National University of Singapore
CS4230/CS5430: Foundations of Modern Cryptography

AY 2022/23, Sem 2
Last edit: July 12, 2023

From Lattice to Learning with Errors to Public-key Cryptosystems

Christoph Schnabl
e1100681@u.nus.edu

Alex Luoyuan Xiong
e0945857@u.nus.edu

1 Introduction

Motivation. The security of cryptographic systems depends on the assumption of the power of adversaries and the hardness of the underlying problem our construction is based on. Since the discovery of Shor's quantum algorithm, many of our public-key cryptosystems that are based on the hardness of solving the Discrete Logarithm Problem (DLP) like ElGamal encryption or the integer factoring problem like RSA and Rabin encryption on classical computers are no longer secure. Luckily, many lattice-based problems are believed to be hard even for quantum computers, thus serving as good candidates for the foundation of post-quantum cryptography.

The groundbreaking result by Ajtai [Ajt96] gave the first *worst-case to average-case reductions* for lattice problems, establishing a connection between provably secure cryptography and the hardness of well-studied computational problems on lattices. In particular, Ajtai showed that the average-case *short integer solution* (SIS) problem and its associated one-way function (OWF) are at least as hard as approximating a variant of the lattice problems in the worst case. Subsequently, Regev [Reg05] introduced the average-case *learning with errors* (LWE) problem and proposed a public-key encryption scheme by showing a quantum reduction from a weaker form of the *Shortest Vector Problem* (SVP) in lattices to an average LWE instance.[1] This connection is proven to persist even with a classical reduction by [Pei09], who also constructed chosen ciphertext-secure (CCA) public-key encryption assuming the worst-case *classical* hardness of approximating the SVP on *general* lattices. We will focus on this work by Peikert in this report.

Worst-case hardness. Notice that most prior hardness assumptions require the problem to be intractable on average (i.e. random instances are hard), which is inherent in any useful cryptography. In contrast, the *worst-case* notion of hardness only requires *some* intractable instances to exist which offers a much stronger security guarantee. In particular, protocols based on the worst-case hardness of lattice problems are infeasible to break unless *all* instances of that lattice problem are easy to solve, which is highly unlikely. Moreover, the average-case hard DLP is for random discrete log tuples for a *specific* choice of a group, whereas our worst-case hard SVP works for *all* general lattices.

2 Preliminary

For a real number $x \in \mathbb{R}$, we denote $\lfloor x \rceil := \lfloor x + 1/2 \rfloor$ as the discretization of x, i.e. the integer closest to x with ties broken upward. We use bold lower-case letters like $\mathbf{x}$ to denote column vectors; bold upper-case letters like $\mathbf{A}$ to denote matrices; $\mathbf{A}^t$ denotes the transpose matrix of $\mathbf{A}$; $\lfloor \mathbf{x} \rceil$ discretize the vector entry-wise.

We use standard asymptotic notation $\mathcal{O}(\cdot), \Omega(\cdot), \Theta(\cdot)$, etc. Furthermore, $\widetilde{O}(\cdot)$ indicates that extra logarithmic factors, e.g. $\widetilde{O}(n) = n \log^c n$ for any constant c represents quasilinear complexity.

[1]This reduction presents a novel quantum algorithm that solves the SVP problem with a search-LWE oracle, extending the short list of useful quantum algorithms we have.

2.1 Lattices

An n-dimensional *lattice* $\mathcal{L}$ is a discrete additive subgroup of $\mathbb{R}^n$. Every (non-trivial) $\mathcal{L}$ is infinite, but always finitely generated as the integer linear combinations of n linearly independent *basis* vectors $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\} \subset \mathbb{R}^n$. The *full-rank* lattice Λ generated by $\mathbf{B}$ is:

$$\Lambda = \mathcal{L}(\mathbf{B}) := \mathbf{B} \cdot \mathbb{N}^n = \left\{ \sum_{i \in [n]} c_i \cdot \mathbf{b}_i : c_i \in \mathbb{N} \right\}$$

We further denote $\left\{ \widetilde{\mathbf{b}}_i \right\}$ the Gram-Schmidt orthogonalized vectors of the ordered basis $\mathbf{B}$.

A commonly used domain is the *fundamental parallelepiped*

$$\mathcal{P}(\mathbf{B}) := \mathbf{B} \cdot \mathbb{T}^n = \left\{ \sum_{i \in [n]} x_i \cdot \mathbf{b}_i : x_i \in [0, 1) \right\}$$

where $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ is an additive group whose group elements are real numbers in $[0, 1)$ and group operation is modulo 1 addition. We map any point to the unique point in the same coset $\mathbb{R}^n/\mathcal{L}$ within the fundamental parallelepiped using $\mathbf{x} \in \mathbb{R}^n \mod \mathbf{B}$.

The *dual* (sometimes called *reciprocal* of a lattice Λ is defined as:

$$\Lambda^* = \mathcal{L}(\mathbf{B}^*) := \{\mathbf{w} \in \mathbb{R}^n : \langle \mathbf{w}, \mathbf{v} \rangle \subseteq \mathbb{Z}, \forall \mathbf{v} \in \Lambda\}$$

By symmetry, it can be seen that $(\Lambda^*)^* = \Lambda$. For full-rank $\mathcal{L}(\mathbf{B})$, the dual basis $\mathbf{B}^* = (\mathbf{B}^{-1})^t$ is the basis for Λ^*.

The *minimal distance* of a lattice Λ is the length of a shortest non-zero lattice vector $\lambda_1(\Lambda) := \min_{\mathbf{v} \in \Lambda \setminus \{\mathbf{0}\}} \|\mathbf{v}\|$ where $\| \cdot \|$ denotes the ℓ_2 (or Euclidean) norm. More generally, the i-th *successive minimum* $\lambda_i(\Lambda)$ is the smallest r such that Λ has i linearly independent vectors of norm at most r (see the rightmost diagram in Fig. 1).

Computational Problems.

We now define some relevant computational problems on lattices (illustrated in Fig. 1):

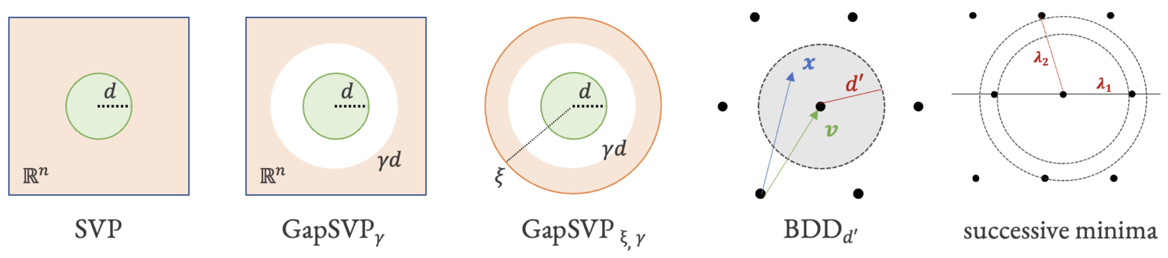


Figure 1: Visualization of SVP variants and BDD. Red regions are NO instances (for $\lambda_1(\Lambda)$ to lie in), green regions are YES instances, and white regions are "gaps" where either decision is fine.

Definition 2.1 (Decisional Shortest Vector Problem (SVP)). Given an arbitrary basis $\mathbf{B}$ of some n-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$ and $d > 0 \in \mathbb{R}$, determine whether $\lambda_1(\Lambda) \leq d$ or $\lambda_1(\Lambda) > d$.

Evidently, SVP is too hard and stringent of a problem to reduce to any (known) average-case problems that are cryptographically useful. The first relaxation is introducing *approximation* problems parameterized by an approximation factor $\gamma = \gamma(n) \geq 1$ that is typically taken to be a function of the lattice dimension. The second relaxation allows for *gap* area within which any decision is acceptable.

Definition 2.2 (Decisional Approximate SVP (GapSVP_γ)). Let $d > 0 \in \mathbb{R}, \gamma(n) \geq 1$. Given an arbitrary basis $\mathbf{B}$ of some n-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$, determine whether $\lambda_1(\Lambda) \leq d$ or $\lambda_1(\Lambda) > \gamma(n) \cdot d$.

Peikert further defines the following generalization of GapSVP_γ with an extra promise on the upper bound of λ_1. Note that for any exponentially large $\zeta(n) \geq 2^{n/2}$, $\mathsf{GapSVP}_{\zeta,\gamma}$ is *equivalent* to the standard GapSVP_γ due to LLL algorithm [LLL82] that can reduce an arbitrary basis $\mathbf{B}'$ to another basis $\mathbf{B}$ so that $\lambda_1(\mathcal{L}(\mathbf{B}') \leq 2^{n/2} \cdot \min_i \|\tilde{\mathbf{b}_i}\|$ in polynomial time. The more interesting condition is when $\zeta(n) = \mathsf{poly}(n)$, as it hinges on an additional promise that the minimal distance lies within a looser range and it's not immediately clear how much easier is $\mathsf{GapSVP}_{\zeta,\gamma}$ compared to GapSVP_γ. Even as a non-standard SVP variant, there's no known algorithm that can solve $\mathsf{GapSVP}_{\zeta,\gamma}$ for $\zeta(n) = \mathsf{poly}(n)$ in time better than $2^{\Omega(()n)}$. This is also what the main result Theorem 3.1 depends on.

Definition 2.3 (ζ-to-γ Approximate SVP ($\mathsf{GapSVP}_{\zeta,\gamma}$))**.** Let $\zeta(n) \geq d \cdot \gamma(n)$, other parameters identically defined as in Definition 2.2. Given an arbitrary basis $\mathbf{B}$ of some n-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$ whose $\lambda_1(\Lambda) \leq \gamma(n)$, determine whether $\lambda_1(\Lambda) \leq d$ or $\lambda_1(\Lambda) > \gamma(n) \cdot d$.

Another important problem seeks to find the lattice vector that is closest to a given target point, known as the *Closest Vector Problem* (CVP). However, similarly to SVP, CVP is too hard that no cryptosystem has yet been proven secure based on it, therefore we employ a relaxation by adding an extra promise that the target point is "somewhat close" to the lattice.

Definition 2.4 (Bounded Distance Decoding Problem (BDD_d))**.** Given an arbitrary basis $\mathbf{B}$ of some n-dimensional lattice $\Lambda = \mathcal{L}(\mathbf{B})$, a promise $d \leq \lambda_1(\Lambda)/2$, and an input vector $\mathbf{x} \in \mathbb{R}^n$, find the unique lattice vector $\mathbf{v} \in \Lambda$ such that $\|\mathbf{x} - \mathbf{v}\| < d$.

2.2 Learning with Errors

The famous [Reg05] introduced the average-case *learning with errors* (LWE) problem. We start with its definition and then describe a few important computational LWE problems. We are primarily concerned with error distributions over $\mathbb{T}$ that is derived from Gaussian. For $\alpha > 0$, define Ψ_α to be the distribution on $\mathbb{T}$ obtained by taking a sample from the one-dimensional Gaussian D_α and reducing modulo 1. We will now give a slightly different distribution for the LWE from what we have seen in the lecture. That, is we will look at the definition that results for a LWE problem.

Definition 2.5. Given a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$, and an error distribution D_α define the LWE distribution $A_{\mathbf{s},D_\alpha}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ as follows:

1. Sample $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ uniformly random and $\mathbf{x} \leftarrow D_\alpha$
2. Output $(\mathbf{a}, \langle \mathbf{a}, \mathbf{x} \rangle + \mathbf{x} \mod q)$

Recall from the lecture that LWE has two main variants SLWE and DLWE, where the search problem refers to finding the secret vector $\mathbf{s}$. The decision problem is two distinguish the LWE distribution from a distribution over the same domain that is uniformly random.
We already know from the lecture that without any perturbations on $\mathbf{s}$, the system of linear equation can efficiently be solved. Let us then look at other hardness results for LWE and in particular how the choice of parameters influences the hardness. First, note that the concrete values of the number of equations m and the group modulus q do not influence the hardness a lot. That is, as long as $q \leq 2\sqrt{n}/\alpha$ and m is chosen large enough that $\mathbf{s}$ can information-theoretically be recovered, but not too large (exponential) such that the problem can easily be solved. Meanwhile we will explain the main theorem how the choices of n, dimension of the field (and by that the dimension of the underlying lattice), and the error rate α matter for hardness.

2.3 Discrete Gaussians

Firstly, we define the open unit ball $\mathcal{B}_n \subset \mathbb{R}^n$ (in the ℓ_2 norm form) as $\mathcal{B}_n = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| < 1\}$. Sampling from the uniform distribution $U(\mathcal{B}_n)$ is efficient.

For any positive integer n and real $r > 0$, define the *Gaussian function* $\rho_r : \mathbb{R}^n \to \mathbb{R}^+$ with parameter (or *width*) r as:

$$\rho_r(\mathbf{x}) := \exp\left(-\pi \cdot \frac{\|\mathbf{x}\|^2}{r^2}\right) = \prod_{i \in [n]} \rho_r(x_i)$$

Notice that ρ_r is invariant under the rotation of $\mathbb{R}^n$.

The (continuous) Gaussian distribution D_r over $\mathbb{R}^n$ is defined to have a probability density function proportional to ρ_r: $D_r(\mathbf{x}) = \rho_r(\mathbf{x})/\int_{\mathbb{R}^n} \rho_r(\mathbf{z})d\mathbf{z} = \rho_r(\mathbf{x})/r^n$. It is possible to sample efficiently from D_r to within any desired level of precision. We can also intuitively interpret r as something close to the standard deviation of the distribution: the larger the r, the higher the deviation, and the closer D_r to the uniform distribution.

Next, we define the *discrete Gaussian* over lattice Λ as $D_{\Lambda,r}(\mathbf{x}) = \rho_r(\mathbf{x})/\rho_r(\Lambda)$ for all $\forall \mathbf{x} \in \Lambda$. Note that the denominator is merely a normalization factor so that the sum of probability for all lattice points adds up to 1, making D a proper distribution, namely $\rho_r(\Lambda) = \sum_{\mathbf{x}\in\Lambda}(\rho_r(\mathbf{x}))$.

Proposition 2.1 ([GPV08] Theorem 4.1)**.** *There exists a p.p.t. algorithm that, given any basis* $\mathbf{B}$ *of an* n*-dimensional lattice* Λ *and any* $r \geq \max_i \|\widetilde{\mathbf{b}}_i\| \cdot \omega(\sqrt{\log n})$*, outputs a sample from a distribution that is within* $\mathsf{negl}(n)$ *statistical distance of* $D_{\Lambda,r}$*.*

Micciancio and Regev [MR04] introduced a very important quantity called the *smoothing parameter* of a lattice $\mathcal{L}$. Informally, this is the amount of Gaussian "blur" required to "smooth out" essentially all the discrete structures of $\mathcal{L}$. Intuitively, a dense lattice corresponds to a smaller smoothing parameter as it requires less noise to look uniformly distributed in $\mathbb{R}^n$. Formally, the smoothing parameter $\eta_\varepsilon(\mathcal{L})$ is parameterized by a tolerance $\varepsilon > 0$, and is defined using the dual lattice as the minimal $r > 0$ such that $\rho_{1/r}(\mathcal{L}^*) \leq \varepsilon$.

Lemma 2.2 ([MR04])**.** *For any full-rank lattice* $\mathcal{L}$*, we have* $\eta_{2^{-n}}(\mathcal{L}) \leq \sqrt{n}/\lambda_1(\mathcal{L}^*)$*.*

3 Main Theorem: Classical Worst-case Reduction

Theorem 3.1 (Main Theorem)**.** *Let the error rate* $\alpha = \alpha(n) \in (0,1)$ *be a real number, and approximation factor* $\gamma = \gamma(n) \geq n/(\alpha\sqrt{\log n})$*. Let the promised range upper-bound* $\zeta = \zeta(n) \geq \gamma$*, and field modulus* $q = q(n) \geq (\zeta/\sqrt{n}) \cdot \omega(\sqrt{\log n}) \geq \omega(\sqrt{n}/\alpha)$*.*

There is a p.p.t. classical reduction from solving $\mathsf{GapSVP}_{\zeta,\gamma}$ *in the worst case with overwhelming probability to solving* $\mathsf{LWE}_{q,\Psi_\alpha}$ *using* $m = \mathsf{poly}(n)$ *samples.*

Notice that Theorem 3.1 allows for a small modulus $q = \mathsf{poly}(n)$, but relies on a relaxed (and less standard) $\mathsf{GapSVP}_{\zeta,\gamma}$ worst-case hardness assumption. Peikert's reduction can be alternatively interpreted as a classical reduction from a more standard GapSVP_γ worst-case hardness assumption but for a larger modulus q, due to the equivalence relationship explained in Section 2.1.

Theorem 3.2 (Alternative Characterization)**.** *For the same parameters as Theorem 3.1, except for an exponentially large modulus* $q \geq 2^{n/2}$*, there exists a p.p.t. classical reduction from* GapSVP_γ *in the worst case to solving* $\mathsf{LWE}_{q,\Psi_\alpha}$ *using* $m = \mathsf{poly}(n)$ *samples.*

3.1 Regev's Reduction

Since Peikert's proof follows the same strategy and reuses many sub-components in that of [Reg05], we give a high-level explanation of Regev's reduction as a preamble. We first rephrase Regev's main theorem here:

Theorem 3.3 ([Reg05])**.** *For any* $m = \mathsf{poly}(n)$*, any modulus* $q \leq 2^{\mathsf{poly}(n)}$*, and any discrete Gaussian error distribution* χ *of parameter* $\alpha q \geq 2\sqrt{n}$ *where the error rate* $\alpha \in (0,1)$*, solving the* $\mathsf{DLWE}_{q,\chi}$ *is at least as hard as* quantumly *solving* GapSVP_γ *and* SIVP_γ *on* arbitrary n*-dimensional lattices, for some approximation factor* $\gamma = \widetilde{O}(n/\alpha)$*.*

Hardness governing parameters. For GapSVP (and its variants), the only parameters that are governing the hardness of the problem are the lattice dimension n and approximation factor γ – since n is already implicit in $\gamma = \gamma(n)$, our complexity class is denoted using GapSVP_γ. Particularly, smaller γ means harder GapSVP_γ problem, thus a more secure worst-case hardness assumption to rely on. For LWE however, the exact values of the number of samples m and the modulus q play no essential role in the ultimate hardness guarantee. The only requirement is that $m \geq n \log q$ so that unique LWE

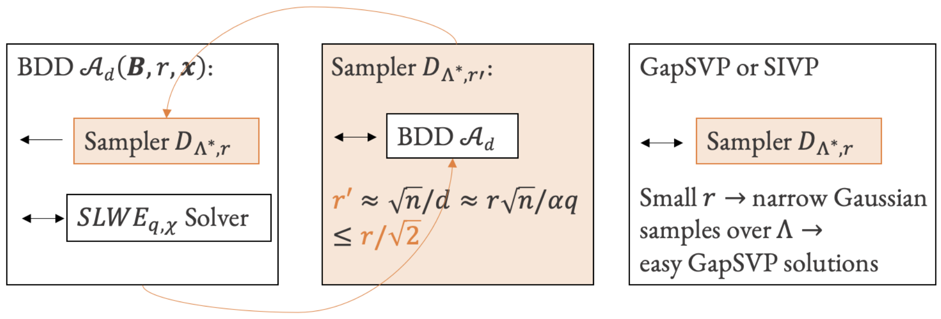

BDD $\mathcal{A}_d(\boldsymbol{B}, r, \boldsymbol{x})$:
Sampler $D_{\Lambda^*,r}$
$SLWE_{q,\chi}$ Solver
Sampler $D_{\Lambda^*,r'}$:
BDD $\mathcal{A}_d$
$r' \approx \sqrt{n}/d \approx r\sqrt{n}/\alpha q \leq r/\sqrt{2}$
GapSVP or SIVP
Sampler $D_{\Lambda^*,r}$
Small $r \rightarrow$ narrow Gaussian samples over $\Lambda \rightarrow$ easy GapSVP solutions

Figure 2: High-level idea of the $\mathsf{GapSVP} \Rightarrow \mathsf{SLWE}$ reduction in [Reg05]. The orange region involves quantum algorithms.

solutions are guaranteed, and we usually use any polynomial number of samples; and that $q \geq 2\sqrt{n}/\alpha$. Instead, the error rate α and dimension n are the governing parameters. Lower error rate α means that our error distribution is lumpy and heavily centered around the span of secret vector $\mathbf{s}$, thus easier the LWE problem; larger dimensions mean harder problems for both LWE and GapSVP. Further notice the connection between the two, the approximation factor γ degrades with the inverse error rate $1/\alpha$, which simply means simpler GapSVP_γ reduces to simpler $\mathsf{DLWE}_{q,\chi}$.

The high-level proof strategy for Theorem 3.3 contains two parts

$$\mathsf{GapSVP} \overset{\text{Sampler } D_{\Lambda^*,r}}{\Longrightarrow} \mathsf{SLWE}_{q,\chi} \overset{q=\mathsf{poly}n}{\equiv} \mathsf{DLWE}_{q,\chi}.$$

The second part is an equivalence between the complexity of SLWE and DLWE (Lemma 4.1 of [Reg05]). Since Peikert directly borrowed this lemma without further innovation, we skip the proof details and take it as a given proposition. We mainly focus on the first part of the reduction, which in itself consists of two main components (illustrated in Fig. 2):

1. A classical BDD_d solver on dual lattice $\mathcal{L}^*$ within distance $d \approx \alpha q/r$ given a $\mathsf{SLWE}_{q,\chi}$ solver and a discrete Gaussian sampler $D_{\Lambda^*,r}$ over the lattice $\mathcal{L}$. This works by combining the BDD instances with Gaussian samples to produce properly distributed LWE samples, whose underlying secret lets us compute the BDD solution.

2. A quantum sampler $D_{\Lambda^*,r'}$ that, given an oracle to a BDD_d solver, generates discrete Gaussian samples over Λ with parameter $r' \approx \sqrt{n}/d$.

The magic of this reduction is the iterative process where the BDD solver and the Gaussian sampler feed back into each other until the step stops working. In particular, notice how the new samples are from a more lumpy distribution with smaller $r' < r/\sqrt{2}$ than the previous iteration. Essentially, we start off with a sampler with large r that effectively is sampling from a close-to-uniform distribution and slowly converges to a better sampler (in the sense that samples drawn are from a lumpier distribution), thus more concentrated around lattice points. With very narrow discrete Gaussian samples over $\mathcal{L}$, we can easily solve SIVP and GapSVP.

3.2 Proof of Main Theorem

Peikert followed the overall strategy from Regev, modified the $\mathsf{GapSVP} \Rightarrow \mathsf{SLWE}$ reduction by replacing the quantum sampler used in the BDD solver with a classical polynomial-time sampler from Proposition 2.1, and inherited the equivalence result between two flavors of LWE problems. Intuitively, the limitation of a classical sampler (compared to Regev's quantum sampler) manifests in a more stringent bound on the width parameter r, which propagates to the extra promise on $\lambda_1 \leq \zeta$, thus the reliance on slightly non-standard $\mathsf{GapSVP}_{\zeta,\gamma}$ hardness. Concretely, we describe the modified (and actually straightforward) reduction below:

Proof of Theorem 3.1. The input to our reduction is an instance of $\mathsf{GapSVP}_{\zeta,\gamma}$ of any lattice Λ, namely a tuple $(\mathbf{B}, d)$. Our reduction runs the following procedure $N = \mathsf{poly}(n)$ times:

1. Choose a point $\mathbf{w}$ uniformly from the unit ball $d' \cdot \mathcal{B}_n$ where $d' = d \cdot \sqrt{n/(4\log n)}$. Compute $\mathbf{x} = \mathbf{w} \bmod \mathbf{B} \in \mathcal{P}(\mathbf{B})$.

2. Run the $\mathsf{BDD}_{d'}(\mathbf{B}, r, \mathbf{x})$ reduction shown in Fig. 2, with $r = \frac{q\sqrt{2n}}{\gamma \cdot d}$. Inside the BDD solver, the sampler $D_{\Lambda^*,r}$ is instantiated using the classical substitude from [GPV08] on the dual lattice Λ^*. The BDD solver outputs a claimed closest lattice vector $\mathbf{v}$.

If $\mathbf{v} \neq \mathbf{x} - \mathbf{w}$ in any of the N iterations, then accpets; otherwise rejects.

To see why this reduction works, we first show that the precondition for our classical GPV08 sampler is met:

$$\begin{aligned} r &= \frac{q \cdot \sqrt{2n}}{\gamma \cdot d} \\ &\geq \frac{q \cdot \sqrt{2n}}{\zeta} \quad \text{(promise from } \mathsf{GapSVP}_{\zeta,\gamma}) \\ &\geq \omega(\sqrt{\log n}) \quad \text{(hypothesis on modulus: } q(n) \geq (\zeta/\sqrt{n}) \cdot \omega(\sqrt{\log n})) \end{aligned}$$

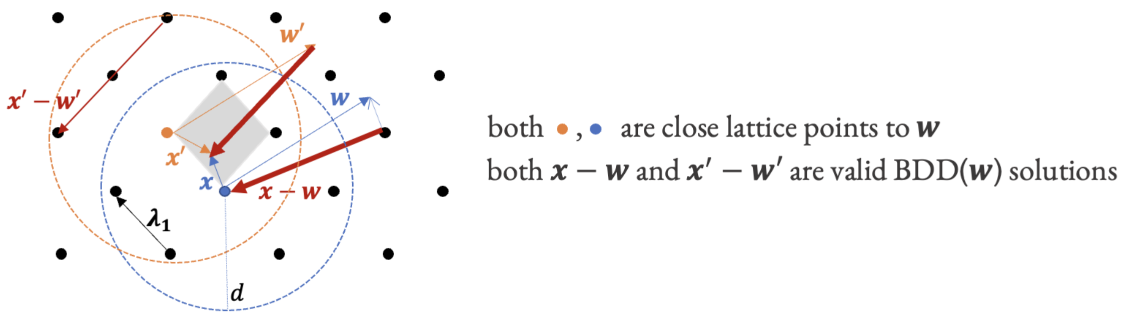
This shows that our BDD solver still works even with replaced sampler.

Next, we show the connection bewteen decision on GapSVP and BDD. When the input $(\mathbf{B}, d)$ is a NO instance, i.e. $\lambda_1(\Lambda) > \gamma \cdot d$, we want to argue that our BDD solver always find the correct solution with overwhelming probability. There are two prerequisites for BDD solver to successfully compute $\mathbf{v}$:

- $r \geq \sqrt{2}q \cdot \eta_{2^{-n}}(\Lambda^*)$: this is required by the original BDD reduction stated as the Lemma 3.4 in [Reg05], and can be derived using Lemma 2.2.
- $\|\mathbf{x}\| \leq \|\mathbf{w}\| = d' \leq \frac{\alpha q}{\sqrt{2}r}$:that $\mathbf{x}$ as the input to BDD indeed lies close enough to the lattices.

Thus, our BDD resolver in this case return the correct $\mathbf{v} = \mathbf{x} - \mathbf{w}$ with overwhelming probability each time, and thus our GapSVP algorithm rejects as desired.

Meanwhile, when the input $(\mathbf{B}, d)$ is a YES instance, i.e. $\lambda_1(\Lambda) \leq d$, we want to argue that our BDD solver will failed at least once. Consider an alternative experiement in which $\mathbf{w}$ is replaced by $\mathbf{w}' = \mathbf{z} + \mathbf{w}$ where $\mathbf{z}$ is the lattice vector that has length $\|\mathbf{z}\| = \lambda_1(\mathbf{\Lambda})$. Intuitively, because λ_1 is so small and lattice points are so close to each other, there could be multiple solutions to "which is the closest lattice vector to $\mathbf{w}$". As a result, the BDD solvers won't be able to guess the correct one every time.[2] More formally, we rely on a result from [GG00] to state the for two n-dimensional balls whose centers are relatively close, the uniform distributions over the balls have statistical distance bounded away from 1 (namely share non-negligible overlaps). Rather than presenting the formal lemma by Goldreich and Goldwasser, it's easier to reason pictorially (see Fig. 3). Therefore, BDD solver will likely to output some $\mathbf{v} \neq \mathbf{x} - \mathbf{w}$ for some iterations and our reduction will accept as desired. □



w'
x' − w'
w
x'
x
x − w
λ₁
d
both ●, ● are close lattice points to w
both x − w and x' − w' are valid BDD(w) solutions

Figure 3: Visualization of [GG00] and its application in the YES case in the Peikert09 reduction.

[2]A similar logic occurs in Rabin Encryption security proof in our lecture.

Comparison with [Reg05].

- Peikert's reduction only works for GapSVP, not SIVP, whereas Regev's works for both.
- The reduction from the standard GapSVP(as stated in Theorem 3.2) requires an exponentially large modulus $q \geq 2^{n/2}$, whereas Regev's works for any modulus $q \geq 2\sqrt{n}/\alpha$. Large q means lower practical efficiency as more bits are needed to represent LWE samples.

4 Public-Key System from LWE

We want to adapt existing CCA-secure cryptosystems to be based on the hardness of GapSVP. This is a stronger assumption to rely on compared to solving $\mathsf{GapSVP}_{\zeta,\gamma}$ classically or SVP in a quantum manner. However, this approach yields an exponential value for the modulus q, which leads to a large plaintext expansion factor (and other problems that are undesirable). To address this issue, Peikert exemplarily adapts a previous cryptosystem based on trapdoors [GPV08] using a clever discretization trick. This results in a more direct and simpler construction that doesn't rely on lossy trapdoor functions (TDFs). In the following section, we will present how to construct injective lattice-based trapdoor function families that are witness-recovering. In contrast to what was covered in the lecture, this means that given a trapdoor, it is possible to also recover a witness for any preimage of the function. We will analyze these constructions using a simpler version of the trapdoor function and then do the same for Peikert's proposed trapdoor construction, which uses the discretization tricks. Lastly, we will introduce the notion of chosen-output security and adapt the previous injective trapdoor family to to a stronger notion of families of preimage sampleable functions (PSF). That is, we get injective trapdoors even if an adversary is tempering with the preimages. We elude how to construct a PKE system from trapdoors, as this was already covered in the lecture. Instead, we will focus on lattice-based trapdoors that can be used to construct CCA-secure cryptosystems.

4.1 CCA-Security from Lattice-Based Trapdoors

In the following notions, many different parameters sampled from different domains and different distributions are involved, so we first give an overview of the setup. First, the below constructions work over four different domains: $\mathbb{R}^n$, $\mathbb{Z}_{q'}^n$, $\mathbb{Z}_q^n$, and $\mathbb{T}^n = [0,1)^n \subset \mathbb{R}^n$. Since there are many different variables, we give an overview of variables and their respective domain: $\in \mathbb{R}^n$, $\overline{\mathbf{b}} \in \mathbb{Z}_{q'}^n$, $\mathbf{s} \in \mathbb{Z}_q^n$, and $\mathbf{x}, \mathbf{b}, \mathbf{b}' \in \mathbb{T}^n$. To understand how the domains relate to each other, let's look at the case for $n = 1$. We can transform $x \in \mathbb{T}$ to $\mathbb{R}$ by $x \cdot q'$, $x \in \mathbb{R}$ to $\mathbb{Z}_{q'}$ by $\lfloor x \rceil \mod q'$, and $x \in \mathbb{Z}_{q'}$ to $\mathbb{T}$ through x/q'. Similarly, we can transform $x \in \mathbb{Z}_q$ to $\mathbb{T}$ by x/q which allows us to map values from the group of larger modulus q to the group of smaller modulus q'.
Finally, we introduce two types of distributions, first the discrete n-dimensional Gaussian distribution with parameter (think standard deviation) D_α^n, and Ψ_α^n over $\mathbb{T}^n$. The latter of the two is obtained by taking samples from the first and reducing them to samples from $\mathbb{T}^n$.

Theorem 4.1. *Let $q \geq 2$, $poly(n) \geq m \geq 2n\log^2 q$. Then there exists a p.p.t. algorithm Gen that samples $(\mathbf{A}, \mathbf{T}) \leftarrow Gen(1^n, q, m)$ with $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{T} \in \mathbb{Z}_q^{m\times m}$ for which the output satisfies:*

1. *$\mathbf{A} \sim_c U$, $\mathbf{A}$ is computationally indistinguishable from a uniformly random sampled matrix*
2. *$||\mathbf{t}_i|| \leq L = 5\sqrt{n\log q}$, every column vector $\mathbf{t}_i \in \mathbf{T}$ is relatively short*
3. *$\mathbf{AT} = 0 \mod q$*

The intuition behind this construction is the following. We use $\mathbf{A}$ as the publicly available evaluation key. $\mathbf{A}$ is interpreted as a parity check matrix that describes some lattice [MR09]. Under this interpretation $\mathbf{T}$ can be viewed as a very short basis for the lattice corresponding to $\mathbf{A}$. Interestingly, each basis described by $\mathbf{T}$ is small enough, however large enough to perturb the parity check matrix such that the product is a multiple of q, which follows from the properties of the parity check matrix. To give a bit more intuition on this, $\mathbf{A}$ defines a set of linear equations that every lattice $\mathbf{v}$ point should fulfill, that is for $\mathbf{v} \in \Lambda(\mathbf{A})$ we have $\mathbf{Av} = 0 \mod q$. That way $\mathbf{T}$ acts as the trapdoor for our construction.
Observe, that $\mathbf{s}$ is determined by $\mathbf{y} = \mathbf{A}^t/q$ and we can efficiently recover $\mathbf{s}$ given $\mathbf{A}$ and $\mathbf{y}$. We now present a family $\{f_\mathbf{A}\}$ of lattice-based trapdoor functions.

Definition 4.1 (Lattice-Based Trapdoor Functions)**.** Denote the family of lattice-based trapdoor functions $\{f_{\mathbf{A}} : \mathbb{Z}_q^n \times \mathbb{T}^m \rightarrow \mathbb{Z}_{q'}^n\}_{q,q',m,\alpha}$ parameterized by moduli q, q' number of $poly(n)$ equations $m \geq 2n \log^2 q$, and an error parameter $\alpha \in (0, 1)$. Note, that *Invert* computes f^{-1} and recovers both $\mathbf{s}, \mathbf{x}$, which is referred to as witness recovering, $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ describes the public verification key and $\mathbf{T} \in \mathbb{Z}_q^{m\times n}$ describes the trapdoor.

- $\underline{KeyGen(1^n, q, m)\textbf{:}}$
  1. Compute $(\mathbf{A}, \mathbf{T}) \leftarrow Gen(1^n, q, m)$ from Theorem 4.1 and output $(\mathbf{A}, \mathbf{T})$
- $\underline{Eval(1^n, \mathbf{A}, \mathbf{s}, \mathbf{x})\textbf{:}}$
  1. Compute $\mathbf{b} = \mathbf{A}^t\mathbf{s}/q + \mathbf{x}$
  2. Output $\overline{\mathbf{b}} = \lfloor \mathbf{b}q \rceil \mod q'$
- $\underline{Invert(1^n, \mathbf{A}, \mathbf{T}, \overline{\mathbf{b}})\textbf{:}}$
  1. Compute $\mathbf{b}' = \overline{\mathbf{b}}/q'$ with $\mathbf{b}' \in \mathbf{T}^m$ and $\mathbf{y} = \mathbf{T}^{-t}\lfloor \mathbf{T}^t b' \rceil \mod 1$
  2. Recover $\mathbf{s}'$ using Gaussian elimination from T
  3. Compute $\mathbf{x}' = \mathbf{b}' - (\mathbf{A}^t\mathbf{s})/q$ with $\mathbf{x}' \in \mathbf{T}^m$ and output $(\mathbf{s}', \mathbf{x}')$

We will later use the construction of [GPV08] to analyze the properties of 4.1. The details, that are different are highlighted below:

- $KeyGen(1^n, q, m)$ is analogous to our construction in Definition 4.1.
- $Eval(1^n, \mathbf{A}, \mathbf{s}, \mathbf{x})$ just outputs $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{x}$
- $Invert(1^n, \mathbf{A}, \mathbf{T}, \overline{\mathbf{b}})$ works by first computing $\mathbf{y} = \mathbf{T}\mathbf{b} = \mathbf{T}\mathbf{x} \mod q$ where equality follows from expanding $\mathbf{b}$ and our construction of $\mathbf{A}\mathbf{T} = 0 \mod q$. Note, that we eluded the transformation from modulus q to the real-valued world that is needed to recover $\mathbf{x} = \mathbf{T}^t\mathbf{y}^-1$. We can now use $\mathbf{x}$ to output $\mathbf{s} = \mathbf{A}^{-t}(\mathbf{b} - \mathbf{x})$

We are now going to analyze that the construction of this trapdoor family satisfies the following three properties that we require from trapdoor functions as defined in the lecture: Sampleability, One-wayness, and Invertibility:

- **Sampleability**: Sampleability follows directly from Theorem 4.1.
- **One-wayness**: Recall that for the [GPV08] construction, inverting $f_{\mathbf{A}} = \mathbf{A}^t\mathbf{s} + \mathbf{x}$ is identical to solving search-$\mathsf{LWE}_{q,\Psi_\alpha}$ which we assume to be hard. A similar argument can be made for the construction in definition 4.1.
- **Invertibility:** We will now look at invertibility for the trapdoor family presented in 4.1. Note, that the added steps of changing worlds through discretizing need us to use a somewhat intricate probabilistic argument in the proof. For that, we will first introduce two helpful bounds on column vectors of our trapdoor and the error term x. Recall, from linear algebra that these bounds on the inner product help us two constrain the length of the projection of these previous vectors given another relatively small vector $\mathbf{w}$. We will use this fact, later on, to show that using an efficiency-increasing coarse discretization over q' is indeed fine enough to still be uniform.

Lemma 4.2. *Suppose $q'(n) \geq 2L\sqrt(m)$, $\mathbf{T}$ from Theorem 4.1 and $\mathbf{w} \in \mathbb{R}^n$ such that $|\mathbf{w}_i| \leq$. Then for all $\mathbf{t}_i \in \mathbf{T}$: $|\langle \mathbf{t}_i, \mathbf{w}\rangle| \leq \frac{1}{4}$*

Proof.

$$|\langle \mathbf{t}_i, \mathbf{w}\rangle| \overset{(a)}{\leq} ||\mathbf{t}_i|| \cdot ||\mathbf{w}|| \overset{(b)}{\leq} L \cdot \frac{\sqrt(m)}{2q'} \overset{(c)}{\leq} \frac{1}{4}$$

We know *(a)* by the Cauchy Schwarz Inequality, *(b)* since each column vector in $\mathbf{T}$ is smaller then L by construction and $\mathbf{w} \leq \sqrt(m)/(2q')$ by assumption. Finally, *(c)* intuitively follows from the fact that L is very small, and q' is relatively larger than $\sqrt(m)$ by assumption. In particular, it follows since we required $q' \geq 2L\sqrt(m)$.

□

Lemma 4.3. *Suppose $\alpha \in [0,1)$ such that $1/\alpha \leq L \cdot \omega\sqrt{\log n}$, with $L, \mathbf{T}$ from Theorem 4.1 and $\mathbf{x}' \leftarrow D_\alpha^m$ such that $|\mathbf{w}_i| \leq$. Then for all $\mathbf{t}_i \in \mathbf{T}$: $Pr[|\langle \mathbf{t}_i, \mathbf{x}' \rangle| \leq \frac{1}{4}] = 1 - negl(n)$*

Proof. Observe that for any $\mathbf{t}_i \langle \mathbf{t}_i, \mathbf{x} \rangle \sim D_r$ where $r = ||\mathbf{t}_i||$. This is because for the inner product we just scale every value $\mathbf{x}_i \leftarrow D_\alpha$ by some $\mathbf{t}_i$. This implies that the resulting distribution is more concentrated. Then using the Gaussian tail bound we can conclude that $Pr[|\langle \mathbf{t}_i, \mathbf{x}' \rangle| \leq \frac{1}{4}] = 1 - negl(n)$.

□

Theorem 4.4 (Invertibility)**.** *Suppose $q' = q'(n) \geq 2L\sqrt{m}$ and $\alpha \in [0,1)$ such that $1/\alpha \leq L \cdot \omega\sqrt{\log n}$. Then $Invert(1^n, \mathbf{A}, \mathbf{T}, \overline{\mathbf{b}})$ outputs the correct output $(\mathbf{s}', \mathbf{x}')$ for $\overline{\mathbf{b}} = f_{\mathbf{A}}(\mathbf{s}, \mathbf{x})$ for any $\mathbf{s} \in \mathbb{Z}_q^n$, $\mathbf{x} \leftarrow \Psi_\alpha^m$,*

Proof. First, we know that there exists $\mathbf{w} \in \mathbb{R}^m$ with $|\mathbf{w}_i| \leq \frac{1}{2q'}$ for all $\mathbf{w}_i$, and $\mathbf{x}' \leftarrow D_\alpha^m$ such that

$$\overline{\mathbf{b}} = (\mathbf{A}^t \mathbf{s})/q + \mathbf{x}' + \mathbf{w} \mod 1$$

Observe that for $\mathbf{x}' \leftarrow D_\alpha^m$ $\mathbf{x}$ and $\mathbf{x}'$ are virtually drawn from the same distribution by the definition of Ψ_α^m. Then, all we state above is that for a $\mathbf{x}'$ that is somewhat similar to $\mathbf{x}$, we can find a relatively small $\mathbf{w}$ such that $\mathbf{x}' + \mathbf{w} = \mathbf{x}$. Note, that this looks similar to the argument we made for the [GPV08] scheme. The usage of $\mathbf{w}$ is induced by the discretizitation trick and the probabilistic nature of the proof. Now, by multiplying with $\mathbf{T}^t$ on both sides of the equation we will get the following:

$$\mathbf{T}^t \cdot \overline{\mathbf{b}} = (\mathbf{AT}/q)^t \cdot \mathbf{s} + \mathbf{T}^t \cdot (\mathbf{x}' + \mathbf{w}) \mod \mathbf{T}^t$$

Finally, we want to discretize both sides to show that we can recover s'. We know $\mathbf{AT} = \mathbf{0} \mod q$ from the construction of our trapdoor and by that $\mathbf{AT}/q = \mathbf{0} \mod 1$, which implies that $\mathbf{AT}/q$ is already discrete. Similarly, $\mathbf{T}^t = 0 \mod 1$ since all values from $\mathbf{T}$ are from $Z_q^{m \times m}$. Then:

$$\lfloor \mathbf{T}^t \cdot \overline{\mathbf{b}} \rceil = (\mathbf{AT}/q)^t \cdot \mathbf{s} + \lfloor \mathbf{T}^t \cdot (\mathbf{x}' + \mathbf{w}) \rceil = \mathbf{T}^t(\mathbf{A}^t \mathbf{s}/q) \mod \mathbf{T}^t$$

where *(a)* equality follows from the facts established above. The equality *(b)* is implied by the bounds established in Lemma 4.2 and Lemma 4.3, since we know that $\mathbf{x}' + \mathbf{w}$ is shorter than $1/2$ and thus will be rounded down during discretization and multpliying with a discrete $\mathbf{T}^t$ does not change this fact. □

4.2 Chosen-Output Security

In our previous construction, we assumed that our trapdoor function family would be injective for specific input distributions. However, we now consider the scenario where an adversary has the additional power to choose $\overline{\mathbf{b}}$ arbitrarily. To account for this, we can adapt our trapdoors to remain injective even when faced with this new adversary capable of tampering with the preimage.

Definition 4.2 (Preimage Sampleable Function (PSF))**.** Let f_A be a trapdoor with $f_A : X \to Y$ and any input distribution D over X. We say f_A is a Preimage Sampleable Function (PSF) if there exists a p.p.t. preimage verifying algorithm $V_{\alpha,t}(1^n, \mathbf{A}, (\mathbf{s}, \mathbf{x}), \overline{\mathbf{b}})$ that satisfies the following properties:

- *Completeness:* For $x \leftarrow D$, $\overline{\mathbf{b}} = f_A(x)$, and $x' = Invert(1^n, \mathbf{A}, \mathbf{T}, \overline{\mathbf{b}})$:

$$Pr[V_{\alpha,t}(1^n, \mathbf{A}, x', \overline{\mathbf{b}}) \text{ accepts }] = 1 - negl(n)$$

- *Unique Preimage:* For all y, V accepts for only one legal preimage x
- *Findable Preimage:* For all y, if there exists a preimage x, V correctly accepts it

To understand this definition better, note that this definition implies the following. Given a trapdoor function sampling $x \leftarrow D$ and computing $y = Eval(x)$ is indistinguishable from uniformly sampling $y \leftarrow Y$ and computing $x = V(y)$.

Definition 4.3 (Preimage Sampleable Lattice-Based Trapdoor Functions)**.** Let $|x| = min\{x, 1-x\}$ for $x \in \mathbb{T}$, $t = t(n) = \omega(\sqrt(n))$. Then define V as follows:
$\underline{V_{\alpha,t}(1^n, \mathbf{A}, (\mathbf{s}, \mathbf{x}), \overline{\mathbf{b}})}$**:**

1. Compute $\mathbf{b}' = \overline{\mathbf{b}}/q'$
2. Accept if for all $\mathbf{x}_i : |\mathbf{x}_i| < \alpha \cdot t$ and $\mathbf{b}' = (\mathbf{A}^t\mathbf{s})/q + \mathbf{x}$

Given the output of the inversion procedure and $(\mathbf{s}, \mathbf{x})$, that is the preimage and $\overline{\mathbf{b}}$) we want to accept as follows: we have a valid preimage if all entries of $\mathbf{x}$ are small and if $\overline{\mathbf{b}}$) is correct with regards to the definition of the evaluation function.

Proof. Completeness: The first acceptance property that every entry of $\mathbf{x}$ is reasonably small follows from the Gaussian tail bound. The second acceptance property is satisfied since we know that our invert function outputs the correct preimage by our construction in Theorem 4.4.
Unique Preimage: For every $\mathbf{s} \neq \mathbf{0}$ we know that there exists at least one entry of $(\mathbf{A}\mathbf{s})/q$ larger than $\frac{1}{4}$. newline *Findable Preimage:* We know that for any preimage $(\mathbf{s}, \mathbf{x})$ of $\overline{\mathbf{b}}$ $\mathbf{x}$ is relatively small. Then from the proof of

$\square$

Note that, for this construction the error rate α is smaller by a factor of $O(\sqrt{m})$ than before due to the findable preimage property. This results in a worst-case approximation factor of $\tilde{O}(n \cdot L\sqrt{m})$. In 2012, Micciancio and Peikert proposed a more direct construction of an IND-CCA-secure PKE scheme based on LWE [MP12].

5 Open Problems

Compared to Regev's quantum reduction from GapSVP to LWE, Peikert's classical reduction has some constraints. First, it is non-adaptive, which means that it uses the LWE oracle in a fixed manner for all $N = poly(n)$ runs. To recall from the reduction, the parameters invoked onto the reduction of Regev and by that, the parameter on the LWE oracle are not adapted either. Second, it works only for decisional variants of GapSVP, and not SIVP. This is connected to the previous caveat as Regev uses an adaptive approach to approximate (and find) the shortest vector. Third, Peikert's reduction is based on a special case $\mathsf{GapSVP}_{\zeta,\gamma}$ problem for small modulus q, where for large modulus these two problems are equivalent: $\mathsf{GapSVP}_{\zeta,\gamma} \equiv \mathsf{GapSVP}$. Note, that as mentioned before we ideally would want to build cryptosystems based on SVP over GapSVP over $\mathsf{GapSVP}_{\zeta,\gamma}$.
In fact, stating such a classical reduction of equal to Regev's quantum reduction remains an open question to this day. Any solutions in this direction would represent a major contribution to Lattice-based cryptography. While constructing the PKE system based on new hardness results, Peikert utilized the equivalence of decisional and search LWE for composite modulus q. Subsequent research has demonstrated that this equivalence holds for practically any value of q.

References

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *Symposium on the Theory of Computing*, 1996.

[GG00] Oded Goldreich and Shafi Goldwasser. On the limits of nonapproximability of lattice problems. *J. Comput. Syst. Sci.*, 60:540–563, 2000.

[GPV08] 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*, pages 197–206, 2008.

[LLL82] Arjen K. Lenstra, Hendrik W. Lenstra, and László Miklós Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *Eurocrypt*, volume 7237, pages 700–718. Springer, 2012.

[MR04] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 372–381, 2004.

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

[Pei09] 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, page 333–342, New York, NY, USA, 2009. Association for Computing Machinery.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Symposium on the Theory of Computing*, 2005.