L}_q(B)$ and $\mathcal{L}_q^\perp(B)$ have a *lattice basis* of the *normal form*

$$\begin{pmatrix} qI_{m-n} & B_1B_2^{-1} \\ 0 & I_n \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0 & I_{m-n} \\ qI_n & -B_2^{-t}B_1^t \end{pmatrix}, \tag{3.3}$$

respectively. When $B \in \mathbb{Z}_q^{m\times n}$ is uniformly random we call $\mathcal{L}_q(B)$ a random q-ary lattice (of parameter n, m, q). Note that if $\mathcal{L}_q(B)$ is a random q-ary lattice, then its q-dual $\mathcal{L}_q^\perp(B)$ is also a random q-ary lattice[6].

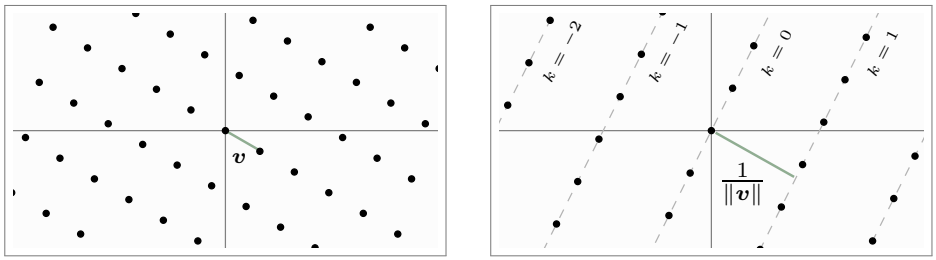
For the key generation, the LWE assumption is used to safely hide a short vector in such a random q-ary lattice. To see this, let us consider an LWE instance $(B, \boldsymbol{b} = B\boldsymbol{x}' + \boldsymbol{s} \bmod q)$, where $B \in \mathbb{Z}_q^{m\times(n-1)}$ and $\boldsymbol{x}' \in \mathbb{Z}_q^{n-1}$ are uniformly random, and $\boldsymbol{s} \in \mathbb{Z}_q^m$ is a short vector sampled from some distribution $s_i \sim \chi$. Firstly, note that $\boldsymbol{b}$ (when lifted to $\mathbb{Z}^m$) is in fact a BDD instance in the random q-ary lattice $\mathcal{L}_q(B)$, as, up to addition by q-vectors, $\boldsymbol{b}$ has distance only $\|\boldsymbol{s}\|$ to the lattice point $B\boldsymbol{x}'$. While this interpretation will be of interest later, we will turn this BDD instance into a planted short vector, focusing on the key generation. By the LWE assumption the vector $\boldsymbol{b} \in \mathbb{Z}_q^m$ is indistinguishable from a uniformly random vector, and thus in particular we can append it as a column

[6] Technically, there is a difference in distributions which can be avoided by restricting prime q and full-rank matrices $B \in \mathbb{Z}_q^{m\times n}$. We will ignore this minor difference.

to B to construct $A := \left[\boldsymbol{b} \mid B \right] \in \mathbb{Z}_q^{m \times n}$, that is indistinguishable from a random q-ary basis.

The lattice $\mathcal{L}_q(A)$ with q-ary basis A is thus indistinguishable from a random q-ary lattice, however, the knowledge of the secret $\boldsymbol{x} := (1, -\boldsymbol{x}')$, allows one to find the *secret short vector* $\boldsymbol{s} \in \mathcal{L}_q(A)$ by computing $A \cdot \boldsymbol{x} \equiv \boldsymbol{s} - B \cdot \boldsymbol{x}' \equiv \boldsymbol{s} \in \mathbb{Z}_q^m$.

Dual vectors and partitions. Recall that the dual lattice $\mathcal{L}^*$ consists of all vectors having an integer inner product with all primal vectors $\boldsymbol{v} \in \mathcal{L}$. As a result, any primal vector $\boldsymbol{v} \in \mathcal{L}$ naturally partitions the dual lattice $\mathcal{L}^*$ into layers $\{\boldsymbol{y} \in \mathcal{L}^* : \langle \boldsymbol{v}, \boldsymbol{y} \rangle = k\}$, for $k \in \mathbb{Z}$, orthogonal to $\boldsymbol{v}$, as illustrated in Fig. 3.1.



(a) $\mathcal{L}$ and its shortest vector $\boldsymbol{v}$ (b) $\mathcal{L}^*$ with partitions given by $\boldsymbol{v}$

Fig. 3.1: The relation between short vectors in $\mathcal{L}$ and partitions in $\mathcal{L}^*$

The distance between these layers is $1/\|\boldsymbol{v}\|$; the shorter the vector, the larger the separation. By considering the shortest vector in the primal, one can generate layers of points in the dual lattice that are at the maximum distance between each other. This space is precisely where a message can be encoded and hidden.

Let us consider a short vector $\boldsymbol{v} \in \mathcal{L}$ and a BDD instance $\boldsymbol{t} = \boldsymbol{y} + \boldsymbol{e} \in \mathbb{R}^m$ where $\boldsymbol{y} \in \mathcal{L}^*$ and $\boldsymbol{e} \in \mathbb{R}^m$ is small. Note that, given the target $\boldsymbol{t}$, it is generally hard to recover the error $\boldsymbol{e}$. However, since we know that $\langle \boldsymbol{y}, \boldsymbol{v} \rangle$ is integer, we can, knowing $\boldsymbol{v} \in \mathcal{L}$, compute

$$\langle \boldsymbol{e}, \boldsymbol{v} \rangle \equiv \langle \boldsymbol{t}, \boldsymbol{v} \rangle \bmod 1 \in [-\tfrac{1}{2}, \tfrac{1}{2}).$$

At this point, if the distance $1/\|\boldsymbol{v}\|$ between the partitions is large and the error $\boldsymbol{e}$ is small, one can hope that the closest partition to $\boldsymbol{t}$ is the one with the lattice point $\boldsymbol{y}$, or equivalently, $|\langle \boldsymbol{e}, \boldsymbol{v} \rangle| < \frac{1}{2}$. In particular we have $|\langle \boldsymbol{e}, \boldsymbol{v} \rangle| \leq \|\boldsymbol{e}\| \cdot \|\boldsymbol{v}\|$. If this is the case, we do not solely recover $\langle \boldsymbol{e}, \boldsymbol{v} \rangle$ modulo 1, but we recover the exact value $\langle \boldsymbol{e}, \boldsymbol{v} \rangle \in \mathbb{R}$, and therefore gain partial information about the error $\boldsymbol{e}$. We can therefore try to encode the message in this error so that one can recover it from this partial information. Moreover, note that we only have interest in the error here and thus might as well consider only the syndrome $\boldsymbol{t} \equiv \boldsymbol{e} \in \mathbb{R}^m / \mathcal{L}^*$.

Note that in the LWE setting, one considers the q-dual $q\mathcal{L}^* = \mathcal{L}_q^\perp(A)$ of $\mathcal{L} = \mathcal{L}_q(A)$ and thus instead of considering the previous inner products modulo 1, one has to consider them modulo q. Because $\mathcal{L}_q^\perp(A) \subset \mathbb{Z}^m$ this conveniently gives an integer space for the syndromes and allows one to consider the inner products modulo q as elements of $\mathbb{Z}_q$.

Visualizing an LWE ciphertext. As already spoiled in the previous subsection, an LWE ciphertext can be interpreted as a BDD instance, or more precisely a syndrome, with respect of the q-dual lattice $\mathcal{L}_q^\perp(A)$ of the trapdoor lattice $\mathcal{L}_q(A)$. To observe this we consider a BGV ciphertext, which (for now, assume a message $\mu = 0$) is sampled as $\boldsymbol{c} := [t \cdot A^t \cdot \boldsymbol{r}]_q \in \mathbb{Z}_q^n$, with small $\boldsymbol{r} \in \{0,1\}^m$ and $t \geq 1$. A GSW ciphertext, for $\mu = 0$ can be considered as a collection of $n\log(q)$ different BGV ciphertexts using $t = 1$. We thus focus on the BGV case.

In LWE terms, the ciphertext is a random subset sum (controlled by $t \cdot \boldsymbol{r}$) of the LWE samples contained in the public key, essentially forming a new LWE sample. We claim however that $[A^t \cdot (t \cdot \boldsymbol{r})]_q$ represents the coset $t \cdot \boldsymbol{r} + \mathcal{L}_q^\perp(A)$, or more precisely the syndrome $t \cdot \boldsymbol{r} \in \mathbb{Z}^m/\mathcal{L}_q^\perp(A)$, where $t \cdot \boldsymbol{r}$ is a small error. Indeed, for any $\boldsymbol{y} \in \mathcal{L}_q^\perp(A)$, we have by definition that $A^t\boldsymbol{y} \equiv 0 \bmod q$, and thus

$$A^t(t \cdot \boldsymbol{r} + \boldsymbol{y}) \equiv A^t\boldsymbol{y} + A^t(t \cdot \boldsymbol{r}) \equiv \boldsymbol{0} + A^t(t \cdot \boldsymbol{r}) \equiv A^t(t \cdot \boldsymbol{r}) \bmod q.$$

The LWE assumption precisely says that this syndrome is indistinguishable from a uniformly random syndrome in the (seemingly) random q-ary lattice $\mathcal{L}_q^\perp(A)$. For the decryption procedure, consider the secret short vector $\boldsymbol{s} \equiv A\boldsymbol{x} \in \mathcal{L}_q(A)$. Note that BGV first computes the modulo-q inner product $[\langle \boldsymbol{c}, \boldsymbol{x}\rangle]_q$. Again, we claim that this is precisely the inner product between the short primal vector $\boldsymbol{s}$, partitioning $\mathcal{L}_q^\perp(A)$, and the syndrome error $t \cdot \boldsymbol{r} \in \mathbb{Z}^m/\mathcal{L}_q^\perp(A)$. Indeed

$$\langle \boldsymbol{c}, \boldsymbol{x}\rangle \equiv \langle A^t \cdot (t \cdot \boldsymbol{r}), \boldsymbol{x}\rangle \equiv \langle t \cdot \boldsymbol{r}, A \cdot \boldsymbol{x}\rangle \equiv \langle t \cdot \boldsymbol{r}, \boldsymbol{s}\rangle \bmod q, \tag{3.4}$$

and assuming that the error $t \cdot \boldsymbol{r}$ and the short vector $\boldsymbol{s} \in \mathcal{L}_q(A)$ are small enough, i.e., $|\langle t \cdot \boldsymbol{r}, \boldsymbol{s}\rangle| < q/2$, we obtain precisely the inner product $t \cdot \langle \boldsymbol{r}, \boldsymbol{s}\rangle \in \mathbb{Z}$.

Since $t \cdot \langle \boldsymbol{r}, \boldsymbol{s}\rangle \equiv 0 \bmod t$, BGV encodes a message $\mu \in \{0, \ldots, t-1\}$ by adding $\mu\boldsymbol{e}_1$ to $\boldsymbol{c}$, using that $\langle \mu\boldsymbol{e}_1, \boldsymbol{x}\rangle = \mu \equiv \mu \bmod t$. Essentially, this corresponds to the addition of some vector $\mu \cdot \boldsymbol{d}$ to $t \cdot \boldsymbol{r}$ for which we know that $\langle \boldsymbol{d}, \boldsymbol{s}\rangle$ is small and satisfies $\langle \boldsymbol{d}, \boldsymbol{s}\rangle \equiv 1 \bmod t$. More details on this *decoding by short primal vector* process are given in Section 4.1.

3.2 Generalizing key generation: a (trapdoor) lattice

We have seen that the key generation of LWE-based encryption schemes constructs a *public* lattice $\mathcal{L}$ along with some short *secret* vector $\boldsymbol{s} \in \mathcal{L}$. This will be the starting point of our generalization. For LWE, the lattice $\mathcal{L}$ is a (seemingly) random q-ary lattice $q\mathbb{Z}^m \subset \mathcal{L} \subset \mathbb{Z}^m$ of dimension m along with a planted short vector. In our framework we will allow $\mathcal{L}$ to be any lattice with a planted short vector $\boldsymbol{s} \in \mathcal{L}$, and we leave the specifics of this to the instantiation of our

framework. We assume that the public lattice $\mathcal{L}$ is represented by some (long) basis B, and that the short vector $\boldsymbol{s} \in \mathcal{L}$ is secret. The basis B will be our public key and the short vector $\boldsymbol{s}$ the secret key. As all the other steps will be generic to any lattice, one essentially only has to change the instantiation of the key generation.

The LWE-based encryption schemes we discussed encrypt by constructing a syndrome decoding instance in the q-dual lattice $q\mathcal{L}^*$ of $\mathcal{L}$. This requires some discretized ambient space $\mathcal{D} \supset q\mathcal{L}^*$ for the error and decoding targets, which we will consider in more detail later. For decryption, the short primal vector gives a partition of the q-dual lattice as seen in Figure 3.1, which allows to partially decode or distinguish the syndrome (i.e., the message).

Definition 3.1 (Lattice Keygen). $\mathsf{LatticeGen}$ *is a probabilistic algorithm that on input a security parameter* λ *generates* $pk, sk := (B, D), \boldsymbol{s} \leftarrow \mathsf{LatticeGen}(1^\lambda)$ *such that*

- $\mathcal{L} := \mathcal{L}(B)$ *and* $\mathcal{D} := \mathcal{L}(D)$ *are lattices satisfying* $q\mathcal{L}^* \subset \mathcal{D} \subset \mathcal{L}^*$,
- *(short vector)* $\boldsymbol{s} \in \mathcal{L}$,
- $\langle \boldsymbol{d}_1, \boldsymbol{s} \rangle \equiv 1 \bmod t$, *where* $\boldsymbol{d}_1$ *is the first column of* D,

for some integer parameters $q \geq 2, t \geq 1$ *and* $m := \dim(\mathcal{L}) \geq 2$.

For security reasons, the decryption procedure must be hard given only the public basis B of $\mathcal{L}$. Several choices can be made here. For LWE-based schemes one usually uses the decision-LWE assumption which, with our interpretation, essentially says that a BDD syndrome in a random q-ary lattice is indistinguishable from a uniformly random syndrome.

Security Assumption 1 (SDP is indistinguishable from uniform) *Let* $(B, D), \boldsymbol{s} \leftarrow \mathsf{LatticeGen}(1^\lambda)$, $\sigma_{enc} \geq \|\tilde{D}\| \cdot \sqrt{\ln(2m+4)/\pi}$ *and* $C := B^{-t}$. *The* $\Delta\text{SDP}_{C,D,\sigma_{enc}}$ *problem asks to distinguish with non-negligible advantage in* λ *between two distributions of syndromes* $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{poly(\dim(C))} \in \mathcal{P}(qC) \cap \mathcal{D}$*: either they are* SDP *instances of the form*

$$\boldsymbol{c}_i \leftarrow \boldsymbol{e}_i \bmod \mathcal{P}(qC) \in \mathcal{P}(qC) \cap \mathcal{D} \text{ with error } \boldsymbol{e}_i \leftarrow \mathcal{G}_{\mathcal{D},\sigma_{enc}},$$

or they are all uniform. The assumption is that the $\text{SDP}_{C,D,\sigma_{enc}}$ *problem is infeasible.*

In Appendix A we show that our Lattice-BGV and Lattice-GSW schemes, to be presented in Section 4, are IND-CPA secure under Assumption 1.

Remark 1. The choice for an unstructured lattice $\mathcal{L}$ and a single secret short vector $\boldsymbol{s} \in \mathcal{L}$ was made to closely follow the classical LWE-based HE schemes, and to present the framework in its simplest form. One could easily replace this by allowing multiple short vectors, or even a dense sublattice $S \subset \mathcal{L}$ with a specific decoding procedure in its dual S^*. This could for instance improve the error resistance and the number of messages that can be packed in a single ciphertext. Moreover, one could simply replace the lattices here by structured (module-)lattices and the main steps of our approach should still work with minor changes.

3.3 The ambient space

Let $\mathcal{L}$ and $q\mathcal{L}^*$ be our primal and q-dual lattice, respectively, as generated by the key generation procedure, and let B be the public basis of $\mathcal{L}$. Recall that in LWE-based schemes, encryptions are BDD instances $\boldsymbol{t} = \boldsymbol{v} + \boldsymbol{e} \in \mathbb{Z}^m$, or more precisely, their syndrome $\boldsymbol{t} \bmod q\mathcal{L}^* \in \mathbb{Z}^m/q\mathcal{L}^* \cong \mathbb{Z}_q^n$. Therefore, these BDD instances, and their errors, live in the ambient space $\mathbb{Z}^m$, which $q\mathcal{L}^*$ is naturally contained in as it is a q-ary lattice. Having $\mathbb{Z}^m$ as an ambient space is useful, as we know how to sample small errors $\boldsymbol{e} \in \mathbb{Z}^m$, needed for the public-key encryptions, and because the syndrome space $\mathbb{Z}^m/q\mathcal{L}^*$ is finite and naturally has a nice group structure to make the homomorphic operations work. Generally however, we do not have that $q\mathcal{L}^*$ lies in $\mathbb{Z}^m$, as we do not restrict ourselves merely to cases were $q\mathcal{L}^*$ is an integer lattice. We thus have to construct some appropriate ambient space $\mathcal{D} \supset q\mathcal{L}^*$ with similar properties. We therefore aim to construct a generalized ambient space that satisfies the following two properties:

1. It has a good public basis D that allows to sample small errors $\boldsymbol{e} \in \mathcal{D}$.
2. It satisfies $q\mathcal{D} \subset q\mathcal{L}^* \subset \mathcal{D}$.

The first point generalizes that the integer lattice $\mathbb{Z}^m$ has an orthogonal basis which allows to efficiently sample small errors. The second point is a generalization of the q-ary property, and forces $\mathcal{D}/q\mathcal{L}^*$ to be a subgroup of $\mathbb{Z}_q^m$, which will later be important to make the homomorphic operations work.

Simple ambient space. To sample a small error $\boldsymbol{e} \in \mathcal{D}$ form a Discrete Gaussian distribution we require that the Gram-Schmidt norm $\|\tilde{D}\|$ of the ambient space basis D is sufficiently small compared q. The simplest way to obtain an appropriate ambient space $\mathcal{D}$ with a sufficiently short basis is to simply pick $\mathcal{D} := \mathcal{L}^*$ with $D := B^{-t}$. For q sufficiently large $\mathcal{D} = \mathcal{L}^*$ is dense enough compared to $q\mathcal{L}^*$ and the basis B^{-t} is sufficiently small compared to that of $q\mathcal{L}^*$ to sample relatively small errors. This choice also trivially satisfies the condition $q\mathcal{D} \subset q\mathcal{L}^* \subset \mathcal{D}$.

Ciphertext compression. While the previous choice of $\mathcal{D} := \mathcal{L}^*$ works, it creates a rather large ambient space. This is reflected in the size of the ciphertext syndromes which are elements in $\mathcal{D}/q\mathcal{L}^*$ and thus require $\log_2(|\mathcal{D}/q\mathcal{L}^*|)$ bits to write down, which with the simple choice above gives $m\log_2(q)$. For GSW ciphertexts, which require multiple syndromes, this even grows to $\log_2(|\mathcal{D}/q\mathcal{L}^*|)^2$ bits.

Recall that, in the case of LWE, we have an ambient space $\mathcal{D}$ such that $\mathcal{D}/q\mathcal{L}^* \cong \mathbb{Z}_q^n$ and thus we only require $n\log_2(q) < m\log_2(q)$ bits for the syndrome. In general we can also construct such a smaller ambient space. The idea is that while $D = B^{-t}$ is an appropriate choice, the resulting Gram-Schmidt vectors $\boldsymbol{d}_1^*, \ldots, \boldsymbol{d}_m^*$ typically wildly vary in size while we only require to bound their maximum value $\max_i \|\boldsymbol{d}_i^*\|$. Instead of picking $D = B^{-t}$ we can instead scale the different basis vectors $\boldsymbol{d}_i$ by some appropriate positive integer factors

$p \mid q$, thereby increasing $\|\boldsymbol{d}_i^*\|$ by a factor p while decreasing $|\mathcal{D}/q\mathcal{L}^*|$ by a factor p. This works especially well if we assume that $q = p^k$ for some small modulus p. For more details about such ciphertext compression, and its relation to small-secret LWE, we refer to Appendix B.

Coefficient representation. So far, to keep the geometric intuition, we have presented our ambient space $\mathcal{D}$ as a lattice with a good basis D, and small errors $\boldsymbol{e} \in \mathcal{D}$ and syndromes $\boldsymbol{c} \in \mathcal{D}/q\mathcal{L}^*$ are simply assumed to be represented by lattice vectors. One of the crucial aspects of the key switching procedure using the `Bits` and `Pow` functions in BGV and the gadget matrix used in the GSW scheme is however that we are working with integer vectors (or say in $\mathbb{Z}_q^m$). Furthermore, working with such integer vectors is more useful in concrete implementations. This is, however, not a problem in our more general setting, as in fact a lattice vector $\boldsymbol{v} \in \mathcal{D}$ can still be represented by their integer coefficient vector $\boldsymbol{y} \in \mathbb{Z}^m$ in the basis D such that $\boldsymbol{v} = D \cdot \boldsymbol{y}$, or in a basis C of $\mathcal{L}^*$ (using that $\mathcal{D} \subset \mathcal{L}^*$). Similarly, the isomorphism $\mathcal{D}/q\mathcal{L}^* \cong G \subset \mathbb{Z}_q^m$ where G is a subgroup of $\mathbb{Z}_q^m$ can be made explicit by the map $\boldsymbol{c} \mapsto C^{-1}\boldsymbol{c} \bmod q \in G \subset \mathbb{Z}_q^m$. Later we will see that applying the `Bits`, `Pow` and gadget matrix can simply be applied to this coefficient representation. E.g. for $\boldsymbol{c} \in \mathcal{D}/q\mathcal{L}^*$ and $\boldsymbol{s} \in \mathcal{L}$ we have that $C^{-1}\boldsymbol{c} \in \mathbb{Z}_q^m$ and $C^t\boldsymbol{s} \in \mathbb{Z}^m$, and we have that

$$\langle \boldsymbol{c}, \boldsymbol{s} \rangle \equiv \langle C^{-t}\boldsymbol{c}, C^t\boldsymbol{s} \rangle \equiv \langle \texttt{Bits}(C^{-t}\boldsymbol{c}), \texttt{Pow}(C^t\boldsymbol{s}) \rangle \bmod q.$$

Remark 2. Multiplication by C^{-1} gives an explicit isomorphism from $\mathcal{D}/q\mathcal{L}^*$ into a subgroup of $\mathbb{Z}_q^m$. Noting that $C^{-1} = B^t$ we can recognize that BGV and GSW are also using this representation (see $A^t \cdot \boldsymbol{r}$ in the encryption). Another explicit isomorphism into some finite abelian group $G' = \mathbb{Z}^m/\mathcal{R}$ is given through multiplication by D^{-1}. In the special case that C, D differ by a matrix in Hermite Normal Form (as is the case for our constructions), the map D^{-1} gives an explicit isomorphism into $G' = \prod_{i=1}^m \mathbb{Z}/p_i\mathbb{Z}$ for some $p_i \mid q$.

4 Abstract HE cryptosystems from lattices

Following the key generation and ambient space construction in the previous section, we now present our lattice-based generalizations of the BGV [BGV12] and GSW [GSW13] HE schemes.

4.1 Lattice-BGV

The Lattice-BGV scheme is a lattice generalization of BGV that can in principle be instantiated, in public key mode, using any hardness assumption that hides the secret shortest vector(s) of a lattice (e.g., LWE, NTRU, LIP and so on), and for which decoding in the dual is hard.

The message space and ciphertext space are respectively $\mathcal{M} := \mathbb{Z}_t$ and $\mathcal{C} := \mathcal{D}/q\mathcal{L}^*$ for some plaintext modulus $t \ll q$, lattice $\mathcal{L}$, and ambient space

$\mathcal{D}$ obtained from the key generation process from Definition 3.1, which satisfies $q\mathcal{L}^* \subset \mathcal{D} \subset \mathcal{L}$. We assume q to be built as a moduli ladder, i.e., a product of many primes, like in the LWE-based BGV.

Encryption and decryption. As introduced in Section 3.1, the encryption process in BGV samples a small error $\boldsymbol{e} \in \mathcal{D}$, containing an encoded message, which is then canonically represented by some syndrome $\boldsymbol{c} \in \mathcal{D}/q\mathcal{L}^*$. The secret short vector $\boldsymbol{s} \in \mathcal{L}$ will be used to partially decode it and recover the encoded message. To sample a small error in the ambient space $\mathcal{D}$ we precisely use its public good basis D that we constructed in Section 3.3. From the key generation we also have that the first basis vector $\boldsymbol{d}_1$ of D satisfies $\langle \boldsymbol{d}_1, \boldsymbol{s} \rangle \equiv 1 \bmod t$. The goal is to generalize the $\mathrm{BGV.Enc}_{pk}(\mu)$ procedure in Eq. (2.2). We start by considering a small vector $\boldsymbol{r} \sim \mathcal{G}_{\mathcal{D},\sigma_{\mathrm{enc}}}$ in the ambient space $\mathcal{D}$, with σ_{enc} chosen as explained in Section 3.3. Then, the message is encoded in the small error $\boldsymbol{e} \in \mathcal{D}$ as follows:

$$\boldsymbol{e} := t \cdot \boldsymbol{r} + D \cdot (\mu, 0, \ldots, 0)^t = t \cdot \boldsymbol{r} + \boldsymbol{d}_1 \mu \in \mathcal{D}, \tag{4.1}$$

where $\mu \in \mathcal{M}$ is the message, and $\boldsymbol{d}_1$ is the first basis vector of D, the good basis of $\mathcal{D}$. The discrete Gaussian sampler $\mathcal{G}_{\mathcal{D},\sigma_{\mathrm{enc}}}$ returns a point $\boldsymbol{r}$ that is close to the origin in the ambient space $\mathcal{D}$, and thus $\boldsymbol{e} \approx t \cdot \boldsymbol{r}$ is also small. The following step is to hide the sampled error $\boldsymbol{e}$, containing the message μ, in a canonical coset representative $\boldsymbol{c} \in \boldsymbol{e} + q\mathcal{L}^*$. For this we use the fundamental region of $q\mathcal{L}^*$ defined by the public basis $qC = qB^{-t}$:

$$\boldsymbol{c} = \boldsymbol{e} \bmod \mathcal{P}(qC) \in \mathcal{D}/q\mathcal{L}^*. \tag{4.2}$$

Because $\mathcal{L}^* \subset \mathcal{D} \subset q\mathcal{L}^*$ one can represent $\boldsymbol{e}$ and $\boldsymbol{c}$ as integer vectors with respect to the basis C and thus the above operation can be simply implemented with modular arithmetic. We now formalize the just presented procedures.

$$\mathcal{L}\text{-BGV.Enc}_{pk} : \mathcal{M} \to \mathcal{C}, \quad \mathcal{L}\text{-BGV.Enc}_{pk}(\mu) := t\boldsymbol{r} + \boldsymbol{d}_1 \mu \bmod \mathcal{P}(qC). \tag{4.3}$$

Recall that the secret key $\boldsymbol{s}$ is a short vector in the primal lattice $\mathcal{L}$. The decryption therefore follows the arguments from Section 3.

$$\mathcal{L}\text{-BGV.Dec}_{sk} : \mathcal{C} \to \mathcal{M}, \quad \mathcal{L}\text{-BGV.Dec}_{sk}(\boldsymbol{c}) := [[\langle \boldsymbol{c}, \boldsymbol{s} \rangle]_q]_t, \tag{4.4}$$

Decryption is correct, since:

$$[[\langle \boldsymbol{c}, \boldsymbol{s} \rangle]_q]_t \equiv [t \cdot \langle \boldsymbol{r}, \boldsymbol{s} \rangle + \langle \boldsymbol{d}_1, \boldsymbol{s} \rangle \cdot \mu]_t \equiv \mu \in \mathbb{Z}_t,$$

assuming that $|t \cdot \langle \boldsymbol{r}, \boldsymbol{s} \rangle + \langle \boldsymbol{d}_1, \boldsymbol{s} \rangle \cdot \mu| < q/2$. Notice that $\langle \boldsymbol{d}_1, \boldsymbol{s} \rangle \equiv 1 \bmod t$ by Definition 3.1. We also define the corresponding homomorphic operations:

$$\mathcal{L}\text{-BGV.Add} : \mathcal{C} \times \mathcal{C} \to \mathcal{C}, \quad \mathcal{L}\text{-BGV.Add}(\boldsymbol{c}, \boldsymbol{c}') := \boldsymbol{c} + \boldsymbol{c}' \in \mathcal{C}. \tag{4.5}$$

Homomorphic multiplications can be evaluated as in the standard BGV scheme, and their correctness follows from the fact that the inner product of two tensor products becomes the product of two inner products,

$$\mathcal{L}\text{-BGV.Mul} : \mathcal{C} \times \mathcal{C} \to \mathcal{C} \otimes \mathcal{C}, \quad \mathcal{L}\text{-BGV.Mul}(\boldsymbol{c}, \boldsymbol{c}') := \boldsymbol{c} \otimes \boldsymbol{c}' \in \mathcal{C} \otimes \mathcal{C}. \tag{4.6}$$

Given two ciphertexts $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{D}/q\mathcal{L}^*$, their tensor product $\boldsymbol{c}^\times = \boldsymbol{c}_1 \otimes \boldsymbol{c}_2 \in (\mathcal{D}/q\mathcal{L}^*) \otimes (\mathcal{D}/q\mathcal{L}^*)$ is a ciphertext under the new secret key $\boldsymbol{s}^\times = \boldsymbol{s} \otimes \boldsymbol{s} \in \mathcal{L} \otimes \mathcal{L}$ representing the product of the encrypted values μ_1 and μ_2:

$$\begin{aligned}
[[\langle \boldsymbol{c}^\times, \boldsymbol{s}^\times \rangle]_q]_t &\equiv [[\langle \boldsymbol{c}_1 \otimes \boldsymbol{c}_2, \boldsymbol{s} \otimes \boldsymbol{s} \rangle]_q]_t \equiv [[\langle \boldsymbol{c}_1, \boldsymbol{s} \rangle \cdot \langle \boldsymbol{c}_2, \boldsymbol{s} \rangle]_q]_t \\
&\equiv [[(t \cdot \langle \boldsymbol{r}_1, \boldsymbol{s} \rangle + \langle \boldsymbol{d}_1, \boldsymbol{s} \rangle \cdot \mu_1) \cdot (t \cdot \langle \boldsymbol{r}_2, \boldsymbol{s} \rangle + \langle \boldsymbol{d}_1, \boldsymbol{s} \rangle \cdot \mu_2)]_q]_t \\
&\equiv [[t \cdot e + \langle \boldsymbol{d}_1, \boldsymbol{s} \rangle^2 \mu_1 \mu_2]_q]_t = [t \cdot e + \langle \boldsymbol{d}_1, \boldsymbol{s} \rangle^2 \mu_1 \mu_2]_t \equiv \mu_1 \cdot \mu_2 \in \mathbb{Z}_t,
\end{aligned}$$

assuming that $|t \cdot e + \langle \boldsymbol{d}_1, \boldsymbol{s} \rangle^2 \mu_1 \mu_2| < q/2$ where we use $t \cdot e \in \mathbb{Z}$ to refer to the decryption noise coming from the other terms to ease readability. Just as for the original BGV scheme, the decryption noise $t \cdot e$ of the product is roughly the square of that of the input ciphertexts. Notice that, again, with a right instantiation as discussed in Section 3.3, the original decryption noise is sufficiently small relative to q and thus for q sufficiently large the above multiplication can be correctly decrypted.

Even if the noise is bounded by say b, independently of q, it still is roughly squared after a single multiplication, and therefore grows exponentially on repeated multiplications, allowing us to do at most $\log_b(q)$ of them. Similarly, the dimension of the lattice and the ciphertext space is potentially squared as $\dim(\mathcal{L} \otimes \mathcal{L}) = \dim(\mathcal{L})^2$ and $\dim(\mathcal{C} \otimes \mathcal{C}) = \dim(\mathcal{C})^2$ if $q = p^k$ for some prime p. Just as in the LWE case we can solve these two problems by modulus switching and key switching, respectively. Notice that key switching for Lattice-BGV will be analyzed in Appendix D.2.

Studying the error growth. Before analyzing more advanced primitives, we want to give some formal definitions of the error $\boldsymbol{e} \in \mathcal{D}$, the decryption error $|\langle \boldsymbol{e}, \boldsymbol{s} \rangle|$ and the correctness of decryption in the Lattice-BGV scheme.

Definition 4.1 (Error in a Lattice-BGV ciphertext). *A ciphertext $\boldsymbol{c} \in \mathcal{D}/q\mathcal{L}^*$ contains an error $\boldsymbol{e} \in \mathcal{D}$ that is defined recursively depending on the sequence of homomorphic operations applied to it.*

- *Base case (fresh encryption). A* fresh ciphertext *$\boldsymbol{c} \in \mathcal{D}/q\mathcal{L}^*$ is generated by hiding some error $\boldsymbol{e} \in \mathcal{D}$ as in Eq. (4.1):*

$$\boldsymbol{e} := t\boldsymbol{r} + \boldsymbol{d}_1 \mu \in \mathcal{D}, \quad \textit{where } \boldsymbol{r} \sim \mathcal{G}_{\mathcal{D}, \sigma_{enc}}.$$

  we say that $\boldsymbol{c}$ contains (or hides) error $\boldsymbol{e}$, where $\boldsymbol{e} \in \mathcal{D}$ is the error *associated to $\boldsymbol{c}$ as $\boldsymbol{c} \equiv \boldsymbol{e} \bmod \mathcal{D}/q\mathcal{L}^*$.*

– *Recursive step (homomorphic operations). Let* $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathcal{D}/q\mathcal{L}^*$ *be two independent ciphertexts such that* $\boldsymbol{c}_1$ *has error* $\boldsymbol{e}_1$ *and* $\boldsymbol{c}_2$ *has error* $\boldsymbol{e}_2$*. Then the error of the resulting ciphertext after each homomorphic operation is defined as:*

$$\begin{aligned} \textit{(Addition)} \quad &= \boldsymbol{e}_1 + \boldsymbol{e}_2 = t(\boldsymbol{r}_1 + \boldsymbol{r}_2) + \boldsymbol{d}_1(\mu_1 + \mu_2), \\ \textit{(Tensor product)} \quad &= \boldsymbol{e}_1 \otimes \boldsymbol{e}_2 = (t\boldsymbol{r}_1 + \boldsymbol{d}_1\mu_1) \otimes (t\boldsymbol{r}_2 + \boldsymbol{d}_1\mu_2), \\ \textit{(Multiplication by } \alpha \in \mathbb{Z}) \quad &= \boldsymbol{e} \cdot \alpha = \alpha t\boldsymbol{r} + \boldsymbol{d}_1(\alpha\mu), \end{aligned}$$

This recursive formulation allows one to compute the error contained in any ciphertext by following the structure of the homomorphic computation tree leading to it.

Definition 4.2 (Decryption error in a Lattice-BGV ciphertext). *Given the previously defined error in ciphertext, the* decryption error *in a ciphertext* $\boldsymbol{c} \equiv \boldsymbol{e} \bmod \mathcal{D}/q\mathcal{L}^*$ *is measured considering the error with respect to the short primal vector* $\boldsymbol{s} \in \mathcal{L}$ *as* $|\langle \boldsymbol{e}, \boldsymbol{s} \rangle| < \|\boldsymbol{e}\|_2 \cdot \|\boldsymbol{s}\|_2$.

Notice that the decryption error is bounded by both the error contained in a ciphertext and the norm of the short vector used to decode. In other words, the shorter the vector $\boldsymbol{s} \in \mathcal{L}$, the smaller the decryption error.

Decoding by short primal vector(s) and correctness. In all presented schemes, decryption is performed by using the shortest vector in the primal lattice. We explain what is needed for correctness in a more general decoding perspective, to indicate what could potentially be gained from alternative decoding methods. Given the one-dimensional lattice $S = s\mathbb{Z} \subset \mathcal{L}$ that is generated by $\boldsymbol{s} \in \mathcal{L}$, this process can be defined as (i) the BDD target $\boldsymbol{e} \in \mathcal{D} \subset \mathcal{L}^*$ is projected onto $\mathrm{span}(S)$ as $\pi_{\boldsymbol{s}}(\boldsymbol{e})$ and (ii) the projected point $\pi_{\boldsymbol{s}}(\boldsymbol{e})$ is rounded to the closest lattice point in the dual $\pi_s(\mathcal{L}^*) = S^* = \frac{\boldsymbol{s}}{\|\boldsymbol{s}\|^2}\mathbb{Z}$. Remark that

$$\pi_{\boldsymbol{s}}(\boldsymbol{e}) = \mathrm{proj}_{\boldsymbol{s}}(\boldsymbol{e}) = \langle \boldsymbol{e}, \boldsymbol{s} \rangle \cdot \boldsymbol{s}/\|\boldsymbol{s}\|^2.$$

In other words, the decryption $\langle \boldsymbol{e}, \boldsymbol{s} \rangle$ can be interpreted as "how far along the direction $\boldsymbol{s}$ the error $\boldsymbol{e}$ lies", in lattice terminology, it is the rounding error of $\pi_{\boldsymbol{s}}(\boldsymbol{e})$ in S^*. In general, to round correctly, we need that the error of some target value must be smaller than half the minimum distance $\lambda_1(S^*)$. A ciphertext hiding $\boldsymbol{e}$ is correctly rounded if the norm of its projection $\|\pi_{\boldsymbol{s}}(\boldsymbol{e})\|$ is smaller than half the shortest vector of S^* (also see Fig. 3.1, where $\lambda_1(S^*)$ is the green vector in (b)). Since $\|\pi(\boldsymbol{e})\| \leq \|\boldsymbol{e}\|$, we have that $\|\boldsymbol{e}\| < \lambda_1(S^*)/2$ to be correctly decryptable. The shortest vector of S^* has norm $\lambda_1(S^*) = 1/\|\boldsymbol{s}\|$, and since we usually work with q-duals (see beginning of Section 3), $\lambda_1(qS^*) = q/\|\boldsymbol{s}\|$. By putting everything together, we have that $\|\boldsymbol{e}\| < q/(2\|\boldsymbol{s}\|)$ to prove correctness of decoding. Finally, by considering the decryption error, we have that an error $\boldsymbol{e}$ can always be uniquely decoded if:

$$|\langle \pi_s(\boldsymbol{e}), \boldsymbol{s} \rangle| \leq \|\pi_s(\boldsymbol{e})\| \cdot \|\boldsymbol{s}\| \leq \|\boldsymbol{e}\| \cdot \|\boldsymbol{s}\| < \frac{q}{2\|\boldsymbol{s}\|} \cdot \|\boldsymbol{s}\| = \frac{q}{2}.$$

Notice that, in our setup, the error $\boldsymbol{e}$ is composed of both some random noise $\boldsymbol{r} \in \mathcal{G}_{\mathcal{L}^*,\sigma_{\text{enc}}}$ and an encoded message $\mu \in \mathbb{Z}_t$ in the first public basis vector $\boldsymbol{d}_1$ of $\mathcal{D}$. In a standard public key scheme encrypting values from $\mathbb{Z}_t$, we prove correctness by showing that the norm of $\boldsymbol{r}$ contained in an error $\boldsymbol{e}$ is such that:

$$\|\boldsymbol{r}\| < \frac{q}{2t} - t\|\boldsymbol{d}_1\|.$$

Conversely, there is a non negligible probability to sample $\boldsymbol{r}$ in such a way that $\|\boldsymbol{r}\| > q/(2t)$ – or equivalently $t\|\boldsymbol{r}\| > q/2$ – changing the congruence modulo t of the underlying message. For instance, if we want to encrypt a message from $\mathbb{Z}_2$, the norm of $\boldsymbol{r}$ can be at most roughly $q/4$. If the plaintext space is $\mathbb{Z}_4$, the norm of $\boldsymbol{r}$ can be at most roughly $q/8$, and so on. Notice that similar arguments can be done to prove correctness in the case of encoding some message $\mu \in \mathbb{Z}_t$ as $\mu \cdot q/t$, without scaling $\boldsymbol{r}$.

In a homomorphic encryption setup, proving actual correctness with overwhelming probability for a ciphertext containing the homomorphic evaluation of some polynomial p is more challenging, and still an open problem, even in the standard LWE setting [MML+23]. Intuitively, one should verify correctness for fresh ciphertext, and set parameters that allow for the correct decryption of homomorphic operations, following the recursive definition of error in Definition 4.1.

Lemma 4.1 (Correctness of decryption). *A ciphertext $\boldsymbol{c} \in \mathcal{D}/q\mathcal{L}^*$ containing an error $\boldsymbol{e}$ is correctly decryptable using the short dual vector $\boldsymbol{s} \in \mathcal{L}$ if its error is such that*

$$\|\boldsymbol{e}\| < \frac{1}{2}\lambda_1(qS^*) = \frac{q}{2\|\boldsymbol{s}\|},$$

hence, its decryption error is such that

$$|\langle \boldsymbol{e}, \boldsymbol{s}\rangle| \leq \|\boldsymbol{e}\| \cdot \|\boldsymbol{s}\| < \left\langle \frac{q}{2\|\boldsymbol{s}\|}, \|\boldsymbol{s}\| \right\rangle = \frac{q}{2}$$

This guarantees unique and correct decoding.

For instance, for a fresh error $\boldsymbol{e} \equiv t\boldsymbol{r} + \boldsymbol{d}_1\mu$, it must be such that $\|\boldsymbol{r}\| < q/(2t) - \|\boldsymbol{d}_1\|t$, so σ_{enc} should be set accordingly.

Average cases and upper bounds for errors. We will show how large decryption errors are on average in fresh ciphertexts and how they can be upper bounded with overwhelming probability. Similar results for additions and multiplications can be easily obtained following their corresponding error definitions. For multiplications, simply consider the property $\|\boldsymbol{c}_1 \otimes \boldsymbol{c}_2\|_2^2 = \|\boldsymbol{c}_1\|_2^2\|\boldsymbol{c}_2\|_2^2$, so the norm of some tensor product is simply the product of the norms of the two multiplied vectors.

Lemma 4.2 (Fresh decryption error, on average). *The average amount of decryption error in a fresh ciphertext in $\mathcal{D}/q\mathcal{L}^*$ is smaller than*

$$(t \cdot \sigma_{enc})(\sqrt{m} + 1/2) \cdot \|\boldsymbol{s}\|_2$$

with $\boldsymbol{s}$ being the shortest vector $\boldsymbol{s} \in \mathcal{L}$ used to decrypt, and $m := \dim(\mathcal{L}) \geq 2$.

Corollary 4.1 (Fresh decryption error, upper bound). *The amount of decryption error in a fresh ciphertext is at most*

$$(t \cdot \sigma_{enc})(\sqrt{m} + \sqrt{2\ln(2)} + 1/2) \cdot \|\boldsymbol{s}\|_2$$

with $\boldsymbol{s}$ being the shortest vector $\boldsymbol{s} \in \mathcal{L}$ used to decrypt, and $m := \dim(\mathcal{L}) \geq 2$.

Both proofs can be found Appendix C. For simplicity these bounds use the worst-case inequality $|\langle \boldsymbol{e}, \boldsymbol{s} \rangle| \leq \|\boldsymbol{e}\| \cdot \|\boldsymbol{s}\|$, which is typically off by a factor $\Theta(\sqrt{m})$. Both could be formally improved by a more involved argument when σ_{Enc} is sufficiently above the smoothing parameter of $\mathcal{D}$. Depending on the parametrization we do not always have this guarantee and thus opted for these simpler bounds.

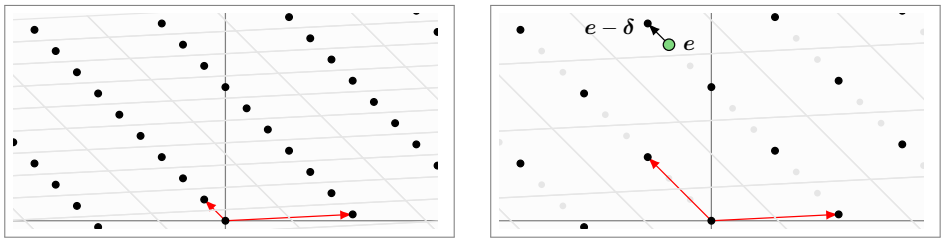
Following usual noise estimations done for BGV, we can set encryption parameters such that the modulus switching reduces the decryption error $|\langle \boldsymbol{e}, \boldsymbol{s} \rangle|$ after a multiplication back to the noise of a fresh encryption. Then, by setting $q \geq \|\langle \boldsymbol{e}, \boldsymbol{s} \rangle\|_2^{\ell+1}$, with $\boldsymbol{e}$ being the error in a fresh ciphertext, this would allow for the evaluation of (roughly) up to ℓ multiplications.

A concrete option could be to pick $q = \Omega((t \cdot \sigma_{\text{enc}})(\sqrt{n} + \sqrt{2\ln(2)} + 1/2) \cdot \|\boldsymbol{s}\|_2)^{\ell+1}$, from Corollary 4.1 which allows for ℓ multiplications, followed by modulus switching, with overwhelming probability depending on m. Notice that of course this is a worst-case bound that is not really practical. Just like in LWE, performance are improved by using short keys, hence it seems natural in our case to use lattices that admit extremely short vectors $\boldsymbol{s} \in \mathcal{L}$, which in turn would allow to use smaller moduli and parameters. Notice that Algorithm B.1 allows to construct an ambient space with respect to arbitrary modulus $q = p^k$ and lattice $\mathcal{L}$, so this is feasible in our setup and security can be improved by setting a sufficiently large dimension n, so that the BDD targets can not be decoded trivially.

Modulus switching. Having a clear view and more freedom in terms of lattices allows us to define a more generic approach to modulus switching. In this more general setting, modulus switching has the goal to switch from some initial modulus q_1 to a smaller modulus q_2 such that $q_2 \,|\, q_1$. In particular, given $q_1\mathcal{L}^* \subset \mathcal{D}_1$ and $q_2\mathcal{L}^* \subset \mathcal{D}_2$ with their respective ambient spaces, the aim is to switch a ciphertext $\boldsymbol{c}_1 \in \mathcal{D}_1/q_1\mathcal{L}^*$ to a ciphertext $\boldsymbol{c}_2 \in \mathcal{D}_2/q_2\mathcal{L}^*$, while reducing the error by roughly a factor q_1/q_2 and while keeping the encoded message intact. We assume that $\mathcal{D}_1$ and $\mathcal{D}_2$ satisfy the properties of Definition 3.1 for moduli q_1 and q_2 respectively, and such that $q_1/q_2 \cdot \mathcal{D}_2 \subset \mathcal{D}_1$, i.e., $(q_1/q_2)\mathcal{D}_2$ is a sparsification of $\mathcal{D}_1$ (see Fig. 4.1).

Similarly to the original LWE case, we proceed in two steps which can be summarized by the following:

$$\mathcal{D}_1/q_1\mathcal{L}^* \xrightarrow[\text{round}]{-\boldsymbol{\delta}_{\boldsymbol{c}_1}} (q_1/q_2)\mathcal{D}_2/q_1\mathcal{L}^* \xrightarrow[\text{scale}]{\times q_2/q_1} \mathcal{D}_2/q_2\mathcal{L}^*,$$



(a) An ambient space $\mathcal{D}_1$ (b) Rounding the error in $q_1/q_2\mathcal{D}_2$

Fig. 4.1: Defining a sparser sublattice $q_1/q_2\mathcal{D}_2 \subset \mathcal{D}_1$ from $\mathcal{D}_1$, assuming $q_1/q_2 = 3$ and $\mathcal{D}_1 = \mathcal{D}_2$ for simplicity. Notice that all the points in $q_1/q_2\mathcal{D}_2$ are divisible by q_1/q_2 by definition

where in the first step we *round* to the sparsification $(q_1/q_2)\mathcal{D}_2$ of the ambient space $\mathcal{D}_1$, and then *scale* down to the new ambient space $\mathcal{D}_2$ with respect to the lattice $q_2\mathcal{L}^*$. We first focus on the rounding part, where, given some ciphertext $\boldsymbol{c}_1 \in \mathcal{D}_1/q_1\mathcal{L}^*$, we need to find some small $\boldsymbol{\delta}_{\boldsymbol{c}_1}$ such that $\boldsymbol{c}_1 - \boldsymbol{\delta}_{\boldsymbol{c}_1} \in (q_1/q_2)\mathcal{D}_2/q_1\mathcal{L}^*$. Note that this is precisely an (approximate) syndrome decoding problem in $\mathcal{D}_1/(q_1/q_2)\mathcal{D}_2$, where $\boldsymbol{\delta}_{\boldsymbol{c}_1}$ is a small error in the coset $\boldsymbol{c}_1 + (q_1/q_2)\mathcal{D}_2/q_1\mathcal{L}^*$. In particular, the better we can decode in $(q_1/q_2)\mathcal{D}_2$, the smaller $\boldsymbol{\delta}_{\boldsymbol{c}_1}$ can be.

To decode we can again use the good basis that our ambient space $\mathcal{D}_2$, and thus $(q_1/q_2)\mathcal{D}_2$ has, generalizing the LWE setting where decoding in $(q_1/q_2)\mathbb{Z}^m$ is also possible due to a good (orthogonal) basis. In particular, we can use Babai's decoding algorithm which always finds a $\boldsymbol{\delta}_{\boldsymbol{c}_1}$ such that $\|\boldsymbol{\delta}_{\boldsymbol{c}_1}\|^2 \leq \frac{q_1^2}{4q_2^2}\sum_{i=1}^m \|\tilde{\boldsymbol{d}}_i\|^2$, where the latter are the Gram-Schmidt norms of the basis D_2.The dependency on q_1/q_2 is natural as it directly determines the sparsity of $q_1/q_2\mathcal{D}_2$, and that factor will be removed by the scaling step. Now that the $\boldsymbol{c}_1 - \boldsymbol{\delta}_{\boldsymbol{c}_1}$ point lies in $q_1/q_2\mathcal{D}_2 \subset \mathcal{D}_1$, it is possible to scale it down, along with the lattice $q_1\mathcal{L}^*$, to obtain a new ciphertext with a smaller error with respect to the lattice $q_1\mathcal{L}^*/(q_1/q_2) = q_2\mathcal{L}^*$. As a result, we obtain a ciphertext that, although scaled from the original lattice $q_1\mathcal{L}^*$ to $q_2\mathcal{L}^*$, has an error equal to $(\boldsymbol{e}_1 - \boldsymbol{\delta}_{\boldsymbol{c}_1})/(q_1/q_2)$, which, ignoring $\boldsymbol{\delta}_{\boldsymbol{c}_1}$, is a factor q_1/q_2 shorter than the error $\boldsymbol{e}_1$ of $\boldsymbol{c}_1$.

Despite this straightforward procedure, we still require caution as this operation could erase the content of the message stored in the first component. In particular, the decoding algorithm could return a ciphertext that decrypts to a different message than the one originally contained in $\boldsymbol{c}_1$. In LWE, this is tackled by using a $\boldsymbol{\delta}_{\boldsymbol{c}_1}$ term that is a multiple of the plaintext modulus t, so that during decryption can easily be removed. The difference is how the rounding is performed: in LWE, the delta to the closest vector in the sparse lattice is given by $t \cdot A_2^t \cdot [A_2^{-t} \cdot \boldsymbol{c}_1 \cdot t^{-1}]_{q_1/q_2}$, which implicitly acts as a syndrome decoding algorithm in $t\mathcal{D}_1/(q_1/q_2)\mathcal{D}_2 \cong \mathbb{Z}_{q_1/q_2}^n$, returning a small $\boldsymbol{\delta}_{\boldsymbol{c}_1} \in t\mathcal{D}_1$. Mimicking this, combined with our more general syndrome decoding approach, we can therefore

compute the $\boldsymbol{\delta}_{\boldsymbol{c}_1}$ term as follows:

$$\boldsymbol{\delta}_{\boldsymbol{c}_1} = t \cdot \text{approxSDP}(\boldsymbol{c}_1 \cdot t^{-1}, (q_1/q_2)\mathcal{D}_2) \in t\mathcal{D}_1, \tag{4.7}$$

where $\text{approxSDP}(\boldsymbol{t}, \mathcal{L}')$ is an efficient procedure that returns a small error $\boldsymbol{e} \in \boldsymbol{t} + \mathcal{L}'$. Finally, we can formalize the general modulus switching procedure

$$\begin{aligned} \mathcal{L}\text{-BGV.GeneralModSwitch} &: \mathcal{D}_1 \times \mathbb{N} \times \mathbb{N} \to \mathcal{D}_2, \\ \mathcal{L}\text{-BGV.GeneralModSwitch}(\boldsymbol{c}_1, q_1, q_2) &:= \left[\left((\boldsymbol{c}_1 - \boldsymbol{\delta}_{\boldsymbol{c}_1}) \cdot \frac{q_2}{q_1} \right) \right]_{q_2}. \end{aligned} \tag{4.8}$$

Lastly, we introduce a lemma that shows how the decryption error in a ciphertext actually decreases under appropriate parameters.

Lemma 4.3. *Let $q_1 > q_2 > t$ be positive integers such that $q_1 \equiv q_2 \equiv 1 \bmod t$. Let $\mathcal{D}_1$ and $\mathcal{D}_2 \subset (q_2/q_1)\mathcal{D}_1$ be ambient spaces satisfying Definition 3.1. Given a $\mathcal{L}$-BGV ciphertext $\boldsymbol{c}_1 \in \mathcal{D}_1/q_1\mathcal{L}_1^*$, the $\mathcal{L}$-BGV.GeneralModSwitch$(\boldsymbol{c}_1, q_1, q_2)$ function returns $\boldsymbol{c}_2 \in \mathcal{D}_2/q_2\mathcal{L}^*$ such that*

- *(Correctness):* $[\langle \boldsymbol{c}_1, \boldsymbol{s} \rangle]_{q_1} \equiv [\langle \boldsymbol{c}_2, \boldsymbol{s} \rangle]_{q_2} \bmod t$
- *(Decryption error reduction):* $|[\langle \boldsymbol{c}_2, \boldsymbol{s} \rangle]_{q_2}| \leq (q_2/q_1)\,|[\langle \boldsymbol{c}_1, \boldsymbol{s} \rangle]_{q_2}| + \rho\|\boldsymbol{s}\|$

where ρ is the approximate decoding radius of the efficient decoder approxSDP on $\mathcal{D}_2$. Using a basis D_2 of $\mathcal{D}_2$ we have $\rho \leq \frac{1}{2}\sqrt{\sum_{i=1}^m \|\tilde{\boldsymbol{d}}_i\|^2}$, where $(\tilde{\boldsymbol{d}}_i)_i = \tilde{D}_2$.

The proof of Lemma 4.3 can be found in Appendix C. The bounds could be improved by either a heuristic average-case argument, or by provable bounds by replacing the decoding step by Discrete Gaussian Sampling over $\mathcal{D}_2$. In theory, the approximate decoding radius ρ can be as low as the covering radius $\nu(\mathcal{D}_2) \leq \rho$ of $\mathcal{D}_2$. For LWE we have $\mathcal{D}_2 = \mathbb{Z}^m$ and $\rho = \nu(\mathcal{D}_2) = \frac{1}{2}\sqrt{n}$, which from the lattice perspective has an optimal covering radius up to a constant factor relative to the normalised volume of the lattice. In theory, using lower bounds on the covering radius, a better lattice could improve ρ by a factor up to $\approx \sqrt{\pi e/2} \approx 2.066$.

4.2 Concrete parameters and requirements

Let $C = B^{-t}$ and D be the public bases of $\mathcal{L}^*$ and the ambient space $\mathcal{D}$ respectively. The basis D combined with Lemma 2.1, allows to sample small errors in $\mathcal{D}$ from a discrete Gaussian with any parameter $\sigma_{\text{enc}} \geq \max \|\tilde{\boldsymbol{d}}_i\| \cdot \sqrt{\ln(2m+4)/\pi}$. For a small error $\boldsymbol{r}$, sampled from this distribution, the decryption error $\langle \boldsymbol{r}, \boldsymbol{s} \rangle$ with the short secret vector $\boldsymbol{s} \in \mathcal{L}$ will roughly be of size $1/2 \cdot t \cdot \sigma_{\text{enc}} \cdot \|\boldsymbol{s}\|$ (Lemma 4.2), which we have to compare to the modulus q.

A choice for q would be a value $1 \leq q \leq \max \|\tilde{\boldsymbol{d}}_i\| / \min \|\tilde{\boldsymbol{c}}_i\|$ which guarantees that $\sigma_{\text{enc}} > q \min \|\tilde{\boldsymbol{c}}_i\|/2$, that is a minimum requirement to make the resulting syndrome decoding problem with respect to $q\mathcal{L}^*$ non-trivial to solve[7].

[7] For a secure instantiation, and depending on the precise basis qC of $q\mathcal{L}^*$, one could need a slightly larger margin, for simplicity we will ignore this minor dependency.

By increasing q we can thus make the lattice $q\mathcal{L}^*$ sparser and allow for a larger decryption error, so naturally one would choose the maximum value for q. Notice that this is also precisely the choice that is made for LWE-schemes, where $q = q/1 = \max \|\tilde{\boldsymbol{d}}_i\| / \min \|\tilde{\boldsymbol{c}}_i\|$.

We thus observe that a smaller secret vector $\boldsymbol{s} \in \mathcal{L}$ reduces the decryption error while a worse public basis C of $\mathcal{L}^*$ (typically giving larger $\max \|\tilde{\boldsymbol{d}}_i\| \geq \max \|\tilde{\boldsymbol{c}}_i\|$ and smaller $\min \|\tilde{\boldsymbol{c}}_i\|$) increases it but can increase the modulus q even more. Recall that for the BGV scheme the decryption error is roughly squared after a single multiplication, so the minimum requirement on $\|\boldsymbol{s}\|$ would be that $(\|\boldsymbol{s}\| \cdot \sigma_{\text{enc}}) < \mathcal{O}(\sqrt{q})$.

To allow for many homomorphic multiplications, however, the decryption error of fresh ciphertexts should be close to constant with respect to q. In particular, one would preferably have $\|\boldsymbol{s}\| \sim \sigma_{\text{enc}}^{-1} \approx \max \|\tilde{\boldsymbol{d}}_i\|^{-1}$. Notice that for the LWE case we have precisely this, as $\|\boldsymbol{s}\| = \Theta(\sqrt{n})$ and $\max \|\tilde{\boldsymbol{d}}_i\| = 1$, and $\Theta(\sqrt{n}) \ll q$. In the rest of this work we assume to have a small enough secret $\boldsymbol{s} \in \mathcal{L}$ such that we can assume that the decryption error of fresh ciphertexts is sufficiently small relative to the modulus q to perform multiplications. In our instantiation based on LIP (Section 5), we will shortly explain how to achieve this.

4.3 Lattice-GSW

We can now use previous results to define the Lattice-GSW scheme. The key generation is the same as in the previous scheme: the public key is a bad basis B of a lattice $\mathcal{L}$ along with a good basis D of the ambient space $q\mathcal{L}^* \subset \mathcal{D} \subset \mathcal{L}^*$ where $q = p^k$ for some $p = t \geq 2$, and the secret key $\boldsymbol{s} \in \mathcal{L}$ is a sufficiently short vector of $\mathcal{L}$. The message message space will be $\mathcal{M} := \mathbb{Z}_p$. For the GSW scheme and its operations, it is useful to express elements more explicitly in their coefficient representation. More precisely, we assume that the bases B^{-t} and D of $\mathcal{L}^*$ and $\mathcal{D} \subset \mathcal{L}^*$ respectively differ from each other by a transformation in Hermite Normal Form such that B^t explicitly maps single ciphertext syndromes into a diagonalized coefficient group $\mathcal{D}/q\mathcal{L}^* \to \prod_{i=1}^m p^{j_i}\mathbb{Z}/q\mathbb{Z} \subset \mathbb{Z}_q^m$ for some $0 \leq j_i \leq \log_p(q)$. Furthermore we assume that $j_1 = 1$ and thus B^{-t} and D have the same first vector. Both our ambient space constructions satisfy this property. Let $\ell = \log_p(|\mathcal{D}/q\mathcal{L}^*|) = q \cdot m - \sum_{i=1}^m j_i$, then the ciphertext space equals $\mathcal{C} := (\mathcal{D}/q\mathcal{L}^*)^\ell$.

Encryption and decryption. As observed in Section 2.4, the encryption process of GSW is slightly different from the BGV one. In particular, a ciphertext will now be a matrix consisting of ℓ encryptions of zero from Lattice-BGV (with $t = 1$), one for each column. So we first define the error matrix E with columns in $\mathcal{D}$ such that:

$$E := \begin{pmatrix} | & | & & | \\ \boldsymbol{e}_1 & \boldsymbol{e}_2 & \dots & \boldsymbol{e}_\ell \\ | & | & & | \end{pmatrix} \text{ where each } \boldsymbol{e}_i \sim \mathcal{G}_{\mathcal{D},\sigma_{\text{enc}}}.$$

We identify this matrix space with $\mathcal{D}^\ell$. Then the syndrome matrix $E \bmod q\mathcal{L}^* \in (\mathcal{D}/q\mathcal{L}^*)^\ell$ represents ℓ encryptions of zero (equivalent to $[A^tR]_q$ in LWE-based GSW). In order to encrypt a message $\mu \in \mathbb{Z}_p$, we simply add a multiple $\mu B^{-t}G$ of the dual basis times the gadget matrix to the coefficient representation matrix E. We just have to be careful as the gadget matrix must be constructed with respect to the base p and to match the explicit group structure of $\mathcal{D}/q\mathcal{L}^*$ mentioned above to make sure that $\mu B^{-t}G \in (\mathcal{D}/q\mathcal{L}^*)^\ell$. To do this one should construct G as before, but with $(p^{j_i}, \dots, p^{\log_p(q)-1})$ on row i. The inverse map G^{-1} is also defined as before, noting that one can always write an element in $p^{j_i}\mathbb{Z}/q\mathbb{Z}$ in the basis $p^{j_i}, \dots, p^{\log_p(q)-1}$ and coefficient in $0, \dots, p-1$. Encryption is thus defined as follows.

$$\mathcal{L}\text{-GSW.Enc}_{pk} : \mathcal{M} \to \mathcal{C}, \quad \mathcal{L}\text{-GSW.Enc}_{pk}(\mu) := E + B^{-t}G\mu \in \mathcal{C}, \tag{4.9}$$

Decryption follows from GSW, it is possible to extract the j-th column of $C \in \mathcal{C}$ corresponding to $\boldsymbol{c}_j = \boldsymbol{e}_j + q/p\boldsymbol{d}_1$, and use the secret key $\boldsymbol{s}$ as follows:

$$\mathcal{L}\text{-GSW.Dec}_{sk} : \mathcal{C} \to \mathcal{M}, \quad \mathcal{L}\text{-GSW.Dec}_{sk}(C) := \lfloor [\langle \boldsymbol{c}_j, \boldsymbol{s} \rangle]_q \rceil_2 \in \mathcal{M} \tag{4.10}$$

This procedure is correct, as shown below:

$$[\langle \boldsymbol{c}_j, \boldsymbol{s} \rangle]_q \rceil_p \equiv \lfloor [e + \langle \boldsymbol{d}_1, \boldsymbol{s} \rangle (q/p)\mu]_q \rceil_p \equiv \mu \in \mathcal{M}$$

where, as in Lattice-BGV, $e = \langle \boldsymbol{e}_j, \boldsymbol{s} \rangle$ is small, and we assumed that $\langle \boldsymbol{d}_1, \boldsymbol{s} \rangle \equiv 1 \bmod p$. Notice that e must be roughly smaller than $q/2p$ to allow for correct decryption. Homomorphic additions immediately follow:

$$\mathcal{L}\text{-GSW.Add} : \mathcal{C} \times \mathcal{C} \to \mathcal{C}, \quad \mathcal{L}\text{-GSW.Add}(C_1, C_2) := C_1 + C_2 \in \mathcal{C} \tag{4.11}$$

The multiplication between two ciphertexts almost immediately follows the original GSW one. Given two ciphertexts $C_1, C_2 \in (\mathcal{D}/q\mathcal{L}^*)^\ell$, – encrypting respectively μ_1 and μ_2 – we define the multiplication as $C^\times := C_1 \cdot G^{-1}(B^tC_2)$. Note the transformation B^tC_2 to the coefficient representation which is required to apply G^{-1}.

$$\mathcal{L}\text{-GSW.Mul} : \mathcal{C} \times \mathcal{C} \to \mathcal{C}^2, \quad \mathcal{L}\text{-GSW.Mul}(C_1, C_2) := C_1 \cdot G^{-1}(B^tC_2) \in \mathcal{C} \tag{4.12}$$

We now show that the decryption of multiplications is correct, i.e., we obtain a new ciphertext

$$\begin{aligned} C_1^\times &\equiv (C_1 \cdot G^{-1}(B^tC_2)) \\ &\equiv (E_1 + \mu_1 \cdot B^{-t}G) \cdot G^{-1}(B^tC_2) \\ &\equiv E_1 \cdot G^{-1}(B^tC_2) + \mu_1 \cdot C_2 \\ &\equiv (E_1 \cdot G^{-1}(B^tC_2) + \mu_1 \cdot E_2) + (\mu_1 \cdot \mu_2) \cdot B^{-t}G \\ &\equiv E + (\mu_1 \cdot \mu_2) \cdot B^{-t}G \in \mathcal{C}, \end{aligned} \tag{4.13}$$

that encodes $\mu_1 \cdot \mu_2$ with the new error $E = (E_1 \cdot G^{-1}(B^tC_2) + \mu_1 \cdot E_2)$. As both μ_1 is small, and the coefficients of $G^{-1}(B^tC_2)$ are bounded by $\ell \cdot (p-1)$, the columns of E consist of small combinations of the initial error columns. For appropriate parameters one therefore correctly decrypts the new ciphertext to $\mu_1 \cdot \mu_2 \in \mathcal{M}$.

5 An instantiation using LIP

When instantiating our Lattice-BGV and Lattice-GSW scheme with an LWE-based key generation we precisely obtain the original BGV and GSW schemes respectively. We now show, as an example to illustrate the flexibility of our framework, a high-level instantiation based on the Lattice Isomorphism Problem. For more details on the CPA-security proof in this setting we refer to Appendix A and A.2.

Key generation. Recall that we only need to instantiate the key generation procedure from Definition 3.1. We first focus on the lattice and consider the ambient space later. Recall that the key generation procedure should output a public bad description of a primal lattice $\mathcal{L}$ and a sufficiently short secret trapdoor vector $\boldsymbol{s} \in \mathcal{L}$. A LIP instance is defined as:

$$B' = O \cdot B \cdot U \in \mathrm{GL}_m(\mathbb{R})$$

where B, B' are public material, and refer to the basis of the chosen lattice and a public rotation of it, respectively, and $O \in \mathcal{O}_m(\mathbb{R})$ or $U \in \mathrm{GL}_m(\mathbb{Z})$ are the secret transformations. The problem asks to recover either O or equivalently U (up to an automorphism). Notice that it is possible to use any good basis B – generating a lattice with nice decoding properties – even $B = I_m$ to generate a rotation of $\mathbb{Z}^m$, which is what is essentially used by the HAWK scheme [DPPW22]. For now we will simply assume that the first vector $\boldsymbol{b}_1$ in B is a sufficiently short vector, but note that this also easily allows to have multiple short vectors. This fits our setting well, since we naturally have all the required ingredients:

- $B' = O \cdot B \cdot U \in \mathrm{GL}_m(\mathbb{R})$: bad description of the lattice $\mathcal{L} = \mathcal{L}(B')$.
- $\boldsymbol{s} := O \cdot \boldsymbol{b}_1 \in \mathcal{O}_m(\mathbb{R})$: a short vector $\boldsymbol{s} \in O \cdot \mathcal{L}(B) = \mathcal{L}(B')$.

Notice that one can securely sample the basis B' of a certain quality following [DvW22, Algorithm 1], or by applying the procedure to the dual to create a basis C of the dual lattice $\mathcal{L}^* := \mathcal{L}(B')^*$ of a certain quality.

Fixing a concrete set of parameters. The next step is to generate a basis $D \in \mathbb{R}^{m \times m}$ for the ambient space according to Algorithm B.1. Ignoring some further technical details we assume that the lattice $\mathcal{L}$ is normalized to have volume 1 and that we can sample a dual basis C of $\mathcal{L}^*$ such that $\|\tilde{C}\| \sim \sigma_{\mathrm{pk}}$ for some sufficiently large parameter $\sigma_{\mathrm{pk}} > 0$. Following Lemma B.1 there is a strong dependency between the $\sigma_{\mathrm{pk}} > 0$ value used to sample C, and the allowed modulus q of the scheme. Intuitively, larger values of σ_{pk} correspond to a worse basis, thus in larger spread $\ell_{\max}/\ell_{\min}$ in the Gram-Schmidt norms $\ell_{\min} = \min_i \|\tilde{\boldsymbol{c}}_i\|$, $\ell_{\max} = \max_i \|\tilde{\boldsymbol{c}}_i\|$ of C. This can be visualized in a profile that spreads along a larger range. By analyzing the profile of C as a straight line, following the Geometric Series Assumption, we roughly expect that $\ell_{\max} \sim \sigma_{\mathrm{pk}}$ and $\ell_{\min} \sim \sigma_{\mathrm{pk}}^{-1}$, and thus $(\ell_{\max}/\ell_{\min}) \sim \sigma_{\mathrm{pk}}^2$ for their ratio.

By setting $q = p^k \leq \ell_{\max}/\ell_{\min}$ with $k = \lfloor \log_p(\ell_{\max}/\ell_{\min}) \rfloor$ we can have a modulus q up to roughly σ_{pk}^2 as well while creating a good basis D for the ambient space $\mathcal{D}$ where decoding BDD instances is non trivial. The value of $\sigma_{\text{enc}} > 0$ used to sample encryptions, on the other hand, depends on the quality of the basis D. In particular, because of Lemma 2.1, and for security reasons, we require σ_{enc} to be slightly larger than $\max_i \|\tilde{\boldsymbol{d}}_i\| = \max_i \|\tilde{\boldsymbol{c}}_i\| \sim \sigma_{\text{pk}}$. Note that while σ_{enc} used to sample an encryption error $\boldsymbol{e}$ might be large, what matters is the decryption noise coming from $|\langle \boldsymbol{e}, \boldsymbol{s} \rangle|$ which is roughly of size $\|\boldsymbol{s}\| \cdot \sigma_{\text{enc}} \sim \|\boldsymbol{s}\| \cdot \sigma_{\text{pk}}$, in contrast to the modulus $q \sim \sigma_{\text{pk}}^2$. In particular, after a BGV-multiplication one would obtain a decryption noise of roughly $(\|\boldsymbol{s}\| \cdot \sigma_{\text{pk}})^2$ which needs to be smaller than q. To allow for many multiplications we would therefore preferably have an exceptionally short vector $\boldsymbol{s} \in \mathcal{L}$, or equivalently, start with a lattice such that $\boldsymbol{b}_1 \in \mathcal{L}$ is roughly of norm σ_{pk}^{-1}. Note that this matches the LWE case where after normalizing the volume $\|\boldsymbol{s}\| \sim 1/\sqrt{q} \approx \sigma_{\text{pk}}^{-1}$. One way to obtain this is similar to the instantiation proposed in [DvW22], picking $\mathcal{L} := \frac{1}{\lambda}\mathcal{L}' \oplus \lambda\mathcal{L}'$ for one's favourite lattice $\mathcal{L}'$, where λ allows to arbitrarily decrease its first minimum.

6 Further extensions and open questions

We presented a lattice-based perspective of two well-known homomorphic encryption schemes: BGV and GSW. This opens up new possibilities in terms of constructions, hardness assumptions and for identifying what could potentially be improved. We therefore consider some further extensions and open questions.

Multiple message packing, better decoding and robustness. We encoded a single message in each ciphertext and used a single short vector $\boldsymbol{s} \in \mathcal{L}$ to decrypt it. The generalization to multiple messages is rather straightforward by going from one short secret vector to $k > 1$ short vectors $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_k \in \mathcal{L}$ to pack k messages in a ciphertext at once. One does have to assume some additional modular constraints between the ambient basis D and these vectors, but these can be accounted for in the key generation.

More generally, one could consider a dense sublattice $S \subset \mathcal{L}$ of rank $k \geq 1$, along with an efficient decoding algorithm for S^*. This could significantly increase the unique decoding radius $\frac{1}{2}\lambda_1(qS^*)$ of S^*, and therefore allow for larger errors before the decryption fails. Such a more general approach, along with more complex message encodings, for example using linear codes in $\mathbb{F}_p^k \cong S^*/pS^*$, could also allow for a trade-off between the number of messages that can be packed into a ciphertext versus their robustness against errors.

Better modulus switching and analysis. Besides the usage of better lattices and decoding algorithms in S^*, one could also use better lattices and decoding algorithms for the ambient space in the modulus switching procedure. While the ambient space $\mathbb{Z}^m$ used in LWE already presents a simple decoding algorithm and covering radius, it is not optimal, and better alternatives exist such as the

A_n root lattices, or direct sums of the Leech lattice (see [CS13] for many more interesting lattices).

Alternative benefits can come from replacing the approximate syndrome decoder with a Discrete Gaussian Sampling procedure. While this might increase the decoding errors slightly it should allow for an easily provable average-case analysis of the error growth. Another idea is to keep track of these *known* decoding errors, to potentially compensate for them in a later modulus switching step.

Sparse errors. On a theoretical side the LWE problem is best studied and understood using a (Discrete) Gaussian error distribution over $\mathcal{D} = \mathbb{Z}^m$. In practice however, the error distributions are often heuristically chosen and much narrower, e.g. opting for sparse binary or ternary errors, to decrease the size of the errors. Unfortunately, the specific sparse structure leads to an increase risk from hybrid attacks [EJK20,NMW+24]. One way to avoid this problem is to use Discrete Gaussian sampling on an ambient lattice with a sufficiently small *smoothing parameter* (see [EWY23,BLL25] for examples of such lattices). Doing so, the sampled errors $\boldsymbol{e} \in \mathcal{D}$ can still be close to the minimum distance $\lambda_1(\mathcal{D})$, while provably maintaining a sufficiently high entropy to make hybrid attacks ineffective.

Structured versions. We presented the schemes for plain lattices to focus on the geometric intuition and ideas. This makes the schemes rather impractical in their current form. One could significantly improve this by using structured *module lattices*, and this generalization follows quite immediately by plugging in such a lattice. This also directly explains why RLWE-based schemes allow to pack multiple messages, as for example in a module lattice over a cyclotomic number field one could multiply the short vector $\boldsymbol{s}$ by a root of unity to obtain another short vector, leading to d short vectors in $\mathcal{L}$ where d is the degree of the number field. So while algebraic structure allows for message packing, it is not a necessity and merely follows from the geometric change of the involved lattice.

Another significant benefit of introducing module structure is that it generally decreases the dimension of $\mathcal{D}/q\mathcal{L}^*$ by a factor d, and therefore the dimension after a BGV multiplication by a factor d^2. The extreme case would be the NTRU lattice where $\mathcal{D}/q\mathcal{L}^*$ has dimension 1, and thus still $1^2 = 1$ after a multiplication. Unfortunately, this property also directly lead to the overstretched attacks on FHE-based NTRU schemes [ABD16,KF17,DvW21]. This raises the question: can we obtain such a 1-dimensional ciphertext space in another way while avoiding the overstretched attack, or are these inherently connected?

Identifying improvements and limits. In this work we have coupled the performance of HE operations directly to geometric and algorithmic lattice properties. For example the modulus switching to the covering radius, the error sampling to Discrete Gaussian Sampling and therefore indirectly to the smoothing parameter, and the error ceiling to the minimal distance of the dual S^* of the secret dense sublattice. This coupling, along with the knowledge of bounds on

how good these properties can get, allows us to directly see where LWE-based schemes are close to optimal, and in which parts they could be improved and for how much. In particular, these properties show where LWE-based schemes and their simple short-dual-vector decoding approach reach their limits.

Key switching and bootstrapping. We did not discuss the bootstrapping procedure, although although standard techniques should easily be applied to our schemes, since the decryption circuit is of the sample simple form as in LWE. The bootstrapping and key switching method are mostly making use of the homomorphic properties and the group structure of the ciphertext space, and are not really specific to lattices. This opens the question: can key switching and/or bootstrapping be performed differently, considering specific families of lattices or specific lattice operations? For example, the main goal of key switching is to move from a large dimensional lattice to a lower dimensional one. For lattices such an operation is naturally associated with projections, and in general, BDD instances can still be BDD instances after an appropriate projection. Could we therefore replace or rephrase key switching as some lattice projection?

References

ABD16. Martin Albrecht, Shi Bai, and Léo Ducas. A Subfield Lattice Attack on Overstretched NTRU Assumptions. In *CRYPTO*, pages 153–178, 2016.

ACPS09. Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *Advances in Cryptology - CRYPTO 2009*, pages 595–618, 2009.

Ajt96. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 99–108, 1996.

AP14. Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In *Advances in Cryptology – CRYPTO 2014*, pages 297–314, Berlin, Heidelberg, 2014.

BGGJ20. Christina Boura, Nicolas Gama, Mariya Georgieva, and Dimitar Jetchev. Chimera: Combining ring-lwe-based fully homomorphic encryption schemes. *Journal of Mathematical Cryptology*, 14(1):316–338, 2020.

BGPSD23. Huck Bennett, Atul Ganju, Pura Peetathawatchai, and Noah Stephens-Davidowitz. Just how hard are rotations of $\mathbb{Z}^n$? Algorithms and cryptography with the simplest lattice. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 252–281. Springer, 2023.

BGV12. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 309–325, 2012.

BIP+22. Charlotte Bonte, Ilia Iliashenko, Jeongeun Park, Hilder VL Pereira, and Nigel P Smart. Final: faster fhe instantiated with ntru and lwe. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 188–215. Springer, 2022.

BLL25. Maiara F Bollauf, Maja Lie, and Cong Ling. On gaussian sampling for q-ary lattices and linear codes with lee weight. In *Annual International Cryptology Conference*, pages 321–352. Springer, 2025.

BLP+13. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of Learning with Errors. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 575–584, 2013.

BMM25. Pedro Branco, Giulio Malavolta, and Zayd Maradni. Fully-homomorphic encryption from lattice isomorphism. In *Theory of Cryptography: 23rd International Conference, TCC 2025*, pages 220–252, 2025.

Bra12. Zvika Brakerski. Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP. In *Advances in Cryptology – CRYPTO 2012*, pages 868–886, 2012.

BV11. Zvika Brakerski and Vinod Vaikuntanathan. Efficient Fully Homomorphic Encryption from (Standard) LWE. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 97–106, 2011.

CCP+24. Jung Hee Cheon, Hyeongmin Choe, Alain Passelègue, Damien Stehlé, and Elias Suvanto. Attacks Against the IND-CPA$^{\mathrm{D}}$ Security of Exact FHE Schemes. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, CCS '24, pages 2505–2519, 2024.

CGGI16. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster Fully Homomorphic Encryption: Bootstrapping in Less Than 0.1 Seconds. In *Advances in Cryptology – ASIACRYPT 2016*, pages 3–33, 2016.

CGGI20. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: Fast fully homomorphic encryption over the torus. *Journal of Cryptology*, 33(1):34–91, 1 2020.

CKKS17. Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic Encryption for Arithmetic of Approximate Numbers. In *Advances in Cryptology – ASIACRYPT 2017*, pages 409–437, 2017.

CS13. John Horton Conway and Neil James Alexander Sloane. *Sphere packings, lattices and groups*, volume 290. Springer Science & Business Media, 2013.

DM15. Léo Ducas and Daniele Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In *Advances in Cryptology – EUROCRYPT 2015*, pages 617–640, 2015.

DPPW22. Léo Ducas, Eamonn W. Postlethwaite, Ludo N. Pulles, and Wessel van Woerden. Hawk: Module LIP Makes Lattice Signatures Fast, Compact and Simple. In *Advances in Cryptology – ASIACRYPT 2022*, pages 65–94, 2022.

DvW21. Léo Ducas and Wessel van Woerden. NTRU fatigue: how stretched is overstretched? In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 3–32. Springer, 2021.

DvW22. Léo Ducas and Wessel van Woerden. On the Lattice Isomorphism Problem, Quadratic Forms, Remarkable Lattices, and Cryptography. In *Advances in Cryptology – EUROCRYPT 2022*, pages 643–673, 2022.

EJK20. Thomas Espitau, Antoine Joux, and Natalia Kharchenko. On a dual/hybrid approach to small secret lwe. In *Progress in Cryptology – INDOCRYPT 2020*, pages 440–462, 2020.

EWY23. Thomas Espitau, Alexandre Wallet, and Yang Yu. On gaussian sampling, smoothing parameter and application to signatures. In *International Con-*

ference on the Theory and Application of Cryptology and Information Security, pages 65–97. Springer, 2023.

FV12. Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. IACR Cryptology ePrint Archive, Paper 2012/144, 2012.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178, 2009.

GHS12. Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic Evaluation of the AES Circuit. In *Advances in Cryptology – CRYPTO 2012*, pages 850–867, 2012.

GINX16. Nicolas Gama, Malika Izabachene, Phong Q Nguyen, and Xiang Xie. Structural lattice reduction: generalized worst-case to average-case reductions and homomorphic cryptosystems. In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*, pages 528–558. Springer, 2016.

GMP19. Nicholas Genise, Daniele Micciancio, and Yuriy Polyakov. Building an Efficient Lattice Gadget Toolkit: Subgaussian Sampling and More. In *Advances in Cryptology – EUROCRYPT 2019*, pages 655–684, 2019.

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology – CRYPTO 2013*, pages 75–92, 2013.

KF17. Paul Kirchner and Pierre-Alain Fouque. Revisiting lattice attacks on overstretched NTRU parameters. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 3–26. Springer, 2017.

LM21. Baiyu Li and Daniele Micciancio. On the security of homomorphic encryption on approximate numbers. In *Advances in Cryptology – EUROCRYPT 2021*, pages 648–677, 2021.

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

MML+23. Johannes Mono, Chiara Marcolla, Georg Land, Tim Güneysu, and Najwa Aaraj. Finding and Evaluating Parameters for BGV. In *Progress in Cryptology - AFRICACRYPT 2023*, pages 370–394, 2023.

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In *Advances in Cryptology – EUROCRYPT 2012*, pages 700–718, 2012.

NMW+24. Niklas Nolte, Mohamed Malhou, Emily Wenger, Samuel Stevens, Cathy Li, François Charton, and Kristin Lauter. The cool and the cruel: Separating hard parts of lwe secrets. In *Progress in Cryptology - AFRICACRYPT 2024: 15th International Conference on Cryptology in Africa, Douala, Cameroon, July 10–12, 2024, Proceedings*, pages 428–453, 2024.

RAD78. Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 84–93, 2005.

Rot11. Ron Rothblum. Homomorphic Encryption: From Private-Key to Public-Key. In *Theory of Cryptography – TTC 2011*, pages 219–234, 2011.

vW24. Wessel van Woerden. Dense and smooth lattices in any genus. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 386–417. Springer, 2024.

A IND-CPA security proof

In this section we will consider the IND-CPA security proof of our Lattice-BGV and Lattice-GSW HE schemes based on Assumption 1. For simplicity, we consider the setting where the evaluation or bootstrapping keys are encrypted under a fresh keypair as to prevent any circularity assumptions. As a result we can simply consider the IND-CPA security of the underlying PKE schemes.

Theorem A.1. *The* Lattice-BGV *and* Lattice-GSW *schemes are* IND-CPA *secure under Assumption 1.*

Proof. Both schemes and Assumption 1 share the same key generation procedure $\mathsf{LatticeGen}$ defined in Definition 3.1. Consider one scheme and let $pk, sk = (B, D), s \leftarrow \mathsf{LatticeGen}(1^\lambda)$, $C := B^{-t}, \mathcal{D} := \mathcal{L}(D)$ and consider the parameters q, t, m and σ_{enc} from the scheme. Let $\mathcal{A}(pk)$ be an adversary breaking the IND-CPA security of the scheme. Consider an instance of the $\Delta SDP_{C,D,\sigma_{\text{enc}}}$ problem given sufficiently many syndromes $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_{\text{poly}(m)} \in \mathcal{P}(qC) \cap \mathcal{D}$. We use $\mathcal{A}$ to break the $\Delta SDP_{C,D,\sigma_{\text{enc}}}$ problem. Here the proof slightly deviates for the two schemes.

Lattice-BGV. We consider a single syndrome $\boldsymbol{c}_1$, query the adversary $\mathcal{A}$ for a pair of messages (m_0, m_1) and sample a uniform bit $b \leftarrow \{0, 1\}$. We return the challenge $\boldsymbol{c}^* := t \cdot \boldsymbol{c}_i + D \cdot (m_b, 0, \ldots, 0) \bmod \mathcal{P}(qC)$, record the response $b' \leftarrow \mathcal{A}$ from the adversary, and return $b + b' \bmod 2$ to the $\Delta SDP_{C,D,\sigma_{\text{enc}}}$ problem. Note that if $\boldsymbol{c}_1$ is sampled from $\mathcal{G}_{\mathcal{D},\sigma_{\text{enc}}}$ then $\boldsymbol{c}^*$ follows the same distribution as a regular Lattice-BGV encryption of m_b and therefore the adversary should be biased towards the correct guess $b' = b$ and thus towards $b + b' = 0 \bmod 2$. Otherwise $\boldsymbol{c}_1$ is uniform, and using that $\gcd(q, t) = 1$ for Lattice-BGV we obtain that $t \cdot \boldsymbol{c}_i + D \cdot (m_b, 0, \ldots, 0)$ is still uniform and statistically independent from b, and thus $b + b' \bmod 2$ is 1 half of the time. We thus solve the distinguishing problem $\Delta SDP_{C,D,\sigma_{\text{enc}}}$ with essentially the same complexity and advantage as $\mathcal{A}$.

Lattice-GSW. We proceed similarly but we consider $\log_p(|\mathcal{D}/q\mathcal{L}^*|) \leq m \log_p(q)$ syndromes (where $q = p^k$). Note that if the syndromes have an error sampled from $\mathcal{G}_{\mathcal{D},\sigma_{\text{enc}}}$ we can add μCG to it modulo $\mathcal{P}(qC)$ for the appropriate gadget matrix G to exactly obtain a Lattice-GSW encryption of μ with the correct distribution. From here the argument is identical to the Lattice-BGV case. □

A.1 NTRU and LWE-based instantiations

To indicate that the key generation (Definition 3.1), the security assumption (Assumption 1) and therefore also the IND-CPA security proof are proper generalisations of the original LWE-based BGV and GSW schemes, we shortly discuss their instantiation. Recall that one can instantiate the key generation procedure with LWE by creating a lattice $\mathcal{L} := \mathcal{L}_q(A)$ with a short trapdoor vector $\boldsymbol{s} \in \mathcal{L}_q(A)$. Implicitly this creates a (public basis of) the q-dual lattice $\mathcal{L}_q^\perp(A)$ along with an identity basis I_m of the ambient space $\mathcal{D} := \mathbb{Z}^m$.

As explained before the security proof proceeds in two steps. Firstly, by the LWE assumption (the public basis of) $\mathcal{L}^\perp(A)$ is indistinguishable from a random q-ary lattice. After replacing $\mathcal{L}^\perp(A)$ with such a random q-ary lattice in the security game, we see that Assumption 1 becomes precisely the (syndrome version of) the LWE assumption, and the remainder of the proof proceeds as in Theorem A.1.

Note that one could similarly also replace the key generation procedure with an NTRU-based assumption. As we do not consider module lattices in this work we will simply consider the matrix-NTRU assumption. With matrix-NTRU one can generate a lattice $\mathcal{L}$ with a trapdoor dense sublattice $S \subset \mathcal{L}$, which in particular contains a short trapdoor vector $s \in S \subset \mathcal{L}$. Under a distinguish matrix-NTRU assumption this lattice is again indistinguishable from a random q-ary lattice. In this setting Assumption 1 thus again becomes the LWE assumption, and the proof proceeds as before. One can thus also instantiate our lattice framework with a combination of the NTRU and LWE assumptions, which is precisely the strategy that is followed by modern FHE schemes like FINAL [BIP+22].

A.2 Security proof under LIP instantiation

In Section 5 we considered an instantiation of our Lattice-BGV scheme based on the Lattice Isomorphism Problem. Here we give some more details how to obtain an IND-CPA security proof in this specific setting. We can consider two general approaches for this. First, one can consider the approach considered in the work by Ducas and van Woerden [DvW22] to prove the IND-CPA security of a KEM purely based on ΔLIP (distinguish-LIP) between two lattices. When combined with further tightness improvements in [vW24] one can create a relatively efficient IND-CPA security proof based on this single assumption. The disadvantage is that there is still some tightness loss in the security reduction, and that the scheme needs to be adjusted slightly to instantiate this particular security proof. This is the approach that is taken in [BMM25] (without the optimizations of [vW24]).

A potentially more natural approach, following the ideas of [vW24] and matching our framework including Assumption 1, is to consider a two step approach in the security proof just as for LWE (or as for the NTRU+LWE combination). This would also require two assumptions instead of a single one. The first assumption is precisely the distinguish variant ΔLIP between the lattice $\mathcal{L}$ used in our scheme and a random lattice $\mathcal{L}'$ in the same genus of $\mathcal{L}$. As the

genus is currently the best-known efficiently computable invariant for LIP, the state-of-the-art cryptanalysis states that it hard to solve ΔLIP between $\mathcal{L}$ and $\mathcal{L}'$ in the same genus. This assumption allows to replace $\mathcal{L}$ with the random lattice $\mathcal{L}'$ in the security proof. The second assumption, matching Assumption 1 in this setting, would (informally) be that it is hard to solve a (decisional) syndrome decoding problem in a random lattice $\mathcal{L}'$ in the genus of $\mathcal{L}$. This second assumption is a very light assumption as [vW24] indicates that random lattices in such a fixed genus behave just like general random lattices, which are generally seen as the hardest instances for decoding problems.

Informally, we thus obtain a LIP-based Homomorphic Encryption scheme based on the following two assumptions. For more details on distinguish-LIP, the genus, and the precise notion of randomness, we refer to [DvW22,vW24].

Security Assumption 2 (Distinguish LIP (informal)) *Given $\mathcal{L}$ and a random lattice $\mathcal{L}' \leftarrow Genus(\mathcal{L})$ following the standard Haar measure on the space of lattices restricted to $Genus(\mathcal{L})$, it is hard to distinguish between $\mathcal{L}$ and $\mathcal{L}'$.*

Security Assumption 3 (Decoding in random lattice (informal)) *Let $\mathcal{L}$ be a lattice and consider a random lattice $\mathcal{L}' \leftarrow Genus(\mathcal{L})$. Then solving ΔSDP in $\mathcal{L}'$ is hard.*

To formalize these assumptions further one has to consider the distribution of the representation of $\mathcal{L}$ and $\mathcal{L}'$ (depending on some σ_{pk}, see [DvW22]), and to consider the precise error parameter σ_{enc}. The parametrization for σ_{pk} and σ_{enc} has been discussed in 5.

B Ciphertext compression and small-secret LWE

We have explained in Sections 3.2 and 3.3 that we require an ambient space $\mathcal{D}$ satisfying $q\mathcal{L}^* \subset \mathcal{D} \subset \mathcal{L}^*$ along with a basis D of $\mathcal{D}$ for which $\|\tilde{D}\|$ is sufficiently bounded. The first property is required for the homomorphic operations to work, while the second property is required to sample sufficiently small errors.

A basic choice is $\mathcal{D} := \mathcal{L}^*$ with $D := B^{-t}$ derived from the public basis B of $\mathcal{L}$. Then the ciphertext space equals $\mathcal{D}/q\mathcal{L}^* \cong \mathbb{Z}_q^m$ which can be rather large, both in the number of elements as well as its dimension. This gives larger ciphertexts of $\log_2(|\mathcal{D}/q\mathcal{L}^*|) = m\log_2(q)$ bits (or squared that in GSW), but also an additional overhead for both the additions and multiplications. In this section we consider a general compression technique that can be used to obtain a sparser ambient space, while maintaining the same bound on $\|\tilde{D}\|$ as the approach above.

B.1 Compressed ambient space

Let $C := B^{-t}$ be the basis of $\mathcal{L}^*$ dual to B, so that qC is a basis of $q\mathcal{L}^*$. Before presenting our generalized ambient space construction we make a further observation about LWE-based schemes, where we assume that the basis qC of $\mathcal{L}_q^{\perp}(A)$ has the block structure as in Eq. (3.3). Namely, one can observe

that the Gram-Schmidt profile $q(\|\tilde{\boldsymbol{c}}_1\|, \ldots, \|\tilde{\boldsymbol{c}}_m\|)$ of qC has a Z-shape: it starts with vectors of norm q, and ends with vectors of norm 1. Now, if one scales down these first q-vectors by q, they would obtain the basis D with vectors $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n, q\boldsymbol{c}_{n+1}, \ldots, q\boldsymbol{c}_m$. Now observe two things: firstly, the Gram-Schmidt profile of D is entirely flat, all Gram-Schmidt vectors have norm 1, and secondly, by construction $q\mathcal{L}^* \subset \mathcal{L}(D) \subset \mathcal{L}^*$, in fact D is a basis of the ambient space $\mathbb{Z}^m$. Furthermore, after size-reducing D one precisely obtains a basis of orthogonal unit vectors.

We will now generalize this construction. Generally, the Gram-Schmidt profile of C does not necessarily have such a Z-shape, so more work is needed. Note that so far we have not specified the value of q, thus we still have the freedom to do so. Let us therefore assume that q is of the form $q = p^k$ for some small integer $p \geq 2$. Instead of scaling vectors by q (or 1) only, we will allow any scaling of the form p^i for $0 \leq i \leq k$. This choice is made for simplicity, for example we could also allow q to be a product of several distinct small integers, or another value that works well for the specific basis we have as input.

Algorithm B.1 Ambient space construction

Input: A basis C of some lattice $\mathcal{L}^* \subset \mathbb{R}^m$, and moduli $p, q = p^k$ for some $k > 0$.
Output: A basis D such that $q\mathcal{L}^* \subset \mathcal{L}(D) \subset \mathcal{L}^*$ and $\max_i \|\tilde{\boldsymbol{d}}_i\| \leq \max_i \|\tilde{\boldsymbol{c}}_i\|$

1: **procedure** ConstructAmbientSpace(C, p, k)
2: Compute the Gram-Schmidt orthogonalisation $\tilde{\boldsymbol{c}}_1, \ldots, \tilde{\boldsymbol{c}}_m$ of C.
3: $q \leftarrow p^k$, $\ell_{\max} \leftarrow \max_i \|\tilde{\boldsymbol{c}}_i\|$
4: $j_i = \min\{k, \lfloor \log_p(\ell_{\max} / \|\tilde{\boldsymbol{c}}_i\|) \rfloor\}$ for all $i = 1, \ldots, m$
5: $\boldsymbol{d}_i \leftarrow p^{j_i} \cdot \boldsymbol{c}_i$ for all $i = 1, \ldots, m$
6: **return** $D = (\boldsymbol{d}_1, \ldots, \boldsymbol{d}_m)$
7: **end procedure**

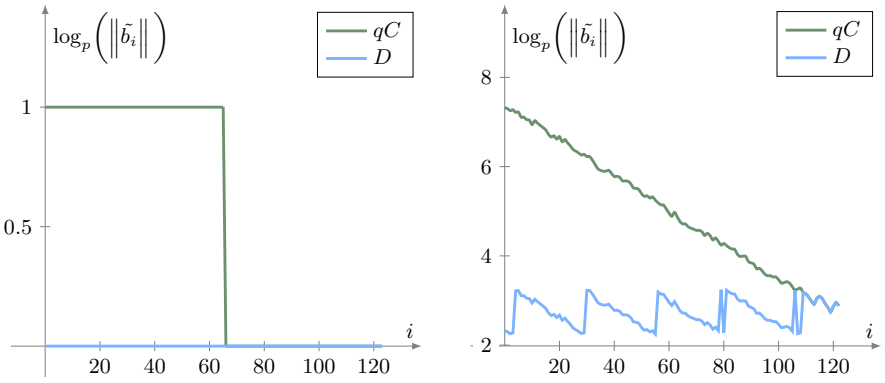
To construct such an ambient space $\mathcal{D}$, we will use the procedure given in Algorithm B.1 that generates an appropriate $\mathcal{D}$ such that $q\mathcal{L}^* \subset \mathcal{D} \subset \mathcal{L}^*$ along with a good basis D (see Figure B.1).

Lemma B.1. *Let C be a basis of some lattice $\mathcal{L}^* \subset \mathbb{R}^m$ and let $p > 0, q = p^k$ be some moduli for some $k > 0$. Let $\ell_{\min} = \min_i \|\tilde{\boldsymbol{c}}_i\|$ and $\ell_{\max} = \max_i \|\tilde{\boldsymbol{c}}_i\|$. Then Algorithm B.1 returns a basis D of an ambient space $\mathcal{D}$ such that $q\mathcal{L}^* \subset \mathcal{D} \subset \mathcal{L}^*$ and such that for all $i = 1, \ldots, m$ we have*

$$\min\{\ell_{max}/p, q\ell_{\min}\} \leq \|\tilde{\boldsymbol{d}}_i\| \leq \ell_{\max}.$$

Proof. Consider the variables used in Algorithm B.1. Note that each j_i satisfies $0 \leq j_i \leq k$, and thus $1 \leq p^{j_i} \leq q$ for all $i = 1, \ldots, m$. So by construction $q\mathcal{L}^* \subset \mathcal{L}(D) \subset \mathcal{L}^*$. For the bounds note that $\|\tilde{\boldsymbol{d}}_i\| = p^{j_i} \cdot \|\tilde{\boldsymbol{c}}_i\|$. The upper bound then follows immediately from the fact that $j_i \leq \log_p(\ell_{\max} / \|\tilde{\boldsymbol{c}}_i\|)$. For the lower bound we consider two cases, first suppose that $\log_p(\ell_{\max} / \|\tilde{\boldsymbol{c}}_i\|) < k + 1$, then

$j_i > \log_p(\ell_{\max}/\|\tilde{\boldsymbol{c}}_i\|) - 1$ and thus $\|\tilde{\boldsymbol{d}}_i\| > \ell_{\max}/p$. For the other case we have that $j_i = k$ and we obtain $\|\tilde{\boldsymbol{d}}_i\| = q \cdot \|\tilde{\boldsymbol{c}}_i\| \geq q\ell_{\min}$. □



(a) qC is the public basis of a random q-ary lattice, with $m = p = q = 123$.

(b) qC is an LLL-reduced basis of (a rotation of) $\mathbb{Z}^{123}$, with $p = 3$ and $q = p^5$. qC has been shifted up by $\log_p(q)$.

Fig. B.1: Gram-Schmidt profiles of the bad basis qC of $q\mathcal{L}^*$ and the good basis D of $\mathcal{D}$ as given by Algorithm B.1

B.2 Connections with short secret LWE

Short secret LWE, firstly introduced in [ACPS09], refers to a particular definition of LWE where the distribution used for $\boldsymbol{s}$ is also used for the *secret* coefficients $\boldsymbol{x}$. Recall that $\boldsymbol{b} = A \cdot \boldsymbol{x} + \boldsymbol{s}$ differs from the standard notation, in particular the "short secret" part refers to $\boldsymbol{x}$. We start from the main result.

Lemma B.2 ([ACPS09] (Informal)). *Let q be a prime power. Short-secret-*$\mathrm{LWE}_{n,m,\chi,q}$ *with m samples is equivalent to standard* $\mathrm{LWE}_{n,m+n,\chi,q}$ *with $m+n$ samples.*

So, up to a difference in the number of samples, the two problems are equivalent. We now want to show that using a short secret in LWE does not change our perspective, as short secret LWE is simply an explicit way of turning BDD into a SDP, so it only changes the "interpretation". At a high-level, short-secret-LWE can therefore be seen as a compression technique, where one is only interested in maintaining information about the small error $\boldsymbol{s}$, and not the particular "close vector" $A\boldsymbol{x} \bmod q$.

We recall the transformation from [ACPS09] that brings a regular LWE instance into a short-secret LWE instance. Consider some $A \in \mathbb{Z}_q^{m \times n}$ which is drawn uniformly, where m is sufficiently larger than n and q is a prime power.

Recall that LWE is a BDD decoding problem in the q-ary lattice $\mathcal{L}_q(A)$ with error $\boldsymbol{s} \in \mathbb{Z}^m$. Then, with overwhelming probability, and up to reordering of the rows, we can re-write the matrix A in the form

$$A = \begin{pmatrix} A_1 \\ A_2 \end{pmatrix} \in \mathbb{Z}_q^{m \times n},$$

where $A_2 \in \mathbb{Z}_q^{n \times n}$ is invertible modulo q. Notice that permuting rows does not change the geometry of the underlying lattice. Now we can multiply A by A_2^{-1} to obtain

$$A \cdot A_2^{-1} = \begin{pmatrix} A_1 A_2^{-1} \\ I_n \end{pmatrix}.$$

Note that this invertible transformation does not change the underlying lattice as it is simply a q-basis transformation.

Lemma B.3. *Let $B \in \mathbb{Z}_q^{n \times n}$ be invertible. Then $\mathcal{L}_q(A \cdot B) = \mathcal{L}_q(A)$.*

We can therefore re-state LWE from $\boldsymbol{b} := A\boldsymbol{x} + \boldsymbol{s}$ to

$$\begin{pmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{pmatrix} = \begin{pmatrix} A' \\ I_n \end{pmatrix} \cdot \boldsymbol{x}' + \begin{pmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \end{pmatrix} \in \mathbb{Z}_q^m,$$

where $A' = A_1 A_2^{-1}$ and $\boldsymbol{x}' = A_2 \boldsymbol{x}$. At this point, simply consider

$$L = \begin{pmatrix} I_{m-n} & -A' \\ 0 & I_n \end{pmatrix},$$

which is computable since it only uses public material, and multiply everything by L:

$$L \cdot \begin{pmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{pmatrix} = L \cdot \begin{pmatrix} A' \\ I_n \end{pmatrix} \cdot \boldsymbol{x}' + L \cdot \begin{pmatrix} \boldsymbol{s}_1 \\ \boldsymbol{s}_2 \end{pmatrix} \in \mathbb{Z}_q^m.$$

This leads to

$$\begin{pmatrix} \boldsymbol{b}_1 - A'\boldsymbol{b}_2 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \boldsymbol{x}' \end{pmatrix} + \begin{pmatrix} \boldsymbol{s}_1 - A'\boldsymbol{s}_2 \\ \boldsymbol{s}_2 \end{pmatrix} \in \mathbb{Z}_q^m.$$

And notice that now we have a new short secret LWE instance in the top part, as

$$\underbrace{\boldsymbol{b}_1 - A'\boldsymbol{b}_2}_{\text{new } \boldsymbol{b}} = \underbrace{A'}_{\text{new } A} \cdot \underbrace{-\boldsymbol{s}_2}_{\text{new } \boldsymbol{x}} + \underbrace{\boldsymbol{s}_1}_{\text{new } \boldsymbol{s}}$$

so the new secret key becomes $-\boldsymbol{s}_2$, and if we recover it we can also find a solution for the original instance with $\boldsymbol{x} = A_2^{-1} \cdot \boldsymbol{x}' = A_2^{-1} \cdot (\boldsymbol{s}_2 - \boldsymbol{b}_2)$. Clearly, the new variables are more compact even though they still contain all the information about $\boldsymbol{s}$ from before, besides the (typically larger) vector $\boldsymbol{x}$. Throughout this process the lattice $\mathcal{L}_q(A)$ in which the original LWE/BDD instance lived is unchanged, and can be represented by A' because

$$\mathcal{L}_q(A) = \begin{pmatrix} A' \\ I_n \end{pmatrix} \cdot \mathbb{Z}^n + q\mathbb{Z}^m.$$

More precisely, $\begin{pmatrix} A' \\ I_n \end{pmatrix}$ and $\begin{pmatrix} qI_{m-n} \\ 0 \end{pmatrix}$ together form a basis of $\mathcal{L}_q(\mathcal{L})$.

We thus merely *explicitly* changed the representation from a BDD target in $\mathbb{Z}^m/q\mathbb{Z}^m$ to a syndrome in $\mathbb{Z}^m/\mathcal{L}_q(A)$. We also want to emphasize that the operations one can do with short-secret-LWE can also be achieved with general-LWE, although this might require some rewriting, is less obvious from pure algebraic equations, and is less efficient by the lack of compression.

In our more general setting we do not always have $\mathcal{D}/q\mathcal{L}^* \cong \mathbb{Z}_q^{m-n}$, but e.g. in our construction we do get $\mathcal{D}/q\mathcal{L}^* \cong \prod_{j=1}^{k} \mathbb{Z}_{p^{i_j}}^{n_j}$ for some distinct $i_1 > i_2 > \ldots > i_k$ and $n_j > 0$. With the appropriate choice of basis D of $\mathcal{D}$ and C of $\mathcal{L}^*$, using an HNF computation, we can then write a (relative) basis of $q\mathcal{L}^*$ of the upper block-diagonal form:

$$D = C \cdot \begin{bmatrix} p^{i_1} I_{n_1} & A_{12} & \cdots & A_{1k} \\ 0 & p^{i_2} I_{n_2} & \cdots & A_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p^{i_k} I_{n_k} \end{bmatrix}$$

Note that without loss of generality one can write A_{jl} with coefficients in $\mathbb{Z}_{p^{i_j}}$. This is the compressed form of a basis of $q\mathcal{L}^*$ w.r.t. a basis of the ambient space $\mathcal{D}$. Similarly, any BDD target $\boldsymbol{t} \in \mathcal{D}$ can uniquely be written as a "compressed" syndrome in the coefficient representation $\boldsymbol{t} \bmod q\mathcal{L}^* = C \cdot [\boldsymbol{b}_1; ...; \boldsymbol{b}_k]$ with $\boldsymbol{b}_j$ in $\mathbb{Z}_{p^{i_j}}^{n_j}$ (e.g. for LWE one has $\mathbb{Z}_{p^{i_k}} = \{0\}$ giving the unique choice $b_k = 0$).

B.3 Connections and differences with structural lattice reduction

Our goal and method for creating an ambient space $\mathcal{D} \supset q\mathcal{L}^*$ along with a nearly orthogonal basis (that we will call D) satisfying $\mathcal{D}/q\mathcal{L}^* \cong G$ for some finite abelian group G is similar to the work [GINX16]. They introduce a method called *structural lattice reduction*, which given any sufficiently large group G (and some minor additional constraints), creates an ambient space as above for this specific group.

In particular, they define a generalization of LWE to *Group* LWE (GLWE); where LWE is a decoding problem in a lattice $\mathcal{L} \subset \mathbb{Z}^m$ and an ambient space $\mathbb{Z}^m$ such that $\mathbb{Z}^m/\mathcal{L} \cong \mathbb{Z}_q^n$, the GLWE problem considers the decoding problem in a lattice $\mathcal{L}$ such that $\mathbb{Z}^m/\mathcal{L} \cong G$ for any finite abelian group G. Furthermore, given a large enough group G, it is shown that for any lattice $\mathcal{L}$ (not necessarily integer) one can construct an ambient space $\mathcal{D}$ with $\mathcal{D}/\mathcal{L} \cong G$ along with a nearly orthogonal basis of $\mathcal{D}$.

In our work we come across similar problems and generalizations. For one, we also need to construct an appropriate ambient space and construct one in a similar way as in [GINX16], but we allow G to be determined by the procedure, only restricting that G must be a subgroup of $\mathbb{Z}_q^m$. Conversely, they fix some group G and adapt the ambient space to it – although we stress that one could equally well use structural lattice reduction within our framework if one aims for a certain group structure.

We have shown that G being a subgroup of $\mathbb{Z}_q^m$ – or, following our notation, $q\mathcal{L}^* \subset \mathcal{D} \subset \mathcal{L}^*$ for $\mathcal{D}/q\mathcal{L}^* \cong G$ – is sufficient to make the homomorphic operations work. This extra freedom allows our method to be simpler and a bit more efficient, in particular, a direct comparison shows that our group can be up to a factor $O(m^m)$ smaller, and the modulus q can be up to a factor $O(m)$ smaller, for the same bound on $\|\tilde{D}\|$. Fixing a particular group G however, could come with some additional benefits and our framework could as well be instantiated using the structural lattice reduction procedure.

Additionally, the generalized modulus switching that we considered is related to the *group switching* method in [GINX16]. Again however, we do not need the full generalization and present a simpler and geometrical method that is sufficient for our construction.

Lastly, one main difference between these works is that our goal is to present and explain HE from a *lattice perspective*, while one could view [GINX16] as presenting it from a group structure or algebraic perspective. Furthermore, while we could instantiate the HE schemes we obtain with (the lattices occurring in) GLWE, our framework is more general and allows it to be instantiated with *any lattice* that contains a sufficiently short secret vector.

C Proofs of lemmas

C.1 Proof of Lemma 4.2

Proof. The decryption error is computed as $\langle \boldsymbol{e}, \boldsymbol{s} \rangle$, therefore

$$\langle \boldsymbol{e}, \boldsymbol{s} \rangle \equiv \langle t \cdot \boldsymbol{r} + D \cdot (\mu, 0, \ldots, 0), \boldsymbol{s} \rangle \bmod \mathcal{D}/q\mathcal{L}^*$$

Notice that $\boldsymbol{s} \in \mathcal{L}^*$ can be considered as a constant, and we have $|\langle \boldsymbol{e}, \boldsymbol{s} \rangle| \leq \|\boldsymbol{e}\| \cdot \|\boldsymbol{s}\|$, so we will focus on $\boldsymbol{e}$. The latter is composed of two terms:

- $(t \cdot \boldsymbol{r})$: observe that if $\boldsymbol{r} \sim \mathcal{G}_{\mathcal{L},\sigma_{\text{enc}}}$ then $\|\boldsymbol{r}\|_2$ is on average bounded by $\sqrt{m} \cdot \sigma_{\text{enc}}$, so this term is on average $t \cdot \sqrt{m} \cdot \sigma_{\text{enc}}$
- $(D \cdot (\mu, 0, \ldots, 0))$: this term can be rewritten as $\boldsymbol{d}_1 \cdot \mu$. Observe that, in order to efficiently sample (Lemma 2.1) from D, we need σ_{enc} to be such that $\sigma_{\text{enc}} > \|\boldsymbol{d}_1\| = \|\boldsymbol{d}_1^*\|$, therefore this can be bounded as $\sigma_{\text{enc}} \cdot (t/2)$ by assuming that, on average, $\mu = t/2$.

Notice though that the euclidean norm is not linear (i.e., $\mathbb{E}[\|\boldsymbol{a}+\boldsymbol{b}\|_2] \neq \mathbb{E}[\|\boldsymbol{a}\|_2] + \mathbb{E}[\|\boldsymbol{b}\|_2]$), however observe that $\mathbb{E}[\|\boldsymbol{a}+\boldsymbol{b}\|_2] < \mathbb{E}[\|\boldsymbol{a}\|_2] + \mathbb{E}[\|\boldsymbol{b}\|_2]$, so we can sum them to have a (conservative) average result. The result follows by refactoring $(t \cdot \sqrt{m} \cdot \sigma_{\text{enc}}) + (t \cdot \sigma_{\text{enc}})/2$. □

C.2 Proof of Corollary 4.1

Proof. By considering the discrete Gaussian $\mathcal{G}_{\mathcal{L},\sigma_{\text{enc}}}$ as a subgaussian, we can use the following tail bound:

$$\Pr[\|X\| \geq \sigma_{\text{enc}}\sqrt{m}(1+\varepsilon)] \leq e^{-\varepsilon^2 m/2}, \quad 0 < \varepsilon < 1.$$

We set $e^{-\varepsilon^2 m/2} = 2^{-m}$, with $m \in \mathbb{Z}^+$, to obtain $\varepsilon = \sqrt{2\ln 2}$. Therefore, by extending the proof of Lemma 4.2, we can move from an average value of

$$(t \cdot \sigma_{\text{enc}})(\sqrt{m} + 1/2) \cdot \|\boldsymbol{s}\|_2,$$

to, with overwhelming probability 2^m, the following upper bound:

$$(t \cdot \sigma_{\text{enc}})(\sqrt{m} + \sqrt{2\ln 2} + 1/2) \cdot \|\boldsymbol{s}\|_2.$$

□

Both estimates can be improved by up to a factor $\sqrt{m}$ when σ_{enc} is above the smoothing parameter of $\mathcal{D}$ (for a sufficiently small ϵ). In this case we can argue that $\langle \boldsymbol{r}, \boldsymbol{s} \rangle$ is negligible close to a 1-dimensional Discrete Gaussian on $\mathbb{Z}$ with parameter $\sigma_{\text{enc}} \cdot \|\boldsymbol{s}\|$ from which a simple tailbound follows.

C.3 Proof of Lemma 4.3

Proof. For correctness, observe that by construction that $\boldsymbol{\delta} \equiv 1 \bmod t$ and $q_2/q_1 \equiv 1 \bmod t$, so we obtain:

$$[\langle \boldsymbol{c}', \boldsymbol{s} \rangle]_{q_2} \equiv \frac{q_2}{q_1}[\langle \boldsymbol{c} + \boldsymbol{\delta}, \boldsymbol{s} \rangle]_{q_1} \equiv [\langle \boldsymbol{c}, \boldsymbol{s} \rangle + \langle \boldsymbol{\delta}, \boldsymbol{s} \rangle]_{q_1} \equiv [\langle \boldsymbol{c}, \boldsymbol{s} \rangle]_{q_1} \bmod t$$

To see that the decryption error decreases, let us consider the decryption of an output of the $\mathcal{L}$-BGV.GeneralModSwitch function. In particular, we evaluate

$$\frac{q_2}{q_1} \langle (\boldsymbol{c} - \boldsymbol{\delta}), \boldsymbol{s} \rangle \equiv \frac{q_2}{q_1}(\langle \boldsymbol{e}, \boldsymbol{s} \rangle - \langle \boldsymbol{\delta}, \boldsymbol{s} \rangle) \bmod q_1$$

From the approximate decoder on $\mathcal{D}_2$ we have that $\|\boldsymbol{\delta}\| \leq \frac{q_1}{q_2} \cdot \rho$. Notice that by using Babai's decoding algorithm we can bound $\|\boldsymbol{\delta}\|$ as $\|\boldsymbol{\delta}\|^2 \leq q_1/(4q_2) \sum_{i=1}^m \|\tilde{\boldsymbol{d}}_i\|^2$, where $\boldsymbol{d}_i$ is the i-th basis vector for $\mathcal{D}_2$. We can therefore always assume that $\rho \leq \frac{1}{2}\sqrt{\sum_{i=1}^m \|\tilde{\boldsymbol{d}}_i\|^2}$, and we bound $q_2/q_1 \cdot |\langle \boldsymbol{e}, \boldsymbol{s} \rangle - \langle \boldsymbol{\delta}, \boldsymbol{s} \rangle|$ as follows:

$$\begin{aligned}[|\langle \boldsymbol{c}', \boldsymbol{s} \rangle]_{q_1}| = \frac{q_2}{q_1}|\langle \boldsymbol{e}, \boldsymbol{s} \rangle - \langle \boldsymbol{\delta}, \boldsymbol{s} \rangle| &\leq \frac{q_2}{q_1}|\langle \boldsymbol{e}, \boldsymbol{s} \rangle| + \frac{q_2}{q_1}\|\boldsymbol{\delta}\| \cdot \|\boldsymbol{s}\| \\ &\leq \frac{q_2}{q_1}|\langle \boldsymbol{e}, \boldsymbol{s} \rangle| + \rho \cdot \|\boldsymbol{s}\| = \frac{q_2}{q_1}[|\langle \boldsymbol{c}, \boldsymbol{s} \rangle]_{q_1}| + \rho\|\boldsymbol{s}\|\end{aligned}$$

so if $\rho\|\boldsymbol{s}\|$ is small compared to $|\langle \boldsymbol{e}, \boldsymbol{s} \rangle|$, the decryption error decreases as it gets scaled by $q_2/q_1 < 1$. □

D More details on key switching

D.1 Original BGV key switching

From a certain point of view, key switching can be seen as a bootstrapping operation but without further removal of the decryption noise, allowing for a

much simpler procedure. Let us forget for a moment about the bit decomposition and analyze the key switching operation from an high-level perspective.

The final matrix generated by BGV.KeySwitchGen (Eq. (2.6)) used to switch the key is, considering the addition over the first column, equal to $\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2} = A_{\text{ks}} + \boldsymbol{x}_1 \in \mathbb{Z}_q^{n_1 \times n_2}$. Since A_{ks} is uniform-looking by LWE, the secret $\boldsymbol{x}_1$ is hard to be recovered from $\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}$. The abstract idea is that this new key is an encryption of the old secret key $\boldsymbol{x}_1$ under $\boldsymbol{x}_2$, and because $\boldsymbol{c}$ is public we can compute an encryption of $[\langle \boldsymbol{c}, \boldsymbol{x}_1 \rangle]_q$ under $\boldsymbol{x}_2$ using only homomorphic scalings and additions.

Indeed, the key switching can be performed by multiplying the (transposed) matrix $\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}$ with the ciphertext $\boldsymbol{c}$. Intuitively, the presence of $\boldsymbol{x}_1$ in the first column of $\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}$ decrypts the message contained in the first coordinate of $\boldsymbol{c}$, but at the same time this message is re-encrypted by adding to the latter the new encryptions of zero under $\boldsymbol{x}_2$:

$$\text{keySwitch}(\boldsymbol{c}, \tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}) := \boldsymbol{c}' := (\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2})^t \cdot \boldsymbol{c} = ([\langle \boldsymbol{c}, \boldsymbol{x}_1 \rangle]_q, 0, \ldots, 0) + (A_{\text{ks}})^t \cdot \boldsymbol{c} \in \mathbb{Z}_q^{n_2}.$$

We observe that applying this approach as is might not produce a valid ciphertext, as $A_{\text{ks}}^t \cdot \boldsymbol{c}$ is a sum of ciphertexts each scaled by c_i. The problem is that $\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}^t \cdot \boldsymbol{c} \in \mathbb{Z}_q^{n_2}$ produces a new encryption of the old message by adding n_1 fresh ciphertexts to it, each scaled by c_i. As each c_i can be as large as $q/2$ the result will contain a significant amount of noise that easily exceeds the $q/2$ threshold. There are different approaches to tackle this issue: BV [BV11], GHS and HYBRID [GHS12]. In this paper we stick to BV: it consists in decomposing the ciphertext $\boldsymbol{c}$ as a longer binary vector using $\texttt{Bits}(\boldsymbol{c})$ and decomposing the secret key $\boldsymbol{x}_1$ as $\texttt{Pow}(\boldsymbol{x}_1)$; this implies that A_{ks} should be extended to $n_1 \log(q)$ fresh encryptions of 0. The noise contained in the resulting ciphertext will be smaller, since the encryptions of 0 contained in A_{ks} will be multiplied with a vector of smaller magnitude, that is $\texttt{Bits}(\boldsymbol{c})$. the partial decryption will still be valid since $\langle \texttt{Bits}(\boldsymbol{c}), \texttt{Pow}(\boldsymbol{x}_1) \rangle = \langle \boldsymbol{c}, \boldsymbol{x}_1 \rangle$. On the other hand, the GHS approach simply adds the secret to A_{ks} as:

$$\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2} = A_{\text{ks}} + P\boldsymbol{s} \in \mathbb{Z}_{Pq}^{n_1 \times n_2}$$

and the ciphertext is scaled by P before the multiplication with $\tau_{\boldsymbol{x}_1 \to \boldsymbol{x}_2}$. Then, after the multiplication, a modulus switching operation is performed to remove the added P modulus.

The advantage is that the decomposition phase – which can be expensive – is skipped, but at the cost of increasing the size of the intermediate ciphertext and the modulus of the underlying RLWE problem. In practice, typical implementations use an hybrid approach, where the ciphertext is decomposed with respect to some radix larger than 2 and the modulus is slightly increased, with $P < q$.

D.2 Key switching in Lattice-BGV

Given two public lattices $\mathcal{L}_1, \mathcal{L}_2$ with secret short vectors $\boldsymbol{s}_1, \boldsymbol{s}_2$, dimensions $m_1, m_2 \in \mathbb{N}$ and ambient spaces $\mathcal{D}_1, \mathcal{D}_2$ for $q\mathcal{L}_1^*, q\mathcal{L}_2^*$ respectively, the goal of the

key switching procedure is to transform a ciphertext $\boldsymbol{c} \in \mathcal{D}_1/q\mathcal{L}_1^*$, with respect to the secret key $\boldsymbol{s}_1$, to a ciphertext in $\mathcal{D}_2/q\mathcal{L}_2^*$ containing the same message with respect to the secret key $\boldsymbol{s}_2$. In particular, this allows us to switch from a larger dimensional lattice, say $\mathcal{L}_1 = \mathcal{L} \otimes \mathcal{L}$ obtained after a multiplication, back to the starting dimension. As the key switching procedure can essentially be generalized to any additive homomorphic scheme, without using any specific lattice properties, we will be rather brief about it. We assume without loss of generality that the bases $C = B^{-t}$ and D_1, D_2 of $\mathcal{L}^*$ and $\mathcal{D}_1, \mathcal{D}_2$ respectively differ from each other by an HNF form such that D_i^{-1} gives an explicit map from $\mathcal{D}_i/q\mathcal{L}^*$ to some diagonal group $\prod_{i=1}^{m} \mathbb{Z}/p^{j_i}\mathbb{Z}$. For simplicity of notation we assume that the `Bits` function on $\mathbb{Z}/p^{j_i}\mathbb{Z}$ is extended to a vector of length $\log_p(q)$ by appending zeros.

Recall from the LWE case that the switching key $\tau_{\boldsymbol{s}_1 \to \boldsymbol{s}_2}$ essentially consists of encryptions $D_2(\boldsymbol{c}_i^{(0)} + ((D_1^t \boldsymbol{s}_1)_i, 0, \ldots, 0)^t) \in \mathcal{D}_2/q\mathcal{L}_2^*$ of each coefficient of $D_1^t \boldsymbol{s}_1$ under $\boldsymbol{s}_2$ (ignoring the modulo t encoding), where $D_2\boldsymbol{c}_i^{(0)}$ is an encryption of 0. Given the public ciphertext $\boldsymbol{c} = D_1(c_1, \ldots, c_{m_1})^t \in \mathcal{D}_1/q\mathcal{L}_1^*$ we simply scale and sum over these encryptions to get

$$\begin{aligned}\boldsymbol{c}' &:= D_2 \cdot \sum_{i=1}^{m_1} c_i \cdot \left(\boldsymbol{c}_i^{(0)} + ((D_1^t \boldsymbol{s}_1)_i, 0, \ldots, 0)^t\right) \\ &= \left(\sum_{i=1}^{m_1} c_i \cdot D_2 \boldsymbol{c}_i^{(0)}\right) + D_2 \cdot ([\langle \boldsymbol{c}, \boldsymbol{s}_1 \rangle]_q, 0, \ldots, 0).\end{aligned}$$

We thus obtain a scaled sum of encryptions of 0 which masks the message $[\langle \boldsymbol{c}, \boldsymbol{s}_1 \rangle]_q$. Assuming $\boldsymbol{c}$ is a valid ciphertext we have $[\langle \boldsymbol{c}, \boldsymbol{s}_1 \rangle]_q = te + \mu$ for the message μ and some *small* decryption noise e. Assuming the c_i are small, the errors coming from the sum of encryptions of 0, and the error $D_2 \cdot (te, 0, \ldots, 0)^t$, should be small enough for the new ciphertext $\boldsymbol{c}'$ to correctly decrypt to μ under $\boldsymbol{s}_2$. For the c_i to be small enough we can use the BV binary decomposition trick.

Remark 3. The encryptions of 0 used to construct the switching key in the LWE case are generated using the public key generation procedure. This must be seen as an artifact of the specific key generation procedure under LWE. More generally, this could be interpreted as a *secret key encryption* procedure, i.e., the valid encryptions are constructed with knowledge of the secret key. This allows to make the error smaller in the direction of $\boldsymbol{s}_2$, while making it uniform in the direction orthogonal to it to compensate for any concrete security loss. One could easily generalize this to our lattice setting, or alternatively just use the *public key encryption* procedure (albeit leading to a larger decryption error).

Summarized we obtain the following key switching procedure.

1. Sample $m_1 \cdot \lceil \log(q) \rceil$ encryptions $D_2 \cdot \boldsymbol{c}_{(i,j)}^{(0)} \in \mathcal{D}_2/q\mathcal{L}_2^*$ of zero indexed by the pair (i, j) for $i = 1, \ldots, m_1$ and $j = 1, \ldots, \lceil \log(q) \rceil$ via a public or secret key encryption procedure.

2. Compute the switching key $\tau_{\boldsymbol{s}_1 \to \boldsymbol{s}_2} = (\boldsymbol{c}^{(0)}_{(i,j)} + (\texttt{Pow}((D_1^t \boldsymbol{s}_1)_i)_j, 0, \ldots, 0)^t)_{(i,j)}$.
3. To switch from $\boldsymbol{c}$ to $\boldsymbol{c}'$ given $\tau := \tau_{\boldsymbol{s}_1 \to \boldsymbol{s}_2}$ compute

$$\boldsymbol{c}' = D_2 \cdot \sum_{i=1}^{m_1} \sum_{j=1}^{\lceil \log(q) \rceil} (\texttt{Bits}((D_1^{-t} \boldsymbol{c})_i)_j \cdot \tau_{(i,j)}.$$

E Connections with [BMM25]

In a recent concurrent work, Branco, Malavolta and Maradni [BMM25] propose an FHE scheme based on the hardness of LIP. While this does not coincide with the main goal of our work, we do want to shortly compare it to a LIP-based instantiation of our lattice framework. We will mostly focus on the main differences.

Firstly, the work of [BMM25] has a large focus on building an FHE scheme along with a security reduction to (distinguish-)LIP with a similar strategy as used in [DvW22]. As our main goal was to give a general lattice framework for (F)HE and its operations, we presented a security proof in the general setting, and left the precise details of a security proof for a LIP-based instantiation out of the scope of our work. We did sketch how to achieve such a reduction in Section A.2.

Secondly, [BMM25] presents a private-key encryption scheme, contrary to our public-key encryption scheme. This is vital for the correctness of their homomorphic multiplications. Following the strategy of [Rot11] one could turn their private-key FHE scheme into a public-key, at a rather large increase in the error. Especially in the setting of [BMM25] where the secret unit vector $\boldsymbol{s}$ in a rotation of $\mathbb{Z}^m$ is rather large, there does not seem to be room for such an increase. Conversely, both our GSW and BGV-like multiplication approaches naturally work in a public-key setting, without resorting to extra constructions à la [Rot11].

Thirdly, the schemes we present naturally allow for the evaluation of circuits of multiplicative depth more than one and we give an explicit connection between the shortness of the secret trapdoor vector and the depth that can be achieved before bootstrapping is required. Our approach shows that an instantiation with a rotation of $\mathbb{Z}^m$, while allowing for even a single multiplication, seems to be difficult to achieve securely from a geometric perspective. Therefore, in [BMM25] only branching programs are considered, where one of the two inputs of the homomorphic multiplication is a fresh ciphertext, which only works with secret-key encryptions.

Lastly, while [BMM25] claims an *FHE* scheme our work claims an *HE* scheme. Concretely, however, both works only present an HE scheme for bounded depth circuits, and argue that from there bootstrapping is possible by standard techniques. To conclude, we mostly consider the goals and execution of our work and [BMM25] as orthogonal.