Learning with Errors and Lattice-Based Post-Quantum Public-Key Cryptography

Mikko Jaskari

Master's Thesis in Technology
December 2025

DEPARTMENT OF MATHEMATICS AND STATISTICS

Reviewers:
Docent Mika Hirvensalo
Docent Ville Junnila
Professor Ion Petre

UNIVERSITY OF TURKU
Department of Mathematics and Statistics

Mikko Jaskari: Learning with Errors and Lattice-Based Post-Quantum Public-Key Cryptography
Master's Thesis in Technology, 39 pages, 4 appendix pages
Cryptography
December 2025

The security of conventional public-key cryptography schemes is based on mathematical problems which are considered to be infeasible to solve by classical computers in a reasonable amount of time. However, the rise of quantum computers might break this security in the future.

Post-quantum cryptography (PQC) aims to achieve secure schemes which are resistant against attacks assisted by quantum computers. Public-key cryptography refers to asymmetric cryptography in which encryption is done with a public key and decryption is done with a private key. Characteristic properties of public-key cryptography include the possibility to derive the private key from public information by solving some presumably hard problem. The hardness of the problem constitutes the basis for the system's security. In the post-quantum era, the security must be based on problems that are too hard to solve even for quantum computers.

In this thesis, we will study lattice-based cryptography which is believed to serve as a good candidate for secure post-quantum public-key cryptography. Our main attention is in the problem of learning with errors, which is in the base of some lattice-based schemes and is linked to the hard problem of finding short vectors in a lattice.

Keywords: learning with errors, lattice-based cryptography, post-quantum cryptography, public-key cryptography, asymmetric cryptography.

LIST OF CERTAIN NOTATIONS

- $\mathbb{N}$ is the set of natural numbers with 0 excluded.
- $\mathbb{P}$ is the set of prime numbers.
- $\mathbb{Z}$ is the set of integers.
- $\mathbb{Z}_q$ is the quotient group $\mathbb{Z}/q\mathbb{Z}$. If q is a prime number, then this is a q-ary field. The distance in $\mathbb{Z}_q$ is defined as a function $|\cdot| : \mathbb{Z}_q \to \mathbb{Z} \cap [0, \lfloor q/2 \rfloor]$, where $|a|$ is the distance of a to 0 in $\mathbb{Z}_q$, assuming that the following order of elements $-\lfloor q/2 \rfloor, \ldots, -2, -1, 0, 1, 2, \ldots, \lfloor q/2 \rfloor$ in $\mathbb{Z}_q$ holds.
- $\mathbb{Q}$ is the set of rational numbers.
- $\mathbb{R}$ is the set of real numbers.
- $\mathbb{C}$ is the set of complex numbers.
- $\mathbb{T}$ is the quotient group $\mathbb{R}/\mathbb{Z}$.
- $\mathbb{E}$ is the expected value.
- $\gcd(m, n)$ is the greatest common divisor of m and n.
- $\phi : \mathbb{N} \to \mathbb{N}$ is the Euler's totient function, where $\phi(n)$ is the amount of positive integers m with properties $m < n$ and $\gcd(m, n) = 1$.
- The notation $f(x) = O(g(x))$ means that there exist a real x_0 and a positive constant C such that $0 \leq f(x) \leq Cg(x)$ holds for all $x \geq x_0$.
- The notation $f(x) = \Omega(g(x))$ means that $g(x) = O(f(x))$ holds.
- The notation $f(x) = o(g(x))$ means that $\lim_{x\to\infty} f(x)/g(x) = 0$.
- The notation $f(x) = \tilde{O}(g(x))$ means that $f(x) = O(\log^k(g(x))g(x))$ holds for any constant k.
- $\text{poly}(n)$ represents some possibly unknown polynomial in n.
- $\lfloor \cdot \rfloor$ is the floor function, that rounds the given real number down to the nearest integer.
- $\lceil \cdot \rceil$ is the ceil function, that rounds the given real number up to the nearest integer.
- $\lfloor \cdot \rceil$ is the rounding function, that rounds the given real number to the nearest integer.
- NP stands for the nondeterministic polynomial time complexity class of decision problems.
- P as complexity class stands for class of decision problems solvable in polynomial time.

Contents

Introduction

Quantum computing is a form of computing in which quantum mechanics is used to gain an advantage over classical computing, an idea raised by R. P. Feynman in the 1980's [21] and elaborated further by D. Deutsch in [17]. As classical computers use bits with two possible values '0' and '1', quantum computers use qubits, which can use superposition, allowing two possible values at the same time. This theoretically allows quantum computers to do some calculations exponentially faster than classical computers. However, the trick with superpositions has its difficulties, as the superposition will collapse if we measure the value of the qubit. This means that the measured value of the qubit is one of the possible values based on some probability distribution. In other words, to achieve the advantage over classical computers, we are, in quantum mechanical terms, not allowed to "*observe*" the qubits in the middle of the computation. An example of a problem which can be solved efficiently by a quantum computer was given in 1992 by D. Deutsch and R. Jozsa in [18]. However, since measuring the answer forces the superpositions of the qubits to collapse, the measured outcome in practical problems is usually just a clue or evidence of the correct answer. This means that we might need to repeat the computation multiple times, and all of those possibly different outcomes reveal something about the correct answer, which allows us to be more certain about it. If the correct answer on the other hand is easily verifiable with total certainty, then this is usually sufficient, and this is the case with factoring integers, for example.

Public-key cryptography refers to asymmetric cryptography in which we have a public key for encryption and a private key for decryption. This kind of cryptography is essential for situations in which two communicating parties have not shared any secrets in a secure environment beforehand, and their only option is to communicate over insecure channels. Usually it is convenient to use public-key cryptosystem in the beginning of the communication to agree upon a shared secret key, which can then be used for some other cryptosystem, which allows faster and more secure communication. The nature of public-key cryptography is that the private key is technically derivable just by using the public information, but we assume that such derivations are infeasible for any reasonable amount of time. These assumptions are based on some mathematical problems for which we lack algorithms that could solve those problems reasonably fast. By reasonably fast we usually mean polynomial time, which means that time required for the solution is a polynomial of the size of the input. However, some of the popular public-key encryption schemes like `RSA` by R. L. Rivest, A. Shamir and L. Adleman in 1978 [54], `ElGamal` by T. Elgamal in 1985 [20] based on the `Diffie-Hellman key exchange` by W. Diffie and M. Hellman in 1976 [19] and elliptic-curve related schemes initially proposed by V. S. Miller [44] and N. Koblitz [36] in the 1980's rely on mathematical problems, which are assumed to be hard for classical computers, but as P. W. Shor in 1994 showed [57], they are probably easy for quantum computers. For example, `RSA` relies on the complexity of factoring integers, `ElGamal` on discrete logarithm problem and elliptic-curve related schemes on elliptic curve discrete logarithm problem. If quantum computers become powerful enough, then they pose a serious security risk, which will require new ways to implement public-key cryptography, which is called *post-quantum public-*

key cryptography. Even though quantum computers might not be efficient enough today, adversary parties might already capture and store encrypted data and decrypt it, should it become possible in the future. The mathematical problems behind post-quantum public-key cryptography schemes have to be in some sense *noisy* so that even verifying the correct result has some uncertainties.

In this thesis we will first discuss about the popular public-key cryptography scheme `RSA` and how Shor's algorithm on quantum computers might break it as an example to address the concerns related to usual public-key cryptography. Then we will dive into one of the promising approaches towards post-quantum cryptography, lattice-based cryptography, and scheme which is based on the mathematical problem of learning with errors LWE, introduced by O. Regev in 2005 [52]. The purpose is to gain insight which is good enough for us to understand the structure of LWE and lattice problems and which improves our trust in the security. After this we check other lattice-based public-key cryptosystems and transition towards post-quantum era. We will also present example `Python` code in the appendix to demonstrate the functionality of the LWE-based cryptosystem. Our engineering perspective regards presenting plausible explanations more important than rigorous and heavy proofs presented in the original research. The reader is assumed to have a comprehensive understanding of algebra, number theory and algorithm design. Some notations of quantum computing are presented, but it is also recommendable that one familiarizes oneself with the topic of quantum computing.

1 Briefly about quantum computing

In this section, we take a look at the key principles of quantum computing so that we develop a sufficient understanding of the topic for the purpose of this thesis. We follow the paper of D. R. Simon from 1994 [58].

Quantum computers use quantum bits, also known as qubits, which allows the state of quantum computer to be in a superposition until it is observed. We may denote this superposition as a sum of all possible states multiplied by their amplitude:

$$\sum_j \alpha_j |c_j\rangle.$$

Here, $|c_j\rangle$ is one of the possible states and the complex number α_j is the amplitude, which corresponds to the probability of that state. When we make a measurement and collapse the superposition, the probability of observing a particular state $|c_k\rangle$ is $|\alpha_k|^2$. Hence,

$$\sum_j |\alpha_j|^2 = 1$$

must hold.

The idea behind quantum computing is to manipulate these superpositions and then make a measurement. Superposition allows processing all the possible states at once, but the measurement in the end will collapse it. Hence, we need some clever ways to manipulate these superpositions and their probability distributions in our favor before observing them, and those observations should reveal some useful information about the result we are looking for.

The operations can be presented as operations in linear algebra. For example, a single qubit in superposition

$$\alpha_1|0\rangle + \alpha_2|1\rangle$$

can be presented in column vector form;

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix}.$$

The usual way to describe logical ports of the quantum computers is using unitary matrices. For example, a matrix representing the *Hadamard gate*

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

corresponds to a 'fair coin flip'. We take a state $|0\rangle$, which in the column vector form is $\binom{1}{0}$, and use this transformation to obtain

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix},$$

which is the superposition $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. A state $|1\rangle$, which in the column vector form is $\binom{0}{1}$, would be transformed into superposition $\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$. In both of these superpositions, observing states $|0\rangle$ and $|1\rangle$ has the equal probability 1/2. We also note that applying our transformation for the second time cancels the fair coin flip.

2 RSA and Shor's algorithm

In 1977 (published in 1978) R. L. Rivest, A. Shamir and L. Adleman presented one of the most common and widely used public-key cryptosystem `RSA` [54]. The security of the system is based on the assumption that factoring semiprimes is a too difficult task for conventional computers. A semiprime is an integer that is a product of two prime numbers. The cryptosystem works as follows.

Setting up the cryptosystem:

(a) Select two large prime numbers p, q and compute $N = pq$ and keep p and q secret.

(b) Compute $\phi(N) = (p-1)(q-1)$ and keep this secret. ϕ is the Euler's totient function.

(c) Select $1 < e < \phi(N)$ so that $\gcd(e, \phi(N)) = 1$.

(d) Compute d so that $de \equiv 1 \pmod{\phi(N)}$ e.g. by Euclidean algorithm and keep this as a secret private key.

(e) Publish e and N.

Encryption and decryption:

- The message $1 < m < N$ is encrypted by computing $m^e \pmod{N}$.
- The encrypted message is decrypted by computing $(m^e)^d \equiv m \pmod{N}$.

The decryption is successful due to the well-known Euler's theorem which states that for any m with $\gcd(m, N) = 1$ we have

$$m^{\phi(N)} \equiv 1 \pmod{N}.$$

If one is to break the cryptosystem, they could do it by factoring N and thus reveal p and q. This would allow the adversary party to compute $\phi(N)$ and d. However, if p and q are large enough, the complexity of factoring grows significantly and the task becomes infeasible for conventional computers. The encryption part also quietly assumes this. If one tries to encrypt m for which $\gcd(m, N) > 1$, then the Euler's theorem is not applicable but on the other hand in such scenario $\gcd(m, n) = p$ or $\gcd(m, n) = q$ and this would break the security of the system. Determining the greatest common divisor is relatively fast and can be done by e.g. Euclidean algorithm.

In 1994 P. W. Shor introduced in [57] a worrying quantum computer algorithm that could speed up the factoring and thus pose a threat to the security of `RSA`. The factoring is based on the following idea.

- Take a random integer $x \in (1, N)$ and determine its order r, which is the smallest positive integer for which $x^r \equiv 1 \pmod{N}$ holds.

- If r is even, then $(x^{r/2}-1)(x^{r/2}+1) \equiv 0 \pmod{N}$ and if $x^{r/2} \not\equiv -1 \pmod{N}$, then for the semiprime N case $\gcd(x^{r/2}-1, N)$ is either p or q.

Now, the hardest part is to determine the order r, which is where quantum computing comes in.

The first part is to determine a smooth $q \in [2N^2, 4N^2]$, which does not have any prime factors larger than $(\log q)^h$ for some fixed h. This smoothness condition is needed to perform the latter described quantum Fourier transform in polynomial time. According to [57, Lemma 3.2], finding the desired smooth q is possible in polynomial time. Next, we put the quantum computer in the superposition of states representing the least residue system modulo q, which is

$$\frac{1}{q^{1/2}} \sum_{a=0}^{q-1} |a\rangle$$

and use this to obtain a superposition

$$\frac{1}{q^{1/2}} \sum_{a=0}^{q-1} |a, x^a \pmod{N}\rangle.$$

We then perform a quantum Fourier transform, which maps our superposition into another superposition, which is

$$\frac{1}{q} \sum_{a=0}^{q-1} \sum_{c=0}^{q-1} \exp(2\pi i ac/q) |c, x^a \pmod{N}\rangle.$$

Now, we proceed with evaluating the probability of observing a particular state $|c, x^k \pmod{N}\rangle$ where $0 \leq k < r$. This probability is obtained by summing over all the possible ways to reach that state. We obtain

$$\left| \frac{1}{q} \sum_{\substack{0 \leq a \leq q-1 \\ x^a \equiv x^k \pmod{N}}} \exp(2\pi i ac/q) \right|^2 .$$

As Shor derives, this probability is eventually at least $1/(3r^2)$ if there exists an integer d such that

$$\left| \frac{c}{q} - \frac{d}{r} \right| \leq \frac{1}{2q}$$

holds. Because $q > N^2$, there can be only one fraction d/r with $r < N$, which satisfies the above inequality. Thus, we just have to find the nearest fraction of c/q with a denominator smaller than N. This can be done in polynomial time by using the continued fraction expansion of c/q. If $\gcd(d, r) = 1$, then we get r directly from the denominator. The number of potential candidates for d with $\gcd(d, r) = 1$ is $\phi(r)$. Finally, Shor concludes that the probability of finding r is at least $\phi(r)/(3r)$, that is at least $\delta / \log\log r$ for some positive constant δ. Thus, we have a high probability of success with $O(\log\log r)$ attempts. Overall, this algorithm should do the factoring in polynomial time, whereas classical computers can do it with the general version of J. Pollard's number field sieve in sub-exponential time at best, as discussed in C. Pomerance's work, see [51].

3 Lattices

Shor's study demonstrates that certain types of public-key cryptosystems are threatened by quantum computers. What is common with factoring integers and discrete logarithm problems is that the given correct answer is easily verifiable. In other words we could compare this situation to a locked door and a collection of keys which is publicly available for everyone. If you try the right key, you can open the door immediately. If a quantum computer finds a pattern that gives a high probability of good guesses, then these problems become much easier. This forces us to think of adding a layer of uncertainty to our mathematical problems. One promising approach is *lattices*.

Definition 3.1. Let $n \in \mathbb{N}$ and $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\}$ be a collection of linearly independent vectors in $\mathbb{R}^n$. A set

$$L = \left\{ \sum_{k=1}^{n} a_k \mathbf{b}_k : a_k \in \mathbb{Z} \right\}$$

is called a n-dimensional lattice generated by basis $\mathbf{B}$.

Hence, lattice is a set of all linear combinations of the basis vectors and trivially also an additive subgroup of $\mathbb{R}^n$.

Definition 3.2. A n-dimensional q-ary lattice, in case q is a prime, is obtained by replacing $\mathbb{R}^n$ and $\mathbb{Z}$ in Definition 3.1 by $\mathbb{Z}_q^n$ and $\mathbb{Z}_q$ respectively.

3.1 Shortest vector problem

In this subsection we will look at lattice problems related to shortest vector.

Definition 3.3. Given a lattice L and a norm $\|\cdot\|$, $\lambda(L)$ is the shortest possible length in the norm $\|\cdot\|$ for any non-zero vector in the lattice L. More precisely,

$$\lambda(L) := \min_{\mathbf{v} \in L \setminus \{\mathbf{0}\}} \|\mathbf{v}\|.$$

Moreover, the k^{th}-successive minimum $\lambda_k(L)$ is the smallest possible radius r for an origin-centered ball that contains k linearly independent vectors in L.

Note that by Definition 3.3 $\lambda(L) = \lambda_1(L)$ holds.

Shortest vector problem SVP. SVP in a lattice L with a given norm $\|\cdot\|$ is about finding a vector $\mathbf{v} \in L$ for which $\lambda(L) = \|\mathbf{v}\|$ holds. The approximate version SVP_γ for $\gamma \geq 1$ is about finding $\mathbf{v} \in L \setminus \{\mathbf{0}\}$ such that $\|\mathbf{v}\| \leq \gamma\lambda(L)$ holds.

Gap shortest vector problem GapSVP. $\text{GapSVP}_{d,\gamma}$ for $\gamma = \gamma(n) \geq 1$ and $d > 0$ in a n-dimensional lattice L with a given norm $\|\cdot\|$ is a promise problem in which the algorithm must decide whether $\lambda(L) \leq d$ or $\lambda(L) > d\gamma$ holds. If neither is true, then the algorithm is allowed to give an erroneous answer.

Shortest independent vectors problem SIVP. SIVP_γ for $\gamma = \gamma(n) \geq 1$ in a n-dimensional lattice L with a given norm $\|\cdot\|$ is about finding n linearly independent vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ in L so that

$$\max_{k=1,2,\ldots,n} \|\mathbf{v}_k\| \leq \gamma \lambda_n(L)$$

holds.

Closest vector problem CVP. Given a point $\mathbf{x}$ in vector space, CVP asks to find a vector $\mathbf{y}$ in a lattice L so that

$$\|\mathbf{x} - \mathbf{y}\| = \min_{\mathbf{v} \in L} \|\mathbf{x} - \mathbf{v}\|$$

holds. CVP has similar variations like SVP.

Note. From this point on, the length of the vector is measured by default in the Euclidean norm.

These presented lattice problems are bit more complicated than factoring integers or discrete logarithm problem. For example, if one has the solution for SVP, verifying it is not so trivial. We do not have easy calculations like multiplication or modular exponentiation to check the correctness. One needs information of other vectors in the lattice to make that call. This is one promising step towards adding some layer of uncertainty as we wanted.

3.2 Hardness results

In order to evaluate the suitability of SVP and its variations for cryptographic purposes, we present some hardness results. D. Micciancio discusses the hardness of SVP in [42]. As for the 2-dimensional lattice the problem is easy, for higher n-dimensional lattices L we lack efficient algorithms that would even compute $\lambda(L)$ in polynomial time. The hardness of the problem seems to be affected by the given basis of the lattice, which means that lattice basis reductions might ease the problem. The Lenstra–Lenstra–Lovász lattice basis reduction algorithm LLL introduced by A. K. Lenstra, H. W. Lenstra Jr. and L. Lovász in 1982 [38] solves the approximate SVP_γ in polynomial time for $\gamma(n) = 2^{O(n)}$. Subsequent works by C. P. Schnorr [56] and M. Ajtai, R. Kumar and D. Sivakumar [6] have then improved the approximate factor in polynomial time algorithms to be $\gamma(n) = 2^{n \log\log n / \log n}$. Solving SVP exactly has the time complexity of $2^{O(n)}$ as the works of R. Kannan in 1983 [34], U. Fincke and M. Pohst in 1985 [22] and N. Gama, P. Q. Nguyen and O. Regev in 2010 [23] indicate.

In the 1990's M. Ajtai provided some groundbreaking results which support the hardness assumption of SVP. In [3] it was shown that there are worst-case to average-case reductions for some lattice problems and in [5] that SVP is NP-hard in Euclidean norm for randomized reductions. The latter result was achieved

by showing that there is a probabilistic Turing machine which in polynomial time reduces any problem in NP to instances of SVP. In the introduction of a recent paper from 2023 H. Bennett and C. Peikert [9] wraps up subsequent work related to hardness of GapSVP. We have that using randomized reductions $\text{GapSVP}_{d,\gamma}$ is NP-hard to approximate for any constant $\gamma \geq 1$, and hard for nearly polynomial factors $\gamma(n) = n^{\Omega(1/\log\log n)}$ assuming stronger complexity assumptions. However, they also note that it is unlikely that $\text{GapSVP}_{d,\gamma}$ is NP-hard for approximation factors $\gamma \geq C\sqrt{n}$ where C is a constant and that even larger, but typically polynomial, factors are used in lattice-based cryptography.

Although studies support the idea that SVP and GapSVP are hard, we are still unsure if they are NP-hard in the first place. So far best we have achieved are the results in randomized reductions, but deterministic reductions are still an open problem.

3.3 Ajtai-Dwork cryptosystem

We present a public-key cryptosystem by M. Ajtai and C. Dwork from 1997 [4] based on the Ajtai's previous work [3]. The security of the system is related to the unique shortest vector problem.

Unique shortest vector problem u-SVP. u-SVP in a n-dimensional lattice L is about finding the shortest vector $\mathbf{v} \in L \setminus \{\mathbf{0}\}$ so that $\mathbf{v}$ is unique in a sense that for given c any other vector in L whose length is at most $n^c\|\mathbf{v}\|$ is parallel to $\mathbf{v}$.

Definition 3.4. (d, M)**-Lattices.** Assume $M > 0$, $d > 0$, $L \subseteq \mathbb{Z}^n$ is a n-dimensional lattice and that L' is $(n-1)$-dimensional sublattice of L with the following properties:

(a) every vector in a basis of L' has a length at most M;

(b) if H is the $(n-1)$-dimensional subspace of $\mathbb{R}^n$ containing L' and $H' \neq H$ is a coset of H intersecting L, then the distance of H and H' is at least d;

then L is a (d, M)-lattice. The minimum distance between H and a coset of H intersecting L will be denoted by d_L.

We note that if $d > M$, then L' is unique and in this case will be denoted by $L^{(d,M)}$.

The key pair generation procedure:

(a) Generate a random $(n-1)$-dimensional lattice L' having a basis $(\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{n-1})$ and determine such $M > 0$ that $\|\mathbf{b}_k\| \leq M$ holds for every $k = 1, 2, \ldots, n-1$.

(b) Choose $d \geq n^c M$.

(c) Choose from a large cube, which depends on a choice of scheme, a random vector $\mathbf{b}_n$ of distance $d \leq d_L \leq 2d$ from H.

(d) The private key is any basis for $L^{(d,M)}$.

(e) Generate L by basis $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\}$ and, in order to keep the basis for $L^{(d,M)}$ hidden, construct another basis $\mathbf{B}'$ for L. The public key is $(\mathbf{B}', M)$.

Encryption and decryption:

The encryption and decryption procedure is based on the idea that the one holding the private key is able to distinguish between a random sum of vectors in L and a purely random point in $\mathbb{R}^n$ by using hyperplanes, which are the cosets of H.

Definition 3.5. Perturbation. For basis $\mathbf{B}'$, $R \in \mathbb{R}$ and $m \in \mathbb{Z}$, let the perturbation $\text{pert}(\mathbf{B}', R, m)$ be the random variable whose value is the sum of m vectors in $\mathbf{B}'$ taken independently and with uniform distribution from the origin centered n-dimensional ball with radius R.

Definition 3.6. The n-dimensional unit cube U^n is defined by

$$U^n = \{\mathbf{v} = (v_1, v_2, \ldots, v_n) \in \mathbb{R}^n : -1/2 \leq v_k \leq 1/2, \forall k = 1, 2, \ldots, n\}.$$

Encryption is done bitwise:

- To encrypt '0', choose a random lattice point $\mathbf{v}$ in the cube KU^n, where U^n is the n-dimensional unit cube and $K \geq 2^n d$. For $m = c_0 n$, $c_0 \geq 4$, and $R = n^3 M$, choose a value $\mathbf{w}$ of $\text{pert}(\mathbf{B}', R, m)$. The ciphertext is $\mathbf{v} + \mathbf{w}$.
- To encrypt '1', choose uniformly at random a point $\mathbf{v}$ in the cube KU^n. The point $\mathbf{v}$ is the ciphertext.

Decryption is done in the following way. Let $\mathbf{u}_H$ be a unit vector orthogonal to the subspace H, and let d_L be the distance between consecutive hyperplanes. To decrypt the ciphertext $\mathbf{z}$, the receiver computes the fractional part of $(\mathbf{u}_H \cdot \mathbf{z})/d_L$. If it is within mR/d_L of 0 or 1, then $\mathbf{z}$ is decrypted as '0', and as '1' otherwise.

As Ajtai and Dwork point out, the security of this system is based on the *hidden hyperplane* assumption which says that computing $L^{(d,M)}$ from a random basis of (d, M)-lattice L is infeasible. This assumption is on the other hand related to the hardness of u-SVP. If Λ is a lattice with an n^c-unique shortest vector $\mathbf{u}$, then the dual lattice of Λ,

$$L = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{x}^T \mathbf{y} \in \mathbb{Z}, \forall \mathbf{y} \in \Lambda\},$$

is a (d, M)-lattice for some $d \geq n^{c'} M$, $c' \approx c - 2$ and $d_L = \|\mathbf{u}\|^{-1}$. If $\mathbf{B}_\Lambda = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n\}$ is the basis of Λ, then the basis of L, $\mathbf{B}_L = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n\}$, can be computed by solving

$$\mathbf{c}_i^T \mathbf{b}_i = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases}$$

We note that $\mathbf{u}$ is orthogonal to H. Thus, solving u-SVP for n^c-unique case in Λ reveals $\mathbf{u}_H$ and d_L, which are needed in the decryption. For the opposite direction if

one knows the subspace H and uses the fact that $\mathbf{u}$ is orthogonal to it, it is sufficient to reveal the direction of $\mathbf{u}$. Then, by computing the gcd of the distances to H of random points in L, d_L can be computed in probabilistic polynomial time, yielding $\|\mathbf{u}\|^{-1}$, which reveals $\mathbf{u}$. Hence, knowing H is sufficient for polynomial time solution of u-SVP for n^c-unique case in Λ.

However, this version of the encryption leaves room for errors. The authors note that there is a small but polynomial probability that encrypted '1' will be decrypted as '0', because the encryption of '1' is basically purely random. This can be seen as a consequence of our attempt to add some uncertainty to our problems on which the security is based.

In 1997 O. Goldreich, S. Goldwasser and S. Halevi published some modifications to the scheme in order to eliminate these errors [26]. In their modified version, the main differences are the slight modifications in public key, encryption of '1' and decryption. We will now briefly explain the differences. Note that the description of the scheme in [26] is somewhat different compared to the original [4] and the details here are traced back to match the original expression. The idea of eliminating cases of wrongfully decrypted '1' is based on noting that if for correctly decrypted '0' the fractional part of $(\mathbf{u}_H \cdot \mathbf{z})/d_L$ is always very close to 0 or 1, then we could change the encryption of '1' in a way that the fractional part is always very close to 1/2.

In the public key section a new parameter is added, which is a vector $\mathbf{a}$ for which $\mathbf{u}_H \cdot \mathbf{a} = (2k+1)d_L$ holds for some $k \in \mathbb{Z}$. Now '1' is encrypted in the same way as '0' is encrypted, but with the difference that we add $\frac{1}{2}\mathbf{a}$ to the ciphertext. Then the decryption is done in the similar way as before by calculating $(\mathbf{u}_H \cdot \mathbf{z})/d_L$ but concluding that if the fractional part is within 1/4 of 0 or 1, then $\mathbf{z}$ is decrypted as '0', and as '1' otherwise. The authors show that now there is no more room for error and that the security of this modified system is based on the same assumptions as the original.

C. Peikert noted in 2016 [50] that at the time Ajtai-Dwork cryptosystem was a theoretical breaktrough and that all lattice-based encryption schemes inherit the basic template of this system. However, from a practical point of view this system is not necessarily great in terms of efficiency. Public keys are of size $\tilde{O}(n^4)$ and private keys and ciphertexts are of size $\tilde{O}(n^2)$, with matching runtimes for encryption and decryption, respectively. P. Nguyen and J. Stern also studied the cryptanalysis of Ajtai-Dwork cryptosystem in 1998 [47] and concluded that in order to prevent key-recovery attacks the value of n must be in the hundreds, which would mean that public keys are measured in gigabits and ciphertexts in several megabits if the message is even in some way meaningful. This highlights that Ajtai-Dwork encryption is important part of the story of lattice-based cryptography even though it is not practically useful in applications.

4 Learning with errors

We will now study the learning with errors problem LWE and a public-key cryptosystem related to it as introduced by O. Regev in 2005 [52]. We follow the updated version of the paper from 2009 [53]. We can motivate this in the spirit of our goal of adding uncertainty in order to improve security against attacks by quantum computing. Let $\mathbf{s} \in \mathbb{Z}_p^n$ be the vector we want to determine. The information we have is p, vectors $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m$ and integers $b_1, b_2, \ldots, b_m$ satisfying

$$\begin{cases} \mathbf{a}_1 \cdot \mathbf{s} \equiv b_1 \pmod{p} \\ \mathbf{a}_2 \cdot \mathbf{s} \equiv b_2 \pmod{p} \\ \vdots \\ \mathbf{a}_m \cdot \mathbf{s} \equiv b_m \pmod{p}. \end{cases} \tag{4.1}$$

Now assuming that p is a prime, $m = n$ and that $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m$ are linearly independent modulo p, then solving $\mathbf{s}$ in polynomial time by e.g. Gaussian elimination with total certainty is easy. However, if one decides to add some errors $e_1, e_2, \ldots, e_m$ and instead of $b_1, b_2, \ldots, b_m$ in (4.1) we are given the erroneous version $b_1', b_2', \ldots, b_m'$ satisfying

$$\begin{cases} \mathbf{a}_1 \cdot \mathbf{s} \equiv b_1 + e_1 \equiv b_1' \pmod{p} \\ \mathbf{a}_2 \cdot \mathbf{s} \equiv b_2 + e_2 \equiv b_2' \pmod{p} \\ \vdots \\ \mathbf{a}_m \cdot \mathbf{s} \equiv b_m + e_m \equiv b_m' \pmod{p}, \end{cases} \tag{4.2}$$

then determining $\mathbf{s}$ becomes much harder and more uncertain. This is due to the facts that the hardness of this task has connections to hard lattice problems and it has statistical structures, as we will later see. Regev states that under reasonable assumptions, the likely successful recovering of $\mathbf{s}$ for $p \leq \text{poly}(n)$ and using $\text{poly}(n)$ equations has the time complexity of $2^{O(n \log n)}$. A similar problem was studied by A. Blum, A. Kalai and H. Wasserman in 2003 [10].

If the errors are relatively small, then for different linear combinations of the equations from (4.2) the error is presumably also small. Now, if we add relatively large error, say $\lfloor p/2 \rfloor$, to a chosen linear combination of equations, then the one who knows $\mathbf{s}$ should notice this unusually large error and know that something was added. Inferring whether something was added or was not added corresponds to detecting values of bits. This idea is in the heart of LWE-based cryptography.

4.1 LWE and the public-key cryptosystem

In this subsection we formally define LWE and the public-key cryptosystem related to it.

Learning with errors LWE. Let $n, m \in \mathbb{N}$ and $p \leq \text{poly}(n)$ be some integer. Let $\mathbf{s} \in \mathbb{Z}_p^n$ be the secret vector. Vectors $\mathbf{a}_k$ are chosen independently and uniformly from $\mathbb{Z}_p^n$ for every $k = 1, 2, \ldots, m$. Let $\chi : \mathbb{Z}_p \to \mathbb{R}_+$ be a probability distribution on

$\mathbb{Z}_p$. Each e_k, $k = 1, 2, \ldots, m$ is chosen independently according to χ. $\mathrm{LWE}_{p,\chi}$ ask for recovering the secret vector $\mathbf{s}$ from the following set of equations in $\mathbb{Z}_p$,

$$\begin{cases} \mathbf{a}_1 \cdot \mathbf{s} + e_1 = b_1 \\ \mathbf{a}_2 \cdot \mathbf{s} + e_2 = b_2 \\ \vdots \\ \mathbf{a}_m \cdot \mathbf{s} + e_m = b_m, \end{cases}$$

where each error e_k, $k = 1, 2, \ldots, m$ is unknown to the solver.

Definition 4.1. For $\beta \in \mathbb{R}_+$, the distribution Ψ_β is the distribution on $\mathbb{T}$, which is the quotient group $\mathbb{R}/\mathbb{Z}$, obtained by sampling a normal variable with mean 0 and standard deviation $\beta/\sqrt{2\pi}$ and reducing the result modulo 1,

$$\forall r \in [0, 1), \quad \Psi_\beta(r) := \sum_{k=-\infty}^{\infty} \frac{1}{\beta} \exp\left(-\pi \left(\frac{r-k}{\beta}\right)^2\right).$$

Definition 4.2. For an arbitrary probability distribution with density function $\Psi : \mathbb{T} \to \mathbb{R}_+$ and some integer $p \geq 1$ its discretization $\bar{\Psi} : \mathbb{Z}_p \to \mathbb{R}_+$ is

$$\bar{\Psi}(k) := \int_{(k-1/2)/p}^{(k+1/2)/p} \Psi(x) \mathrm{d}x.$$

Next, we will present the public-key cryptosystem. The security parameter is $n \in \mathbb{N}$ and we also use parameters $m \in \mathbb{N}$ and $p \in \mathbb{P}$ chosen so that $n^2 \leq p \leq 2n^2$ and $m = (1 + \varepsilon)(1 + n) \log p$ holds for some $\varepsilon > 0$. The probability distribution χ is taken to be $\bar{\Psi}_{\alpha(n)}$ for $\alpha(n) = o(1/(\sqrt{n} \log n))$, for example $\alpha(n) = 1/(\sqrt{n} \log^2 n)$ is a valid choice.

In the following cryptosystem all additions are performed in $\mathbb{Z}_p$.

Key pair generation:

- Choose $\mathbf{s} \in \mathbb{Z}_p^n$ uniformly at random. This the private key.
- For $k = 1, 2, \ldots, m$, choose m vectors $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m \in \mathbb{Z}_p^n$ independently from the uniform distribution. Choose elements $e_1, e_2, \ldots, e_m \in \mathbb{Z}_p$ independently according to χ. The public key is $(\mathbf{a}_k, b_k)_{k=1}^m$, where $b_k = \mathbf{a}_k \cdot \mathbf{s} + e_k$.

Encyption and decryption:

- Encryption and decryption are both done bitwise.

- Let $\mathcal{M} = \{1, 2, \ldots, m\}$. Choose a random set $\mathcal{S}$ uniformly from the power set $2^{\mathcal{M}}$.
  - The encryption of '0' is the pair $(\sum_{k\in\mathcal{S}} \mathbf{a}_k, \sum_{k\in\mathcal{S}} b_k)$.
  - The encryption of '1' is the pair $(\sum_{k\in\mathcal{S}} \mathbf{a}_k, \lfloor p/2 \rfloor + \sum_{k\in\mathcal{S}} b_k)$.
- To decrypt the received pair $(\mathbf{a}, b)$ one computes $b - \mathbf{a}\cdot\mathbf{s}$. If it is closer to 0 than to $\lfloor p/2 \rfloor$, then the decryption is '0'. Otherwise, the decryption is '1'.

See appendix A for a demonstration of this system in `Python`.

With our choice of parameters, the public key size is $O(mn \log p) = \tilde{O}(n^2)$. The size of the message is increased by a factor of $O(n \log p) = \tilde{O}(n)$.

4.2 Correctness and security

We will now analyze some central structures of LWE. Introducing random errors might, in an unlucky scenario, lead into too large sum of errors, and thus, incorrectly decrypted bits. Our first goal is to study the probability of incorrect decryption.

Definition 4.3. A function $f : \mathbb{N} \to \mathbb{R}$ is negligible if and only if for every $c \in \mathbb{N}$ there exists $N_c \in \mathbb{N}$ such that

$$|f(n)| < \frac{1}{n^c}$$

holds whenever $n > N_c$.

Definition 4.4. Let $\chi : \mathbb{Z}_p \to \mathbb{R}_+$ be a probability distribution on $\mathbb{Z}_p$. Distribution $\chi^{\star k}$ is obtained by summing together k independent samples from χ, where addition is performed in $\mathbb{Z}_p$. For $k = 0$ we define $\chi^{\star 0}$ as the distribution that is constantly 0.

Lemma 4.1. *For the choice of parameters in the presented LWE-based public-key cryptosystem in the section 4.1, it holds that for any $k \in \{0, 1, 2, \ldots, m\}$,*

$$\Pr_{e\sim\bar{\Psi}_\alpha^{\star k}} \left(|e| < \left\lfloor \frac{p}{2} \right\rfloor /2 \right) > 1 - \delta(n)$$

for some negligible function $\delta(n)$.

Proof. A sample from $\bar{\Psi}_\alpha^{\star k}$ can be obtained by sampling $x_1, x_2, \ldots, x_k$ from Ψ_α and outputting $\sum_{j=1}^k \lfloor px_j \rceil \pmod p$. Notice that

$$\left| \sum_{j=1}^{k} (\lfloor px_j \rceil - px_j) \pmod p \right| \leq k \leq m < p/32,$$

since $m = (1+\varepsilon)(1+n)\log p$ holds for some $\varepsilon > 0$. Hence, it is enough to show that $|\sum_{j=1}^k px_j \pmod p| < p/16$ with high probability. This condition is equivalent to the condition that $|\sum_{j=1}^k x_j \pmod 1| < 1/16$. Since $\sum_{j=1}^k x_j \pmod 1$ is distributed as $\Psi_{\alpha\sqrt{k}}$ and $\alpha\sqrt{k} = o(1/\sqrt{\log n})$, the probability that $\sum_{j=1}^k x_j \pmod 1 < 1/16$ is $1 - \delta(n)$ for some negligible function $\delta(n)$. □

Theorem 4.2. *For the choice of parameters in the presented LWE-based public-key cryptosystem in the section 4.1, the probability of erroneous decryption of a bit is at most $\delta(n)$, where $\delta(n)$ is some negligible function.*

Proof. In the decryption stage one computes $b - \mathbf{a} \cdot \mathbf{s}$ and in case of encrypted '0', this is exactly $\sum_{k \in \mathcal{S}} e_k$ and in case of encrypted '1', $\lfloor p/2 \rfloor + \sum_{k \in \mathcal{S}} e_k$. According to Lemma 4.1, $|\sum_{k \in \mathcal{S}} e_k| < \lfloor p/2 \rfloor / 2$ holds with probability at least $1 - \delta(n)$. Hence, $\sum_{k \in \mathcal{S}} e_k$ is closer to 0 than to $\lfloor p/2 \rfloor$ and $\lfloor p/2 \rfloor + \sum_{k \in \mathcal{S}} e_k$ is closer to $\lfloor p/2 \rfloor$ than to 0, as it is necessary for correct decryption, with the probability at least $1 - \delta(n)$. This completes the proof. □

Next, we will study the security of the cryptosystem.

Definition 4.5. 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 the distribution χ, and outputting $(\mathbf{a}, \mathbf{a} \cdot \mathbf{s} + e)$, where additions are performed in $\mathbb{Z}_p$. We also define U as the uniform distribution on $\mathbb{Z}_p^n \times \mathbb{Z}_p$.

It may seem intuitively reasonable, that being able to distinguish $A_{\mathbf{s},\chi}$ from U hints that the distinguisher has solved the secret $\mathbf{s}$, and hence LWE, or at least has some information of it. We will now present without proof a result which is formal statement of this reasoning.

Lemma 4.3. *Let $n \geq 1$ be an integer and $2 \leq p \leq \text{poly}(n)$ be a prime. Let Φ be some probability density function on $\mathbb{T}$ and let $\bar{\Phi}$ be its discretization on $\mathbb{Z}_p$. Assume that we have access to a distinguisher that distinguishes $A_{\mathbf{s},\bar{\Phi}}$ from U for a non-negligible fraction of all possible $\mathbf{s}$. Then, there exists an efficient algorithm that solves $LWE_{p,\Phi}$.*

Proof. See [53, Lemma 4.4] □

The following theorem relates the security of the cryptosystem with LWE by Lemma 4.3. We will present only a simplified version of the proof.

Theorem 4.4. *For any $\varepsilon > 0$ and $m \geq (1+\varepsilon)(1+n)\log p$, if there exists a polynomial time algorithm `W` that distinguishes between encryptions of 0 and 1 then there exists a distinguisher `Z` that distinguishes between $A_{\mathbf{s},\chi}$ and U for a non-negligible fraction of all possible $\mathbf{s}$.*

> Note that in order to make the proof simpler, we will regard the distinguisher `W` to be an algorithm, which is able to distinguish between two distributions with perfect accuracy. In [53] distinguisher is an algorithm, which accepts instances of one distribution notably more likely than instances of the other distribution. By notably more likely we mean that the probabilities of accepting differ by some non-negligible function.

Proof. For the complete proof with more rigorous and accurate version, see [53, Lemma 5.4].

Let $\mathbf{x} := ((\mathbf{a}_k, b_k)_{k=1}^m, (\mathbf{a}, b))$ where $(\mathbf{a}_k, b_k)_{k=1}^m$ is the public key of the cryptosystem and $(\mathbf{a}, b)$ is either an encryption of '0' (case $\mathbf{x} = \mathbf{x}_0$) or encryption of '1' (case $\mathbf{x} = \mathbf{x}_1$). Let $p(\mathbf{x})$ be the probability of `W` accepting $\mathbf{x}$. By our special assumptions mentioned in the note, we assume that `W` always accepts $\mathbf{x}_0$ and always rejects $\mathbf{x}_1$. In other words $p(\mathbf{x}_0) = 1$ and $p(\mathbf{x}_1) = 0$.

Let $\mathbf{y} := ((\mathbf{a}_k, b_k)_{k=1}^m, (\mathbf{a}, b))$ where $(\mathbf{a}_k, b_k)_{k=1}^m$ is the public key of the cryptosystem and $(\mathbf{a}, b)$ is taken uniformly at random from $\mathbb{Z}_p^n \times \mathbb{Z}_p$, which makes this a sample of U. Note that every sample from the distribution $A_{\mathbf{s},\chi}$ is $\mathbf{x}_0$. We will build an algorithm `Z` to distinguish whether a given input is $\mathbf{x}_0$ or $\mathbf{y}$. Our goal is to evaluate the probability of `W` accepting $\mathbf{y}$, which is $p(\mathbf{y})$. Given a $\mathbf{x}_0$ we can easily transform it into $\mathbf{x}_1$ just by mapping $(\mathbf{a}, b) \to (\mathbf{a}, \lfloor p/2 \rfloor + b)$. As we perform this mapping on $\mathbf{y}$ we obtain $\mathbf{y}'$.

- Using the argument [53, Claim 5.3], we may conclude that the two distributions of samples of $\mathbf{a}$ that are:

  (a) uniformly chosen sum over vectors $(\mathbf{a}_k)_{k=1}^m$ in the public key, representing the distribution of encrypted bits with value '0'; and

  (b) uniform choice from $\mathbb{Z}_p^n$, representing $A_{\mathbf{s},\chi}$

  are statistically extremely close to one another. Therefore, we may assume that `W` accepts every sample from $A_{\mathbf{s},\chi}$.

- By definition, the sample $\mathbf{y}$ from the uniform distribution U has the equal probability of appearing as a sample as $\mathbf{y}'$ does.

- If `W` always accepts only one of the samples $\{\mathbf{y}, \mathbf{y}'\}$, then $p(\mathbf{y}) = 1/2$.

- If $p(\mathbf{y}) > 1/2$, then by pigeonhole principle there are instances for which `W` accepts both $\{\mathbf{y}, \mathbf{y}'\}$, which is not possible for samples from the distribution $A_{\mathbf{s},\chi}$.

Assume that `Z` is given some samples from distribution R which is either $A_{\mathbf{s},\chi}$ or U. For each sample $\mathbf{z}$, `Z` calls `W`. If `W` rejects $\mathbf{z}$, then `Z` rejects R. If `W` accepts the initial sample, then `W` is also called to check $\mathbf{z}'$. If $\mathbf{z}'$ is also accepted, then `Z` rejects R. If every sample passes this, then `Z` accepts R. From the above discussion we conclude that the probability that a single sample $\mathbf{z}$ from U passes, is at most $1/2$. Hence, the probability of erroneous acceptation of U by `Z` decreases exponentially as the amount of samples from U increases. □

4.3 LWE and lattice problems

The main theorem in Regev's work is proving that if there exists an algorithm that solves $\mathrm{LWE}_{p,\chi}$ efficiently, then there exists a quantum algorithm that approximates GapSVP and SIVP efficiently. Between the original [52] and the updated [53] versions, a work by C. Peikert from 2009 [49] appeared and in that Peikert shows that for exponentially large modulus p there is a classical reduction from GapSVP to LWE.

The strategy in proving the desired results is to link these lattice problems to *discrete Gaussian sampling problem* DGS, which can be linked to LWE.

Definition 4.6. For a given lattice L, any $\mathbf{v} \in L$ and any $r > 0$, the discrete Gaussian probability distribution on lattice L is defined by

$$D_{L,r}(\mathbf{v}) := \frac{\exp(-\pi\|\mathbf{v}/r\|^2)}{\sum_{\mathbf{x}\in L}\exp(-\pi\|\mathbf{x}/r\|^2)}.$$

Discrete Gaussian sampling problem DGS. Assume that φ is a real-valued function on lattices. Let $r > \varphi(L)$. DGS_φ is about outputting a sample from $D_{L,r}$.

Definition 4.7. For an n-dimensional lattice L and $\varepsilon > 0$, we define the smoothing parameter $\eta_\varepsilon(L)$ to be the smallest s such that

$$\sum_{\mathbf{x}\in L^*\setminus\{\mathbf{0}\}} \exp(-\pi\|s\mathbf{x}\|^2) \leq \varepsilon,$$

where L^* is the dual lattice (introduced in section 3.3) of L, holds.

The meaning of the smoothing parameter $\eta_\varepsilon(L)$ is that it gives the smallest r starting from which $D_{L,r}$ behaves like a continuous Gaussian distribution. For instance, if $r \geq \eta_\varepsilon(L)$, then $D_{L,r}$ outputs vectors that have a norm roughly $\|r\sqrt{n}\|$ with high probability. On the other hand, if r is sufficiently small, then the samples are most likely nearly zero vectors.

LWE and DGS

The modified closest vector problem $\mathbf{CVP}_{L,d}$ asks to output a closest vector in L to a given point $\mathbf{x}$, which is promised to be within a distance d of lattice L.

> Let L be an n-dimensional lattice. Let $\varepsilon = \varepsilon(n)$ be a negligible function, $p = p(n) \geq 2$ be an integer, and $\alpha = \alpha(n) \in (0,1)$ be a real number such that $\alpha p > 2\sqrt{n}$ holds. L^* is the dual lattice of L.

Our goal is to show that being able to solve LWE efficiently implies the ability to solve the modified closest vector problem (part A) in a way that allows us to use this ability to output samples in DGS (part B) by applying quantum computation. This links LWE with DGS.

Part A. The first part is to show there exists $c > 0$ so that the existence of an algorithm, which is given n^c samples from $D_{L,r}$, and is able to solve $\mathrm{LWE}_{p,\Psi_\alpha}$ with polynomial amount of samples implies that there exists an algorithm, which solves $\mathrm{CVP}_{L^*,\alpha p/(\sqrt{2}r)}$ where $r > \sqrt{2}p\eta_\varepsilon(L)$.

This is proven in the following way. First it is shown that $\mathrm{CVP}_{L,d}$ can be solved efficiently if we can solve the same problem in $\mathbb{Z}_p$ version [53, Lemma 3.5]. We note

that we can denote L as $(n \times n)$-matrix whose column vectors are the basis of lattice L. We also denote $\kappa_{L^*}(\mathbf{x})$ as the closest vector to $\mathbf{x}$ in L^*. The essential part of the proof is to use the fact that for a sample $\mathbf{v}$ from $D_{L,r}$ we have that

$$\kappa_{L^*}(\mathbf{x}) \cdot \mathbf{v} \equiv (L^*)^{-1}\kappa_{L^*}(\mathbf{x}) \cdot L^{-1}\mathbf{v} \pmod{p},$$

since $L^{-1} = (L^*)^T$. This holds between any choice of vectors of L^* and L. Now, we note that distribution of $L^{-1}\mathbf{v}$ is very close to uniform. This implies that we could think $L^{-1}\mathbf{v}$ as uniformly chosen $\mathbf{a}$ and $(L^*)^{-1}\kappa_{L^*}(\mathbf{x})$ as secret vector $\mathbf{s}$. With the chosen parameters we may conclude that this distribution is indeed very close to $A_{\mathbf{s},\Psi_\beta}$ for some $\beta \leq \alpha$ and solving LWE gives the wanted shortest vector $\kappa_{L^*}(\mathbf{x})$. Hence, we can solve $\mathrm{CVP}_{L^*,\alpha p/(\sqrt{2}r)}$ modulo p and thus the main $\mathrm{CVP}_{L^*,\alpha p/(\sqrt{2}r)}$.

Part B. The second part is to show that if there exists an efficient algorithm that is able to solve $\mathrm{CVP}_{L^*,d}$, with $d < \lambda(L^*)/2$, then there exists an quantum algorithm which is able to output a sample from $D_{L,\sqrt{n}/(\sqrt{2}d)}$.

We follow the informal description of the proof. We begin with creating a uniform superposition of L^*,

$$\sum_{\mathbf{x}\in L^*} |\mathbf{x}\rangle. \tag{4.3}$$

An important note compared to what we agreed in section 1 is that now the values of the amplitudes are taken from probability density function. On a separate register, we create a Gaussian state

$$\sum_{\mathbf{z}\in\mathbb{R}^n} \exp(-\pi\|r\mathbf{z}\|^2)|\mathbf{z}\rangle. \tag{4.4}$$

Next, we combine (4.3) and (4.4) and obtain

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

Now, the trick is that by uncomputing the first part $\mathbf{x}$ we obtain a state

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

and applying quantum Fourier transform we obtain a state that is very close to state

$$\sum_{\mathbf{y}\in L} D_{L,r}(\mathbf{y})|\mathbf{y}\rangle,$$

and measuring this state gives us samples from $D_{L,r}$. However, uncomputing $\mathbf{x}$ has its consequences and only way to recover $\mathbf{x}$ is solving CVP with a given $\mathbf{x}+\mathbf{z}$. As almost all of the probability mass of $\exp(-\pi\|r\mathbf{z}\|^2)$ is on $\mathbf{z}$ such that $\|\mathbf{z}\| \leq \sqrt{n}/r$, being able to solve $\mathrm{CVP}_{L^*,\sqrt{n}/r}$ is sufficient.

Hence, assuming that we have n^c samples from $D_{L,r}$, we are able to solve $\mathrm{CVP}_{L^*,\alpha p/r}$ by part A and produce n^c samples from $D_{L,r\sqrt{n}/(\alpha p)}$ by part B and repeat. This creates a link between LWE and DGS, which is that being able to solve LWE implies that there exists an efficient quantum algorithm which is able to solve harder cases of DGS.

DGS and lattice problems

We will now show the connection between DGS and lattice problems that are GapSVP and SIVP. The connection between DGS and SIVP is fairly 'easily' achieved. We just have to show that we get the desired n short linearly independent vectors from the samples of $D_{L,r}$.

Lemma 4.5. ***Poisson summation formula.*** *For any lattice L and any function $f : \mathbb{R}^n \to \mathbb{C}$,*

$$f(L) = \det(L^*)\hat{f}(L^*),$$

where $\hat{f}$ is the Fourier transform of f. Lattice L is again considered to be $(n \times n)$-matrix whose column vectors are the basis of lattice L.

Proof. [53, Lemma 2.14] □

Lemma 4.6. *Let L be an n-dimensional lattice and let r be such that $r \geq \sqrt{2}\eta_\varepsilon(L)$ where $\varepsilon \leq 1/10$. Then for any subspace H of dimension at most $n-1$ the probability that $\mathbf{x} \notin H$ where $\mathbf{x}$ is chosen from $D_{L,r}$ is at least $1/10$.*

Proof. Assume without loss of generality that the vector $(1, 0, \ldots, 0)$ is orthogonal H. Using lemma 4.5 twice we obtain,

$$\begin{aligned}
&\mathbb{E}_{\mathbf{x}\sim D_{L,r}}[\exp(-\pi(x_1/r)^2)] \\
&= \frac{1}{\sum_{\mathbf{x}\in L}\exp(-\pi\|\mathbf{x}/r\|^2)} \sum_{\mathbf{x}\in L} \exp(-\pi(\sqrt{2}x_1/r)^2)\exp(-\pi(x_2/r)^2)\ldots\exp(-\pi(x_n/r)^2) \\
&= \frac{\det(L^*)r^n}{\sqrt{2}\sum_{\mathbf{x}\in L}\exp(-\pi\|\mathbf{x}/r\|^2)} \sum_{\mathbf{y}\in L^*} \exp(-\pi(ry_1/\sqrt{2})^2)\exp(-\pi(ry_2)^2)\ldots\exp(-\pi(ry_n)^2) \\
&\leq \frac{\det(L^*)r^n}{\sqrt{2}\sum_{\mathbf{x}\in L}\exp(-\pi\|\mathbf{x}/r\|^2)} \sum_{\mathbf{y}\in L^*} \exp(-\pi\|r\mathbf{y}/\sqrt{2}\|^2) \\
&\leq \frac{\det(L^*)r^n}{\sqrt{2}\sum_{\mathbf{x}\in L}\exp(-\pi\|\mathbf{x}/r\|^2)}(1+\varepsilon) \\
&\leq \frac{\det(L^*)r^n\sum_{\mathbf{y}\in L^*}\exp(-\pi\|r\mathbf{y}\|^2)}{\sqrt{2}\sum_{\mathbf{x}\in L}\exp(-\pi\|\mathbf{x}/r\|^2)}(1+\varepsilon) \\
&= \frac{\sum_{\mathbf{x}\in L}\exp(-\pi\|\mathbf{x}/r\|^2)}{\sqrt{2}\sum_{\mathbf{x}\in L}\exp(-\pi\|\mathbf{x}/r\|^2)}(1+\varepsilon) \leq \frac{1}{\sqrt{2}}(1+\varepsilon) < \frac{9}{10}.
\end{aligned}$$

□

Corollary 4.7. *Let L be an n-dimensional lattice and let r be such that $r \geq \sqrt{2}\eta_\varepsilon(L)$ where $\varepsilon \leq 1/10$. Then, the probability that a set of n^2 vectors chosen independently from $D_{L,r}$ contains no n linearly independent vectors is exponentially small.*

Proof. Let $\mathbf{x}_1, \ldots, \mathbf{x}_{n^2}$ be n^2 vectors chosen independently from $D_{L,r}$. For $i = 1, 2, \ldots, n$, let B_i be the event that

$$\dim \operatorname{span}(\mathbf{x}_1, \ldots, \mathbf{x}_{(i-1)n}) = \dim \operatorname{span}(\mathbf{x}_1, \ldots, \mathbf{x}_{in}) < n.$$

If for any i, B_i does not happen, then $\dim \operatorname{span}(\mathbf{x}_1, \ldots, \mathbf{x}_{n^2}) = n$. We fix the value of i and assume that

$$\dim \operatorname{span}(\mathbf{x}_1, \ldots, \mathbf{x}_{(i-1)n}) < n.$$

Now, by Lemma 4.6 the probability that

$$\mathbf{x}_{(i-1)n+1}, \ldots, \mathbf{x}_{in} \in \operatorname{span}(\mathbf{x}_1, \ldots, \mathbf{x}_{(i-1)n})$$

is at most $(9/10)^n = 2^{-\Omega(n)}$. This implies that $\Pr(B_i) \leq 2^{-\Omega(n)}$, which is sufficient for completing the proof. □

We now have the elements necessary to link DGS with SIVP. Essentially, we just have to use LLL-algorithm to solve SIVP_γ in $\gamma(n) = 2^n$ case and denote the length of the result as $\hat{\lambda}_n$. Then in the spirit of Corollary 4.7, we use DGS oracle to output n^2 samples from D_{L,r_i} where $r_i = \hat{\lambda}_n 2^{-i}$, $i \in \{0, 1, \ldots, 2n\}$ and look for the shortest solution we find. This idea is used to prove the following lemma.

Lemma 4.8. *For any $\varepsilon = \varepsilon(n) \leq 1/10$ and any $\varphi(L) \geq \sqrt{2}\eta_\varepsilon(L)$, there is a polynomial time reduction from $SIVP_{\tilde{O}(n/\alpha)}$ to DGS_φ.*

Proof. See [53, Lemma 3.17]. □

Lastly, we want to link DGS with GapSVP. First, we introduce some closest vector variants.

In $\mathbf{GapCVP}_\gamma$ given a lattice L, distance $d > 0$ and a vector $\mathbf{x}$ the output is YES, if $\mathrm{dist}(L, \mathbf{x}) \leq d$ and NO, if $\mathrm{dist}(L, \mathrm{x}) > \gamma(n) \cdot d$.

In $\mathbf{GapCVP}'_\gamma$ given a lattice L, distance $d > 0$ and a vector $\mathbf{x}$ the output is YES, if $\mathrm{dist}(L, \mathbf{x}) \leq d$ and NO, if $\mathrm{dist}(L, \mathrm{x}) > \gamma(n) \cdot d$ and $\lambda(L) > \gamma(L) \cdot d$.

According to paper by O. Goldreich, D. Micciancio, S. Safra and J.-P. Seifert from 1999 [28], there is a polynomial time reduction from GapSVP_γ to GapCVP'_γ. Hence, linking DGS to GapCVP′ is sufficient.

Lemma 4.9. *For any $\gamma = \gamma(n) \geq 1$, there is a polynomial time reduction from $GapCVP'_{100\sqrt{n}\cdot\gamma(n)}$ to $DGS_{\sqrt{n}\gamma(n)/\lambda(L^*)}$.*

Proof. We approach this by NP verifier of coGapCVP introduced by D. Aharonov and O. Regev in 2004 [2]. Assume that algorithm V takes lattice L, distance $d > 0$, vector $\mathbf{x}$ and a sequence of vectors $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N$ in L^* for some $N = \text{poly}(n)$. When $\text{dist}(L, \mathbf{x}) \leq d$ the algorithm is guaranteed to reject. When $\text{dist}(L, \mathbf{x}) > 100\sqrt{n}d$ and $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N$ are chosen from $D_{L^*,1/(100d)}$ the algorithm accepts with probability exponentially close to 1. We can use the DGS oracle to make this choice of sequence $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N$. We accept if and only if V rejects.

In case V accepts we have that $1/(100d) > \sqrt{n}\gamma(n)/\lambda(n)$, hence $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N$ are valid samples from $D_{L^*,1/(100d)}$. Moreover, $\text{dist}(L, \mathbf{x}) > 100\sqrt{n}\gamma(n)d \geq 100\sqrt{n}d$, and hence we have correct result with probability exponentially close to 1. □

We hereby have enough justification for the following informal version of Regev's main theorem.

Theorem 4.10. *Let n, p be integers and $\alpha \in (0, 1)$ be such that $\alpha p > 2\sqrt{n}$. If there exists an efficient algorithm that solves LWE_{p,Ψ_α} then there exists an efficient quantum algorithm that approximates GapSVP and SIVP to within $\tilde{O}(n/\alpha)$ in the worst case.*

As mentioned before, in Peikert's work [49] it is shown, that there are classical reductions from GapSVP to LWE. However, removing the quantum part would force us to relinquish the SIVP part and to assume exponentially large p.

4.4 Ring learning with errors R-LWE

While we now have a strong public-key cryptosystem and evidence of the hardness of LWE, the cryptosystem is still impractical in terms of efficiency. For instance, in works by R. Lindner and C. Peikert [39] from 2011 and D. Micciancio and O. Regev [43] from 2009 it is concluded that the security parameter has to have value in several hundreds for sufficient security. Combining this with the fact that key size and required computation times are quadratic in the main security parameter, we see that efficiency suffers while security is improved. This is one of the motivations in the works by V. Lyubashevsky, C. Peikert and O. Regev, published in 2013 [40][41], in which ring learning with errors R-LWE is introduced. This takes the idea of original LWE to new algebraic levels where similar security is achieved but we also obtain much more efficient and practical public-key cryptosystem. We give a brief overview of R-LWE-based public-key cryptosystem.

Let $f(x) = x^n + 1 \in \mathbb{Z}[x]$, where the security parameter n is a power of 2, which also means that $f(x)$ is irreducible over the rationals. Let $R = \mathbb{Z}[x]/\langle f(x)\rangle$ be the ring of integer polynomials modulo $f(x)$. Let $q \equiv 1 \pmod{2n}$ be sufficiently large public prime modulus, bounded by a polynomial in n. We define $R_q = \mathbb{Z}_q[x]/\langle f(x)\rangle$.

Key pair generation

- Choose a polynomial $a(x) \in R_q$ uniformly at random.
- Choose the secret key polynomial $s(x) \in R$ with *"small"* coefficients based on some probability distribution which we call *error distribution* that is some n-dimensional Gaussian.

- Choose an error polynomial $e(x) \in R$ similarly as $s(x)$.
- The public key is $(a(x), b(x)) \in R_q$ where $b(x) = a(x)s(x) + e(x)$.

Encryption and decryption

- To encrypt n-bit message, present the message as a polynomial $m(x) \in R$ where the coefficients are either 0 or 1, representing the values of the bits in the message.
- Next, choose polynomials $r(x), e_1(x), e_2(x) \in R$ randomly from the error distribution.
- Compute two polynomials in R_q: $u(x) \equiv a(x)r(x) + e_1(x) \pmod q$ and $v(x) \equiv b(x)r(x) + e_2(x) + \lfloor q/2 \rceil m(x) \pmod q$.
- Output the pair $(u(x), v(x)) \in R_q^2$.
- The message is then decrypted by computing

  $$v(x) - u(x)s(x) \equiv (r(x)e(x) - s(x)e_1(x) + e_2(x)) + \lfloor q/2 \rceil m(x) \pmod q,$$

  where for each coefficient in the resulting polynomial we determine whether the coefficient is closer to 0 (decryption is '0') or closer to $\lfloor q/2 \rceil$ (decryption is '1') modulo q.
- Since all the polynomials $s(x), e(x), e_1(x), e_2(x), r(x)$ are chosen from the appropriate error distribution, we way trust that decryptions are done correctly with very high probability like in the original LWE-system.

Semantic security is obtained by showing that the public key is pseudorandom even though the secret key is a sample of the error distribution. Recovering $s(x)$ from the public key, which is essentially R-LWE, is anchored to worst-case approximate SVP on ideal lattices by quantum reductions.

For the concept of ideal lattices we fix an additive isomorphism σ mapping the ring R to some lattice $\sigma(R)$ in $\mathbb{R}^n$. A trivial example of this is a function that maps any element from R to a vector in $\mathbb{Z}^n$ so that the coefficients of the selected polynomial from R corresponds to the coordinates of the vector in $\mathbb{Z}^n$. The family of ideal lattices for the ring R with embedding σ is the set of all lattices $\sigma(I)$ for ideals I in R.

The strength of R-LWE versus standard LWE is that each computation of $b(x)$ gives n simultaneous pseudorandom values over $\mathbb{Z}_q$ whereas in standard version we get just one scalar. Polynomial multiplications can be performed in $O(n \log n)$ scalar operations and in parallel depth $O(\log n)$ using the Fast Fourier Transform, which makes the cost of this option small. Moreover, each sample $(a, b) \in R_q \times R_q$ in R-LWE can replace n samples $(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ in standard LWE. This allows reducing the size of public key and often also the secret key by a factor of n.

The authors interestingly mentions that this system resembles `Diffie-Hellman key exchange` and `ElGamal`, which are based on the quantum-vulnerable discrete logarithm problem. The discrete logarithm problem can be seen as an attempt to

recover s from $a^s = b$ in G, where a is the generator of cyclic group G and where a, b and G are publicly known. The noisy equation $a(x)s(x) + e(x) = b(x)$ in the public key of R-LWE can thus be seen as noisy version of the same idea as in discrete logarithm problem. This aligns with our philosophy of trying to add uncertainty to resist the attacks by quantum computation, which threatens these previous classical problems that are too deterministic.

5 Other lattice-based public-key cryptosystems

In this section, we will check two other lattice-based public-key cryptosystems. These are `NTRU`, which is promising for real-life applications, and broken `GGH` as a warning example.

5.1 NTRUEncrypt

`NTRUEncrypt`, introduced by J. Hoffstein, J. Pipher and J. H. Silverman in 1998 [30] is ring-based public-key cryptosystem, which also precedes earlier presented LWE and R-LWE systems.

Let N, p, q be integers such that $\gcd(p, q) = 1$ and q is considerably larger than p. Let $\mathcal{L}_f, \mathcal{L}_g, \mathcal{L}_\varphi, \mathcal{L}_m$ be sets of polynomials of degree $N-1$ with integer coefficients. We work in the ring $R = \mathbb{Z}[x]/\langle x^N - 1\rangle$.

Key generation

- Choose random polynomials $g \in \mathcal{L}_g$ and $f \in \mathcal{L}_f$ in a way that f also has inverses modulo p and modulo q. Compute (by a modification of the Euclidean algorithm) these inverses F_p and F_q respectively.
- Compute the public key polynomial h satisfying

$$F_q(x)g(x) \equiv h(x) \pmod{q}.$$

- The private key is the polynomial pair $(f(x), F_p(x))$.

Encryption and decryption

- Select the plaintext polynomial m from $\mathcal{L}_m$ representing the message.
- Choose randomly a polynomial φ from $\mathcal{L}_\varphi$.
- Compute ciphertext polynomial $e(x)$ satisfying

$$p\varphi(x)h(x) + m(x) \equiv e(x) \pmod{q}$$

- The ciphertext $e(x)$ is then decrypted by following steps. First, we compute $a(x)$ satisfying

$$\begin{aligned} &f(x)e(x) \\ &\equiv p\varphi(x) \underbrace{f(x)h(x)}_{=f(x)F_q(x)g(x)} + f(x)m(x) \equiv p\varphi(x)g(x) + f(x)m(x) \equiv a(x) \pmod{q}. \end{aligned}$$

- Next, the coefficients in $a(x)$ are adjusted to be from the interval $[-q/2, q/2]$. This allows, in the case of suitable parameters, the decrypter almost always to recover $a(x)$ in $\mathbb{Z}[x]/\langle x^N - 1\rangle$ as its coefficients are naturally from the interval $[-q/2, q/2]$.
- Finally, the message is recovered by computing

$$F_p(x)a(x) \equiv 0 + \underbrace{F_p(x)f(x)}_{\equiv 1 \pmod{p}} m(x) \equiv m(x) \pmod{p}.$$

Like in LWE, this system leaves room for errors. It turns out that in order to have a successful decryption we need

$$\|\mathbf{f} \cdot \mathbf{m} + p\varphi \cdot \mathbf{g}\|_\infty < q$$

to hold, where we changed the introduced polynomials to vectors, respectively. However, by right choices, the probability of successful decryption is in *"virtually always"* class. This happens if

$$\|\mathbf{f} \cdot m\|_\infty < q/4 \quad \text{and} \quad \|p\varphi \cdot \mathbf{g}\|_\infty < q/4.$$

For suitable choices of sets $\mathcal{L}$ we can define

$$\mathcal{L}(d_1, d_2) := \left\{ F \in R : \begin{array}{c} F \text{ has } d_1 \text{ coefficients equal to } 1, \\ d_2 \text{ coefficients equal to } -1, \text{ rest } 0 \end{array} \right\}.$$

Example cases of different levels of security by J. Hoffstein, J. Pipher, J. M. Schanck, J. H. Silverman, W. Whyte and Z. Zhang [31, Table 4], 2017:

- Fix $p = 3$.
- The following parameters are related to definitions of $\mathcal{L}$.
- Parameters d_1, d_2, d_3 are counts for non-zero coefficients for product form polynomial terms.
- d_g is a non-zero coefficient count for private key component g.
- d_m is the Hamming weight of message m.

Classical security est	Quantum security est	N	q	$(d_1, d_2, d_3, d_g, d_m)$
128	128	443	2048	$(9, 8, 5, 148, 115)$
192	128	587	2048	(10, 10, 8, 196, 157)
256	128	743	2048	(11, 11, 15, 247, 204)

Meet-in-the middle attacks. A meet-in-the middle attack, credited to be pointed out by Andrew Odlyzko, is used against φ. The attack is based on splitting $f(x) = f_1(x) + f_2(x)$ and then trying to find $f(x)$ by trying to match $f_1(x)e(x)$

approximately with $-f_2(x)e(x)$ with different candidate pairs (f_1, f_2). See the report by N. Howgrave-Graham, J. H. Silverman and W. Whyte from 2003, [32] for further details.

Multiple transmission attacks. If the same message $m(x)$ is sent multiple times with different choices for φ, then it is possible to recover parts of the message. Assume that the same message is sent k times and φ_i, $i = 1, 2, \ldots, k$ represent the choice of φ for each of these instances. Now, the attacker is able to compute $(e_i(x) - e_1(x))h^{-1}(x) \pmod q$, revealing $\varphi_i(x) - \varphi(x) \pmod q$, and because coefficients of φ are small, recovering $\varphi_i(x) - \varphi(x)$ in R is possible. Hence, it is more and more easier to recover φ_1, if k is larger, which is enough to recover $m(x)$. However, this attack works only against the same message $m(x)$ and other messages are not threatened.

On lattice-based attack on the private key. Consider $2N \times 2N$ matrix

$$\left(\begin{array}{cccc|cccc} \alpha & 0 & \ldots & 0 & h_0 & h_1 & \ldots & h_{N-1} \\ 0 & \alpha & \ldots & 0 & h_{N-1} & h_0 & \ldots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \alpha & h_1 & h_2 & \ldots & h_0 \\ \hline 0 & 0 & \ldots & 0 & q & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 & 0 & q & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 & 0 & 0 & \ldots & q \end{array}\right)$$

where α is an adjustable parameter and $\mathbf{h} = (h_0, h_1, \ldots, h_{N-1})$ is the public key polynomial h transformed into vector form. Let the lattice L be generated by this matrix. Since $h(x) = g(x)f^{-1}(x)$, lattice L contains a vector

$$\tau = (\alpha f_0, \alpha f_1, \ldots, \alpha f_{N-1}, g_0, g_1, \ldots, g_{N-1}).$$

Recovering τ will recover the private key. By heuristic arguments we may expect that the length of the shortest vector is

$$s = \sqrt{\frac{N\alpha q}{\pi e}},$$

and that we have

$$c_h = \sqrt{\frac{2\pi e \|\mathbf{f}\| \|\mathbf{g}\|}{Nq}}$$

so that $\|\tau\| = c_h s$ holds. Note that $\|\mathbf{f}\|$ and $\|\mathbf{g}\|$ are public quantities. Now finding τ is easier, if we know the shortest vector in L but this is essentially the shortest vector problem and the problem of finding τ is harder if $c_h \approx 1$. Hence, we have evidence that solving SVP would break `NTRUEncrypt`, but in the other hand there is no equivalence.

As we earlier compared R-LWE-based system to `Diffie-Hellman key exchange` and `ElGamal`, it is fair to mention that `NTRUEncrypt` is compared to `RSA`. This is because the security is based on the hardness assumption of factoring polynomials in a certain way. This is also more efficient than LWE-based systems but on the other hand, the security is not as strongly proved, see for example work by Y. Wang and M. Wang from 2022 [62].

5.2 Broken GGH

After the discussion of security of `NTRUEncrypt` we will now study lattice-based public-key cryptosystem `GGH`, introduced by O. Goldreich, S. Goldwasser and S. Halevi in 1997 [27]. The system has been then broken by serious vulnerability discoveries by P. Nguyen in 1999 [48].

Definition 5.1. Let B be a real non-singular $n \times n$ matrix. The orthogonality defect of B is defined by

$$\text{orth-defect}(B) := \frac{\prod_{k=1}^{n} \|\mathbf{b}_k\|}{|\det(B)|},$$

where $\mathbf{b}_k$ is the k-th column in B. Moreover, the dual orthogonality defect of B is defined by

$$\text{orth-defect}^*(B) := \frac{\prod_{k=1}^{n} \|\hat{\mathbf{b}}_k\|}{|\det(B^{-1})|},$$

where $\hat{\mathbf{b}}_k$ is the k-th column in B^{-1}.

Key pair generation

- Generate two matrices B and R that represent two different bases of the same lattice L so that B has high dual-orthogonality defect and R has low. Select real number σ. We declare (B, σ) as the public key and R as the private key.

Encryption and decryption

- The message is put to vector $\mathbf{v} \in \mathbb{Z}^n$. The error vector $\mathbf{e}$ is constructed in a way that each entry of this vector is chosen uniformly at random from the set $\{-\sigma, \sigma\}$.

- The ciphertext is $\mathbf{c} = B\mathbf{v} + \mathbf{e}$.

- To decrypt, we use Babai's round-off algorithm by L. Babai, 1986 [7]. We represent $\mathbf{c}$ as a linear combination on the columns of R and then round the coefficients of the linear combination to nearest integers to get a lattice point. The representation of this lattice point as a linear combination on the columns of B is $\mathbf{v}$.

At the time it was presented that solving CVP, which is known to be NP-hard by P. van Emde Boas [11] from 1981, would suffice for recovering $\mathbf{c}$. This is based on the *nearest-plane attack*, which is based on another algorithm presented in [7]. The algorithm is given $\mathbf{c}$ and a `LLL`-reduced basis of the lattice L and it outputs a certain hyperplane closest to $\mathbf{c}$ and the projection of $\mathbf{c}$ onto the hyperplane. As the projection lies on the hyperplane, spanned by the basis of $(n-1)$-dimensional sublattice, all we essentially have to do is to find the vector in this sublattice closest to the projection.

However, as we warned earlier, there are two major flaws in this cryptosystem, as Nguyen pointed out in [48]. Firstly, any ciphertext leaks information about the plaintext, and secondly, the CVP instances can be reduced to special cases that are much easier to solve. The first vulnerability is based on the observation that since all the entries in $\mathbf{e}$ are either $-\sigma$ or σ, we can by $\mathbf{s} = (\sigma, \sigma, \ldots, \sigma)$ obtain

$$\mathbf{c} + \mathbf{s} \equiv B\mathbf{v} \pmod{2\sigma},$$

which essentially recovers the secret part modulo 2σ. It is then shown, that by very high probability solving $\mathbf{v}$ modulo 2σ leads to a few easily computable possibilities for the original $\mathbf{v}$ and thus recovering $\mathbf{v}$ at least partially becomes feasible.

The second vulnerability is based on the same observation. Let $\mathbf{v}_{2\sigma}$ denote $\mathbf{v} \pmod{2\sigma}$. Now

$$\mathbf{c} - B\mathbf{v}_{2\sigma} = B(\mathbf{v} - \mathbf{v}_{2\sigma}) + \mathbf{e}.$$

By defining $\mathbf{v}'$ so that $\mathbf{v} - \mathbf{v}_{2\sigma} = 2\sigma\mathbf{v}'$, holds we obtain

$$\frac{\mathbf{c} - B\mathbf{v}_{2\sigma}}{2\sigma} = B\mathbf{v}' + \frac{\mathbf{e}}{2\sigma}.$$

As we know the rational point $(\mathbf{c} - B\mathbf{v}_{2\sigma})/(2\sigma)$, solving $\mathbf{v}'$ is a CVP instance, but now every entry of $\mathbf{e}/(2\sigma)$ is either $-1/2$ or $1/2$, making this CVP instance much easier. In this new case the length of error vector is $\sqrt{n/4}$ and in the earlier case it was $\sigma\sqrt{n}$. Solving $\mathbf{v}'$ allows us to recover $\mathbf{v}$.

The main flaw in `GGH` seems to be the too precise form of the error vector. This could be fixed by choosing the entries uniformly from the interval $[-\sigma, \sigma]$, but it would create much shorter error vectors, which creates easier instances of CVP. However, these attacks are statistical in a way, as they reveal information about the secrets and attacks are likely to succeed in a feasible amount of time. This could point to just increasing dimensions in the scheme to make it more secure, but the drawback is that efficiency suffers crucially and it is hard to regard `GGH` practical anymore.

This example can be seen as a warning example of design flaws. Even though it seems that solving a hard lattice problem recovers the private key, it does not mean that the recovering problem's hardness is equal to the hardness of the lattice problem. In the case of `NTRUEncrypt`, we might see something similar, even though it is still considered secure. `NTRUEncrypt` however does not seem to have similar structure of *"too predictable randomness"* as `GGH`.

6 On post-quantum era

In this section we look into the current state of quantum computing and discuss the needed capabilities for quantum computers to make classical public-key cryptography schemes vulnerable. We also discuss the National Institute of Standards and Technology of the United States NIST project of standardization of post-quantum cryptography and also a little bit about quantum hype and the question whether quantum advantage or supremacy is ever going to be achieved.

6.1 Current state of quantum computing

Ever since the theoretical model of quantum computing has been around, attempts to actually build a functional quantum computer has been a goal for companies and governments worldwide. From early 2-qubit computers by N. Gershenfeld and I. L. Chuang [24] and a demonstration by J. A. Jones and M. Mosca [33] in the late 1990's the current quantum computers have evolved to around 1000 qubits at best like Atom Computing's computer [64] or IBM Condor [67]. First successful demonstration of Shor's algorithm was achieved in 2001 by a group from IBM (Vandersypen et al) [61] where semiprime 15 was successfully factored into $3 \cdot 5$ by a 7-qubit quantum computer.

Quantum advantage or *quantum supremacy* refers to a point where quantum computers achieves something that is considered to be infeasible even for the most efficient classical computers. In case of the Shor's algorithm, current estimates for feasible factorizations of semiprimes meeting the current security standards of `RSA` (2048 bit key) are in millions of qubits (see e.g. Google's recent preprint by C. Gidney from 2025 [25]). Comparing this to the current situation, we are still very far away. However, the study reduced the earlier estimate from around 20 million qubits to one million. Some estimates say that quantum advantage could be achieved somewhere in the 2030's. Even today there is a common saying that goes "*harvest now, decrypt later*" referring to the fact that if encrypted messages are captured and stored, then they might be decrypted in the future. This is a good reason to consider using post-quantum cryptography schemes even today.

6.2 NIST post-quantum cryptography standardization

National Institute of Standards and Technology NIST, an agency operating under the government of the United States, runs a project of standardizing post-quantum cryptography. In 2024 NIST published [68] first post-quantum encryption standards where `FIPS203` [46] is considered to match the public-key cryptography perspective we have in this thesis. It gives a standard of *Module-Lattice-Based Key-Encapsulation Mechanism ML-KEM* based on `CRYSTALS-KYBER`, introduced by Bos et al in 2018 [12]. The purpose of this mechanism is to create a shared secret key between two parties over insecure channels, which allows symmetric encryption schemes in further communication. Security is based on the hardness of module learning with errors M-LWE. This is another iteration after R-LWE as the used modules are basically cartesian products of certain polynomial rings.

Interestingly, while this ML-KEM standard is approved by NIST, it includes some public-key cryptography schemes that are not approved to be used as stand-alone schemes [46, Section 5]. Also, ML-KEM is approved to be used for protection of sensitive, but non-classified communication systems of the U.S. Federal Government [46, Section 1.1]. This might be reasonable due to the fact that this is relatively new scheme, and, as it is mentioned in the document, ML-KEM is "*presently believed*" to be secure. To address these concerns, NIST recently announced in 2025 [69] that `HQC` algorithm is selected to serve as backup, should weaknesses in ML-KEM to be discovered. The algorithm was proposed in 2018 by C. Aguilar-Melchor, O. Blazy, J.-C. Deneuville, P. Gaborit and G. Zemor [1] and it is based on the difficulty of decoding random quasi-cyclic codes, which is inspired by R-LWE but is significantly adapted to the coding theory setting.

NIST's 2024 report on transition to post-quantum cryptography standards [45] mentions year 2035 as a goal for adapting post-quantum era standards (see [45, Section 4.1]). Level of urgency depends on levels of confidentiality and risks, as we keep the harvest now, decrypt later mentality in mind. Legacy systems might pose structural challenges for this project. Year 2035 as a target is also mentioned in the European Union's post-quantum cryptography transition roadmap [66].

6.3 Quantum hype and practical challenges

As the race towards achieving quantum supremacy is on, it is also fair to discuss about the practical problems related to quantum computing and how the hype around the quantum affects the field. One may see analogies in recent advances in artificial intelligence technologies and the race towards artificial general intelligence, but we will not discuss about AI any further.

Since quantum computing would theoretically provide efficient algorithms to solve problems from cryptanalysis to other fields, it does raise a lot of expectations, fear and funds of investors. Therefore it might be tempting to make claims, which exaggerate the current or near-future capabilities of quantum computing. I. Barmes, C. Soutar and A. Veliz from Deloitte in 2025 write nicely about the hype from cryptanalysis perspective [65]. They point out that non-specialists following the progress might misunderstand the meaning of some achievements, which leads to exaggerating reports in the media. For example the amount of qubits is not the only issue since dealing with quantum error correction is a major challenge. The authors offer a framework to bridge the gap between non-professionals and researchers in the field.

We mentioned earlier that there has been successful demonstrations of factoring small semiprimes by Shor's algorithm with quantum computers. However, we should be more critical towards the meaning of these demonstrations. In 2013 paper [59] J. A. Smolin, G. Smith and A. Vargo point out that demonstrations of Shor's algorithm have always been simplifications which depend on knowing the correct answer in advance. They then demonstrate that basically any large semiprime can be factored even with a small quantum computer if we simplify the Shor's algorithm and exploit the fact that we already know the factors. Hence, it is not the size of the semiprime with respect to the amount of qubits what matters the most if we have this approach.

J.-Y. Cai in 2024 [14] showed that Shor's algorithm will almost surely fail with factoring a semiprime $N = pq$ even in presence of small noise if $p - 1$ has a prime factor larger than $p^{2/3}$ and $q - 1$ has a prime factor larger than $q^{2/3}$ respectively. Primes with such property have a positive density among all primes [14, Theorem 4 (Fouvry)]. Cai is also skeptical about the possibility of achieving arbitrary precision from a point of view that quantum mechanics is not and is not meant to be infinitely accurate description of reality. To support this view, the author gives a reasoned opinion that the standard model used to describe qubit is only approximately true. In a 2024 follow-up preprint J.-Y. Cai and B. Young [15] show that similar failure of Shor's algorithm in presence of noise is expected in the version designed for solving the discrete logarithm problem. Y. Kurman, L. Ella, N. Halay, O. Wertheim and Y. Cohen in 2024 [37], studied the hardware needed for factoring 21 by Shor's algorithm with error corrections, which in successful case would be a step towards error corrected Shor's algorithm.

In 2023 paper T. Hoefler, T. Häner and M. Troyer [29] offer more pessimistic views towards quantum computing by claiming that most of today's quantum algorithms may not achieve practical speedups. However, they still consider Shor's algorithm with exponential quantum speedup to be in the practical class. The question whether quantum advantage is actually achieved might not always be clear either. For example a group from IBM (Kim et al.) published in 2023 [35] a result suggesting that even in pre-fault-tolerant era they found a physics problem which they managed to solve with a noisy 127-qubit processor "*at a scale beyond brute-force classical computation.*" After this in 2024 J. Tindall, M. Fishman, E. M. Stoudenmire and D. Sels [60] showed that in fact, this problem can after all be solved by a classical computer in a way that outperforms the quantum processor. We have to remember that many problems are just believed to be hard for classical computers but we do not know if they actually are that hard. For example we do not even know whether $P \neq NP$ is true even though many believe so.

From a cryptographic perspective, it is justified to be critical towards the future capabilities of quantum computing and whether there is actually need to perform arduous and expensive transition into post-quantum schemes. However, the research in quantum computing is advancing and hardwares and error corrections are only getting better and software design is only getting better and problems might be solvable with less qubits. Most importantly, being wrong with an assumption that quantum computers would not pose a threat against classical encryption schemes is the most expensive outcome. It is too late to perform the transition after current schemes are no longer secure.

In an article by M. Brooks from 2023 [13] it is nicely summarized that quantum computers are now good for absolutely nothing but researchers and firms are optimistic. One should expect that advance happens slowly but steadily. A quote by W. Hensinger from University of Sussex goes "*There's not going to be this one point when suddenly we have a rainbow coming out of our lab and all problems can be solved.*" This is also good to keep in mind after being critical towards the significance of some achievements in quantum computing.

7 Discussion

Advances in the research of quantum computing combined with the theoretical capabilities foresee uncertain future in secure communication. The basic template of public-key cryptography has remained simple with relatively simple encryption schemes and their security is based on simple number theoretic assumptions. The rules of the game are about to change and even with the most pessimistic opinions of the possibility of achieving quantum advantage there is bound to be some feelings of urgency with implementing post-quantum schemes. Achieving quantum advantage will most likely do a lot of good things in many fields of science but obviously this is not necessary the case in cryptography.

The change in the philosophy of public-key cryptography is that we can no longer rely on simple hard problems with clear hidden solution and we have to implement controlled randomness and structures that are easy to generate but hard to explore. Lattices are like this, since it is easy to just define it to be a linear combination of given base vectors but it does not immediately tell how the lattice "*looks like*". However, something similar can be seen with the set of prime numbers. It is easy to define the set of prime numbers but we still do not understand the patterns of its distribution completely even though prime number theorem gives us good approximations. Shor's algorithm, however exploits the known algebraic structure of residue systems, but also the reason why `RSA` works in the first place is the known structures of these systems.

Lattice-based schemes and especially learning with errors introduce ways to hide structures in the randomness. A distribution that seems completely random, but is actually manipulated in a way that the one who knows how it was manipulated can distinguish whether the samples are from it or from actually uniform distribution, opens a door for bitwise encryption. Same message can be encrypted in multiple ways without padding with different encryption values, which is not the case in `RSA`. The secret is not hard-coded in the structure in a similar sense as in earlier public-key schemes because of the introduced errors. What is important is not the exact key vector the decrypter has but any vector which gives similar distribution of small errors. If one tries to break the system, they have to make a statistical judgment call in evaluation whether the obtained key vector works or not and there is no exact answer to this unlike in factorization of integers or discrete logarithm problem. The downturn of adding random errors is the possibility of errors in decryption, but this usually has negligible probability.

We may evaluate the time complexity of `RSA` encryptions and decryptions by the time complexity of modular exponentiation. In a book by K. H. Rosen, a fast algorithm for modular exponentiation is presented [55, Algorithm 5, Section 4.2.4] and according to that the number of bit operations for encryption is $O((\log N)^2 \log e)$ and for decryption $O((\log N)^2 \log d)$, where e is the public key, d is the private key and N is the public modulus, which is also the key representing the security of the whole system. Currently 2048-bit key for N is widely considered secure, but stronger keys might be needed in the future. We mentioned in section 4.4 that in standard LWE, the key sizes and computation times are quadratic in the main security parameter. The key size, however has to be in several hundreds if the

message carries even some way meaningful information like a 128-bit key. Hence, the standard LWE is not expected to be efficient especially if the length of the message grows. This was one motivation for developing LWE system in rings and modules.

It is fair to acknowledge that what makes lattice-based cryptography a good candidate for post-quantum cryptography is in the end based on the fact that we just believe it is secure even against attacks by quantum computers. Nobody has (yet) figured out quantum algorithms to perfom attacks efficiently. There has been attempts and sometimes preprints appear where author(s) claim it is possible. One example was in 2024, when Y. Chen [16] suggested that LWE could be solved in polynomial time by quantum computers. However, a bug was later found and even the suggested discovery would not have directly threatened the practical R-LWE-based schemes. Still, papers like this appearing might be scary from the security perspective. It is also good to keep in mind that belief is also the case with classical security today. We have not proved that there is no polynomial time classical algorithms able to factor integers but we just assume so. Indeed, this aligns with the more general question of whether $P = NP$ or $P \neq NP$ where the latter option is usually considered to be more likely.

Concluding remarks

Development of quantum computing is currently probably the second largest buzz after AI in the technological sector. At the moment we can just speculate what is in the end possible and what is actually realistic timetable regardless of aggressive roadmaps of big companies. Cryptography, like all the cyber security related areas, is constantly impacted by the changing world. There is so much at stake that it is highly recommendable to begin the transformation to post-quantum era in the cryptography schemes. Public-key cryptography is much more vulnerable to quantum attacks than symmetric cryptography, where two parties already have shared secrets.

Following the standardization project of NIST is a good way to see what schemes are currently recommended. Learning with errors and lattice problems seems to have been accepted as strong enough base for security and in this thesis we have seen the basic structure how it works and why we should believe in its security. Still, like NIST pointed out, we are maybe not confident enough to leave backups out of consideration. Standard LWE is not directly applied in practical schemes but the same idea is used in developed versions of LWE. The structures of new schemes are very different compared to `RSA` or `ElGamal` we are used to and this means that cryptography designs have to take much more complex and theoretical structures into account. Hence, the transformation project will probably take a lot of time.

It is exciting to see what future brings despite the possible threats against security. Whether one believes in quantum mechanics giving us computational advantage, the impact of threats realizing is so strong that there is no good reason not to start preparing. Studying lattice problems, learning with errors and schemes like `NTRUEncrypt` will gives us tools what we will likely need in the transformation process. Quantum mechanics might also offer some help in secure communication.

Quantum key distribution QKD applies quantum mechanics to create a shared secret key between two parties in a secure way. Applying quantum mechanics this way which is beneficial in cryptography rather than cryptanalysis, like in the main perspective of this thesis, was already proposed by S. Wiesner in the late 1960's, but the paper was way ahead of its time. The manuscript [63] was eventually published in 1983. In 1984, C. Bennett and G. Brassard [8] introduced *BB84* QKD protocol in which one party prepares quantum states of photons and another party measures them and the laws of quantum physics reveal whether the transmission has been disturbed by an eavesdropper.

References

[1] C. Aguilar-Melchor, O. Blazy, J.-C. Deneuville, P. Gaborit and G. Zemor. Efficient Encryption From Random Quasi-Cyclic Codes. In *IEEE Transactions on Information Theory, 64(5), 3927–3943, 2018.*

[2] D. Aharonov and O. Regev. Lattice Problems in NP ∩ coNP. In *45th Annual IEEE Symposium on Foundations of Computer Science (pp. 362–371), 2004.*

[3] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proc. 28th ACM Symp. on Theory of Computing (STOC), pages 99–108, 1996.*

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

[5] M. Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions (extended abstract). In *Proc. 30th ACM Symp. on Theory of Computing (STOC), pages 10–19, 1998.*

[6] 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 (STOC), pages 601–610, 2001.*

[7] L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. In *Combinatorica, 6(1), 1–13, 1986.*

[8] C. Bennett and G. Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Theoretical Computer Science, 560, 7–11, 2014.*

[9] H. Bennett and C. Peikert. Hardness of the (Approximate) Shortest Vector Problem: A Simple Proof via Reed-Solomon Codes. In *Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.*

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

[11] P. van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. In *Reprot 81-04, Mathematische Instituut, University of Amsterdam, 1981.*

[12] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler and D. Stehle. CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P) (pp. 353–367). 2018 IEEE European Symposium on Security and Privacy (EuroS&P). IEEE, 2018.*

[13] M. Brooks. Quantum computers: what are they good for? In *Nature, 617(7962), S1–S3, 2023.*

[14] J.-Y. Cai. Shor's algorithm does not factor large integers in the presence of noise. In *Science China Information Sciences, 67(7), 2024.*

[15] J.-Y. Cai and B. Young. Quantum Algorithms for Discrete Log Require Precise Rotations. *Preprint available at arXiv:2412.17269, 2024.*

[16] Y. Chen. Quantum Algorithms for Lattice Problems. *Cryptology ePrint Archive, Paper 2024/555, 2024.*

[17] D. Deutsch. Quantum theory, the Church–Turing principle and the universal quantum computer. In *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences, 400(1818), 97–117, 1985.*

[18] D. Deutsch and R. Josza. Rapid solution of problems by quantum computation. In *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences, 439(1907), pages 553–558. 1992.*

[19] W. Diffie and M. Hellman. New directions in cryptography. In *IEEE Transactions on Information Theory, 22(6), pages 644–654, 1976*

[20] T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *IEEE Transactions on Information Theory, vol. 31, no. 4, pages 469-472, 1985.*

[21] R. P. Feynman. Simulating physics with computers. In *Int J Theor Phys 21, 467–488, 1982.*

[22] U. Fincke and M. Pohst. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. In *Mathematics of Computation, 44(170), 463–471, 1985.*

[23] N. Gama, P. Q. Nguyen and O. Regev. Lattice Enumeration Using Extreme Pruning. In *Lecture Notes in Computer Science (pp. 257–278). Springer Berlin Heidelberg, 2010.*

[24] N. Gershenfeld and I. L. Chuang. Quantum Computing with Molecules. In *Scientific American, 278(6), 66–71, 1998.*

[25] C. Gidney. How to factor 2048 bit RSA integers with less than a million noisy qubits. *Preprint available at arXiv:2505.15917, 2025.*

[26] O. Goldreich, S. Goldwasser and S. Halevi. Eliminating decryption errors in the Ajtai-Dwork Cryptosystem. In *Lecture Notes in Computer Science (pp. 105–111). Springer Berlin Heidelberg, 1997.*

[27] O. Goldreich, S. Goldwasser and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *Lecture Notes in Computer Science (pp. 112–131). Springer Berlin Heidelberg, 1997.*

[28] O. Goldreich, D. Micciancio, S. Safra and J.-P. Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. In *Information Processing Letters, 71(2), 55–61, 1999.*

[29] T. Hoefler, T. Häner and M. Troyer. Disentangling Hype from Practicality: On Realistically Achieving Quantum Advantage. In *Communications of the ACM, 66(5), 82–87, 2023.*

[30] J. Hoffstein, J. Pipher and J. H. Silverman. NTRU: A ring-based public key cryptosystem. In *Lecture Notes in Computer Science (pp. 267–288). Springer Berlin Heidelberg, 1998.*

[31] J. Hoffstein, J. Pipher, J. M. Schanck, J. H. Silverman, W. Whyte and Z. Zhang. Choosing Parameters for NTRUEncrypt. In *Lecture Notes in Computer Science (pp. 3–18). Springer International Publishing, 2017.*

[32] N. Howgrave-Graham, J. H. Silverman and W. Whyte. A Meet-In-The-Middle Attack on an NTRU Private Key. In *NTRU Cryptosystems Technical Report #004, Version 2.*

[33] J. A. Jones and M. Mosca. Implementation of a quantum algorithm on a nuclear magnetic resonance quantum computer. In *The Journal of Chemical Physics, 109(5), 1648–1653, 1998.*

[34] R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proc. 15th ACM Symp. on Theory of Computing (STOC), pages 193–206, 1983.*

[35] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. van den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme, and A. Kandala. Evidence for the utility of quantum computing before fault tolerance. In *Nature, 618(7965), 500–505, 2023.*

[36] N. Koblitz. Elliptic curve cryptosystems. In *Mathematics of Computation, 48(177), 203–209, 1987.*

[37] Y. Kurman, L. Ella, N. Halay, O. Wertheim and Y. Cohen. Controller-decoder system requirements derived by implementing Shor's algorithm with surface code. *Preprint availabe ar arXiv:2412.00289, 2024.*

[38] A. K. Lenstra, H. W. Lenstra Jr. and L. Lovász. Factoring polynomials with rational coefficients. In *Mathematische Annalen, 261(4), 515–534, 1982.*

[39] R. Lindner and C. Peikert. Better Key Sizes (and Attacks) for LWE-Based Encryption. In *Lecture Notes in Computer Science (pp. 319–339). Springer Berlin Heidelberg, 2011.*

[40] V. Lyubashevsky, C. Peikert and O. Regev. On Ideal Lattices and Learning with Errors over Rings. In *Journal of the ACM, 60(6), 1–35, 2013.*

[41] V. Lyubashevsky, C. Peikert and O. Regev. A Toolkit for Ring-LWE Cryptography. In *Lecture Notes in Computer Science (pp. 35–54). Springer Berlin Heidelberg, 2013.*

[42] D. Micciancio. Shortest Vector Problem. In *van Tilborg, H.C.A. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA, 2005.*

[43] D. Micciancio and O. Regev. Lattice-based Cryptography. In *Post-Quantum Cryptography (pp. 147–191). Springer Berlin Heidelberg, 2009.*

[44] V. S. Miller. Use of Elliptic Curves in Cryptography. In *Lecture Notes in Computer Science (pp. 417–426), Springer Berlin Heidelberg, 1986.*

[45] D. Moody, R. Perlner, A. Regenscheid, A. Robinson and D. Cooper. Transition to Post-Quantum Cryptography Standards. *(National Institute of Standards and Technology, Gaithersburg, MD), NIST Internal Report (IR) NIST IR 8547 ipd, 2024.*

[46] National Institute of Standards and Technology (U.S.). Module-lattice-based key-encapsulation mechanism standard. `https://doi.org/10.6028/nist.fips.203`, *2024.*

[47] P. Nguyen and J. Stern. Cryptanalysis of the Ajtai-Dwork cryptosystem. In *Lecture Notes in Computer Science (pp. 223–242). Springer Berlin Heidelberg, 1998.*

[48] P. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem from Crypto '97. In *Lecture Notes in Computer Science (pp. 288–304). Springer Berlin Heidelberg, 1999.*

[49] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *Proc. 41st ACM Symp. on Theory of Computing (STOC), pages 333–342, 2009.*

[50] C. Peikert. A Decade of Lattice Cryptography. In *Foundations and Trends® in Theoretical Computer Science, 10(4), 283–424, 2016.*

[51] C. Pomerance. A Tale of Two Sieves. In *Notices of the AMS. Vol. 43, no. 12. pages 1473–1485, 1996.*

[52] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proc. 37th ACM Symp. on Theory of Computing (STOC), pages 84-93, 2005.*

[53] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. Updated version in *Journal of the ACM, 56(6), 1–40, 2009.*

[54] R. L. Rivest, A. Shamir and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. In *Communications of the ACM, Volume 21, Issue 2, pages 120-126, 1978.*

[55] K. H. Rosen. Discrete Mathematics and Its Applications. Eighth edition. *New York, NY : McGraw-Hill, 2019.*

[56] C. P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. In *Theoretical Computer Science, 53(2–3), 201–224, 1987.*

[57] P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proc. 35th Annual Symposium on Foundations of Computer Science, pages 124-134, 1994.*

[58] D. R. Simon. On the power of quantum computation. In *Proc. 35th Annual Symposium on Foundations of Computer Science, pages 116-123, 1994.*

[59] J. A. Smolin, G. Smith and A. Vargo. Oversimplifying quantum factoring. In *Nature, 499(7457), 163–165, 2013.*

[60] J. Tindall, M. Fishman, E. M. Stoudenmire and D. Sels. Efficient Tensor Network Simulation of IBM's Eagle Kicked Ising Experiment. In *PRX Quantum, 5(1), 2024.*

[61] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang. Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance. In *Nature, 414(6866), 883–887, 2001.*

[62] Y. Wang and M. Wang. On the hardness of NTRU problems. In *Frontiers of Computer Science, 16(6), 2022.*

[63] S. Wiesner. Conjugate coding. In *ACM SIGACT News, 15(1), 78–88, 1983.*

News articles and press releases:

[64] Atom Computing. Quantum startup Atom Computing first to exceed 1,000 qubits. `https://atom-computing.com/quantum-startup-atom-computing-first-to-exceed-1000-qubits/`, October 24, 2023 (retrieved July 25, 2025).

[65] Deloitte, I. Barmes, C. Soutar and A. Veliz. Why Quantum Computers Aren't Cracking RSA Yet: A Practical Guide to Quantum Error Correction. `https://www.deloitte.com/content/dam/assets-shared/docs/services/risk-advisory/2025/why-quantum-computers-are-not-cracking-rsa-yet.pdf`, 2025 (retrieved July 26, 2025).

[66] European Comission. A Coordinated Implementation Roadmap for the Transition to Post-Quantum Cryptography. `https://digital-strategy.ec.europa.eu/en/library/coordinated-implementation-roadmap-transition-post-quantum-cryptography` 23 June 2025 (retrieved July 26, 2025).

[67] IBM, J. Gambetta. The hardware and software for the era of quantum utility is here. `https://www.ibm.com/quantum/blog/quantum-roadmap-2033`, December 4, 2023 (retrieved July 25, 2025).

[68] National Institute of Standards and Technology (U.S.). NIST Releases First 3 Finalized Post-Quantum Encryption Standards. https://www.nist.gov/news-events/news/2024/08/nist-releases-first-3-finalized-post-quantum-encryption-standards, August 13, 2024 (retrieved July 26, 2025).

[69] National Institute of Standards and Technology (U.S.). NIST Selects HQC as Fifth Algorithm for Post-Quantum Encryption. https://www.nist.gov/news-events/news/2025/03/nist-selects-hqc-fifth-algorithm-post-quantum-encryption, March 11, 2025 (retrieved July 26, 2025).

A Python demonstration of the LWE-cryptosystem

The following `Python` code creates the LWE-based cryptosystem introduced in section 4.1 with a given security parameter `n`, which is adjustable in the beginning. It also takes a parameter `length` and creates a message, which is `length` randomly chosen bits and demonstrates the encryption and decryption procedure by printing the results bitwise. It also prints the total amount of errors in the decryption. The code is meant for illustrative purposes. Increasing the security parameter to secure levels easily increases the running time as was discussed in the motivation of R-LWE in section 4.4.

```
import numpy as np
from numpy import random

#adjustable parameters:

n = 50 #security parameter
length = 10 #length of the message

#Finding parameter p
def isPrime(n):
    if n < 2:
        return False
    if n == 2:
        return True
    if n%2 == 0:
        return False
    for i in range(3,int(np.sqrt(n))+1,2):
        if n%i == 0:
            return False
    return True

SearchPrime = True
while SearchPrime:
    p = random.randint(n**2,2*n**2+1)
    if isPrime(p):
        SearchPrime = False

#Other parameters

m = int((1+n)*np.log(p))+1
alpha = 1/(np.sqrt(n)*(np.log(n))**2)

#Setting up the cryptosystem

#generating secret vector s
s = {}

for i in range(n):
    s[i] = random.randint(0,p)

#generating vectors a
a = {}

for i in range(m):
```

```
    temp = {}
    for k in range(n):
        temp[k] = random.randint(0,p)
    a[i] = temp

#generating b's with errors
b={}

for i in range(m):
    temp = a[i]
    b[i] = 0
    for k in range(n):
        b[i] = (b[i]+a[i][k]*s[k])%p
    e = int(np.round((random.normal(loc=0,scale=(alpha/(np.sqrt(2*
    np.pi))))*p)%p))
    b[i] = b[i] + e

#Creating random message based on the given length
#Message can be customized and also list format works in the code.
    However, length parameter has to match with the customized
    message.

message = {}

for i in range(length):
    message[i] = random.randint(0,2)


#Encryption and decryption methods

def encryption(bit,n,m,p,a,b):
    r = random.randint(2,m)
    subset = np.random.choice(m,r,replace = False)
    enc_a = {}
    for i in range(n):
        enc_a[i] = 0
        for sub in subset:
            enc_a[i] = (enc_a[i]+a[sub][i])%p
    enc_b = 0
    for sub in subset:
        enc_b = (enc_b + b[sub])%p
    if bit == 1:
        enc_b = (enc_b + int(p/2))%p
    return (enc_a,enc_b)

def decryption(s,p,n,enc_a,enc_b):
    check = 0
    for i in range(n):
        check = (check + s[i]*enc_a[i])%p
    result = (check-enc_b)%p
    result = result/p
    if (result > 0.25 and result < 0.75):
        return 1
    else:
        return 0

```

```
98 #Encryption and decryption demonstration
99
100 errors = 0
101
102 for i in range(length):
103     (x,y) = encryption(message[i],n,m,p,a,b)
104     d = decryption(s,p,n,x,y)
105     print("Sample: "+str(i)+" | Encrypted bit: "+str(message[i])+"
    | Decrypted bit: " + str(d)) #prints bitwise results, can be
    removed
106     if (message[i] != d):
107         errors = errors + 1
108 print("Decryption errors: " + str(errors)) #prints the total amount
     of errors
```

B Use of AI

In this thesis, Writefull for Overleaf integration has been used for copyediting purposes.