On Lattices, Learning with Errors, Random Linear Codes, and Cryptography

Oded Regev [*]
Department of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel

ABSTRACT

Our main result is a reduction from worst-case lattice problems such as SVP and SIVP to a certain learning problem. This learning problem is a natural extension of the 'learning from parity with error' problem to higher moduli. It can also be viewed as the problem of decoding from a random linear code. This, we believe, gives a strong indication that these problems are hard. Our reduction, however, is quantum. Hence, an efficient solution to the learning problem implies a *quantum* algorithm for SVP and SIVP. A main open question is whether this reduction can be made classical.

Using the main result, we obtain a public-key cryptosystem whose hardness is based on the worst-case quantum hardness of SVP and SIVP. Previous lattice-based public-key cryptosystems such as the one by Ajtai and Dwork were only based on unique-SVP, a special case of SVP. The new cryptosystem is much more efficient than previous cryptosystems: the public key is of size $\tilde{O}(n^2)$ and encrypting a message increases its size by $\tilde{O}(n)$ (in previous cryptosystems these values are $\tilde{O}(n^4)$ and $\tilde{O}(n^2)$, respectively). In fact, under the assumption that all parties share a random bit string of length $\tilde{O}(n^2)$, the size of the public key can be reduced to $\tilde{O}(n)$.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity; E.3 [**Data Encryption**]: Public key cryptosystems

General Terms

Theory, Algorithms, Security

Keywords

lattices, computational learning theory, cryptography, public key encryption, quantum computing, statistical queries

1. INTRODUCTION

Main theorem. Let n be some integer and let $\varepsilon \geq 0$ be some real. Consider the 'learning from parity with error' problem, defined as follows: find $\mathbf{s} \in \mathbb{Z}_2^n$ given a list of 'equations with errors'

$$\langle \mathbf{s}, \mathbf{a}_1 \rangle \approx_\varepsilon b_1 \pmod 2$$
$$\langle \mathbf{s}, \mathbf{a}_2 \rangle \approx_\varepsilon b_2 \pmod 2$$
$$\vdots$$

where the $\mathbf{a}_i$'s are chosen independently from the uniform distribution on $\mathbb{Z}_2^n$ and $\langle \mathbf{s}, \mathbf{a}_i \rangle = \sum_j s_j (a_i)_j$ is the inner product modulo 2 of $\mathbf{s}$ and $\mathbf{a}_i$. The input to the problem consists of the pairs $(\mathbf{a}_i, b_i)$ and the output is a guess for $\mathbf{s}$. By the $\approx_\varepsilon$ symbol we mean that each equation is independently chosen to be correct with probability $1-\varepsilon$ and incorrect with probability ε. Notice that the case $\varepsilon = 0$ can be solved efficiently by, say, Gaussian elimination. This requires $O(n)$ equations and poly(n) time.

The problem seems to become significantly harder when we take any positive $\varepsilon > 0$. For example, let us consider again the Gaussian elimination process and assume we are interested in recovering only the first bit of $\mathbf{s}$. Using Gaussian elimination, we can find a set S of $O(n)$ equations such that $\sum_S \mathbf{a}_i$ is $(1, 0, \ldots, 0)$. Summing the corresponding values b_i gives us a guess for the first bit of $\mathbf{s}$. However, a standard calculation shows that this guess is correct with probability $\frac{1}{2} + 2^{-\Theta(n)}$. Hence, in order to obtain the first bit with good confidence, we have to repeat the whole procedure $2^{\Theta(n)}$ times. This yields an algorithm that uses $2^{O(n)}$ equations and $2^{O(n)}$ time. In fact, it can be shown that given only $O(n)$ equations, the $\mathbf{s}' \in \mathbb{Z}_2^n$ that maximizes the number of satisfied equations is with high probability $\mathbf{s}$. This yields a simple maximum likelihood algorithm that requires only $O(n)$ equations and runs in time $2^{O(n)}$.

Blum, Kalai, and Wasserman [8] provided the first subexponential algorithm for this problem. Their algorithm requires only $2^{O(n/\log n)}$ equations/time and is currently the best known algorithm for the problem. It is based on a clever idea that allows to find a small set S of equations (say, $O(\sqrt{n})$) among $2^{O(n/\log n)}$ equations, such that $\sum_S \mathbf{a}_i$ is, say, $(1, 0, \ldots, 0)$. This gives us a guess for the first bit of $\mathbf{s}$ that is correct with probability $\frac{1}{2} + 2^{-\Theta(\sqrt{n})}$. We can

[*] Supported by an Alon Fellowship, by the Israel Science Foundation, and by the Army Research Office grant DAAD19-03-1-0082.

STOC'05, May 22-24, 2005, Baltimore, Maryland, USA.

obtain the correct value with high probability by repeating the whole procedure only $2^{O(\sqrt{n})}$ times. Their algorithm was later shown to have other important applications, such as the first $2^{O(n)}$-time algorithm for solving the shortest vector problem in a lattice [12, 5].

An important open question is to explain the apparent difficulty in finding efficient algorithms for this learning problem. Our main theorem explains this difficulty for a natural extension of this problem to higher moduli, defined next.

Let $p = p(n) \leq \text{poly}(n)$ be some prime integer and consider a list of 'equations with error'

$$\langle \mathbf{s}, \mathbf{a}_1 \rangle \approx_\chi b_1 \pmod{p}$$
$$\langle \mathbf{s}, \mathbf{a}_2 \rangle \approx_\chi b_2 \pmod{p}$$
$$\vdots$$

where this time $\mathbf{s} \in \mathbb{Z}_p^n$, $\mathbf{a}_i$ are chosen independently and uniformly from $\mathbb{Z}_p^n$, and $b_i \in \mathbb{Z}_p$. The error in the equations is now specified by a probability distribution $\chi : \mathbb{Z}_p \to \mathbb{R}^+$ on $\mathbb{Z}_p$. Namely, for each equation i, $b_i = \langle \mathbf{s}, \mathbf{a}_i \rangle + e_i$ where each $e_i \in \mathbb{Z}_p$ is chosen independently according to χ. We denote the problem of recovering $\mathbf{s}$ from such equations by $\mathsf{LWE}_{p,\chi}$ (learning with error). For example, the learning from parity problem with error ε is the special case where $p = 2$, $\chi(0) = 1 - \varepsilon$, and $\chi(1) = \varepsilon$. Under a reasonable assumption on χ (namely, that $\chi(0) > 1/p + 1/\text{poly}(n)$), the maximum likelihood algorithm described above solves $\mathsf{LWE}_{p,\chi}$ for $p \leq \text{poly}(n)$ using $\text{poly}(n)$ equations and $2^{O(n \log n)}$ time. Under a similar assumption, an algorithm resembling the one by Blum et al. [8] requires only $2^{O(n)}$ equations/time. This is the best known algorithm for the LWE problem.

Our main theorem shows that for certain choices of p and χ, a solution to $\mathsf{LWE}_{p,\chi}$ implies a quantum solution to worst-case lattice problems.

Theorem 1.1 (Informal). *Let n, p be integers and $\alpha \in (0, 1)$ be some real such that $\alpha p > 2\sqrt{n}$. If there exists a polynomial time algorithm that solves $\mathsf{LWE}_{p,\bar{\Psi}_\alpha}$ then there exists a quantum algorithm that approximates the shortest vector problem (SVP) and the shortest independent vectors problem (SIVP) to within $\tilde{O}(n/\alpha)$ in the worst case.*

The exact definition of $\bar{\Psi}_\alpha$ will be given later. For now, it is enough to know that it is a distribution on $\mathbb{Z}_p$ that has the shape of a discrete Gaussian centered around 0 with standard deviation αp, as in Figure 1. Also, the probability of 0 (i.e., no error) is roughly $1/(\alpha p)$. A possible setting for the parameters is $p = O(n^2)$ and $\alpha = 1/(\sqrt{n} \log n)$ (in fact, these are the parameters that we use in our cryptographic application).

The SVP and SIVP are two of the main computational problems on lattices. The best known polynomial time algorithms yield only mildly subexponential approximation factors. It is conjectured that there is no classical polynomial time algorithm that approximates them to within any polynomial factor. Lattice-based constructions of one-way functions, such as the one by Ajtai [2], are based on this conjecture.

One might guess that the same conjecture holds in the quantum world, i.e., there is no quantum polynomial time algorithm that approximates SVP (or SIVP) to within any polynomial factor. Thus one can interpret the main theorem as saying that based on this conjecture, the LWE problem is hard. The only evidence supporting this conjecture is that there are no quantum algorithms for lattice problems that are known to outperform classical algorithms, even though this is probably one of the most important open questions in the field of quantum computing. We do not know, however, if this conjecture is true.

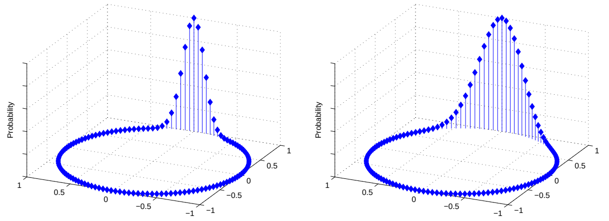

Figure 1: $\bar{\Psi}_\alpha$ for $p = 127$ with $\alpha = 0.05$ (left) and $\alpha = 0.1$ (right). The elements of $\mathbb{Z}_p$ are arranged on the circle.

In fact, one could also interpret our main theorem as a way to disprove this conjecture: if one finds an efficient algorithm for LWE, then one also obtains a quantum algorithm for approximating worst-case lattice problems. Such a result would be of tremendous importance on its own. Finally, we would like to stress that it is possible that our result can be made classical. This would make all our results stronger and the above discussion unnecessary.

The LWE problem can be equivalently presented as the problem of decoding random linear codes. More specifically, let $m = \text{poly}(n)$ be arbitrary and let $\mathbf{s} \in \mathbb{Z}_p^n$ be some vector. Then, consider the following problem: given a random matrix $Q \in \mathbb{Z}_p^{m \times n}$ and the vector $\mathbf{t} = Q\mathbf{s} + \mathbf{e} \in \mathbb{Z}_p^m$ where each coordinate of the error vector $\mathbf{e} \in \mathbb{Z}_p^m$ is chosen independently from $\bar{\Psi}_\alpha$, recover $\mathbf{s}$. The Hamming weight of $\mathbf{e}$ is roughly $m(1 - 1/(\alpha p))$ (since a value chosen from $\bar{\Psi}_\alpha$ is 0 with probability roughly $1/(\alpha p)$). Hence, the Hamming distance of $\mathbf{t}$ from $Q\mathbf{s}$ is roughly $m(1 - 1/(\alpha p))$. Moreover, it can be seen that for large enough m, for any other word $\mathbf{s}'$, the Hamming distance of $\mathbf{t}$ from $Q\mathbf{s}'$ is roughly $m(1 - 1/p)$. Hence, we obtain that approximating the nearest codeword problem to within factors smaller than $(1-1/p)/(1-1/(\alpha p))$ on random codes is as hard as quantumly approximating worst-case lattice problems. This gives a partial answer to the important open question of understanding the hardness of decoding from random linear codes.

It turns out that certain problems, which are seemingly easier than the LWE problem, are in fact equivalent to the LWE problem. We establish these equivalences using elementary reductions. For example, being able to distinguish a set of equations as above from a set of equations in which the b_i's are chosen uniformly from $\mathbb{Z}_p$ is equivalent to solving LWE. Moreover, it is enough to correctly distinguish these two distributions for some non-negligible fraction of all $\mathbf{s}$. The latter formulation is the one we use in our cryptographic applications.

Cryptosystem. Using our main theorem, we obtain a public key cryptosystem whose security is based on the worst-cast quantum hardness of approximating SIVP and SVP to within $\tilde{O}(n^{1.5})$. In other words, breaking our cryptosystem implies an efficient quantum algorithm for approximating SIVP and SVP to within $\tilde{O}(n^{1.5})$. We should emphasize that the cryptosystem is completely classical; it is only in the proof of its security that we use quantum computation. The cryptosystem itself is quite simple; the reader is encouraged to glimpse at the beginning of Section 4.

Previous cryptosystems, such as the Ajtai-Dwork cryptosystem [4] and the one by Regev [17], were only based on the worst-case hardness of the unique-SVP problem, a special case of SVP whose hardness is not so well understood. Basing a cryptosystem on the worst-case hardness of SVP and SIVP is an important open problem. Our cryptosystem does not quite solve this open problem because of its quantum aspects.

Another important feature of our cryptosystem is its improved efficiency. In previous cryptosystems, the public key size is $\tilde{O}(n^4)$ and the encryption increases the size of messages by a factor of $\tilde{O}(n^2)$. In our cryptosystem, the public key size is only $\tilde{O}(n^2)$ and encryption increases the size of messages by a factor of only $\tilde{O}(n)$. This possibly makes our cryptosystem practical. Moreover, using an idea of Ajtai [3], we can reduce the size of the public key to $\tilde{O}(n)$. This requires all users of the cryptosystem to share some random bit string of length $\tilde{O}(n^2)$. This can be achieved by, say, distributing such a bit string as part of the encryption and decryption software.

Finally, we mention that learning problems related to ours were already suggested as possible sources of cryptographic hardness, e.g., [7, 6], although this was done without establishing any connection to lattice problems.

Why quantum? This paper is almost entirely classical. In fact, quantum is needed only in one step in the proof of the main theorem. Making this step classical would make the entire reduction classical. To demonstrate the difficulty, consider the following situation. Let L be some lattice and let $d = \lambda_1(L)/n^{10}$ where $\lambda_1(L)$ is the length of the shortest nonzero vector in L. We are given an oracle that for any point $\mathbf{x} \in \mathbb{R}^n$ within distance d of L finds the closest lattice vector to $\mathbf{x}$. If $\mathbf{x}$ is not within distance d of L, the output of the oracle is undefined. Intuitively, such an oracle seems quite powerful; the best known algorithms for performing such a task require exponential time. Nevertheless, we do not see any way to use this oracle classically. Indeed, it seems to us that the only way to generate inputs to the oracle is the following: somehow choose a lattice point $\mathbf{y} \in L$ and let $\mathbf{x} = \mathbf{y} + \mathbf{z}$ for some perturbation vector $\mathbf{z}$ of length at most d. Clearly, on input $\mathbf{x}$ the oracle outputs $\mathbf{y}$. But this is useless since we already know $\mathbf{y}$!

It turns out that quantumly, such an oracle is quite useful. Indeed, being able to compute $\mathbf{y}$ from $\mathbf{x}$ allows us to *uncompute* $\mathbf{y}$. More precisely, it allows us to transform the quantum state $|\mathbf{x}, \mathbf{y}\rangle$ to the state $|\mathbf{x}, 0\rangle$ in a reversible (i.e., unitary) way. This ability to erase the contents of a memory cell in a reversible way seems useful only in the quantum setting.

Techniques. Our cryptosystem and its analysis are similar to those in the Ajtai-Dwork cryptosystem [4] and the one by Regev [17] (in fact, our analysis is somewhat simpler technically). We believe that the novel part of the paper is the main theorem and in particular, its proof. We now describe some of the new techniques in our proof.

All lattice-based constructions of one-way functions [2, 9, 13, 14, 16] use a construction known as an *iterative reduction*. Essentially, this means that instead of 'immediately' finding very short vectors in a lattice, the reduction proceeds in steps where in each step shorter lattice vectors are found. Typically, in each step the length of the vectors found decreases by some factor, say 2, so that after a polynomial number of steps we end up with vectors that are within some polynomial factor of the shortest. Iterative reductions, however, were so far never used in the construction of lattice-based public-key cryptosystems. Our main theorem is the first to apply iterative reductions in this context. Moreover, we believe that the use of iterative reductions is what allows us to relate the LWE problem to lattices, and in particular to problems like SVP and SIVP (as opposed to unique-SVP).

Our proof is based on the Fourier transform of Gaussian measures, a technique that was developed in previous papers [17, 16, 1]. More specifically, we use a parameter known as the smoothing parameter, as introduced in [16]. We also use the discrete Gaussian distribution and approximations to its Fourier transform, ideas that were developed in [1].

Open questions. The main open question raised by this work is whether our results can be dequantized, leading to stronger results based on the classical hardness of SIVP and SVP. We see no reason why this should be impossible. However, despite our efforts, we were not able to show this. As mentioned above, the difficulty is that there seems to be no classical way to use an oracle that solves the closest vector problem within small distances. Quantumly, however, such an oracle turns out to be quite useful.

Another important open question is to determine the hardness of the learning from parity with errors problem (i.e., the case $p = 2$). Our theorem only works for $p > 2\sqrt{n}$. It seems that in order to prove similar results for smaller values of p, substantially new ideas are required. Alternatively, one might interpret our inability to prove hardness for small p as an indication that substantially better algorithms should exist for this case. Currently, the best known algorithms do not perform significantly better for constant p.

Finally, let us mention another promising direction for future research. In a recent work of Feige [11], a conjecture was presented regarding the hardness of certain random instances of 3SAT. Based on this conjecture, Feige showed some interesting hardness of approximation results. In a follow-up work, Alekhnovich [6] showed strong inapproximability results for the nearest codeword problem and the problem of approximation matrix rigidity based on several other conjectures. His conjectures are in fact closely related to the LWE problem and perhaps using our main theorem, some inapproximability results can be derived based on the quantum-hardness of lattices.

1.1 Overview

In this subsection, we give a brief informal overview of the proof of our main theorem. We do not include a description of the cryptosystem (Section 4) as this part of the paper is more similar to previous work. A complete list of definitions is given in Section 2. The only definition required for this overview is that of the discrete Gaussian distribution $D_{L,r}$. This is the distribution in which the probability of each $\mathbf{x} \in L$ is proportional to $\exp\left(-\pi \|\mathbf{x}/r\|^2\right)$ (see Eq. (4) and Figure 2). Its support is the lattice L and for large enough r (which is the case in all our applications), vectors chosen from it have length roughly $r\sqrt{n}$ with high probability.

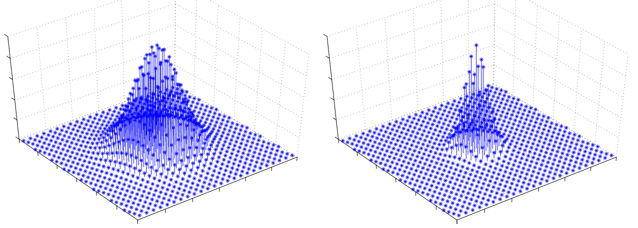

Figure 2: $D_{L,2}$ (left) and $D_{L,1}$ (right) for a two-dimensional lattice L. The z-axis represents probability.

The proof of the main theorem shows a quantum algorithm that solves worst-case lattice problems using an oracle for the $\mathsf{LWE}_{p,\bar{\Psi}_\alpha}$ problem. For simplicity, we consider the SIVP, defined as follows. For a rank n lattice L, let $\lambda_n(L)$ denote the minimum length of a set of n linearly independent lattice vectors from L, where the length of a set is defined as the length of longest vector in it. Our goal is to find a set of n linearly independent vectors whose length is at most $\text{poly}(n)\lambda_n(L)$. In the actual algorithm, an attempt is made to make this polynomial as small as possible. In this overview, we will not be concerned with this issue.

Our algorithm is iterative, in the sense that it starts with some long vectors and repeatedly finds shorter and shorter vectors. More precisely, let $c > 0$ be some constant. Our algorithm starts with n^c samples from $D_{L,r}$ for some large enough r. Each iteration takes as input n^c samples from $D_{L,r}$ and generates n^c samples from $D_{L,r'}$ for $r' = r\sqrt{n}/(\alpha p)$. Notice that since $\alpha p > 2\sqrt{n}$, $r' < r/2$. Hence, by repeating this iterative step, we obtain samples from $D_{L,r}$ for smaller and smaller values of r. The algorithm stops when we no longer find n linearly vectors of length at most $r\sqrt{n}$ among the n^c samples from $D_{L,r}$.

There are two technical conditions that need to be checked. First, we should be able to efficiently produce samples from $D_{L,r}$ for some large enough r. This is indeed possible for $r \geq 2^{O(n)}\lambda_n(L)$ using a simple algorithm. Second, we should show that when the algorithm stops, $r \leq \text{poly}(n)\lambda_n(L)$. This implies that we have a set of n linearly independent vectors of length at most $r\sqrt{n} \leq \text{poly}(n)\lambda_n(L)$, as required. In the rest of this subsection, we ignore these issues and concentrate on the heart of the algorithm: the iterative step.

In the iterative step, we have as input n^c samples from $D_{L,r}$ and we are supposed to generate n^c samples from $D_{L,r'}$. The parameter r can be assumed to be large enough, say $r \geq n^{10}\lambda_n(L)$. We do this in two steps (see Figure 3). In the first step, we use these samples to construct an algorithm that solves $\mathsf{CVP}_{L^*,\alpha p/r}$, i.e., solves the closest vector problem on L^* for points that are within distance $\alpha p/r$ of the lattice. This algorithm is classical and uses the LWE oracle. In the second step, we use this algorithm to generate samples from $D_{L,r'}$. This step is quantum (and in fact, the only quantum part of our proof). In the following, we describe each of these steps briefly.

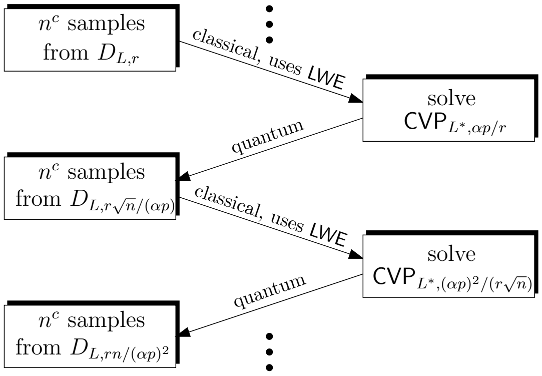
n^c samples from $D_{L,r}$
classical, uses LWE
solve $\mathsf{CVP}_{L^*,\alpha p/r}$
quantum
n^c samples from $D_{L,r\sqrt{n}/(\alpha p)}$
classical, uses LWE
solve $\mathsf{CVP}_{L^*,(\alpha p)^2/(r\sqrt{n})}$
quantum
n^c samples from $D_{L,rn/(\alpha p)^2}$

Figure 3: Two iterations of the algorithm

Step 1: Before describing this step, it would be helpful to recall the main idea in [1]. Consider some probability distribution D on some lattice L and consider its Fourier transform, call it f. This is a function from $\mathbb{R}^n$ to $\mathbb{R}$ that is L^*-periodic, i.e., $f(\mathbf{x}) = f(\mathbf{x}+\mathbf{y})$ for any $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in L^*$. In [1] it was shown that given a polynomial number of samples from D, one can compute an approximation to f to within $\pm 1/\text{poly}(n)$. To see this, we use the definition of the Fourier transform to write f as

$$f(\mathbf{x}) = \sum_{\mathbf{y}\in L} D(\mathbf{y})\exp\left(2\pi i\langle\mathbf{x},\mathbf{y}\rangle\right) = \mathop{\text{Exp}}_{\mathbf{y}\sim D}[\exp\left(2\pi i\langle\mathbf{x},\mathbf{y}\rangle\right)]$$

where in the second equality we simply rewrite the sum as an expectation. By the Chernoff-Hoeffding bound, we see that if $\mathbf{y}_1,\ldots,\mathbf{y}_N$ are $N = \text{poly}(n)$ independent samples from D, then

$$f(\mathbf{x}) \approx \frac{1}{N}\sum_{i=1}^{N}\exp\left(2\pi i\langle\mathbf{x},\mathbf{y}_i\rangle\right)$$

where the approximation is to within $\pm 1/\text{poly}(n)$, given that N is a large enough polynomial.

If we take our samples from the distribution $D_{L,r}$, we obtain a good approximation to its Fourier transform, which we denote by $f_{1/r}$. We omit the exact expression for $f_{1/r}$. It is enough to know that since r is large enough, $f_{1/r}$ is closely approximately by

$$f_{1/r}(\mathbf{x}) \approx \exp\left(-\pi(r\cdot\text{dist}(L^*,\mathbf{x}))^2\right). \tag{1}$$

Hence, $f_{1/r}(\mathbf{x}) = 1$ for any $\mathbf{x} \in L^*$ and as one gets away from L^*, its value decreases. For points within distance, say, $1/r$ from the lattice, its value is still some positive constant $\exp(-\pi)$. However, as the distance from L^* increases, the value of the function soon becomes negligible (see Figure 4). We remark that for our choice of r, the distance between any two vectors in L^* is much larger than $1/r$ and hence these Gaussians are well-separated.

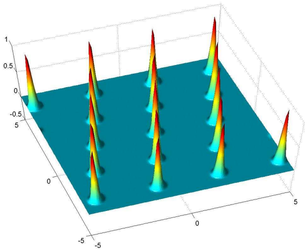

Figure 4: $f_{1/r}$ for a two-dimensional lattice

Although not needed in this paper, let us briefly outline how one can solve $\mathsf{CVP}_{L^*,1/r}$ using samples from $D_{L,r}$ (recall that our final goal is to solve the more difficult $\mathsf{CVP}_{L^*,\alpha p/r}$). Assume we are given some point $\mathbf{x}$ within distance $1/r$ of L^*. Intuitively, this $\mathbf{x}$ is located on one of the Gaussians of $f_{1/r}$. We start 'walking uphill' on $f_{1/r}$ in an attempt to find its 'peak'. This peak corresponds to the closest lattice point to $\mathbf{x}$. This procedure does not quite work: due to the error in our approximation of $f_{1/r}$, we cannot find the closest lattice point exactly. It is possible to overcome this difficulty; we omit further details. Notice that this procedure, unlike our solution to $\mathsf{CVP}_{L^*,\alpha p/r}$, does not need an LWE oracle. It is the LWE oracle that allows us to gain this extra factor of αp in the radius.

It turns out that the $\mathsf{CVP}_{L^*,\alpha p/r}$ can be reduced to the following problem. Given some input vector $\mathbf{x}$ within distance at most $\alpha p/r$ of L^*, let $\mathbf{y} \in L^*$ be the closest vector to $\mathbf{x}$ (for our choice of r, $\mathbf{y}$ is unique). Then the goal is to find the vector of coefficients of $\mathbf{y}$ reduced modulo p. Equivalently, our goal is to find $\tau(\mathbf{x}) := (L^*)^{-1}\mathbf{y} \bmod p \in \mathbb{Z}_p^n$. Hence, in the following it is enough to show how to find $\tau(\mathbf{x})$ given $\mathbf{x}$.

We first notice that the lattice L can be partitioned into p^n translates of the lattice pL. Namely, for each $\mathbf{a} \in \mathbb{Z}_p^n$, consider the set

$$pL + L\mathbf{a} = \{L\mathbf{b} \mid \mathbf{b} \in \mathbb{Z}^n, \quad \mathbf{b} \bmod p = \mathbf{a}\}.$$

Then the set $\{pL + L\mathbf{a} \mid \mathbf{a} \in \mathbb{Z}_p^n\}$ forms a partition of L. We show that for any $\mathbf{a} \in \mathbb{Z}_p^n$, the probability that a point sampled from $D_{L,r}$ is in $pL + L\mathbf{a}$ is very close to p^{-n}. This holds because r is large enough, and each of the translated lattice $pL+L\mathbf{a}$ has roughly the same probability under $D_{L,r}$.

Consider the following simple procedure, call it SamplePair: sample $\mathbf{y}'$ from $D_{L,r}$, let $\mathbf{a} \in \mathbb{Z}_p^n$ be such that $\mathbf{y}' \in pL + L\mathbf{a}$, let $\mathbf{y} = \mathbf{y}'/p$, and output the pair $(\mathbf{a}, \mathbf{y})$. Notice that we can perform this procedure efficiently since we assumed we are able to sample from $D_{L,r}$. From the above discussion we have that the marginal distribution of $\mathbf{a}$ is essentially uniform. Moreover, by definition we have that the distribution of $\mathbf{y}'$ conditioned on any $\mathbf{a}$ is $D_{pL+L\mathbf{a},r}$. Hence the distribution of $\mathbf{y}$ conditioned on any $\mathbf{a}$ is $D_{L+L\mathbf{a}/p,r/p}$. These facts imply that the distribution produced by SamplePair is essentially identical to the following distribution: choose $\mathbf{a} \in \mathbb{Z}_p^n$ uniformly at random, sample $\mathbf{y}$ from $D_{L+L\mathbf{a}/p,r/p}$ and output $(\mathbf{a}, \mathbf{y})$. We use the latter formulation in the following analysis of SamplePair.

If we could somehow obtain samples $(\mathbf{a}, \mathbf{y})$ with $\mathbf{a} = \mathbf{0}$, then the distribution of $\mathbf{y}$ would be $D_{L,r/p}$. As described above, using a polynomial number of such samples, we can solve $\mathsf{CVP}_{L^*,p/r}$ which is even better than what we need (since $\alpha < 1$). However, the probability that SamplePair produces a pair with $\mathbf{a} = 0$ is exponentially small, namely, p^{-n}.

So let us now examine the Fourier transform of $D_{L+L\mathbf{a}/p,r/p}$ for nonzero $\mathbf{a}$ (see Figure 5). Intuitively, since $D_{L+L\mathbf{a}/p,r/p}$ looks like a translation of the distribution $D_{L,r/p}$, we expect its Fourier transform to be the same as that of $D_{L,r/p}$ up to phase. Indeed, a standard calculation shows that the Fourier transform of $D_{L+L\mathbf{a}/p,r/p}$ is given by

$$\exp\left(2\pi i\langle \mathbf{a}, \tau(\mathbf{x})\rangle/p\right) \cdot f_{p/r}(\mathbf{x}). \tag{2}$$

Notice that the absolute value of (2) is $f_{p/r}(\mathbf{x})$. If could somehow obtain a polynomial number of samples from $D_{L+L\mathbf{a}/p,r/p}$ *for the same* $\mathbf{a}$, then we could also obtain a good approximation to (2). By taking its absolute value, we would obtain a good approximation to $f_{p/r}$ from which we can proceed as before to solve $\mathsf{CVP}_{L^*,p/r}$. The problem with this solution is that SamplePair produces pairs $(\mathbf{a}, \mathbf{y})$ for some uniform value $\mathbf{a} \in \mathbb{Z}_p^n$. With very high probability, we never obtain any value of $\mathbf{a}$ more than once. So it seems that we cannot hope to approximate (2) directly; a new approach is needed.

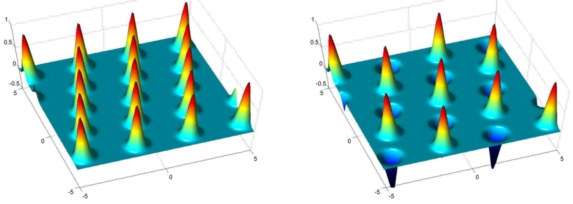

Figure 5: The Fourier transform of $D_{L+L\mathbf{a}/p,r/p}$ with $n = 2$, $p = 2$, $\mathbf{a} = (0, 0)$ (left), $\mathbf{a} = (1, 1)$ (right).

To explain our approach, let us step back and examine the expression $\exp(2\pi i\langle \mathbf{x}, \mathbf{y}\rangle)$. We know that its average over $\mathbf{y} \sim D_{L+L\mathbf{a}/p,r/p}$ is given by (2). For $\mathbf{x} \in L^*$, (2) is equal to $\exp(2\pi i\langle \mathbf{a}, \tau(\mathbf{x})\rangle/p)$. Since the absolute value of this expression is 1, we see that for such $\mathbf{x}$, the random variable $\langle \mathbf{x}, \mathbf{y}\rangle \bmod 1$ (where $\mathbf{y} \sim D_{L+L\mathbf{a}/p,r/p}$) is deterministically equal to $\langle \mathbf{a}, \tau(\mathbf{x})\rangle/p \bmod 1$ (this fact can also be shown more directly).

The situation becomes more interesting once we take a vector $\mathbf{x}$ whose distance to L^* is $\beta p/r$ for some $0 \leq \beta \leq \alpha$. Then the phase component of (2) remains the same (i.e., $\exp(2\pi i\langle \mathbf{a}, \tau(\mathbf{x})\rangle/p)$) but its magnitude decreases to some positive constant (namely, $\exp\left(-\pi\beta^2\right)$). One of our main technical contributions is to show that for such $\mathbf{x}$, the distribution of $\langle \mathbf{x}, \mathbf{y}\rangle \bmod 1$ (where $\mathbf{y} \sim D_{L+L\mathbf{a}/p,r/p}$) is close to a Gaussian centered around $\langle \mathbf{a}, \tau(\mathbf{x})\rangle/p \bmod 1$ with standard deviation β. Notice that for the case $\beta = 0$ we obtain the situation described before. Also note that as β increases, $\langle \mathbf{x}, \mathbf{y}\rangle \bmod 1$ becomes less concentrated. This causes the absolute value of (2) to decrease. Finally, let us remark that the distribution $\langle \mathbf{x}, \mathbf{y}\rangle \bmod 1$ is discrete; when we say that it is close to a Gaussian, we actually mean that by adding a small amount of noise, the distribution becomes a continuous distribution that is very close to a Gaussian.

Fix some $\mathbf{x}$ whose distance to L^* is $\beta p/r$ for some $0 \leq \beta \leq \alpha$. Then by the above discussion we know that if $\mathbf{y}$ is sampled from $D_{L+L\mathbf{a}/p,r/p}$ then $p\langle \mathbf{x}, \mathbf{y}\rangle \bmod p$ is distributed like a Gaussian around $\langle \mathbf{a}, \tau(\mathbf{x})\rangle \bmod p$ with standard deviation βp. By rounding, we obtain that $\lfloor p\langle \mathbf{x}, \mathbf{y}\rangle\rceil \bmod p$ is distributed like $\bar{\Psi}_\beta$ around $\langle \mathbf{a}, \tau(\mathbf{x})\rangle \bmod p$. This can be written as an 'equation with error'

$$\langle \mathbf{a}, \tau(\mathbf{x})\rangle \approx_{\bar{\Psi}_\beta} \lfloor p\langle \mathbf{x}, \mathbf{y}\rangle\rceil \pmod{p}.$$

We now create a polynomial number of equations and use the LWE oracle to find $\tau(\mathbf{x})$.

To summarize, our algorithm for solving $\mathsf{CVP}_{L^*,\alpha p/r}$ is the following. Let $\mathbf{x}$ be the input vector. Using SamplePair, obtain a polynomial number of pairs $(\mathbf{a}_i, \mathbf{y}_i)$. We call the LWE oracle with the pairs $(\mathbf{a}_i, \lfloor p\langle \mathbf{x}, \mathbf{y}_i\rangle\rceil)$. By the arguments above, the LWE oracle outputs $\tau(\mathbf{x})$. As mentioned before, this is enough to solve $\mathsf{CVP}_{L^*,\alpha p/r}$.

The above description is not entirely accurate as the LWE oracle is only supposed to work for the noise distribution $\bar{\Psi}_\alpha$ and might not work for noise distribution $\bar{\Psi}_\beta$ for $\beta < \alpha$. We address this issue in the paper by showing how to transform an oracle that works with $\bar{\Psi}_\alpha$ to an oracle that works with $\bar{\Psi}_\beta$ for any (unknown) $\beta < \alpha$.

Step 2: In this step, we describe a quantum algorithm that, using a $\mathsf{CVP}_{L^*,\alpha p/r}$ oracle, generates one sample from $D_{L,r\sqrt{n}/(\alpha p)}$ (one can generate more samples by simply repeating the algorithm). Equivalently, let us show how to produce a sample from $D_{L,r}$ given a $\mathsf{CVP}_{L^*,\sqrt{n}/r}$ oracle. The procedure is essentially the following: first, by using the CVP oracle, create a quantum state corresponding to $f_{1/r}$. Then, apply the quantum Fourier transform and obtain a quantum state corresponding to $D_{L,r}$. By measuring this state we obtain a sample from $D_{L,r}$.

In the following, we describe this procedure in more detail. Our first goal is to create a quantum state corresponding to $f_{1/r}$. Informally, this can be written as

$$\sum_{\mathbf{x}\in\mathbb{R}^n} f_{1/r}|\mathbf{x}\rangle. \tag{3}$$

This state is clearly not well-defined. In the actual procedure, $\mathbb{R}^n$ is replaced with some finite set (namely, all points inside the basic parallelepiped of L^* that belong to some fine grid). This introduces several technical complications and makes the computations rather tedious. Therefore, in the rest of this subsection, we opt to continue with informal expressions as in (3).

Let us now continue our description of the procedure. In order to prepare the state in (3), we first create the uniform superposition on L^*, $\sum_{\mathbf{x}\in L^*}|\mathbf{x}\rangle$. On a separate register, we create a 'Gaussian state' of width $1/r$, $\sum_{\mathbf{z}\in\mathbb{R}^n}\exp\left(-\pi\|r\mathbf{z}\|^2\right)|\mathbf{z}\rangle$. This can be done using known techniques. The combined state of the system can be written as

$$\sum_{\mathbf{x}\in L^*,\mathbf{z}\in\mathbb{R}^n}\exp\left(-\pi\|r\mathbf{z}\|^2\right)|\mathbf{x},\mathbf{z}\rangle.$$

We now add the first register to the second (a reversible operation), and obtain

$$\sum_{\mathbf{x}\in L^*,\mathbf{z}\in\mathbb{R}^n}\exp\left(-\pi\|r\mathbf{z}\|^2\right)|\mathbf{x},\mathbf{x}+\mathbf{z}\rangle.$$

Finally, we would like to *erase*, or *uncompute*, the first register to obtain

$$\sum_{\mathbf{x}\in L^*,\mathbf{z}\in\mathbb{R}^n}\exp\left(-\pi\|r\mathbf{z}\|^2\right)|\mathbf{x}+\mathbf{z}\rangle \approx \sum_{\mathbf{z}\in\mathbb{R}^n} f_{1/r}|\mathbf{z}\rangle.$$

However, 'erasing' a register is in general not a reversible operation. In order for it to be reversible, we need to able to compute $\mathbf{x}$ from the remaining register $\mathbf{x}+\mathbf{z}$. This is precisely why we need the $\mathsf{CVP}_{L^*,\sqrt{n}/r}$ oracle. It can be shown that almost all the mass of $\exp\left(-\pi\|r\mathbf{z}\|^2\right)$ is on $\mathbf{z}$ such that $\|\mathbf{z}\| \le \sqrt{n}/r$. Hence, $\mathbf{x}+\mathbf{z}$ is within distance $\sqrt{n}/r$ of the lattice and the oracle finds the closest lattice point, namely, $\mathbf{x}$. This allows us to erase the first register in a reversible way.

In the final part of the procedure, we apply the quantum Fourier transform. This yields the quantum state corresponding to $D_{L,r}$, namely, $\sum_{\mathbf{y}\in L} D_{L,r}|\mathbf{y}\rangle$. By measuring this state, we obtain a sample from the distribution $D_{L,r}$ (in fact, the probability distribution is $D^2_{L,r} = D_{L,r/\sqrt{2}}$ but this is a minor issue).

2. PRELIMINARIES

In this section we include some notation that will be used throughout the paper. Most of the notation is standard. Less standard notation includes: the Gaussian function ρ, the Gaussian distribution ν, the periodic normal distribution Ψ, the discretization of a distribution on $\mathbb{T}$, the discrete Gaussian distribution D, the unique closest lattice vector κ, and the smoothing parameter η.

General. All logarithms are of base 2 unless otherwise specified. For two real numbers x and $y > 0$ we define $x \bmod y$ as $x - \lfloor x/y \rfloor y$. For $x \in \mathbb{R}$ we define $\lfloor x \rceil$ as the integer closest to x or, in case two such integers exist, the smaller of the two. For any integer $p \ge 2$, we write $\mathbb{Z}_p$ for the cyclic group $\{0,1,\ldots,p-1\}$ with addition modulo p. We also write $\mathbb{T}$ for $\mathbb{R}/\mathbb{Z}$, i.e., the segment $[0,1)$ with addition modulo 1.

We define a negligible amount in n as an amount that is asymptotically smaller than n^{-c} for any constant $c > 0$. More precisely, $f(n)$ is a negligible function in n if $\lim_{n\to\infty} n^c f(n) = 0$ for any $c > 0$. Similarly, a non-negligible amount is one which is at least n^{-c} for some $c > 0$. Also, when we say that an expression is exponentially small in n we mean that it is at most $2^{-\Omega(n)}$. Finally, when we say that an expression (most often, some probability) is exponentially close to 1, we mean that it is $1 - 2^{-\Omega(n)}$.

We say that an algorithm $\mathcal{A}$ with oracle access is a distinguisher between two distributions if its acceptance probability when the oracle outputs samples of the first distribution and its acceptance probability when the oracle outputs samples of the second distribution differ by a non-negligible amount.

Essentially all algorithms and reductions in this paper are probabilistic and work with probability exponentially close to 1.

For clarity, we present some of our reductions in a model that allows operations on real numbers. It is possible to modify them in a straightforward way so that they operate in a model that approximates real numbers up to an error of 2^{-n^c} for arbitrary large constant c in time polynomial in n.

Given two probability density functions ϕ_1, ϕ_2 on $\mathbb{R}^n$, we define the statistical distance between them as

$$\Delta(\phi_1,\phi_2) := \int_{\mathbb{R}^n} |\phi_1(\mathbf{x}) - \phi_2(\mathbf{x})| d\mathbf{x}$$

(notice that with this definition, the statistical distance ranges in $[0,2]$). A similar definition holds for discrete random variables. We sometimes abuse notation, and use the same notation for two arbitrary functions. We note that the triangle inequality

$$\Delta(f_1,f_3) \le \Delta(f_1,f_2) + \Delta(f_2,f_3)$$

holds for any functions f_1, f_2, f_3. One important fact that we use is that the statistical distance cannot increase by applying a (possibly randomized) function f, i.e.,

$$\Delta(f(X),f(Y)) \le \Delta(X,Y),$$

see, e.g., [15]. In particular, this implies that the acceptance probability of any algorithm on inputs from X differs from its acceptance probability on inputs from Y by at most $\frac{1}{2}\Delta(X,Y)$ (the factor half coming from the choice of normalization in our definition of Δ).

Gaussians and other distributions. Recall that the *normal distribution* with mean 0 and variance σ^2 is the distribution on $\mathbb{R}$ given by the density function $\frac{1}{\sqrt{2\pi}\cdot\sigma}\exp\left(-\frac{1}{2}(\frac{x}{\sigma})^2\right)$ where $\exp(y)$ denotes e^y. Also recall that the sum of two independent normal variables with mean 0 and variances σ_1^2 and σ_2^2 is a normal variable with mean 0 and variance $\sigma_1^2 + \sigma_2^2$. For a vector $\mathbf{x}$ and any $s > 0$, let

$$\rho_s(\mathbf{x}) = \exp\left(-\pi\|\mathbf{x}/s\|^2\right)$$

be a Gaussian function scaled by a factor of s. Note that $\int_{\mathbf{x}\in\mathbb{R}^n} \rho_s(\mathbf{x})d\mathbf{x} = s^n$. So, $\nu_s := \rho_s/s^n$ is an n-dimensional probability density function. The dimension n is implicit. Notice that the Gaussian distribution ν_s can be expressed as the sum of n orthogonal 1-dimensional Gaussian distributions, and each of them can be efficiently approximated with arbitrary precision using standard techniques. So, the distribution ν_s can be efficiently approximated. For simplicity, in this paper we assume we can sample from ν_s exactly.[1] When s is not specified, we assume that it is 1. Functions are extended to sets in the usual way; e.g., $\rho_s(A) = \sum_{\mathbf{x}\in A} \rho_s(\mathbf{x})$ for any countable set A. For any vector $\mathbf{c} \in \mathbb{R}^n$, we define $\rho_{s,\mathbf{c}}(\mathbf{x}) := \rho_s(\mathbf{x} - \mathbf{c})$ to be a shifted version of ρ_s.

For any countable set A and a parameter $s > 0$, we define the discrete Gaussian probability distribution $D_{A,s}$ as

$$\forall \mathbf{x} \in A,\ D_{A,s}(\mathbf{x}) = \frac{\rho_s(\mathbf{x})}{\rho_s(A)}. \qquad (4)$$

See Figure 2 for an illustration.

For $\beta \in \mathbb{R}^+$ the distribution Ψ_β is the distribution on $\mathbb{T}$ obtained by sampling from a normal variable with mean 0 and standard deviation $\frac{\beta}{\sqrt{2\pi}}$ and reducing the result modulo 1 (i.e., a periodization of the normal distribution):

$$\Psi_\beta(r) := \sum_{k=-\infty}^{\infty} \frac{1}{\beta} \cdot \exp\left(-\pi\Big(\frac{r-k}{\beta}\Big)^2\right).$$

Clearly, one can efficiently sample from Ψ_β. For an arbitrary probability distribution with density function $\phi : \mathbb{T} \to \mathbb{R}^+$ and some integer $p > 0$ we define its discretization $\bar{\phi} : \mathbb{Z}_p \to \mathbb{R}^+$ as the discrete probability distribution obtained by sampling from ϕ, multiplying by p, and rounding to the closest integer modulo p. More formally,

$$\bar{\phi}(i) := \int_{(i-1/2)/p}^{(i+1/2)/p} \phi(x)dx.$$

As an example, $\bar{\Psi}_\beta$ is shown in Figure 1.

Let $p \geq 2$ be some integer, and let $\chi : \mathbb{Z}_p \to \mathbb{R}^+$ be some probability distribution on $\mathbb{Z}_p$. Let n be an integer and let $\mathbf{s} \in \mathbb{Z}_p^n$ be a vector. We define $A_{\mathbf{s},\chi}$ as the distribution on $\mathbb{Z}_p^n \times \mathbb{Z}_p$ obtained by choosing a vector $\mathbf{a} \in \mathbb{Z}_p^n$ uniformly at random, choosing $e \in \mathbb{Z}_p$ according to χ, and outputting $(\mathbf{a}, \langle\mathbf{a},\mathbf{s}\rangle + e)$, where additions are performed in $\mathbb{Z}_p$, i.e., modulo p. We also define U as the uniform distribution on $\mathbb{Z}_p^n \times \mathbb{Z}_p$.

For a probability density function ϕ on $\mathbb{T}$, we define $A_{\mathbf{s},\phi}$ as the distribution on $\mathbb{Z}_p^n \times \mathbb{T}$ obtained by choosing a vector $\mathbf{a} \in \mathbb{Z}_p^n$ uniformly at random, choosing $e \in \mathbb{T}$ according to ϕ, and outputting $(\mathbf{a}, \langle\mathbf{a},\mathbf{s}\rangle/p + e)$, where the addition is performed in $\mathbb{T}$, i.e., modulo 1.

Learning with errors. For an integer $p = p(n)$ and a distribution χ on $\mathbb{Z}_p$, we say that an algorithm solves $\mathsf{LWE}_{p,\chi}$ if, given samples from $A_{\mathbf{s},\chi}$ for some arbitrary $\mathbf{s}$, it outputs $\mathbf{s}$ with probability exponentially close to 1. Similarly, for a probability density function ϕ on $\mathbb{T}$, we say that an algorithm solves $\mathsf{LWE}_{p,\phi}$ if, given samples from $A_{\mathbf{s},\phi}$ for some arbitrary $\mathbf{s}$, it outputs $\mathbf{s}$ with probability exponentially close to 1. In both cases, we say that the algorithm is efficient if it runs in polynomial time in n. Finally, we note that p is assumed to be prime only in the cryptographic application. In the main theorem, p can be any arbitrary integer.

Lattices. We briefly review some basic definitions; for a good introduction to lattices, see [15]. A lattice in $\mathbb{R}^n$ is defined as the set of all integer combinations of n linearly independent vectors. This set of vectors is known as a basis of the lattice and is not unique. Given a basis $(\mathbf{v}_1, \ldots, \mathbf{v}_n)$ of a lattice L, the fundamental parallelepiped is defined as

$$\mathcal{P}(\mathbf{v}_1, \ldots, \mathbf{v}_n) = \left\{ \sum_{i=1}^{n} x_i\mathbf{v}_i \;\middle|\; x_i \in [0,1) \right\}.$$

When the choice of basis is clear, we write $\mathcal{P}(L)$ instead of $\mathcal{P}(\mathbf{v}_1, \ldots, \mathbf{v}_n)$. Note that a lattice has a different fundamental parallelepiped for every possible basis. We denote by $\det(L)$ the volume of the fundamental parallelepiped of L or equivalently, the determinant of the matrix whose columns are the basis vectors of the lattice ($\det(L)$ is a lattice invariant, i.e., it is independent of the choice of basis). The dual of a lattice L in $\mathbb{R}^n$, denoted L^*, is the set of all vectors $y \in \mathbb{R}^n$ such that $\langle x, y\rangle \in \mathbb{Z}$ for all vectors $x \in L$. Similarly, given a basis $(\mathbf{v}_1, \ldots, \mathbf{v}_n)$ of a lattice, we define the dual basis as the set of vectors $(\mathbf{v}_1^*, \ldots, \mathbf{v}_n^*)$ such that $\langle\mathbf{v}_i, \mathbf{v}_j^*\rangle = \delta_{ij}$ for all $i, j \in [n]$ where δ_{ij} denotes the Kronecker delta, i.e., 1 if $i = j$ and 0 otherwise. With a slight abuse of notation, we sometimes write L for the $n \times n$ matrix whose columns are $\mathbf{v}_1, \ldots, \mathbf{v}_n$. With this notation, we notice that $L^* = (L^T)^{-1}$. From this it follows that $\det(L^*) = 1/\det(L)$. As another example of this notation, for a point $\mathbf{v} \in L$ we write $L^{-1}\mathbf{v}$ to indicate the coefficient vector of $\mathbf{v}$.

The point $\mathbf{x} \in \mathbb{R}^n$ reduced modulo the parallelepiped $\mathcal{P}(L)$, denoted $x \bmod \mathcal{P}(L)$, is the unique point $\mathbf{y} \in \mathcal{P}(L)$ such that $\mathbf{y} - \mathbf{x} \in L$.

Let $\lambda_1(L)$ denote the length of the shortest nonzero vector in the lattice L. We denote by $\lambda_n(L)$ the minimum length of a set of n linearly independent vectors from L, where the length of a set is defined as the length of longest vector in it. For a lattice L and a point v whose distance from L is less than $\lambda_1(L)/2$ we define $\kappa_L(v)$ as the (unique) closest point to v in L.

Two main lattice problem are the SVP and the SIVP. The former is concerned with finding (or approximating) $\lambda_1(L)$, while the latter is concerned with $\lambda_n(L)$. We also define a non-standard variant of the closest vector. For an n-dimensional lattice L, and some $d > 0$, we say that an algorithm solves $\mathsf{CVP}_{L,d}$ if, given a point $\mathbf{x} \in \mathbb{R}^n$ whose distance to L is at most d, the algorithm finds the closest lattice point to $\mathbf{x}$. We note that in our applications, d will always be smaller than $\lambda_1(L)/2$. Hence, the closest vector is unique.

[1] In practice, when only finite precision is available, ν_s can be approximated by picking a fine grid, and picking points from the grid with probability approximately proportional to ν_s. All our arguments can be made rigorous by selecting a sufficiently fine grid.

The smoothing parameter. In our analysis, we use a relatively new lattice parameter known as the *smoothing parameter* [16]. This makes the analysis tighter and allows us to obtain better approximation factors. Intuitively, this parameter quantifies the amount of noise that one has to add to a lattice in order for it to lose its discrete structure. The precise definition is the following:

DEFINITION 2.1. *For an n-dimensional lattice L, and positive real $\varepsilon > 0$, we define its smoothing parameter $\eta_\varepsilon(L)$ to be the smallest s such that $\rho_{1/s}(L^* \setminus \{\mathbf{0}\}) \leq \varepsilon$.*

In other words, $\eta_\varepsilon(L)$ is the smallest s such that a Gaussian measure on the dual lattice L^* scaled by $1/s$ gives all but $\varepsilon/(1+\varepsilon)$ of its weight to the origin. We usually take ε to be some negligible function of the lattice dimension. Notice that $\rho_{1/s}(L^* \setminus \{\mathbf{0}\})$ is a continuous and strictly decreasing function of s such that $\lim_{s\to 0} \rho_{1/s}(L^* \setminus \{\mathbf{0}\}) = \infty$ and $\lim_{s\to\infty} \rho_{1/s}(L^* \setminus \{\mathbf{0}\}) = 0$. So, the parameter $\eta_\varepsilon(L)$ is well defined for any $\varepsilon > 0$, and $\varepsilon \mapsto \eta_\varepsilon(L)$ is the inverse function of $s \mapsto \rho_{1/s}(L^* \setminus \{\mathbf{0}\})$. In particular, $\eta_\varepsilon(L)$ is also a continuous and strictly decreasing function of ε.

The motivation for this definition (and the name 'smoothing parameter') comes from the following result, shown in [16]. Informally, it says that if we choose a 'random' lattice point from an n-dimensional lattice L and add noise ν_s for some $s > \eta_\varepsilon(L)$ then the resulting distribution is within statistical distance ε of the 'uniform distribution on $\mathbb{R}^n$' (think of ε as some negligible function of n). The formal statement is given by considering distributions on $\mathcal{P}(L)$, see [16] for further details. In this paper, we show another important property of this parameter: for $s > \sqrt{2}\eta_\varepsilon(L)$, if we sample a point from $D_{L,s}$ and add Gaussian noise ν_s, we obtain a distribution whose statistical distance to a continuous Gaussian $\nu_{\sqrt{2}s}$ is at most 4ε. Notice that $\nu_{\sqrt{2}s}$ is the distribution one expects to obtain when summing two independent samples from ν_s. Hence, intuitively, the noise ν_s is enough to hide the discrete the discrete structure of $D_{L,s}$.

The following two upper bounds on the smoothing parameter appear in [16].

LEMMA 2.2. *For any n-dimensional lattice L, $\eta_\varepsilon(L) \leq \sqrt{n}/\lambda_1(L^*)$ where $\varepsilon = 2^{-n}$.*

LEMMA 2.3. *For any n-dimensional lattice L and any superlogarithmic function $\omega(\log n)$, $\eta_{\varepsilon(n)}(L) \leq \sqrt{\omega(\log n)} \cdot \lambda_n$ for some negligible function $\varepsilon(n)$.*

We also need the following simple lower bound on the smoothing parameter (proof omitted).

CLAIM 2.4. *For any lattice L, any negligible function $\varepsilon(n)$, and any constant $c > 0$, $\eta_{\varepsilon(n)}(L) > c/\lambda_1(L^*)$ for large enough n.*

The Fourier transform. We briefly review some of the important properties of the Fourier transform. In the following, we omit certain technical conditions as these will always be satisfied in our applications. For a more precise and in-depth treatment, see, e.g., [10]. The Fourier transform of a function $h : \mathbb{R}^n \mapsto \mathbb{R}$ is defined to be

$$\hat{h}(\mathbf{w}) = \int_{\mathbb{R}^n} h(\mathbf{x}) e^{-2\pi i \langle \mathbf{x},\mathbf{w}\rangle} d\mathbf{x}.$$

From the definition we can obtain two useful formulas; first, if h is defined by $h(\mathbf{x}) = g(\mathbf{x}+\mathbf{v})$ for some function g and vector $\mathbf{v}$ then

$$\hat{h}(\mathbf{w}) = e^{2\pi i \langle \mathbf{v},\mathbf{w}\rangle} \hat{g}(\mathbf{w}). \tag{5}$$

Similarly, if h is defined by $h(\mathbf{x}) = e^{2\pi i \langle \mathbf{x},\mathbf{v}\rangle} g(\mathbf{x})$ for some function g and vector $\mathbf{v}$ then

$$\hat{h}(\mathbf{w}) = \hat{g}(\mathbf{w} - \mathbf{v}). \tag{6}$$

Another important fact is that the Gaussian is its own Fourier transform, i.e., $\hat{\rho} = \rho$. More generally, for any $s > 0$ it holds that $\widehat{\rho_s} = s^n \rho_{1/s}$. We will use the following formulation of the Poisson summation formula:

LEMMA 2.5. *For any lattice L and any function $f : \mathbb{R}^n \to \mathbb{C}$, $f(L) = \det(L^*)\hat{f}(L^*)$ where $\hat{f}$ denotes the Fourier transform of f.*

3. MAIN THEOREM

THEOREM 3.1. *Let $\varepsilon = \varepsilon(n)$ be some negligible function of n. Also, let $\alpha = \alpha(n) \in (0,1)$ be some real and let $p = p(n)$ be some integer such that $\alpha p > 2\sqrt{n}$. Assume there exists an efficient (possibly quantum) algorithm W that solves $\mathsf{LWE}_{p,\Psi_\alpha}$. Then there exists an efficient quantum algorithm for solving the following worst-case lattice problems:*

- *Find a set of n linearly independent lattice vectors of length at most $2n \cdot \eta_\varepsilon(L)/\alpha \leq \tilde{O}(\lambda_n(L) \cdot n/\alpha)$.*
- *Approximate $\lambda_1(L)$ to within $\tilde{O}(n/\alpha)$.*

PROOF. The proof involves several steps. However, due to space limitation, only parts of the iterative step are included. We refer the reader to the full version for more details. □

3.1 The iterative step

LEMMA 3.2. *Let L be an n-dimensional lattice, $\varepsilon = \varepsilon(n)$ be some negligible function, $\alpha = \alpha(n) \in (0,1)$ be some real, and $p = p(n) \geq 2$ be some integer. Assume we are given n^{c_1} samples from $D_{L,r}$ for some $r > p\eta_\varepsilon(L)$ and a large enough c_1. Also assume that we have an efficient algorithm W that solves $\mathsf{LWE}_{p,\Psi_\alpha}$. Then, there exists an efficient quantum algorithm that produces any polynomial number of samples from $D_{L,r\sqrt{n}/(\alpha p)}$.*

PROOF. The algorithm consists of two main parts. The first part is shown in Lemma 3.3. There, we describe a (classical) algorithm that using W and the samples from $D_{L,r}$, solves $\mathsf{CVP}_{L^*,\alpha p/r}$. In the second part we describe a quantum algorithm that, given an oracle that solves $\mathsf{CVP}_{L^*,\alpha p/r}$, outputs samples from $D_{L,r\sqrt{n}/(\alpha p)}$. This is the only quantum part in this paper and is omitted.

Note that by Claim 2.4,

$$\frac{\alpha p}{r} \leq \frac{\alpha}{\eta_\varepsilon(L)} \leq \frac{1}{\eta_\varepsilon(L)} \leq \frac{\lambda_1(L^*)}{2}$$

and hence the $\mathsf{CVP}_{L^*,\alpha p/r}$ is well-defined. □

3.1.1 *From samples to* CVP

The main result of this subsection is the following:

Lemma 3.3. *Let L be an n-dimensional lattice, $\varepsilon = \varepsilon(n)$ be some negligible function, $\alpha = \alpha(n) \in (0,1)$ be some real, and $p \geq 2$ be some integer. Assume we can sample from $D_{L,r}$ for some $r > p\eta_\varepsilon(L)$. Also assume that we have an efficient algorithm W that solves $\mathsf{LWE}_{p,\Psi_\alpha}$. Then, there exists an efficient algorithm that solves $\mathsf{CVP}_{L^*,\alpha p/r}$.*

Proof. By scaling, we can assume without loss of generality that $r = 1$. The proof follows by combining several lemmas. First, using Lemma 3.4, we obtain an algorithm W' that, given samples from $A_{\mathbf{s},\psi_\beta}$ for some (unknown) $\beta < \alpha$, outputs $\mathbf{s}$ with probability exponentially close to 1. Using Lemma 3.6 with W' we obtain an algorithm that finds, given any point $\mathbf{x}$ within distance αp of L^*, $(L^*)^{-1}\kappa_{L^*}(\mathbf{x}) \bmod p \in \mathbb{Z}_p^n$, the coefficient vector of the closest vector to $\mathbf{x}$ reduced modulo p. By the remark above we know that $\alpha p < \lambda_1(L^*)/2$. Hence, using Lemma 3.5, we obtain a solution for $\mathsf{CVP}_{L^*,\alpha p}$. $\square$

The following lemma shows that being able to extract $\mathbf{s}$ from samples of $A_{\mathbf{s},\psi_\alpha}$ implies being able to extract $\mathbf{s}$ from samples of $A_{\mathbf{s},\psi_\beta}$ for any $\beta < \alpha$, even without knowing β.

Lemma 3.4. *Assume there exists an efficient algorithm W that solves $\mathsf{LWE}_{p,\psi_\alpha}$ for some $\alpha < 1$. Then, there exists an algorithm W' that, given samples from $A_{\mathbf{s},\psi_\beta}$ for some (unknown) $\beta < \alpha$, outputs $\mathbf{s}$ with probability exponentially close to 1.*

Proof. The proof is based on the following idea: by adding the right amount of noise, we can transform samples from $A_{\mathbf{s},\psi_\beta}$ to samples from $A_{\mathbf{s},\psi_\alpha}$. We omit the details. $\square$

The following lemma shows that in order to solve CVP, it is enough to be able to find the coefficients modulo p of the closest lattice point (proof omitted).

Lemma 3.5. *Let L be an n-dimensional lattice, let d be a positive real number smaller than $\lambda_1(L)/2$, and let $p \geq 2$ be some integer. Assume we are given an oracle that given a point $\mathbf{x} \in \mathbb{R}^n$ within distance d of L, finds $L^{-1}\kappa_L(\mathbf{x}) \bmod p \in \mathbb{Z}_p^n$. Then, there exists an algorithm that solves $\mathsf{CVP}_{L,d}$ using this oracle.*

Lemma 3.6. *Let L be an n-dimensional lattice, let $\alpha = \alpha(n) < 1$ be some positive real, and let $p \geq 2$ be some integer. Assume we can sample from $D_{L,1}$. Assume, moreover, that $\eta_\varepsilon(L) \leq 1/p$ for some negligible function $\varepsilon = \varepsilon(n)$. Also assume that we have an oracle W that finds $\mathbf{s}$ given samples from $A_{\mathbf{s},\psi_\beta}$ for all $\beta < \sqrt{2}\alpha$. Then, given any $\mathbf{x}$ point within distance αp of L^*, we can find $(L^*)^{-1}\kappa_{L^*}(\mathbf{x}) \bmod p \in \mathbb{Z}_p^n$, the coefficient vector of the closest vector to $\mathbf{x}$ reduced modulo p.*

Proof. We describe a procedure that given $\mathbf{x}$ as above outputs samples from the distribution $A_{\mathbf{s},\psi_\beta}$ for some $\beta < \sqrt{2}\alpha$ where $\mathbf{s} = (L^*)^{-1}\kappa_{L^*}(\mathbf{x}) \bmod p$. By performing this procedure a polynomial number of times and then using W, we can find $\mathbf{s}$.

The procedure works as follows. We first sample a vector $\mathbf{v} \in L$ from $D_{L,1}$. Let $\mathbf{a} = L^{-1}\mathbf{v} \bmod p$. We then compute $r = \langle \mathbf{x}, \mathbf{v}\rangle/p + e \bmod 1$ where $e \in \mathbb{R}$ is chosen according to a normal distribution with standard deviation $\alpha/\sqrt{2\pi}$. The output of the procedure is $(\mathbf{a}, r)$. We claim that the distribution given by this procedure is exponentially close to $A_{\mathbf{s},\psi_\beta}$ for some $\beta < \sqrt{2}\alpha$.

Let us first show that the distribution of $\mathbf{a}$ is very close to uniform. Fix some $\mathbf{a} \in \mathbb{Z}_p^n$. Then, the probability of obtaining this $\mathbf{a}$ is proportional to

$$\begin{aligned}\sum_{\mathbf{x}\in pL+L\mathbf{a}} \rho(\mathbf{x}) &= \sum_{\mathbf{x}\in pL} \rho_{-L\mathbf{a}}(x)\\ &= p^{-n}\det(L^*) \sum_{\mathbf{y}\in L^*/p} \widehat{\rho_{-L\mathbf{a}}}(\mathbf{y}) \quad \text{(by Lemma 2.5)}\\ &= p^{-n}\det(L^*) \sum_{\mathbf{y}\in L^*/p} \exp\left(-2\pi i\langle L\mathbf{a}, \mathbf{y}\rangle\right)\widehat{\rho}(\mathbf{y})\\ &= p^{-n}\det(L^*)(1 \pm \varepsilon) \quad \text{(since } \eta_\varepsilon(pL) = p\eta_\varepsilon(L) \leq 1).\end{aligned}$$

This implies that the statistical distance between the distribution of $\mathbf{a}$ and the uniform distribution is negligible.

Next, we consider the distribution of r conditioned on some fixed value of $\mathbf{a}$. Define $\mathbf{x}' = \mathbf{x} - \kappa_{L^*}(\mathbf{x})$. Then,

$$\langle \mathbf{x}, \mathbf{v}\rangle/p + e \bmod 1 = \langle \mathbf{x}', \mathbf{v}\rangle/p + e + \langle \kappa_{L^*}(\mathbf{x}), \mathbf{v}\rangle/p \bmod 1.$$

Now, $\langle \kappa_{L^*}(\mathbf{x}), \mathbf{v}\rangle = \langle L^* \cdot (L^*)^{-1} \cdot \kappa_{L^*}(\mathbf{x}), L \cdot L^{-1} \cdot \mathbf{v}\rangle = \langle (L^*)^{-1}\kappa_{L^*}(\mathbf{x}), L^{-1}\mathbf{v}\rangle$ where the last equality follows since $(L^*)^T = L^{-1}$. In words, this says that the inner product between $\kappa_{L^*}(\mathbf{x})$ and $\mathbf{v}$ (and in fact, between any vector in L and a vector in L^*) is the same as the inner product between the corresponding coefficient vectors. Since the coefficient vectors are integer,

$$\langle \kappa_{L^*}(\mathbf{x}), \mathbf{v}\rangle \bmod p = \langle \mathbf{s}, \mathbf{a}\rangle \bmod p$$

from which it follows that $\langle \kappa_{L^*}(\mathbf{x}), \mathbf{v}\rangle/p \bmod 1$ is exactly $\langle \mathbf{s}, \mathbf{a}\rangle/p \bmod 1$. Hence, in the following it is enough to show that the statistical distance between the distribution of $\langle \mathbf{x}', \mathbf{v}\rangle/p + e \bmod 1$ and that of ψ_β is negligible for some $\beta < \sqrt{2}\alpha$. In fact, we will show the stronger statement that the statistical distance between the distribution of $\langle \mathbf{x}', \mathbf{v}\rangle/p + e$ and that of a normal variable with mean 0 and standard deviation $\beta/\sqrt{2\pi}$ is negligible for some $\beta < \sqrt{2}\alpha$ (this statement is stronger since statistical distance cannot increase by applying a function, mod 1 in this case).

So let us consider the distribution of $\langle \mathbf{x}', \mathbf{v}\rangle/p + e$. Notice that conditioned on the value $\mathbf{a}$, the distribution of $\mathbf{v}$ is $D_{pL+L\mathbf{a}}$. It can be seen that the distribution of $\langle \mathbf{x}', \mathbf{v}\rangle/p + e$ is identical to the distribution obtained by sampling $\mathbf{v}$ from $D_{pL+L\mathbf{a}}$ and a noise vector $\mathbf{h}$ from $\rho_{p\alpha/\|\mathbf{x}'\|}/(p\alpha/\|\mathbf{x}'\|)^n$ and outputting $\langle \mathbf{x}', \mathbf{v} + \mathbf{h}\rangle/p$. This holds since $\langle \mathbf{x}', \mathbf{h}\rangle/p$ is distributed like a normal variable with standard deviation $\alpha/\sqrt{2\pi}$.

Lemma 3.7 tells us that the statistical distance between $\mathbf{v} + \mathbf{h}$ and $\nu_{\sqrt{1+(p\alpha/\|\mathbf{x}'\|)^2}}$ is negligible. Since statistical distance cannot increase by applying a function, we obtain that the statistical distance between $\langle \mathbf{x}', \mathbf{v} + \mathbf{h}\rangle/p$ and a normal variable with mean 0 and standard deviation

$$\begin{aligned}\frac{1}{\sqrt{2\pi}} \cdot \sqrt{1 + (p\alpha/\|\mathbf{x}'\|)^2} \cdot \|\mathbf{x}'\|/p &= \frac{1}{\sqrt{2\pi}} \cdot \sqrt{(\|\mathbf{x}'\|/p)^2 + \alpha^2}\\ &\leq \frac{1}{\sqrt{2\pi}} \cdot \sqrt{2}\alpha\end{aligned}$$

is negligible. $\square$

The proof of the following lemma is omitted.

Lemma 3.7. *Let L be a lattice, let $\mathbf{u} \in \mathbb{R}^n$ be any vector, let $s, t > 0$ be two reals, and let r denote $\sqrt{s^2 + t^2}$. Assume that $\varepsilon := \rho_{r/st}(L^* \setminus \{\mathbf{0}\})$ satisfies $\varepsilon < \frac{1}{2}$. Consider the* *continuous distribution Y on $\mathbb{R}^n$ obtained by sampling from $D_{L+\mathbf{u},t}$ and then adding a noise vector taken from ν_s. Then, the statistical distance between Y and ν_r is at most 4ε.*

4. PUBLIC KEY CRYPTOSYSTEM

We let n be the security parameter of the cryptosystem. Our cryptosystem is parameterized by two integers m, p and a probability distribution χ on $\mathbb{Z}_p$. A setting of these parameters that guarantees both security and correctness is the following. Choose $m = 5n$ and set $p \geq 2$ to be some prime number between n^2 and $2n^2$. The probability distribution χ is taken to be $\bar{\Psi}_{\alpha(n)}$ for $\alpha(n) = o(1/\sqrt{n \log n})$, i.e., $\alpha(n)$ is such that $\lim_{n\to\infty} \alpha(n)/\sqrt{n \log n} = 0$. For example, we can choose $\alpha(n) = 1/(\sqrt{n} \log n)$. In the following description, all additions are performed in $\mathbb{Z}_p$, i.e., modulo p.

- **Private key:** Choose $\mathbf{s} \in \mathbb{Z}_p^n$ uniformly at random. The private key is $\mathbf{s}$.
- **Public Key:** For $i = 1, \ldots, m$, choose m vectors $\mathbf{a_1}, \ldots, \mathbf{a_m} \in \mathbb{Z}_p^n$ independently from the uniform distribution. Also choose elements $e_1, \ldots, e_m \in \mathbb{Z}_p$ independently according to χ. The public key is given by $(\mathbf{a_i}, b_i)_{i=1}^m$ where $b_i = \langle \mathbf{a_i}, \mathbf{s} \rangle + e_i$.
- **Encryption:** In order to encrypt a bit we choose a random subset S of $[m]$. The encryption is $(\sum_{i\in S} \mathbf{a_i}, \sum_{i\in S} b_i)$ if the bit is 0 and $(\sum_{i\in S} \mathbf{a_i}, \lfloor \frac{p}{2} \rfloor + \sum_{i\in S} b_i)$ if the bit is 1.
- **Decryption:** The decryption of a pair $(\mathbf{a}, b)$ is 0 if $b - \langle \mathbf{a}, \mathbf{s} \rangle$ is closer to 0 than to $\lfloor \frac{p}{2} \rfloor$ modulo p. Otherwise, the decryption is 1.

Notice that with our choice of parameters, the public key size is $O(mn \log p) = \tilde{O}(n^2)$ and the encryption process increases the size of a message by a factor of $O(n \log p) = \tilde{O}(n)$. In fact, it is possible to reduce the size of the public key to $O(m \log p) = \tilde{O}(n)$ by the following idea of Ajtai [3]. Assume all users of the cryptosystem share some fixed random choice of $\mathbf{a_1}, \ldots, \mathbf{a_m}$. This can be achieved by, say, distributing these vectors as part of the encryption and decryption software. Then the public key need only consist of $b_1, \ldots, b_m$. This modification does not affect the security of the cryptosystem.

For certain choices of χ, m, and p (such as the one mentioned above), it can be shown that the probability of decryption error is small and that the cryptosystem is secure. We omit further details.

5. ACKNOWLEDGMENTS

I would like to thank Michael Langberg, Miklos Santha, and Madhu Sudan for useful comments.

6. REFERENCES

[1] D. Aharonov and O. Regev. Lattice problems in NP intersect coNP. In *Proc. 45th Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 362–371, 2004.

[2] M. Ajtai. Generating hard instances of lattice problems. In *ECCCTR: Electronic Colloquium on Computational Complexity, technical reports*, 1996.

[3] M. Ajtai. Representing hard lattices with $O(n \log n)$ bits. In *Proc. 37th Annual ACM Symp. on Theory of Computing (STOC)*, 2005.

[4] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proc. 29th Annual ACM Symp. on Theory of Computing (STOC)*, pages 284–293, 1997.

[5] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. 33rd ACM Symp. on Theory of Computing*, pages 601–610, 2001.

[6] M. Alekhnovich. More on average case vs approximation complexity. In *Proc. 44th Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 298–307, 2003.

[7] A. Blum, M. Furst, M. Kearns, and R. J. Lipton. Cryptographic primitives based on hard learning problems. In *Advances in cryptology—CRYPTO '93 (Santa Barbara, CA, 1993)*, volume 773 of *Lecture Notes in Comput. Sci.*, pages 278–291. Springer, Berlin, 1994.

[8] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM*, 50(4):506–519, 2003.

[9] J.-Y. Cai and A. Nerurkar. An improved worst-case to average-case connection for lattice problems. In *Proc. 38th Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 468–477, 1997.

[10] W. Ebeling. *Lattices and codes*. Advanced Lectures in Mathematics. Friedr. Vieweg & Sohn, Braunschweig, revised edition, 2002. A course partially based on lectures by F. Hirzebruch.

[11] U. Feige. Relations between average case complexity and approximation complexity. In *Proc. 34th Annual ACM Symp. on Theory of Computing (STOC)*, pages 534–543, 2002.

[12] R. Kumar and D. Sivakumar. On polynomial approximation to the shortest lattice vector length. In *Proc. 12th Annual ACM-SIAM Symp. on Discrete Algorithms*, pages 126–127, 2001.

[13] D. Micciancio. Improved cryptographic hash functions with worst-case/average-case connection. In *Proc. 34th Annual ACM Symp. on Theory of Computing (STOC)*, pages 609–618, 2002.

[14] D. Micciancio. Almost perfect lattices, the covering radius problem, and applications to Ajtai's connection factor. *SIAM Journal on Computing*, 2004. Accepted for publication. Available from author's web page at URL http://www.cse.ucsd.edu/users/daniele.

[15] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: A Cryptographic Perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, Mar. 2002.

[16] D. Micciancio and O. Regev. Worst-case to average-case reductions based on Gaussian measures. In *Proc. 45th Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, 2004.

[17] O. Regev. New lattice based cryptographic constructions. In *Proc. 35th Annual ACM Symp. on Theory of Computing (STOC)*, pages 407–416, 2003.