Karlsruhe Institute of Technology

KASTEL

The Lattice Incompatibility Problem: Standard-Model CCA2 Security from Lattice Isomorphisms and K-Repetition

Master's Thesis of

Philip Scherer

At the KIT Department of Informatics
KASTEL – Institute of Information Security and Dependability

First examiner:	Prof. Dr. Jörn Müller-Quade
Second examiner:	Prof. Dr. Thorsten Strufe
First advisor:	M.Sc. Laurin Benz
Second advisor:	Dr. Marcel Tiepelt

30. April 2025 – 30. October 2025

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

Abstract

The hardness of the lattice isomorphism problem (LIP) is a new lattice-based assumption used in asymmetric cryptography. However, there is yet to be a standard-model IND-CCA2-secure public-key encryption scheme based on LIP. One promising candidate for such a scheme is the combination of an existing LIP-based IND-CPA-secure key encapsulation mechanism (KEM) by Ducas and van Woerden with the k-repetition framework by Rosen and Segev to construct an IND-CCA2-secure KEM.

In this thesis, we provide strong evidence for our claim that this combination cannot be securely realized due to inherent qualities of LIP that prevent correlation across multiple ciphertexts of the KEM with independent public keys. We dub the sum of these qualities the *Lattice Incompatibility Problem*. Our analysis is extensive and provides insight into several fundamental properties of average-case LIP instances and their coefficient vectors. Additionally, we establish formal requirements for any other combination of an IND-CPA-secure asymmetric encryption scheme with k-repetition and demonstrate that this combination cannot be black-box for arbitrary IND-CPA-secure schemes. As part of our empirical evaluations, we also present a description, proof of correctness, and implementation of a sampling algorithm for the discrete Gaussian distribution on arbitrary quadratic forms that avoids taking the Cholesky decomposition for improved numerical stability.

Zusammenfassung

Die Schwierigkeit des Gitterisomorphismusproblems (LIP) wird als neue gitterbasierte Annahme zur Konstruktion von asymmetrischer Kryptographie genutzt. Jedoch gibt es bis jetzt noch kein gegen adaptive aktive Angreifer sicheres asymmetrisches Verschlüsselungsverfahren, das auf der Schwierigkeit von LIP beruht. Ein möglicher Kandidat für ein solches Verfahren entsteht durch die Kombination aus einem existierenden LIP-basierten Schlüsselenkapsulierungsmechanismus (KEM) von Ducas und Woerden zusammen mit dem k-Wiederholungs-Rahmenwerk von Rosen und Segev, die zu einem entsprechend gegen solche Angreifer sicherem KEM führen könnte.

In dieser Arbeit präsentieren wir starke Argumente dafür, dass diese Kombination durch inhärente Merkmale von LIP, die eine Korrelation über mehrere Chiffrate des KEMs mit unabhängigen öffentlichen Schlüsseln verhindern, nicht auf sichere Art realisierbar ist. Wir bezeichnen die Summe dieser Eigenschaften als das *Gitterinkompatibilitätsproblem*. Unsere Analyse ist umfänglich und zeigt einige grundlegende Eigenschaften durchschnittlicher LIP-Instanzen und ihrer Koeffizientenvektoren auf. Zusätzlich geben wir formale Anforderungen an, die für jede Kombination eines gegen passive Angreifer sicheren asymmetrischen Verschlüsselungsverfahrens mit der k-Wiederholung erforderlich sind. Wir demonstrieren zudem, dass diese Kombination nicht generisch für beliebige Verfahren umgesetzt werden kann. Weiter stellen wir als Teil unserer empirischen Evaluationen einen Algorithmus samt Beschreibung, Korrektheitsbeweis und Implementierung vor, der Stichproben der diskreten Gauß-Verteilung über beliebigen quadratischen Formen zieht, ohne dabei die Cholesky-Zerlegung der Form zu ermitteln, was zu verbesserter numerischer Stabilität führt.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

The field of asymmetric cryptography is currently undergoing a transition: Due to the threat posed to classical hardness assumptions like the Diffie-Hellman or RSA assumptions by quantum computers via Shor's algorithm [87], there is a need to develop asymmetric cryptographic schemes based on post-quantum-secure assumptions. Lattice problems are strong candidates for these, with several assumptions and schemes having been standardized or undergoing standardization. The hardness of the *lattice isomorphism problem* (LIP) in particular is a recent addition to the growing list of lattice-cryptographic assumptions; it comes with two main advantages: First, it has a worst-to-average-case reduction by Ducas and van Woerden [33] that increases confidence in the difficulty of random instances. Second, its structure allows the use of well-studied lattices with desirable (or "remarkable") properties like efficient decoding algorithms with large decoding radii. Contrast this with assumptions like Learning with Errors (LWE), which require sampling random lattices of a specific class and thus cannot benefit from these properties [33].

LIP has been used to construct multiple different asymmetric cryptographic primitives in the literature, including an IND-CPA-secure key encapsulation mechanism (KEM) by Ducas and van Woerden [33]. This KEM has been adopted into a practical IND-CCA2-secure KEM in the random oracle model (ROM) by de Castro Biage et al. [29] using the standard Fujisaki-Okamoto transformation [36]. However, there is yet to be a *standard-model* IND-CCA2-secure asymmetric encryption primitive built on LIP. In particular, the standard model of cryptography does not offer a black-box transformation from IND-CPA security to IND-CCA security like the ROM does. Given that random oracles can be proven to be uninstantiable by efficient algorithms [65], it is of interest to show that LIP implies IND-CCA2-secure asymmetric encryption in the standard model. One possible approach to fill this gap in the state of the art is to combine Ducas and van Woerden's [33] LIP-KEM with the k-repetition framework as used by Döttling et al. [31]. This framework allows one to construct an IND-CCA2-secure scheme from an IND-CPA-secure one by correlating multiple instances of the base scheme (in our case, the LIP-KEM), with each instance using an independent keypair. This must be done in such a way that knowing just one of the secret keys allows decrypting and verifying the validity of the entire ciphertext.

Unfortunately, we discover that there is likely no way to correlate instances of the LIP-KEM in the required manner due to the nature of the LIP-KEM's public keys — these are differently rotated, reparametrized versions of some base lattice. Every one of our discussed attempts to correlate parts of a ciphertext across instances of the LIP-KEM is thwarted by the insurmountable differences between these transformed lattices: The different rotations cause the lattices to misalign with each other, while the parametrizations scale the lattice

coordinates in unpredictable and loosely bounded ways. It is therefore precisely the elements of LIP that make the problem hard for an adversary that also make correlations across multiple instances of it infeasible. Since these difficulties are so intricately tied to the nature of LIP itself, we refer to them as the *Lattice Incompatibility Problem*.

In this work, we extensively investigate the possible avenues for combining the LIP-KEM with the k-repetition framework by correlating LIP-KEM instances. Our investigation shows that none of the approaches we discuss are viable — they either fail to be efficient, fail to provide provable security, or, in one case, fail to be compatible with LIP at all. As part of this investigation, we establish formal requirements for the use of k-repetition with IND-CPA-secure primitives and show that there is no way to apply k-repetition to arbitrary public key encryption schemes or KEMs in a black-box manner in the standard model. We also highlight some of the limitations of LIP as a cryptographic tool, namely the lack of control over the public key lattices and the need to rely on statistical arguments for security. Our heuristic analysis of different variants of the LIP-KEM provides deeper insights into the behavior of its public keys and ciphertexts. In addition, we systematize the state of the art on discrete Gaussian sampling algorithms for arbitrary lattices, formalizing a previously-unpublished technique to improve numerical stability and adapt the final algorithm to the native quadratic form representation of lattices used in LIP. This re-evaluation also resulted in our discovery of an error in a lower bound on a parameter of the algorithm; said error had become pervasive in the literature up to that point. We implement the resulting algorithm as an empirical reference for our investigation of the LIP-KEM.

In summary, our main contributions are:

- the establishment of formal requirements for combining k-repetition with asymmetric encryption primitives,
- an extensive argument for the impossibility of a k-repetition LIP-KEM in the standard model of cryptography,
- an in-depth analysis of the properties of LIP-KEM public keys and ciphertexts,
- and a formal description and implementation of a sampling algorithm for the discrete Gaussian distribution on quadratic forms.

Related Work: LIP has been used to construct several cryptographic primitives and protocols in the literature. Ducas and van Woerden [33] propose a zero-knowledge proof, the IND-CPA-secure LIP-KEM used in this work, and an EUF-CMA-secure signature scheme. This signature scheme was later adapted into the practical signature scheme HAWK by Ducas et al. [34], which is a NIST standardization candidate at the time of writing [3]. de Castro Biage et al. [29] give a practical instantiation of the LIP-KEM using the Barnes-Wall lattices. In a similar vein, Ackermann, Roux-Langlois, and Wallet [1] construct a one-bit IND-CPA-secure PKE from LIP using almost the same structure as the LIP-KEM: Their scheme is based on an adversary not being able to distinguish a LIP-KEM ciphertext from uniform randomness over the ciphertext space. Ackermann et al.'s work is a generalization of a previous one-bit PKE by Bennett et al. [15] which is not directly based on LIP, but

uses closely related hardness assumptions concerning recognizing rotations of $\mathbb{Z}^n$. Leporati, Rovida, and van Woerden [51] propose LIP as a candidate assumption for hiding short vectors in a lattice with the goal of generalizing the homomorphic encryption schemes by Brakerski, Gentry, and Vaikuntanathan [20] and Gentry, Sahai, and Waters [39]. Branco, Malavolta, and Maradni [22], on the other hand, directly construct a new fully homomorphic encryption scheme from a circular variant of LIP.

Benčina et al. [11] introduce the lattice isomorphism group action (LIGA), which enables the use of LIP to instantiate generic group-action constructions. Jiang et al. [45] and Luo et al. [53] use the LIGA to instantiate commitment schemes. In addition, Khuc et al. [48] defined a linkable ring signature based on the LIGA. However, their scheme was shown to have a vulnerability by Budroni, Chi-Domínguez, and Franch [23], who proved that the LIGA does not satisfy the weak unpredictability or weak pseudorandomness properties for group actions.

Moving to the topic of k-repetition, Rosen and Segev's [84] framework has been instantiated with many different lossy and correlation-secure TDOWFs to create IND-CCA2-secure encryption schemes in the standard model across the literature. It has been combined with TDOWFs based on SVP in ideal lattices [89], LWE [47, 58, 74, 76], and the decisional and computational Diffie-Hellman assumptions [37, 76]. Some of the lossy TDOWFs above are only slightly lossy; their authors make use of a work by Mol and Yilek [66], who show that even these TDOWFs are correlation-secure. Freeman et al. [35] build additional lossy TDOWFs from the quadratic and composite residuosity assumptions as well as the d-linear assumption and create correlation-secure TDOWFs from syndrome decoding. Hemenway and Ostrovsky [43] show that homomorphic encryption schemes with a cyclic plaintext space can also be used to achieve IND-CCA2-secure encryption using the k-repetition framework. Döttling et al. [31] adapt k-repetition for use with PKEs and combine it with the McEliece assumption to create an IND-CCA2-secure PKE. Persichetti [77] gives an alternate definition of Döttling et al.'s scheme that is closer to the original k-repetition framework.

On the subject of sampling from discrete Gaussian distributions, the literature offers both algorithms that work on generic lattices as well as techniques that are specialized to certain classes of lattices or individual lattices. Klein's [49] randomized CVP solver was adapted into an algorithm with an output distribution statistically close to the discrete Gaussian distribution on arbitrary lattices by Gentry, Peikert, and Vaikuntanathan [38]. Brakerski et al. [21] further refined this algorithm to match the discrete Gaussian exactly. Specialized samplers for q-ary lattices are provided by Bollauf, Lie, and Ling [16] and Peikert [73]. Finally, Micciancio and Walter [60] devise a constant-time sampler for the discrete Gaussian on $\mathbb{Z}^n$.

Outline: Chapter 2 introduces preliminaries required throughout this thesis, including our notational conventions in Section 2.1, basic cryptographic notions in Section 2.2, and the fundamentals of lattice cryptography in Section 2.3. Chapter 3 contains our formalization and implementation of discrete Gaussian sampling. It is separated into three sections, which address sampling in one dimension (Section 3.1), sampling in n dimensions (Section 3.2), and our implementation (Section 3.3). In Chapter 4, we explain how k-repetition is used

to achieve IND-CCA2-security using IND-CPA-secure primitives. Chapter 5 encompasses our investigation into combining the LIP-KEM with k-repetition. We begin by discussing the base LIP-KEM in Section 5.1, then establish our formal requirements in Section 5.2. Following that, Section 5.3, Section 5.4, and Section 5.5 discuss how correlating different parts of the LIP-KEM does not lead to a viable construction for use with k-repetition. In order, these parts are the error terms, the lattice points, and external components. Section 5.6 explores an alternative approach involving trapdoor functions that we also show to fail. Finally, Chapter 6 concludes this thesis.

2 Preliminaries

This section introduces the necessary background for and core concepts used throughout this thesis. We begin by detailing our notational conventions in Section 2.1, then explain basic cryptographic notions and primitives in Section 2.2 before moving on to lattice cryptography in Section 2.3.

2.1 Notation

We adhere to the notation used in the related work [33, 93] where possible, but deviate where it would lead to overloading of notation or possible confusion.

We use the common notation $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ to refer to the integers, rationals, and real numbers. The ring of integers modulo q is denoted by $\mathbb{Z}_q \coloneqq \mathbb{Z}/q\mathbb{Z}$ throughout this work. To avoid confusion, the p-adic integers are referred to as $\mathbb{P}_p$. Vectors are always denoted using bold letters (e.g., $\boldsymbol{x}$) and are interpreted as column vectors. The concatenation of two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ is $(\boldsymbol{x} \| \boldsymbol{y}) \in \mathbb{R}^{2n}$. The vector $\boldsymbol{o} = (0 \cdots 0)^T$ is the zero vector; its dimension is implied from context. Similarly, $\mathbb{1}_n$ refers to the identity matrix of dimension n. Its columns $\hat{\boldsymbol{e}}_1, \ldots, \hat{\boldsymbol{e}}_n$ are the standard basis of $\mathbb{R}^n$. The default vector and matrix norms in this work are the Euclidean norm $\|\boldsymbol{x}\| = \|\boldsymbol{x}\|_2$ for vectors and the maximum column 2-norm $\|B\| = \max_{i \in [m]} \|\boldsymbol{b}_i\|$ for matrices $B \in \mathbb{R}^{n \times m}$ with columns $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m$. $B_r(\boldsymbol{x})$ is the Euclidean ball of radius r around $\boldsymbol{x}$. The standard scalar product between two vectors is given by $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^{n} x_i y_i$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. The Gram-Schmidt-orthogonalization of an ordered vector sequence $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ is the sequence $(\tilde{\boldsymbol{b}}_1, \ldots, \tilde{\boldsymbol{b}}_n)$, where each $\tilde{\boldsymbol{b}}_i$ is the orthogonal projection of $\boldsymbol{b}_i$ onto the space orthogonal to $\operatorname{span}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1})$. Where not noted otherwise, this projection uses the standard scalar product. The Gram-Schmidt-orthogonalization $\widetilde{B}$ of a matrix B treats the columns of B as an ordered sequence of vectors. Given a function $f \colon A \to B$, we define $f(S) \coloneqq \sum_{x \in S} f(x)$ for countable $S \subseteq A$. The ceiling and floor functions are written as $\lceil x \rceil$ and $\lfloor x \rfloor$ for $x \in \mathbb{R}$. Similarly, the sign function $\operatorname{sign}(\boldsymbol{x})$ follows its standard definition. If two functions f and g are proportional to each other (i.e., $f = cg$ for constant c), we write $f \propto g$.

We distinguish algorithms from functions through the use of Small Caps. For probabilistic algorithms that take external randomness as input, the randomness input is separated from the other arguments by a semicolon as in $\textsc{Alg}(x; r)$. The security parameter of any cryptographic schemes is λ. To avoid confusion, any cryptographic keys are denoted by a two-letter combination like pk or sk. The individual letter k is reserved as a parameter of the discussed schemes. The notation $(x)_{i \in [a]}$ refers to the ordered sequence $(x_{[i \mapsto 1]}, \ldots, x_{[i \mapsto a]})$

for $i \in [a] \coloneqq \{1, \ldots, a\}$, where $x_{[i \mapsto j]}$ syntactically replaces occurrences of i in x with j. For example, the sequence $(c_1, \ldots, c_k)$ is denoted as $(c_i)_{i \in [k]}$, while $(a_1 + b_j x_1, \ldots, a_k + b_j x_k)$ is written as $\left(a_i + b_j x_i\right)_{i \in [k]}$. The index range is $[k]$ in both of these cases.

The groups of permutation matrices and signed permutation matrices in n dimensions are $P_n \subset \{0,1\}^{n \times n}$ and $SP_n \subset \{0, \pm 1\}^{n \times n}$ respectively. For any set $A \subset [0,1]$, $O_n(A)$ is the group of orthogonal matrices with entries in A. These $O \in O_n(A)$ satisfy $O^T O = \mathbb{1}_n$ and $\det(O) = \pm 1$ with the semantics of real-valued arithmetic. The general linear groups of invertible matrices of dimension n over a ring R or a field F are denoted as $\mathrm{GL}_n(R)$ and $\mathrm{GL}_n(F)$. The set $S_n^{>0}(R)$ is the set of positive definite matrices with entries in R. Its elements $Q \in S_n^{>0}(R)$ are identified with the corresponding quadratic forms $Q(\boldsymbol{x}) = \boldsymbol{x}^T Q \boldsymbol{x}\ \forall \boldsymbol{x} \in \mathbb{R}^n$ and referred to as such.

The left arrow in $X \leftarrow \mathcal{X}$ indicates that X is sampled according to the distribution $\mathcal{X}$ and is also used in algorithms as the assignment operator. For a random variable X following $\mathcal{X}$, we also write $X \sim \mathcal{X}$. The support of a distribution $\mathcal{X}$ is given by $\mathrm{supp}(\mathcal{X}) = \{\boldsymbol{x} \mid \Pr_{X \sim \mathcal{X}}[X = \boldsymbol{x}] > 0\}$. The notation $\mathcal{U}(A)$ refers to the uniform distribution over the set A. In addition, the shorthand $X \leftarrow\!\!\$\ A$ is equivalent to $X \leftarrow \mathcal{U}(A)$. For any set A, $|A|$ is the cardinality of A. Big-O notation is used in the standard manner.

2.2 Cryptographic Notions

Here, we give a brief overview of the cryptographic notions used in this thesis. We refer readers to the standard literature [17] for details on these definitions.

Definition 2.1 (Negligible functions)
A function $f\colon \mathbb{N} \to \mathbb{R}$ is *negligible* if $f(\lambda) \in o(1/p(\lambda))$ for every polynomial p. Conversely, f is called *overwhelming* if $1 - f$ is negligible. We also write $f \in \mathrm{negl}(\lambda)$ for negligible f.

Definition 2.2 (Efficiency)
A Turing machine (TM) is *probabilistic polynomial-time* (PPT) if it has access to a tape of randomness and has its running time bounded by a polynomial in the length of its input. An algorithm is *efficient* if it can be implemented by a PPT TM. Similarly, a function is *efficiently computable* if it can be computed by a PPT TM, a set is *efficiently recognizable* if its membership can be decided by a PPT TM, and a distribution is *efficiently samplable* if samples following it can be computed by a PPT TM.

Definition 2.3 (Indistinguishability)
Let $\{\mathcal{X}\}_{\lambda \in \mathbb{N}}$ and $\{\mathcal{Y}\}_{\lambda \in \mathbb{N}}$ be families of efficiently samplable distributions and $\mathcal{A}$ be an arbitrary halting TM. The *distinguishing game* or "experiment" $\mathrm{Exp}^{\mathrm{dist}}_{\mathcal{X},\mathcal{Y}}(\mathcal{A}, \lambda)$ is the following process:

$$\frac{\mathsf{Exp}^{\mathrm{dist}}_{\mathcal{X},\mathcal{Y}}(\mathcal{A},\lambda)}{\begin{array}{l} b \leftarrow\!\!\$\ \{0,1\} \\ z_0 \leftarrow \mathcal{X} \\ z_1 \leftarrow \mathcal{Y} \\ \textbf{return } \mathcal{A}(1^\lambda, z_b) \end{array}}$$

The distinguishing *advantage* of $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{\mathrm{dist}}_{\mathcal{X},\mathcal{Y}}(\mathcal{A},\lambda) := \left|\Pr\left[\mathsf{Exp}^{\mathrm{dist}}_{\mathcal{X},\mathcal{Y}}(\mathcal{A},\lambda) = 1 \;\middle|\; b = 1\right] - \Pr\left[\mathsf{Exp}^{\mathrm{dist}}_{\mathcal{X},\mathcal{Y}}(\mathcal{A},\lambda) = 1 \;\middle|\; b = 0\right]\right|.$$

If $\mathsf{Adv}^{\mathrm{dist}}_{\mathcal{X},\mathcal{Y}}(\mathcal{A},\lambda) \in \mathrm{negl}(\lambda)$ for every $\mathcal{A}$, we write $\mathcal{X} \overset{\mathrm{s}}{\approx} \mathcal{Y}$ and call $\mathcal{X}$ and $\mathcal{Y}$ *statistically indistinguishable*. If the advantage is negligible for every PPT $\mathcal{A}$, we write $\mathcal{X} \overset{\mathrm{c}}{\approx} \mathcal{Y}$ and call $\mathcal{X}$ and $\mathcal{Y}$ *computationally indistinguishable*. Note that $\mathcal{X} \overset{\mathrm{s}}{\approx} \mathcal{Y} \implies \mathcal{X} \overset{\mathrm{c}}{\approx} \mathcal{Y}$. Processes like $\mathsf{Exp}^{\mathrm{dist}}$ with a security parameter input λ and a TM input $\mathcal{A}$ are referred to as *security games*. In this context, $\mathcal{A}$ is also typically called an *adversary* or *attacker*. After this point, security parameter inputs are suppressed for brevity.

Definition 2.4 (Random oracle)
A *random oracle* O for efficiently recognizable and finite input and output sets A and B is a probabilistic algorithm that computes a uniformly random function $f_R \leftarrow\!\!\$\ B^A$.

In the *standard model* of security, TMs receive no inputs beyond those specified in their corresponding security games and do not have access to random oracles. Contrast this with the *random oracle model* (ROM), where TMs can access one or more random oracles during their execution. Compare also the *common reference string* (CRS) model, in which TMs receive an additional trusted input string following some distribution [25].

2.2.1 One-way Functions

Definition 2.5 (One-way function)
Let $\{F\}_{\lambda\in\mathbb{N}}$ be a family of efficiently computable functions with the domain $\{A\}_{\lambda\in\mathbb{N}}$ and let $\{\mathcal{I}\}_{\lambda\in\mathbb{N}}$ be a family of efficiently samplable distributions on A. We define the *one-way game* $\mathsf{Exp}^{\mathrm{OWF}}_{(F,\mathcal{I})}(\mathcal{A})$ as:

$$\frac{\mathsf{Exp}^{\mathrm{OWF}}_{(F,\mathcal{I})}(\mathcal{A})}{\begin{array}{l} x \leftarrow \mathcal{I} \\ x' \leftarrow \mathcal{A}(F(x)) \\ \textbf{return } x' \in A \wedge F(x') \overset{?}{=} F(x) \end{array}}$$

The advantage of $\mathcal{A}$ in the one-way game is given by

$$\mathsf{Adv}^{\mathrm{OWF}}_{(F,\mathcal{I})}(\mathcal{A}) := \Pr\left[\mathsf{Exp}^{\mathrm{OWF}}_{(F,\mathcal{I})}(\mathcal{A}) = 1\right].$$

The tuple $(\{F\}_{\lambda\in\mathbb{N}}, \{\mathcal{I}\}_{\lambda\in\mathbb{N}})$ is a *one-way function* if $\mathsf{Adv}^{\mathrm{OWF}}_{(F,\mathcal{I})}(\mathcal{A}) \in \mathrm{negl}(\lambda)$ for all PPT $\mathcal{A}$.

Intuitively, one-way functions are hard to invert. *Trapdoor one-way functions* (TDOWFs) are an extension of this concept — these are hard to invert unless one knows a secret trapdoor that makes computing preimages efficient.

Definition 2.6 (TDOWF)
A TDOWF is a tuple $(\text{Gen}_T, F, F^{-1}, \{\mathcal{I}\}_{\lambda\in\mathbb{N}})$ of three PPT algorithms and a family of efficiently samplable input distributions $\mathcal{I}$ where:

- $\text{Gen}_T \to (s, td)$: Gen_T generates a public function description s and a trapdoor td.
- $F(s, \cdot)$ computes an injective function such that $(F, \mathcal{I})$ is one-way[1] and
- $F^{-1}(td, \cdot)$ inverts F with $F^{-1}(td, F(s, x)) = x \; \forall (s, td) \leftarrow \text{Gen}_T$.

A closely related notion is the *lossy trapdoor function*:

Definition 2.7 (Lossy trapdoor function ([76], notation following [84]))
A (λ, l)-*lossy trapdoor function* is a tuple of PPT algorithms $(\text{Gen}_T, F, F^{-1})$ such that:

- $\text{Gen}_T(\texttt{injective})$ outputs (s, td) such that $F(s, \cdot)$ computes an injective function with the domain $\{0,1\}^\lambda$. $F(s, \cdot)$ is inverted by $F^{-1}(td, \cdot)$ as in Definition 2.6.
- $\text{Gen}_T(\texttt{lossy})$ outputs s such that $F(s, \cdot)$ is a function with the domain $\{0,1\}^\lambda$ and an image of cardinality at most $2^{\lambda-l}$.
- It holds that $\{s \mid (s, \cdot) \leftarrow \text{Gen}_T(\texttt{injective})\} \overset{c}{\approx} \text{Gen}_T(\texttt{lossy})$.

Essentially, the injective functions of a (λ, l)-lossy trapdoor function are indistinguishable from functions that lose at least l bits of information. Peikert and Waters [76, Lemma 3.1] show that $l \in \omega(\log \lambda)$ suffices for the injective functions of a (λ, l)-lossy trapdoor function to be one-way, in which case we refer to it as a (λ, l)-lossy TDOWF. One-way functions can additionally have *hardcore predicates*, which are predicates of the input that are hard to predict given only the output of the function:

Definition 2.8 (Hardcore predicate)
A *hardcore predicate* for a one-way function $(\{F\}_{\lambda\in\mathbb{N}}, \{\mathcal{I}\}_{\lambda\in\mathbb{N}})$ with the domain $\{A\}_{\lambda\in\mathbb{N}}$ is a function $h\colon A \to \{0,1\}$ such that

$$\big\{(F(x), h(x)) \mid x \leftarrow \mathcal{I}\big\} \overset{c}{\approx} \big\{(F(x), r) \mid x \leftarrow \mathcal{I}, r \leftarrow\!\!\$\; \{0,1\}\big\}.$$

The Goldreich-Levin theorem [40] states that any one-way function can be turned into a one-way function with a hardcore predicate. This transformation preserves the input distribution.

[1] In this parametrized setting, the descriptor s is freshly generated using Gen_T at the start of the one-way game and given to $\mathcal{A}$. $\mathcal{A}$ does not receive td.

2.2.2 Encryption Schemes

Definition 2.9 (Secret-key encryption schemes)
A *secret-key encryption scheme* (SKE) is a tuple of PPT algorithms (Gen, Enc, Dec) where

- $\textsc{Gen} \longrightarrow sk$: Gen generates a secret key sk.
- $\textsc{Enc}(sk, m) \longrightarrow c$: Enc encrypts a message $m \in \mathcal{M}$ using the secret key sk and outputs a ciphertext c.
- $\textsc{Dec}(sk, c) \longrightarrow m$: Dec decrypts the ciphertext c with the secret key sk to recover the message m or fails on an invalid ciphertext and outputs $\perp$.

We demand *correctness* of SKE schemes, which means that $\textsc{Dec}(sk, \textsc{Enc}(sk, m)) = m$ for all $sk \leftarrow \textsc{Gen}, m \in \mathcal{M}$. SKE schemes are also referred to as *symmetric* encryption schemes.

Definition 2.10 (IND-CPA)
Given an SKE $\mathrm{SKE} = (\textsc{Gen}, \textsc{Enc}, \textsc{Dec})$, define the following *indistinguishability under chosen-plaintext attack* (IND-CPA) game:

$\mathsf{Exp}^{\text{IND-CPA}}_{\mathrm{SKE}}(\mathcal{A})$
$b \xleftarrow{\$} \{0,1\}$
$sk \leftarrow \textsc{Gen}$
$(m_0, m_1, \textbf{state}) \leftarrow \mathcal{A}(\texttt{find})$
if $|m_0| \neq |m_1|$ **then abort**
$b' \leftarrow \mathcal{A}^{\textsc{Enc}(sk,\cdot)}(\texttt{guess}, \textsc{Enc}(sk, m_b), \textbf{state})$
return $b' \stackrel{?}{=} b$

Here, $\mathcal{A}^{\textsc{Enc}(sk,\cdot)}$ indicates that the challenger (the hypothetical operator of the game, often denoted C) provides the function $\textsc{Enc}(sk, \cdot)$ as an encryption oracle for $\mathcal{A}$ to use with arbitrary messages. If

$$\mathrm{Adv}^{\text{IND-CPA}}_{\mathrm{SKE}}(\mathcal{A}) := \left| \Pr\left[\mathsf{Exp}^{\text{IND-CPA}}_{\mathrm{SKE}}(\mathcal{A}) = 1\right] - \frac{1}{2} \right| \in \mathrm{negl}(\lambda),$$

we say that SKE is IND-CPA-secure.[2]

Definition 2.11 (Public-key encryption)
A *public-key encryption scheme* (PKE) is a tuple of PPT algorithms (Gen, Enc, Dec) such that

- $\textsc{Gen} \longrightarrow (pk, sk)$: Gen generates both a *public key* pk used for encryption and a *secret key* sk used for decryption.
- $\textsc{Enc}(pk, m) \longrightarrow c$: Enc encrypts a message $m \in \mathcal{M}$ with the public key pk to form a ciphertext c.

[2] The subtraction of $1/2$ in the definition of the advantage accounts for the fact that an adversary $\mathcal{A}$ randomly guessing b would trivially win the game with a probability of $1/2$.

- $\textsc{Dec}(sk, c) \to m$: Dec decrypts a ciphertext c with the secret key sk to retrieve m. Dec can also fail and output $\perp$.

Correctness for a PKE is defined analogously to that of an SKE. The IND-CPA security game $\mathsf{Exp}^{\text{IND-CPA}}_{\text{PKE}}$ for a PKE works nearly identically to $\mathsf{Exp}^{\text{IND-CPA}}_{\text{SKE}}$ with the exception that the adversary is provided with the public key pk instead of an encryption oracle. We also call PKEs *asymmetric* encryption schemes.

Definition 2.12 (IND-CCA1 and IND-CCA2)
Let PKE = (Gen, Enc, Dec) be a PKE. We define the following security game and call it *indistinguishability under adaptive chosen-ciphertext attacks*:

$\mathsf{Exp}^{\text{IND-CCA2}}_{\text{PKE}}(\mathcal{A})$

$b \xleftarrow{\$} \{0,1\}$
$(pk, sk) \leftarrow \textsc{Gen}$
$c^* \leftarrow \perp$
$(m_0, m_1, \textbf{state}) \leftarrow \mathcal{A}^{\textsc{Dec}}(\texttt{find}, pk)$
if $|m_0| \neq |m_1|$ **then abort**
$c^* \leftarrow \textsc{Enc}(pk, m_b)$
$b' \leftarrow \mathcal{A}^{\textsc{Dec}}(\texttt{guess}, pk, c^*, m_b, \textbf{state})$
return $b' \stackrel{?}{=} b$

$\textsc{Dec}(c)$

if $c = c^*$ **then abort**
return $\textsc{Dec}(sk, c)$

Again, PKE is considered IND-CCA2-secure if $\text{Adv}^{\text{IND-CCA2}}_{\text{PKE}}(\mathcal{A})$ (defined analogously to $\text{Adv}^{\text{IND-CPA}}_{\text{SKE}}(\mathcal{A})$) is negligible for all PPT $\mathcal{A}$. In this game, the adversary additionally gets access to a decryption oracle, making IND-CCA2 security a stronger property than IND-CPA security. We also define *IND-CCA1*, which is a weaker version of IND-CCA2 where the adversary is no longer allowed to query the Dec oracle in the guessing phase. Both IND-CCA1 and IND-CCA2 can also be defined for SKE schemes.

A cryptographic primitive closely related to PKEs is the *key encapsulation mechanism* (KEM). Instead of directly encrypting messages, a KEM "encapsulates" a random key for use with another primitive (often an SKE in a scheme known as *hybrid encryption*).

Definition 2.13 (Key encapsulations)
A KEM is a tuple of PPT algorithms (Gen, Encaps, Decaps) with the following definitions:

- $\textsc{Gen} \to (pk, sk)$: Gen generates a public-private keypair just as for a PKE.
- $\textsc{Encaps}(pk) \to (c, ek)$: Encaps generates a random key $ek \in \mathcal{K}$ and a ciphertext c encapsulating that key.
- $\textsc{Decaps}(sk, c) \to ek$: Decaps uses the secret key sk to decapsulate the key ek from c. It outputs $\perp$ on failure.

We require correctness just like for encryption schemes: For $(pk, sk) \leftarrow \textsc{Gen}$ and $(c, ek) \leftarrow \textsc{Encaps}(pk)$, we must have $\textsc{Decaps}(sk, c) = ek$.

While the definitions of IND-CPA security and IND-CCA2 security for KEMs are very similar to the corresponding definitions for PKEs, we present them in full due to their significance for this thesis:

Definition 2.14 (KEM indistinguishability)
Let KEM = (Gen, Encaps, Decaps) be a KEM and define the two games $\mathsf{Exp}^{\text{KEM-IND-CPA}}_{\text{KEM}}$ and $\mathsf{Exp}^{\text{KEM-IND-CCA2}}_{\text{KEM}}$:

$\mathsf{Exp}^{\text{KEM-IND-CPA}}_{\text{KEM}}(\mathcal{A})$
$b \leftarrow\$ \{0,1\}$
$(pk, sk) \leftarrow \textsc{Gen}$
$(c^*, ek_0) \leftarrow \textsc{Encaps}(pk)$
$ek_1 \leftarrow\$ \mathcal{K}$
$b' \leftarrow \mathcal{A}(pk, c^*, ek_b)$
return $b' \stackrel{?}{=} b$

$\mathsf{Exp}^{\text{KEM-IND-CCA2}}_{\text{KEM}}(\mathcal{A})$
$b \leftarrow\$ \{0,1\}$
$(pk, sk) \leftarrow \textsc{Gen}$
$(c^*, ek_0) \leftarrow \textsc{Encaps}(pk)$
$ek_1 \leftarrow\$ \mathcal{K}$
$b' \leftarrow \mathcal{A}^{\textsc{Decaps}}(pk, c^*, ek_b)$
return $b' \stackrel{?}{=} b$

$\textsc{Decaps}(c)$
if $c = c^*$ **then abort**
return $\textsc{Decaps}(sk, c)$

In both games, the adversary is challenged with distinguishing the real encapsulated key ek_0 from a random key ek_1. A KEM is IND-CPA-secure or IND-CCA2-secure if any PPT adversary's advantage in the corresponding game is negligible. We abbreviate KEM-IND-CPA and KEM-IND-CCA2 as IND-CPA and IND-CCA2 in the context of KEMs.

2.2.3 Signatures

A *signature scheme* (SIG) is a cryptographic primitive that produces unforgeable signatures for messages, preventing an adversary from forging or manipulating messages from an honest sender.

Definition 2.15 (Signatures)
A SIG is a tuple of PPT algorithms ($\textsc{Gen}_S$, $\textsc{Sign}_S$, $\textsc{Verify}_S$) with the following properties:

- $\textsc{Gen}_S \rightarrow (sk, vk)$: $\textsc{Gen}_S$ creates a keypair consisting of a secret *signing key* sk and a public *verifying key* vk.
- $\textsc{Sign}_S(sk, m) \rightarrow \sigma$: $\textsc{Sign}_S$ generates a signature σ for the message $m \in \mathcal{M}$ using the signing key sk.
- $\textsc{Verify}_S(vk, m, \sigma) \rightarrow b$: $\textsc{Verify}_S$ checks whether the signature σ is valid for the message m under the verification key vk. If so, it outputs 1. Otherwise, its output is 0.

Correctness for SIGs demands that honestly generated signatures pass verification:

$$\textsc{Verify}_S(vk, m, \textsc{Sign}_S(sk, m)) = 1 \quad \forall (sk, m) \leftarrow \textsc{Gen}_S, m \in \mathcal{M}.$$

Definition 2.16 (sEUF-1-CMA)
Given a SIG $\mathrm{SIG} = (\textsc{Gen}_S, \textsc{Sign}_S, \textsc{Verify}_S)$, let $\mathsf{Exp}^{\text{sEUF-1-CMA}}_{\mathrm{SIG}}$ be the following *strong one-time unforgeability under chosen-message attack* game:

$\mathsf{Exp}^{\text{sEUF-1-CMA}}_{\mathrm{SIG}}(\mathcal{A})$	$\textsc{Sign}(m)$
$\mathcal{L} \leftarrow \emptyset,\ q \leftarrow 0$	**if** $q \geq 1$ **then abort**
$(sk, vk) \leftarrow \textsc{Gen}_S$	$\sigma \leftarrow \textsc{Sign}_S(sk, m)$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\textsc{Sign}}(vk)$	$\mathcal{L} \leftarrow \mathcal{L} \cup \{(m, \sigma)\}$
return $(m^*, \sigma^*) \notin \mathcal{L} \wedge \textsc{Verify}_S(vk, m^*, \sigma^*) = 1$	$q \leftarrow q + 1$
	return σ

Essentially, $\mathcal{A}$ must forge a fresh signature σ^* for a message while only being allowed to request a single signature from the $\textsc{Sign}$ oracle. SIG satisfies the sEUF-1-CMA property if it holds that

$$\mathsf{Adv}^{\text{sEUF-1-CMA}}_{\mathrm{SIG}}(\mathcal{A}) \coloneqq \Pr[\mathsf{Exp}^{\text{sEUF-1-CMA}}_{\mathrm{SIG}}(\mathcal{A}) = 1] \in \mathrm{negl}(\lambda)$$

for any PPT $\mathcal{A}$.

2.2.4 Extractors

The last primitive we define here is the *randomness extractor*: It combines a source of entropy with a public, uniformly random seed to produce fresh uniform randomness as output. The definition of the minimum entropy (as used by multiple authors [33, 69]) of a random variable X with values in $\mathcal{X}$ as measured in bits is

$$H_\infty(X) \coloneqq -\log_2\big(\max_{x \in \mathcal{X}} \Pr[X = x]\big)$$

and represents a lower bound on the regular Shannon entropy

$$H(X) \coloneqq -\sum_{x \in \mathcal{X}} \Pr[X = x] \log_2\big(\Pr[X = x]\big).$$

Definition 2.17 (Extractor (simplified from [69, Definition 2], notation following [33]))
An efficiently computable function $\mathcal{E} \colon \mathcal{X} \times \{0,1\}^z \to \{0,1\}^l$ is an *l-extractor* if, for a random variable X following a distribution $\mathcal{D}$ over $\mathcal{X}$ with $H_\infty(X) \geq l$, we have

$$\big\{(Z, \mathcal{E}(X, Z)) \;\big|\; X \leftarrow \mathcal{D},\ Z \leftarrow\!\$\ \{0,1\}^z\big\} \stackrel{s}{\approx} \big\{(Z, Y) \;\big|\; Y \leftarrow\!\$\ \{0,1\}^l,\ Z \leftarrow\!\$\ \{0,1\}^z\big\}.$$

2.3 Lattice Cryptography

This section introduces the fundamental concepts of lattice cryptography required for this thesis before moving to explain discrete Gaussian distributions in Section 2.3.1, lattice isomorphisms in Section 2.3.2, and the Learning with Errors problem in Section 2.3.3. The definitions in this section are taken from the dissertation of van Woerden [93] and adjusted for notational differences except where noted otherwise.

A lattice $\Lambda \subset \mathbb{R}^n$ is a set of discrete points in a regular pattern embedded in $\mathbb{R}^n$. For any $\boldsymbol{v}, \boldsymbol{w} \in \Lambda$ and $a \in \mathbb{Z}$, it holds that $\boldsymbol{v}+\boldsymbol{w} \in \Lambda$ and $a\boldsymbol{v} \in \Lambda$. A lattice can therefore be interpreted as a "discrete vector space" in $\mathbb{R}^n$ over the integers. For a simple example, consider the trivial lattice $\mathbb{Z}^n$. Analogously to a vector space, a lattice Λ is also described by a basis $B \in \mathbb{R}^{n \times m}$ of full rank with $m \leq n$ such that

$$\Lambda(B) \coloneqq B\mathbb{Z}^m = \left\{ \sum_{i=0}^{m} x_i \boldsymbol{b}_i \;\middle|\; x_i \in \mathbb{Z} \; \forall i \in [m] \right\},$$

where we denote the columns of B as $(\boldsymbol{b}_1 \cdots \boldsymbol{b}_m)$ and call them the *basis vectors* of the lattice. Every matrix $B \in \mathbb{R}^{n \times m}$ defines a lattice $\Lambda(B)$ and every lattice has a basis. For instance, we have $\mathbb{Z}^n = \mathbb{1}_n \mathbb{Z}^n$, so $\mathbb{1}_n$ is a basis of $\mathbb{Z}^n$. The *dimension* of a lattice is the dimension n of the $\mathbb{R}^n$ it is embedded in, while its *rank* is the $\operatorname{rank}(B)$ of its basis. A *full-rank* lattice is one with a full-rank basis. We define $\operatorname{span}(\Lambda(B)) \coloneqq \operatorname{span}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m)$ and note that full-rank lattices always have $\operatorname{span}(\Lambda) = \mathbb{R}^n$. For a lattice point $\boldsymbol{v} = B\boldsymbol{x} \in \Lambda(B)$ with $\boldsymbol{x} \in \mathbb{Z}^m$, we call $\boldsymbol{x}$ the *coefficient vector* of the point $\boldsymbol{v}$. The distance of any vector $\boldsymbol{t} \in \mathbb{R}^n$ to the lattice Λ is given by $\operatorname{dist}(\Lambda, \boldsymbol{t}) \coloneqq \min_{\boldsymbol{v} \in \Lambda} \|\boldsymbol{t} - \boldsymbol{v}\|$.

We call two bases B and B' *equivalent* if $\Lambda(B) = \Lambda(B')$, i.e., they generate the same lattice. This is the case if and only if $B' = BU$ for a matrix $U \in \mathrm{GL}_m(\mathbb{Z})$ by straightforward reasoning: If B and B' are equivalent, then every $\boldsymbol{b}'_i$ must be representable as a sum of integer multiples of the $\boldsymbol{b}_i$, which means that $B' = BU$ for some $U \in \mathbb{Z}^{m \times m}$. Since the same also applies when swapping the roles of B and B', we also have $B = B'U^{-1}$ for $U^{-1} \in \mathbb{Z}^{m \times m}$, so it follows that $U \in \mathrm{GL}_m(\mathbb{Z})$. Conversely, if $B' = BU$ for such a U, then the bases generate the same lattices and are thus equivalent. The $U \in \mathrm{GL}_m(\mathbb{Z})$ are called *unimodular matrices* and have $\det(U) = \pm 1$, which is equivalent to being invertible over the ring $\mathbb{Z}$. As discussed by Regev [79, Lemma 4], unimodular matrices are also characterized by a constructive approach — The set of matrices generated by applying a sequence of the following operations to the identity matrix $\mathbb{1}_m$ with the columns $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$ are exactly the unimodular matrices: permutations ($\boldsymbol{u}_i \leftrightarrow \boldsymbol{u}_j$), inversions ($\boldsymbol{u}_i \leftarrow -\boldsymbol{u}_i$), and integer column additions ($\boldsymbol{u}_i \leftarrow \boldsymbol{u}_i + a\boldsymbol{u}_j,\ a \in \mathbb{Z}$). The constructive approach makes it clear that a lattice with $m > 1$ has infinitely many bases.

A basis B of a lattice $\Lambda(B)$ also defines a *fundamental parallelepiped*

$$\mathcal{P}(B) \coloneqq \left\{ B(x_1 \cdots x_m) \;\middle|\; 0 \leq x_i < 1 \; \forall i \in [m] \right\}$$

with $\mathcal{P}(B) \cap \Lambda(B) = \{0\}$ [79, Lemma 1]. In general, we find that $\mathrm{vol}(\mathcal{P}(B)) = \sqrt{\det(B^T B)}$, which simplifies to $\mathrm{vol}(\mathcal{P}(B)) = |\det(B)|$ if B is of full rank. By the above equalities and the fact that $|\det(U)| = 1$ for any unimodular matrix U, we see that $\mathrm{vol}(\mathcal{P}(B)) =: \det(\Lambda(B))$ is a property of the lattice $\Lambda(B)$ independent of the choice of basis [79, Definition 5]. Figure 2.1 visualizes two different bases for a lattice along with the corresponding parallelepipeds.

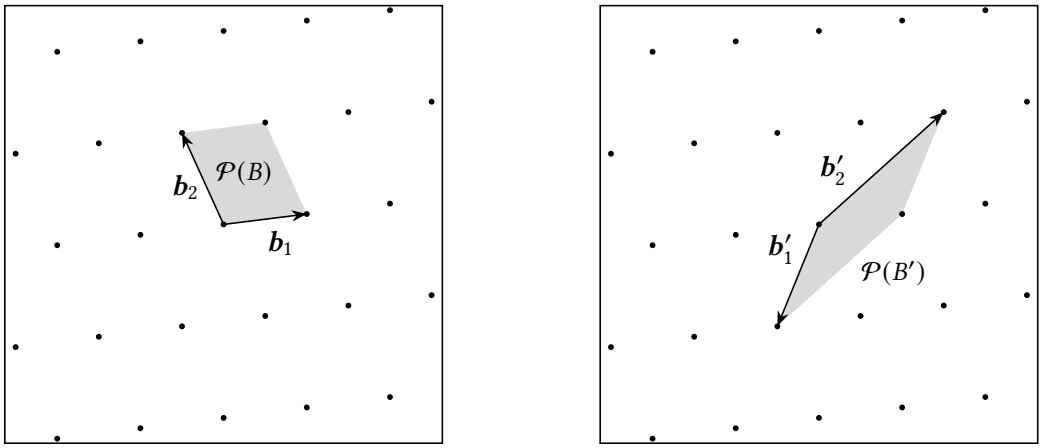


Figure 2.1: A lattice Λ with two different bases. Note how $\mathrm{vol}(\mathcal{P}(B)) = \mathrm{vol}(\mathcal{P}(B')) = \det(\Lambda)$.

Full-rank lattices can also be described through the Gram matrices $Q = B^T B$ of their bases. These Q are positive definite quadratic forms and induce the geometry of the lattice $\Lambda(B)$ on the lattice point coefficients $\boldsymbol{x} \in \mathbb{Z}^n$. We define the inner product $\langle \cdot, \cdot \rangle_Q \colon \mathbb{Z}^n \times \mathbb{Z}^n \to \mathbb{R}$ induced by Q with

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_Q := \boldsymbol{x}^T Q \boldsymbol{y} = \boldsymbol{x}^T B^T B \boldsymbol{y} = \langle B\boldsymbol{x}, B\boldsymbol{y} \rangle$$

and its corresponding norm $\|\boldsymbol{x}\|_Q := \sqrt{\langle \boldsymbol{x}, \boldsymbol{x} \rangle_Q}$. While the quadratic form preserves the geometry among the lattice points, it is invariant to orthogonal transformations $O\Lambda(B)$ of the lattice (with $O \in O_n(\mathbb{R})$ since $B^T O^T O B = B^T B$. Representing a lattice using its quadratic form thus loses its orientation in $\mathbb{R}^n$. For $\Lambda(B)$ described by $Q = B^T B$, the Cholesky decomposition $Q = B_Q^T B_Q$ with an upper diagonal matrix B_Q can be used to efficiently recover a basis $B_Q = OB$ of a lattice $O\Lambda(B)$ for some $O \in O_n(\mathbb{R})$.[3] In general, we use the term *lattice space* to refer to the $\mathbb{R}^n \supset \Lambda(B)$ with the standard Euclidean geometry. Coefficient vectors, on the other hand, reside in *coefficient space*, which is the $\mathbb{R}^n \supset \mathbb{Z}^n$ with the geometry induced by Q.

An important property of a lattice is its sequence of successive minima

$$\lambda_i(\Lambda) := \min\left\{ r \in \mathbb{R}_{>0} \;\middle|\; \dim(\mathrm{span}(\Lambda \cap B_r(\boldsymbol{o}))) \geq i \right\} \quad \forall i \in [\mathrm{rank}(\Lambda)]$$

with the first minimum $\lambda_1(\Lambda) = \min_{\boldsymbol{v} \in \Lambda \setminus \{0\}} \|\boldsymbol{v}\|$ in particular corresponding to the minimum distance between lattice points [79, Definition 7]. In Figure 2.1, we have $\|\boldsymbol{b}_1\| = \lambda_1(\Lambda)$ and $\|\boldsymbol{b}_2\| = \lambda_2(\Lambda)$. A bound on the first minimum is provided by Minkowski's theorem:

[3] Specifically, the Cholesky decomposition recovers the unique orientation of the lattice where the first basis vector is aligned with the first axis, the second basis vector is in the span of the first two axes, and so on.

Theorem 2.1 (Minkowski's Theorem ([79, Corollary 2])). *For a lattice $\Lambda \subset \mathbb{R}^n$ of full rank,*

$$\lambda_1(\Lambda) \leq \sqrt{n}\det(\Lambda)^{1/n}.$$

While the above is a guaranteed bound on the first minimum of a lattice, it is also useful to consider the first minimum of an "average" lattice with a certain determinant. The Gaussian heuristic fulfills this purpose — by treating the lattice as a point cloud of density $1/\det(\Lambda)$, we get the following estimate:

Definition 2.18 (Gaussian heuristic ([93, Heuristic claim 57]))
For a lattice $\Lambda \subset \mathbb{R}^n$ of full rank,

$$\mathrm{gh}(\Lambda) \coloneqq \frac{\det(\Lambda)^{1/n}}{\mathrm{vol}(B_1(\boldsymbol{o}))^{1/n}} \approx \sqrt{\frac{n}{2\pi e}}\det(\Lambda)^{1/n}$$

is the Gaussian heuristic estimate of the first minimum $\lambda_1(\Lambda)$.

Since both the $\lambda_i(\Lambda)$ and $\mathrm{gh}(\Lambda)$ are independent of the orientation of the lattice, we also write $\lambda_i(Q)$ and $\mathrm{gh}(Q)$ to refer to those properties of the lattice $\Lambda(B_Q)$ for a quadratic form $Q = B_Q^T B_Q$. For a given lattice Λ, we define its dual Λ^* as

$$\Lambda^* \coloneqq \left\{\boldsymbol{w} \in \mathrm{span}(\Lambda) \;\middle|\; \langle \boldsymbol{w}, \boldsymbol{v}\rangle \in \mathbb{Z}\ \forall \boldsymbol{v} \in \Lambda\right\}.$$

A basis D of Λ^* is given by the transposed pseudoinverse $D = B(B^TB)^{-1}$, with which $D^TB = \mathbb{1}_n$. For full-rank lattices, $D = B^{-T}$. It follows that $(\Lambda^*)^* = \Lambda$ and $\det(\Lambda^*) = 1/\det(\Lambda)$. For a quadratic form representation Q of a full-rank lattice, the quadratic form of the dual lattice is $D^TD = B^{-1}B^{-T} = Q^{-1}$, which is also positive definite. Many lattice problems can be approached via either the primal or the dual lattice. We also define the *s-hull* of a lattice Λ for $s \in \mathbb{R} \setminus \{0\}$ as $H_s(\Lambda) \coloneqq \Lambda \cap s\Lambda^*$. The s-hull is significant for certain cryptanalytic approaches to lattices.

As argued by Regev [79], lattices are cryptographically interesting because several geometric problems on lattices are believed to be computationally difficult even in a post-quantum setting. We present some basic problems of widespread relevance in lattice cryptography and that are used in this thesis:

Definition 2.19 (Shortest Vector Problem (SVP))
Given a lattice basis B, find a vector $\boldsymbol{v} \in \Lambda(B)$ with $\|\boldsymbol{v}\| = \lambda_1(\Lambda(B))$.

Definition 2.20 (f-approximate Shortest Vector Problem (f-SVP))
Given a lattice basis B and an $f \geq 1$, find a vector $\boldsymbol{v} \in \Lambda(B)$ with $\|\boldsymbol{v}\| \leq f\lambda_1(\Lambda(B))$.

Definition 2.21 (Closest Vector Problem (CVP) (generalized from [78, Definition 1]))
Given a lattice basis B and a vector $\boldsymbol{t} \in \mathbb{R}^n$, find a vector $\boldsymbol{v} \in \Lambda(B)$ with $\|\boldsymbol{v} - \boldsymbol{t}\| = \mathrm{dist}(\Lambda, \boldsymbol{t})$.

f-approximate-CVP (f-CVP) is defined analogously to f-SVP. The solution to an instance of CVP or f-CVP need not be unique; a variant where a unique solution is guaranteed to exist is given by the Bounded Distance Decoding (BDD) problem.

Definition 2.22 (δ-BDD (simplified from [93, Definition 24]))
Given a lattice basis B, a distance promise $\delta \in [0, 1/2)$ and a vector $\boldsymbol{t} \in \mathbb{R}^n$ with $\operatorname{dist}(\Lambda, \boldsymbol{t}) \leq \delta\lambda_1(\Lambda(B))$, find the unique closest vector $\boldsymbol{v} \in \Lambda(B)$ to $\boldsymbol{t}$.

The difficulty of the above problems varies depending on the lattice and the specific lattice basis given. For instance, Babai's nearest planes algorithm [9] gives better approximations to CVP if $\|\widetilde{B}\|$ is small relative to the lattice. Generally, having access to a "short" basis (e.g., one where $\|\boldsymbol{b}_i\| = \lambda_i(\Lambda(B))\ \forall i \in [n]$) where the vectors are almost orthogonal makes many lattice problems easy. Such bases are called *good* or *well-reduced*, while bases with long, near-parallel vectors are referred to as *bad*.

Some lattices admit efficient *decoding algorithms* $\textsc{Decode}\colon \mathbb{R}^n \to \Lambda$ in their canonical bases. These algorithms solve δ-BDD for some $\delta \in [0, 1/2)$. We say that such a decoding algorithm has a *decoding radius* of $r = \delta\lambda_1(\Lambda)$. For example, $\mathbb{Z}^n$ with the basis $\mathbb{1}^n$ can be decoded by simply rounding to the nearest integer vector for a decoding radius of $r \lessapprox 1/2$, while the Barnes-Wall lattices [10] have a decoding algorithm by Micciancio and Nicolosi [57] with $\delta \lessapprox 1/2$ and $r = 2^{n/2}/2$ [57]. When using decoding algorithms with quadratic forms, the input and output are considered to be coefficient vectors with the input in $\mathbb{R}^n$ and the output in $\mathbb{Z}^n$.

2.3.1 Discrete Gaussians

Gaussian functions as evaluated on the points of a lattice have been studied extensively in the literature [2, 21, 33, 38, 59, 75, 81]. The foundation for much of their current use in lattice cryptography was laid by Micciancio and Regev [59]. We adhere to Ducas and van Woerden's [33] formulation of the discrete Gaussians using quadratic forms instead of lattice bases; the content of this section is taken from their work except where annotated differently.

Consider the n-dimensional Gaussian function

$$\rho_{s,\boldsymbol{c}}(\boldsymbol{x}) \coloneqq \exp\left(-\pi\frac{\|\boldsymbol{x}-\boldsymbol{c}\|^2}{s^2}\right)$$

over $\mathbb{R}^n$ with the center $\boldsymbol{c} \in \mathbb{R}^n$ and the scaling parameter $s > 0$. When $\boldsymbol{c}$ is the origin, we also write $\rho_s \coloneqq \rho_{s,\boldsymbol{o}}$; the same applies to all of the following definitions as well. When normalized over its measure $\int_{\boldsymbol{x}\in\mathbb{R}^n} \rho_{s,\boldsymbol{c}}(\boldsymbol{x})\, d\boldsymbol{x} = s^n$, $\rho_{s,\boldsymbol{c}}$ becomes the density function of the continuous Gaussian distribution with the mean $\boldsymbol{c}$ and variance $s\sqrt{n/2\pi}$ in n dimensions [59].

We *discretize* the Gaussian over a lattice $\Lambda(B)$ by limiting the support to only the lattice points $\boldsymbol{v} = B\boldsymbol{x}$ for $\boldsymbol{x} \in \mathbb{Z}^n$. Using a quadratic form Q to represent the lattice, we let

$$\rho_{Q,s,\boldsymbol{c}}(\boldsymbol{x}) \coloneqq \exp\left(-\pi \frac{\|\boldsymbol{x} - \boldsymbol{c}\|_Q^2}{s^2}\right)$$

be the Gaussian function in coefficient space, noting that $\rho_{Q,s,\boldsymbol{c}}(\boldsymbol{x}) = \rho_{s,B\boldsymbol{c}}(B\boldsymbol{x})$ for $Q = B^T B$. Using quadratic forms instead of lattice bases means that different orientations $O\Lambda$ of a lattice with $O \in O_n(\mathbb{R})$ and a basis OB induce the same $\rho_{Q,s,\boldsymbol{c}}$. However, using a different basis BU with $U \in \mathrm{GL}_n(\mathbb{Z})$ (so $R = U^T Q U$) does change the shape of the Gaussian since it affects the mapping of coefficients vectors to lattice vectors. We can now discretize and normalize $\rho_{Q,s,\boldsymbol{c}}$ over the support $\mathbb{Z}^n$ to get the lattice-dependent *discrete Gaussian distribution* in coefficient space:

Definition 2.23 (Discrete Gaussian distribution ([33]))
The discrete Gaussian distribution $\mathcal{D}_{Q,s,\boldsymbol{c}}$ (with $\mathcal{D}_{Q,s} \coloneqq \mathcal{D}_{Q,s,\boldsymbol{o}}$) for a quadratic form $Q \in S_n^{>0}(\mathbb{R})$, scaling parameter $s > 0$, and center $\boldsymbol{c} \in \mathbb{R}^n$ is the distribution with

$$\Pr_{X \sim \mathcal{D}_{Q,s,\boldsymbol{c}}}[X = \boldsymbol{x}] = \frac{\rho_{Q,s,\boldsymbol{c}}(\boldsymbol{x})}{\rho_{Q,s,\boldsymbol{c}}(\mathbb{Z}^n)} \text{ if } \boldsymbol{x} \in \mathbb{Z}^n, \text{ and } 0 \text{ otherwise.}$$

We require a technical lemma in the following:

Lemma 2.1 (adapted from [59, Lemma 2.9])**.** *For a quadratic form Q, a parameter $s > 0$, and a center $\boldsymbol{c} \in \mathbb{R}^n$, $\rho_{Q,s,\boldsymbol{c}}(\mathbb{Z}^n) \leq \rho_{Q,s}(\mathbb{Z}^n)$.*

The size of the parameter s in relation to the density of the lattice has a large impact on the behavior of $\mathcal{D}_{Q,s,\boldsymbol{c}}$: For very small s, the vast majority of the distribution's probability mass in concentrated in only a few points in $\mathbb{Z}^n$, and the distribution does not behave "like a Gaussian distribution". Micciancio and Regev [59] show that the requisite lower bound on s for the distribution to be well-behaved can be formally defined; it is called the *smoothing parameter*:

Definition 2.24 (Smoothing parameter ([33, Definition 2.4, 59, Definition 3.1]))
Given a quadratic form Q and an $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(Q)$ is the smallest parameter s with $\rho_{Q^{-1},1/s}(\mathbb{Z}^n \setminus \{0\}) \leq \epsilon$.

The smoothing parameter can be bounded both as a function of the first minimum and as a function of the length of the Gram-Schmidt vectors of the lattice basis.

Lemma 2.2 (adapted from [59, Lemma 3.2])**.** *For a quadratic form Q and $\epsilon = 2^{-n}$,*

$$\eta_\epsilon(Q) \leq \frac{\sqrt{n}}{\lambda_1(Q^{-1})}.$$

Lemma 2.3 (adapted from [38, Lemma 3.1])**.** *For a quadratic form Q with a Cholesky decomposition $Q = B_Q^T B_Q$ and $\epsilon > 0$,*

$$\eta_\epsilon(Q) \leq \left\|\widetilde{B}_Q\right\| \sqrt{\ln(2n(1+1/\epsilon))/\pi}.$$

For s above the smoothing parameter, the discretization of the distribution does not cause significant distortion, so the total probability mass is similar to that of the continuous distribution:

Lemma 2.4 (adapted from [82, Claim 3.8], compare [33, Lemma 2.5])**.** *Given a quadratic form Q, $\epsilon > 0$, a parameter $s \geq \eta_\epsilon(Q)$, and a center $\boldsymbol{c} \in \mathbb{R}^n$,*

$$\rho_{Q,s,\boldsymbol{c}} \in [1-\epsilon, 1+\epsilon] \cdot \frac{s^n}{\sqrt{\det(Q)}}.$$

Combining Lemma 2.1 and Lemma 2.4 leads to the following result, which is stated without a proof in the original work:

Lemma 2.5 (adapted from [21, Lemma 2.7])**.** *For a quadratic form Q, $\epsilon > 0$, a parameter $s \geq \eta_\epsilon(Q)$, and a center $\boldsymbol{c} \in \mathbb{R}^n$,*

$$\rho_{Q,s,\boldsymbol{c}}(\mathbb{Z}^n) \in \left[\frac{1-\epsilon}{1+\epsilon}, 1\right] \cdot \rho_{Q,s}(\mathbb{Z}^n).$$

The commonalities between the continuous and discrete Gaussian for s above a constant multiple of the smoothing parameter also extend to a tailbound and a minimum-entropy bound for the discrete Gaussian.

Lemma 2.6 (Tailbound (adapted from [59, Lemma 4.4]))**.** *Given a quadratic form Q, $\epsilon \in (0,1)$, a parameter $s \geq \eta_\epsilon(Q)$, and a center $\boldsymbol{c} \in \mathbb{R}^n$,*

$$\Pr_{\boldsymbol{x} \sim \mathcal{D}_{Q,s,\boldsymbol{c}}}\left[\|\boldsymbol{x}-\boldsymbol{c}\|_Q > s\sqrt{n}\right] \leq \frac{1+\epsilon}{1-\epsilon} \cdot 2^{-n}.$$

Lemma 2.7 (Minimum entropy (adapted from [75, Lemma 2.10], compare [33, Lemma 2.8]))**.** *For a quadratic form Q, $\epsilon > 0$, a parameter $s \geq 2\eta_\epsilon(Q)$, and a center $\boldsymbol{c} \in \mathbb{R}^n$, and any $\boldsymbol{x} \in \mathbb{Z}^n$,*

$$\Pr_{X \sim \mathcal{D}_{Q,s,\boldsymbol{c}}}[X = \boldsymbol{x}] \leq \frac{1+\epsilon}{1-\epsilon} \cdot 2^{-n}.$$

The discrete Gaussian distribution is also efficiently samplable by the following result, which we discuss and prove in detail in this thesis.

Theorem 2.2 (compare [21, Lemma 2.3, 33, Lemma 2.9]). *There is an efficient algorithm* SampleD$(Q, s, \boldsymbol{c})$ *that takes a quadratic form* Q*, a parameter* $s \geq \left\|\widetilde{B}_Q\right\| \sqrt{\ln(2n(n+2))/\pi}$,[4] *and a center* $\boldsymbol{c} \in \mathbb{R}^n$ *and outputs* $\boldsymbol{v} \sim \mathcal{D}_{Q,s,\boldsymbol{c}}$*.* SampleD *terminates in polynomial time with overwhelming probability.*

Note that the original statement of this result by Brakerski et al. [21, Lemma 2.3] contains a typo in the lower bound on s that has unfortunately been reproduced in several follow-up papers [1, 11, 15, 22, 23, 28, 33, 34, 51]. The NIST post-quantum signature candidate [3] HAWK [34] is not affected by this error because it uses a different sampling algorithm.

2.3.2 Lattice Isomorphisms

This section introduces lattice isomorphisms, the *lattice isomorphism problem* (LIP) in both worst- and average-case variants and the necessary elements of LIP cryptanalysis for this thesis. Many of the fundamentals concerning the use of LIP for cryptography were introduced by Ducas and van Woerden [33]. We accordingly source the contents of this section from the second author van Woerden's [93] dissertation except where noted differently.

Two n-dimensional lattices Λ and Λ' are *isomorphic* if they are related by an orthogonal transformation $\Lambda' = O\Lambda$ with $O \in O_n(\mathbb{R})$. This definition is motivated by the fact that Λ and Λ' have exactly the same geometry among their lattice points — they are simply rotated, reflected, or otherwise differently embedded copies of each other in $\mathbb{R}^n$. In its classic form, LIP asks one to find the isomorphism $O \in O_n(\mathbb{R})$ given the two lattices Λ and Λ'. This is believed to be computationally difficult; existing algorithms generally require finding short vectors in at least one of the lattices, their duals, or their hulls (i.e., approximating SVP). Accounting for the fact that lattices are typically represented by a basis B and that any BU with a unimodular U generates the same lattice as B, we define *search-LIP* (sLIP) in the *lattice formulation* as:

Definition 2.25 (sLIP^B — lattice formulation)
Let $B \in \mathbb{R}^{n\times m}$ be a basis of a lattice $\Lambda(B)$. Given a basis $B' \in \mathbb{R}^{n\times n}$ of a lattice $\Lambda(B')$ isomorphic to $\Lambda(B)$, find an orthogonal matrix $O \in O_n(\mathbb{R})$ and a unimodular matrix $U \in \mathrm{GL}_m(\mathbb{Z})$ such that $B' = OBU$.

Intuitively, the difficulty of the problem lies in the simultaneous action of both the orthogonal and the unimodular transformation: If one of these were known, the other could be efficiently determined via matrix inversion.

Recall that representing a full-rank lattice with a quadratic form $Q = B^T B$ loses precisely the orientation $O \in O_n(\mathbb{R})$ of the lattice, meaning that isomorphic lattices are represented by the same set of quadratic forms. In other words, we have a correspondence between $S_n^{>0}(\mathbb{R})$ and

[4] Despite the lower bound on s being identical to the upper bound on $\eta_{1/n+1}(Q)$ in Lemma 2.3, $s \geq \eta_{1/n+1}(Q)$ is not sufficient for this theorem.

the left quotient $O_n(\mathbb{R})\backslash \mathrm{GL}_n(\mathbb{R})$.[5] With quadratic forms, a change of basis by $U \in \mathrm{GL}_n(\mathbb{Z})$ from B to BU induces the change $Q' = U^T B^T B U = U^T Q U$. The positive definite symmetric matrices of the form $U^T Q U$ for a given $Q \in S_n^{>0}(\mathbb{R})$ therefore represent a full-rank lattice up to isomorphism just like the transformed bases OBU. We call $Q, Q' \in S_n^{>0}(\mathbb{R})$ *equivalent* if $Q' = U^T Q U$. The equivalence class of Q is accordingly defined as

$$[Q] := \left\{ U^T Q U \,\middle|\, U \in \mathrm{GL}_n(\mathbb{Z}) \right\}.$$

There can be multiple unimodular matrices that map Q to a particular $Q' \in [Q]$. The set of all such transformations is called the set of *isometries* between Q and Q': $\mathrm{Isom}(Q, Q') := \left\{ U \in \mathrm{GL}_n(\mathbb{Z}) \,\middle|\, Q' = U^T Q U \right\}$.[6] Next to the isometries to other lattices, a lattice also has automorphisms to itself. In the quadratic form representation, these make up the *automorphism group* $\mathrm{Aut}(Q) := \left\{ U \in \mathrm{GL}_n(\mathbb{Z}) \,\middle|\, U^T Q U = Q \right\}$, which is finite. For example, we have $\mathrm{Aut}(\mathbb{1}_n) = SP_n$ since any signed permutation of coordinates maps $\mathbb{Z}^n$ to itself. For $Q' = U^T Q U$, it follows that $\mathrm{Isom}(Q, Q') = \mathrm{Aut}(Q)U$. Using these definitions, we can construct sLIP for full-rank lattices in the *quadratic form formulation*:

Definition 2.26 (sLIP^Q — quadratic form formulation)
Let Q be a quadratic form. Given $Q' \in [Q]$, find a unimodular matrix $U \in \mathrm{GL}_n(\mathbb{Z})$ with $Q' = U^T Q U$.

The correspondence between bases and quadratic forms ensures that the lattice formulation and quadratic form formulation are equivalent: Let B and B' be bases of two isomorphic lattices of full rank with $Q = B^T B$ and $Q' = B'^T B'$. A solution U to sLIP^Q can be used to efficiently find a solution $(O = B' U^{-1} B^{-1}, U)$ to sLIP^B. Conversely, a solution (O, U) to sLIP^B contains a solution U to sLIP^Q. Using quadratic forms instead of lattice bases has the advantage of avoiding the need to handle real-valued orthogonal transformations — if we have $Q \in S_n^{>0}(\mathbb{Z})$, then $[Q] \subset S_n^{>0}(\mathbb{Z})$ as well, so sLIP^Q can be treated using precise integer and rational arithmetic.

We also define *distinguishing* variants of LIP where the goal is not to find a transformation between two representations, but to distinguish which of two equivalence classes a representation is from. Since the definitions for quadratic forms and lattice bases are analogous, we only present the former explicitly:

Definition 2.27 ($\Delta\mathrm{LIP}^{Q_0,Q_1}$ — quadratic form formulation)
Let Q_0, Q_1 be two quadratic forms. Find the bit $b \leftarrow\!\!\$\ \{0,1\}$ given $Q \in [Q_b]$.

ΔLIP is of interest in cryptography because it lets one take advantage of so-called "remarkably decodable" lattices in cryptographic constructions. The idea behind many of these is that Q_0 is the quadratic form to a canonical good basis of a lattice like $\mathbb{Z}^n$ that can be

[5] The backslash here does not represent the set difference, but the quotient of $\mathrm{GL}_n(\mathbb{R})$ over $O_n(\mathbb{R})$ where the orthogonal matrices act from the left.

[6] These are called isometries because $\langle U\boldsymbol{x}, U\boldsymbol{y} \rangle_Q = \langle \boldsymbol{x}, \boldsymbol{y} \rangle_{Q'}$ for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, so U^{-1} is an isometry from the geometry induced by Q to the one induced by Q'.

decoded efficiently [22, 33, 34] or that has a known short vector [51]. On the other hand, Q_1 is chosen such that decoding in a given radius is ambiguous or impossible. The good quadratic form Q_0 is hidden by a secret unimodular U in the bad public quadratic form $Q = U^T Q_0 U \in [Q_0]$. With knowledge of the secret U, coefficient vectors for Q can still be decoded efficiently. An adversary without U, on the other hand, cannot tell whether Q is in $[Q_0]$ or $[Q_1]$ if $\Delta\text{LIP}^{Q_0,Q_1}$ is hard. This means the public Q could be drawn from $[Q_1]$ instead of $[Q_0]$ in a security proof without the adversary noticing. Since Q_1 cannot be usefully decoded, then it follows that the adversary cannot usefully decode coefficient vectors for Q either.

The above LIP definitions are worst-case in that they place no constraints on how Q or Q' are chosen, but this choice has a significant impact on the difficulty of the problems. For instance, if U were chosen as $\mathbb{1}_n$, both sLIP and ΔLIP would become trivial. For LIP to be easily usable in a cryptographic context, a distribution for sampling $Q' \in [Q]$ is required. Ducas and van Woerden [33] define precisely such a distribution. It works by sampling random lattice vectors using a discrete Gaussian, then extracting a unimodular transformation from the samples. We replicate their result in brief here. It requires the following technical definition:

Definition 2.28 (Hermite Normal Form (HNF) ([93, Definition 167]))
For an integer matrix $M \in \mathbb{Z}^{n\times m}$ with $\operatorname{rank}(M) = r \leq n$, the HNF of M is the unique $T \in \mathbb{Z}^{n\times m}$ with entries $t_{i,j}$ such that $M = UT$ with $U \in \mathrm{GL}_n(\mathbb{Z})$, T is upper-diagonal, and

- exactly the first r rows of T are nonzero,
- the first nonzero entry t_{i,j_i} in each row i (the *pivot* in the *pivot column* $\boldsymbol{t}_{j_i}$) is to the left of any nonzero entries in the rows below it (so $j_1 < \cdots < j_r$),
- the pivots are positive ($t_{i,j_i} > 0$),
- and the pivot columns are non-negative ($\boldsymbol{t}_{j_i} \in \mathbb{N}_0^n$).

Micciancio and Warinschi [61] show that the HNF can be computed efficiently.

Lemma 2.8 (adapted from [33, Lemma 3.4])**.** *Let $Q \in S_n^{>0}(\mathbb{Z})$ be a quadratic form and*

$$s \geq \max\left\{\lambda_n(Q),\ \left\|\widetilde{B}_Q\right\|\sqrt{\ln(2n(n+2))/\pi}\right\}$$

be a scaling parameter.[7] *$\textsc{SampleForm}(Q, s)$ as defined in Algorithm 2.1 terminates within polynomial time with overwhelming probability. It outputs $R \in S_n^{>0}(\mathbb{Z})$ and $U \in GL_n(\mathbb{Z})$ such that $R = U^T QU \in [Q]$, with U being uniform over $Isom(Q, R)$ conditioned on the output R. The distribution of R as output by $\textsc{SampleForm}(Q, s)$ is called $\mathcal{D}_s([Q])$. It depends only on the equivalence class $[Q]$, not on the specific representative Q.*

Since we use both the quadratic form and the lattice formulation in this work, we add that one can construct a similar distribution in the lattice formulation: Given a basis B

[7] As in Theorem 2.2, the lower bound on s is corrected from the original.

Algorithm 2.1 Sampling procedure for $\mathcal{D}_s([Q])$ (adapted from [33])

function SampleForm$(Q, s) \to (R \sim \mathcal{D}_s([Q]),\ U)$
  $C \leftarrow 1 - (1 + e^{-\pi})^{-1}$
  $m \leftarrow \left\lceil \frac{2n}{C} \right\rceil$
  repeat
    $(\boldsymbol{y}_i)_{i \in [m]} \leftarrow (\mathcal{D}_{Q,s})_{i \in [m]}$
    $Y \leftarrow (\boldsymbol{y}_1 \cdots \boldsymbol{y}_m)$
  until $\operatorname{rank}(Y) = n$
  $T \leftarrow$ HNF of Y with $Y = U^{-1}T,\ U \in \mathrm{GL}_n(\mathbb{Z})$
  return $(U^T Q U, U)$

and a parameter s, first find a unimodular U using SampleForm$(B^T B, s)$. This leaves only the sampling of a random orthogonal matrix. Seeing as the group $O_n(\mathbb{R})$ is compact, the distribution $\mathcal{U}(O_n(\mathbb{R}))$ is well-defined; Mezzadri [55] shows that it is efficiently samplable as well.[8] We can therefore take $O \leftarrow \mathcal{U}(O_n(\mathbb{R}))$ and output a basis $B' = OBU$ with analogous properties to $Q' = U^T B^T B U$.

Ducas and van Woerden [33] establish the following bound on the length of the Gram-Schmidt vectors for $B_{Q'}$ with $Q' \sim \mathcal{D}_s([Q])$, which analogously applies to our distribution in the lattice formulation:

Lemma 2.9 (adapted from [33]). *For $Q \in S_n^{>0}(\mathbb{Z})$, $\epsilon \in (0, 1)$, and*

$$s \geq \max\left\{\eta_\epsilon(Q),\ \lambda_n(Q),\ \left\|\widetilde{B}_Q\right\| \sqrt{\ln(2n(n+2))/\pi}\right\},$$

we have

$$\Pr_{Q' \sim \mathcal{D}_s([Q])}\left[\left\|\widetilde{B}_{Q'}\right\| > s\sqrt{n}\right] \leq \frac{1+\epsilon}{1-\epsilon} \cdot 100n \cdot 2^{-n}.$$

Ducas and van Woerden [33] define *average-case* variants of sLIP^Q and ΔLIP^{Q_0,Q_1} based on the distribution $\mathcal{D}_s([Q_{(b)}])$ and show that both search- and distinguish-LIP admit worst-case to average-case reductions for large s. This makes it possible to build cryptographic schemes from the average-case variants using the efficiently samplable distribution $\mathcal{D}_s$ with the confidence that these instances are likely not significantly easier than the hardest instances of LIP. We present only the average-case variant ac-ΔLIP here and refer to Ducas and van Woerden [33] for details on the other variants and the worst-to-average-case reduction. We similarly omit the definition in the lattice formulation as it is entirely analogous.

Definition 2.29 (ac-ΔLIP$_s^{Q_0,Q_1}$ — quadratic form formulation ([33]))
Let $Q_0, Q_1 \in S_n^{>0}(\mathbb{Z})$ be two quadratic forms and $s > 0$ be the scaling parameter. Find the bit $b \hookleftarrow \$ \{0, 1\}$ given $Q \leftarrow \mathcal{D}_s([Q_b])$.

[8] We ignore the precision issues involved in representing real-valued numbers here since they are inherent to the lattice formulation of LIP.

With the problem of interest defined, we move to explain the necessary notions concerning the hardness of $\Delta\text{LIP}^{Q_0,Q_1}$ as a function of Q_0 and Q_1. The first of these notions is that of *invariants* within an equivalence class $[Q]$. These are functions of quadratic forms that are constant on $[Q]$. If an efficiently computable invariant I differed between $[Q_0]$ and $[Q_1]$, $\Delta\text{LIP}^{Q_0,Q_1}$ could be easily solved by checking whether $I(Q) = I(Q_0)$ or $I(Q) = I(Q_1)$. van Woerden [93] categorizes invariants into *arithmetic* and *geometric* invariants.

Arithmetic invariants are generally efficiently computable and include the determinant $\det(Q)$ as well as the greatest common divisor $\gcd(Q)$ of the entries of Q. These invariants are eclipsed by the equivalence over larger rings: For a ring R, let $[Q]_R := \left\{U^T Q U \,\middle|\, U \in \mathrm{GL}_n(R)\right\}$. If $R \supset \mathbb{Z}$, it also follows that $[Q]_R \supset [Q]$. The equivalence classes $[Q]_{\mathbb{P}_p}$ over the p-adic integers for all primes p combine to form the *genus* of Q, which covers all known efficiently computable arithmetic invariants.[9] The genus of Q_0 must be equal to that of Q_1 for $\Delta\text{LIP}^{Q_0,Q_1}$ to be hard.

Geometric invariants include the first minimum $\lambda_1(Q)$ and the size $|\mathrm{Aut}(Q)|$ of the automorphism group. Since determining these using existing algorithms requires finding short vectors in the lattices, they are not believed to be efficiently computable. Therefore, we do not require them to be equal for Q_0 and Q_1 used to instantiate $\Delta\text{LIP}^{Q_0,Q_1}$.

Assuming that the genus of Q_0 and Q_1 match, Ducas and Gibbons [32] conjecture that solving the worst-case variant $\Delta\text{LIP}^{Q_0,Q_1}$ requires finding short vectors in at least one of the lattices, their duals, or their s-hulls. In practice, existing attacks solve f-SVP in one of these lattices for $f = \mathrm{gh}(Q')/\lambda_1(Q')$ where Q' is the quadratic form of a lattice, dual, or s-hull. The difficulty of the problem is thus closely related to how "unusually small" $\lambda_1(Q')$ is for these lattices — the greater the gap between the expected length $\mathrm{gh}(Q')$ of the shortest vector and the actual length $\lambda_1(Q')$, the easier f-SVP heuristically becomes. With the definition of the primal-dual gap

$$\mathrm{gap}(Q) := \max\left\{\frac{\mathrm{gh}(Q)}{\lambda_1(Q)}, \frac{\mathrm{gh}(Q^{-1})}{\lambda_1(Q^{-1})}\right\}$$

and the hull gap

$$\mathrm{hullgap}(Q) := \max_{s\in\mathbb{R}\setminus\{0\}} \mathrm{gap}(H_s(\Lambda(B))) \quad \text{for } Q = B^T B,$$

a heuristic measure of the hardness of $\Delta\text{LIP}^{Q_0,Q_1}$ is therefore given by the maximum of the hull gaps of Q_0 and Q_1.[10]

Conjecture 1 (Hardness of ΔLIP [32])**.** *For quadratic forms $Q_0, Q_1 \in S_n^{>0}(\mathbb{Z})$ with equal genera and $1 \le \mathrm{hullgap}(Q_i) \le f$, the best attack on $\Delta LIP^{Q_0,Q_1}$ solves f-SVP for Q_0 or Q_1.*

[9] While Ling, Liu, and Mendelsohn [52] study the use of the more specific spinor genus of Q, they find that it is rarely more specific than the standard genus and cannot currently be computed efficiently for most quadratic forms.

[10] Note that $H_1(\Lambda(B)) = \Lambda(B)$ for $B \in \mathbb{Z}^{n\times n}$, so the hull gap is no less than the primal-dual gap of the lattice itself for the classes of lattices commonly used [32].

2.3.3 Codes and Learning with Errors

In this section, we give an overview of linear codes and the *Learning with Errors* (LWE) problem as required in this thesis. The definitions here are deliberately brief; we refer the reader to the standard literature on these topics [26, 82] for a thorough introduction.

Linear codes are closely related to lattices in that both are discrete additive subspaces embedded in a larger space. For lattices, this space is usually $\mathbb{R}^n$, while linear codes exist in $\mathbb{Z}_q^n$ for prime q.

Definition 2.30 (Linear code [26])
A linear $[n, l]_q$-code C with $l \leq n$ and prime q is a dimension-l subspace of the vector space $\mathbb{Z}_q^n$. A matrix $G \in \mathbb{Z}_q^{n \times l}$ with $C = G\mathbb{Z}_q^l$ is called a *generator matrix* of C. A generator matrix G is in *systematic form* if

$$G = \begin{pmatrix} \mathbb{1}_l \\ A \end{pmatrix}$$

for $A \in \mathbb{Z}_q^{(n-l)\times l}$. Every code has a generator matrix in systematic form up to coordinate permutation.

A class of lattices that mirrors the structure of linear codes is the class of q-ary lattices. Following Costa et al. [26, Definition 3.1] A q-ary lattice Λ has $q\mathbb{Z}^n \subset \Lambda \subset \mathbb{Z}^n$, so the lattice is integral and repeats across a tiling of $\mathbb{R}^n$ into a grid of hypercubes $q\boldsymbol{x} + [0, q]^n$ for $\boldsymbol{x} \in \mathbb{Z}^n$. The q-ary lattices are precisely those that can be constructed from linear codes over $\mathbb{Z}_q^n$ via *Construction A*:

Definition 2.31 (Construction A lattices (adapted from [26]))
Given a linear $[n, l]_q$-code C, the Construction A lattice Λ_C is defined as

$$\Lambda_C \coloneqq \bigcup_{\boldsymbol{x} \in \mathbb{Z}^n} q\boldsymbol{x} + C,$$

where the addition is performed in $\mathbb{Z}^n$. If C has a generator matrix G in systematic form with a matrix $A \in \mathbb{Z}_q^{(n-l)\times l}$, a basis of Λ_C is given by

$$\begin{pmatrix} \mathbb{1}_l & 0 \\ A & q\mathbb{1}_{n-l} \end{pmatrix}.$$

LWE is a lattice problem that also makes use of $\mathbb{Z}_q^n$ in its average-case variant. It is assumed to be hard and admits a worst-to-average-case reduction similarly to LIP. LWE is well-studied and widely used in lattice cryptography [19, 58, 74, 76, 82, 83, 94]. We repeat only the definition of its decisional variant here: It asks that, given a random matrix A, one distinguish a system $A\boldsymbol{s} + \boldsymbol{e}$ of random linear equations with a secret vector $\boldsymbol{s}$ and added noise $\boldsymbol{e}$ from uniform randomness.

Definition 2.32 (Decisional LWE (notation adapted from [82]))
For a prime q, an $n \in \Theta(\lambda)$, and $m \in \text{poly}(n)$, sample uniformly random $A \hookleftarrow\$ \mathbb{Z}_q^{m \times n}$, $\boldsymbol{s} \hookleftarrow\$ \mathbb{Z}_q^n$, and $\boldsymbol{r} \hookleftarrow\$ \mathbb{Z}_q^m$. Let χ be an error distribution on $\mathbb{Z}_q$, from which we sample m errors $e_1, \ldots, e_m \hookleftarrow\$ \chi$ to form an error vector $\boldsymbol{e} = (e_1 \cdots e_m)^T \in \mathbb{Z}_q^m$. Set $W_0 = (A, A\boldsymbol{s} + \boldsymbol{e})$ and $W_1 = (A, \boldsymbol{r})$. The decisional LWE problem is to recover the bit $b \hookleftarrow\$ \{0, 1\}$ given W_b.

Micciancio [56] shows that the definition of the LWE problem can be extended to arbitrary lattices instead of the compact space $\mathbb{Z}_q^n$, but there is no average-case version of this extension in the literature.

3 Discrete Gaussian Sampler

As part of our investigation into LIP-based cryptography, we implemented a sampling algorithm for the discrete Gaussian distribution $\mathcal{D}_{Q,s,\boldsymbol{c}}$ on quadratic forms Q (see Definition 2.23) for the purposes of both closer analysis and empirical evaluation of potential constructions. Specifically, we aimed to write an exact sampler according to Theorem 2.2, which is the definition used in many of the existing LIP-based schemes [22, 28, 33, 51]. Specifically, the definition demands an exact sampler; In addition, the sampled discrete Gaussian $\mathcal{D}_{Q,s,\boldsymbol{c}}$ is defined in terms of quadratic forms and has $\operatorname{supp}(\mathcal{D}_{Q,s,\boldsymbol{c}}) = \mathbb{Z}^n$. However, we are not aware of any such sampler existing in the literature. While Ducas and van Woerden [33] claim that Brakerski et al. [21] prove the existence of such a sampling algorithm, the sampler by Brakerski et al. actually samples a closely related, but different distribution with a different support. The sampler presented by Gentry, Peikert, and Vaikuntanathan [38] similarly did not meet our requirements because it is not exact. We therefore synthesized a sampler that precisely satisfies Theorem 2.2 based on the existing samplers and describe it in this section. Our sampler also benefits from increased numerical stability using a technique from another sampler implementation by de Castro Biage [28]. We both prove the correctness of our sampler's description and evaluate its implementation on lattices with up to two dimensions. Our work constitutes, to the best of our knowledge, the first complete description, proof and implementation of an exact sampler for the discrete Gaussian distribution on quadratic forms in the literature.

Let us first examine the existing samplers in the literature to see how they differ from Theorem 2.2: First, Gentry, Peikert, and Vaikuntanathan's [38] sampler is designed for the lattice formulation, so it expects a lattice basis B as input instead of a quadratic form Q and outputs lattice vectors $\boldsymbol{b} \in \Lambda(B)$. These issues could be circumvented by taking the Cholesky decomposition $B^T B = Q$, passing B to the sampler, and finally multiplying the sampler's output by B^{-1} to get a corresponding coefficient vector. However, taking the Cholesky decomposition of quadratic forms sampled from $\mathcal{D}_s([Q])$ is numerically ill-conditioned and slow to compute[1] while also introducing imprecisions due to the real numbers in the resulting basis B. In addition, their algorithm outputs samples according to a distribution that is not exactly $\mathcal{D}_{B^T B,s,\boldsymbol{c}}$, but statistically close to it for negligible ϵ.[2]

Brakerski et al.'s [21] sampler outputs samples that match its theoretical distribution exactly for $\epsilon \leq \frac{1}{n+1}$, but also expects a lattice basis B as input. Most notably, however, their

[1] In particular, the Python library NumPy fails to find decompositions for $P \leftarrow \mathcal{D}_s([Q])$ with the parameters $Q = \mathbb{1}_n$, $n = 40$, and $s \gtrapprox 1.3$. The library SageMath successfully decomposes significantly larger quadratic forms ($n > 200$, $s > 300$), but requires multiple seconds to do so on our hardware.

[2] Here, ϵ is the parameter of the smoothing parameter $n_\epsilon(Q)$.

algorithm samples the distribution $\mathcal{D}_{\Lambda(B)+\boldsymbol{c},s}$ instead of $\mathcal{D}_{B^T B,s,\boldsymbol{c}}$. These distributions are similar, but differ in their use of the $\boldsymbol{c}$ parameter: $\operatorname{supp}(\mathcal{D}_{B^T B,s,\boldsymbol{c}}) = \Lambda(B)$ (or $\mathbb{Z}^n$ in the quadratic form formulation) with the probability of each lattice point being weighted by a Gaussian with its center in $\boldsymbol{c}$. On the other hand, $\operatorname{supp}(\mathcal{D}_{\Lambda(B)+\boldsymbol{c},s}) = \Lambda(B) + \boldsymbol{c}$ with the probability of each point in the coset $\Lambda(B) + \boldsymbol{c}$ weighted by a Gaussian with its center in the origin. These two distributions are equivalent in the sense that $\mathcal{D}_{B^T B,s,\boldsymbol{c}}$ can be sampled by taking $\boldsymbol{x}' \leftarrow \mathcal{D}_{\Lambda(B)-\boldsymbol{c},s}$ and outputting $\boldsymbol{x} = \boldsymbol{x}' + \boldsymbol{c}$, so an exact sampler for $\mathcal{D}_{B^T B,s,\boldsymbol{c}}$ could be implemented in terms of Brakerski et al.'s algorithm. Sampling in lattice cosets, however, makes their algorithm more complex. The additional layer of indirection would also hinder our analysis of the algorithm in this work. We therefore decide to recombine both existing samplers into an exact sampler for $\mathcal{D}_{Q,s,\boldsymbol{c}}$ in the quadratic form formulation.

Like the existing samplers, our sampling algorithm is based on the separability of Gaussian functions. To produce an n-dimensional sample from $\mathcal{D}_{Q,s,\boldsymbol{c}}$, we sample n one-dimensional discrete Gaussians $\mathcal{D}_{\mathbb{1},s_i',c_i'}$ for $i \in [n]$ with parameters s_i', c_i' depending on the lattice geometry and combine them into one coefficient vector. The sampling algorithm is similarly decomposed into a procedure for sampling from the one-dimensional $\mathcal{D}_{\mathbb{1},s,c}$ distribution and a second procedure that uses the first as a subroutine to produce the final output. We discuss both procedures in order in Section 3.1 and Section 3.2. Throughout this section, we use the shorthand $\rho_{s,c}(x) \coloneqq \rho_{\mathbb{1},s,c}(x) = \exp(-\pi(x-c)^2/s^2)$ to refer to the one-dimensional Gaussian function on $\mathbb{Z}$ with parameters $c, s \in \mathbb{R}$.

3.1 One-Dimensional Sampler

The one-dimensional sampler $\textsc{Sample1D}(s, c)$ is adapted from Brakerski et al. [21]. Recall that this procedure's goal is to output $z \in \mathbb{Z}$ with

$$\Pr[z \leftarrow \textsc{Sample1D}(s,c)] = \frac{\rho_{s,c}(z)}{\rho_{s,c}(\mathbb{Z})} \text{ if } z \in \mathbb{Z}, \text{ else } 0.$$

By taking $c' \leftarrow c \bmod 1$, we always have $c' \in [0,1)$ and can simply add the integral part of c back to the resulting sample before returning it. To generate a sample from the distribution given $0 \le c' < 1$, we split the support $\mathbb{Z}$ into four pieces: $\{0\}$, $\{1\}$, $(-\infty, 0) \cap \mathbb{Z}$, and $(1, \infty) \cap \mathbb{Z}$. The algorithm chooses one of these pieces at random with weights roughly corresponding to their probability masses. The singletons $\{0\}$ and $\{1\}$ are chosen with weights $\rho_{s,c'}(0)$ and $\rho_{s,c'}(1)$ respectively, while the weights of the tails are $Z_0 \leftarrow \int_{-\infty}^{0} \rho_{s,c'}(x)\,dx$ and $Z_1 \leftarrow \int_{1}^{\infty} \rho_{s,c'}(x)\,dx$ (see Figure 3.1 for an illustration). These integrals can be evaluated efficiently using the error function by the equalities

$$Z_0 = \int_{-\infty}^{0} \rho_{s,c'}(x)\,dx = \int_{-\infty}^{0} e^{-\pi\frac{(x-c')^2}{s^2}}\,dx = \int_{-\infty}^{0} e^{-\left(\frac{\sqrt{\pi}(x-c')}{s}\right)^2}\,dx = \frac{s}{\sqrt{\pi}} \int_{-\infty}^{\frac{-c'\sqrt{\pi}}{s}} e^{-y^2}\,dy,$$

where the fourth equality substitutes $y = \sqrt{\pi}(x-c')/s$ for x with $dx = \left(s/\sqrt{\pi}\right) dy$.

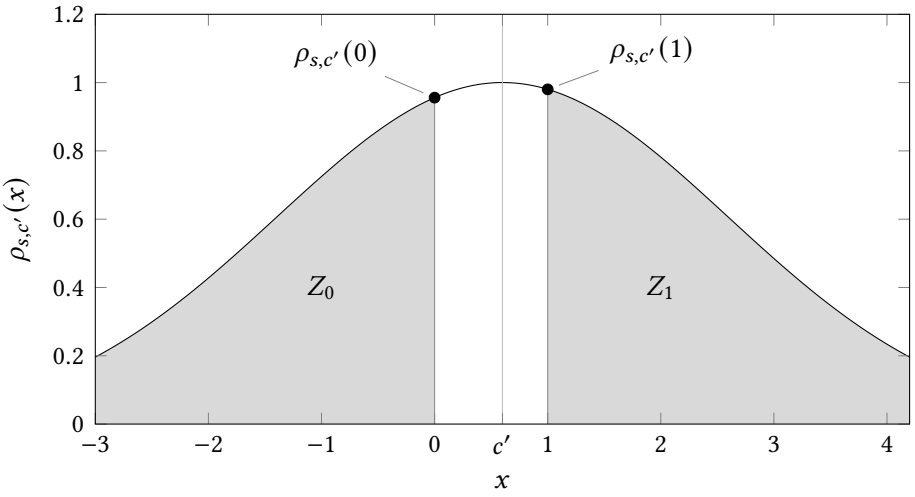


Figure 3.1: Example of the Gaussian function $\rho_{s,c'}(x)$ for $s = 5$ and $c' = 0.6$. The weights of the decomposition of $\mathbb{Z}$ are also shown.

Using the well-known identity $\int_{-\infty}^{\infty} e^{-y^2}\,dy = \sqrt{\pi}$ and the definition of the error function $\operatorname{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-y^2}\,dy$ along with its complement $\operatorname{erfc}(z) = 1 - \operatorname{erf}(z)$ [7], we have

$$\begin{aligned}
\frac{s}{\sqrt{\pi}} \int_{-\infty}^{\frac{-c'\sqrt{\pi}}{s}} e^{-y^2}\,dy &= \frac{s}{\sqrt{\pi}} \left(\int_{-\infty}^{0} e^{-y^2}\,dy - \int_{\frac{-c'\sqrt{\pi}}{s}}^{0} e^{-y^2}\,dy \right) \\
&= \frac{s}{2} \left(1 - \operatorname{erf}\left(\frac{c'\sqrt{\pi}}{s} \right) \right) \\
&= \frac{s}{2} \operatorname{erfc}\left(\frac{c'\sqrt{\pi}}{s} \right).
\end{aligned}$$

An analogous calculation shows that $Z_1 = \frac{s}{2}\operatorname{erfc}\left((1-c')\sqrt{\pi}/s\right)$. We let $Z \leftarrow \rho_{s,c'}(0) + \rho_{s,c'}(1) + Z_0 + Z_1$ to normalize the weights into probabilities. After a piece is chosen, the algorithm samples from that piece. If $\{0\}$ or $\{1\}$ are chosen, the respective integer is output. Conversely, if one of the tails is chosen, a sample x is drawn from the continuous truncated normal distribution $\mathrm{TN}_{a,b}(c', \frac{s^2}{2\pi})$ on that tail[3] and rounded away from 0 to produce an integer. An X that has the truncated normal distribution $\mathrm{TN}_{a,b}(\mu, \sigma^2)$ follows the normal distribution $\mathcal{N}(\mu, \sigma^2)$ conditioned on $a \le X \le b$. Essentially, this means the normal distribution is truncated to the interval $[a, b]$ and renormalized accordingly [18]. The rounded samples $\lfloor x \rfloor$ or $\lceil x \rceil$ are output with a probability of $\frac{\rho_{s,c'}(\lfloor x \rfloor)}{\rho_{s,c'}(x)}$ or $\frac{\rho_{s,c'}(\lceil x \rceil)}{\rho_{s,c'}(x)}$ respectively. These probabilities are always less than 1 due to the monotonicity of $\rho_{s,c'}(x)$ on each tail. If the sample is rejected, the algorithm loops back to choose a piece of the support again, repeating this process until it produces output. The full algorithm is presented in pseudocode in Algorithm 3.1.

[3] The variance is scaled by 2π to account for the difference between ρ_s and the standard Gaussian function.

Algorithm 3.1 Exact sampling procedure for $\mathcal{D}_{\mathbb{1},s,c}$ (adapted from [21])

function SAMPLE1D$(s, c) \to z \sim \mathcal{D}_{\mathbb{1},s,c}$
  $c' \leftarrow c \bmod 1$
  $Z_0 \leftarrow \int_{-\infty}^{0} \rho_{s,c'}(x)\,dx = \frac{s}{2}\mathrm{erfc}\left(\frac{c'\sqrt{\pi}}{s}\right)$
  $Z_1 \leftarrow \int_{1}^{\infty} \rho_{s,c'}(x)\,dx = \frac{s}{2}\mathrm{erfc}\left(\frac{(1-c')\sqrt{\pi}}{s}\right)$
  $Z \leftarrow \rho_{s,c'}(0) + \rho_{s,c'}(1) + Z_0 + Z_1$
  $z \leftarrow \perp$
  while $z = \perp$ **do**
    $\begin{cases} z \leftarrow 0 & \text{with prob. } \frac{\rho_{s,c'}(0)}{Z} \\ z \leftarrow 1 & \text{with prob. } \frac{\rho_{s,c'}(1)}{Z} \\ x \leftarrow \mathrm{TN}_{-\infty,0}\left(c', \frac{s^2}{2\pi}\right),\ z \leftarrow \lfloor x \rfloor \text{ with prob. } \frac{\rho_{s,c'}(\lfloor x \rfloor)}{\rho_{s,c'}(x)} & \text{with prob. } \frac{Z_0}{Z} \\ x \leftarrow \mathrm{TN}_{1,\infty}\left(c', \frac{s^2}{2\pi}\right),\ z \leftarrow \lceil x \rceil \text{ with prob. } \frac{\rho_{s,c'}(\lceil x \rceil)}{\rho_{s,c'}(x)} & \text{with prob. } \frac{Z_1}{Z} \end{cases}$
  return $z + \lfloor c \rfloor$

Lemma 3.1 (adapted from [21]). *For any $s, c \in \mathbb{R}$, SAMPLE1D(s, c) terminates in linear time with overwhelming probability and outputs $z \sim \mathcal{D}_{\mathbb{1},s,c}$.*

Proof (adapted from [21]). To see that the sampler has the correct output distribution, consider the probability of outputting a specific $z \in \mathbb{Z}$ in any given iteration: The probabilities of outputting $0 + \lfloor c \rfloor$ or $1 + \lfloor c \rfloor$ are exactly $\rho_{s,c'}(0)/Z$ and $\rho_{s,c'}(1)/Z$, while the probability that a specific $z + \lfloor c \rfloor$ from the left tail is produced as output is

$$\frac{Z_0}{Z}\frac{1}{Z_0}\int_{z}^{z+1} \rho_{s,c'}(x)\frac{\rho_{s,c'}(z)}{\rho_{s,c'}(x)}\,dx \;=\; \frac{1}{Z}\int_{z}^{z+1} \rho_{s,c'}(z)\,dx \;=\; \frac{\rho_{s,c'}(z)}{Z}$$

and similarly for the right tail. It follows that the probability mass function of the outputs is proportional to that of $\mathcal{D}_{\mathbb{1},s,c}$. Therefore, conditioned on the algorithm producing output in an iteration, the output z indeed follows the required discrete Gaussian $\mathcal{D}_{\mathbb{1},s,c}$.

The probability of the algorithm terminating after an iteration is the same as that of it producing output, which is $\rho_{s,c'}(\mathbb{Z})/Z$ as shown above. We also have

$$Z_0 = \int_{-\infty}^{0} \rho_{s,c'}(x)\,dx \;=\; \sum_{k=0}^{\infty}\int_{-k-1}^{-k} \rho_{s,c'}(x)\,dx \;\le\; \sum_{k=0}^{\infty} \rho_{s,c'}(-k) \;=\; \rho_{s,c'}((-\infty, 0] \cap \mathbb{Z}),$$

where the inequality is due to $\rho_{s,c'}(x)$ being monotonically increasing for $x < c'$ and the length of each interval being 1. Through an analogous argument for Z_1, we see that $Z_0 + Z_1 \le \rho_{s,c'}(\mathbb{Z})$. Finally,

$$\frac{\rho_{s,c'}(\mathbb{Z})}{Z} = \frac{\rho_{s,c'}(\mathbb{Z})}{Z_0 + Z_1 + \rho_{s,c'}(0) + \rho_{s,c'}(1)} \ge \frac{\rho_{s,c'}(\mathbb{Z})}{\rho_{s,c'}(\mathbb{Z}) + \rho_{s,c'}(0) + \rho_{s,c'}(1)} \ge \frac{\rho_{s,c'}(\mathbb{Z})}{2\rho_{s,c'}(\mathbb{Z})} = \frac{1}{2},$$

so the algorithm has at least a $\frac{1}{2}$ chance of terminating after every iteration and a 2^{-m} chance of running for more than m iterations. □

3.2 n-Dimensional Sampler

With a one-dimensional sampler for $\mathcal{D}_{\mathbb{1},s,c}$ in hand, we move on to defining an exact sampling procedure SampleD for $\mathcal{D}_{Q,s,\boldsymbol{c}}$ given any n-dimensional quadratic form Q, a parameter s, and a center $\boldsymbol{c}$. We adapt Gentry, Peikert, and Vaikuntanathan's [38] algorithm for lattice bases B to quadratic forms while including a rejection step inspired by Brakerski et al.'s [21] sampler. The rejection step allows us to achieve an exact correspondence with $\mathcal{D}_{Q,s,\boldsymbol{c}}$ instead of only statistical closeness. Gentry et al.'s algorithm is essentially a randomized version of Babai's nearest planes algorithm [9] for approximating CVP[4] and was proposed in a similar form by Klein [49] as a Monte Carlo algorithm for CVP as well. We begin by describing Gentry et al.'s algorithm in the lattice formulation given a basis B with basis vectors $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ before explaining how it can be adapted to quadratic forms Q and made exact with rejection sampling. Finally, we prove its correctness in the quadratic form formulation.

As mentioned previously, SampleD$(B, s, \boldsymbol{c})$ is based on the separability of Gaussian functions, constructing an n-dimensional sample $\boldsymbol{v}$ from n one-dimensional samples z_i for all $i \in [n]$ drawn using Sample1D. Naively, one may try to sample "along each basis vector" by projecting $\boldsymbol{c}$ and scaling the parameter s for each basis vector in turn. To this end, let $c_i = \langle \boldsymbol{c}, \boldsymbol{b}_i \rangle / \langle \boldsymbol{b}_i, \boldsymbol{b}_i \rangle$ be the length of the projection of $\boldsymbol{c}$ onto $\boldsymbol{b}_i$ and $s_i = s/\|\boldsymbol{b}_i\|$ be the i-th parameter both in units of $\|\boldsymbol{b}_i\|$. Now, an integer $z_i \leftarrow$ Sample1D(s_i, c_i) corresponds exactly to a multiple $z_i \boldsymbol{b}_i$ of the basis vector sampled discretely around the center $c_i \boldsymbol{b}_i$ with a parameter of s. However, calculating $\boldsymbol{v} \leftarrow \sum_{i=1}^{n} z_i \boldsymbol{b}_i$ like this would not actually produce a $\boldsymbol{v} \sim \mathcal{D}_{B^T B,s,\boldsymbol{c}}$ since the lattice basis vectors $\boldsymbol{b}_i$ are not orthogonal. We can instead use the Gram-Schmidt orthogonalization $(\tilde{\boldsymbol{b}}_1, \ldots, \tilde{\boldsymbol{b}}_n)$ of the basis as a set of orthogonal basis vectors for the algorithm. In the new basis, $c'_i = \langle \boldsymbol{c}, \tilde{\boldsymbol{b}}_i \rangle / \langle \tilde{\boldsymbol{b}}_i, \tilde{\boldsymbol{b}}_i \rangle$ and $s'_i = s/\|\tilde{\boldsymbol{b}}_i\|$, now in units of $\|\tilde{\boldsymbol{b}}_i\|$. Naturally, we cannot use the Gram-Schmidt-vectors $\tilde{\boldsymbol{b}}_i$ as a basis for the lattice $\Lambda(B)$, so we still set $\boldsymbol{v} \leftarrow \sum_{i=1}^{n} z_i \boldsymbol{b}_i$. The distortion $\boldsymbol{d}_i = z_i(\boldsymbol{b}_i - \tilde{\boldsymbol{b}}_i)$ caused by this can be compensated by drawing the one-dimensional samples z_i in reverse order from n to 1 and shifting the center $\boldsymbol{c}$ after every sample: $\boldsymbol{c}_{i-1} \leftarrow \boldsymbol{c}_i - z_i \boldsymbol{b}_i$, where $\boldsymbol{c}_i$ is the center used to draw z_i for all $i \in [n]$ and $\boldsymbol{c}_n \leftarrow \boldsymbol{c}$. To see why this undoes the distortion, consider the definition of the Gram-Schmidt-orthogonalization:

$$\tilde{\boldsymbol{b}}_i = \boldsymbol{b}_i - \sum_{j=0}^{i-1} \frac{\langle \boldsymbol{b}_i, \tilde{\boldsymbol{b}}_j \rangle}{\langle \tilde{\boldsymbol{b}}_j, \tilde{\boldsymbol{b}}_j \rangle} \tilde{\boldsymbol{b}}_j,$$

so, using a representation by Regev [78], the lattice basis B and center $\boldsymbol{c}$ can be written in the basis $\widetilde{B}$ as

$$\begin{pmatrix} 1 & \cdots & * \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{pmatrix} \text{ and } \begin{pmatrix} \frac{\langle \boldsymbol{c}, \tilde{\boldsymbol{b}}_1 \rangle}{\langle \tilde{\boldsymbol{b}}_1, \tilde{\boldsymbol{b}}_1 \rangle} \\ \vdots \\ \frac{\langle \boldsymbol{c}, \tilde{\boldsymbol{b}}_n \rangle}{\langle \tilde{\boldsymbol{b}}_n, \tilde{\boldsymbol{b}}_n \rangle} \end{pmatrix}. \tag{3.1}$$

[4] Note that while Babai's algorithm requires an LLL-reduced basis as input for its approximation guarantees [9], SampleD works with any quadratic form as long as $s \geq \eta_\epsilon(Q)$ for $\epsilon \leq \frac{1}{n+1}$.

When adding $z_i \boldsymbol{b}_i$ to the final result $\boldsymbol{v}$, the result is shifted by a multiple of the i-th column of the matrix in (3.1). The distortion $\boldsymbol{d}_i$ caused by adding $z_i \boldsymbol{b}_i$ instead of $z_i \tilde{\boldsymbol{b}}_i$ is thus z_i times the above-diagonal entries of the i-th column. We therefore have $z_i \boldsymbol{d}_i \in \operatorname{span}(\tilde{\boldsymbol{b}}_1, \ldots, \tilde{\boldsymbol{b}}_{i-1})$ and $\pi_{i-1}(z_i \boldsymbol{b}_i) = \boldsymbol{d}_i$ where π_i is the orthogonal projection onto $\operatorname{span}(\tilde{\boldsymbol{b}}_1, \ldots, \tilde{\boldsymbol{b}}_i)$. Noting that only the projection $\pi_{i-1}(\boldsymbol{c}_{i-1})$ of $\boldsymbol{c}_{i-1}$ is relevant to the result after iteration i (as seen in (3.1)), we see that

$$\pi_{i-1}(\boldsymbol{c}_{i-1}) \;=\; \pi_{i-1}(\boldsymbol{c}_i - z_i \boldsymbol{b}_i) \;=\; \pi_{i-1}(\boldsymbol{c}_i) - \pi_{i-1}(z_i \boldsymbol{b}_i) \;=\; \pi_{i-1}(\boldsymbol{c}_i) - \boldsymbol{d}_i.$$

The shifted center used in the following iterations thus exactly compensates the distortion caused by adjusting $\boldsymbol{v}$ by $z_i \boldsymbol{b}_i$ instead of $z_i \tilde{\boldsymbol{b}}_i$. Figure 3.2 illustrates the first of three iterations of this sampling process in a three-dimensional lattice, including the correction of the distortion $\boldsymbol{d}_3$.

Before outputting the resulting $\boldsymbol{v}$, we incorporate Brakerski et al.'s [21] rejection step: $\boldsymbol{v}$ is only output with a probability of $\prod_{i=1}^{n} \rho_{s_i', c_i'}(\mathbb{Z}) / \rho_{s_i'}(\mathbb{Z})$. Otherwise, the algorithm loops from the beginning and samples a new $\boldsymbol{v}$. This serves to renormalize the probability of outputting any specific $\boldsymbol{v}$ and make the output distribution match $\mathcal{D}_{B^T B, s, \boldsymbol{c}}$ exactly, as we

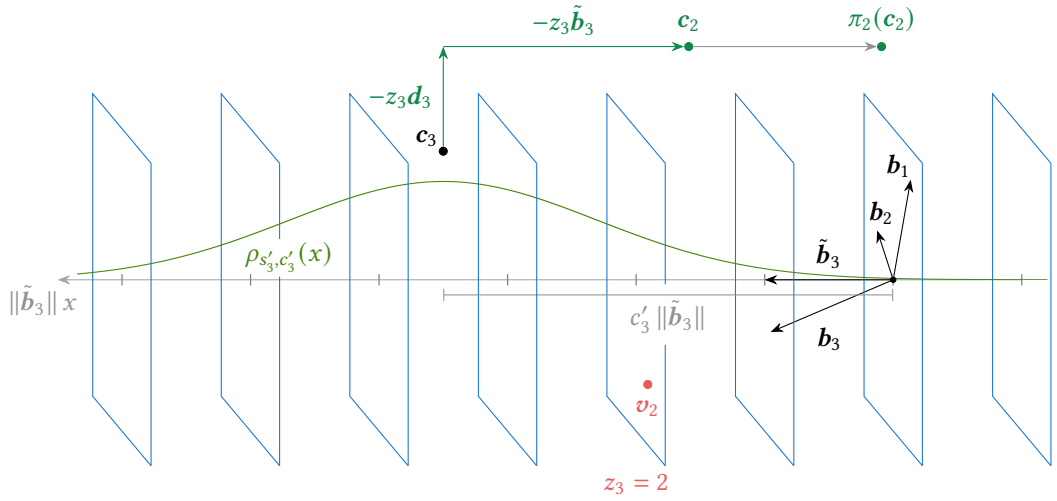


Figure 3.2: Illustration of the first iteration of SampleD in a three-dimensional lattice with the basis $(\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3)$, center $\boldsymbol{c} = \boldsymbol{c}_3$ and parameters $c_3' = 3.5$ and $s_3' = 3$. This basis is oriented such that the Gram-Schmidt vector $\tilde{\boldsymbol{b}}_3$ is horizontal and $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are contained in the plane orthogonal to it at the origin. The sampling of z_3 using SampleD can be understood as choosing one of the planes orthogonal to $\tilde{\boldsymbol{b}}_3$ with the probabilities proportional to the values of the Gaussian $\rho_{s_3', c_3'}$ at its intersections with the planes. Each of these planes corresponds to an integer multiple of $\tilde{\boldsymbol{b}}_3$. In this example, the multiple $z_3 = 2$ was chosen, resulting in $\boldsymbol{v}_2 = 2\boldsymbol{b}_3$. The resulting shift in centers $\boldsymbol{c}_2 \leftarrow \boldsymbol{c}_3 - z_3 \boldsymbol{b}_3$ is also shown. Note how $\boldsymbol{d}_3$ is contained in the plane spanned by $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$. Adding a lattice point sampled around $\pi_2(\boldsymbol{c}_2)$ in the $\boldsymbol{b}_1$–$\boldsymbol{b}_2$-plane to $\boldsymbol{v}_2$ in the following two iterations will correctly center the final result around $\boldsymbol{c}$.

explain in our later proof. The rejection probability can be computed efficiently either by direct evaluation or Poisson summation: If $s_i' < 1$,

$$\rho_{s_i',c_i'}(\mathbb{Z}) = \sum_{k\in\mathbb{Z}} \exp\left(-\pi \frac{(k-c_i')^2}{s_i'^2}\right)$$

decays quickly for $\|k\| \to \infty$ and can be approximated arbitrarily well in polynomial time [21]. On the other hand, the Poisson sum

$$\rho_{s_i',c_i'}(\mathbb{Z}) = s_i' \sum_{k\in\mathbb{Z}} \exp(-\pi k^2 s_i'^2)\cos(2\pi c_i' k) \tag{3.2}$$

decays quickly for $\|k\| \to \infty$ and $s_i' \geq 1$, so an arbitrarily good approximation can be calculated efficiently for any parameters [21].[5] Equation (3.2) follows from the Poisson summation formula $\sum_{n\in\mathbb{Z}} f(n) = \sum_{k\in\mathbb{Z}} F(k)$ (compare the work of Benedetto and Zimmermann [12]) where $F(k)$ is the unitary Fourier transform of $f(n)$:

$$F(k) \coloneqq \int_{-\infty}^{\infty} f(x) e^{-2\pi i k x}\, dx$$

Given the Fourier transform of the Gaussian $e^{-x^2} \overset{\mathcal{F}}{\longleftrightarrow} \sqrt{\pi} e^{-\pi^2 k^2}$ and the elementary properties (compare the work of Campbell and Foster [24, pp. 39, 82])

$$f(ax) \overset{\mathcal{F}}{\longleftrightarrow} \frac{1}{|a|} F\left(\frac{k}{|a|}\right), \quad f(x-c) \overset{\mathcal{F}}{\longleftrightarrow} e^{-i2\pi c k} F(k),$$

we arrive at (3.2) by taking only the real component of each term [21].

Moving to the quadratic form formulation, let Q be the form of interest. We now want to sample integer coefficient vectors instead of lattice vectors. Changing the output of the procedure into a coefficient vector is trivial as we can simply use the sampled z_i as coefficients directly and set $\boldsymbol{v} \leftarrow (z_1 \cdots z_n)^T$. Given a coefficient center $\boldsymbol{c}$ as input, the centers $\boldsymbol{c}_i$ for each iteration can similarly be treated as coefficient vectors by setting $\boldsymbol{c}_{i-1} \leftarrow \boldsymbol{c}_i - z_i \hat{\boldsymbol{e}}_i$ and adjusting the only other calculation using $\boldsymbol{c}_i$ to use the corresponding lattice vector $B\boldsymbol{c}_i$ instead: $c_i' \leftarrow \langle B\boldsymbol{c}_i, \tilde{\boldsymbol{b}}_i\rangle / \langle \tilde{\boldsymbol{b}}_i, \tilde{\boldsymbol{b}}_i\rangle$. To completely eliminate the need for the Cholesky decomposition of Q, we replace $\widetilde{B}$ with the Gram-Schmidt-orthogonalization $\bar{B}$ of $\mathbb{1}_n$ under the inner product $\langle\cdot,\cdot\rangle_Q$ as previously done by de Castro Biage [28]. With the new Gram-Schmidt vectors $\bar{\boldsymbol{b}}_i$, we can set $c_i' \leftarrow \langle \boldsymbol{c}_i, \bar{\boldsymbol{b}}_i\rangle_Q / \langle \bar{\boldsymbol{b}}_i, \bar{\boldsymbol{b}}_i\rangle_Q$ and $s_i' \leftarrow s/\|\bar{\boldsymbol{b}}_i\|_Q$. We claim that these definitions are equal to the previous ones in the lattice formulation: For any basis B with $B^T B = Q$, we have $\langle B\cdot, B\cdot\rangle = \langle\cdot,\cdot\rangle_Q$ by definition. In addition, $\mathbb{1}_n$ is the coefficient

[5] In practice, the requirement that $s \geq \eta_{1/(n+1)}(B)$ implies $s > 1.5\,\|\widetilde{B}\|$ for $n \geq 24$, so $s_i' > 1.5$ for all $i \in [n]$ and the Poisson sum (3.2) will always be used. This sum decays extremely rapidly: For the minimal parameter $s_i' = 1.5$ and $|k| = \pm 3$, $\exp(-\pi k^2 s_i'^2) \approx 2 \cdot 10^{-28} \approx 2^{-92}$. Given that the series term at $k = 0$ is always s_i', any terms with $|k| > 3$ are irrelevant even at high numerical precisions. Note that this also implies $\rho_{s_i',c_i'}(\mathbb{Z})/\rho_{s_i'}(\mathbb{Z}) \approx 1$ for any practical choices of parameters and precisions.

matrix for $B = B\mathbb{1}_n$. Therefore, $\langle \boldsymbol{b}_i, \boldsymbol{b}_j \rangle = \langle \hat{\boldsymbol{e}}_i, \hat{\boldsymbol{e}}_j \rangle_Q \ \forall i, j \in [n]$ and, by linearity of the Gram-Schmidt-orthogonalization, $\widetilde{B} = B\bar{B}$. The correctness of our redefinitions of c'_i and s'_i follows. The final procedure SampleD can be seen in pseudocode in Algorithm 3.2.

Algorithm 3.2 Exact sampling procedure for $\mathcal{D}_{Q,s,\boldsymbol{c}}$ (synthesized from [21, 28, 38])

function SampleD$(Q, s, \boldsymbol{c}) \to \boldsymbol{v} \sim \mathcal{D}_{Q,s,\boldsymbol{c}}$
  $(\bar{\boldsymbol{b}}_1, \ldots, \bar{\boldsymbol{b}}_n) \leftarrow$ Gram-Schmidt-orthogonalization of $(\hat{\boldsymbol{e}}_1, \ldots, \hat{\boldsymbol{e}}_n)$ with Q
  loop
    $\boldsymbol{v}_n \leftarrow 0$
    $\boldsymbol{c}_n \leftarrow \boldsymbol{c}$
    for $i \in \{n, \ldots, 1\}$ **do**
      $c'_i \leftarrow \dfrac{\langle \boldsymbol{c}_i, \bar{\boldsymbol{b}}_i \rangle_Q}{\langle \bar{\boldsymbol{b}}_i, \bar{\boldsymbol{b}}_i \rangle_Q}$
      $s'_i \leftarrow \dfrac{s}{\|\bar{\boldsymbol{b}}_i\|_Q}$
      $z_i \leftarrow \mathcal{D}_{\mathbb{1}, s'_i, c'_i}$
      $\boldsymbol{c}_{i-1} \leftarrow \boldsymbol{c}_i - z_i \hat{\boldsymbol{e}}_i$
      $\boldsymbol{v}_{i-1} \leftarrow \boldsymbol{v}_i + z_i \hat{\boldsymbol{e}}_i$
    return $\boldsymbol{v}_0$ with prob. $\displaystyle\prod_{i=1}^{n} \frac{\rho_{s'_i, c'_i}(\mathbb{Z})}{\rho_{s'_i}(\mathbb{Z})}$

To prove the sampler's correctness, we employ the same proof strategy used by Gentry, Peikert, and Vaikuntanathan [38]:

Lemma 3.2 (adapted from [38]). *For any call to SampleD with parameters $Q \in S_n^{>0}(\mathbb{Z})$, $s \in \mathbb{R}$, and $\boldsymbol{c} \in \mathbb{R}^n$, when $\boldsymbol{v}_0$ is output,*

$$\boldsymbol{v}_0 - \boldsymbol{c} = \sum_{i=1}^{n} (z_i - c'_i) \bar{\boldsymbol{b}}_i$$

where the z_i and c'_i are as set during the algorithm's execution and $\bar{B}$ is the Gram-Schmidt-orthogonalization of $\mathbb{1}_n$ with the inner product $\langle \cdot, \cdot \rangle_Q$.

Proof (adapted from [38]). We perform a proof by induction of

$$(\boldsymbol{v}_0 - \boldsymbol{v}_j) - \bar{\pi}_j(\boldsymbol{c}_j) = \sum_{i=1}^{j} (z_i - c'_i) \bar{\boldsymbol{b}}_i \quad \forall j \in [n], \tag{3.3}$$

where $\bar{\pi}_j \colon \mathbb{R}^n \to \mathbb{R}^n$ is the projection onto $\operatorname{span}(\bar{\boldsymbol{b}}_1, \ldots, \bar{\boldsymbol{b}}_j)$ with the inner product $\langle \cdot, \cdot \rangle_Q$ (so $\bar{\pi}_0(\boldsymbol{x}) = 0$ for all $\boldsymbol{x} \in \mathbb{R}^n$). The claim follows by setting $j = n$ using $\boldsymbol{v}_n = 0$ and $\boldsymbol{c}_n = \boldsymbol{c}$

since $\text{span}(\bar{\boldsymbol{b}}_1, \ldots, \bar{\boldsymbol{b}}_n) = \mathbb{R}^n$. For $j = 0$, Equation (3.3) is obviously true. Now assume that it holds for some $j \in [n-1]$. We show that it is then true for $j+1$ as well:

$$\begin{aligned}
(\boldsymbol{v}_0 - \boldsymbol{v}_{j+1}) - \bar{\pi}_{j+1}(\boldsymbol{c}_{j+1}) &= (\boldsymbol{v}_0 - \boldsymbol{v}_j + z_{j+1}\hat{\boldsymbol{e}}_{j+1}) - \bar{\pi}_j(\boldsymbol{c}_{j+1}) - c'_{j+1}\bar{\boldsymbol{b}}_{j+1} \\
&= (\boldsymbol{v}_0 - \boldsymbol{v}_j) - \bar{\pi}_j(\boldsymbol{c}_j + z_{j+1}\hat{\boldsymbol{e}}_{j+1}) + z_{j+1}\hat{\boldsymbol{e}}_{j+1} - c'_{j+1}\bar{\boldsymbol{b}}_{j+1} \\
&= (\boldsymbol{v}_0 - \boldsymbol{v}_j) - \bar{\pi}_j(\boldsymbol{c}_j) - \bar{\pi}_j(z_{j+1}\hat{\boldsymbol{e}}_{j+1}) + z_{j+1}\hat{\boldsymbol{e}}_{j+1} - c'_{j+1}\bar{\boldsymbol{b}}_{j+1} \\
&= \sum_{i=0}^{j}(z_i - c'_i)\bar{\boldsymbol{b}}_i + z_{j+1}\bar{\boldsymbol{b}}_{j+1} - c'_{j+1}\bar{\boldsymbol{b}}_{j+1} \\
&= \sum_{i=0}^{j+1}(z_i - c'_i)\bar{\boldsymbol{b}}_i.
\end{aligned}$$

The fourth equality follows from the induction hypothesis (3.3) for j and the definition of $\bar{\boldsymbol{b}}_{j+1} = \hat{\boldsymbol{e}}_{j+1} - \bar{\pi}_{j+1}(\hat{\boldsymbol{e}}_{j+1})$. □

Theorem 2.2. *For any $Q \in S_n^{>0}(\mathbb{Z})$, $s \geq \left\|\widetilde{B}_Q\right\| \sqrt{\ln(2n(n+2))/\pi}$, and $\boldsymbol{c} \in \mathbb{R}^n$, the algorithm* SampleD$(Q, s, \boldsymbol{c})$ *terminates in polynomial time with overwhelming probability and outputs $\boldsymbol{v} \sim \mathcal{D}_{Q,s,\boldsymbol{c}}$.*

Proof. We follow Gentry, Peikert, and Vaikuntanathan's [38] proof strategy again, but achieve equality to $\mathcal{D}_{Q,s,\boldsymbol{c}}$ instead of just statistical closeness thanks to the rejection step by Brakerski et al. [21]. To begin, note that

$$\Pr[z_i \leftarrow \text{Sample1D}(s'_i, c'_i) \mid z_j \leftarrow \text{Sample1D}(s'_j, c'_j)\ \forall j > i] = \frac{\rho_{s'_i,c'_i}(z_i)}{\rho_{s'_i,c'_i}(\mathbb{Z})}$$

by the definition of the algorithm and its use of Sample1D. Let $\boldsymbol{V}$ be the event that SampleD$(Q, s, \boldsymbol{c})$ outputs a given $\boldsymbol{v} \in \mathbb{Z}^n$ in any given iteration. By the bijective correspondence between $\boldsymbol{v}$ and the z_i, we then have

$$\Pr[\boldsymbol{V}] = \prod_{i=0}^{n}\frac{\rho_{s'_i,c'_i}(z_i)}{\rho_{s'_i,c'_i}(\mathbb{Z})} \cdot \prod_{i=0}^{n}\frac{\rho_{s'_i,c'_i}(\mathbb{Z})}{\rho_{s'_i}(\mathbb{Z})} = \prod_{i=0}^{n}\frac{\rho_{s'_i,c'_i}(z_i)}{\rho_{s'_i}(\mathbb{Z})},$$

where the second factor is due to the rejection probability. In addition, using Lemma 3.2 gives us

$$\begin{aligned}
\rho_{Q,s,\boldsymbol{c}}(\boldsymbol{v}) = \rho_{Q,s}(\boldsymbol{v} - \boldsymbol{c}) &= \rho_{Q,s}\left(\sum_{i=0}^{n}(z_i - c'_i)\bar{\boldsymbol{b}}_i\right) \\
&= \prod_{i=0}^{n}\rho_s((z_i - c'_i)\,\|\bar{\boldsymbol{b}}_i\|_Q) = \prod_{i=0}^{n}\rho_{s'_i}(z_i - c'_i) = \prod_{i=0}^{n}\rho_{s'_i,c'_i}(z_i),
\end{aligned}$$

so we have $\Pr[\boldsymbol{V}] = \rho_{Q,s,\boldsymbol{c}}(\boldsymbol{v}) / \prod_{i=0}^{n} \rho_{s_i'}(\mathbb{Z})$. Since the probability mass function of $\boldsymbol{V}$ is proportional to the probability mass function of $\mathcal{D}_{Q,s,\boldsymbol{c}}$,[6] we conclude that $\boldsymbol{v} \sim \mathcal{D}_{Q,s,\boldsymbol{c}}$ for $\boldsymbol{v} \leftarrow \textsc{SampleD}(Q, s, \boldsymbol{c})$.

To see that SampleD terminates with overwhelming probability after a bounded number of iterations, we consider the probability $\prod_{i=0}^{n} \rho_{s_i',c_i'}(\mathbb{Z}) / \rho_{s_i'}(\mathbb{Z})$ to produce output at the end of an iteration once again. By our assumption on s, we can set $\epsilon = \frac{1}{n+1}$ and have $s \geq \left\|\widetilde{B}_Q\right\| \sqrt{\ln(2n(n+2))/\pi} \geq \eta_\epsilon(Q)$.[7] By definition, $\|\bar{\boldsymbol{b}}_i\|_Q = \|\tilde{\boldsymbol{b}}_i\| \leq \left\|\widetilde{B}_Q\right\|$, so it follows that

$$s_i' = s / \|\bar{\boldsymbol{b}}_i\|_Q \geq \sqrt{\ln(2n(n+2))/\pi} \geq \eta_\epsilon(\mathbb{1})$$

using Lemma 2.3. Therefore, following Brakerski et al. [21], we can apply Lemma 2.5 to each one-dimensional sample's normalizing constant and get

$$\rho_{s_i',c_i'}(\mathbb{Z}) \in \left[\frac{1-\epsilon}{1+\epsilon}, 1\right] \rho_{s_i'}(\mathbb{Z}) = \left[\frac{n}{n+2}, 1\right] \rho_{s_i'}(\mathbb{Z}).$$

Combining all n dimensions,

$$\prod_{i=0}^{n} \rho_{s_i',c_i'}(\mathbb{Z}) \in \left[\left(\frac{n}{n+2}\right)^n, 1\right] \prod_{i=0}^{n} \rho_{s_i'}(\mathbb{Z}).$$

The probability of producing output in an iteration is thus contained in the interval $\left[\left(\frac{n}{n+2}\right)^n, 1\right]$. The left edge of this interval is monotonically decreasing in n by

$$\frac{d}{dn}\left(\frac{n}{n+2}\right)^n = \left(\frac{n}{n+2}\right)^n \left(\ln\left(\frac{n}{n+2}\right) - \frac{n}{n+2}\right) < 0$$

due to the basic inequality $x \geq \ln(x+1)\ \forall x \in \mathbb{R}$. Elementary transformations and identities also show that

$$\left(\frac{n}{n+2}\right)^n = \left(1 + \frac{-2}{n+2}\right)^n \geq \left(1 + \frac{-2}{n}\right)^n \xrightarrow{n\to\infty} e^{-2}.$$

Since $\left(\frac{n}{n+2}\right)^n$ is monotonically decreasing and bounded from below by a sequence that converges to e^{-2}, it must itself never be less than e^{-2} for any n. Following Brakerski et al.'s [21] argument, we conclude that $\left[\left(\frac{n}{n+2}\right)^n, 1\right] \subset [e^{-2}, 1]$, so the probability of SampleD producing output in an iteration is at least e^{-2} and the probability of it running for more than m iterations is at most $(1-e^{-2})^m$. □

[6] Note that the s_i' depend only on the inputs Q and s, not on any choices made in SampleD, so $\prod_{i=0}^{n} \rho_{s_i'}(\mathbb{Z})$ is a normalizing constant.

[7] Choosing $\epsilon \geq \frac{1}{n+1}$ specifically ensures that $\epsilon(n) \in (0,1)$ is a decreasing function in n for all $n \in \mathbb{N}$.

3.3 Implementation

We implement Algorithm 3.1, Algorithm 3.2, and Ducas and van Woerden's [33] sampler for the public key distribution $\mathcal{D}_s([Q])$ as presented in Algorithm 2.1 in Python using NumPy[8], SciPy[9], and SageMath[10]. The implementation can be found in the code accompanying this work[11] in the file `sampler.py`. Due to the numerical limitations of finite-precision number formats in Python and SageMath, the code does not evaluate the real-valued calculations in the algorithms exactly. However, the removal of the Cholesky decomposition greatly improves the numerical stability of these algorithms, thus reducing the impact of floating-point imprecision. The Gram-Schmidt-orthogonalization and calculation of $\langle \boldsymbol{c}_i, \bar{\boldsymbol{b}}_i \rangle_Q$ in SampleD remain as the only potential sources of significant numerical errors because of the risk of cancellation when calculating the inner products. This could be addressed by switching to an arbitrary-precision rational number format for these steps, which we leave as future work. To verify our implementation's correctness, we compare the theoretical probability density given by $\mathcal{D}_{Q,s,\boldsymbol{c}}$ with the empirical density of samples generated by our code in two cases: The first case, shown in Figure 3.3, tests the implementation of the one-dimensional discrete Gaussian sampler for $\mathcal{D}_{\mathbb{1},s,c}$. With 10 000 samples, the empirical density of values drawn using our implementation of Sample1D closely approximates the theoretical density.

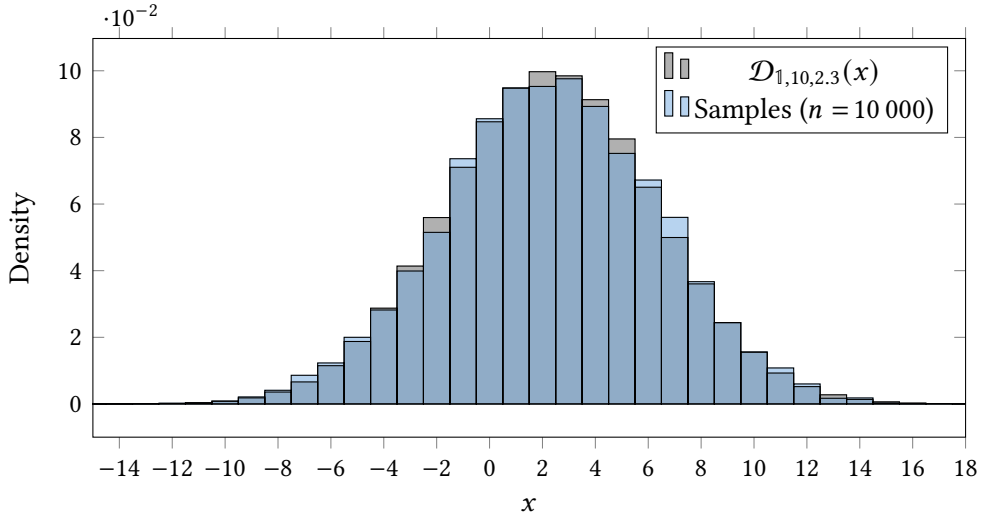


Figure 3.3: Comparison between the theoretical density of the one-dimensional discrete Gaussian distribution $\mathcal{D}_{\mathbb{1},s,c}$ and the empirical density of 10 000 samples drawn using our implementation with the example parameters $s = 10$, $c = 2.3$.

Our second test includes SampleD and compares the theoretical and empirical densities in two dimensions: We begin by sampling a quadratic form $Q \leftarrow \mathcal{D}_s([\mathbb{1}_2])$ and then compare the densities of $\mathcal{D}_{Q,s,\boldsymbol{c}}$ and 180 000 samples (corresponding to approximately 60 samples per

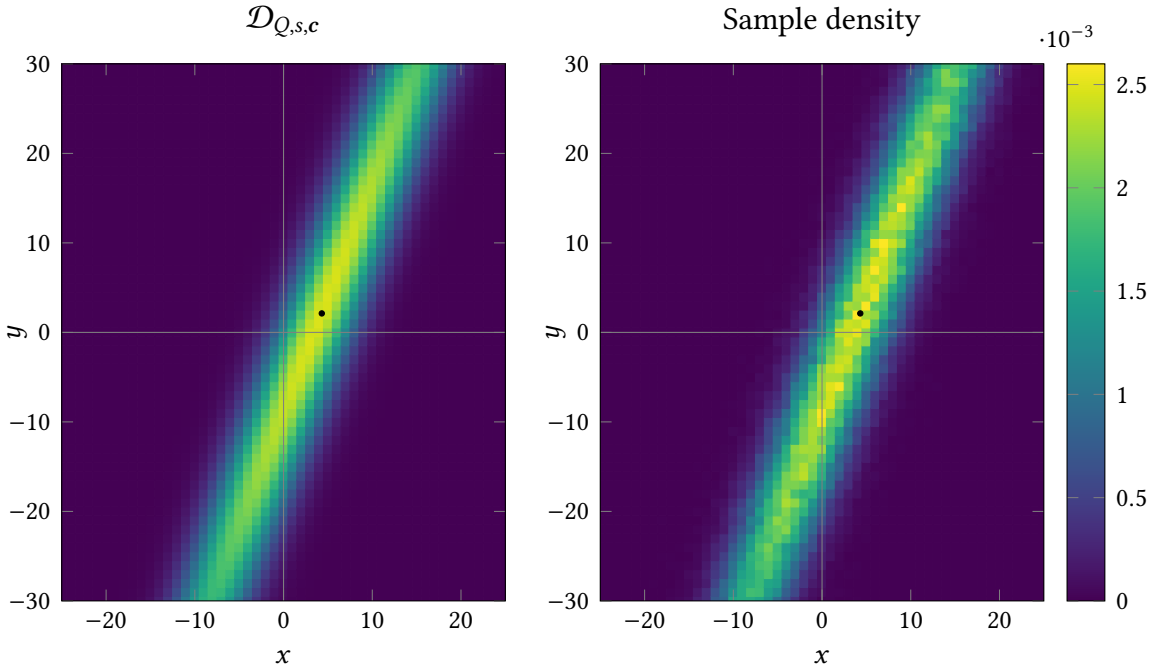
[8] `https://numpy.org/`
[9] `https://scipy.org/`
[10] `https://www.sagemath.org/`
[11] `https://gitlab.kit.edu/ulila/ma-lip-kem/-/tree/master/code/kem_sampling`

bin) drawn using our SampleD code in Figure 3.4. Again, the empirical distribution is close to the theoretical distribution by inspection.

Note that it is no coincidence for the distribution $\mathcal{D}_{Q,s,\boldsymbol{c}}$ in Figure 3.4 to be oriented close to the main diagonal $x = y$: Points $\boldsymbol{x}$ sampled as $\boldsymbol{x} = (\boldsymbol{x}_1 \cdots \boldsymbol{x}_n)^T \leftarrow \mathcal{D}_{Q,s}$ for $Q \leftarrow \mathcal{D}_s([S])$ will typically have $\operatorname{sign}(\boldsymbol{x}_1) = \cdots = \operatorname{sign}(\boldsymbol{x}_n)$ as a consequence of how $\mathcal{D}_s$ samples quadratic forms (compare Algorithm 2.1). Recall that the unimodular matrix U chosen by the sampler is determined such that $T = U^{-1}Y$ is the HNF of $Y = (\boldsymbol{y}_1 \ \cdots \ \boldsymbol{y}_m)$, a full-rank matrix of samples $\boldsymbol{y}_j \leftarrow \mathcal{D}_{S,s'}$ [33]. Equivalently, $UT = Y$, so U maps columns $\boldsymbol{t}_i$ of T to samples of the discrete Gaussian (i.e., vectors of bounded length $\|\boldsymbol{y}_j\|_S \leq s'\sqrt{n}$). This implies that

$$\langle \boldsymbol{t}_j, \boldsymbol{t}_j \rangle_Q = \boldsymbol{t}_j^T U^T S U \boldsymbol{t}_j = \boldsymbol{y}_j^T S \boldsymbol{y}_j \leq s'^2 n,$$

meaning the columns of T form a generating set of bounded-length vectors in the inner product space defined by Q. The definition of the HNF as presented in Definition 2.28 and used by van Woerden [93, Definition 167] as well as the library Sagemath [90] enforces $\boldsymbol{t}_{j_i} \in \mathbb{N}_0^n \ \forall j_i \in J_i$, where $J_i \subset [m]$ is the index set of the pivot columns. These pivot columns



(a) Theoretical probability density of $\mathcal{D}_{Q,s,\boldsymbol{c}}$ renormalized over the displayed area.

(b) Empirical sample density of 180 000 samples drawn using our implementation, normalized over the displayed area.

Figure 3.4: Comparison between the theoretical density of the discrete Gaussian $\mathcal{D}_{Q,s,\boldsymbol{c}}$ and the empirical density of 180 000 samples drawn using our implementation for example parameters. The underlying quadratic form is $Q = \left(\begin{smallmatrix} 13 & -5 \\ -5 & 2 \end{smallmatrix}\right)$, which is sampled from $\mathcal{D}_{s'}([\mathbb{1}_2])$ with $s' = 20$. The corresponding unimodular matrix is $U = \left(\begin{smallmatrix} 2 & -1 \\ -3 & 1 \end{smallmatrix}\right)$. The distribution's parameter is chosen as $s = s'\sqrt{2\ln(16)/\pi}$ as required by Theorem 2.2 together with Lemma 2.9. The center $\boldsymbol{c} = (4.31 \ 2.12)^T$ is marked with a black dot in both plots.

$\boldsymbol{t}_{j_i}$ thus all point into the non-negative hyperoctant $\mathbb{R}^n_{\geq 0}$.[12] Define the convex cone of the $\boldsymbol{t}_{j_i}$ to be

$$\operatorname{cone}(T_{J_i}) \coloneqq \left\{ \sum_{i=1}^{n} \lambda_i \boldsymbol{t}_{j_i} \mid \lambda_i \geq 0 \ \forall i \in [n] \right\}.$$

For $\boldsymbol{x} \in \operatorname{cone}(T_{J_i})$ with $\sum_{i=1}^{n} \lambda_i = \lambda$, we see that

$$\|\boldsymbol{x}\|_Q = \left\| \sum_{i=1}^{n} \lambda_i \boldsymbol{t}_{j_i} \right\|_Q \leq \sum_{i=1}^{n} \lambda_i \left\| \boldsymbol{t}_{j_i} \right\|_Q \leq \lambda s' \sqrt{n},$$

so any $\boldsymbol{x} \in \operatorname{cone}(T_{J_i})$ have a similarly bounded length to the $\boldsymbol{t}_{j_i}$ themselves. We therefore generally expect to find "short" vectors in the hyperoctant $\mathbb{R}^n_{\geq 0}$ that the $\boldsymbol{t}_{j_i}$ are contained in as well as in its negative $\mathbb{R}^n_{\leq 0}$ (since $\|-\boldsymbol{x}\|_Q = \|\boldsymbol{x}\|_Q$). Note that $\boldsymbol{x} \in \mathbb{R}^n_{>0} \cup \mathbb{R}^n_{<0} \iff \operatorname{sign}(\boldsymbol{x}_1) = \cdots = \operatorname{sign}(\boldsymbol{x}_n)$, so $\boldsymbol{x} \leftarrow \mathcal{D}_{Q,s}$ is expected to have entries with the same sign in most cases. Empirically, this occurs for the vast majority of quadratic forms and points sampled from them, especially as the s' used to sample Q grows and narrows the distribution. Despite this, we remark that neither the orientation of the distribution along $\mathbb{R}^n_{\geq 0} \cup \mathbb{R}^n_{\leq 0}$ nor the equal-sign property of the sampled points is guaranteed — for example, Figure 3.4 shows that it is clearly possible, though less likely, to sample points from the other two quadrants.[13]

While it would be interesting to evaluate the sampler in more dimensions, doing so is infeasible due to the "curse of dimensionality": Calculating an empirical density over the region $[-20, 20]^n$ for a typical lattice dimension of $n = 256$ would require $41^{256} \approx 7 \cdot 10^{413}$ bins. Even for smaller dimensions, performing meaningful statistical tests would require prohibitively many samples for the large number of bins. The code used to generate the data sets for Figure 3.3 and Figure 3.4 along with any other figures based on sampled data in this work can be found in the file `paper-figures.py` in the code accompanying this thesis. We make use of our implementation throughout the course of this work as a source of empirical evidence for the behavior of the discrete Gaussian distribution $\mathcal{D}_{Q,s,\boldsymbol{c}}$ and public key distribution $\mathcal{D}_s([Q])$.

[12] This could also be interpreted as $\boldsymbol{t}_{j_i}$ forcing the i-th basis vector $\boldsymbol{b}_i$ of B_Q to "point against" the basis vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1}$ since a sum of these vectors with non-negative weights must be short.

[13] Also note that this has no impact on the average-case hardness of LIP, which is proven by Ducas and van Woerden [33] for precisely the distribution $\mathcal{D}_s$ exhibiting this property.

4 CCA2 Security from k-Repetition

The cryptographic framework k-repetition is used to create IND-CCA2-secure PKEs by repeating an IND-CPA-secure PKE k times. It was first used by Dolev, Dwork, and Naor [30] in conjunction with non-interactive zero-knowledge proofs[1] and later adapted for use with TDOWFs in the standard model by Rosen and Segev [84]. Döttling et al. [31] combined the ideas from Dolev et al. and Rosen et al. to create a CCA2-secure PKE based on the McEliece PKE in the standard model. In this section, we describe k-repetition as used by Döttling et al. We also explain the differences when applying the framework to a KEM instead of a PKE. Finally, we give a brief overview of the security proof used with k-repetition.

The k-repetition framework derives its name from its ciphertexts, which consist of k repeated encryptions of the same message with different public keys and randomness. Here, $k \in \Theta(\lambda)$ is a parameter of the final IND-CCA2-secure scheme. These repeated encryptions are combined with a signature scheme and a property called *verifiability* to permit the IND-CCA2 game's decryption oracle to be simulated. Verifiability ensures that a ciphertext can be decrypted and checked for validity even given only one of the secret keys. The first building block of the k-repetition framework is the k-repeated PKE. It makes use of an *encoding*, which consists of two efficiently computable functions (E, D). $E(m; r) \to m_e$ encodes messages into a different representation m_e while incorporating the randomness r, while $D(m_e) \to m$ decodes the original message. We refer readers to Döttling et al. [31] for details on the properties of their encoding.

Definition 4.1 (k-repeated PKE (PKE_k) (notation adapted from [31, Definition 9]))
For a given PKE ($\textsc{Gen}$, $\textsc{Enc}$, $\textsc{Dec}$) and an encoding (E, D), the k-repeated PKE_k[2] is defined as a tuple of PPT algorithms $(\textsc{Gen}_k, \textsc{Enc}_k, \textsc{Dec}_k)$ such that:

- $\textsc{Gen}_k \to (pk, sk)$: $\textsc{Gen}_k$ generates k keys $(pk_i, sk_i) \leftarrow \textsc{Gen}$ for $i \in [k]$ and sets $pk = \left(pk_i\right)_{i\in[k]}$ and $sk = (sk_i)_{i\in[k]}$.
- $\textsc{Enc}_k(pk, m; r_s, r_1, \ldots, r_k) \to c = (c_1, \ldots, c_k)$ with $c_i \leftarrow \textsc{Enc}(pk_i, E(m; r_s), r_i)$: Here, m is the plaintext message, r_s is shared randomness used in every encryption, and $r_1, \ldots, r_k$ are independent randomness for every individual c_i.
- $\textsc{Dec}_k(sk, c) \to m$: $\textsc{Dec}_k$ decrypts and decodes every component c_i of the ciphertext as $m_i \leftarrow D(\textsc{Dec}(sk_i, c_i))$. If $m_1 = \cdots = m_k$ and m_1 is a valid message from the PKE's message space $\mathcal{M}$, then the algorithm outputs m_1. Otherwise, the output is $\perp$.

[1] An earlier scheme by Naor and Yung [68] is similar in construction, but only achieves IND-CCA1 security.
[2] Döttling et al. call this the k-repetition PKE, a term we reserve for the final, IND-CCA2-secure scheme.

We can analogously define a k-repeated KEM (KEM_k) ($\text{Gen}_k, \text{Encaps}_k, \text{Decaps}_k$) from a given KEM (Gen, Encaps, Decaps) by letting $(c_i, ek_i) \leftarrow \text{Encaps}(pk_i; r_s, r_i)$ in the definition of Encaps_k and $ek_i \leftarrow \text{Decaps}(sk_i, c_i)$ in the definition of Decaps_k. In Encaps, r_s serves as shared randomness separate from the individual randomness r_i. The encapsulated key ek_i generated by Encaps must *only* depend on r_s to ensure that each c_i decrypts to the same key. We refer to each of the k copies of the PKE or KEM in the construction as *instances*. To be usable in the k-repetition framework, PKE_k (or KEM_k) must be IND-CPA-secure and verifiable:

Definition 4.2 (Verifiability (adapted from [31, Definition 11]))
A k-repeated PKE scheme PKE_k or KEM scheme KEM_k is *verifiable* if there is a PPT algorithm $\text{Dec}_{\text{Ver}}(pk, c, sk_i)$[3] with $\text{Dec}_{\text{Ver}}(pk, c, sk_i) = \text{Dec}_k(sk, c)$ (or $= \text{Decaps}_k(sk, c)$), i.e., an algorithm that *decrypts* the key or message and *verifies* the ciphertext given one secret key sk_i for $i \in [k]$.[4]

Dec_{Ver} is a generalization of the Verify algorithm used by Döttling et al. [31] — instead of just outputting a bit indicating whether the ciphertext is valid or not, Dec_{Ver} also outputs the message or key. The idea behind the Dec_{Ver} algorithm is the following: Given one of the k secret keys sk_i, the corresponding ciphertext c_i can be decrypted to recover the plaintext message m (or key ek) and some part of the shared randomness r_s. For a verifiable PKE_k or KEM_k, these then suffice to determine that the other ciphertexts $c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_k$ would also correctly decrypt to the same message or key. Note that this implies that the base PKE (or KEM) is *not* IND-CPA-secure since knowledge of the message $E(m; r_s)$ (or key ek) is enough to determine whether the ciphertext would decrypt to said message (or key) [31]. We can now define the full k-repetition framework.

Definition 4.3 (k-repetition-PKE (KPKE) (notation adapted from [31]))
Given an IND-CPA-secure and verifiable PKE_k and a sEUF-1-CMA-secure signature scheme SIG ($\text{Gen}_S, \text{Sign}_S, \text{Verify}_S$) with verification keys that are k-bit strings,[5] we define a KPKE ($\text{Gen}_K, \text{Enc}_K, \text{Dec}_K$) with:

- $\text{Gen}_K \rightarrow (pk, sk)$: The public key is set as $pk = (pk_1^0, pk_1^1, \ldots, pk_k^0, pk_k^1)$ and the secret key is set as $sk = (sk_1^0, sk_1^1, \ldots, sk_k^0, sk_k^1)$. The keypairs (pk_i^b, sk_i^b) for $i \in \{1, \ldots, k\}, b \in \{0, 1\}$ are generated by calling Gen a total of $2k$ times.
- $\text{Enc}_K(pk, m) \rightarrow c = (c', vk, \sigma)$: To encrypt the message m, Enc_K first generates a new signing keypair $(sk_{sig}, vk) \leftarrow \text{Gen}_S$. The public key for PKE_k is then derived as $pk^{vk} = (pk_1^{vk_1}, \ldots, pk_k^{vk_k})$, i.e., by choosing between pk_i^0 and pk_i^1 depending on the i-th bit vk_i of vk. The ciphertext c' is built as $\text{Enc}_k(pk^{vk}, m; r)$ with fresh randomness r and signed via $\sigma \leftarrow \text{Sign}_S(sk_{sig}, c')$ and (c', vk, σ) is output.

[3] In general, Dec_{Ver} would also need the index i as an extra argument to find the right ciphertext c_i unless Dec or Decaps can reliably detect whether a ciphertext is valid for the given key. We omit this minor technical detail for the sake of brevity.

[4] It is permissible for Dec_{Ver} to fail negligibly often; we do not require that in this work.

[5] This limitation can be bypassed by using $h(vk)$ instead of vk for an appropriate hash function h [84].

- $\text{Dec}_K(sk, (c', vk, \sigma)) \to m$: Dec_K first verifies the signature with $\text{Verify}_S(vk, c', \sigma)$. If this fails, the function returns $\perp$. Otherwise, the message is decrypted as $m \leftarrow \text{Dec}_k(sk^{vk}, c')$ and output.

Theorem 4.1 (Döttling et al. [31, Theorem 1])**.** *If PKE_k is an IND-CPA-secure PKE and verifiable and SIG is a sEUF-1-CMA-secure signature scheme, KPKE is an IND-CCA2-secure PKE.*

Proof sketch. We only give a brief overview of the proof here — see the work of Döttling et al. [31] for the full proof. Consider an attacker $\mathcal{A}$ on the IND-CCA2 security of KPKE and construct an attacker $\mathcal{B}$ on the IND-CPA security of PKE_k: $\mathcal{B}$ receives the public key pk' from the IND-CPA challenger C and generates the signature keypair (sk^*_{sig}, vk^*) for its challenge ciphertext ahead of time. It then sets $pk_i^{vk^*} \leftarrow pk'_i$ and generates the remaining k (pk_i^b, sk_i^b) keypairs for the KPKE itself. $\mathcal{B}$ passes $\mathcal{A}$'s challenge messages through to C and signs the resulting ciphertext with sk^*_{sig} before passing it back to $\mathcal{A}$.

We now conclude that $\mathcal{A}$ cannot generate another valid KPKE ciphertext with the same vk^* thanks to the unforgeability of the signature scheme. Therefore, any valid ciphertexts $\mathcal{A}$ sends to the decryption oracle must use a $vk \neq vk^*$. Any such vk must differ from vk^* in at least one bit i — it follows that $\mathcal{B}$ knows at least one of the secret keys $sk_i^{vk_i}$ used in the ciphertext, with which it can extract the message and verify that the ciphertext is valid using $\text{Dec}_{\text{Ver}}(pk^{vk}, c, sk_i^b)$. This allows $\mathcal{B}$ to simulate the decryption oracle correctly. Figure 4.1 shows the generation of the keypairs and their selection via the verification keys for both vk^* and an adversary-chosen vk for $k = 5$.

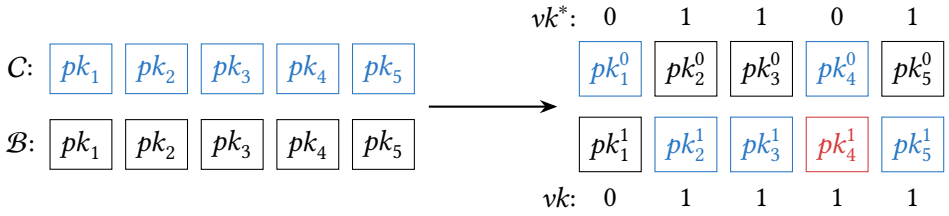


Figure 4.1: Key structure in KPKE's security proof with $k = 5$. Here, the adversary-chosen vk differs from vk^* in the highlighted fourth bit, so sk_4^1 can be used to run Dec_{Ver}.

As before, we can define a KKEM analogously to Definition 4.3 and apply an equivalent proof. The main difference between the proofs is that $\mathcal{A}$ does not choose two messages that it sends to $\mathcal{B}$. Instead, C runs Encaps_k and sends either the real key ek or a uniformly random key $\overline{ek}$ to $\mathcal{B}$ along with the ciphertext. $\mathcal{B}$ forwards both to $\mathcal{A}$ unchanged. After the challenge ciphertext and key have been passed to $\mathcal{A}$, the rest of the simulation works just as for a KPKE.

Theorem 4.2. *If KEM_k is an IND-CPA-secure KEM and verifiable and SIG is a sEUF-1-CMA-secure signature scheme, KKEM is an IND-CCA2-secure KEM.*

Note that the construction of the KPKE or KKEM is completely generic and independent of the specific construction of PKE_k or KEM_k. To the best of our knowledge, this second step of going from a k-repeated scheme to a k-repetition scheme is also essentially identical across all instances of k-repetition in the literature [30, 31, 77, 84]. On the other hand, constructing a scheme suitable for use with this second step in the standard model based on a generic PKE or KEM has yet to be done: The only generic k-repetition constructions rely either on lossy TDOWFs [84] or require switching to the CRS model or ROM [30].[6] Thus, the strategy to construct an IND-CCA2-secure PKE or KEM using k-repetition based on a new assumption or primitive is to build a verifiable, IND-CPA-secure PKE_k or KEM_k using that assumption or primitive.

[6] Dolev, Dwork, and Naor [30]'s approach requires non-interactive zero-knowledge proofs, which Goldreich and Oren [42, Theorem 4.3] show to be impossible in the standard model for the class of statements that need to be proven for k-repetition.

5 k-Repetition with LIP

In this section, we present our approaches to building a verifiable, IND-CPA-secure k-repeated PKE or KEM as defined in the previous section using LIP. This would allow us to apply the generic transformation from Definition 4.3 to create an LIP-based IND-CCA2-secure PKE or KEM in the standard model. We begin by introducing Ducas and van Woerden's [33] IND-CPA-secure KEM based on LIP in Section 5.1, then determine the requirements that any successful approach must satisfy in order to be both IND-CPA-secure and verifiable in Section 5.2. In Section 5.3 and Section 5.4, we discuss approaches based on correlating components of the LIP-KEM to create a k-repeated KEM. Section 5.5 explores using the LIP-KEM generically to build a k-repeated PKE. Finally, we examine whether lossy TDOWFs could be constructed from LIP in Section 5.6.

5.1 LIP-KEM

Ducas and van Woerden [33] define an IND-CPA-secure KEM based on LIP, which we refer to as the LIP-KEM throughout this work. We use the LIP-KEM as a base for our approaches to constructing a k-repeated KEM or PKE. In this section, we give an overview of the LIP-KEM's definition, its security proof, and cryptanalytic concerns of relevance to this work. Except where noted otherwise, the contents of this section are taken from the work of Ducas and van Woerden [33].

The LIP-KEM is fundamentally based on generating a small random error term $\boldsymbol{e}$ in a lattice and deriving a random key from it using an extractor. The ciphertext is a syndrome of $\boldsymbol{e}$ that can be decoded efficiently to retrieve $\boldsymbol{e}$ and recover the encapsulated key. An adversary without the secret key, on the other hand, cannot decode the syndrome given the hardness of ΔLIP.

To define the LIP-KEM, we require a quadratic form $S \in S_n^{>0}(\mathbb{Z})$ that has an efficient decoding radius of $r < \lambda_1(S)/2$. Public keys are given by $P \leftarrow \mathcal{D}_s([S])$ such that $P = U^T S U$ with the unimodular secret key $U \in \mathrm{GL}_n(\mathbb{Z})$. Using a discrete Gaussian sampler, we sample an $\|\boldsymbol{e}\|_P \leq r$. The syndrome $\boldsymbol{c}$ is calculated as $\boldsymbol{c} \leftarrow \boldsymbol{e} \bmod \mathbb{Z}^n$. It follows that $\boldsymbol{x} := \boldsymbol{c} - \boldsymbol{e} \in \mathbb{Z}^n$, so $\boldsymbol{c} = \boldsymbol{x} + \boldsymbol{e}$ and $\|\boldsymbol{c} - \boldsymbol{x}\|_P = \|\boldsymbol{e}\|_P \leq r$, meaning that $\boldsymbol{e}$ is uniquely reconstructable by decoding $\boldsymbol{c}$ to $\boldsymbol{x}$.

The IND-CPA security proof relies on another quadratic form Q with a dense sublattice such that ac-$\Delta\mathrm{LIP}_s^{S,Q}$ is hard. In the IND-CPA game, this allows us to swap Q in for S when generating the public key: $P \leftarrow \mathcal{D}_s([Q])$. Since P now has a dense sublattice, there are an

exponential number of lattice points $\boldsymbol{x}'$ in the vicinity of the challenge $\boldsymbol{c}$, making decoding to the right $\boldsymbol{x}$ and thus the right $\boldsymbol{e}$ statistically hard. We present the full description of the LIP-KEM in pseudocode in Algorithm 5.1.

The correctness of the LIP-KEM is shown using the following argument: First, note that s is chosen such that $\mathcal{D}_s([S])$ can be correctly and efficiently sampled according to Lemma 2.8. Next, $\|\boldsymbol{e}\|_P \leq r$ with overwhelming probability by the choice of parameters: Since

$$\frac{qr}{\sqrt{n}} \geq s\sqrt{n}\sqrt{\frac{\ln(2n^2+4n)}{\pi}} \geq \left\|\widetilde{B}_P\right\| \sqrt{\frac{\ln(2n^2+4n)}{\pi}}$$

by definition of q and s and Lemma 2.9, $\mathcal{D}_{P,qr/\sqrt{n}}$ can also be sampled efficiently using SampleD by Theorem 2.2 and returns a lattice point $q\boldsymbol{e}$ with $\|q\boldsymbol{e}\|_P \leq qr$ with overwhelming probability by Lemma 2.6. As discussed above, $\boldsymbol{c}$ is uniquely decodable to a lattice point $\boldsymbol{x}$ in P. Equivalently, $U\boldsymbol{c}$ is decodable to a lattice point $\boldsymbol{y} = U\boldsymbol{x}$ in S since $\|\cdot\|_P = \|U\cdot\|_S$. Decaps

Algorithm 5.1 LIP-KEM definition [33]

▹ **Parameters:**

- Quadratic form $S \in S_n^{>0}(\mathbb{Z})$
- Decoding radius $r < \lambda_1(S)/2$ of S
- Efficient decoding algorithm $\textsc{Decode}(\boldsymbol{z}) \to \boldsymbol{x}$ for S with decoding radius r
- l-extractor $\mathcal{E}\colon \frac{1}{q}\mathbb{Z}^n \times \{0,1\}^z \to \{0,1\}^l$ with $l \in \Theta(\lambda)$
- $s \geq \max\left\{\lambda_n(S),\ \left\|\widetilde{B}_S\right\| \cdot \sqrt{\frac{\ln(2n^2+4n)}{\pi}}\right\}$
- $q = \left\lceil \frac{s \cdot n}{r} \cdot \sqrt{\frac{\ln(2n^2+4n)}{\pi}} \right\rceil$

function $\textsc{Gen} \to (pk, sk)$
  $P \leftarrow \mathcal{D}_s([S])$ with $U \in \mathrm{GL}_n(\mathbb{Z}),\ U^T S U = P$
  return $(pk, sk) = (P, U)$

function $\textsc{Encaps}(P) \to (c, ek)$
  $\boldsymbol{e} \leftarrow \frac{1}{q}\mathcal{D}_{P,\frac{qr}{\sqrt{n}}} \in \frac{1}{q}\mathbb{Z}^n$
  $\boldsymbol{c} \leftarrow \boldsymbol{e} \bmod \mathbb{Z}^n \implies \boldsymbol{c} \in \left\{0, \frac{1}{q}, \ldots, \frac{q-1}{q}\right\}^n$
  $Z \xleftarrow{\$} \{0,1\}^z$
  $ek \leftarrow \mathcal{E}(\boldsymbol{e}, Z)$
  return $((\boldsymbol{c}, Z), ek)$

function $\textsc{Decaps}(U, (\boldsymbol{c}, Z)) \to ek$ or $\perp$
  $\boldsymbol{y} \leftarrow \textsc{Decode}(S, U\boldsymbol{c})$
  if $\|\boldsymbol{y} - U\boldsymbol{c}\|_S \geq r$ or decoding fails, **then return** $\perp$
  $ek \leftarrow \mathcal{E}(\boldsymbol{c} - U^{-1}\boldsymbol{y}, Z)$
  return ek

can then recover the error term $\boldsymbol{e}$ by transforming $\boldsymbol{y}$ back into $\boldsymbol{x} = U^{-1}\boldsymbol{y}$ and computing $\boldsymbol{e} = \boldsymbol{c} - \boldsymbol{x}$, which lets one extract the same *ek* using the extractor $\mathcal{E}$ and the seed Z.

As previously described, we need a second quadratic form Q with a dense sublattice to prove the security of the LIP-KEM. The coefficient vectors of the lattice points in this sublattice of rank r take the form $D\mathbb{Z}^r$ with $D \in \mathbb{Z}^{n\times r}$. For the sublattice to be dense, we require that $r/\sqrt{n} \geq 2\eta_\epsilon(D^T Q D)$. Under these conditions, Lemma 5.1 guarantees a minimum entropy bound:

Lemma 5.1 ([33, Lemma 5.1])**.** *Given a quadratic form $Q \in S_n^{>0}(\mathbb{Z})$ with a sublattice $D\mathbb{Z}^r$ of rank r, $\epsilon > 0$, $\boldsymbol{c} \in \mathbb{R}^n$, and $s = r/\sqrt{n} \geq 2\eta_\epsilon(D^T Q D)$,*

$$\Pr_{X\sim\mathcal{D}_{Q,s,\boldsymbol{c}}}[X = \boldsymbol{x}] \leq \frac{1+\epsilon}{1-\epsilon}2^{-r} \quad \forall \boldsymbol{x} \in \mathbb{Z}^n.$$

Theorem 5.1 (LIP-KEM is IND-CPA-secure [33, Theorem 5.2])**.** *Let $S, Q \in S_n^{>0}(\mathbb{Z})$ be two quadratic forms such that S can be efficiently decoded in a radius of r and Q has a dense rank-r sublattice with $r \in \Theta(n)$ and $\frac{r}{\sqrt{n}} \geq 2\eta_{1/2}(D^T Q D)$. Set $l \leq r - \log_2(3)$, $\epsilon := \frac{1}{2}$ and choose the parameter s and extractor $\mathcal{E}$ according to Algorithm 5.1. If ac-ΔLIP$_s^{S,Q}$ is hard, then the LIP-KEM with these parameters is IND-CPA-secure.*

Proof (expanded from [33]). Consider the following two games:

G_1: This is the normal KEM-IND-CPA game (see Definition 2.14) with the challenger generating the public key $P \leftarrow \mathcal{D}_s([S])$ using Gen and serving the challenge using Encaps as defined in Algorithm 5.1.

G_2: This game is the same as G_1 except that the public key is now sampled from $\mathcal{D}_s([Q])$ instead of $\mathcal{D}_s([S])$.

$G_1 \overset{\text{c}}{\approx} G_2$**:** A distinguisher for G_1 and G_2 is easily converted into an attacker on ac-ΔLIP$_s^{S,Q}$. Since ac-ΔLIP$_s^{S,Q}$ is hard by assumption, G_1 and G_2 are indistinguishable. Note that the secret key U is not required to simulate the IND-CPA challenger correctly.

We present an expanded version of the second part of Ducas et al.'s proof here for improved clarity. Let us consider the attacker's perspective in G_2: The attacker is given the public key $P \leftarrow \mathcal{D}_s([Q])$ and the challenge ciphertext $(\boldsymbol{c}, Z)$, but not the true $\boldsymbol{e}$ nor the true $\boldsymbol{x} = \boldsymbol{c} - \boldsymbol{e}$. Knowing $\boldsymbol{c}$, any candidate $\boldsymbol{e}'$ must be of the form $\boldsymbol{e}' = \boldsymbol{c} - \boldsymbol{x}'$ for some $\boldsymbol{x}' \in \mathbb{Z}^n$ by construction since $\boldsymbol{c} = \boldsymbol{e} \bmod \mathbb{Z}^n$. Every $\boldsymbol{x}'$ thus induces a candidate $\boldsymbol{e}'$. Since the true $\boldsymbol{e}$ is sampled as $\boldsymbol{e} \leftarrow q^{-1}\mathcal{D}_{P,qr/\sqrt{n}}$, an attacker that does not include $\boldsymbol{c}$ in its considerations would believe that

$$\Pr_{q\boldsymbol{e}\sim\mathcal{D}_{P,\frac{qr}{\sqrt{n}}}}[q\boldsymbol{e} = q\boldsymbol{e}'] = \frac{\rho_{P,\frac{qr}{\sqrt{n}}}(q\boldsymbol{e}')}{\rho_{P,\frac{qr}{\sqrt{n}}}(\mathbb{Z}^n)} \text{ if } q\boldsymbol{e}' \in \mathbb{Z}^n, \text{ and 0 otherwise}$$

by Definition 2.23. A more intelligent adversary will of course make use of $\boldsymbol{c}$ and will thus be able to combine its knowledge that $\boldsymbol{e} \sim q^{-1}\mathcal{D}_{P,qr/\sqrt{n}}$ with the fact that $\boldsymbol{e} = \boldsymbol{c} - \boldsymbol{x}$ to narrow down the possible $\boldsymbol{e}'$. We first show that $\boldsymbol{e}' \in \operatorname{supp}(q^{-1}\mathcal{D}_{P,qr/\sqrt{n}})$ for every candidate $\boldsymbol{e}' = \boldsymbol{c} - \boldsymbol{x}'$ (i.e., that $q\boldsymbol{e}' \in \mathbb{Z}^n$):

$$\begin{aligned} & \boldsymbol{e}' = \boldsymbol{c} - \boldsymbol{x}' \\ \iff & \boldsymbol{e}' = \boldsymbol{c} - \boldsymbol{x} + (\boldsymbol{x} - \boldsymbol{x}') \\ \iff & \boldsymbol{e}' = \boldsymbol{e} + (\boldsymbol{x} - \boldsymbol{x}') \\ \iff & q\boldsymbol{e}' = q\boldsymbol{e} + q(\boldsymbol{x} - \boldsymbol{x}') \in \mathbb{Z}^n, \end{aligned}$$

so every candidate $\boldsymbol{e}'$ is indeed a scaled lattice vector: $q\boldsymbol{c} - q\mathbb{Z}^n \subset \mathbb{Z}^n$. Conversely, any vector $q\tilde{\boldsymbol{e}} \in \mathbb{Z}^n \setminus (q\boldsymbol{c} - q\mathbb{Z}^n)$ is not a valid candidate since $\boldsymbol{c} - \tilde{\boldsymbol{e}} \notin \mathbb{Z}^n$, so $\tilde{\boldsymbol{e}}$ is not a coefficient vector from a lattice point $\boldsymbol{x}'$ to $\boldsymbol{c}$. See Figure 5.1 for a visualization of these points and vectors. With the potential options for $\boldsymbol{e}$ thus reduced from $\mathbb{Z}^n$ to $q\boldsymbol{c} - q\mathbb{Z}^n$, we can again consider the probability of each remaining candidate $\boldsymbol{e}'$ having been sampled as the true $\boldsymbol{e}$ (i.e., the probability of $\boldsymbol{e} = \boldsymbol{e}'$ conditioned on $\boldsymbol{c} = \boldsymbol{e} \bmod \mathbb{Z}^n$). These probabilities are still in the same proportions $\rho_{P,qr/\sqrt{n}}(q\boldsymbol{e}')$ relative to each other as before, but are now normalized over the smaller support $q\boldsymbol{c} - q\mathbb{Z}^n$:

$$\Pr_{q\boldsymbol{e}\sim\mathcal{D}_{P,\frac{qr}{\sqrt{n}}}}[q\boldsymbol{e} = q\boldsymbol{e}' \mid \boldsymbol{c} = \boldsymbol{e} \bmod \mathbb{Z}^n] = \frac{\rho_{P,\frac{qr}{\sqrt{n}}}(q\boldsymbol{e}')}{\rho_{P,\frac{qr}{\sqrt{n}}}(q\boldsymbol{c} - q\mathbb{Z}^n)} \text{ if } q\boldsymbol{e}' \in q\boldsymbol{c} - q\mathbb{Z}^n, \text{ else } 0$$

$$\iff \Pr_{q\boldsymbol{e}\sim\mathcal{D}_{P,\frac{qr}{\sqrt{n}}}}[q\boldsymbol{e} = q\boldsymbol{c} - q\boldsymbol{x}' \mid \boldsymbol{c} = \boldsymbol{e} \bmod \mathbb{Z}^n] = \frac{\rho_{P,\frac{qr}{\sqrt{n}}}(q\boldsymbol{c} - q\boldsymbol{x}')}{\rho_{P,\frac{qr}{\sqrt{n}}}(q\boldsymbol{c} - q\mathbb{Z}^n)} \text{ if } \boldsymbol{x}' \in \mathbb{Z}^n, \text{ else } 0$$

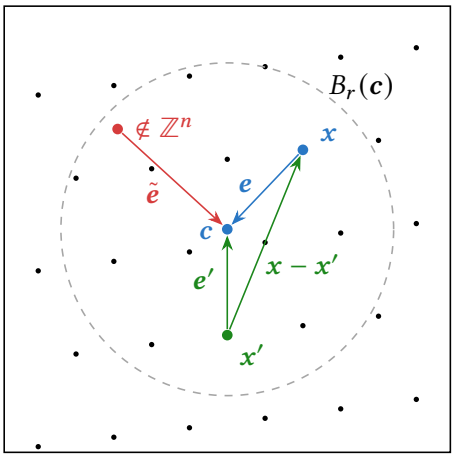


Figure 5.1: Diagram showing a ciphertext $\boldsymbol{c}$ in a dense lattice along with several other vectors used in the LIP-KEM security proof. The true decoding $\boldsymbol{x}$ is shown in blue, a valid candidate $\boldsymbol{x}'$ in green, and an invalid candidate with the error $\tilde{\boldsymbol{e}}$ in red. Note that the points and vectors displayed in the figure have been transformed into the lattice for the sake of visualization, so each coefficient vector $\boldsymbol{a}$ is actually rendered as $B\boldsymbol{a}$ for some $B^T B = P$.

and with $\rho_{P,\alpha s}(\alpha \boldsymbol{y}) = \rho_{P,s}(\boldsymbol{y})$ and $\rho_{P,s}(\boldsymbol{a} - \boldsymbol{y}) = \rho_{P,s,\boldsymbol{a}}(\boldsymbol{y})$ $\forall \alpha \in \mathbb{R},\ \boldsymbol{a}, \boldsymbol{y} \in \mathbb{R}^n$:

$$\iff \quad \Pr_{\boldsymbol{e} \sim \frac{1}{q}\mathcal{D}_{P,\frac{qr}{\sqrt{n}}}}[\boldsymbol{e} = \boldsymbol{c} - \boldsymbol{x}' \mid \boldsymbol{c} = \boldsymbol{e} \bmod \mathbb{Z}^n] = \frac{\rho_{P,\frac{r}{\sqrt{n}}}(\boldsymbol{c} - \boldsymbol{x}')}{\rho_{P,\frac{r}{\sqrt{n}}}(\boldsymbol{c} - \mathbb{Z}^n)} \text{ if } \boldsymbol{x}' \in \mathbb{Z}^n, \text{ else } 0$$

$$\iff \quad \Pr_{\boldsymbol{e} \sim \frac{1}{q}\mathcal{D}_{P,\frac{qr}{\sqrt{n}}}}[\boldsymbol{e} = \boldsymbol{c} - \boldsymbol{x}' \mid \boldsymbol{c} = \boldsymbol{e} \bmod \mathbb{Z}^n] = \frac{\rho_{P,\frac{r}{\sqrt{n}},\boldsymbol{c}}(\boldsymbol{x}')}{\rho_{P,\frac{r}{\sqrt{n}},\boldsymbol{c}}(\mathbb{Z}^n)} \text{ if } \boldsymbol{x}' \in \mathbb{Z}^n, \text{ else } 0.$$

Equivalently, we see that $\boldsymbol{e} \sim \boldsymbol{c} - \mathcal{D}_{P,r/\sqrt{n},\boldsymbol{c}}$ conditioned on $\boldsymbol{c} = \boldsymbol{e} \bmod \mathbb{Z}^n$. Since the attacker receives no further information about the challenge, $\mathcal{D}_{P,r/\sqrt{n},\boldsymbol{c}}$ corresponds to the attacker's remaining uncertainty about $\boldsymbol{e}$. With $\epsilon = \frac{1}{2}$, Lemma 5.1 tells us that each candidate $\boldsymbol{e}'$ has a probability of $\leq 3 \cdot 2^{-r}$ to be the true $\boldsymbol{e}$. It follows that $\boldsymbol{e}$ has an entropy of at least $-\log_2(3 \cdot 2^{-r}) = r - \log_2(3)$ bits. By our choice of l, this is enough for the extractor $\mathcal{E}$ to produce a key ek indistinguishable from uniform randomness on $\{0,1\}^l$. An attacker therefore has at most a negligible advantage in G_2. □

Instantiating the LIP-KEM requires two quadratic forms S and Q such that S is decodable, Q has a dense sublattice, and both S and Q have the same genus. Ducas and van Woerden [33] gave a generic construction with which any full-rank lattice Λ with an efficient decoder up to a radius of r and with dimension $n/2$ can be turned into a pair of full-rank lattices Λ_S and Λ_Q suitable for instantiating S and Q, respectively. This construction was later improved by van Woerden [92], who shows that, for any Λ, there exists a Λ' with the same genus and an asymptotically optimal $\mathrm{gap}(\Lambda') \in O(1)$. We present his construction in brief here:

Choose $f(n)$ and $g(n)$ such that $\mathrm{gap}(\Lambda) \leq f(n)$ and $r = \Theta(\frac{1}{f(n)})\mathrm{gh}(\Lambda)$ while $g \in \Theta(f)$. The two lattices are then defined as follows:[1]

$$\Lambda_S \coloneqq g\Lambda \oplus (g+1)\Lambda \quad \text{and} \quad \Lambda'_Q \coloneqq \Lambda' \oplus g(g+1)\Lambda'.$$

Λ_S is decodable up to a radius of $r' = gr$ by decoding the first $n/2$ dimensions and the second $n/2$ dimensions of a point separately. On the other hand, Λ'_Q contains a rank-$n/2$ dense sublattice in its first $n/2$ dimensions. For Theorem 5.1 to apply, we need $2\eta_{1/2}(D^T Q D) = 2\eta_{1/2}(\Lambda') \leq r'/\sqrt{n}$. We have

$$2\eta_{\frac{1}{2}}(\Lambda') \leq 2\eta_{2^{-n}}(\Lambda') \overset{\text{Lem. 2.2}}{\leq} \frac{2\sqrt{n}}{\lambda_1(\Lambda'^*)}.$$

[1] Here, the $\oplus$ operator refers to the orthogonal concatenation (i.e., the Cartesian product) of two lattices of dimension $n/2$ to form a lattice of dimension n. The term $(g+1)$ can be relaxed to any $\tilde{g} \geq g$ coprime to g as done by Branco, Malavolta, and Maradni [22], but the choice of $\tilde{g} = g + 1$ optimizes the gap of the resulting construction.

By applying $\mathrm{gh}(\Lambda'^*) \leq \Theta(\lambda_1(\Lambda'^*))$ for $\mathrm{gap}(\Lambda') \in \Theta(1)$, we find that

$$2\eta_{\frac{1}{2}}(\Lambda') \leq \frac{2\sqrt{n}}{\lambda_1(\Lambda'^*)} = \Theta\left(\frac{\sqrt{n}}{\mathrm{gh}(\Lambda'^*)}\right) \overset{\text{Def. 2.18}}{=} \Theta\left(\det(\Lambda'^*)^{\frac{-1}{n}}\right) = \Theta\left(\det(\Lambda')^{\frac{1}{n}}\right)$$
$$= \Theta\left(\det(\Lambda)^{\frac{1}{n}}\right) \overset{\text{Def. 2.18}}{=} \Theta\left(\frac{\mathrm{gh}(\Lambda)}{\sqrt{n}}\right) = \Theta\left(\frac{f(n)r}{\sqrt{n}}\right) = \Theta\left(\frac{g(n)r}{\sqrt{n}}\right) = \Theta\left(\frac{r'}{\sqrt{n}}\right),$$

so, following van Woerden [92], the requirements of Theorem 5.1 are satisfied by setting S as the quadratic form of Λ_S and Q as that of Λ'_Q with a sufficiently large $g(n) \in \Theta(f(n))$. We refer to Ducas and van Woerden [33] for the proof that Λ_S and Λ'_Q have matching genera. By construction, the gaps of both lattices are in $O(f(n))$. Candidates for Λ that have been used to implement ΔLIP-based schemes include the trivial lattice $\mathbb{Z}^n$ [34] and the Barnes-Wall lattices [29].

5.2 Requirements

This section justifies our use of the LIP-KEM as the base scheme upon which we aim to build a CCA2-secure KPKE or KKEM from LIP using k-repetition. We also establish requirements that any such construction must satisfy to be successful. In addition, this section introduces our strategy for exploring ways to meet these requirements.

Recall that we aim to construct a scheme with a structure like that of Definition 4.1 that is IND-CPA-secure and verifiable as defined in Definition 4.2. To accomplish this using LIP, we first consider existing cryptographic constructions based on LIP to determine which approaches lend themselves to constructing a PKE or KEM that could be turned into a PKE_k or KEM_k. In doing so, we find that existing work can be classified into two major categories:

- "Hidden Points": This first category of constructions is based on sampling lattice points or small errors in a lattice and using LIP to prevent an adversary from decoding them. This includes the LIP-KEM as well as a very similar signature scheme also by Ducas and van Woerden [33] that has been implemented in form of the NIST candidate HAWK [34]. A single-bit PKE that is structurally almost identical to the LIP-KEM was proposed by Ackermann, Roux-Langlois, and Wallet [1]. Two homomorphic encryption schemes based on LIP can also be found in the literature; both are based on this paradigm [22, 51].
- "Isomorphisms": These schemes are built around the lattice isomorphism directly and do not involve any points in the lattice. In this category, the literature features a zero-knowledge proof-of-knowledge protocol by Ducas and van Woerden [33] analogous to the well-known protocol for graph isomorphisms [41] and an interpretation of lattice isomorphisms as a group action by Benčina et al. [11]. This interpretation allows lattice isomorphisms to be used in cryptographic constructions similar to group-based

cryptography [5]. The LIP group action has been used by multiple authors [45, 53] to instantiate commitment schemes.

Of these two categories, the first is more promising since it already contains schemes for public-key encryption. The second category has produced no such schemes to the best of our knowledge. In addition, Budroni, Chi-Domínguez, and Franch [23] show that lattice isomorphisms do not satisfy multiple common notions required for some group-action-based schemes. This limits their utility in group action constructions. Within the first category of constructions, the LIP-KEM is closest to the desired outcome of a verifiable and IND-CPA-secure KEM_k or PKE_k.[2] We thus base our investigation into the construction of a k-repeated KEM or PKE mainly on the existing LIP-KEM. The goal is to adapt it to make it verifiable without losing its IND-CPA security.

Our exploration of different approaches to making a k-repeated LIP-KEM_k is guided by the structure of the k-repetition framework. To achieve the required IND-CPA security and verifiability, we demand three core properties of any potential instantiation: Some component of each instance must be correlated with the others, its IND-CPA security proof must feature a "long jump", and the instances' public keys must be independent of each other. The reason for each of these properties is explained in the following:

Correlated components: In order to implement the Dec_{Ver} algorithm, having the secret key to one of the instances must be enough to verify that each of the other ciphertexts also decrypts to the same key or message. Consider the case where each instance is completely independent: In order to achieve IND-CPA security for the entire k-repeated scheme, the instances themselves would have to be IND-CPA-secure. Implementing an efficient Dec_{Ver} for this scheme would be impossible since the IND-CPA security of the individual instances explicitly rules out verifying the contents of the ciphertexts. Therefore, the instances must be correlated somehow: In the abstract, decrypting one instance allows one to learn a common *verification secret* that can then be used to verify the other ciphertexts.[3] On the other hand, an adversary that knows none of the secret keys would not learn this common secret.

Long jump: We show that a k-repeated PKE or KEM cannot be proven IND-CPA-secure by any proof that only features hybrid arguments iterating over the k instances. Any valid proof must instead contain at least one step of a non-hybrid nature, i.e., a *long jump* that crosses all k instances of the scheme at once.

Hybrid arguments are the standard method in the literature for proving the security of schemes that contain polynomially many repeated copies of the same cryptographic primitive [64]. In a hybrid argument, the indistinguishability of two k-repeated ensembles of

[2] Ackermann, Roux-Langlois, and Wallet's [1] one-bit PKE is structurally almost identical to the LIP-KEM. We choose the latter since it offers more flexibility in the definition of its ciphertexts.

[3] Rosen and Segev [84] show that it is also possible to construct k-repetition from a more general t-to-all correlation where t secret keys must be known to verify all of the ciphertexts. We do not find any approaches where this would be helpful.

random variables $(X, \ldots, X)$ and $(Y, \ldots, Y)$ with $X \overset{c}{\approx} Y$ and distributions $\mathcal{D}_X$ and $\mathcal{D}_Y$ is proven through a series of hybrids

$$\begin{aligned} H_0 &:= (X, X, \ldots, X, X), \\ H_1 &:= (Y, X, \ldots, X, X), \\ &\vdots \\ H_{k-1} &:= (Y, Y, \ldots, Y, X), \\ H_k &:= (Y, Y, \ldots, Y, Y). \end{aligned}$$

The goal of the argument is to construct an attacker $\mathcal{B}$ on the indistinguishability of X from Y based on an attacker $\mathcal{A}$ that tries to distinguish H_0 from H_k. This is typically done by making $\mathcal{B}$ embed its challenge z into the repeated ensembles at a random position (compare the textbook definition by Mittelbach and Fischlin [64]):

$\mathcal{B}(z)$

$i \leftarrow\!\!\$ \{1, \ldots, k\}$
for $j \in \{1, \ldots, i-1\}$ **do**
  $z'_j \leftarrow \mathcal{D}_Y$
$z'_i \leftarrow z$
for $j \in \{i+1, \ldots, k\}$ **do**
  $z'_j \leftarrow \mathcal{D}_X$
return $\mathcal{A}(z')$

Assume that the IND-CPA security proof for a PKE_k or KEM_k consists of a sequence of hybrid arguments beginning at the honest, original scheme and ending in a game where $\mathcal{A}$ cannot infer the bit b from the ciphertext $(c_1, \ldots, c_k)$. In each argument, $\mathcal{B}$ must generate $k-1$ of the k challenge ciphertexts c_i that it outputs to $\mathcal{A}$ on its own. Since the instances of the scheme must have some correlated component, we also know that there is a verification secret common to all of the instances in the scheme. We inductively reason that $\mathcal{B}$ knows the verification secret in each hybrid argument and that the proof cannot be valid.

In the first hybrid argument, $\mathcal{B}$ must generate the ciphertexts $c_{i+1}, \ldots, c_k$ honestly. This means $\mathcal{B}$ must know the verification secret and can verify that the $c_1, \ldots, c_{i-1}$ as well as c_i form a valid ciphertext together with the $c_{i+1}, \ldots, c_k$. Notably, the hybrid argument cannot "break compatibility" with the verification secret: Otherwise, $\mathcal{B}$ could distinguish $c_i \sim \mathcal{D}_X$ from $c_i \sim \mathcal{D}_Y$. The same argument can be applied to each subsequent hybrid argument. In the final hybrid argument, $c_1, \ldots, c_{i-1}$ must be independent of the bit b in $\mathcal{A}$'s IND-CPA game — otherwise, the proof wouldn't be valid. It follows that these first $i-1$ ciphertexts are also independent of the challenge message m (or key ek). On the other hand, $c_{i+1}, \ldots, c_k$ are not yet independent of the challenge m or ek because the proof would have been complete after the previous argument if they were. As previously established, $\mathcal{B}$ has access to the verification secret and can verify that $(c_1, \ldots, c_k)$ forms a valid ciphertext. However, this contradicts $c_1, \ldots, c_{i-1}$ being independent of m or ek — if ciphertexts c_j that are completely independent of the message or key could pass the verification check, $\textsc{Dec}_{\textsc{Ver}}$ would not satisfy Definition 4.2. We conclude that the proof must be invalid.

The core issue with a purely hybrid proof is that the verifiability property of the scheme can never be broken. Doing so requires a non-hybrid long jump after which either the proof is completed or any further hybrid arguments leave $\mathcal{B}$ without the ability to verify the ciphertexts. Accordingly, every standard-model k-repetition framework in the literature makes use of a long jump that involves concatenating the k instances of the scheme into one large instance that can be worked with as a whole: Rosen and Segev [84] use a special property of lossy TDOWFs that lets them combine all of them into one large lossy TDOWF, while Döttling et al. [31] and Persichetti [77] concatenate k matrices and error vectors into one large matrix and error vector. We have shown that these similarities are not a coincidence, but a necessity for a successful proof.

Independent public keys: The final requirement of any k-repeated scheme is that the public keys of the scheme must be sampled independently without a common secret. This is because of the structure of the final KPKE or KKEM's proof: In it, the IND-CPA challenger $\mathcal{C}$ for the PKE_k or KEM_k generates k keypairs and gives the attacker $\mathcal{B}$ the public keys. $\mathcal{B}$ must then generate k additional complementary keypairs and give the $2k$ total public keys to the IND-CCA2-attacker $\mathcal{A}$. Since the sequence of public keys used in an encryption is determined by the bits of the verification key, the public keys generated by $\mathcal{B}$ and $\mathcal{C}$ will mix in both the challenge ciphertext and any ciphertexts submitted to the decryption oracle by $\mathcal{A}$ (compare Figure 4.1). $\mathcal{B}$ does not know the common secret $\mathcal{C}$ used to generate its public keys, so $\mathcal{B}$ cannot use the same secret. Ciphertexts would use mismatched public keys in this scenario, breaking the assumptions of the k-repeated scheme and thus invalidating the KPKE or KKEM's security proof.

The three requirements above inform our attempts to realize a PKE_k or KEM_k from the LIP-KEM. This is especially true of the first requirement: To satisfy it, some component of the LIP-KEM must be correlated across instances in any approach. Given that the public keys must remain independent, there are only three components that lend themselves to correlation: The error term $\boldsymbol{e}$ (Section 5.3), the lattice point $\boldsymbol{x}$ (Section 5.4), or some external component added to the KEM (Section 5.5).[4] We proceed to consider each of these options in turn in the following sections.

5.3 Correlated Errors

One component of the LIP-KEM that might be correlated in order to create a KEM_k is the error term $\boldsymbol{e}$. In the course of this section, we demonstrate that this approach is unlikely to result in achieving both IND-CPA security and verifiability. To do so, we establish a common structure for a LIP-KEM_k based on correlated errors and show that several candidate distributions for these error terms do not meet our requirements. Finally, we conjecture that there is no efficient sampling method for correlated errors that would lead to an IND-CPA-secure, verifiable LIP-KEM_k.

[4] While the extractor seed Z is also a component that could theoretically be correlated across instances, doing so is unlikely to be fruitful.

Recall that the LIP-KEM as defined in Algorithm 5.1 works by sampling an error term $\boldsymbol{e} \leftarrow q^{-1}\mathcal{D}_{P,qr/\sqrt{n}}$ which is hidden in the ciphertext and from which the encapsulated key is derived. To correlate these errors across multiple instances of the LIP-KEM, we have to sample the error terms $(\boldsymbol{e}_i)_{i\in[k]}$from some distribution that lets us implement $\textsc{Dec}_{\textsc{Ver}}$: Given only one $\boldsymbol{e}_j$, we must be able to reconstruct all k error terms.[5] We formalize this notion using two algorithms:

Definition 5.1 (Correlated Error Sampler)
A *correlated error sampler (CES)* with the codomain $E \subseteq \mathbb{R}^n$ is a tuple of PPT algorithms (GenError, Recons) such that:

- $\textsc{GenError}((P_i)_{i\in[k]}, r, s) \to (\boldsymbol{e}_i)_{i\in[k]} \in E^k$, which samples k correlated error terms for k instances of the LIP-KEM. Here, the P_i are quadratic forms sampled as $(P_i)_{i\in[k]} \leftarrow (\mathcal{D}_s([S]))_{i\in[k]}$ for some quadratic form S and parameter s. The parameters r and s have the same purpose as in the original LIP-KEM: r represents the decoding radius of S, while $s \geq \eta_\epsilon(S)$ is an upper bound on the smoothing parameter. GenError returns k error terms, one for each instance.
- $\textsc{Recons}((P_i)_{i\in[k]}, (\boldsymbol{c}_i)_{i\in[k]}, \boldsymbol{e}_j) \to (\boldsymbol{e}_i)_{i\in[k]}$. This algorithm reconstructs the k error terms from the public keys P_i, LIP-KEM ciphertexts $\boldsymbol{c}_i = \boldsymbol{e}_i \bmod \mathbb{Z}^n \in E/\mathbb{Z}^n$, and one known error term $\boldsymbol{e}_j \in E$.[6] If reconstruction fails, Recons outputs $(\perp)_{i\in[k]}$.

In addition, we demand that a CES be *correct* and *consistent*:

- A CES is correct if, for any quadratic form S, $(P_i)_{i\in[k]} \leftarrow (\mathcal{D}_s([S]))_{i\in[k]}$, $s \geq \eta_\epsilon(S)$ and r as the decoding radius of S and $(\boldsymbol{e}_i)_{i\in[k]} \leftarrow \textsc{GenError}((P_i)_{i\in[k]}, r, s)$, $(\boldsymbol{c}_i)_{i\in[k]} = (\boldsymbol{e}_i \bmod \mathbb{Z}^n)_{i\in[k]}$, it holds that
  - $\|\boldsymbol{e}_i\|_{P_i} \leq r \ \forall i \in [k]$ and
  - $\textsc{Recons}((P_i)_{i\in[k]}, (\boldsymbol{c}_i)_{i\in[k]}, \boldsymbol{e}_j) = (\boldsymbol{e}_i)_{i\in[k]} \ \forall j \in [k]$.
- A CES is consistent if, for any quadratic form S, public keys $(P_i)_{i\in[k]} \leftarrow (\mathcal{D}_s([S]))_{i\in[k]}$, $(\boldsymbol{c}_i)_{i\in[k]} \in (E/\mathbb{Z}^n)^k$, and $(\boldsymbol{e}_i)_{i\in[k]} \in E^k$,

$$\textsc{Recons}((P_i)_{i\in[k]}, (\boldsymbol{c}_i)_{i\in[k]}, \boldsymbol{e}_1) = \cdots = \textsc{Recons}((P_i)_{i\in[k]}, (\boldsymbol{c}_i)_{i\in[k]}, \boldsymbol{e}_k).$$

While correctness ensures that honestly generated error terms can be reconstructed correctly, consistency demands that error terms that may have been chosen adversarially still reconstruct to a consistent set of error terms. We require this property to ensure the output of $\textsc{Dec}_{\textsc{Ver}}$ is independent of which secret key is given. Note that Recons is permitted to fail (i.e., by outputting $\perp$ or incorrect $\boldsymbol{e}_i$ with $\|\boldsymbol{e}_i\|_{P_i} > r$ or $\boldsymbol{c}_i \neq \boldsymbol{e}_i \bmod \mathbb{Z}^n$) for adversarial inputs as long as it remains consistent in doing so.

[5] Note that, while simply using each encapsulated key ek_j to encrypt all the other $\boldsymbol{e}_i$ and attaching those ciphertexts to the LIP-KEM$_k$ would trivially permit verification, doing so breaks the IND-CPA proof since the $\boldsymbol{e}_i$ have no entropy if they are recoverable from the ciphertexts. We discuss the general category of approaches similar to this one in Section 5.5.

[6] As in Definition 4.2, we suppress the index input to Recons.

Given a CES, we can define a LIP-KEM$_k$ as in Algorithm 5.2 with correlated errors analogously to the original LIP-KEM by replacing the error sampling with GenError and extracting the encapsulated key from all k errors: $ek \leftarrow \mathcal{E}((\boldsymbol{e}_i)_{i\in[k]}, Z)$. Dec$_{\text{Ver}}$ can be defined in terms of Recons by using the secret key U_j to decode one error term $\boldsymbol{e}_j$ and running Recons on that error term to recover the others. To ensure that Recons's output remains in sync with that of Decaps, Decaps also runs Recons using $\boldsymbol{e}_1$ and outputs $\perp$ if the output is different from its own decoded result.

Lemma 5.2. *A LIP-KEM$_k$ as defined in Algorithm 5.2 with a correct and consistent CES is both correct and verifiable.*

Proof. Correctness follows from the correctness of the original LIP-KEM and the CES — since GenError always outputs error terms with $\|\boldsymbol{e}_i\|_{P_i} \leq r$, each error term is uniquely decodable, so Decaps will output the same ek as Encaps. To see that the LIP-KEM$_k$ is verifiable, we show that

$$\text{Decaps}((U_i)_{i\in[k]}, ((\boldsymbol{c}_i)_{i\in[k]}, Z)) \neq \perp \iff \text{Dec}_{\text{Ver}}((P_i)_{i\in[k]}, ((\boldsymbol{c}_i)_{i\in[k]}, Z), U_j) = 1$$

for honestly sampled public and private keys and any ciphertext.

If Decaps outputs a valid key, then every $\boldsymbol{c}_i$ must have uniquely decoded to a valid $\boldsymbol{e}_i$ such that Recons$((P_i)_{i\in[k]}, (\boldsymbol{c}_i)_{i\in[k]}, \boldsymbol{e}_1) = (\boldsymbol{e}_i)_{i\in[k]}$ by construction. In this case, Dec$_{\text{Ver}}$ will also decode $\boldsymbol{c}_j$ to the same $\boldsymbol{e}_j$. Since the CES is consistent, Recons called with $\boldsymbol{e}_j$ will have the same output as when called with $\boldsymbol{e}_1$ in Decaps, which means both Dec$_{\text{Ver}}$ and Decaps will end up with the same $(\boldsymbol{e}_i)_{i\in[k]}$.

On the other hand, if Dec$_{\text{Ver}}$ outputs 1, the $(\boldsymbol{e}_i)_{i\in[k]}$reconstructed by Recons correspond to the unique decodings of $(\boldsymbol{c}_i)_{i\in[k]}$since there is only one $\boldsymbol{e}_i$ with $\|\boldsymbol{e}_i\|_{P_i} \leq r$ and $\boldsymbol{c}_i = \boldsymbol{e}_i \bmod \mathbb{Z}^n$. Therefore, Decaps will calculate the same $(\boldsymbol{e}_i)_{i\in[k]}$as Dec$_{\text{Ver}}$. □

While Lemma 5.2 gives us correctness and verifiability for a LIP-KEM$_k$, we also require IND-CPA security. Since the encapsulated key ek is derived using an extractor, the goal of any security proof must be to reach a state where $(\boldsymbol{e}_i)_{i\in[k]}$contains a polynomial amount of entropy — only then is the extractor guaranteed to produce a key indistinguishable from randomness. Note that this step also meets our requirements for a *long jump* in the proof, thus satisfying the second criterion for a successful LIP-KEM$_k$. However, while we are able to find several error distributions leading to a correct and consistent CES, none of them produce the required entropy in a k-repeated setting. We discuss the potential candidates and their flaws in the following. In short, these candidates involve either using the same discrete Gaussian distribution on the lattices and correlating the sampled errors directly or sampling one error term in a common space and transforming it into each of the lattice spaces.

Algorithm 5.2 LIP-KEM$_k$ with correlated errors

▷ **Parameters:**
- Quadratic form $S \in S_n^{>0}(\mathbb{Z})$
- Decoding radius $r < \lambda_1(S)/2$ of S
- Efficient decoding algorithm $\textsc{Decode}(\boldsymbol{z}) \to \boldsymbol{x}$ for S with decoding radius r
- $(l, \mathrm{negl}(n))$-extractor $\mathcal{E}\colon \frac{1}{q}\mathbb{Z}^n \times \{0,1\}^z \to \{0,1\}^l$ with $l \in \Theta(n)$
- $s \geq \max\left\{\lambda_n(S),\ \left\|\widetilde{B}_S\right\| \cdot \sqrt{\dfrac{\ln(2n^2+4n)}{\pi}}\right\}$
- A correlated error sampler ($\textsc{GenError}$, $\textsc{Recons}$)
- *Note: Index ranges in the following implicitly have $i \in [k]$.*

function $\textsc{Gen} \to (pk, sk)$
  $(P_i)_i \leftarrow (\mathcal{D}_s([S]))_i$ with $U_i \in \mathrm{GL}_n(\mathbb{Z}),\ U_i^T S U_i = P_i\ \forall i$
  return $(pk, sk) = ((P_i)_i, (U_i)_i)$

function $\textsc{Encaps}((P_i)_i) \to (c, ek)$
  $(\boldsymbol{e}_i)_i \leftarrow \textsc{GenError}((P_i)_i, r, s)$
  $(\boldsymbol{c}_i)_i \leftarrow (\boldsymbol{e}_i \bmod \mathbb{Z}^n)_i$
  $Z \leftarrow\!\!\$\ \{0,1\}^z$
  $ek \leftarrow \mathcal{E}((\boldsymbol{e}_i)_i, Z)$
  return $(((\boldsymbol{c}_i)_i, Z), ek)$

function $\textsc{Decaps}((U_i)_i, ((\boldsymbol{c}_i)_i, Z)) \to ek$ or $\perp$
  $(\boldsymbol{y}_i)_i \leftarrow (\textsc{Decode}(S, U_i\boldsymbol{c}_i))_i$
  if $\exists i\colon \|\boldsymbol{y}_i - U_i\boldsymbol{c}_i\|_S \geq r$ or decoding fails, **then return** $\perp$
  $(\boldsymbol{e}_i)_i \leftarrow \left(\boldsymbol{c}_i - U_i^{-1}\boldsymbol{y}_i\right)_i$
  if $\textsc{Recons}((P_i)_i, (\boldsymbol{c}_i)_i, \boldsymbol{e}_1) \neq (\boldsymbol{e}_i)_i$ **then return** $\perp$
  $ek \leftarrow \mathcal{E}\left((\boldsymbol{e}_i)_i, Z\right)$
  return ek

function $\textsc{Dec}_{\textsc{Ver}}((P_i)_i, ((\boldsymbol{c}_i)_i, Z), U_j) \to b$
  $\boldsymbol{y}_j \leftarrow \textsc{Decode}(S, U_j\boldsymbol{c}_j)$
  $\boldsymbol{e}_j \leftarrow \boldsymbol{c}_j - U_j^{-1}\boldsymbol{y}_j$
  $(\boldsymbol{e}_i)_i \leftarrow \textsc{Recons}((P_i)_i, (\boldsymbol{c}_i)_i, \boldsymbol{e}_j)$
  if $\forall i\colon \boldsymbol{e}_i \neq \perp$ and $\|\boldsymbol{e}_i\|_{P_i} \leq r$ and $\boldsymbol{c}_i = \boldsymbol{e}_i \bmod \mathbb{Z}^n$ **then return** 1
  else return 0

5.3.1 Discrete Gaussians in the Lattice

The original LIP-KEM draws its error term as $\boldsymbol{e} \leftarrow q^{-1}\mathcal{D}_{P,s'}$ for some parameter s'. Extending this procedure to construct a CES is thus a natural approach. In addition, we sample each quadratic form public key $P_i \leftarrow \mathcal{D}_s([S])$ isomorphic to the same quadratic form S. This means each P_i shares the same geometry and suggests compatibility between these quadratic forms. We show that this is not the case — in fact, the hardness of LIP itself prevents us from using $\mathcal{D}_{P_i,s'}$ to sample correlated error terms. Consider the following attempts at using error terms of the form $\boldsymbol{e}_i \leftarrow a\mathcal{D}_{Q,s'}$ for some $a \in \mathbb{R}^n$ and $Q \in \{P_1, \ldots, P_k\}$ for a CES:

Using the same or transformed error terms: It is obviously infeasible to let GenError sample one error term $\boldsymbol{e}_1 \leftarrow q^{-1}\mathcal{D}_{P_1,s'}$ and output $(\boldsymbol{e}_1)_{i\in[k]}$— since each quadratic form $P_i = U_i^T S U_i$ for some unimodular U_i, having $\|\boldsymbol{e}_1\|_{P_1} = \|U_1\boldsymbol{e}_1\|_S \leq r$ says nothing about $\|\boldsymbol{e}_1\|_{P_i} = \|U_i\boldsymbol{e}_1\|_S$ for $i \neq 1$. Unimodular matrices can affect the lengths of coefficient vectors dramatically, which is important for the hardness of LIP: If $\|x\|_{Q_1} \approx \|x\|_{Q_2}$ were a property of isomorphic quadratic forms Q_1 and Q_2 for some $x \in \mathbb{Z}^n$, it could be used to solve ΔLIP. Using the same error term across multiple instances would therefore lead to an incorrect CES.

In a similar vein, one could try to sample one error term $\boldsymbol{e}_1 \leftarrow \mathcal{D}_{P_1,s'}$ and then transform that term from P_1 "into" the other lattices. The goal would be to have a transformation $T_i(\boldsymbol{e})$ such that $\|T_i(\boldsymbol{e}_1)\|_{P_i} = \|\boldsymbol{e}_1\|_{P_1} \leq qr$. However, this is also unlikely to be compatible with the hardness of LIP since T_i would have to act similarly to $U_iU_1^{-1}$. For the special case of linear transformations that map lattice points to lattice points, we can prove that this approach breaks sLIP:

Lemma 5.3. *Given an instance $P \leftarrow \mathcal{D}_s([S])$ of ac-sLIP$_s^S$ and a linear transformation M with $\|M\boldsymbol{x}\|_S = \|\boldsymbol{x}\|_P$ and $M\boldsymbol{x} \in \mathbb{Z}^n$ for all $\boldsymbol{x} \in \mathbb{Z}^n$, M is a solution to the instance.*

Proof. By assumption, we have

$$\boldsymbol{x}^T M^T S M \boldsymbol{x} = \boldsymbol{x}^T U^T S U \boldsymbol{x} \iff \boldsymbol{x}^T (M^T S M - U^T S U)\boldsymbol{x} = 0.$$

Since this equation holds for all $\boldsymbol{x} \in \mathbb{Z}^n$, we have $M^T S M - U^T S U = 0$ and thus $M^T S M = U^T S U$. In addition, the requirement that M is to map all integer vectors to integer vectors implies $M \in \mathbb{Z}^{n\times n}$. Finally,

$$\det(M)^2 \det(S) = \det(U)^2 \det(S) = \det(S) \implies \det(M) \in \{-1, 1\},$$

so $M \in \mathrm{GL}_n(\mathbb{Z})$ and therefore a valid solution to the ac-sLIP$_s^S$ instance. □

It stands to reason that a transformation of a more general class or one that only works for terms sampled from $\mathcal{D}_{P_1,s'}$ specifically would still leak information about U_1 and U_i and would thus be hard to compute.

Correlating the sampler: Since using error terms sampled from $\mathcal{D}_{P,s'}$ in a black-box manner does not lead to a working CES, one might try to modify the sampling algorithm SampleD in Algorithm 3.2 to produce correlated outputs instead. In essence, this means letting

$$\text{GenError}((P_i)_{i\in[k]}, r, s) \to (\text{SampleD}'(P_i, s', 0; r))_{i\in[k]}$$

for some common random seed r and a modified sampler SampleD′. Unfortunately, there does not appear to be a useful way to incorporate r such that the output $\boldsymbol{e}_i$ of SampleD′ satisfies $\|\boldsymbol{e}_i\|_{P_i} \leq r$ as well as the entropy requirement for the IND-CPA proof while becoming reconstructable for Recons. If SampleD were an invertible sampler, meaning that its input randomness could be reconstructed given its output [27], Recons would not pose an issue. However, SampleD heavily relies on rejection sampling, making it not invertible.[7] We argue that making Recons possible would require removing the rejection steps from the algorithm. These are line 8 in Sample1D (Algorithm 3.1) and line 12 in SampleD (Algorithm 3.2). While line 12 can be removed from SampleD while keeping the output distribution statistically close to the discrete Gaussian (as done by Gentry, Peikert, and Vaikuntanathan [38]), we are not aware of any sampler without a rejection step in Sample1D. Even if the required rejection steps were to be removed, the width of the distribution of the one-dimensional samples produced by Sample1D for each dimension depends on the quadratic form,[8] further complicating reconstruction: These samples cannot be directly mapped to each other since they are each rounded to the nearest integer, with narrower distributions losing more information about the real-valued sample in the rounding process than wider distributions. Allowing for the reconstruction of the other instances' error terms or r while retaining the discrete Gaussian output distribution appears to be infeasible.[9]

In any case, the potential proof strategy for an entropy argument in the IND-CPA proof of any construction of this kind remains unclear: If SampleD's output distribution is changed, the minimum-entropy guarantee of the discrete Gaussian distribution would no longer apply and entirely new bounds for the new distribution would have to be proven. However, even if a change in distribution could be avoided, the approach presents further challenges: Assume that SampleD is adjusted such that the new SampleD′ still samples $\mathcal{D}_{P_i,s',c}$ for $k = 1$ and Recons can be implemented. Any argument establishing the minimum entropy of $(\boldsymbol{e}_i)_{i\in[k]}$given $(\boldsymbol{c}_i)_{i\in[k]}$for $k > 1$ would have to involve the behavior of SampleD′ under the common seed r: The algorithm needs to output $(\boldsymbol{e}_i)_{i\in[k]}$such that there are exponentially many common seeds r' that would lead to the same $(\boldsymbol{c}_i)_{i\in[k]} = (\boldsymbol{e}_i \bmod \mathbb{Z}^n)_{i\in[k]}$, which is a strong requirement. Modifying SampleD will thus most likely not lead to a satisfactory CES.

[7] In addition, Ishai et al. [44] show that not all sampling algorithms can be made invertible under common cryptographic assumptions.

[8] Compare Figure 3.2, noting that the s'_i-input to Sample1D is given by $s/\|\widetilde{\boldsymbol{b}}_i\|_Q$, where $\widetilde{\boldsymbol{b}}_i$ is the i-th coefficient vector of the Gram-Schmidt-orthogonalization of Q.

[9] Of course, given that the P_i are all isomorphic, it is technically possible to map the samples from one instance to another by "matching up the right dimensions". Trying to find the appropriate mapping, however, brings us back to Lemma 5.3.

5.3.2 Discrete Gaussians in Euclidean Space

In the previous section, we show that it is not feasible to build a CES from $\mathcal{D}_{P_i,s'}$, i.e., the discrete Gaussian distribution on the lattices P_i. In this section, we discuss the possibility of using other distributions for sampling Gaussian-distributed error terms in GenError. The motivation for this approach is that, while sampling correlated Gaussian-distributed *lattice point coefficients* for different lattices may not be efficiently possible, choosing a different discretization for the Gaussian can make correlation and reconstruction simple in exchange for no longer producing integer coefficient vectors. This leaves establishing the required entropy of the sampled $(\boldsymbol{e}_i)_{i\in[k]}$as the only remaining obstacle to building a working CES. As we demonstrate in the following, said obstacle is nevertheless insurmountable: No lattice-independent discretization of the Gaussian distribution (and, by extension, no lattice-independent discrete distribution over $\mathbb{R}^n$) can achieve the required minimum entropy for a CES. Throughout this section, we switch to the lattice formulation of LIP: Public keys are considered to be of the form $B_i = O_i B U_i$ with $O_i \in O_n(\mathbb{R})$ and $U_i \in \mathrm{GL}_n(\mathbb{Z})$ for some full-rank lattice basis B.

As previously established, sampling the discrete Gaussian distribution $q^{-1}\mathcal{D}_{B_i^T B_i,s'}$ outputs lattice point coefficients $\boldsymbol{e}_i$ with $q\boldsymbol{e}_i \in \mathbb{Z}^n$ such that the probability of sampling $\boldsymbol{e}_i$ is proportional to a Gaussian function of $\|\boldsymbol{e}_i\|_{B_i^T B_i}$. Equivalently, $q\boldsymbol{v}_i = qB_i\boldsymbol{e}_i$ is a lattice point which is sampled with a probability proportional to a Gaussian function of $\|\boldsymbol{v}_i\|_2$. We take advantage of this equivalence by reversing it and sampling $\boldsymbol{v}_i \in \mathbb{R}^n$ according to a Gaussian of $\|\boldsymbol{v}_i\|_2$, then letting $\boldsymbol{e}_i \leftarrow B_i^{-1}\boldsymbol{v}_i$.[10] Achieving a correlation between the $\boldsymbol{e}_i$ is as simple as setting $\boldsymbol{v}_1 = \cdots = \boldsymbol{v}_k = \boldsymbol{v}$ for one common $\boldsymbol{v}$. After this, we can reconstruct all the $\boldsymbol{e}_i$ from one $\boldsymbol{e}_j$ through $\boldsymbol{e}_i = B_i^{-1}B_j\boldsymbol{e}_j$. Of course, multiplying by the real-valued B_i^{-1} means that we generally no longer have $c\boldsymbol{e}_i \in \mathbb{Z}^n$ for a constant $c \in \mathbb{R}$, so the $\boldsymbol{e}_i$ are not scaled coefficient vectors of lattice points anymore.

The canonical Gaussian distribution over $\mathbb{R}^n$ is the continuous multivariate normal distribution. However, its continuous nature make this distribution impossible to sample exactly, forcing us to choose some discretization of $\mathbb{R}^n$ as the support for our distribution instead.[11] We choose $q^{-1}\mathcal{D}_{\mathbb{1}_n,s'}$ (i.e., the discrete Gaussian on $\mathbb{Z}^n$ scaled down by a factor of q) as our model discretized Gaussian distribution with the same $s' = qr/\sqrt{n}$ as in the original LIP-KEM. This allows us to reuse Lemma 2.6 to show that $\|\boldsymbol{e}_i\|_{B_i^T B_i} = \|\boldsymbol{v}\|_2 \leq r$, guaranteeing correctness. However, the following argument shows that, even for $k = 1$ and a lattice B with a dense sublattice as in Theorem 5.1, $\boldsymbol{e}_1 = B_1^{-1}\boldsymbol{v}$ cannot have the minimum entropy required for an IND-CPA proof given $\boldsymbol{c}_1$:

Recall from the proof of Theorem 5.1 that the IND-CPA attacker on the LIP-KEM receives the public key ($B_1 = O_1 B U_1$ instead of $P_1 = U_1^T S U_1$ in the lattice formulation) and the challenge ciphertext $\boldsymbol{c}_1 = \boldsymbol{e}_1 \bmod \mathbb{Z}^n$, but not the true $\boldsymbol{e}_1$ nor the true $\boldsymbol{x}_1 = \boldsymbol{c}_1 - \boldsymbol{e}_1$. The attacker can

[10] B_i is full-rank and therefore invertible over $\mathbb{R}^n$.

[11] While it is common to ignore the precision issues presented by real-valued numbers in the field of lattice cryptography [15, 59, 82], the distinction between a discrete support and a continuous support is of key importance for the following argument.

thus consider the possible candidates $\boldsymbol{e}'_1 = \boldsymbol{c}_1 - \boldsymbol{x}'_1$ for $\boldsymbol{e}_1$ knowing that $\boldsymbol{e}_1 \leftarrow q^{-1}B_1^{-1}\mathcal{D}_{\mathbb{1}_n,s'}$. From the distribution's definition,

$$\Pr_{qB_1\boldsymbol{e}_1\sim\mathcal{D}_{\mathbb{1}_n,s'}}[qB_1\boldsymbol{e} = qB_1\boldsymbol{e}'] = \frac{\rho_{\mathbb{1}_n,s'}(qB_1\boldsymbol{e}')}{\rho_{\mathbb{1}_n,s'}(\mathbb{Z}^n)} \text{ if } qB_1\boldsymbol{e}' \in \mathbb{Z}^n, \text{ and 0 otherwise.}$$

Though unlike in the LIP-KEM's proof, we now have

$$\begin{aligned}
& \boldsymbol{e}'_1 = \boldsymbol{c}_1 - \boldsymbol{x}'_1 \\
\iff \quad & \boldsymbol{e}'_1 = \boldsymbol{c}_1 - \boldsymbol{x}_1 + (\boldsymbol{x}_1 - \boldsymbol{x}'_1) \\
\iff \quad & \boldsymbol{e}'_1 = \boldsymbol{e}_1 + (\boldsymbol{x}_1 - \boldsymbol{x}'_1) \\
\iff & qB_1\boldsymbol{e}'_1 = qB_1\boldsymbol{e}_1 + qB_1(\boldsymbol{x}_1 - \boldsymbol{x}'_1) \\
\iff & qB_1\boldsymbol{e}'_1 = \underbrace{\boldsymbol{v}}_{\in\mathbb{Z}^n} + \underbrace{qB_1}_{\in\mathbb{R}^{n\times n}}\underbrace{(\boldsymbol{x}_1 - \boldsymbol{x}'_1)}_{\in\mathbb{Z}^n}.
\end{aligned} \tag{5.1}$$

A uniformly random orthogonal matrix $O_1 \in O_n(\mathbb{R})$ will always have exclusively irrational entries since the rationals are a countable set with a measure of 0. With $B_1 = O_1BU_1$, we should thus expect $qB_1(\boldsymbol{x}_1 - \boldsymbol{x}'_1) \in \mathbb{Z}^n$ never to occur for $\boldsymbol{x}_1 \neq \boldsymbol{x}'_1$ since an integer result would imply that the entries of B_1 are rational. Therefore, the only valid candidate $\boldsymbol{e}'_1$ is $\boldsymbol{e}_1$ itself (where $\boldsymbol{x}_1 = \boldsymbol{x}'_1$), so $\boldsymbol{e}_1$ is uniquely determined by $\boldsymbol{c}_1$ and has an entropy of 0 from the attacker's perspective, as visualized in Figure 5.2a. This holds for any discretization of the Gaussian distribution since discretizing implies making the distribution's support countable.

Of course, a real-valued orthogonal matrix O_1 would itself have to be discretized to be used in a public key.[12] The hardness of LIP for orthogonal matrices not sampled from $O_n(\mathbb{R})$ has yet to be investigated. Nonetheless, we perform this *gedankenexperiment* to show that, even given a suitable discretization of $O_n(\mathbb{R})$, a lattice-independent discretized Gaussian still cannot achieve a usable minimum-entropy result. To discretize $O_n(\mathbb{R})$ in a manner suitable for this purpose, we make use of the fact that q as used in the original LIP-KEM must only satisfy the lower bound

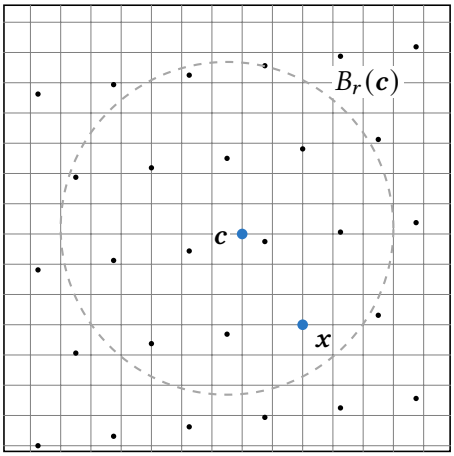
$$q \geq \frac{s \cdot n}{r} \cdot \sqrt{\frac{\ln(2n^2 + 4n)}{\pi}}.$$

Above this bound, q can be chosen to be exponentially large. We set $q = 2^t$ for an appropriate $t \in \mathbb{N}$ and let $O_i \in O_n(q^{-1}\mathbb{Z}^n)$. Note that this corresponds to storing the entries of O_i in a binary fixed-point format with t bits after the decimal. With a sufficiently large t, any desired finite precision could be achieved in this format.

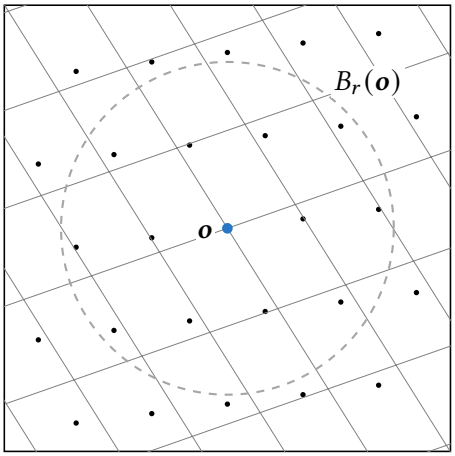
Given that $B_i = O_iBU_i$ with $O_i \in O_n(q^{-1}\mathbb{Z}^n)$ and assuming $B \in \mathbb{Z}^{n\times n}$,[13] we have $qB_i \in \mathbb{Z}^{n\times n}$. Using (5.1), we see that $qB_i\boldsymbol{e}'_i = \boldsymbol{v} + qB_i(\boldsymbol{x}_i - \boldsymbol{x}'_i) \in \mathbb{Z}^n$, so the entropy argument from

[12] Recall that this is the reason why Ducas and van Woerden [33] introduce the quadratic form formulation.

[13] We make this assumption for the sake of simplicity and w.l.o.g.: If $B \in \mathbb{R}^{n\times n}$, we can similarly discretize it to $B \in q^{-1}\mathbb{Z}^{n\times n}$ and let $q \leftarrow q^2$.



(a) Misalignment between the scaled $\mathbb{Z}^n$ and the lattice $\Lambda(B_1)$ when sampling error terms as $\boldsymbol{e} \leftarrow B_i^{-1}\boldsymbol{v}$.



(b) Misalignment between two rotations of the same 2D lattice $\Lambda(B_1)$ in every point in $B_r(\boldsymbol{o})$ except $\boldsymbol{o}$ itself.

Figure 5.2: Misalignments between different lattices cause any CES using discrete, lattice-independent Gaussians to fail. In both figures, the black dots represent the lattice $\Lambda(B_1)$. The scaled $\mathbb{Z}^n$ and the rotated $O\Lambda(B_1)$ are rendered as a gray grid with the lattice points at the intersections. The points $\boldsymbol{x}$ and $\boldsymbol{c}$ in **(a)** have been transformed into the lattice for the sake of visualization.

Theorem 5.1's proof can be applied analogously. This shows that, in the $k = 1$ case, $\boldsymbol{e}_1$ has the same minimum entropy when sampled from $q^{-1}B_1^{-1}\mathcal{D}_{\mathbb{1}_n,s'}$ as when sampled from $q^{-1}\mathcal{D}_{B_1^TB_1,s'}$ for a discretized O_1. We consider $k = 2$ next: The attacker now receives both $\boldsymbol{c}_1 = \boldsymbol{e}_1 \bmod \mathbb{Z}^n$ and $\boldsymbol{c}_2 = \boldsymbol{e}_2 \bmod \mathbb{Z}^n$ where $\boldsymbol{e}_1 = q^{-1}B_1^{-1}\boldsymbol{v}$ and $\boldsymbol{e}_2 = q^{-1}B_2^{-1}\boldsymbol{v}$. An entropy argument would have to establish that there are many possible $\boldsymbol{v}$ that could have resulted in the $(\boldsymbol{c}_1, \boldsymbol{c}_2)$ the attacker sees. Through rearrangement, we find that $\boldsymbol{v} = qB_1\boldsymbol{c}_1 - qB_1\boldsymbol{x}_1 = qB_2\boldsymbol{c}_2 - qB_2\boldsymbol{x}_2$. Analogously to (5.1), any potential candidate $\boldsymbol{v}'$ must satisfy $\boldsymbol{v}' = \boldsymbol{v} + qB_1(\boldsymbol{x}_1 - \boldsymbol{x}_1') = \boldsymbol{v} + qB_2(\boldsymbol{x}_2 - \boldsymbol{x}_2')$ for $\boldsymbol{x}_1', \boldsymbol{x}_2' \in \mathbb{Z}^n$, with each instance imposing its own constraint on $\boldsymbol{v}'$. With $\boldsymbol{w}_i = \boldsymbol{x}_i - \boldsymbol{x}_i' \in \mathbb{Z}^n$, it follows that

$$\begin{aligned} qB_1\boldsymbol{w}_1 &= qB_2\boldsymbol{w}_2 \\ \iff B_1\boldsymbol{w}_1 &= B_2\boldsymbol{w}_2 \qquad (5.2) \\ \iff \boldsymbol{w}_1 &= B_1^{-1}B_2\boldsymbol{w}_2 \qquad (5.3) \end{aligned}$$

along with

$$\begin{aligned} \|B_i\boldsymbol{w}_i\|_2 &= \left\|B_i(\boldsymbol{x}_i - \boldsymbol{x}_i')\right\|_2 \\ &= \left\|B_i(\boldsymbol{x}_i - \boldsymbol{c}_i) + B_i(\boldsymbol{c}_i - \boldsymbol{x}_i')\right\|_2 \\ &\leq \left\|B_i(\boldsymbol{x}_i - \boldsymbol{c}_i)\right\|_2 + \left\|B_i(\boldsymbol{c}_i - \boldsymbol{x}_i')\right\|_2 \\ &\leq 2r. \end{aligned} \qquad (5.4)$$

These constraints can be interpreted in two ways. First, (5.2) implies that the lattices $\Lambda(B_1) = O_1\Lambda(B)$ and $\Lambda(B_2) = O_2\Lambda(B)$ must align at least partially, i.e., there must be many points $\boldsymbol{y} \in O_1\Lambda(B) \cap O_2\Lambda(B)$. Equation (5.4) additionally imposes the constraint that these alignments must be common for $\|\boldsymbol{y}\|_2 \leq 2r$. This is unlikely to occur even with $O_i \in O_n(q^{-1}\mathbb{Z})$ since the vast majority of orthogonal transformations will misalign the lattices (see Figure 5.2b).

Even for the most symmetrical lattice $\mathbb{Z}^n$, only a small number of transformations will lead to a significant overlap. It is easy to see that any non-trivial lattice will have even fewer alignments on average. In fact, we should expect an alignment to become even less likely with growing q since increasing the precision also causes the proportion of misaligning O_i to grow.[14] Increasing $k > 2$ adds more orthogonal transformations that make alignments even more unlikely to these constraints.

Second, (5.3) shows that $B_1^{-1}B_2$ must map many integer vectors $\boldsymbol{w}_2$ (with $\|B_2\boldsymbol{w}_2\|_2 \leq 2r$) to integer vectors $\boldsymbol{w}_1$ for the entropy argument to work. This could only be guaranteed if $B_i \in \mathrm{GL}_n(\mathbb{Z})$ — however, this means that both B and O_i would have to be chosen from $\mathrm{GL}_n(\mathbb{Z})$. If B is unimodular, then it is a basis of the trivial lattice $\mathbb{Z}^n$. A unimodular O_i, on the other hand, means that $O \in \mathrm{SP}_n$. Effectively, this limits the construction to the trivial lattice under signed permutations. sLIP is trivial for these instances since the basis B_i itself is a unimodular matrix and $\mathbb{1}_n$ a signed permutation with $\mathbb{1}_n\mathbb{1}_nB_i = B_i$.[15] The requirements of (5.2), (5.3), and (5.4) and the following argument apply independently of the discretization of the Gaussian, so using a different discretization would not solve the issue. This problem is inherent to LIP and is therefore a clear instance of the Lattice Incompatibility Problem. We see that, even with additional affordances for the discretization of the orthogonal matrices, it is impossible to securely instantiate a CES that has the necessary minimum entropy for an IND-CPA-secure LIP-KEM$_k$ from discretized, lattice-independent Gaussians.

To conclude this section, we discuss possible alternatives to the CES approaches treated above and argue that these would not circumvent the fundamental problems a CES construction for an IND-CPA-secure LIP-KEM$_k$ encounters:

- *Other distributions*: Throughout Section 5.3, we have exclusively considered (discrete) Gaussian distributions in lattices or Euclidean space. We do so because the alternatives are worse: For correctness, any suitable distribution must have the overwhelming part of its probability mass within the ball of radius $2r$ around the origin in the lattice norm. To maximize the potential entropy, the distribution should have good coverage of that disk and should be invariant under rotation (aside from its discretization). These considerations leave no room for alternatives that are efficiently samplable and meaningfully different from a Gaussian in a way that avoids the problems discussed above.
- *Other derivations for* $\boldsymbol{c}$: We let $(\boldsymbol{c}_i)_{i\in[k]} \leftarrow (\boldsymbol{e}_i \bmod \mathbb{Z}^n)_{i\in[k]}$ in Algorithm 5.2 because it enables the original LIP-KEM's entropy argument: Every $\boldsymbol{e}$ is uniquely tied to a $\boldsymbol{x} = \boldsymbol{c} - \boldsymbol{e}$, so the probability of a candidate $\boldsymbol{x}'$ being the "true" decoding in the lattice with a dense sublattice is exactly the probability of the corresponding $\boldsymbol{e}' = \boldsymbol{c} - \boldsymbol{x}'$ being sampled in the first place. Derivations of $\boldsymbol{c}$ where the decoding $\boldsymbol{x}$ is not uniquely linked to a specific $\boldsymbol{e}$ given the ciphertext would add even more complexity to the issue without addressing any of the existing problems. For instance, letting $\boldsymbol{c} = \boldsymbol{e} + \boldsymbol{x}$

[14] The lattice $\mathbb{Z}^n$ is once again an illustrative example here: For $q = 1$, every $O_i \in O_n(q^{-1}\mathbb{Z})$ leads to a perfect alignment between $O_i\mathbb{Z}^n$ and $\mathbb{Z}^n$ since $O_n(\mathbb{Z}) = \mathrm{SP}_n$, containing exactly the symmetries of $\mathbb{Z}^n$. Increasing q introduces transformations that misalign $O_i\mathbb{Z}^n$ from the untransformed lattice. Other rational lattices may have symmetries at $q' > 1$, but scaling $q > q'$ will also create more misalignments in these cases.

[15] Careful inspection reveals that this is a consequence of Lemma 5.3.

for $\boldsymbol{x} \sim \mathcal{X}$ given a distribution $\mathcal{X}$ over $\mathbb{Z}^n$ adds another constraint for the attacker to use: Every candidate $\boldsymbol{e}'$ must not only be samplable by GenError, but must also match a $\boldsymbol{x}' = \boldsymbol{c} - \boldsymbol{e}'$ that could have been sampled from $\mathcal{X}$.

- *Using $\boldsymbol{c}$ as a hint for reconstruction*: According to Definition 5.1, Recons may use $(\boldsymbol{c}_i)_{i\in[k]}$when reconstructing the error terms $(\boldsymbol{e}_i)_{i\in[k]}$, but none of our schemes for a CES take advantage of this capability. This is because we were unable to find a CES scheme that helpfully integrates the $\boldsymbol{c}_i$ into the reconstruction process: While the schemes in Section 5.3.2 can already reconstruct $(\boldsymbol{e}_i)_{i\in[k]}$without using $(\boldsymbol{c}_i)_{i\in[k]}$, the schemes in Section 5.3.1 fail to produce usefully correlated error terms in the first place.
- *General t-to-all reconstructions*: Like above, we only consider 1-to-all reconstructions of the error terms instead of the more general t-to-all reconstructions because these would not address the problems our schemes encounter in their constructions or proofs.
- *Other proof strategies*: In Section 5.3.2, we show that our approaches do not achieve the necessary entropy for an IND-CPA proof even in a lattice with a dense sublattice. Of course, this does not categorically rule out that inserting other intermediate steps into the proof might lead to a successful entropy argument. However, we are not aware of any such steps, noting that our arguments hold for arbitrary full-rank lattices and any lattice-independent distribution and discretization.[16]

Conjecture 2. *There is no correct and consistent CES such that LIP-KEM$_k$ as defined in Algorithm 5.2 can be proven IND-CPA-secure in any model of security.*

5.4 Correlated Lattice Points

The second correlatable component of the LIP-KEM is the lattice point coefficient vector $\boldsymbol{x}$. Throughout this section, we analyze several attempts at realizing an IND-CPA-secure and verifiable LIP-KEM$_k$ by correlating these vectors. All but one of these attempts fail to achieve provable IND-CPA security. A construction using LWE has a proof strategy, but does not actually make use of LIP and is equivalent to existing code-based cryptography. At the end of the section, we claim that a verifiable and IND-CPA-secure KEM$_k$ based on correlated lattice points does not exist.

Our first step is to define how a k-repeated scheme based on correlating $\boldsymbol{x}$ is structured. In the original LIP-KEM's encapsulation algorithm, the lattice point is entirely implicit: Encaps samples an error term $\boldsymbol{e}$ and sets the ciphertext $\boldsymbol{c} \leftarrow \boldsymbol{e} \bmod \mathbb{Z}^n$. We call the lattice point decoding of the ciphertext $\boldsymbol{x} \coloneqq \boldsymbol{c} - \boldsymbol{e}$. It is used by Decaps to reconstruct the error term $\boldsymbol{e}$ and extract the encapsulated key. In this scheme, each $\boldsymbol{e}$ implicitly defines its corresponding

[16] Also note that our arguments are mainly based on the supports of the distributions and the misalignments of the public key lattices, the former being easily recognizable by the adversary (and thus difficult to change) and the latter a consequence of the use of LIP.

$\boldsymbol{x}$ through the modulo operation. In order to correlate the lattice points across multiple ciphertexts without also correlating the errors (which is discussed in Section 5.3), this implicit link must be broken. We do this by explicitly choosing $\boldsymbol{x} \leftarrow \mathcal{X}$ according to some efficiently samplable distribution $\mathcal{X}$ and setting $\boldsymbol{c} \leftarrow \boldsymbol{x} + \boldsymbol{e}$ as the ciphertext of an instance. Correlating multiple instances of the LIP-KEM is then simply implemented by choosing one shared $\boldsymbol{x}$ for all instances. In the k-repeated scheme using this approach, Encaps samples the error terms $\boldsymbol{e}_i$ independently[17] as $\boldsymbol{e}_i \leftarrow q^{-1}\mathcal{D}_{P_i, qr/\sqrt{n}}$ and adds them to a shared $\boldsymbol{x} \leftarrow \mathcal{X}$ to form the ciphertexts $\boldsymbol{c}_i \leftarrow \boldsymbol{x} + \boldsymbol{e}_i$. In Decaps, $\boldsymbol{c}_1$ is decoded just like in the original LIP-KEM to retrieve $\boldsymbol{x}$ before outputting the extracted key if the ciphertext is valid. This is the case if every $\boldsymbol{c}_i$ decodes to the same $\boldsymbol{x}$. Dec_{Ver} works similarly, using the known secret key U_j to decode $\boldsymbol{c}_j$ to $\boldsymbol{x}$ and check that $\|\boldsymbol{c}_i - \boldsymbol{x}\|_Q < r \ \forall i \in [k]$. If this check passes, the $\boldsymbol{c}_i$ all decode to $\boldsymbol{x}$, thus verifying the validity of the ciphertext. To generate the encapsulated key, the extractor $\mathcal{E}$ can be applied to either $\boldsymbol{x}$ or $(\boldsymbol{e}_i)_{i\in[k]}$— since both are related bijectively given $(\boldsymbol{c}_i)_{i\in[k]}$, they have exactly the same entropy in a potential IND-CPA proof using a dense sublattice.

With the general structure of the k-repeated LIP-KEM defined, we move to explain the motivation behind using correlated lattice points. This approach is mainly motivated by Döttling et al.'s [31] work, in which the authors construct a code-based PKE_k from the McEliece cryptosystem. In McEliece [54], a generator matrix $G \in \mathbb{F}_2^{n\times k}$ for an efficiently decodable code is hidden using a permutation matrix $P \in P_n$ and an invertible matrix $S \in GL_k(\mathbb{F}_2)$ to form a public key PGS. Messages m are encrypted by encoding them and adding a small error vector e to the encoded message:[18] $c \leftarrow PGSm + e$. Döttling et al. [31] build a PKE_k from this system by encrypting the same message m using k independent public keys and error vectors. Now consider a LIP-KEM with an explicitly chosen lattice point: We sample a lattice point $\boldsymbol{x}$ and an error term $\boldsymbol{e}$ with $\|\boldsymbol{e}\|_Q \leq r$ and output $\boldsymbol{c} \leftarrow \boldsymbol{x} + \boldsymbol{e}$. In the lattice formulation, where the public key is $B' = OBU$, we could equivalently output $\boldsymbol{c}' \leftarrow B'(\boldsymbol{x} + \boldsymbol{e}) = OBU\boldsymbol{x} + \boldsymbol{e}'$ with $\|\boldsymbol{e}'\|_2 \leq r$. The obvious structural similarity to the McEliece cryptosystem suggests that a similar construction to that of Döttling et al. [31] based on the LIP-KEM and using correlated lattice points could be possible.

Before moving on to discussing specific constructions and proof arguments, we explain the basic properties and challenges of this approach. First, given that using the same $\boldsymbol{x}$ across instances makes verification easy, the only challenge for building a LIP-KEM_k with correlated lattice points is the IND-CPA proof. Achieving IND-CPA security in turn requires establishing some form of "correlated decoding hardness" — For any proof to work, it must be difficult for an adversary to decode $(\boldsymbol{c}_i)_{i\in[k]}$to $(\boldsymbol{x})_{i\in[k]}$. Otherwise, the adversary could just decode the ciphertext and get access to the encapsulated key. Preventing the adversary from doing so is made more difficult by the fact that each instance adds an additional

[17] Sampling the error terms independently for each instance avoids the issues encountered in Section 5.3 that arise from correlated errors. We do not set $\boldsymbol{x}$ as the center and sample $\boldsymbol{e}_i \leftarrow q^{-1}\mathcal{D}_{P_i, qr/\sqrt{n}, \boldsymbol{x}}$ because the scaling factor $1/q$ would also scale down $\boldsymbol{x}$ and the resulting $\boldsymbol{e}_i$ would not be within the decoding radius of $\boldsymbol{x}$.

[18] This simplified "textbook" version of the encryption scheme only achieves OW-CPA security. See Nojima et al.'s [70] work for an IND-CPA-secure randomized variant that is also used by Döttling et al. [31] in their construction.

constraint of the form $\|\boldsymbol{c}_i - \boldsymbol{x}\|_Q \leq r$ on $\boldsymbol{x}$. These constraints must not help the adversary reconstruct the lattice point. We can recontextualize this challenge through the lens of the original LIP-KEM by considering the orthogonally concatenated quadratic form

$$\boldsymbol{P} = \begin{pmatrix} P_1 & & 0 \\ & \ddots & \\ 0 & & P_k \end{pmatrix} \tag{5.5}$$

and the concatenated ciphertext $\boldsymbol{c} = (\boldsymbol{c}_1 \| \cdots \| \boldsymbol{c}_k)$. Using the separability of the Gaussian and $r' = \sqrt{k}r$, the concatenated error term $\boldsymbol{e} = (\boldsymbol{e}_1 \| \cdots \| \boldsymbol{e}_k)$ precisely satisfies

$$\boldsymbol{e} \sim \frac{1}{q}\mathcal{D}_{\boldsymbol{P}, \frac{qr}{\sqrt{n}}} = \frac{1}{q}\mathcal{D}_{\boldsymbol{P}, \frac{qr'}{\sqrt{kn}}}.$$

The concatenated instance therefore forms an instance of the original LIP-KEM with an nk-dimensional lattice. However, the original LIP-KEM's proof does not apply here because the decoding of the concatenated $\boldsymbol{c}$ is not any lattice point $\boldsymbol{x}$', but specifically of the form $(\boldsymbol{x} \| \cdots \| \boldsymbol{x})$. Any proof strategy must therefore show that the adversary cannot efficiently make use of this constraint.

By switching to the lattice formulation, we can also demonstrate the link between our correlated decoding problem and the standard BDD problem: Each ciphertext of the LIP-KEM$_k$ is of the form $\boldsymbol{c}_i = B_i\boldsymbol{x} + \boldsymbol{e}'_i$ with a Gaussian-distributed, small $\boldsymbol{e}'_i = B_i\boldsymbol{e}_i$, so "stacking" the bases, error terms, and ciphertexts results in

$$\boldsymbol{B}_* = \begin{pmatrix} B_1 \\ \vdots \\ B_k \end{pmatrix} \quad \text{and} \quad \boldsymbol{e}'_* = \begin{pmatrix} B_1\boldsymbol{e}_1 \\ \vdots \\ B_k\boldsymbol{e}_k \end{pmatrix} \quad \text{with} \quad \boldsymbol{c}_* = \boldsymbol{B}_*\boldsymbol{x} + \boldsymbol{e}'_*. \tag{5.6}$$

The stacked ciphertext $\boldsymbol{c}_*$ can be interpreted as an instance of BDD for a lattice $\Lambda(\boldsymbol{B}_*)$ of rank n and dimension nk and with a radius of r'.[19] Unfortunately, this connection is not immediately useful — As discussed by Bennett and Peikert [13], BDD is only expected to be difficult in the worst case, making it difficult to base a cryptographic scheme on it. Compounding the issue, BDD is a search problem that could at most grant a form of one-wayness for $\boldsymbol{x}$, but not the entropy we would need for a proof. LWE, on the other hand, is an average-case form of BDD [13] that we do attempt to make use of in Section 5.4.2.

Another notable property of the correlated lattice point approach is that each public key P_i (or B_i in the lattice formulation) maps the same $\boldsymbol{x}$ to a completely different point in the lattice space. On one hand, this could be considered a promising advantage: If the lattice geometry around $\boldsymbol{x}$ changes unpredictably depending on the public key, then it should be difficult for an adversary to link the different ciphertexts together. On the other hand, this property turns out to be the main difficulty in any potential IND-CPA proof — Since

[19] Note that the stacked basis $\boldsymbol{B}_*$ is *not* the lattice basis corresponding to the concatenated quadratic form $\boldsymbol{P}$ introduced in equation (5.5). Instead, $\boldsymbol{B}_*^T\boldsymbol{B}_* = \sum_{i=0}^{k} P_i$. For lattices not of full rank, we cannot effectively use quadratic forms anyway since the Gram map is no longer bijective.

$\boldsymbol{x}$ could end up effectively anywhere in a very large radius in k different lattice spaces, a successful proof argument needs to work even with very large discrepancies between the geometries of the different public keys. Of the proof strategies we present in the following, those that fail do so precisely because of these discrepancies. In contrast, our LWE-based argument handles the different lattice geometries by taking the modulo and working in a compact space. This limits the possible deviations between the lattices to be contained in said space, making the proof work. In total, we discuss three possible proof strategies: Section 5.4.1 shows that a statistical argument cannot work for correlated lattice points. Next, Section 5.4.2 explains how LWE could be used to create a scheme with a working proof strategy that would however be equivalent to the existing code-based PKE_k by Döttling et al. [31]. Finally, we introduce a new assumption for correlated decoding hardness in Section 5.4.3 and show that it likely does not hold for generic lattices.

5.4.1 Statistical Decoding Hardness

Since the original LIP-KEM uses a statistical argument as the second step of its IND-CPA proof, one may attempt to apply an analogous proof to the $\mathrm{LIP\text{-}KEM}_k$ with correlated lattice points. In this section, we demonstrate that this is not possible since both $\boldsymbol{x}$ and $(\boldsymbol{e}_i)_{i\in[k]}$ lack the required minimum entropy for the extractor to produce a uniformly random encapsulated key.[20] Our argument is based on both theoretical and empirical findings about the lattice geometries. Throughout the entire section, we assume that the uniquely decodable quadratic forms sampled from $\mathcal{D}_s([S])$ have already been replaced with quadratic forms $P_i \leftarrow \mathcal{D}_s([Q])$ for a lattice Q with a dense sublattice like in the LIP-KEM's proof.

In order for $\boldsymbol{x}$ to have some nonzero entropy given $(\boldsymbol{c}_i)_{i\in[k]}$, there must be multiple possible candidates $\boldsymbol{x}' \in \mathbb{Z}^n$ that could have resulted in the same ciphertext. Formally, this means

$$(\boldsymbol{c}_i)_{i\in[k]} = \left(\boldsymbol{x}' + \boldsymbol{e}'_i\right)_{i\in[k]} \text{ with } \left\|\boldsymbol{e}'_i\right\|_{P_i} \leq r \quad \forall i \in [k],$$

or, equivalently,

$$\left\|\boldsymbol{c}_i - \boldsymbol{x}'\right\|_{P_i} \leq r \quad \forall i \in [k].$$

For the $k = 1$ case, Lemma 5.1 tells us that there are many such $\boldsymbol{x}'$.[21] To simplify further analysis, we loosen this condition to one that is necessary, but insufficient. Let $\boldsymbol{y} := \boldsymbol{x}' - \boldsymbol{x}$ and see that

$$\|\boldsymbol{y}\|_{P_i} = \left\|\boldsymbol{x}' - \boldsymbol{c}_i + \boldsymbol{c}_i - \boldsymbol{x}\right\|_{P_i} \leq \left\|\boldsymbol{x}' - \boldsymbol{c}_i\right\|_{P_i} + \left\|\boldsymbol{x} - \boldsymbol{c}_i\right\|_{P_i} \leq 2r \quad \forall i \in [k]. \tag{5.7}$$

This bound tells us that, for every potential candidate $\boldsymbol{x}' \in \mathbb{Z}^n$, there must be a coefficient vector $\boldsymbol{y} \in \mathbb{Z}^n$ that is short in every inner product space given by one of the public keys.

[20] As previously noted, $\boldsymbol{x}$ and $(\boldsymbol{e}_i)_{i\in[k]}$ have the same entropy given $(\boldsymbol{c}_i)_{i\in[k]}$ since knowing one uniquely determines the other through the relation $\boldsymbol{c}_i = \boldsymbol{x} + \boldsymbol{e}_i$.

[21] Recall from the proof of Theorem 5.1 that our choice of the distribution $\mathcal{D}_{P_i,s}$ for the error terms guarantees that every $\boldsymbol{x}' \in \mathbb{Z}^n$ is indeed a possible candidate.

Note that the true $\boldsymbol{x}$ corresponds to the vector $\boldsymbol{y} = \boldsymbol{o}$, which is always guaranteed to be short. Condition (5.7) is a strong requirement: Due to the unimodular matrices changing the basis of the lattice, a coefficient vector that maps to a short lattice vector in one basis can map to a long lattice vector in another. In addition, there must be an exponential number of such vectors $\boldsymbol{y}$ in order for $\boldsymbol{x}$ to have a polynomial amount of entropy, which is required for the extractor to be able to extract a sufficiently long encapsulated key. We present both a theoretical and an empirical argument for why there cannot be many $\boldsymbol{y}$ satisfying (5.7).

To further analyze condition (5.7) from a theoretical perspective, we use the unimodular matrices U_i associated with the $P_i = U_i^T Q U_i$. Given that $\|\cdot\|_{P_i} = \|U_i \cdot\|_Q$, we can rewrite (5.7) as

$$\|U_i \boldsymbol{y}\|_Q \leq 2r \quad \forall i \in [k].$$

In Euclidean space, each set

$$E_i = \{\boldsymbol{v} \mid \boldsymbol{v} \in \mathbb{R}^n,\ \|U_i \boldsymbol{v}\|_Q^2 = \boldsymbol{v}^T U_i^T Q U_i \boldsymbol{v} \leq 4r^2\} \tag{5.8}$$

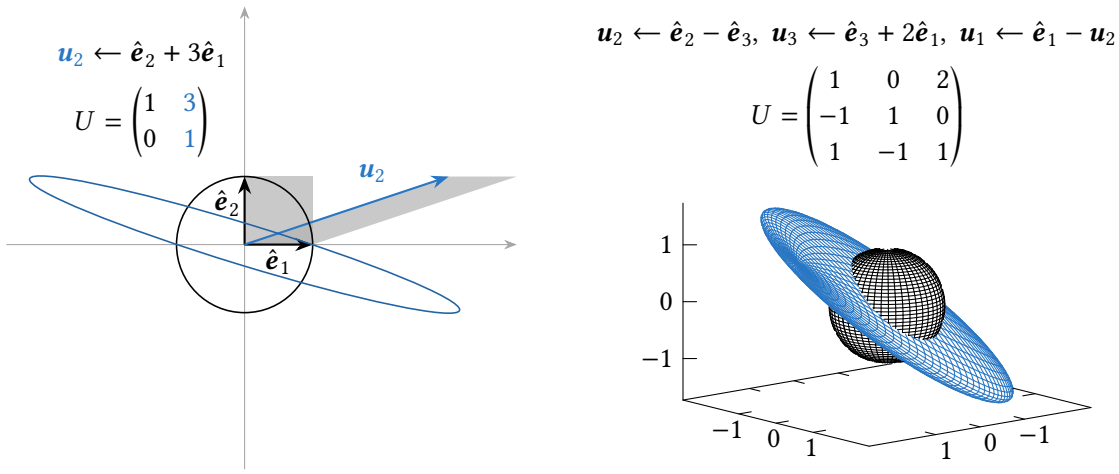
is bounded by an n-dimensional ellipsoid around the origin. If $\boldsymbol{x}' \in \mathbb{Z}^n$ is a possible decoding candidate, we must therefore have

$$\boldsymbol{y} = \boldsymbol{x}' - \boldsymbol{x} \in \bigcap_{i=1}^{k} E_i \cap \mathbb{Z}^n.$$

However, the bounding ellipsoids of the E_i can be expected to be eccentric as a consequence of $|\det(U_i)| = 1$, $U_i \in \mathbb{Z}^{n \times n}$, and $U_i \not\approx \mathbb{1}_n$: There simply are very few "non-eccentric" unimodular matrices U_i (i.e., those that do not significantly distort the parallelepiped $\mathcal{P}(U_i)$). This is because even minimal column operations of the form $\boldsymbol{u}_i \leftarrow \boldsymbol{u}_i + a\boldsymbol{u}_{i'}$ for $i, i' \in [n]$ and small $a \in \mathbb{Z} \setminus \{0\}$ on a unimodular matrix (compare Section 2.3) heavily skew the parallelepiped, as illustrated in Figure 5.3. The average-case hardness of LIP also tells us that these non-eccentric unimodular matrices must be sampled only negligibly often — otherwise, LIP would be easy since such a U_i would satisfy $U_i \approx \mathbb{1}_n$ and thus barely change the basis of the lattice. In addition, both $\|U_i\|$ and the eigenvalues of the U_i are only very loosely bounded.[22] The E_i also all have the same volume since each ellipsoid is related to the base ellipsoid defined by Q via the linear transformation U_i with determinant $|\det(U_i)| = 1$. An eccentric ellipsoid with a constant volume has a long major axis, but its other axes must be short to maintain the volume (they are "long and skinny"). The ellipsoids bounding the E_i will also generally have their major axes point in different directions. While we showed that the major axes typically lie in $\mathbb{R}^n_{\geq 0} \cup \mathbb{R}^n_{\leq 0}$ in Section 3.3, this still leaves a large amount of space for the ellipsoids to misalign.[23] In addition, since $k \in \Theta(\lambda)$, increasing the security parameter actually makes the problem worse since a larger number of ellipsoids have to align. Accordingly, we expect the intersection of the E_i to be small. In many cases, the only integer coefficient vector in the intersection will be $\boldsymbol{o}$, so $\boldsymbol{x}$ will be uniquely identified by $(\boldsymbol{c}_i)_{i \in [k]}$.

[22] While we know that $\left\|U_i^*\right\|_Q \leq \left\|B_i^*\right\| \leq s\sqrt{n}$ with overwhelming probability (compare Ducas and van Woerden [33, Lemma 3.1]), this does not provide an upper bound for the actual columns of U_i.

[23] In particular, the curse of dimensionality implies that this space grows exponentially with n.



(a) Parallelepipeds and ellipses of volume 1 for the identity and a unimodular U. The ellipse with the equation $\boldsymbol{v}^T U^T U \boldsymbol{v} = 1$ is shown in blue.

(b) Three-dimensional ellipsoids of volume 1 for the identity and a unimodular U. The ellipsoid with the equation $\boldsymbol{v}^T U^T U \boldsymbol{v} = 1$ is shown in blue.

Figure 5.3: Unimodular matrices U heavily distort the parallelepiped $\mathcal{P}(U)$ and the unit ball even for "small" differences to the identity. **(a)** and **(b)** show this for two- and three-dimensional examples, respectively. Note that the matrices chosen for these examples are deliberately close to the identity; in practice, the entries of U sampled using $\mathcal{D}_s([Q])$ for a quadratic form Q and appropriate s are much larger.

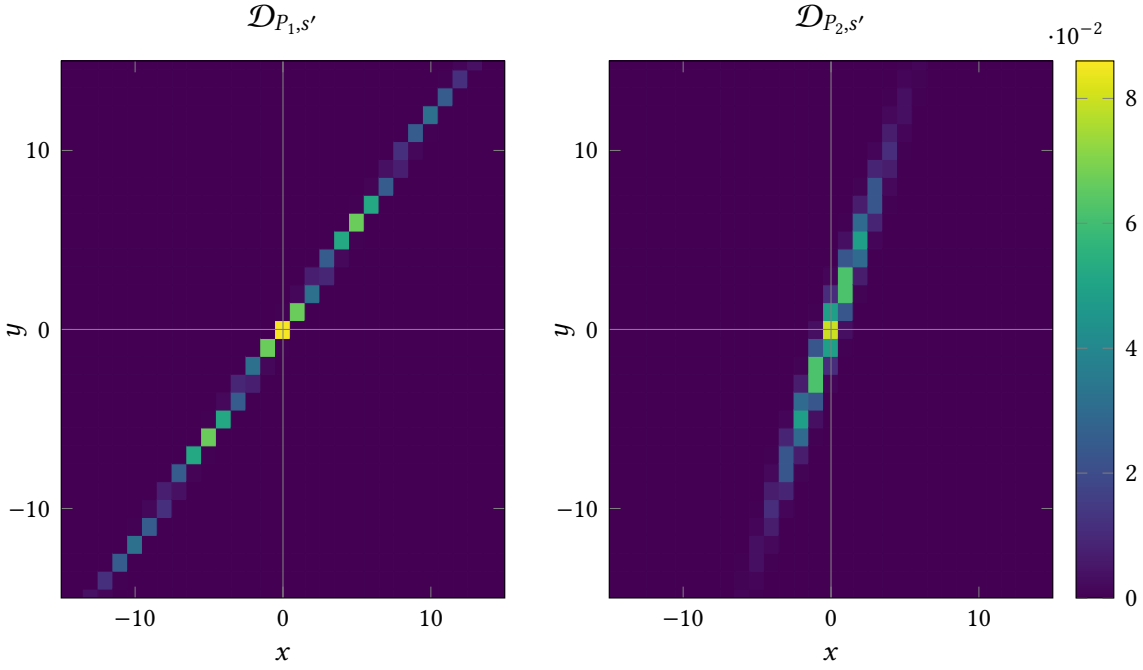
We can also empirically verify that there are few vectors $\boldsymbol{y}$ that satisfy the bound (5.7) by testing a minimal example with $k = n = 2$. We consider the discrete Gaussian distributions

$$\mathcal{D}_{P_1, \frac{2r}{\sqrt{n}}} \text{ and } \mathcal{D}_{P_2, \frac{2r}{\sqrt{n}}}$$

for two public keys $P_1, P_2 \leftarrow \mathcal{D}_s([\mathbb{1}_2])$. This models a best-case scenario where the entire lattices form the dense sublattices required by Lemma 5.1. Recall that $\|\boldsymbol{z}\|_{P_i} \leq 2r$ for $\boldsymbol{z} \leftarrow \mathcal{D}_{P_1, 2r/\sqrt{n}}$ with overwhelming probability, so almost all of the probability mass of each distribution is contained within E_1 and E_2. We calculate the product of the density functions, which corresponds to the intersection of the two sets. The distributions' sampling parameter $s' \approx \frac{2r}{\sqrt{n}}$ must be chosen such that $\frac{2r}{\sqrt{n}} \geq 4\eta_{\frac{1}{2}}(\mathbb{1}_2)$ according to Lemma 5.1, so let

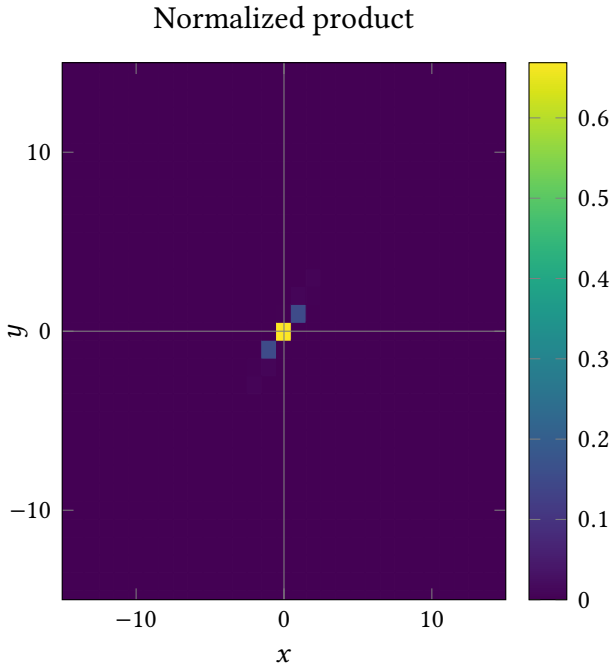
$$s' = 4\sqrt{\ln(6n)/\pi} \geq 4 \|\mathbb{1}_2\| \sqrt{\ln(2n(1+2))/\pi} \geq 4\eta_{\frac{1}{2}}(\mathbb{1}_2).$$

On the other hand, we set the parameter s for the public key distribution $\mathcal{D}_s([\mathbb{1}_2])$ to be larger than technically necessary for the lattice $\mathbb{Z}^2$ at $s = 20$. This is to accommodate security margins and compensate for the small choice of n. Our results are depicted in Figure 5.4. Clearly, if the sets E_i already fail to have a significant intersection in this minimal case, we cannot expect their intersection to be larger for greater n and especially for greater k. We consider this another instance of the Lattice Incompatibility Problem.



(a) Approximation of E_1 for $P_1 = \left(\begin{smallmatrix} 37 & -31 \\ -31 & 26 \end{smallmatrix}\right)$.

(b) Approximation of E_2 for $P_2 = \left(\begin{smallmatrix} 13 & -5 \\ -5 & 2 \end{smallmatrix}\right)$.



(c) Approximation of $E_1 \cap E_2$.

Figure 5.4: Empirical test for the overlap between two balls as defined in (5.8) with two public keys P_1 and P_2 sampled from $\mathcal{D}_s([\mathbb{1}_2])$ with $s = 20$. Figures **a)** and **b)** show the discrete Gaussian distributions $\mathcal{D}_{P_i,s'}$ for $s' = 2\sqrt{\ln(6n)/\pi}$, which have almost all of their probability mass inside E_1 and E_2 respectively. Figure **c)** shows the normalized product of the two density functions. Note how only three integer points have a non-negligible probability, with the origin contributing more than 60% of the probability mass.

The outcomes of our theoretical and empirical arguments are even more severe considering that the bound in (5.7) is not sufficient, so even if $\bigcap_{i=1}^{k} E_i$ were not almost empty, not all of the vectors $\boldsymbol{y}$ in the intersection would actually induce a valid decoding candidate $\boldsymbol{x}'$. It stands to reason that a lattice with an exponentially dense sublattice (i.e., $2r/\sqrt{n} \geq 2^l \eta_\epsilon(D^T Q D)$ for the sublattice D and some $l \in \Theta(\lambda)$) would be required for $\boldsymbol{x}$ to have a sufficient minimum entropy for the extractor. Of course, this would make $\mathrm{gap}(Q)$ exponential as well. Since the LLL algorithm [50] solves f-SVP for exponential $f \geq 2^{(n-1)/2}$ in polynomial time [80], Q having an exponential gap would make the scheme trivially insecure.

Since the cause of the lack of entropy is the difference between the unimodular matrices U_i, it is worth considering whether these differences could be eliminated such that the E_i would align. To this end, recall the stacked basis $\boldsymbol{B}_*$ in the lattice formulation with $B^T B = Q$ from (5.6). If it could be shown that

$$\boldsymbol{B}_* = \begin{pmatrix} O_1 B U_i \\ \vdots \\ O_k B U_k \end{pmatrix} \overset{\mathfrak{C}}{\approx} \begin{pmatrix} O_1 B U \\ \vdots \\ O_k B U \end{pmatrix} = \begin{pmatrix} O_1 B \\ \vdots \\ O_k B \end{pmatrix} U =: \boldsymbol{B}'_*$$

for appropriately sampled $O_i \in O_n(\mathbb{R})$ and $U, U_i \in \mathrm{GL}_n(\mathbb{Z})$, switching to using only one U would make the E_i align perfectly, enabling an entropy argument. Unfortunately, $\boldsymbol{B}_*$ and $\boldsymbol{B}'_*$ are easily distinguished by taking the Gram matrix of any submatrix $O_i B$ of the basis: For $\boldsymbol{B}'_*$, this results in the same $U^T B O_i^T O_i B U = U^T B U = P$ for every submatrix. Doing the same for $\boldsymbol{B}_*$ produces different $P_i = U_i^T Q U_i$, so an adversary will be able to distinguish $\boldsymbol{B}_*$ from $\boldsymbol{B}'_*$ unless we happen to have $U_i \in \mathrm{Aut}(Q) U_1$ for all of the U_i sampled for $\boldsymbol{B}_*$, in which case $P_1 = \cdots = P_k$. As the automorphism group of any $Q \in S_n^{>0}(\mathbb{Z})$ is finite [93, Definition 151], is is unlikely for this to occur. This tells us that we cannot "re-align" the different quadratic forms to increase the entropy of $\boldsymbol{x}$ either.

Given that a statistical argument for decoding hardness using only ΔLIP is likely impossible, we conclude that another computational assumption is strictly required for the IND-CPA security of a LIP-KEM$_k$ with correlated lattice points. This assumption needs to ensure that the adversary cannot take advantage of the almost-unique decodability of $(\boldsymbol{c}_i)_{i \in [k]}$. We find two candidate assumptions that do so by breaking the correlation between the $\boldsymbol{c}_i$ and thus also meet the criteria for a *long jump* in our security proof. These candidates are discussed in the following sections.

5.4.2 Decoding Hardness from LWE

On the search for a computational assumption to establish decoding hardness for a LIP-KEM$_k$ using correlated lattice points, one could again take inspiration from the code-based approach by Döttling et al. [31]. We demonstrate the structural similarity of our potential LIP-KEM$_k$ to their PKE$_k$ based on the McEliece assumption at the start of this section. Going further, their PKE$_k$'s security proof also makes use of the decisional Learning Parity with Noise (LPN) assumption [46], which Regev [83] shows is structurally equivalent to the decisional LWE assumption for $q = 2$. LWE, in turn, is a well-studied assumption in lattice

cryptography. It would therefore make sense to "lift" Döttling et al.'s security assumption from LPN to LWE the same way the correlated-lattice-point construction for a LIP-KEM$_k$ is "lifted" from codes to lattices. An IND-CPA proof for our LIP-KEM$_k$ could then be achieved by combining LIP and LWE. In this section, we show that using LWE unfortunately forces us to drop the LIP assumption in favor of code-based cryptography, making the resulting scheme and its proof fundamentally equivalent to Döttling et al.'s [31] existing work. LWE thus also fails to act as a computational assumption with which we could assert decoding hardness for the LIP-KEM$_k$.

On a conceptual level, the decisional LWE assumption allows us to replace something of the form $A\boldsymbol{x} + \boldsymbol{e} \bmod q\mathbb{Z}^m$ with randomness, where it demands $A \in \mathbb{Z}_q^{m\times l}$, $\boldsymbol{x} \in \mathbb{Z}_q^l$, and $\boldsymbol{e} \in \mathbb{Z}_q^m$ with $\boldsymbol{e} \sim \chi$ for some error distribution χ. Recalling the lattice formulation of the LIP-KEM$_k$'s ciphertext in its stacked representation $\boldsymbol{c}_*$ from (5.6), we see that its structure is similar to that of an LWE instance. The natural strategy in applying LWE to the LIP-KEM$_k$ is thus to replace the stacked ciphertext $\boldsymbol{c}_*$ of the IND-CPA challenge with randomness, making $\boldsymbol{x}$ independent of the challenge ciphertext. Since it replaces all of the k ciphertexts $\boldsymbol{c}_i$ at once, this use of LWE also satisfies the need for a *long jump* in the security proof. With an independently random $\boldsymbol{x}$, the extractor would then produce an encapsulated key indistinguishable from uniform randomness, completing the proof.

Naturally, we cannot use a $\boldsymbol{B}_*$ assembled from arbitrary lattice bases $B_i = O_i B U_i$ for $O_i \in O_n(\mathbb{R})$, $B \in \mathbb{Z}^{n\times n}$ and $U_i \in \mathrm{GL}_n(\mathbb{Z})$ to instantiate LWE: The assumption requires the matrix A to be an element of $\mathbb{Z}_q^{m\times l}$ for some $m, l \in \mathbb{N}$. Conforming to this limitation demands two main concessions:

- *q-ary lattices*: To enforce $\boldsymbol{B}_* \in \mathbb{Z}_q^{nk\times n}$, we must take the remainder modulo q of the individual lattice bases. However, for the modular lattice $\Lambda(B)/q\mathbb{Z}^n$ (and thus $\Lambda(BU_i)/q\mathbb{Z}^n$) to be well-defined, it needs to have $q\mathbb{Z}^n \subset \Lambda(B)$. Similarly, $B \in \mathbb{Z}^{n\times n}$ implies $\Lambda(B) \subset \mathbb{Z}^n$. The lattices that satisfy these two properties are precisely the q-ary lattices. These have bases of the form

$$B = \begin{pmatrix} \mathbb{1}_l & 0 \\ D & q\mathbb{1}_{n-l} \end{pmatrix},$$

  where $\begin{pmatrix} \mathbb{1}_l \\ D \end{pmatrix}$ is the generator matrix for a linear $[n, l]_q$-code C in systematic form with a prime q (compare Definition 2.30). We are thus forced to work with q-ary lattices to make use of LWE.

- *Signed Permutations*: Since the transformed lattice $O_i\Lambda(B)$ must conform to the same restrictions as the original lattice $\Lambda(B)$, we must choose orthogonal transformations that retain the q-ary property. These transformations are exactly the $O_i \in O_n(\mathbb{Z}) = \mathrm{SP}_n$,[24] so the only permitted transformations are signed permutations.

[24] Equality follows by a straightforward argument: Every column of an integer orthogonal matrix must have length 1. An integer column therefore has exactly one nonzero coordinate, which is ±1. As the columns are also orthogonal, each row also has at most one nonzero coordinate and the matrix is a signed permutation matrix.

Combining these constraints, each basis matrix should have the form

$$B_i = O_i \begin{pmatrix} \mathbb{1}_l & 0 \\ D & q\mathbb{1}_{n-l} \end{pmatrix} U_i \bmod q\mathbb{Z}^{n\times n},$$

with $O_i \in \mathrm{SP}_n$, $D \in \mathbb{Z}^{(n-l)\times l}$, and $U_i \in \mathrm{GL}_n(\mathbb{Z})$. After these changes, inspection reveals that B_i is no longer of full rank because the last $(n-l)$ columns of B are zero modulo q. The last $(n-l)$ coordinates of $U_i\boldsymbol{x}$ simply being lost would make decoding ambiguous even given the secret key, so the LIP-KEM$_k$ with these bases fails to achieve correctness. Given that the lattice formulation does not technically require full-rank lattices, however, we can mitigate this issue by dropping these last $(n-l)$ columns from B.[25] This also shrinks the unimodular matrix to $U_i \in \mathrm{GL}_l(\mathbb{Z})$, though there is no reason to insist on $|\det(U_i)| = 1$ when working modulo $\mathbb{Z}_q^{l\times l}$ for prime q: Every matrix $T \in \mathbb{Z}_q^{l\times l}$ with $\det(T) \not\equiv 0 \bmod q$ has an integer inverse $T^{-1} \in \mathbb{Z}_q^{l\times l}$ since every nonzero determinant is itself invertible. We can therefore replace the unimodular U_i with general invertible $T_i \in \mathrm{GL}_n(\mathbb{Z}_q)$.[26]

At this point, we are left with bases of the form $B_i = O_iBT_i \bmod q\mathbb{Z}^{n\times l}$, where O_i is a signed permutation, B is a generator matrix for a linear code, and T_i is an invertible matrix. Ciphertexts for these basis matrices follow the format $B_i\boldsymbol{x} + \boldsymbol{e}$ with $\boldsymbol{x}, \boldsymbol{e} \in \mathbb{Z}_q^k$, and our goal is for the stacked $\boldsymbol{B}_*$ to look like a random matrix $A \in \mathbb{Z}_q^{nk\times l}$. A change in variables shows that what we are looking for is precisely given by Assumption 5.1:

Assumption 5.1 (Generalized McEliece (adapted from [54, 86])). *Given a family of linear $[n, l]_q$-codes with a sampling algorithm $\textsc{Gen}_C \to G$ for generator matrices,*

$$\left\{PGS \mid P \xleftarrow{\$} SP_n, G \leftarrow \textsc{Gen}_C, S \xleftarrow{\$} GL_l(\mathbb{Z}_q)\right\} \overset{c}{\approx} \left\{R \mid R \xleftarrow{\$} \mathbb{Z}_q^{n\times l}\right\}.$$

We have thus essentially recovered a slightly generalized form of the McEliece assumption, and, by extension, Döttling et al.'s [31] PKE$_k$ construction for $q > 2$.[27] Since the KEM$_k$ resulting from this approach would only be a minimal generalization of existing work and makes no use of LIP, we do not pursue it further here. Nevertheless, we remark that Assumption 5.1 would also lack candidate code families for an instantiation with $q > 2$ — while the McEliece PKE and its variant Niederreiter have been instantiated with several different code families [62, 71, 72, 88, 91], some of the few families for which the McEliece assumption is currently assumed to hold are the families of binary Goppa codes [85] and binary quasicyclic MDPC codes [4, 63, 85], for which $q = 2$.

[25] While it would technically be possible to keep the columns and demand that any encoded $\boldsymbol{x}$ be the canonical representative of its fiber (i.e., that $(U_i\boldsymbol{x} \bmod q\mathbb{Z}^l)\times\{0\}^{n-l}$), this is functionally equivalent to just removing the zero columns of B.

[26] The matrices $T_i \xleftarrow{\$} \mathrm{GL}_l(\mathbb{Z}_q)$ can be efficiently sampled: For linearly independent vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m \in \mathbb{Z}_q^l$ with $m \le l-1$, the probability of a random vector $\boldsymbol{v}_{m+1}$ also being linearly independent of the existing vectors is at least $\frac{q-1}{q} \ge \frac{1}{2}$ (compare the work of Applebaum et al. [8, implicit in Lemma 2]).

[27] Note that "LIP" for these bases is the signed permutation equivalence problem, which Bennett and Win [14] show can be reduced to the general lattice formulation of LIP.

As described above, using LWE with the LIP-KEM$_k$ based on correlated lattice points forced the construction from a lattice-based scheme into a code-based scheme. We consider this a consequence of the symmetries between codes and lattices, with both LWE and LIP being at the interface of the two: The LIP assumption as proposed by Ducas and van Woerden [33] is explicitly intended to provide a lattice equivalent to code assumptions, while LWE is itself more structurally similar to a code assumption with reductions from common lattice problems. For our purposes, LWE is not general enough — We would need an assumption that fills a similar role without forcing the structure of a code onto the LIP-KEM. The next section discusses a potential candidate for such an assumption.

As a final note, we also point out that both Peikert [74] and Katz and Vaikuntanathan [47] combine k-repetition with LWE to achieve IND-CCA2-secure encryption in the standard model without any additional assumptions, making LIP technically redundant in this assumption-framework combination. Other methods to achieve IND-CCA2 security using only LWE in the standard model have also been proposed [19, 58, 76, 94].

5.4.3 Decoding Hardness from Lattice Point Scattering

Having determined in the previous sections that none of the LIP, BDD, or LWE assumptions lead to a successful IND-CPA proof for the LIP-KEM$_k$ with correlated lattice points, we have exhausted the canon of lattice assumptions related to decoding hardness in generic lattices in the literature — To the best of our knowledge, other available assumptions either pertain only to special families of lattices, are only hard in the worst case like BDD, or are structurally unrelated to the LIP-KEM$_k$. This includes problems like SVP or Short Integer Solutions (SIS) [56], for which no suitable reduction strategies are available. We therefore turn to defining a new cryptographic problem called Lattice Point Scattering (LPS). Our hope is that we can achieve an IND-CPA proof for the LIP-KEM$_k$ contingent on the hardness of LPS. To this end, we define the LPS problem as follows:

Definition 5.2 (Lattice Point Scattering (LPS))
Given parameters $n, k \in \Theta(\lambda)$, a quadratic form $S \in S_n^{>0}(\mathbb{Z})$ with the decoding radius r,[28] s and q as defined in Algorithm 5.1, $P_1, \dots, P_k \leftarrow \mathcal{D}_s([S])$, and an efficiently samplable distribution $\mathcal{X}$ over $\mathbb{Z}^n$, define the distributions

$$D_0 := \left\{ \left((P_i)_{i\in[k]}, (\boldsymbol{x} + \boldsymbol{e}_i)_{i\in[k]} \right) \;\middle|\; \boldsymbol{x} \leftarrow \mathcal{X},\ (\boldsymbol{e}_i)_{i\in[k]} \leftarrow \left(\mathcal{D}_{P_i, \frac{qr}{\sqrt{n}}} \right)_{i\in[k]} \right\} \text{ and}$$

$$D_1 := \left\{ \left((P_i)_{i\in[k]}, (\boldsymbol{x}_i + \boldsymbol{e}_i)_{i\in[k]} \right) \;\middle|\; \boldsymbol{x}_1, \dots, \boldsymbol{x}_k \leftarrow \mathcal{X},\ (\boldsymbol{e}_i)_{i\in[k]} \leftarrow \left(\mathcal{D}_{P_i, \frac{qr}{\sqrt{n}}} \right)_{i\in[k]} \right\}.$$

[28] We could equally require a quadratic form Q with a dense sublattice here — The hardness of $\Delta\mathrm{LIP}_s^{S,Q}$ for appropriate pairs of S and Q ensures that this would be asymptotically equivalent.

The LPS_χ game corresponds to the distinguishing game $\mathrm{Exp}^{\mathrm{dist}}_{D_0,D_1}(\mathcal{A})$ presented in Definition 2.3. The advantage of an attacker $\mathcal{A}$ in the LPS_χ game with the lattice point distribution χ is therefore given by

$$\mathrm{Adv}^{\mathrm{LPS}_\chi}(\mathcal{A}) := \mathrm{Adv}^{\mathrm{dist}}_{D_0,D_1}(\mathcal{A}).$$

The LPS_χ problem is hard if $\mathrm{Adv}^{\mathrm{LPS}_\chi}(\mathcal{A})$ is negligible for any PPT attacker $\mathcal{A}$.

The idea behind using the LPS problem is that it allows us to circumvent the statistical issues encountered in Section 5.4.1. The first distribution D_0 corresponds to the adversary's input in the LIP-KEM_k's IND-CPA game. If LPS_χ is hard, we can swap D_1 in for D_0 in the proof to break the correlation between the ciphertexts. Given the right distribution χ for the lattice points, we could then once again resort to an entropy argument on the independent $\boldsymbol{x}_i$ to show that the extractor produces an encapsulated key with a distribution negligibly close to uniform. Assuming LPS_χ with an appropriate χ thus creates a clear proof strategy.

We motivate the hardness of LPS_χ using the variance in the public key lattices: The different unimodular matrices U_i drastically change the mapping of the coefficient vectors $\boldsymbol{x}_i$ and $\boldsymbol{c}_i = \boldsymbol{x}_i + \boldsymbol{e}_i$ to lattice vectors. As a result, it should be hard for an attacker to use the lattice geometry to tell whether the $\boldsymbol{x}_i$ are all the same or not. In particular, the hardness of LIP ensures that the attacker cannot even tell if the public keys P_i all correspond to the same lattice. This leaves only the coefficient representation $\boldsymbol{c}_i$ of the ciphertexts as a tool for distinguishing the distributions. If χ is chosen such that the coordinates of $\boldsymbol{x}_i$ are hidden by the error terms $\boldsymbol{e}_i$, we would expect distinguishing D_0 and D_1 using the coefficient vectors directly to be difficult as well.

In the lattice formulation, LPS_χ is functionally similar to LWE. As discussed in Section 5.4.2, the stacked ciphertext $\boldsymbol{c}_*$ of the LIP-KEM_k in the lattice formulation is structurally similar to an LWE instance. The LWE assumption states that its instances $(A, A\boldsymbol{x} + \boldsymbol{e})$ with a secret vector $\boldsymbol{x}$ are indistinguishable from randomness. The LPS assumption is comparable, but gives a weaker result: It states that IND-CPA challenge instances $((P_i)_{i\in[k]}, (\boldsymbol{x} + \boldsymbol{e}_i)_{i\in[k]})$ of the LIP-KEM_k are indistinguishable from instances where the singular secret vector $\boldsymbol{x}$ has been replaced with k different secret vectors.[29] We thus consider LPS to be a generalization of LWE to lattices in $\mathbb{R}^n$ instead of random matrices in $\mathbb{Z}_q^n$. This generalization accounts for the weaker result — there is no uniformly random distribution on $\mathbb{R}^n$ or $q^{-1}\mathbb{Z}^n$ that LIP-KEM_k ciphertexts could be indistinguishable from, nor would it be reasonable to assume that a tuple of the form $(\boldsymbol{x} + \boldsymbol{e}_i)_{i\in[k]}$ looks "uniformly random" over some large region of $q^{-1}\mathbb{Z}^n$. On the contrary, LPS_χ appears to be close to the minimal assumption that would need to hold for a successful IND-CPA proof of the LIP-KEM_k with correlated lattice points to exist: Section 5.4.1 shows that some method of decoupling the individual ciphertexts $\boldsymbol{c}_i$ is strictly required, and their only shared component that could be decoupled is $\boldsymbol{x}$. Nevertheless, two factors deter us from placing confidence in the utility of LPS, which we detail in the following: First, we show that a reduction from ΔLIP to LPS_χ fails; second, there is a lack of

[29] A straightforward reduction shows that the hardness of decisional LWE implies a similar result to LPS in the LWE setting: If $A\boldsymbol{x} + \boldsymbol{e} = (\langle \boldsymbol{a}_1, \boldsymbol{x}\rangle + e_1 \cdots)^T$ looks like randomness, then $(\langle \boldsymbol{a}_1, \boldsymbol{x}_1\rangle + e_1 \cdots)^T$ does too, where $\boldsymbol{a}_i$ is the i-th row of A. Since both distributions are indistinguishable from randomness, they are also indistinguishable from each other.

a candidate distribution $\mathcal{X}$ to plausibly instantiate $\mathrm{LPS}_{\mathcal{X}}$ with such that an entropy argument in the IND-CPA proof would be possible.

Reducing ΔLIP to $\mathrm{LPS}_{\mathcal{X}}$: Ideally, the hardness of the $\mathrm{LPS}_{\mathcal{X}}$ problem would be backed by a reduction from another hard lattice problem. A significant hurdle to any such reductions to $\mathrm{LPS}_{\mathcal{X}}$ is their dependence on $\mathcal{X}$: There are $\mathcal{X}$ such that $\mathrm{LPS}_{\mathcal{X}}$ is vacuously hard and $\mathcal{X}$ for which $\mathrm{LPS}_{\mathcal{X}}$ is trivial. To illustrate, consider the two simple distributions $\mathcal{X}_1$ and $\mathcal{X}_2$, where $X \sim \mathcal{X}_1$ has $\Pr[X = \boldsymbol{o}] = 1$ while $Y \sim \mathcal{X}_2$ has $\Pr[Y = \boldsymbol{o}] = 0.5$ and $\Pr[Y = \hat{\boldsymbol{e}}_1] = 0.5$. $\mathrm{LPS}_{\mathcal{X}_1}$ is vacuously hard since $D_0 = D_1$, while $\mathrm{LPS}_{\mathcal{X}_2}$ is trivial for $k > 1$: An attacker on $\mathrm{LPS}_{\mathcal{X}_2}$ can simply check if $\|\boldsymbol{c}_i\|_{P_i} \leq r \ \forall i \in [k]$ or $\|\boldsymbol{c}_i - \hat{\boldsymbol{e}}_1\|_{P_i} \leq r \ \forall i \in [k]$. If neither is the case, the challenge must be sampled from D_1. The same argument holds for any distribution with insufficient entropy and $D_0 \overset{s}{\not\approx} D_1$.

Even given a suitable $\mathcal{X}$, our options for hard lattice problems to reduce to $\mathrm{LPS}_{\mathcal{X}}$ are limited to ΔLIP for reasons mentioned at the start of this section and in Section 5.4.2. However, the ΔLIP problem is based on distinguishing lattices, not coefficient vectors for those lattices. In order to make use of ΔLIP, the differences between the coefficient vectors $\boldsymbol{x}$ and $\boldsymbol{x}_i$ in each of the k ciphertexts in D_0 and D_1 respectively would have to be turned into differences between quadratic forms P_i and P'_i: In the reduction, D_0 would sample using P_i and D_1 using P'_i such that $(\boldsymbol{x} + \boldsymbol{e}_i)_{i\in[k]}$ with $\boldsymbol{e}_i \leftarrow \mathcal{D}_{P_i, qr/\sqrt{n}}$ is indistinguishable from $(\boldsymbol{x}_i + \boldsymbol{e}_i)_{i\in[k]}$ with $\boldsymbol{e}_i \leftarrow \mathcal{D}_{P'_i, qr/\sqrt{n}}$.[30] This approach presents two issues:

- *Confusion of linear and affine transformations*: Changing lattices B_i or quadratic forms $P_i = B_i^T B_i$ is a fundamentally linear operation, moving from a $B_i = O_i B U_i$ to a $B'_i = O'_i B' U'_i = B'_i B_i^{-1} B_i$. On the other hand, going from $\boldsymbol{x}$ to $\boldsymbol{x}_i$ like in $\mathrm{LPS}_{\mathcal{X}}$ is an affine transformation since $\boldsymbol{x}_i + \boldsymbol{e}_i = \boldsymbol{x} + \boldsymbol{e}_i + (\boldsymbol{x}_i - \boldsymbol{x})$. Importantly, the affine translation in question is large, making it difficult to "hide" with a linear transformation: If finding short lattice vectors in $\Lambda(B_i)$ and $\Lambda(B'_i)$ is hard (which it must be for $\Delta\mathrm{LIP}^{P_i, P'_i}$ to be hard), $\|\boldsymbol{x} - \boldsymbol{x}_i\|_{P_i}$ and $\|\boldsymbol{x} - \boldsymbol{x}_i\|_{P'_i}$ will both be large with overwhelming probability for any choices of $\boldsymbol{x}, \boldsymbol{x}_i \leftarrow \mathcal{X}$. This applies independently of the distribution $\mathcal{X}$.
- *Long jump requirement*: Applying ΔLIP to move from D_0 to D_1 would have to involve a *long jump* for the same reasons the IND-CPA proof of a LIP-KEM_k requires one (see Section 5.2): A hybrid argument would leave the ΔLIP attacker $\mathcal{B}$ with no way of knowing whether to use the same $\boldsymbol{x}$ or a fresh $\boldsymbol{x}_i$ for the ciphertext $\boldsymbol{c}_i$ corresponding to the $\Delta\mathrm{LIP}^{P_i, P'_i}$ challenge instance. This means that all of the quadratic forms P_i need to be replaced at once. The only way of accomplishing this using ΔLIP would be to use the orthogonally concatenated quadratic form $\boldsymbol{P} \in S^{>0}_{nk}(\mathbb{Z})$ from (5.5) and replace that with some $\boldsymbol{P'} \in S^{>0}_{nk}(\mathbb{Z})$. However,

$$\boldsymbol{P} \nsim \mathcal{D}_s\left(\left[\begin{pmatrix} S & & 0 \\ & \ddots & \\ 0 & & S \end{pmatrix}\right]\right)$$

[30] Definition 5.2 has D_0 and D_1 use the same quadratic forms P_i. In the reduction, swapping from the P'_i back to the P_i would have to be accomplished through another instance of ΔLIP.

since samples from this distribution generally do not have a block-diagonal structure. Similarly, the potential replacement $\boldsymbol{P}'$ also would not have that structure and thus could not be separated into different P_i' for sampling the $\boldsymbol{e}_i$.

For these reasons, it seems unlikely that ΔLIP could be reduced to $\mathrm{LPS}_\mathcal{X}$. This does not necessarily preclude usage of the $\mathrm{LPS}_\mathcal{X}$ problem in a LIP-KEM$_k$, though it weakens our confidence in its hardness. In contrast, the instantiation of the distribution $\mathcal{X}$ presents an immediate and insurmountable problem for a potential LIP-KEM$_k$ based on $\mathrm{LPS}_\mathcal{X}$.

Candidate distributions: The distribution $\mathcal{X}$ over $\mathbb{Z}^n$ that the lattice point coefficient vectors are sampled from must meet two requirements for our use case. The first of these is that $\mathrm{LPS}_\mathcal{X}$ must be hard, which necessitates $\boldsymbol{x} \leftarrow \mathcal{X}$ being hidden by, and thus smaller than, the error terms $\boldsymbol{e}_i$. In addition to this, $\mathcal{X}$ must also be chosen such that, after making the ciphertexts independent using $\mathrm{LPS}_\mathcal{X}$ and switching to a lattice with a dense sublattice, $\boldsymbol{x}_i + \boldsymbol{e}_i$ with $\boldsymbol{x}_i \leftarrow \mathcal{X}$ has sufficient entropy for the extractor. Unfortunately, there does not appear to be a distribution that satisfies both of these requirements. In the following, we consider three categories of distributions and show that each either fails to be samplable, does not provide enough entropy, or makes $\mathrm{LPS}_\mathcal{X}$ easy.

- *Lattice-specific discrete Gaussians*: A similar approach to that of Section 5.3 would entail sampling the $\boldsymbol{x}$ either from the lattices' discrete Gaussians $\mathcal{D}_{P_i,s'}$ directly or from some lattice-independent discretization like $B_i^{-1}\mathcal{D}_{\mathbb{1}_n,s'}$. Predictably, using the same method as for the correlated-errors approach results in the same problems: Correlating the $\mathcal{D}_{P_i,s'}$ is not possible (so we could not sample an $\boldsymbol{x}$ for D_0) and transforming samples using the B_i^{-1} does not result in integer vectors. The category of lattice-specific discrete Gaussians thus does not contain a viable instantiation for $\mathcal{X}$.[31]
- *Distributions with small* $\|\boldsymbol{x}\|_\infty$: For the $\mathrm{LPS}_\mathcal{X}$ problem to be hard, the coordinates of $\boldsymbol{x} \leftarrow \mathcal{X}$ must be significantly smaller than the coordinates of the $\boldsymbol{e}_i$ that hide them. Effectively, this means the maximum norm $\|\boldsymbol{x}\|_\infty$ of $\boldsymbol{x} \leftarrow \mathcal{X}$ should be bounded. However, $\boldsymbol{x}_i + \boldsymbol{e}_i$ is unlikely to have the necessary entropy when using these distributions. To see why, recall that $\boldsymbol{x} \sim \mathcal{D}_{P,r/\sqrt{n},\boldsymbol{c}}$ in the proof of the original LIP-KEM in Theorem 5.1. This holds because each candidate $\boldsymbol{x}'$ is bijectively linked to an $\boldsymbol{e}'$ that could have been produced by the discrete Gaussian distribution. If $\boldsymbol{x}_i \sim \mathcal{X}$, that is no longer the case: In addition to the constraint the discrete Gaussian $q^{-1}\mathcal{D}_{P_i,r/\sqrt{n},\boldsymbol{c}_i}$ imposes on which $\boldsymbol{x}_i'$ are likely to be hidden in $\boldsymbol{c}_i$, we have an additional constraint due to $\boldsymbol{x}_i \sim \mathcal{X}$. From the adversary's perspective (i.e., for a fixed $\boldsymbol{c}_i$), the effective probability of a candidate $\boldsymbol{x}_i' \in \mathbb{Z}^n$ being the true $\boldsymbol{x}_i$ is governed by

$$\Pr[\boldsymbol{x}_i = \boldsymbol{x}_i' \mid \boldsymbol{c}_i = \boldsymbol{x}_i + \boldsymbol{e}_i] \propto \underset{\boldsymbol{e}\sim\frac{1}{q}\mathcal{D}_{P_i,\frac{qr}{\sqrt{n}}}}{\Pr}[\boldsymbol{e} = \boldsymbol{c}_i - \boldsymbol{x}_i'] \cdot \underset{\boldsymbol{x}\sim\mathcal{X}}{\Pr}[\boldsymbol{x} = \boldsymbol{x}_i']$$

[31] Even if the $\mathcal{D}_{P_i,s'}$ could be used to sample $\boldsymbol{x}$, this coefficient vector would not be hidden by the error terms $\boldsymbol{e}_i$ since those follow the same distribution, but scaled down by a factor of q. The sum $\boldsymbol{x} + \boldsymbol{e}_i$ would thus leave the most significant $\log_2(q)$ bits of $\boldsymbol{x}$ exposed.

due to Bayes' theorem. The entropy of $\boldsymbol{x}_i$ is thus only large if there are a large number of candidates $\boldsymbol{x}_i'$ that could have been sampled from $\mathcal{X}$ and for which $\left\|\boldsymbol{c}_i - \boldsymbol{x}_i'\right\|_{P_i} \leq r$. For a distribution $\mathcal{X}$ with small $\|\boldsymbol{x}\|_\infty$, this is not likely since $\left\|\boldsymbol{x}_i - \boldsymbol{x}_i'\right\|_{P_i} > 2r$ for the vast majority of coefficient vectors — there simply will not be enough $\boldsymbol{x} \in \mathrm{supp}(\mathcal{X})$ for the overlap with the ball of radius r around $\boldsymbol{c}_i$ in the lattice to be large. For the case of $\mathcal{X} = \mathcal{U}(\mathbb{Z}_2^n)$, this is visualized in Figure 5.5.

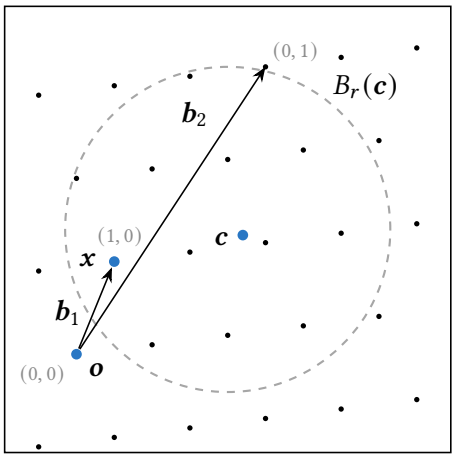


Figure 5.5: Example showing how sampling $\boldsymbol{x} \leftarrow \mathcal{U}(\mathbb{Z}_2^n)$ can cause $\boldsymbol{x}$ to have little entropy despite the lattice being dense. Possible choices of $\boldsymbol{x}$ in the bounds of the image are labeled in gray. The points $\boldsymbol{x}$ and $\boldsymbol{c}$ have been transformed into the lattice for this visualization.

- *Distributions with high entropy*: The complementary approach to the above would be to deliberately choose $\mathcal{X}$ such that $\boldsymbol{x}_i$ has high entropy given a ciphertext $\boldsymbol{c}_i = \boldsymbol{x}_i + \boldsymbol{e}_i$ for a quadratic form P_i with a dense sublattice. For an $\mathcal{X}$ to meet this criterion, most of the $\boldsymbol{x}_i'$ with $\left\|\boldsymbol{c} - \boldsymbol{x}_i'\right\|_{P_i} \leq r$ must be samplable from $\mathcal{X}$ with a sufficient probability. We can use this to determine how "wide" $\mathcal{X}$ must be, i.e., what the required bounds on $\|\boldsymbol{x}'\|_\infty$ for $\boldsymbol{x}' \leftarrow \mathcal{X}$ are. Using (5.7), we know that $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{d}$ for $\|\boldsymbol{d}\|_{P_i} \leq 2r$ and derive

$$\begin{aligned}
\|\boldsymbol{x}'\|_\infty &= \|\boldsymbol{x} + \boldsymbol{d}\|_\infty \\
&\leq \|\boldsymbol{x}\|_\infty + \|\boldsymbol{d}\|_\infty \\
&\leq \|\boldsymbol{x}\|_\infty + \left\|B_i^{-1} B_i \boldsymbol{d}\right\|_2 \\
&\leq \|\boldsymbol{x}\|_\infty + \left\|B_i^{-1}\right\|_2 \|B_i \boldsymbol{d}\|_2 \\
&\leq \|\boldsymbol{x}\|_\infty + 2r \left\|B_i^{-1}\right\|_2,
\end{aligned}$$

where the second inequality follows from

$$\|\boldsymbol{y}\|_2 = \sqrt{\sum_{i=0}^{n} y_i^2} \geq \sqrt{\max_{i \in [n]} y_i^2} = \max_{i \in [n]} |y_i| = \|\boldsymbol{y}\|_\infty \quad \forall \boldsymbol{y} \in \mathbb{R}^n.$$

If all of these $\boldsymbol{x}'$ can be sampled from $\mathcal{X}$ with sufficient probability independently of the quadratic form P_i, we have $\|\boldsymbol{x}\|_\infty \in \Theta(r \left\|B_i^{-1}\right\|_2)$ for many $\boldsymbol{x} \leftarrow \mathcal{X}$. However, $\boldsymbol{x}$ with

coordinates this large are not hidden by the $\boldsymbol{e}_i$: Table 5.1 shows typical values of both $\|\boldsymbol{e}_i\|_\infty$ and $r\left\|B_i^{-1}\right\|_2$ for quadratic forms $P_i \leftarrow \mathcal{D}_s([\mathbb{1}_n])$ and error terms $\boldsymbol{e}_i \leftarrow \mathcal{D}_{P_i,\frac{qr}{\sqrt{n}}}$, demonstrating that $\|\boldsymbol{x}\|_\infty \gg \|\boldsymbol{e}_i\|_\infty$ for many $\boldsymbol{x} \leftarrow \mathcal{X}$. $\mathrm{LPS}_\mathcal{X}$ is thus trivial for $\mathcal{X}$ with an entropy guarantee.

We conclude that there is likely no distribution $\mathcal{X}$ such that $\mathrm{LPS}_\mathcal{X}$ is hard and the LIP-KEM's entropy argument is possible — the two constraints are mutually exclusive. This makes it infeasible to construct a LIP-KEM$_k$ with correlated lattice points using $\mathrm{LPS}_\mathcal{X}$ in $\mathbb{R}^n$.

The difficulty in finding an appropriate $\mathcal{X}$ for $\mathrm{LPS}_\mathcal{X}$ mainly arises due to the fact that $\mathbb{R}^n$ and the quadratic forms P_i of the lattices in that space are unbounded. This permits the large discrepancies between the individual lattices we refer to as the Lattice Incompatibility Problem. Compare this to the LWE setting, where the problem does not occur precisely because everything is limited to the compact space $\mathbb{Z}_q^n$. A possible loophole could therefore be to find a similarly compact space $\mathbb{D} \subset \mathbb{R}^n$ to use with a LIP-KEM$_k$ and $\mathrm{LPS}_\mathcal{X}$. This LIP-KEM$_k$ would have $\boldsymbol{x} \in \mathbb{D} \cap \mathbb{Z}^n$ and likewise $\boldsymbol{c}_i \in \mathbb{D}\ \forall i \in [k]$. For this LIP-KEM$_k$ to function, we must be able to efficiently perform arithmetic in $\mathbb{D}$ — otherwise, there would be no way to sample the $\boldsymbol{e}_i$, calculate $\boldsymbol{c}_i \leftarrow \boldsymbol{x} + \boldsymbol{e}_i$, or decode $\boldsymbol{c}_i$ to $\boldsymbol{x}$. This constraint on $\mathbb{D}$ severely limits our options, especially given that $\mathbb{Z}_q^n$ is out of the question since it denatures the lattices into codes. We also identify the following additional requirements on $\mathbb{D}$:

- It must be independent of the individual lattices $\Lambda(B_i)$ to allow us to sample a common $\boldsymbol{x} \in \mathbb{D}$.
- At the same time, it should be "compatible" with each of the $\Lambda(B_i)$ in the sense that applying $\mathbb{D}$'s arithmetic to the lattice basis still results in a decodable lattice. Figure 5.6 shows an example of how an incompatible space can cause a lattice to "wrap over itself".
- We must be able to efficiently sample $\boldsymbol{x} \in \mathbb{D}$, which precludes any approaches that require finding short lattice vectors.

					Est. visibility of $\boldsymbol{x}$	
n	s	q	$\|\boldsymbol{e}\|_\infty$	$r\left\|B^{-1}\right\|_2$	Bits [b]	Fraction [%]
10	1.32	35	$4.23{\cdot}10^{-1}$	$4.55{\cdot}10^{0}$	2	100
50	1.65	273	$2.04{\cdot}10^{13}$	$6.37{\cdot}10^{22}$	32	42
100	1.78	632	$5.09{\cdot}10^{12}$	$1.88{\cdot}10^{64}$	170	80

Table 5.1: Experimental data comparing the magnitude $\|\boldsymbol{e}\|_\infty$ of the coordinates of $\boldsymbol{e} \leftarrow \mathcal{D}_{P,qr/\sqrt{n}}$ with the magnitude of the coordinates of $\boldsymbol{x}$ sampled with $\|\boldsymbol{x}\|_\infty \in \Theta\left(r\left\|B^{-1}\right\|_2\right)$. Ten different public keys $P = B^TB \leftarrow \mathcal{D}_s([\mathbb{1}_n])$ and one error term $\boldsymbol{e}$ per public key were sampled for each n with s and q chosen as in the LIP-KEM. The decoding radius is $r = 0.5$. Each entry in the table represents the mean of the column's metric over those ten samples. The visibility of $\boldsymbol{x}$ in bits is estimated as $\log_2\left(r\left\|B^{-1}\right\|_2\right) - \max\left\{\log_2(\|\boldsymbol{e}\|_\infty), 0\right\}$ and corresponds to the approximate number of most significant bits of $\boldsymbol{x}$ that remain unchanged in the sum $\boldsymbol{c} = \boldsymbol{x} + \boldsymbol{e}$. The visibility percentage relates this metric to the total bit length of $\boldsymbol{x}$.

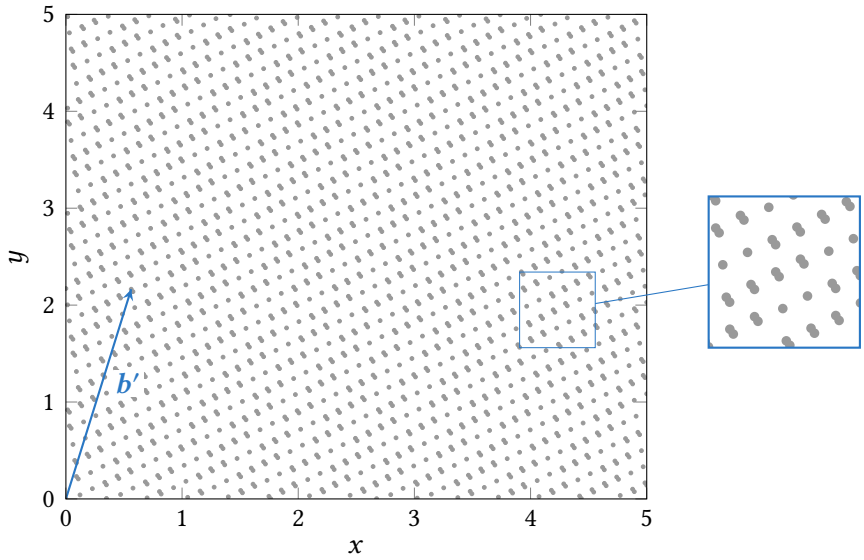


Figure 5.6: The two-dimensional lattice $\Lambda(\boldsymbol{b})$ of rank 1 with $\boldsymbol{b} = \binom{1}{2}$, when rotated counterclockwise by 12°, misaligns with the space $\mathbb{R}^2/5\mathbb{Z}^2 = [0,5)^2$ to create an irregular, non-repeating pattern that does not correspond to a decodable lattice. The figure displays the first 2000 multiples of the rotated vector $\boldsymbol{b}'$. Note how many of the points overlap without being identical.

We are not aware of any candidates for $\mathbb{D}$ that are not $\mathbb{Z}_q^n$, isomorphic to $\mathbb{Z}_q^n$, or encounter the same issues discussed in Section 5.4.2. For example, Leporati, Rovida, and van Woerden [51] construct an ambient space $\mathcal{D} \supset q\Lambda$ for any lattice Λ such that $\mathcal{D}/q\Lambda$ has a group structure and would be a candidate for $\mathbb{D}$. However, this candidate fails to be suitable for our use case since it is both isomorphic to a subgroup of $\mathbb{Z}_q^n$ [51] and lattice-dependent. More generally, it is unclear how a viable $\mathbb{D}$ would be lattice-independent while retaining compatibility with arbitrary $O_i \in O_n(\mathbb{R})$ misaligning the lattices as in Section 5.3.2. Conversely, Ducas and van Woerden [33] argue that limiting the O_i to signed permutations makes LIP significantly easier and has led to several broken cryptosystems in the literature. It therefore stands to reason that a suitable compact space $\mathbb{D}$ does not exist.

Concluding this section, we see that every discussed approach fails to enable an entropy argument for $\boldsymbol{x}$ such that the extractor can produce a uniformly random encapsulated key. More specifically, $\boldsymbol{x}$ does not have the required entropy by itself (Section 5.4.1), existing hard lattice problems like BDD or LWE do not allow for successful reductions (Section 5.4.2), and even the hypothetical assumption LPS_χ cannot be securely instantiated (Section 5.4.3). Considering these failed approaches and the foundational reasons behind their failure, an entropy argument for the LIP-KEM$_k$ with correlated lattice points appears to be impossible — the incompatibilities between the public-key lattices prevent any attempt from working. In addition, as with the correlated-errors approaches in Section 5.3, there is no available alternate proof strategy besides an entropy argument to establish IND-CPA security for the LIP-KEM$_k$. We thus make the following claim:

Conjecture 3. *There is no verifiable, IND-CPA-secure LIP-KEM$_k$ based on correlated lattice points.*

5.5 External Correlations

In the previous sections, we discuss correlating an existing component of the LIP-KEM to achieve verifiability. In contrast, the idea behind using an external correlation is to add a new component to the LIP-KEM$_k$ that encapsulates the correlation between the ciphertexts. This approach need not be fully black-box — it can still take advantage of the LIP-KEM's structure, but the error terms $\boldsymbol{e}_i$ and lattice point coefficients $\boldsymbol{x}_i$ remain uncorrelated. By externalizing the correlation, one would hope to avoid the entropy issues encountered by the approaches in Section 5.3 and Section 5.4 while still attaining verifiability. Unfortunately, this approach also has a fundamental issue that prevents any LIP-KEM$_k$ from using external correlations for verifiability in the standard model. In the following, we discuss the general construction of a LIP-KEM$_k$ with external correlations and demonstrate its infeasibility with an abstract argument.

Without loss of generality, we assume that the additional component added to the LIP-KEM$_k$ takes the form of some auxiliary information aux_i in each instance's ciphertext $c_i = (\boldsymbol{c}_i, Z, aux_i)$. As opposed to the LIP-KEM$_k$ variants in the previous sections, this LIP-KEM$_k$ does not encapsulate the same key k times because the $\boldsymbol{c}_i$ are completely independent in this construction. This divergence from Definition 4.1 can be addressed by adding a layer of indirection: Instead of the output of each instance being the LIP-KEM's encapsulated key ek_i, one can use that key to symmetrically encrypt a common message m (to form a PKE$_k$) or a common key mk (to form a KEM$_k$) as is typical for hybrid encryption schemes. This k-repeated scheme's $\textsc{Dec}_{\textsc{Ver}}$ algorithm must decrypt and verify both the k LIP-KEM instances as well as the m or mk encrypted using the ek_i. Hereafter, we ignore the verification of these added symmetric ciphertexts as it mechanically follows from the verification of the LIP-KEM instances.

Since the aux_i encapsulate the correlation between the c_i, we can infer how they must behave from the IND-CPA and verifiability properties that the LIP-KEM$_k$ must satisfy: Given one secret key U_j, $\textsc{Dec}_{\textsc{Ver}}$ learns ek_j and must use both the information in aux_j as well as the other aux_i to verify the k-repeated ciphertext. Vice versa, the aux_i must not leak any information about the ek_i to an adversary that does not know any of the secret keys in order to retain IND-CPA security. Stronger security models like the CRS model or ROM can instantiate the auxiliary information with a non-interactive zero-knowledge proof of the well-formedness of the ciphertext. Doing so permits verification even without knowing any secret keys and without leaking any information due to the zero-knowledge property [30]. The impossibility of such proofs in the standard model [42, Theorem 4.3] implies that it is strictly necessary to lock the information required for verification behind access to a secret (in this case, any one U_i) — otherwise, the "proof of well-formedness" would be in the clear, and since it is not zero-knowledge, it would leak non-negligible amounts of information to an adversary. We call the information used to perform the verification the *verification secret* vs, reflecting how it must be hidden from the adversary.[32] For example, vs could be an encryption key used to symmetrically encrypt each $\boldsymbol{e}_i$ with the ciphertexts stored in the aux_i. Given access to vs, one could then verify the instance's ciphertexts by symmetrically

[32] Not to be confused with the verification key vk belonging to the signature scheme used in Definition 4.3.

decrypting the $\boldsymbol{e}_i$, checking that $\|\boldsymbol{e}_i\|_{P_i} \leq r$ and $\boldsymbol{c}_i - \boldsymbol{e}_i \in \mathbb{Z}^n$, and extracting the encapsulated key $ek_i = \mathcal{E}(\boldsymbol{e}_i, Z)$.

Hiding vs from the adversary must be done such that any one secret key U_j ultimately grants access to it — else, DEC_{VER} would not be able to use it. We identify exactly two options for doing so without introducing additional asymmetric primitives, both of which encounter unsolvable problems:

- *Hide vs using each P_i*: The obvious choice for hiding vs is to encrypt it using the LIP-KEM as an already-existing asymmetric scheme. In general, this could be implemented by encrypting vs symmetrically with k new encapsulated keys ek'_i for k new instances of the LIP-KEM. Closer inspection reveals that this approach actually recursively creates a new LIP-KEM$'_k$ encrypting the verification secret vs for the original LIP-KEM$_k$. The new LIP-KEM$'_k$, in turn, must also be IND-CPA-secure (to hide vs) and verifiable to ensure that the correct vs can be recovered given any secret key U_j. Since the LIP-KEM$'_k$ would thus also require its own verification secret vs', this recursion makes no progress and does not terminate.
- *Hide vs using each ek_i*: The alternative to hiding vs using the public keys is to use the encapsulated keys ek_i instead.[33] For instance, vs could be symmetrically encrypted once using each ek_i and the ciphertexts stored in the aux_i. However, this creates the cryptographic equivalent of a dependency cycle: The ek_i are used as keys to hide vs, but vs, as the verification secret, hides information that would reveal the ek_i. In a security proof, this results in a situation where the IND-CPA game for the symmetric encryption scheme used to hide vs cannot be simulated without the ek_i being independently random (such that they could be sampled by the IND-CPA challenger), but the ek_i are not independently random as long as vs could be used to reveal them. The same logic applies in reverse, where any security properties using vs as a secret cannot be used in a reduction until vs is independent of the aux_i, but that is not the case while the aux_i contain a symmetric ciphertext encrypting vs with the ek_i as the keys. Without a method of breaking this cycle, progressing the IND-CPA proof of the LIP-KEM$_k$ becomes impossible. We are not aware of any such method.

Seeing that there is no viable way to hide vs such that an IND-CPA proof of the LIP-KEM$_k$ would be possible, we once again conclude that external correlations cannot produce a secure LIP-KEM$_k$. Remarkably, no part of our argument in this section actually relies on any specific feature of the LIP-KEM. We therefore end this section with a more general conjecture:

Conjecture 4. *There is no verifiable and IND-CPA-secure KEM$_k$ or PKE$_k$ based on a single IND-CPA-secure KEM or PKE with external correlations in the standard model.*

[33] Any secret component of the LIP-KEM (e.g., the $\boldsymbol{e}_i$ or $\boldsymbol{x}_i$) could technically be used here, but they are functionally equivalent to the ek_i for this purpose.

5.6 Trapdoor Functions

In this section, we undertake the final attempt at building IND-CCA2-secure public-key encryption from the LIP-KEM in this thesis and show that it fails as well. For this endeavor, we take inspiration from Rosen and Segev's [84] work on k-repetition with TDOWFs. We briefly introduce their approach, then demonstrate that the LIP-KEM cannot be used to instantiate the TDOWFs required for it.

The k-repetition construction by Rosen and Segev [84] uses the same technique as in Definition 4.3 to achieve IND-CCA2 security, but replaces the KEM_k or PKE_k with a concatenation of multiple TDOWFs with the same input from a family (G, F, F^{-1}) of TDOWFs. Essentially, k instances $(F(s_1, \cdot), \ldots, F(s_k, \cdot))$ of the TDOWF are evaluated on an input $x \leftarrow \mathcal{I}$ following the TDOWF family's input distribution $\mathcal{I}$ to produce an output $(F(s_1, x), \ldots, F(s_k, x))$. This construction is easily verifiable given one trapdoor td_j by inverting the j-th TDOWF to recover x, then re-evaluating $F(s_i, x)$ for every $i \neq j$ and checking whether the results are the same as in the given output. For security, Rosen and Segev [84] introduce the concept of security for TDOWFs under correlated products: A family of TDOWFs is considered secure *under the uniform k-repetition distribution* if the function $(F(s_1, \cdot), \ldots, F(s_k, \cdot))$ is one-way for the input distribution $\mathcal{I}$.[34] Using the Goldreich-Levin theorem [40] to ensure the existence of a hardcore predicate h, a single bit $m \in \{0, 1\}$ can be encrypted by sampling an $x \leftarrow \mathcal{I}$ and outputting $(F(s_1, x), \ldots, F(s_k, x))$ along with $m \oplus h(s_1, \ldots, s_k, x)$.[35] This scheme is IND-CPA-secure due to the hardcore predicate being indistinguishable from a random bit. Definition 4.3 can now be analogously applied to create an IND-CCA2-secure public-key encryption scheme following Rosen and Segev [84].[36]

An IND-CCA2-secure k-repetition construction can therefore be created using a family of TDOWFs that is secure under the uniform k-repetition distribution. Some TDOWFs satisfy this requirement by design, such as the LWE-based TDOWF by Peikert [74]. Rosen and Segev [84, Theorem 3.3] also show that (λ, l)-lossy TDOWFs with a sufficiently large l are secure under the uniform k-repetition distribution in a fully black-box manner — the proof relies on the concatenation of these lossy TDOWFs still being a lossy TDOWF as a whole. Mol and Yilek [66] expand on this result by proving that an adapted form of k-repetition can be used with (λ, l)-lossy TDOWFs that are only slightly lossy, i.e., where $l = 1/\mathrm{poly}(\lambda)$. It follows that standard-model IND-CCA2 security using the LIP-KEM could be achieved by building a slightly lossy TDOWF or a TDOWF secure under the uniform k-repetition distribution by other means. Doing so is potentially easier than building a $\mathrm{LIP\text{-}KEM}_k$ since the scheme must only achieve one-wayness or lossiness, not full IND-CPA security. Despite this, we show that the LIP-KEM still cannot be used to build a TDOWF fulfilling these requirements.

[34] We abbreviate the definition of security under correlated products here for the sake of clarity. The more general definition can be found in the original work by Rosen and Segev [84].

[35] Here, $\oplus$ is the XOR operator on bits.

[36] This scheme can be extended to encrypt multiple bits both generically using a transformation by Myers and Shelat [67] or using an extension to k-repetition by Rosen and Segev [84].

First, note that a LIP-KEM-based scheme cannot formally be a suitable TDOWF because it would fail to be a family of functions with a well-defined input distribution in the first place: The main action of the LIP-KEM is to calculate a ciphertext $\boldsymbol{c} \leftarrow \boldsymbol{x} + \boldsymbol{e}$ for some $\boldsymbol{x} \in \mathbb{Z}^n$ and an $\boldsymbol{e} \leftarrow q^{-1}\mathcal{D}_{P,qr/\sqrt{n}}$. Neither $\boldsymbol{x}$ nor $\boldsymbol{e}$ is suitable as an input to a LIP-KEM-based TDOWF: The error term $\boldsymbol{e}$'s distribution is dependent on the public key (and cannot be sampled independently of it, as we show in Section 5.3), so there is no suitable $\mathcal{I}$ with $\boldsymbol{e} \sim \mathcal{I}$ for arbitrary public keys $P \leftarrow \mathcal{D}_s([S])$. Meanwhile, choosing $\boldsymbol{x}$ as the input would leave $\boldsymbol{e}$ to be sampled as part of the TDOWF, thus making the output depend on randomness; this contradicts the definition of a function as a deterministic mapping of inputs to outputs.

These definitional issues could be solved by adjusting the TDOWF definition to admit internal randomness[37] such that $\boldsymbol{x}$ could be used as an input to a "LIP-KEM-TDOWF". However, both approaches towards achieving security under the uniform k-repetition distribution would encounter more problems. A direct proof is hindered by the same issues we discuss in Section 5.4, while a lossiness argument for the LIP-KEM does not appear to be possible by the following reasoning: Although Ducas and van Woerden describe the LIP-KEM's IND-CPA proof as a "lossy trapdoor argument" [33], it is lossy in a weaker sense than required for a lossy TDOWF: In the LIP-KEM's proof, switching to a lattice with a dense sublattice (i.e., switching to a descriptor of a "lossy function") does lose information — this is exactly what the entropy argument shows. However, it does not reduce the size of the image. That size is given by $\left|B_r(\boldsymbol{o}) \cap \Lambda(B)\right|$ for any basis B with $B^TB = S$ in the injective case and $B^TB = Q$ in the "lossy" case.[38] Notably, the increased density of the lattice with a dense sublattice actually *increases* the size of the image. The LIP-KEM is therefore not lossy in the manner of a lossy TDOWF.[39] We infer the following:

Conjecture 5. *There is no direct adaptation of the LIP-KEM into a TDOWF that is secure under the uniform k-repetition distribution.*

Combining the results across all of the previous sections (see Conjectures 2, 3, 4, and 5), we come to the final conjecture that:

Conjecture 6. *There is no IND-CCA2-secure KEM or PKE directly based on the LIP-KEM in the k-repetition framework in the standard model.*

[37] This requires careful analysis — see the work of Hemenway and Ostrovsky [43] for examples.

[38] We discount the negligible probability of $\|\boldsymbol{e}\|_P > r$ here. It could be addressed by simply rejecting any such $\boldsymbol{e}$ and sampling anew.

[39] Another way to see this is the following: If the LIP-KEM were indeed lossy in the required manner, the statistical argument discussed in Section 5.4.1 would work since a suitable concatenation of lossy TDOWFs is still lossy.

6 Conclusion

As it stands, a standard-model IND-CCA2-secure asymmetric encryption scheme based on LIP has yet to be devised. In this work, we make the claim that combining the LIP-KEM with k-repetition cannot lead to a such a construction. We provide strong evidence to support this conjecture in the form of an extensive analysis across all components of the LIP-KEM. In this manner, we channel further research efforts towards more successful LIP-based chosen-ciphertext-secure schemes. Furthermore, our heuristic investigation into the properties of LIP public-key lattices and the behavior of coefficient vectors in those lattices serves as a heretofore missing, approachable introduction to average-case LIP lattices.

As part of this work, we identify limitations in the state of the art on LIP as a cryptographic tool: While it benefits from its worst-to-average-case reduction and use of decodable lattices, current applications are limited to applying LIP either as a group action (as which it is comparatively weak) or to switch between an easily-decodable lattice and a lattice with a dense or sparse sublattice. More specifically, as of writing, the only use of LIP in the literature outside of group-action schemes is to vary the density of a lattice, then apply a statistical argument. This lack of flexibility is a major contributor to our impossibility argument for an IND-CPA-secure LIP-KEM$_k$; there is simply no other available proof strategy besides an entropy argument. We consider developing additional applications for LIP (e.g., manipulating other lattice attributes besides density) to be future work.

Another limitation of LIP is the lack of control over the public key; this is the second major driver of our impossibility conjecture. A public-key lattice in average-case LIP is hard to bound or to harmonize with other public-key lattices. This problem is unlikely to be fixable since the hardness of LIP states that it is computationally difficult to recognize if two different public keys are even in the same equivalence class. Nevertheless, this behavior is a strong deterrent to basing cryptographic schemes that would require combining multiple lattices on LIP.

In addition to our contributions regarding LIP, we also put the k-repetition framework as applied to IND-CPA-secure PKEs and KEMs on solid theoretical footing by establishing necessary requirements for the corresponding k-repeated schemes. In particular, we show that k-repetition cannot be applied to arbitrary PKEs or KEMs in a black-box manner. Thanks to our formal analysis of the requirements of k-repetition and the incompatibility properties of the LIP-KEM, our work is instructive to both authors using k-repetition in their research and those attempting to create LIP-based constructions using multiple lattices at once. Moreover, we present a complete version of a discrete Gaussian sampling algorithm for arbitrary quadratic forms including a proof of correctness and an implementation, discovering and fixing a widespread error in the bound on the scaling parameter in the process. Ours is the first published exact sampler for quadratic forms to avoid taking the Cholesky decomposition.

Of course, our results beg the question of what other techniques could be applied to LIP or an LIP-based scheme in order to actually achieve standard-model IND-CCA2 security. These techniques would likely need to avoid directly correlating multiple instances of schemes like the LIP-KEM. One potential approach might be to take advantage of the homomorphic encryption schemes proposed in the related work: If these schemes could, for example, be adapted to have the properties required by one of the constructions of Hemenway and Ostrovsky [43], then LIP would imply chosen-ciphertext-secure PKEs in the standard model. Alamati et al. [6] similarly provide transformations from some types of homomorphic schemes to IND-CCA2-secure asymmetric encryption. We leave the investigation of these approaches as future work.

Bibliography

[1] Léo Ackermann, Adeline Roux-Langlois, and Alexandre Wallet. "Public-Key Encryption from the Lattice Isomorphism Problem". In: *WCC 2024 - The Thirteenth International Workshop on Coding and Cryptography*. Perugia, Italy, June 2024, pp. 1–11. URL: https://inria.hal.science/hal-04924507.

[2] Dorit Aharonov and Oded Regev. "Lattice problems in NP ∩ coNP". In: *J. ACM* 52.5 (Sept. 2005), pp. 749–765. ISSN: 0004-5411. DOI: 10.1145/1089023.1089025.

[3] G. Alagic et al. *Status Report on the First Round of the Additional Digital Signature Schemes for the NIST Post-Quantum Cryptography Standardization Process*. Tech. rep. NIST IR 8528. Gaithersburg, MD, USA: National Institute of Standards and Technology, Oct. 2024. DOI: 10.6028/NIST.IR.8528.

[4] G. Alagic et al. *Status Report on the Fourth Round of the NIST Post-Quantum Cryptography Standardization Process*. Tech. rep. NIST IR 8545. Gaithersburg, MD, USA: National Institute of Standards and Technology, Mar. 2025. DOI: 10.6028/NIST.IR.8545.

[5] Navid Alamati et al. "Cryptographic Group Actions and Applications". In: *Advances in Cryptology – ASIACRYPT 2020*. Ed. by Shiho Moriai and Huaxiong Wang. Cham: Springer International Publishing, 2020, pp. 411–439. ISBN: 978-3-030-64834-3.

[6] Navid Alamati et al. "Minicrypt Primitives with Algebraic Structure and Applications". In: *Advances in Cryptology – EUROCRYPT 2019*. Ed. by Yuval Ishai and Vincent Rijmen. Cham: Springer International Publishing, 2019, pp. 55–82. ISBN: 978-3-030-17656-3.

[7] Larry C. Andrews. "Other Functions Defined by Integrals". In: *Special Functions of Mathematics for Engineers*. Second. Bellingham, WA, USA: SPIE Optical Engineering Press, 1998, pp. 109–140. ISBN: 0-8194-2616-4.

[8] Benny Applebaum et al. "Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems". In: *Advances in Cryptology - CRYPTO 2009*. Ed. by Shai Halevi. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 595–618. ISBN: 978-3-642-03356-8.

[9] L. Babai. "On Lovász' lattice reduction and the nearest lattice point problem". In: *Combinatorica* 6.1 (1986), pp. 1–13. ISSN: 1439-6912. DOI: 10.1007/BF02579403.

[10] E. S. Barnes and G. E. Wall. "Some extreme forms defined in terms of Abelian groups". In: *Journal of the Australian Mathematical Society* 1.1 (1959), pp. 47–63. ISSN: 2059-9161. DOI: 10.1017/S1446788700025064.

[11] Benjamin Benčina et al. “Properties of Lattice Isomorphism as a Cryptographic Group Action”. In: *Post-Quantum Cryptography*. Ed. by Markku-Juhani Saarinen and Daniel Smith-Tone. Cham: Springer Nature Switzerland, 2024, pp. 170–201. ISBN: 978-3-031-62743-9.

[12] John J. Benedetto and Georg Zimmermann. “Sampling multipliers and the Poisson Summation Formula”. In: *Journal of Fourier Analysis and Applications* 3.5 (1997), pp. 505–523. ISSN: 1531-5851. DOI: 10.1007/BF02648881.

[13] Huck Bennett and Chris Peikert. *Hardness of Bounded Distance Decoding on Lattices in ℓ_p Norms*. 2020. arXiv: 2003.07903 [cs.CC].

[14] Huck Bennett and Kaung Myat Htay Win. “Relating code equivalence to other isomorphism problems”. In: *Designs, Codes and Cryptography* 93.3 (2025), pp. 701–723. ISSN: 1573-7586. DOI: 10.1007/s10623-024-01542-3.

[15] Huck Bennett et al. “Just How Hard Are Rotations of $\mathbb{Z}^n$? Algorithms and Cryptography with the Simplest Lattice”. In: *Advances in Cryptology – EUROCRYPT 2023*. Ed. by Carmit Hazay and Martijn Stam. Cham: Springer Nature Switzerland, 2023, pp. 252–281. ISBN: 978-3-031-30589-4.

[16] Maiara F. Bollauf, Maja Lie, and Cong Ling. “On Gaussian Sampling for q-ary Lattices and Linear Codes with Lee Weight”. In: *Advances in Cryptology – CRYPTO 2025*. Ed. by Yael Tauman Kalai and Seny F. Kamara. Cham: Springer Nature Switzerland, 2025, pp. 321–352. ISBN: 978-3-032-01855-7.

[17] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. Jan. 2023. URL: https://toc.cryptobook.us (visited on 10/10/2025).

[18] Zdravko Botev and Pierre L’Ecuyer. “Simulation from the Normal Distribution Truncated to an Interval in the Tail”. In: *Proceedings of the 10th EAI International Conference on Performance Evaluation Methodologies and Tools*. Ed. by Antonio Puliafito et al. Taormina, Italy: ACM, May 2017. ISBN: 978-1-63190-141-6.

[19] Xavier Boyen, Malika Izabachène, and Qinyi Li. “Secure Hybrid Encryption in the Standard Model from Hard Learning Problems”. In: *Post-Quantum Cryptography*. Ed. by Jung Hee Cheon and Jean-Pierre Tillich. Cham: Springer International Publishing, 2021, pp. 399–418. ISBN: 978-3-030-81293-5.

[20] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. “(Leveled) Fully Homomorphic Encryption without Bootstrapping”. In: *ACM Trans. Comput. Theory* 6.3 (July 2014). ISSN: 1942-3454. DOI: 10.1145/2633600.

[21] Zvika Brakerski et al. “Classical hardness of learning with errors”. In: *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*. STOC ’13. Palo Alto, California, USA: Association for Computing Machinery, 2013, pp. 575–584. ISBN: 978-1-450-32029-0.

[22] Pedro Branco, Giulio Malavolta, and Zayd Maradni. *Fully-Homomorphic Encryption from Lattice Isomorphism*. Cryptology ePrint Archive, Paper 2025/993. Sept. 2025. URL: https://eprint.iacr.org/2025/993.

[23] Alessandro Budroni, Jesús-Javier Chi-Domínguez, and Ermes Franch. "Don't Use It Twice: Reloaded! On the Lattice Isomorphism Group Action". In: *IACR Communications in Cryptology* 2.2 (July 2025). ISSN: 3006-5496. DOI: 10.62056/ay76chdj.

[24] George A. Campbell and Ronald M. Foster. *Fourier Integrals for Practical Applications*. Princeton, NJ, USA: D. Van Nostrad Company, 1948.

[25] Ran Canetti and Marc Fischlin. "Universally Composable Commitments". In: *Advances in Cryptology — CRYPTO 2001*. Ed. by Joe Kilian. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 19–40. ISBN: 978-3-540-44647-7.

[26] Sueli I. R. Costa et al. "Lattices from Codes". In: *Lattices Applied to Coding for Reliable and Secure Communications*. Cham: Springer International Publishing, 2017, pp. 37–58. ISBN: 978-3-319-67882-5.

[27] Ivan Damgård and Jesper Buus Nielsen. "Improved Non-committing Encryption Schemes Based on a General Complexity Assumption". In: *Advances in Cryptology — CRYPTO 2000*. Ed. by Mihir Bellare. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 432–450. ISBN: 978-3-540-44598-2.

[28] Gustavo de Castro Biage. *LIPb-KEM - QFUtils.sage*. Git version control. Dec. 2023. URL: https://github.com/gustavobiage/LIPb-KEM/blob/master/QFUtils.sage (visited on 09/08/2025).

[29] Gustavo de Castro Biage et al. "A Concrete LIP-Based KEM With Simple Lattices". In: *IEEE Access* 12 (2024), pp. 16408–16420. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3358670.

[30] Danny Dolev, Cynthia Dwork, and Moni Naor. "Non-malleable cryptography". In: *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*. STOC '91. New Orleans, Louisiana, USA: Association for Computing Machinery, 1991, pp. 542–552. ISBN: 0897913973.

[31] Nico Döttling et al. "A CCA2 Secure Variant of the McEliece Cryptosystem". In: *IEEE Transactions on Information Theory* 58.10 (Oct. 2012), pp. 6672–6680. ISSN: 1557-9654. DOI: 10.1109/TIT.2012.2203582.

[32] Léo Ducas and Shane Gibbons. "Hull Attacks on the Lattice Isomorphism Problem". In: *Public-Key Cryptography – PKC 2023*. Ed. by Alexandra Boldyreva and Vladimir Kolesnikov. Cham: Springer Nature Switzerland, 2023, pp. 177–204. ISBN: 978-3-031-31368-4.

[33] Léo Ducas and Wessel van Woerden. "On the Lattice Isomorphism Problem, Quadratic Forms, Remarkable Lattices, and Cryptography". In: *Advances in Cryptology – EUROCRYPT 2022*. Ed. by Orr Dunkelman and Stefan Dziembowski. Cham: Springer International Publishing, 2022, pp. 643–673. ISBN: 978-3-031-07082-2.

[34] Léo Ducas et al. "Hawk: Module LIP Makes Lattice Signatures Fast, Compact and Simple". In: *Advances in Cryptology – ASIACRYPT 2022*. Ed. by Shweta Agrawal and Dongdai Lin. Cham: Springer Nature Switzerland, 2022, pp. 65–94. ISBN: 978-3-031-22972-5.

[35] David Mandell Freeman et al. “More Constructions of Lossy and Correlation-Secure Trapdoor Functions”. In: *Public Key Cryptography – PKC 2010*. Ed. by Phong Q. Nguyen and David Pointcheval. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 279–295. ISBN: 978-3-642-13013-7.

[36] Eiichiro Fujisaki and Tatsuaki Okamoto. “Secure Integration of Asymmetric and Symmetric Encryption Schemes”. In: *Advances in Cryptology — CRYPTO’ 99*. Ed. by Michael Wiener. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 537–554. ISBN: 978-3-540-48405-9.

[37] Sanjam Garg and Mohammad Hajiabadi. “Trapdoor Functions from the Computational Diffie-Hellman Assumption”. In: *Advances in Cryptology – CRYPTO 2018*. Ed. by Hovav Shacham and Alexandra Boldyreva. Cham: Springer International Publishing, 2018, pp. 362–391. ISBN: 978-3-319-96881-0.

[38] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. “Trapdoors for hard lattices and new cryptographic constructions”. In: *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*. STOC ’08. Victoria, British Columbia, Canada: Association for Computing Machinery, 2008, pp. 197–206. ISBN: 978-1-605-58047-0.

[39] 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*. Ed. by Ran Canetti and Juan A. Garay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 75–92. ISBN: 978-3-642-40041-4.

[40] O. Goldreich and L. A. Levin. “A hard-core predicate for all one-way functions”. In: *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*. STOC ’89. Seattle, Washington, USA: Association for Computing Machinery, 1989, pp. 25–32. ISBN: 0-8979-1307-8.

[41] Oded Goldreich, Silvio Micali, and Avi Wigderson. “Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems”. In: *J. ACM* 38.3 (July 1991), pp. 690–728. ISSN: 0004-5411. DOI: 10.1145/116825.116852.

[42] Oded Goldreich and Yair Oren. “Definitions and properties of zero-knowledge proof systems”. In: *Journal of Cryptology* 7.1 (Dec. 1994), pp. 1–32. ISSN: 1432-1378. DOI: 10.1007/BF00195207.

[43] Brett Hemenway and Rafail Ostrovsky. “On Homomorphic Encryption and Chosen-Ciphertext Security”. In: *Public Key Cryptography – PKC 2012*. Ed. by Marc Fischlin, Johannes Buchmann, and Mark Manulis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 52–65. ISBN: 978-3-642-30057-8.

[44] Yuval Ishai et al. “On Invertible Sampling and Adaptive Security”. In: *Advances in Cryptology - ASIACRYPT 2010*. Ed. by Masayuki Abe. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 466–482. ISBN: 978-3-642-17373-8.

[45] Kaijie Jiang et al. “Re-randomize and Extract: A Novel Commitment Construction Framework Based on Group Actions”. In: *Advances in Cryptology – EUROCRYPT 2025*. Ed. by Serge Fehr and Pierre-Alain Fouque. Cham: Springer Nature Switzerland, 2025, pp. 124–153. ISBN: 978-3-031-91124-8.

[46] Jonathan Katz and Ji Sun Shin. “Parallel and Concurrent Security of the HB and HB + Protocols”. In: *Advances in Cryptology - EUROCRYPT 2006*. Ed. by Serge Vaudenay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 73–87. ISBN: 978-3-540-34547-3.

[47] Jonathan Katz and Vinod Vaikuntanathan. “Smooth Projective Hashing and Password-Based Authenticated Key Exchange from Lattices”. In: *Advances in Cryptology – ASIACRYPT 2009*. Ed. by Mitsuru Matsui. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 636–652. ISBN: 978-3-642-10366-7.

[48] Xuan Thanh Khuc et al. “Logarithmic-Size (Linkable) Ring Signatures from Lattice Isomorphism Problems”. In: *Security, Privacy, and Applied Cryptography Engineering*. Ed. by Francesco Regazzoni, Bodhisatwa Mazumdar, and Sri Parameswaran. Cham: Springer Nature Switzerland, 2024, pp. 214–241. ISBN: 978-3-031-51583-5.

[49] Philip Klein. “Finding the closest lattice vector when it’s unusually close”. In: *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA ’00. San Francisco, California, USA: Society for Industrial and Applied Mathematics, 2000, pp. 937–941. ISBN: 0-8987-1453-2.

[50] A. K. Lenstra, H. W. Lenstra, and L. Lovász. “Factoring polynomials with rational coefficients”. In: *Mathematische Annalen* 261.4 (Dec. 1982), pp. 515–534. ISSN: 1432-1807. DOI: 10.1007/BF01457454.

[51] Alberto Leporati, Lorenzo Rovida, and Wessel van Woerden. *Beyond LWE: a Lattice Framework for Homomorphic Encryption*. Cryptology ePrint Archive, Paper 2025/1171. June 2025. URL: https://eprint.iacr.org/2025/1171.

[52] Cong Ling, Jingbo Liu, and Andrew Mendelsohn. “On the Spinor Genus and the Distinguishing Lattice Isomorphism Problem”. In: *Advances in Cryptology – ASIACRYPT 2024*. Ed. by Kai-Min Chung and Yu Sasaki. Singapore: Springer Nature Singapore, 2025, pp. 329–358. ISBN: 978-981-96-0894-2.

[53] Hengyi Luo et al. *Commitment Schemes Based on Module-LIP*. Cryptology ePrint Archive, Paper 2025/431. Mar. 2025. URL: https://eprint.iacr.org/2025/431.

[54] R. J. McEliece. *A Public-Key Cryptosystem Based on Algebraic Coding Theory*. Tech. rep. Deep Space Network Progress Report 42-44, 1978, pp. 114–116.

[55] Francesco Mezzadri. *How to Generate Random Matrices from the Classical Compact Groups*. 2007. arXiv: math-ph/0609050 [math-ph].

[56] Daniele Micciancio. *Lattice Algorithms and Applications: Random Lattices and Lattice-Based Cryptography*. 2019. URL: https://cseweb.ucsd.edu/classes/fa21/cse206A-a (visited on 09/26/2025).

[57] Daniele Micciancio and Antonio Nicolosi. “Efficient bounded distance decoders for Barnes-Wall lattices”. In: *2008 IEEE International Symposium on Information Theory*. Toronto, Canada: IEEE, 2008, pp. 2484–2488. ISBN: 978-1-4244-2256-2.

[58] Daniele Micciancio and Chris Peikert. "Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller". In: *Advances in Cryptology – EUROCRYPT 2012*. Ed. by David Pointcheval and Thomas Johansson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 700–718. ISBN: 978-3-642-29011-4.

[59] Daniele Micciancio and Oded Regev. "Worst-Case to Average-Case Reductions Based on Gaussian Measures". In: *SIAM Journal on Computing* 37.1 (2007), pp. 267–302. ISSN: 1095-7111. DOI: 10.1137/S0097539705447360.

[60] Daniele Micciancio and Michael Walter. "Gaussian Sampling over the Integers: Efficient, Generic, Constant-Time". In: *Advances in Cryptology – CRYPTO 2017*. Ed. by Jonathan Katz and Hovav Shacham. Cham: Springer International Publishing, 2017, pp. 455–485. ISBN: 978-3-319-63715-0.

[61] Daniele Micciancio and Bogdan Warinschi. "A linear space algorithm for computing the hermite normal form". In: *Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation*. ISSAC '01. London, Ontario, Canada: Association for Computing Machinery, 2001, pp. 231–236. ISBN: 1-5811-3417-7.

[62] Lorenz Minder and Amin Shokrollahi. "Cryptanalysis of the Sidelnikov Cryptosystem". In: *Advances in Cryptology - EUROCRYPT 2007*. Ed. by Moni Naor. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 347–360. ISBN: 978-3-540-72540-4.

[63] Rafael Misoczki et al. "MDPC-McEliece: New McEliece variants from Moderate Density Parity-Check codes". In: *2013 IEEE International Symposium on Information Theory*. 2013, pp. 2069–2073. ISBN: 978-1-4799-0446-4.

[64] Arno Mittelbach and Marc Fischlin. "The Hybrid Argument". In: *The Theory of Hash Functions and Random Oracles: An Approach to Modern Cryptography*. Information Security and Cryptography. Springer Cham, 2021. Chap. 3.2.2, pp. 111–124. ISBN: 978-3-030-63287-8.

[65] Arno Mittelbach and Marc Fischlin. "The Random Oracle Controversy". In: *The Theory of Hash Functions and Random Oracles: An Approach to Modern Cryptography*. Information Security and Cryptography. Springer Cham, 2021. Chap. 12, pp. 461–475. ISBN: 978-3-030-63287-8.

[66] Petros Mol and Scott Yilek. "Chosen-Ciphertext Security from Slightly Lossy Trapdoor Functions". In: *Public Key Cryptography – PKC 2010*. Ed. by Phong Q. Nguyen and David Pointcheval. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 296–311. ISBN: 978-3-642-13013-7.

[67] Steven Myers and Abhi Shelat. "Bit Encryption Is Complete". In: *50th Annual IEEE Symposium on Foundations of Computer Science*. 2009, pp. 607–616. ISBN: 978-1-4244-5116-6.

[68] M. Naor and M. Yung. "Public-key cryptosystems provably secure against chosen ciphertext attacks". In: *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*. STOC '90. Baltimore, Maryland, USA: Association for Computing Machinery, 1990, pp. 427–437. ISBN: 0897913612.

[69] Noam Nisan and David Zuckerman. "Randomness is Linear in Space". In: *Journal of Computer and System Sciences* 52.1 (1996), pp. 43–52. ISSN: 0022-0000. DOI: 10.1006/jcss.1996.0004.

[70] Ryo Nojima et al. "Semantic security for the McEliece cryptosystem without random oracles". In: *Designs, Codes and Cryptography* 49.1 (2008), pp. 289–305. ISSN: 1573-7586. DOI: 10.1007/s10623-008-9175-9.

[71] Ayoub Otmani, Jean-Pierre Tillich, and Léonard Dallot. "Cryptanalysis of McEliece cryptosystem based on quasi-cyclic LDPC codes". In: *Proceedings of the First International Conference on Symbolic Computation and Cryptography*. Ed. by Jean-Charles Faugère and Dongming Wang. LMIB Beihang University. Beijing, China, Apr. 2008, pp. 69–81.

[72] R. Overbeck. "Structural Attacks for Public Key Cryptosystems based on Gabidulin Codes". In: *Journal of Cryptology* 21.2 (2008), pp. 280–301. ISSN: 1432-1378. DOI: 10.1007/s00145-007-9003-9.

[73] Chris Peikert. "An Efficient and Parallel Gaussian Sampler for Lattices". In: *Advances in Cryptology – CRYPTO 2010*. Ed. by Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 80–97. ISBN: 978-3-642-14623-7.

[74] Chris Peikert. "Public-key cryptosystems from the worst-case shortest vector problem: extended abstract". In: *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*. STOC '09. Bethesda, MD, USA: Association for Computing Machinery, 2009, pp. 333–342. ISBN: 978-1-605-58506-2.

[75] Chris Peikert and Alon Rosen. "Efficient Collision-Resistant Hashing from Worst-Case Assumptions on Cyclic Lattices". In: *Theory of Cryptography*. Ed. by Shai Halevi and Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 145–166. ISBN: 978-3-540-32732-5.

[76] Chris Peikert and Brent Waters. "Lossy trapdoor functions and their applications". In: *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*. STOC '08. Victoria, British Columbia, Canada: Association for Computing Machinery, 2008, pp. 187–196. ISBN: 978-1-605-58047-0.

[77] Edoardo Persichetti. "On the CCA2 Security of McEliece in the Standard Model". In: *Provable Security*. Ed. by Joonsang Baek, Willy Susilo, and Jongkil Kim. Cham: Springer International Publishing, 2018, pp. 165–181. ISBN: 978-3-030-01446-9.

[78] Oded Regev. *Lattices in Computer Science: CVP Algorithm*. 2009. URL: https://cims.nyu.edu/~regev/teaching/lattices_fall_2004/ln/cvp.pdf (visited on 09/10/2025).

[79] Oded Regev. *Lattices in Computer Science: Introduction*. 2009. URL: https://cims.nyu.edu/~regev/teaching/lattices_fall_2004/ln/introduction.pdf (visited on 09/18/2025).

[80] Oded Regev. *Lattices in Computer Science: LLL Algorithm*. 2009. URL: https://cims.nyu.edu/~regev/teaching/lattices_fall_2004/ln/lll.pdf (visited on 09/18/2025).

[81] Oded Regev. "New lattice-based cryptographic constructions". In: *J. ACM* 51.6 (Nov. 2004), pp. 899–942. ISSN: 0004-5411. DOI: 10.1145/1039488.1039490.

[82] Oded Regev. "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography". In: *J. ACM* 56.6 (Sept. 2009). ISSN: 0004-5411. DOI: 10.1145/1568318.1568324.

[83] Oded Regev. "The Learning with Errors Problem (Invited Survey)". In: *2010 IEEE 25th Annual Conference on Computational Complexity*. 2010, pp. 191–204. ISBN: 978-1-4244-7215-4.

[84] Alon Rosen and Gil Segev. "Chosen-Ciphertext Security via Correlated Products". In: *SIAM Journal on Computing* 39.7 (2010), pp. 3058–3088. ISSN: 1095-7111. DOI: 10.1137/100782929.

[85] Nicolas Sendrier. "Code-Based Cryptography: State of the Art and Perspectives". In: *IEEE Security & Privacy* 15.4 (2017), pp. 44–50. ISSN: 1540-7993. DOI: 10.1109/MSP.2017.3151345.

[86] Nicolas Sendrier. "On the Use of Structured Codes in Code Based Cryptography". In: *Coding Theory and Cryptography III. Contactforum*. Ed. by S. Nikova, B. Preneel, and L. Strome. 2009, pp. 59–68.

[87] P.W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134. ISBN: 0-8186-6580-7.

[88] V. M. Sidelnikov and S. O. Shestakov. "On insecurity of cryptosystems based on generalized Reed-Solomon codes". In: *Discrete Mathematics and Applications* 2.4 (1992), pp. 439–444. ISSN: 0924-9265. DOI: 10.1515/dma.1992.2.4.439.

[89] Damien Stehlé et al. "Efficient Public Key Encryption Based on Ideal Lattices". In: *Advances in Cryptology – ASIACRYPT 2009*. Ed. by Mitsuru Matsui. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 617–635. ISBN: 978-3-642-10366-7.

[90] William Stein et al. *SageMath Version 10.6 Reference Manual: Dense Matrices over the Integer Ring*. URL: https://doc.sagemath.org/html/en/reference/matrices/sage/matrix/matrix_integer_dense.html (visited on 09/15/2025).

[91] Valerie Gauthier Umaña and Gregor Leander. "Practical Key Recovery Attacks on Two McEliece Variants". In: *Proceedings of the Second International Conference on Symbolic Computation and Cryptography*. Ed. by Carlos Cid and Jean-Charles Faugère. Egham, UK, June 2010.

[92] Wessel van Woerden. "Dense and Smooth Lattices in Any Genus". In: *Advances in Cryptology – ASIACRYPT 2024*. Ed. by Kai-Min Chung and Yu Sasaki. Singapore: Springer Nature Singapore, 2025, pp. 386–417. ISBN: 978-981-96-0894-2.

[93] Wessel van Woerden. "Lattice Cryptography: From Cryptanalysis to New Foundations". PhD thesis. Universiteit Leiden, Feb. 2023. URL: https://hdl.handle.net/1887/3564770.

[94] Jiang Zhang et al. "Improved lattice-based CCA2-secure PKE in the standard model". In: *Science China Information Sciences* 63.8 (2020), p. 182101. ISSN: 1869-1919. DOI: 10.1007/s11432-019-9861-3.