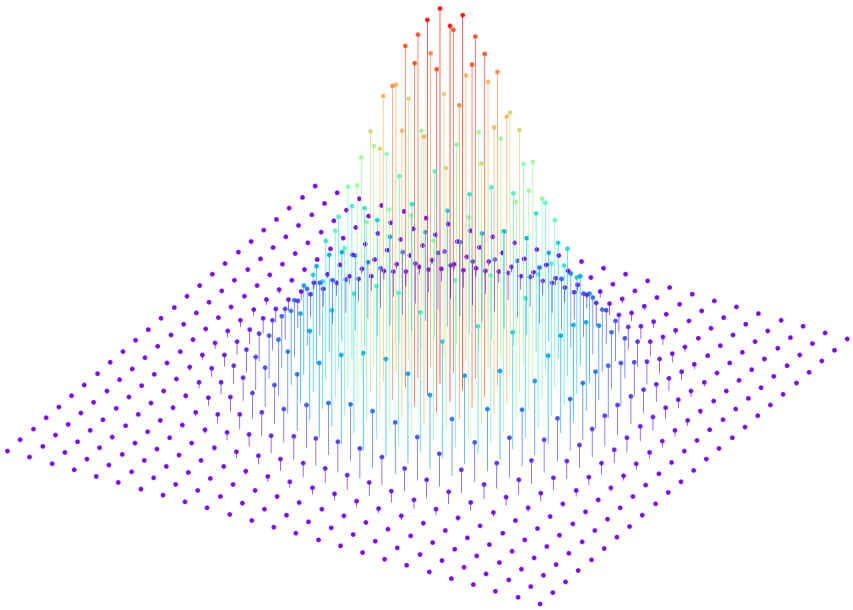
Oded Regev's Quantum Factoring Algorithm

Midas Kiebert

June 30, 2024

Bachelor thesis Mathematics and Computer Science

Supervisors: dr. Jeroen Zuiddam, dr. John van de Wetering



Informatics Institute
Korteweg-de Vries Institute for Mathematics
Faculty of Sciences
University of Amsterdam

Abstract

We describe Shor's algorithm and show that it can factor an n-bit integer using $\mathcal{O}(1)$ calls to an $\mathcal{O}(n^2 \log n)$ quantum subroutine. We give the necessary background in lattices to understand the proof of Regev's new quantum factoring algorithm, for readers with no familiarity with lattices. Then we describe Regev's algorithm and prove its correctness and efficiency (dependent upon an unproven assumption) in much greater detail than in Regev's own paper [Reg24]. In particular, we show that Regev's algorithm can factor an n-bit integer using $\mathcal{O}(\sqrt{n})$ calls to an $\mathcal{O}(n^{3/2} \log n)$ quantum subroutine.

Title: Oded Regev's Quantum Factoring Algorithm
Author: Midas Kiebert, midas.kiebert@student.uva.nl, 13978659
Supervisors: dr. Jeroen Zuiddam, dr. John van de Wetering
Second graders: dr. Maris Ozols, dr. Florian Speelman
End date: June 30, 2024

Informatics Institute
University of Amsterdam
Science Park 904, 1098 XH Amsterdam
http://www.ivi.uva.nl

Korteweg-de Vries Institute for Mathematics
University of Amsterdam
Science Park 904, 1098 XH Amsterdam
http://www.kdvi.uva.nl

Contents

1. Introduction

Cryptography relies on certain hardness assumptions. One of these assumptions is that factoring an integer is a hard problem. This means it is assumed that there does not exist an efficient[1] algorithm that can factor any integer. At the time of writing, there is no known polynomial-time *classical* algorithm that can factor any integer. However, if we allow *quantum* algorithms, Peter Shor showed in 1994 that this hardness assumption is false by giving a polynomial-time algorithm that can, with high probability, factor any integer using a quantum computer [Sho97]. More precisely, Shor's algorithm is a polynomial-time algorithm that uses a constant number of calls to a quantum subroutine, which requires $\mathcal{O}(n^2 \log n)$ quantum gates to factor an n-bit integer.

While this quantum subroutine is efficient in the sense that the number of gates grows polynomially, in practice, as of the time of writing, a physical quantum computer is not able to perform that many operations with any reasonable accuracy. Thus an algorithm that uses a different quantum subroutine that uses asymptotically even fewer quantum gates could be more feasible to implement.

In 2023, Oded Regev showed a polynomial-time algorithm that, given a certain assumption, uses $\mathcal{O}(\sqrt{n})$ independent calls to a quantum subroutine using $\mathcal{O}(n^{3/2} \log n)$ quantum gates [Reg24]. Adding each run of the quantum subroutine together this algorithm takes (asymptotically) just as many quantum gates as Shor's algorithm, but each independent run would take fewer gates. Generally it should be easier to have a quantum computer run a shorter circuit many times than a long circuit once, especially if there is a way to check if each individual run is successful.

Regev's algorithm can, in a way, be seen as a higher dimensional analogue of Shor's algorithm. Regev's algorithm makes extensive use of lattices, which are introduced in Chapter 4 of this thesis. We use lattices to describe the periodic behaviour of functions, in particular functions with multiple variables. This is reminiscent of Shor's algorithm, where an important part of the algorithm is finding the period of a function.

In this thesis we give a more comprehensive description and proof (both of correctness and complexity) of Regev's algorithm than given in Regev's paper. The goal of this thesis is to be accessible for readers who only know the basics of quantum computing and have no experience with lattices. Really only complexity theory and basic group theory and linear algebra are taken as assumed prior knowledge for the reader. Chapter 3 gives a description of Shor's algorithm. While it is not strictly necessary to fully understand Shor's algorithm to understand Regev's algorithm, it is helpful to see how exactly Regev's algorithm is similar to Shor's algorithm and to see where they differ. This comparison also shows exactly why the quantum subroutine in Regev's algorithm

[1] Efficient meaning polynomial time in the number of bits of the integer to be factored.

has a better time complexity than the subroutine in Shor's algorithm. In Chapter 4 we introduce lattices, we prove some basic properties of lattices and give many useful theorems that we will need in Chapter 5, the most important chapter in this thesis, describing and proving the correctness and complexity of Regev's algorithm. Much of the proofs in Chapter 5 follow the proofs given in Regev's paper, but we will be far more thorough, giving more explanations and intermediate steps, with the goal to make this thesis easier to follow than Regev's paper.

As of now, unconditional correctness of Regev's original algorithm has not been proven. We know that the algorithm does work, but its complexity is dependent on an unproven assumption, which is only suspected to hold true. However, a slight variation of Regev's algorithm, which is only slower than Regev's algorithm by a logarithmic factor, has been shown to be unconditionally correct by Cédric Pilatte [Pil24]. Another improvement to Regev's algorithm is given by Seyoon Ragavan and Vinod Vaikuntanathan [RV24], which shows that the space complexity of Regev's algorithm can be improved and it can be modified to torelate a certain fraction of the calls of the quantum subroutine to be incorrect. Regev's original algorithm uses $\mathcal{O}(n^{3/2}\log n)$ qubits compared to Shor's $\mathcal{O}(n)$. With this improvement Regev's algorithm would only need $\mathcal{O}(n\log n)$ qubits. Pilatte's unconditionally correct variation is also compatible with these improvements, only having a worse space complexity by a logarithmic factor. We will not go into detail about these improvements, as they are beyond the scope of this thesis.

2. Preliminaries

2.1. Quantum Computation

We model a quantum state as an element (vector) of a complex Hilbert space. Since we will only consider states with a finite amount of memory, we only consider the finite-dimensional complex vector space $\mathbb{C}^n$. We write a quantum state as a vector $|z\rangle$, which has norm 1. We can write a state as a complex-linear combination of some basis. The so-called computational basis states are

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Any single qubit state can be written as $\alpha_0|0\rangle + \alpha_1|1\rangle$, with $\alpha_0, \alpha_1 \in \mathbb{C}$ and $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

If we have multiple states $|z_1\rangle \in \mathbb{C}^n$ and $|z_2\rangle \in \mathbb{C}^m$, we can combine them into a single state $|z_1 z_2\rangle \in \mathbb{C}^{nm}$ defined by the tensor product $|z_1\rangle \otimes |z_2\rangle$. If we have the state

$$\sum_{i \in I} \alpha_i |i\rangle, \tag{2.1}$$

for some index set I where each $|i\rangle$ is a basis state and $\sum_{i \in I} |\alpha_i|^2 = 1$. There is no direct way of knowing what all the coefficients of a quantum state are. The only way to get output is by measuring (part of) a state. Measurement is probabilistic, it will give a result according to some probability distribution. For the purposes of this thesis we can see measurement as an operation that when applied to the state (2.1) returns the index i of the basis state $|i\rangle$ with probability $|\alpha_i|^2$. Every program on a quantum computing can be seen as a combination of multiplying states by unitary matrices and measuring. In general it is hard to apply some arbitrary unitary operation to a state, so we break it up into smaller "quantum gates", similarly to how we break up the computation on a classical computer into simple logic gates. The time complexity of a quantum algorithm is measured in the number of quantum gates used. Any computation that can be done on a classical computer can be done on a quantum computer, with at most the same time complexity [NC00, Section 4.5].

2.2. Quantum Fourier transform

Let n be a positive integer and set $N = 2^n$. We give the definition of the quantum Fourier transform as given in [NC00, p. 217]. The *quantum Fourier transform* with respect to an

orthonormal basis $|0\rangle, |1\rangle, \ldots, |N-1\rangle$ (corresponding to the elements of a cyclic group of order N) is the linear transformation with the following action on basis states:

$$\mathrm{QFT}|j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i jk/N} |k\rangle.$$

This is a unitary transformation, and therefore it can be implemented in a quantum computer.

In Regev's algorithm our orthonormal basis will be indexed by d-dimensional vectors instead of just the integers 0 through $N-1$. If z is a vector of d integers from $\{0, 1, \ldots, N-1\}$, we can write its corresponding basis state as

$$|z\rangle = |z_1, z_2, \ldots, z_d\rangle = \bigotimes_{j=1}^{d} |z_j\rangle.$$

Now we can apply the quantum Fourier transform with respect to $|0\rangle, |1\rangle, \ldots, |N-1\rangle$ to each coordinate z_i separately, giving

$$\begin{aligned}
\mathrm{QFT}|z\rangle &= \bigotimes_{j=1}^{d} \mathrm{QFT}|z_j\rangle \\
&= \bigotimes_{j=1}^{d} \left(\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i z_j k/N} |k\rangle \right) \\
&= \frac{1}{\sqrt{N^d}} \sum_{w \in \{0,\ldots N-1\}^d} e^{2\pi i \langle z,w\rangle / N} |w\rangle.
\end{aligned}$$

We can implement an approximate d-dimensional quantum Fourier transform with error $\epsilon > 0$ using $\mathcal{O}(dn(\log n + \log(1/\epsilon)))$ quantum gates [Cop02].

2.3. Quantum phase estimation

The quantum subroutine of Shor's algorithm is simply an application of the quantum phase estimation algorithm. Let U be a unitary operator with eigenstate $|u\rangle$, then this eigenstate has corresponding eigenvalue $e^{2\pi i \varphi_u}$. The quantum phase estimation gives an approximation to the phase φ_u. The algorithm consists of three parts: preparing the eigenstate, then applying the following transformation:

$$|j\rangle|u\rangle \mapsto |j\rangle U^j |u\rangle,$$

where $|j\rangle$ is a basis state. Finally we perform an inverse quantum Fourier transform.

The cost of each of these steps scale with the number of qubits we use. We already know that the (inverse) quantum Fourier transform costs $\mathcal{O}(n \log(n/\epsilon))$ quantum gates if we use n qubits. The complexity of the other two steps depend on what the unitary operator U is. If we want our phase estimation to be accurate to t bits with all but probability $\epsilon > 0$, we need $t + \left\lceil \log(2 + \frac{1}{2\epsilon}) \right\rceil$ qubits for our phase estimation [NC00, p. 221-225].

2.4. Notation and conventions

Before we begin, we explain some notation and conventions that might be non-standard or ambiguous.

2.4.1. Succeeding with "high probability"

Many steps in both Shor's algorithm and Regev's algorithm have some probability of failing. If we can quickly check if a subroutine has failed and if the probability of failure does not increase with larger inputs we say that a subroutine succeeds with high probability. Since for any arbitrary target probability of success, we can choose a constant number of times to retry the subroutine, which has no impact on the asymptotic complexity. This is equivalent to succeeding with probability $\Theta(1)$ while being able to detect errors efficiently.

2.4.2. Element of a quotient group

Often we take an element of a some quotient group G/H. In this thesis $x \in G/H$ means that x is a representative of the coset $x + H$. Unless otherwise specified and whenever this makes sense, it should be assumed that x is the "minimal non-negative" representative of its coset. For example if we write $x \in \mathbb{R}^d/\mathbb{Z}^d$, it should be assumed that $x \in [0, 1)^d$.

2.4.3. Distance to an integer point

For some $v \in \mathbb{R}^d$ we use the notation

$$\|v\|_{\mathbb{R}^d/\mathbb{Z}^d} \coloneqq \min_{u \in \mathbb{Z}^d} \|v - u\|$$

to denote the ℓ_2 distance of v to the nearest integer point. Despite the notation this is not a norm.

3. Shor's algorithm

In this chapter we describe Shor's algorithm for factoring an n-bit integer N and show that its quantum subroutine has complexity $\mathcal{O}(n^2 \log n)$ and the rest of the algorithm is polynomial in n. In Chapter 5 we will see that Regev's algorithm can be seen as a higher dimensional analogue of Shor's algorithm and it has the same general structure.

First we describe the reduction that Shor uses to reduce the problem of factoring N to finding the order of an arbitrary element of the multiplicative group $(\mathbb{Z}/N\mathbb{Z})^\times$. The reduction in Regev's algorithm will almost be a direct generalization of this idea to d dimensions.

Then we describe how modular exponentiation is done efficiently, this is important as it is the bottleneck of Shor's quantum subroutine. The equivalent part in Regev's algorithm is the place where the time complexity is saved.

Then we show how quantum phase estimation can be used in combination with some classical post-processing to solve the order-finding problem efficiently. While the corresponding steps in Regev's algorithm are much more involved, they still have roughly the same structure as in Shor's algorithm.

3.1. Reduction of factoring to order finding

The part of Shor's algorithm that requires the use of a quantum computer is efficiently finding the order of an element of $(\mathbb{Z}/N\mathbb{Z})^\times$, the multiplicative group of integers modulo N. In particular, if we have an n-bit integer N and some element $a \in (\mathbb{Z}/N\mathbb{Z})^\times$, we would like to find the minimal $r \in \mathbb{Z}_{>0}$ such that $a^r = 1$ in a time complexity that is polynomial in n. This is the only part of Shor's algorithm for which there is no known efficient classical algorithm. Before getting into how this problem can be solved using a quantum computer, we assume that we can call some polynomial time subroutine to give the order of any element in $(\mathbb{Z}/N\mathbb{Z})^\times$. In this section we show that we can use such a subroutine to efficiently find a non-trivial factor of N. Thus reducing a problem of factoring to a problem of order finding.

Since even numbers trivially have 2 as a factor, we are allowed to assume that N is odd. Moreover, if N is a prime power, then it can be efficiently factored by calculating roots and checking if they are integral, so we may assume N is not a prime power. We fix some (random) $a \in \{2, \ldots, N-1\}$. If $\gcd(a, N)$ is a non-trivial factor of N, then we are done, since computing the greatest common divisor of two n-bit number can be done efficiently using the Euclidean algorithm, only costing $\mathcal{O}(n^2)$ time. In the (far more likely) case that a and N are coprime, we have $a \in (\mathbb{Z}/N\mathbb{Z})^\times \setminus \{1\}$. Now we can

apply the order finding subroutine to find the order r of a. This gives

$$a^r - 1 \equiv 0 \bmod N. \tag{3.1}$$

The goal is to use the difference of squares identity to rewrite (3.1) in a way that it gives a non-trivial factor of N. This will not always work, in particular we need the order to be even, and we need $a^{r/2} \not\equiv -1 \bmod N$. Fortunately we can easily check if these conditions hold and, due to the following lemma, there is a good probability that a random a will satisfy these conditions.

LEMMA 3.1.1. [NC00, Thm. A4.13]. Let $N \in \mathbb{Z}_{>0}$ be odd and not a prime power. With probability at least $1/2$ over the choice of a from $(\mathbb{Z}/N\mathbb{Z})^\times \setminus \{1\}$, the order r of a is even and $a^{r/2} \not\equiv -1 \bmod N$.

We simply retry with a new random a until the conditions of Lemma 3.1.1 are satisfied. Once we have found a suitable a, the following theorem shows how we can find a non-trivial factor of N.

THEOREM 3.1.2. Let $N \in \mathbb{Z}_{>0}$ be odd and not a prime power. Let $a \in (\mathbb{Z}/N\mathbb{Z})^\times$ with order r satisfy the conditions of Lemma 3.1.1. Then at least one of $\gcd(N, a^{r/2}+1)$ and $\gcd(N, a^{r/2}-1)$ is a non-trivial factor of N.

Proof. We can recognize a difference of squares in (3.1):

$$a^r - 1 = (a^{r/2})^2 - 1^2 = (a^{r/2}-1)(a^{r/2}+1) \equiv 0 \bmod N,$$

which implies $(a^{r/2}-1)(a^{r/2}+1) = kN$ for some $k \in \mathbb{Z}_{>0}$. Because of the conditions of Lemma 3.1.1 and the fact that $r/2$ is strictly less than the order of a we have $a^{r/2} \not\equiv \pm 1 \bmod N$. This implies that N cannot divide either of the factors $a^{r/2}-1$ and $a^{r/2}+1$. This means that each of $\gcd(N, a^{r/2}-1)$ and $\gcd(N, a^{r/2}+1)$ are either 1, or a non-trivial factor of N. If both of the factors are coprime to N, their product kN must also be coprime to N, which it is clearly not. Thus, at least one of $\gcd(N, a^{r/2}-1)$ and $\gcd(N, a^{r/2}+1)$ must be a non-trivial factor of N. □

We simply compute both $\gcd(N, a^{r/2}-1)$ and $\gcd(Na^{r/2}+1)$, check which is a non-trivial factor, and return it.

3.2. Modular exponentiation

Our goal will be to efficiently find the order of some element $a \in (\mathbb{Z}/N\mathbb{Z})^\times$, where N is an n-bit integer. In other words, we want to find the smallest positive period of the function

$$\mathbb{Z} \to (\mathbb{Z}/N\mathbb{Z})^\times, \quad x \mapsto a^x \bmod N.$$

This function is called *modular exponentiation*. Before we can find its period we must be able to compute the function. It is important to understand how we efficiently compute it for high N, since this computation happens to be the most expensive part of Shor's algorithm. We need to compute powers high enough to find the order, which can be as high as $N-1$, which is bounded from above by 2^n since N is an n-bit integer.

Powers of integers are computed using many multiplications. The cost of multiplying two integers is a function of the amount of bits needed to represent them. It turns out that the fastest known algorithm to multiply two n-bit numbers has a time complexity of $\mathcal{O}(n \log n)$ [HvdH21].

REMARK 3.2.1. The algorithm to multiply two n-bit numbers in $\mathcal{O}(n \log n)$ time, known as "Fast Integer Multiplication", is asymptotically the fastest algorithm to multiply integers. However, the point at which it becomes efficient to use Fast Integer Multiplication uses numbers that are many orders of magnitude larger than what would be used in practice. A naive long multiplication algorithm takes $\mathcal{O}(n^2)$ operations, and there exist several algorithms achieving slightly better complexities. We will assume the use of Fast Integer Multiplication for the time complexities in this thesis, but actual implementations of these algorithms would achieve slightly worse time complexities. The complexities of both Shor's and Regev's algorithm assume the use of Fast Integer Multiplication and using a different multiplication algorithm does not change the fact that the quantum subroutine in Regev's algorithm has a better time complexity than the quantum subroutine in Shor's algorithm.

The naive way to calculate exponentiation is just to repeatedly multiply by a. If we wanted to calculate a^{2^n} this would mean doing around 2^n multiplications on numbers that can be up to $N-1$ in size (which is an n-bit number), since we are working modulo N. Applying the modulo itself is not expensive, as this just costs a division which is asymptotically as expensive as a multiplication. This gives a complexity of $\mathcal{O}(2^n n \log n)$, which is too expensive.

There is a much faster way of calculating high exponents using a technique known as *repeated squaring*. First observe that we can calculate any n-bit integer efficiently if we start from 1 and only use two kinds of operations: doubling and incrementing by one. This is easy to see from the binary representation of an integer. Namely, doubling appends a 0 and incrementing an even number by one changes the least significant bit from a 0 to a 1. This means we can append a 0 to binary representation of a number in one operation and append a 1 in two operations, by first doubling and then incrementing by one. Thus it only takes $\mathcal{O}(n)$ doubling and incrementing operations to produce any n-bit number. We can easily translate this method to calculating exponents, since doubling and incrementing an exponent can both be done with a multiplication. Incrementing the exponent is simply multiplying by the base number, and doubling the exponent corresponds to squaring, hence the name repeated squaring. Since both of those operations are just a multiplication between n-bit integers, this method only requires $\mathcal{O}(n)$ multiplications between n-bit integers. This gives a time complexity of $\mathcal{O}(n^2 \log n)$. We finish this section with an example that illustrates how the repeated

squaring technique is performed.

EXAMPLE 3.2.2. Say we want to calculate 3^{19} mod 26. If we want to build 19 efficiently with doubling and incrementing, we can look at its binary representation, which is 10011. Now we can build the binary exponent 10011 with doubling and incrementing as shown on the left, and apply the corresponding operations on the right.

$$\begin{array}{ll} 0 & \\ \quad\downarrow +1 & \\ 1 & 1 \bmod 26 \\ \quad\downarrow \times 2 & \quad\downarrow \times 3 \\ 10 & 3 \bmod 26 \\ \quad\downarrow \times 2 & \quad\downarrow \textit{square} \\ 100 & 9 \bmod 26 \\ \quad\downarrow \times 2 & \quad\downarrow \textit{square} \\ 1000 & 3 \bmod 26 \\ \quad\downarrow +1 & \quad\downarrow \textit{square} \\ 1001 & 9 \bmod 26 \\ \quad\downarrow \times 2 & \quad\downarrow \times 3 \\ 10010 & 1 \bmod 26 \\ \quad\downarrow +1. & \quad\downarrow \textit{square} \\ 10011 & 1 \bmod 26 \\ & \quad\downarrow \times 3. \\ & 3 \bmod 26 \end{array}$$

So we calculated 3^{19} mod 26 to be 3.

3.3. The quantum subroutine

This section is based on [NC00, Section 5.3]. We would like a quantum subroutine that takes as an input some $a \in (\mathbb{Z}/N\mathbb{Z})^\times$ and returns the order r of a. The algorithm shown in this section will not directly give the order r, but an approximation of the value s/r, where s is a sample taken from a uniform distribution of $\{0, \ldots, r-1\}$. In the next section we show that a constant number of calls to such a subroutine suffices to find the order r with high probability.

Fix some $a \in (\mathbb{Z}/N\mathbb{Z})^\times$ of which we want to find the order. We can create a unitary matrix U_a that satisfies

$$U_a|x\rangle = |ax \bmod N\rangle. \tag{3.2}$$

Note that we have $U_a^k|x\rangle = |a^k x \bmod N\rangle$. Say $|u\rangle$ is some eigenvector of the unitary matrix U_a with corresponding eigenvalue $e^{2\pi i\varphi}$. If r is the order of a, then $e^{2\pi ir\varphi}|u\rangle = U_a^r|u\rangle = |u\rangle$, so φr must be an integer. Since the phase is in the interval $[0, 1)$, we know that $\varphi = s/r$ for some $s \in \{0, \ldots, r-1\}$.

If we already know the order r we can explicitly find all the eigenvectors. Define for $s \in \{0, \ldots, r-1\}$ the state[1]

$$|u_s\rangle := \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi isk/r} \left|a^k \bmod N\right\rangle.$$

[1] This state of course also depends on a, but we don't write this to avoid a double subscript.

This is an eigenvector of U_a with phase s/r. We can verify this by calculating

$$\begin{aligned} U_a|u_s\rangle &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi isk/r} \left| a^{k+1} \bmod N \right\rangle \\ &= \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi is(k-1)/r} \left| a^{k} \bmod N \right\rangle \\ &= e^{2\pi is/r}|u_s\rangle. \end{aligned}$$

Obviously we don't know r yet, so we cannot prepare a state $|u_s\rangle$. Fortunately, we can prepare a superposition of the $|u_s\rangle$ states over all possible s values without knowing what the order r is:

LEMMA 3.3.1. Defining the $|u_s\rangle$ state as above, the equality $\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle$ holds. In particular this does not depend on r.

Proof. Simply filling in gives

$$\begin{aligned} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle &= \frac{1}{r} \sum_{s=0}^{r-1} \sum_{k=0}^{r-1} e^{-2\pi isk/r} \left| a^{k} \bmod N \right\rangle \\ &= \frac{1}{r} \sum_{k=0}^{r-1} \left(\sum_{s=0}^{r-1} e^{-2\pi isk/r} \right) \left| a^{k} \bmod N \right\rangle. \end{aligned}$$

In the case that $k = 0$, the inner sum becomes r. In the case that $k \neq 0$, we can recognize a geometric sum giving

$$\sum_{s=0}^{r-1} e^{-2\pi isk/r} = \frac{1 - e^{-2\pi ik}}{1 - e^{-2\pi ik/r}} = 0.$$

So we get

$$\begin{aligned} \frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle &= \frac{1}{r} \sum_{k=0}^{r-1} r\delta_{k,0} \left| a^{k} \bmod N \right\rangle \\ &= \left| a^{0} \bmod N \right\rangle \\ &= |1\rangle. \end{aligned}$$

□

Also note that $U_a^j|1\rangle = \left|a^j \bmod N\right\rangle$, which we know how to compute efficiently from the discussion in Section 3.2, so we can perform each part of the phase estimation efficiently. Applying phase estimation (Section 2.3) with $|1\rangle$ as our "eigenstate", the result will be the superposition $\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |\varphi_s\rangle$, where $\varphi_s \approx s/r$ with high probability. Thus applying phase estimation on $|1\rangle$ and measuring gives an estimation of some s/r, where s is uniformly chosen from $\{0, \ldots, r-1\}$.

3.4. Recovering the order using continued fractions

We are given an integer $N < 2^n$. There exists a positive integer $r < N$ and we have access to some subroutine that can give an approximation φ_s of s/r, for some uniformly random $s \in \{0, \ldots, r-1\}$ (sampled each time we run the subroutine) and we want to find the value r efficiently.

Continued fractions can be used to get the best rational approximation to a real number given a bound for the denominator. Because we know that our phase must be a rational number with denominator lower than N, if φ_s is an accurate enough approximation of s/r, we can guarantee that s/r is such a good rational approximation of φ_s that a continued fraction expansion must find it. First we give an example to show how continued fraction expansions work:

EXAMPLE 3.4.1. Say we get the rational number $32/155$ as our approximation of s/r (the approximation will always have finite precision, so it will always be rational). Let's also say we somehow know that r is at most 5. We can do the following to approximate $32/155$:

$$\frac{32}{155} = \frac{1}{\frac{155}{32}} = \frac{1}{4 + \frac{27}{32}} \approx \frac{1}{4}.$$

So we write a fraction as the reciprocal of its inverse, then rewrite the denominator as 1 plus a fraction less than 1. We get an approximation by simply removing the final fraction. We can apply the same trick to $27/32$, which gives an even better approximation:

$$\frac{32}{155} = \frac{1}{4 + \frac{27}{32}} = \frac{1}{4 + \frac{1}{1 + \frac{5}{27}}} \approx \frac{1}{4 + \frac{1}{1}} = \frac{1}{5}.$$

Clearly the denominators will only get bigger if we continue this, so (assuming $32/155$ was an accurate enough approximation of s/r) we can conclude that $s/r = 1/5$, so r must be a multiple of 5. In this case, since we assumed $r \leq 5$, we can even conclude that $r = 5$.

We write $[a_1, \ldots, a_k]$, with $a_i \in \mathbb{Z}_{>0}$ to denote the number

$$\frac{1}{a_1 + \frac{1}{\cdots + \frac{1}{a_k}}}.$$

We also define $a_0 = 0$. Let $x = [a_1, \ldots, a_k]$, we then call $[a_1, \ldots, a_m]$ with $m \leq k$ the m-th convergent of x.

Given a rational number in $[0, 1)$ we can find the numerator and denominator in lowest terms of its m-th convergent (along with all previous convergents) in only $\mathcal{O}(m)$ multiplications using recursive formulas [HW79, p. 131, Thm. 149 & 157]. In particular the recursion for the denominator of the m-th convergent is

$$q_m = a_m q_{m-1} + q_{m-2}, \tag{3.3}$$

with base cases $q_0 = 1$ and $q_1 = a_1$. This immediately implies $q_m \geq 2q_{m-2}$, so the denominators increase exponentially. The following theorem will reveal how continued fraction will help us find the order.

THEOREM 3.4.2. Say p/q is a rational number written in lowest terms, and x is a real number such that

$$\left|\frac{p}{q} - x\right| \leq \frac{1}{2q^2}.$$

Then there exists an m such that p/q is the m-th convergent of x. This is the best rational approximation of x with denominator at most q. Moreover, m is bounded from above by $2\log_2(q) + 1$.

Proof. The first two claims are proven in [HW79, Thm. 184] and [HW79, Thm. 181] respectively. All that remains is to prove the bound on m. Assume, for a contradiction, that $m > 2\log_2(q) + 1$, this is equivalent to $2^{m/2-1} > q$. Because of (3.3) we know that $q_m \geq 2^{m/2-1}q_0$, (the -1 is added in case m is odd). We also know that $q_0 = 1$, giving the inequality

$$q_m \geq 2^{m/2-1} > q.$$

This contradicts that the m-th convergent of x equals p/q, since q_m is the denominator of a fraction in lowest terms. Thus $m \leq 2\log_2(q) + 1$. □

The above theorem shows that we only need to calculate the first $\mathcal{O}(\log r)$ convergents of the approximation φ_s, the following corollary shows how we can easily identify which convergent exactly equals s/r.

COROLLARY 3.4.3. If $s, r \in (\mathbb{Z}/N\mathbb{Z})^\times$ and φ_s is an approximation of s/r such that

$$\left|\frac{s}{r} - \varphi_s\right| \leq \frac{1}{2N^2},$$

then s/r is the unique convergent of φ_s with denominator less than N and error at most $1/(2N^2)$. This will be one of the first $2\log_2(N) + 1$ convergents.

Proof. The existence of a convergent of φ_s equalling s/r and the bound on the convergent are direct consequences of Theorem 3.4.2, using $r < N$. Let p/q be a rational number different from s/r such that $q < N$. This implies

$$\left|\frac{s}{r} - \frac{p}{q}\right| = \frac{|sq - pr|}{rq} > \frac{1}{N^2}.$$

Now the triangle inequality states that as an approximation of φ_s, the fraction p/q has an error greater than $\frac{1}{2N^2}$,

$$\left|\frac{p}{q} - \varphi_s\right| \geq \left|\frac{s}{r} - \frac{p}{q}\right| - \left|\frac{s}{r} - \varphi_s\right| > \frac{1}{2N^2}.$$

Thus proving that there is no rational other than s/r with denominator less than N that approximates φ_s with error at most $1/(2N^2)$, so in particular s/r is the unique convergent with these properties. □

If s and r happen to be coprime, it suffices to simply find the denominator of the convergent that equals s/r, since this must then equal r. However, if s and r are not coprime the resulting denominator will give a factor of r. We can take two samples of the quantum subroutine, giving two phases s_1/r and s_2/r. When these fractions are reduced they give two guesses for the order, r_1 and r_2. Then the probability that $\text{lcm}(r_1, r_2) = r$ is at least $1/4$ [NC00, p. 231], and this chance increases when you take more samples, importantly the probability does not decrease as N increases. Now we have an algorithm that, with high probability, gives us the order. Of course we can easily check if it was succesful and run it again if it was not. Now that we can solve the order finding problem we can also solve the factoring problem by the reduction described in Section 3.1.

3.5. Putting it all together

We want to factor an n-bit integer N. We start by choosing a random non-trivial $a \in (\mathbb{Z}/N\mathbb{Z})^\times$ and finding its order. To do this, we apply phase estimation on the unitary U_a, as defined in (3.2), with vector $|1\rangle$, which is a superposition of eigenvectors. We need the error of our phase estimation to be at most $1/(2N^2)$ to use Corollary 3.4.3. This means we need the phase estimate to be accurate to at least $2n+1$ bits. So we will need $2n+1+\lceil \log(2+1/(2\epsilon)) \rceil$ qubits for our phase estimation to get enough accuracy with probability $1-\epsilon$ (Section 2.3). As described in Section 3.2 we need $\mathcal{O}(n^2 \log n)$ gates for modular exponentiation and we need $\mathcal{O}(n \log n)$ gates for the rest of the phase estimation. Adding all these complexities, the entire quantum subroutine costs $\mathcal{O}(n^2 \log n)$ gates. This subroutine returns $\varphi_s \approx s/r$, for some random $s \in (\mathbb{Z}/N\mathbb{Z})$.

Now according to Corollary 3.4.3 if we calculate the first $\mathcal{O}(n)$ convergents of the continued fraction expantion of φ_s, then the the last convergent with denominator less than N is a factor of r. Finding the convergents should only take $\mathcal{O}(n)$ multiplications, which is $\mathcal{O}(n^2 \log n)$ time. We call the subroutine some constant number of times and take the lowest common multiple of the results. With high probability we should now have the order, we can easily check if the order is correct, if it is not, we simply run the order finding algorithm again. If r is odd or if $a^{r/2} \equiv -1 \bmod N$, then we retry with a another random non-trivial a. When we have an a with order r such that $a^{r/2} \not\equiv -1 \bmod N$, we calculate $\gcd(N, a^{r/2}-1)$ and $\gcd(N, a^{r/2}+1)$, according to Theorem 3.1.2 at least one of these is a non-trivial factor of N. In total we have found a non-trivial factor of N, with high probability, in only $\mathcal{O}(n^2 \log n)$ time.

4. Lattices

The order finding in Shor's algorithm is a special case of finding periods of periodic functions. Regev's algorithm generalizes this to higher dimensions using lattices. In this chapter we define lattices, give the proofs of some elementary properties of lattices. The quantum subroutine of Regev's algorithm applies a quantum Fourier transform to a Gaussian superposition over a lattice, to understand this we introduce Fourier analysis and Gaussians on lattices in this chapter. Lastly we introduce the Lenstra-Lenstra-Lovász algorithm, which is required for the classical post-processing of Regev's algorithm.

4.1. Introduction

Before introducing lattices we give a precise definition for what we mean by a period.

DEFINITION 4.1.1 (PERIOD). Let f be a function whose domain is an abelian group $(A, *)$. An element $x \in A$ is said to be a *period* of f if for all $a \in A$ the equality

$$f(a * x) = f(a)$$

holds. A function that has a non-trivial period is said to be *periodic*.

The reader is likely already familiar with many periodic functions. Examples include functions from $\mathbb{Z}$ to $\mathbb{Z}/n\mathbb{Z}$ defined by $a \mapsto a \bmod n$, these trivially have a period of n. Examples of functions with a continuous domain are also commonplace, functions like sin and cos have periods of 2π. Note that every multiple of a period is itself a period. We can define a set containing all the periods of a function, for the first function we would have

$$\{x \in \mathbb{Z} \mid x \text{ is a period of } a \mapsto a \bmod n\} = n\mathbb{Z}.$$

Note that this is a subgroup of our domain.

What all these examples have in common is that they only take a single input, but nothing in our definition requires this. We could let the domain be, for example, $\mathbb{R}^2$ or $\mathbb{Z}^d$ to define periodicity in functions with many inputs. A simple example of a periodic function with multiple inputs is $f\colon \mathbb{Z}^2 \to \mathbb{Z}/2\mathbb{Z}$, given by $(a, b) \mapsto a + b \bmod 2$. It is easily verified that both $(2, 0)$ and $(1, 1)$ are periods of f. This example reveals a

meaningful difference from the case of periodic functions from $\mathbb{Z}$ or $\mathbb{R}$, we can have periods that are not multiples of each other:

$$\{(x,y) \in \mathbb{Z}^2 \mid (x,y) \text{ is a period of } f\} = \{a(2,0) + b(1,1) \mid a,b \in \mathbb{Z}\} = \begin{pmatrix} 2 & 1 \\ 0 & 1 \end{pmatrix} \mathbb{Z}^2.$$

Every set of periods will follow a similar pattern, containing all the $\mathbb{Z}$-linear combinations of some "basis" periods. This naturally gives rise to the definition of a *lattice*, the algebraic object we will use to study periodic functions.

DEFINITION 4.1.2 (LATTICE). Given linearly independent vectors $b_1, b_2, \ldots, b_d \in \mathbb{R}^d$, the *lattice* $\mathcal{L}$ generated by $b_1, b_2, \ldots, b_d$ is defined as

$$\mathcal{L} = \Big\{ \sum_{i=1}^{d} x_i b_i \mid x_i \in \mathbb{Z} \Big\}.$$

We call $b_1, b_2, \ldots, b_d$ *basis vectors* of $\mathcal{L}$. Equivalently, letting $B \in \mathrm{GL}(d, \mathbb{R})$ we may write

$$\mathcal{L}(B) := B\mathbb{Z}^d.$$

We refer to B as a *basis* of $\mathcal{L}(B)$ [Reg04, Lecture 1, Def. 1].

We can visually represent a lattice as a set of points in the plane, as shown in Figure 4.1.

REMARK 4.1.3. Note that we require a basis of a lattice to have as many basis vectors as its dimension. This is because we will only be considering lattices with this property, which are usually called *full rank* lattices [Reg04]. Every lattice in this thesis is assumed to be of full rank.

The idea to use lattices in cryptography is not new. Although we will use lattices to efficiently factor large numbers, and thus breaking encryption schemes like RSA, they are also widely used to do the exact opposite: creating stronger encryption schemes that are thought to be secure against quantum computers. This is because many problems related to lattices are computationally difficult, such as finding the shortest vector in a lattice given some basis, or finding the basis with the smallest vectors generating the same lattice.

Even though finding the shortest vector in a lattice is a computationally hard problem, there exist algorithms that can find approximate solutions, that is, vectors that are "reasonably" close in length to the shortest vector. An algorithm that does this is the Lenstra-Lenstra-Lóvasz (LLL) algorithm, which has been used as a subroutine in many computational problems, such as efficiently factoring polynomials, or finding minimum polynomials given good enough approximations of algebraic numbers [Reg04, Lecture 2]. The LLL algorithm will also make an appearance later on in this thesis, to help us find vectors from our lattice that are not too large.

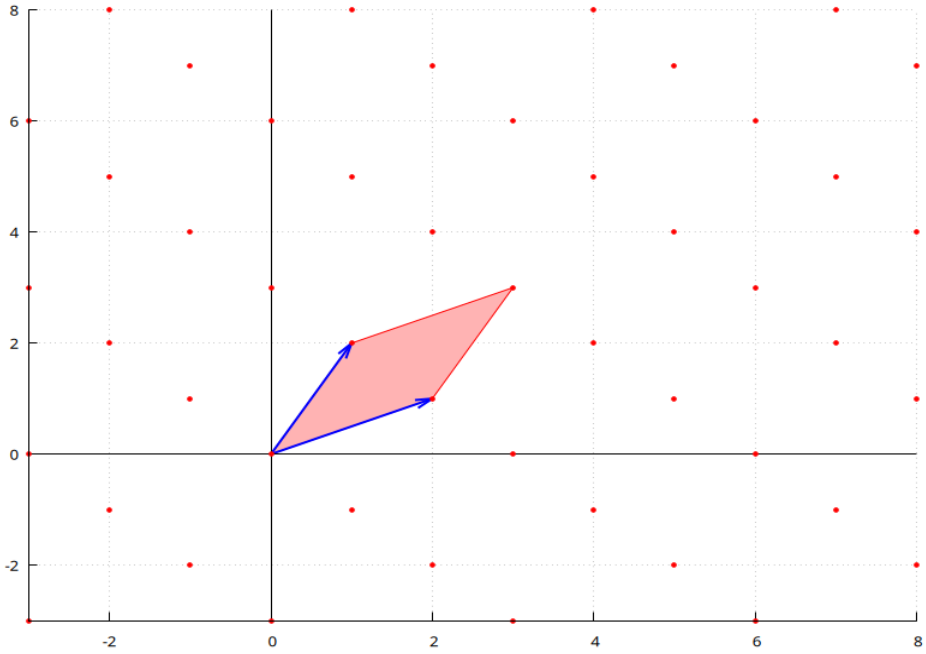


Figure 4.1.: 2-dimensional lattice generated by the basis $(1,2),(2,1)$ with the fundamental parallelotope highlighted (See Definition 4.2.1).

Lattices provide a way to design a higher dimensional variant of Shor's algrotihm. The part of Shor's algorithm that cannot be done efficiently on a classical computer is using the quantum Fourier transform to find periods. In particular we want to find the period of the modular exponentiation function $z \mapsto a^z \mod N$, where N is the number we want to factorise, and $a \in (\mathbb{Z}/N\mathbb{Z})^\times$. This is the same as finding the multiplicative order of the group element a, although we will no longer use this termininology as this doesn't make sense when we generalize to higher dimensions. The d-dimensional analogue to modular exponentiation is

$$(z_1, z_2, \ldots, z_d) \mapsto \prod_{i=1}^{d} a_i^{z_i} \bmod N.$$

The periods of this function will be a subgroup of $\mathbb{Z}^d$ forming a lattice $\mathcal{L}$. We call a function that has a lattice $\mathcal{L}$ as its set of periods $\mathcal{L}$-periodic. Similarly to Shor's algorithm, we can use a quantum Fourier transform to find periods of this function, although this becomes slightly more complicated in higher dimensions, as will be explored in depth in Chapter 5.

The benefit of going to higher dimensions is due to some fundamental properties of lattices proved by Minkowski, which will be explored in the next section of this chapter. They essentially put a bound on how large the samples of this function need to be to be able to find a period. This means we don't have to perform as many large integer mul-

tiplications in superposition before applying the quantum Fourier transform, which is the main bottleneck in Shor's algorithm.

4.2. Minkowski's theorem

The main goal of Regev's algorithm is to find a non-trivial period of a function with multiple inputs, this is equivalent to finding a non-zero vector in a lattice. The benefit of higher dimensional periods is that we don't have to look as "far" to find a period. To make this more precise, we prove an upper bound on the shortest (according to the standard norm on $\mathbb{R}^d$) non-zero vector in a lattice. Before we state this theorem we will need an exact notion of the intuitive idea of the "density" of a lattice, this is given by the *fundamental parallelotope* and the *determinant* of a lattice:

Definition 4.2.1 (Fundamental parallelotope). For any d-dimensional lattice basis B we define the *fundamental parallelotope* as

$$\mathcal{P}(B) \coloneqq \{Bx \mid x \in [0,1)^d\}.$$

See Figure 4.1 for an example a fundamental parallelotope of a lattice.

Since a d-dimensional lattice $\mathcal{L}$ with basis matrix B is a subgroup of the abelian group $\mathbb{Z}^d$, we may consider the quotient group $\mathbb{Z}^d/\mathcal{L}$. When taking a representative $x \in \mathbb{Z}^d/\mathcal{L}$, as discussed in Subsection 2.4.3, we take the "minimal non-negative" representative. In this case this means that $x \in \mathcal{P}(B)$.

Notice that if you see a lattice vector as a line segment, no lattice vector fits completely in the fundamental parallelotope without rotating it. We will show that any set of larger volume cannot have this property, showing that a smaller fundamental parallelotope corresponds to a "denser" lattice. Another useful definition will be the *determinant* of a lattice:

Definition 4.2.2 (Determinant). For a lattice $\mathcal{L}$ with basis B, the *determinant* of $\mathcal{L}$, denoted by $\det \mathcal{L}$, is defined as the volume of the fundamental parallotope of B.

If you are familiar with the notion of a determinant in linear algebra, you might be able to see that in a lattice $\mathcal{L}$ with basis B, we have $\det \mathcal{L} = |\det B|$, justifying the use of the term "determinant". It is a routine exercise to prove that the determinant of a lattice is independent of the choice of basis, and thus well-defined [Reg04, Lecture 1, p. 4]. We now have the vocabulary to state the upper bound as a theorem:

Theorem 4.2.3 (Minkowski's first theorem). For any d-dimensional lattice $\mathcal{L}$, let λ be the norm of its shortest non-zero vector. The following upper bound holds:

$$\lambda \leq \sqrt{d}(\det \mathcal{L})^{1/d}.$$

In short, the strategy we will use to prove this is to take the open ball of radius λ, centered at the origin, and prove some bounds on the volume of this ball. First we show that we can fit a lattice vector into any set of volume greater than the determinant of the lattice:

LEMMA 4.2.4 (BLICHFELDT). For any d-dimensional lattice $\mathcal{L}$ and (measurable) set $S \subseteq \mathbb{R}^d$ with $\operatorname{vol} S > \det \mathcal{L}$, there exist two non-equal points $z_1, z_2 \in S$, such that $z_1 - z_2 \in \mathcal{L}$.

Proof. Choose a basis B of $\mathcal{L}$, and let $\mathcal{P}(B)$ be its fundamental parallelotope. Each vector $v \in \mathbb{R}^d$ can be written uniquely as $u + w$, where $u \in \mathcal{L}$ and $w \in \mathcal{P}(B)$, this can be seen by setting $u = \lfloor v \rfloor_B$ and $w = v - u$, where $\lfloor \cdot \rfloor_B$ is the component-wise floor function in the basis B. This implies that the sets $u + \mathcal{P}(B) = \{u + w \mid w \in \mathcal{P}(B)\}$ form a partition of $\mathbb{R}^d$ as u ranges over $\mathcal{L}$. Define the map

$$\varphi \colon \mathbb{R}^d \to \mathcal{P}(B) \colon v \mapsto v - \lfloor v \rfloor_B .$$

This can also be seen as $u + w \mapsto w$ with u, w as above. Since $\operatorname{vol} S > \det \mathcal{L} = \operatorname{vol} \mathcal{P}$, the restriction $\varphi|_S$ cannot be injective, so there must exist two non-equal $z_1, z_2 \in S$ such that $z_1 - \lfloor z_1 \rfloor_B = z_2 - \lfloor z_2 \rfloor_B$. This gives

$$z_1 - z_2 = \lfloor z_1 \rfloor_B + (z_1 - \lfloor z_1 \rfloor_B) - (\lfloor z_2 \rfloor_B + (z_2 - \lfloor z_2 \rfloor_B)) = \lfloor z_1 \rfloor_B - \lfloor z_2 \rfloor_B \in \mathcal{L},$$

as desired. □

This lemma gives us an upper bound on a ball around the origin containing no non-zero lattice vectors:

LEMMA 4.2.5. Let $\mathcal{L}$ be a d-dimensional lattice. Let $\mathcal{B} \subset \mathbb{R}^d$ be an open ball of radius r, centered at the origin. Assume $\mathcal{B}$ contains no non-zero lattice vectors of $\mathcal{L}$. The following inequality holds:

$$\operatorname{vol} \mathcal{B} \leq 2^d \det \mathcal{L}.$$

Proof. Assume, for a contradiction, that $\operatorname{vol} \mathcal{B} > 2^d \det \mathcal{L}$. Consider the smaller open ball $\mathcal{B}/2 := \{x \mid 2x \in \mathcal{B}\}$. Since this is a d-dimensional object we have $\operatorname{vol} \mathcal{B}/2 = 2^{-d} \operatorname{vol} \mathcal{B}$. Its volume would thus be greater than $\det \mathcal{L}$. By Blichfeldt's lemma (4.2.4) there must exist two non-equal $z_1, z_2 \in \mathcal{B}/2$ such that $z_1 - z_2 \in \mathcal{L}$. The triangle inequality gives

$$\|z_1 - z_2\| \leq \|z_1\| + \|z_2\| < r/2 + r/2 = r,$$

meaning that $z_1 - z_2 \in \mathcal{B}$. This, however, contradicts our assumption that $\mathcal{B}$ does not contain any non-zero lattice vectors of $\mathcal{L}$. Thus $\mathcal{B}$ must have volume less than or equal to $2^d \det \mathcal{L}$, as desired. □

Finally, we obtain a lower bound the volume of a ball by some elementary geometry :

LEMMA 4.2.6. The volume of an d-dimensional ball of radius r is at least $(2r/\sqrt{d})^d$.

Proof. Assume without loss of generality that this ball is centered at the origin. We show the inequality by fitting the largest possible hypercube in the ball. Consider the d-dimensional hypercube of sidelength $2r/\sqrt{d}$ centered at the origin. Letting e_i denote the i-th standard basis vector, we can compute the distance of a corner of the hypercube to the origin as follows:

$$\left\| \sum_{i=1}^{d} \frac{r}{\sqrt{d}} e_i \right\| = \sqrt{\sum_{i=1}^{d} \frac{r^2}{d}} = r.$$

Thus this hypercube fits entirely in an d-dimensional ball of radius r, meaning the volume of the ball must be at least $(2r/\sqrt{d})^d$, as desired. □

Now we have all the necessary ingredients to prove the upper bound on the smalles non-zero vector of a lattice.

Proof of Minkowski's First Theorem (4.2.3). Let $\mathcal{B}$ be the open ball of radius λ. Since λ is by definition the norm of the smallest non-zero lattice vector of $\mathcal{L}$, the ball $\mathcal{B}$ cannot contain any non-zero lattice vectors. Now lemma's (4.2.5) and (4.2.6) give the inequality

$$\left(\frac{2\lambda}{\sqrt{d}}\right)^d \leq \text{vol}(\mathcal{B}) \leq 2^d \det \mathcal{L}.$$

Rearranging gives

$$\lambda \leq \sqrt{d}(\det \mathcal{L})^{1/d},$$

as desired. □

4.3. Dual Lattices, integer lattices and sublattices

If you recall, the quantum subroutine in Shor's algorithm used to find a period does not directly find the period (Section 3.3). Instead it returns some integer divided by the period. In other words it returns some element x such that any product of x with a period is integral. Something similar happens in higher dimensions, where we use the standard inner product as a generalization of the product. These are elements of the so-called "dual lattice".

DEFINITION 4.3.1 (DUAL LATTICE). For a lattice $\mathcal{L}$ we define its *dual lattice*

$$\mathcal{L}^* = \{y \in \mathbb{R}^d \mid \text{for all } x \in \mathcal{L}, \langle x, y \rangle \in \mathbb{Z}\}.$$

As the name suggests, the dual lattice is also a lattice. This is easily verified by elementary properties of the inner product. The dual of the dual lattice is the original lattice.

The dual lattice is sometimes called the *reciprocal lattice*, the following theorem explains why this is a fitting name:

THEOREM 4.3.2. [Reg04, Lecture 8, Claim 3]. For any lattice $\mathcal{L}$, the following equality holds:

$$\det(\mathcal{L}^*) = \frac{1}{\det(\mathcal{L})}.$$

One of the simplest and most important lattices is $\mathbb{Z}^d$. It is known as the d-dimensional *integer lattice*, since it contains all integer points of $\mathbb{R}^d$. Some basic properties of $\mathbb{Z}^d$ which are trivial to verify are that it has determinant 1 and that it is its own dual.

Since we will only be looking at periods of functions with discrete domains (like modular exponentiation) that only have integral periods. The lattices of these functions will be be subgroups of the integer lattice, which we call a sublattice:

DEFINITION 4.3.3 (SUBLATTICE). If $\mathcal{M}$ is a d-dimensional lattice, and $\mathcal{L} \subset \mathcal{M}$ is also a d-dimensional lattice, we call $\mathcal{L}$ a *sublattice* of $\mathcal{M}$.

Since sublattices are subgroups of abelian groups, the cosets also form a group. Theorem 4.3.4 gives a connection between determinants and the cardinality of the quotient group. This can be more convenient when dealing with integer lattices since it is often easier to determine the cardinality of $\mathbb{Z}^d/\mathcal{L}$ than to directly calculate the determinant.

THEOREM 4.3.4. [Dad18, Lecture 2, Lemma 10.2]. Let $\mathcal{L} \subset \mathcal{M}$ be d-dimensional lattices. The quotient group $\mathcal{M}/\mathcal{L}$ is a finite group of order $\det \mathcal{L}/\det \mathcal{M}$.

We give the special case where $\mathcal{M}$ is the integer lattice $\mathbb{Z}^d$ as a corollary, which uses the fact that $\det \mathbb{Z}^d = 1$.

COROLLARY 4.3.5. Let $\mathcal{L}$ be a sublattice of $\mathbb{Z}^d$. The quotient group $\mathbb{Z}^d/\mathcal{L}$ has order $\det \mathcal{L}$.

We can also say something about the dual of a sublattice of an integer lattice. If we take a d-dimensional sublattice $\mathcal{L} \subset \mathbb{Z}^d$, then the dual lattice $\mathcal{L}^*$ contains every point in $\mathbb{R}^d$ such that the inner product with any vector in $\mathcal{L}$ is integral. Note that since $\mathcal{L} \subset \mathbb{Z}^d$, the inner product of a vector in $\mathcal{L}$ and a vector in $\mathbb{Z}^d$ is integral. This implies that $\mathbb{Z}^d$ is a sublattice of $\mathcal{L}^*$. Now using theorem 4.3.4 the quotient $\mathcal{L}^*/\mathbb{Z}^d$ is a group of order $\det \mathcal{L}$.

4.4. Fourier analysis

We can embed a d-dimensional lattice into the Euclidean space $\mathbb{R}^d$. We define the Fourier transform on functions from $\mathbb{R}^d$.

Definition 4.4.1 (Fourier transform). Let $\mathcal{L} \subset \mathbb{R}^d$ be a d-dimensional lattice and let $f\colon \mathbb{R}^d \to \mathbb{C}$ be a function such that $\int_{\mathbb{R}^d} |f(x)|dx < \infty$, then the *Fourier transform* of f is a function $\widehat{f}\colon \mathbb{R}^d \to \mathbb{C}$ defined by

$$\widehat{f}(y) \coloneqq \int_{\mathbb{R}^d} f(x)e^{-2\pi i\langle x,y\rangle}dx.$$

We also define a similar notion on functions that are $\mathcal{L}$-periodic.

Definition 4.4.2 (Fourier coefficient). Let B be a basis of some d-dimensional lattice $\mathcal{L}$ and let $f\colon \mathbb{R}^d \to \mathbb{C}$ be a $\mathcal{L}$-periodic function. The Fourier coefficient with respect to $y \in \mathcal{L}^*$ is

$$\widehat{f}(y) \coloneqq \frac{1}{\det(\mathcal{L})} \int_{\mathcal{P}(B)} f(x)e^{-2\pi i\langle x,y\rangle}dx.$$

Note that when we have a function f, the notation $\widehat{f}$ refers to the Fourier coefficient when f is $\mathcal{L}$-periodic, and to the Fourier transform otherwise. We can also write a nice enough $\mathcal{L}$-periodic function in terms of its Fourier coefficients, using the following lemma.

Lemma 4.4.3. [Dad18, Lecture 6, Thm. 14]. Let $\mathcal{L} \subset \mathbb{R}^d$ be a d-dimensional lattice. Let $f\colon \mathbb{R}^d \to \mathbb{C}$ be a continuous $\mathcal{L}$-periodic function such that $\sum_{y\in\mathcal{L}^*} |\widehat{f}(y)| < \infty$. The following identity holds for all $x \in \mathbb{R}^d$:

$$f(x) = \sum_{y\in\mathcal{L}^*} \widehat{f}(y)e^{2\pi i\langle y,x\rangle}.$$

The *Poisson Summation Theorem* (Theorem 4.4.4) gives a useful identity between the mass of a (nice enough) function f over a lattice $\mathcal{L}$ and the mass of its Fourier transform $\widehat{f}$ over the dual lattice $\mathcal{L}^*$.

Theorem 4.4.4 (Poisson Summation Formula). Let $\mathcal{L} \subset \mathbb{R}^d$ be a d-dimensional lattice with basis B and $f\colon \mathbb{R}^d \to \mathbb{C}$ be a function satisfying the following conditions:

(1) $\int_{\mathbb{R}^d} |f(x)|dx < \infty$;

(2) $\sum_{z\in\mathcal{L}} |f(x+z)|$ converges uniformly for all $x \in \mathbb{R}^d$;

(3) $\widehat{f}(\mathcal{L}^*)$ is an absolutely convergent sum.

Then the following identity holds:

$$f(\mathcal{L}) = \det(\mathcal{L}^*)\widehat{f}(\mathcal{L}^*).$$

Proof. Define $\varphi(x) \coloneqq f(x + \mathcal{L})$, by condition (2) this converges uniformly. This function is by construction $\mathcal{L}$-periodic, so we may consider its Fourier coefficients. Let $y \in \mathcal{L}^*$, we have

$$\widehat{\varphi}(y) = \det(\mathcal{L}^*) \int_{\mathcal{P}(B)} \sum_{z\in\mathcal{L}} f(x+z)e^{-2\pi i\langle x,y\rangle}dx.$$

Since the sum is absolutely convergent by condition (2), we may interchange the sum and integral, giving

$$\widehat{\varphi}(y) = \det(\mathcal{L}^*) \sum_{z\in\mathcal{L}} \int_{\mathcal{P}(B)} f(x+z)e^{-2\pi i\langle x,y\rangle}dx.$$

By definition of the dual lattice we have $\langle z, y\rangle \in \mathbb{Z}$ for any $z \in \mathcal{L}$, thus $e^{-2\pi i\langle z,y\rangle} = 1$, so we may write

$$\begin{aligned}
\widehat{\varphi}(y) &= \det(\mathcal{L}^*) \sum_{z\in\mathcal{L}} \int_{\mathcal{P}(B)} f(x+z)e^{-2\pi i\langle x,y\rangle}e^{-2\pi i\langle z,y\rangle}dx \\
&= \det(\mathcal{L}^*) \sum_{z\in\mathcal{L}} \int_{\mathcal{P}(B)} f(x+z)e^{-2\pi i\langle x+z,y\rangle}dx \\
&= \det(\mathcal{L}^*) \sum_{z\in\mathcal{L}} \int_{z+\mathcal{P}(B)} f(x)e^{-2\pi i\langle x,y\rangle}dx.
\end{aligned}$$

Since $\coprod_{z\in\mathcal{L}}(z + \mathcal{P}(B)) = \mathbb{R}^d$, we may write this sum of integrals as a single integral over all of $\mathbb{R}^d$,

$$\begin{aligned}
\widehat{\varphi}(y) &= \det(\mathcal{L}^*) \int_{\mathbb{R}^d} f(x)e^{-2\pi i\langle x,y\rangle}dx \\
&= \det(\mathcal{L}^*)\widehat{f}(y).
\end{aligned}$$

Now applying the Fourier inversion formula to $\varphi(0)$ gives the desired identity:

$$f(\mathcal{L}) = \varphi(0) = \sum_{y\in\mathcal{L}^*} \widehat{\varphi}(y)e^{2\pi i\langle 0,y\rangle} = \widehat{\varphi}(\mathcal{L}^*) = \det(\mathcal{L}^*)\widehat{f}(\mathcal{L}^*). \qquad \square$$

We finish this section by stating two well-known properties of Fourier transforms for future reference, which can easily be verified.

PROPOSITION 4.4.5. If $h(x) \coloneqq f(x + z)$, then $\widehat{h}(y) = e^{2\pi i\langle y,z\rangle}\widehat{f}(y)$.

PROPOSITION 4.4.6. If $h(x) = e^{2\pi i\langle x,z\rangle} f(x)$, then $\widehat{g}(y) = \widehat{f}(y-z)$.

4.5. Gaussians on lattices

We will be particularly interested in using the Poisson Summation Formula (Theorem 4.4.4) in the special case where f is a Gaussian. First we define the d-dimensional Gaussian:

DEFINITION 4.5.1 (GAUSSIAN). We define the d-dimensional Gaussian with parameter $s > 0$ as

$$\rho_s : \mathbb{R}^d \to \mathbb{R}_{\geq 0}, \quad x \mapsto e^{-\pi \|x/s\|^2}$$

REMARK 4.5.2. The function ρ_s also depends on the dimension d, since the norm function depends on d. It should be clear from the context what d is.

We can calculate the Fourier transform of the d-dimensional Gaussian of parameter s to be

$$\widehat{\rho_s} = s^d \rho_{1/s}.$$

It can be easily verified that ρ_s satisfies the conditions of Theorem 4.4.4, giving the following corollary:

COROLLARY 4.5.3. Let $\mathcal{L} \subset \mathbb{R}^d$ be a d-dimensional lattice with basis B and let ρ_s be the d-dimensional Gaussian with parameter s. The following identity holds

$$\rho_s(\mathcal{L}) = \frac{s^n}{\det \mathcal{L}} \rho_{1/s}(\mathcal{L}^*).$$

A very useful fact due to Banaszczyzk is that all but a negligible amount of the Gaussian mass of a lattice is concentrated near the origin.

LEMMA 4.5.4. [Ban93]. For any d-dimensional lattice $\mathcal{L}$, $x \in \mathbb{R}^d$, and $s > 0$, the following inequality holds:

$$\rho_s((\mathcal{L} + x) \setminus \overline{\mathcal{B}_{\sqrt{d}s}}) < 2^{-d} \rho_s(\mathcal{L}),$$

where $\overline{\mathcal{B}_{\sqrt{d}s}}$ is the closed ball centered around the origin of radius $\sqrt{d}s$.

Often it is useful to construct lattices which only contain large vectors, then we can use the above lemma to show that the total Gaussian mass of these lattices is very small outside of the point 0. The precise statement is given in the following corollary.

COROLLARY 4.5.5. [Reg24, Corollary A.2]. If $\mathcal{L}$ is a lattice containing no nonzero vectors of norm at most $\sqrt{d}s$ then $\rho_s(\mathcal{L} \setminus \{0\}) \leq 2^{-d+1}$.

Proof. By the condition on $\mathcal{L}$ we have $\mathcal{L} \setminus \{0\} = \{y \in \mathcal{L} \mid \|y\| > \sqrt{d}s\}$. now the previous lemma gives

$$\rho_s(\mathcal{L} \setminus \{0\}) < 2^{-d} \cdot \rho_s(\mathcal{L}).$$

Since $\rho_s(0) = 1$ we have $2^{-d} \cdot \rho_s(\mathcal{L}) = 2^{-d}(1 + \rho_s(\mathcal{L} \setminus \{0\}))$, rearranging now gives

$$(1 - 2^{-d})\rho_s(\mathcal{L} \setminus \{0\}) < 2^{-d}$$

which can be further rearranged to

$$\rho_s(\mathcal{L} \setminus \{0\}) < (1 - 2^{-d})^{-1} \cdot 2^{-d} \leq 2^{-d+1}.$$

□

4.6. LLL-algorithm

This section is based on [Reg04, Lecture 2]. The Lenstra-Lenstra-Lovász algorithm (LLL-algorithm for short) is an algorithm that "reduces" a basis of a lattice $\mathcal{L}$ to a different basis of $\mathcal{L}$ that has some desirable properties.

Before we begin we need to recall the Gram-Schmidt orthogonalization process from Linear Algebra, which can be used to transform any basis into an orthogonal basis.

DEFINITION 4.6.1 (GRAM-SCHMIDT ORTHOGONALIZATION). Given some basis vectors $b_1, \ldots, b_d \in \mathbb{R}^d$, the *Gram-Schmid orthogonalization* of $b_1, \ldots, b_d$ is defined by

$$\widetilde{b_i} = b_i - \sum_{j=1}^{i-1} \mu_{i,j} \widetilde{b_j},$$

where $\mu_{i,j} = \frac{\langle b_i, \widetilde{b_j} \rangle}{\|\widetilde{b_j}\|^2}$.

This new basis is orthogonal, but not necessarily orthonormal. Using this we can define what it means for a basis to be LLL-reduced.

DEFINITION 4.6.2 (LLL-REDUCED BASIS). A basis $b_1, \ldots, b_d \in \mathbb{R}^d$ is said to be *LLL-reduced* if the following two conditions hold:

(1) For all $1 \leq j < i \leq d$ we have $|\mu_{i,j}| \leq \frac{1}{2}$.

(2) For all $1 \leq i \leq d$ we have $\frac{3}{4}\|\widetilde{b_i}\|^2 \leq \|\mu_{i+1,i}\widetilde{b_i} + \widetilde{b}_{i+1}\|^2$.

Here $\mu_{i,j}$ is defined as in Definition 4.6.1.

REMARK 4.6.3. This is actually the definition of a 3/4-LLL-reduced basis, you can substitute the factor 3/4 for any $1/4 < \delta < 1$ to get the definition of a δ-LLL-reduced basis. However, for the purposes of this thesis we will only need the special case where $\delta = 3/4$.

In this thesis we will use the LLL-algorithm performing the following transformation:

THEOREM 4.6.4 (LLL-ALGORITHM). [LLL82]. There exists a polynomial-time (classical) algorithm which, given a basis of a d-dimensional lattice $\mathcal{L}$, produces an LLL-reduced basis for $\mathcal{L}$.

We end this section by proving some properties of LLL reduced bases that will be useful later.

PROPOSITION 4.6.5. Let $b_1, \ldots, b_d$ be an LLL-reduced basis. For each $1 \leq i < d$ we have that

$$\|\widetilde{b}_{i+1}\| \geq \|\widetilde{b}_i\|/\sqrt{2}.$$

Proof. Since $\widetilde{b}_{i+1}$ and $\widetilde{b}_i$ are orthogonal vectors, the Pythagorean Theorem gives

$$\|\mu_{i+1,i}\widetilde{b}_i + \widetilde{b}_{i+1}\|^2 = \mu_{i+1,i}^2\|\widetilde{b}_i\|^2 + \|\widetilde{b}_{i+1}\|^2.$$

By the first condition of LLL-reduced bases we have $\mu_{i+1,i}^2 \leq \frac{1}{4}$. Now the second condition of LLL-reduced bases gives

$$\frac{3}{4}\|\widetilde{b}_i\|^2 \leq \frac{1}{4}\|\widetilde{b}_i\| + \|\widetilde{b}_{i+1}\|^2.$$

This can be rewritten to

$$\|\widetilde{b}_{i+1}\|^2 \geq \frac{1}{2}\|\widetilde{b}_i\|^2,$$

taking the square root gives the desired inequality. □

PROPOSITION 4.6.6. For an LLL reduced basis $b_1, \ldots, b_d$, we have for each $1 \leq j \leq d$ the following inequality

$$\|b_j\|^2 \leq \sum_{i=1}^{j} \|\widetilde{b}_i\|^2.$$

In particular, $\|b_j\| \leq \|\widetilde{b}_j\|$.

Proof. By Definition 4.6.1 we have

$$b_j = \widetilde{b}_j + \sum_{i=1}^{j-1} \mu_{j,i} \widetilde{b}_i,$$

since the set $\{\widetilde{b}_i \mid 1 \leq i \leq j\}$ is orthogonal the Pythagorean Theorem gives

$$\|b_j\|^2 = \|\widetilde{b}_j\|^2 + \sum_{i=1}^{j-1} |\mu_{j,i}|^2 \|\widetilde{b}_i\|^2 \leq \sum_{i=1}^{j} \|\widetilde{b}_i\|^2. \qquad \square$$

5. Regev's algorithm

This chapter is dedicated to using the results from Chapter 4 to prove the correctness and efficiency of Regev's quantum factoring algorithm. The main result of this thesis is given in Theorem 5.6.2, which simply states that given some assumption (5.1.2), there exists a polynomial-time quantum factoring algorithm that uses $\mathcal{O}(\sqrt{n})$ calls to an $\mathcal{O}(n^{3/2}\log n)$-time quantum subroutine. We give a constructive proof of this theorem.

We start this chapter by giving an overview of the algorithm and introducing the assumption that is necessary for efficiency. We then extend the reduction from Shor's algorithm to the d-dimensional case, by reducing factoring to lattice-vector finding. Then we see how modular exponentation can be performed more efficiently using more dimensions.

After this, we describe the Regev's quantum subroutine. We go into detail how this can be efficiently implemented on a quantum computer.

Finally, we describe how we can use the LLL-algorithm to use the output of the quantum subroutine to find a lattice vector, which we can use to factor an integer.

This chapter is mostly based on Regev's paper [Reg24], but the proofs given in this chapter are far more detailed.

5.1. Overview of the algorithm

We reduce the factoring of an n-bit integer N to finding a period of the d-dimensional analogue of modular exponentation, which we define as follows:

Definition 5.1.1 (Vector exponentiation). For vectors $a = (a_1, \ldots, a_d)$ and $z = (z_1, \ldots, z_d)$ in $\mathbb{R}^d$ we define exponentiation as follows:

$$a^z = \prod_{i=1}^{d} a_i^{z_i}.$$

For some vector $a \in \mathbb{Z}^d$ we define the lattice $\mathcal{L}$ to be the periods of $z \mapsto a^z \bmod N$. Note that this is dependent on N and a, though we do not write this explicitly. As a set this lattice looks like

$$\mathcal{L} = \{(z_1, \ldots, z_d) \in \mathbb{Z}^d \mid \prod_{i=1}^{d} a_i^{z_i} \equiv 1 \bmod N\}. \tag{5.1}$$

We may choose a to be a vector of squares (say $a_i = b_i^2$), which means each period in $\mathcal{L}$ is a vector of even integers. As in Shor's algorithm, finding such a lattice vector will,

under certain conditions, prove to be sufficient to find a factor of N (Section 5.2). The first of these conditions is that the lattice consists only of even integers, which is always the case as we chose a to be a vector of squares. The second condition is that the vector must not lie in the following sublattice:

$$\mathcal{L}_0 = \{(z_1, \ldots, z_d) \in \mathbb{Z}^d \mid \prod_{i=1}^{d} b_i^{z_i} \equiv \pm 1 \bmod N\}. \tag{5.2}$$

We make the following **unproven** assumption, which we will need to ensure that the quantum subroutine has the desired time complexity:

ASSUMPTION 5.1.2. Let N be an odd n-bit integer and not a prime power. Let b be a vector of $d < n$ primes and let a be the vector consisting of the squares of b. Define the d-dimensional lattices $\mathcal{L}$ and $\mathcal{L}_0$ as in (5.1) and (5.2) respectively. The norm of the smallest vector $\mathcal{L} \setminus \mathcal{L}_0$ is asymptotically bounded by $2^{\mathcal{O}(d+n/d)}$.

The quantum subroutine will give an approximation to a uniform sample of representatives in $[0,1)^d$ of the quotient group $\mathcal{L}^*/\mathbb{Z}^d$. If we take $d = 1$ we see that this is exactly what happens in Shor's algorithm, there the quantum subroutine returns an approximation to a rational number such that if you multiply it by any period of modular exponentiation you get an integer, which is exactly the definition of an element of the dual lattice. If we assume that Assumption 5.1.2 is true and take $d \approx \sqrt{n}$ the quantum subroutine will cost $\mathcal{O}(n^{3/2} \log n)$ quantum gates.

In Section 5.4 we explore the quantum subroutine in Regev's algorithm in great detail. The quantum subroutine itself is much more involved than in Shor's algorithm, as it's no longer just an application of quantum phase estimation. However, the structure is similar to quantum phase estimation, in the sense that it consists of the following three steps:

(1) Prepare an initial superposition over $\mathbb{Z}^d$, with large enough vectors to find periodic behaviour of modular exponentiation (Subsection 5.4.1).

(2) Apply modular exponentiation in superposition (Subsection 5.4.2).

(3) Apply a quantum Fourier transform and measure (Subsection 5.4.3).

We show in Section 5.5 that with at least $d + 4$ independent samples of the quantum subroutine we can, with good probability, obtain a list of vectors $z_1, \ldots, z_\ell \in \mathcal{L}$ that generate every vector in $\mathcal{L}$ up to a norm bound that is large enough that $z_1, \ldots, z_\ell$ must generate a vector in $\mathcal{L} \setminus \mathcal{L}_0$. Since $\mathcal{L}_0$ is a sublattice of $\mathcal{L}$, at least one of $z_1, \ldots, z_\ell$ must itself lie in $\mathcal{L} \setminus \mathcal{L}_0$.

5.2. Reduction of factoring to finding a lattice vector

In this section we show that we can generalize Shor's reduction of factoring to order-finding to the d-dimensional case. This means we reduce factoring an integer N to

finding a lattice vector in $\mathcal{L} \setminus \mathcal{L}_0$, where $\mathcal{L}$ and $\mathcal{L}_0$ are defined as in (5.1) and (5.2) respectively. In the next section we show why the modular exponentiation step is (asymptotically) faster in higher dimensions. We show that given a vector in $\mathcal{L} \setminus \mathcal{L}_0$ we can efficiently find a non-trivial factor of N. This is very similar to Theorem 3.1.2.

THEOREM 5.2.1. Let N be a positive integer. Let $b \in \mathbb{Z}^d$ be a non-zero vector and define a by $a_i = b_i^2$ for $1 \leq i \leq d$. Now define the lattice

$$\mathcal{L} = \{z \in \mathbb{Z}^d \mid a^z \equiv 1 \bmod N\}$$

and the sublattice

$$\mathcal{L}_0 = \{z \in \mathbb{Z}^d \mid b^z \equiv \pm 1 \bmod N\}.$$

Let $z \in \mathcal{L} \setminus \mathcal{L}_0$, then at least one of $\gcd(b^z - 1, N)$ and $\gcd(b^z + 1, N)$ is a non-trivial factor of N.

Proof. We have

$$a^z - 1 \equiv (b^z)^2 - 1^2 \equiv 0 \bmod N,$$

which can be rewritten as $(b^z - 1)(b^z + 1) = kN$ for $k \in \mathbb{Z}_{>0}$. Since $z \notin \mathcal{L}_0$, we know that N cannot divide either of the factors $b^z - 1$ or $b^z + 1$. Now the proof finishes in the exact same way as Theorem 3.1.2, we cannot have that both $b^z - 1$ and $b^z + 1$ are coprime to N, so one of $\gcd(N, b^z - 1)$ and $\gcd(N, b^z + 1)$ must be a non-trivial factor of N. □

Thus once we have found a vector in $\mathcal{L} \setminus \mathcal{L}_0$, we simply compute both $\gcd(N, b^z - 1)$ and $\gcd(N, b^z + 1)$, check which one is a non-trivial factor and return it, just as in Shor's algorithm.

5.3. Fast modular exponentiation in multiple variables

Like in Shor's algorithm, to find a nonzero lattice vector in $\mathcal{L}$ (the lattice of periods of $z \mapsto a^z \bmod N$, for fixed $a \in \mathbb{Z}^d$) we must be able to compute $z \mapsto a^z \bmod N$ efficiently. In section 3.2 we showed that in the one-dimensional case we can perform the modular exponentation in $\mathcal{O}(n)$ multiplications, which is the slowest part of Shor's algorithm. In the d-dimensional case we will still need about as many (even slightly more) total multiplications, but only a small portion of these multiplications will be between (large) n-bit numbers, while the rest will be between (small) $\mathcal{O}(\log d)$-bit numbers which are fast to calculate.

The first observation is that we actually don't have to calculate all the exponents up to 2^n as in the one-dimensional case. The reason we needed to go all the way to roughly 2^n in Shor's algorithm is that the period could be that large. In the d-dimensional case

we have a tighter upper bound on the smallest non-zero vector in a lattice, so we only need large enough exponents to find this lattice vector. The d-dimensional lattice $\mathcal{L}$ must contain a non-zero vector of length at most $\sqrt{d}(\det \mathcal{L})^{1/d}$ due to Minkowski's first theorem (4.2.3). We want to find an upper bound for the determinant of $\mathcal{L}$, by Theorem 4.3.4 we know that $|\mathbb{Z}^d/\mathcal{L}| = \det \mathcal{L}$, so it suffices to find an upper bound on the number of cosets in $\mathbb{Z}^d/\mathcal{L}$. By definition, two lattice vectors $x, y \in \mathbb{Z}^d$ are in the same coset of $\mathcal{L}$ if and only if their difference $x - y$ is in $\mathcal{L}$. It is easy to see that this is equivalent to to $a^z \equiv b^z \bmod N$, So there cannot be more cosets in $\mathbb{Z}^d/\mathcal{L}$ then there are residue classes modulo N. Since N is an n-bit number, there are at most 2^n cosets in $\mathbb{Z}^d/\mathcal{L}$, so $\det \mathcal{L} \leq 2^n$.

We are guaranteed to encounter periodic behaviour if we take each coordinate of a to exponents up to $\sqrt{d}2^{n/d}$. However, due to the way we will obtain a lattice vector (described in Section 5.5), we need to overshoot this bound by a factor of $2^{\mathcal{O}(d)}$. This means we need to be able to perform modular exponentiation with exponents up to $\sqrt{d}2^{\mathcal{O}(d)+n/d}$, so with $\mathcal{O}(d + n/d)$-bit exponents. Moreover, even though we are guaranteed to get far enough to find a vector in $\mathcal{L}$, we don't know if this is far enough to find a vector in $\mathcal{L} \setminus \mathcal{L}_0$. For the rest of this section say we need to go up to at least t-bit exponents to be able to recover a lattice vector in $\mathcal{L} \setminus \mathcal{L}_0$. If Assumption 5.1.2 is true we have $t = \mathcal{O}(d + n/d)$.

Using the method described in Section 3.2 we can calculate such an exponentiation in $\mathcal{O}(t)$ multiplications between n-bit numbers. However, we need to do this for every coordinate of a and then multiply the results, meaning in total we need to do $\mathcal{O}(dt)$ multiplications, which even in the case where $t = \mathcal{O}(d + n/d)$ seems to be even worse than the one-dimensional case.

However, we can adapt the repeated squaring technique from section 3.2 to force almost all of these multiplications to be between small numbers. We will calculate $\prod_{i=1}^{d} a_i^{z_i} \bmod N$, where each a_i and z_i is non-negative by starting at 1 and using two kinds of operations: squaring and multiplying by the product of a subset of $\{a_1, \ldots, a_d\}$. The effect of squaring is that it doubles each exponent, so appending a 0 to the binary expansion of each exponent. Then if each exponent is even, multiplying by the product of a subset of $\{a_1, \ldots, a_d\}$ will change the last bit of each exponent of the bases in the subset from a 0 to a 1. Since every exponent is a $\mathcal{O}(t)$-bit integer, we need $\mathcal{O}(t)$ squaring operations and we need just as many to multiply with our running total, which are all multiplications between n-bit numbers. Using Fast Integer Multiplication all of these multiplications can be done in $\mathcal{O}(tn \log n)$ time. We give an example to illustrate how this technique works.

EXAMPLE 5.3.1. Say we want to calculate $3^{12} \cdot 5^{14} \cdot 6^6 \cdot 7^{19} \bmod 43$. The binary expansions of the exponents are 01100, 01110, 00110, and 10011 respectively. In between each squaring operation we want to multiply by the product of a subset of $\{3, 5, 6, 7\}$, which will increment a subset of the exponents. We can determine which numbers are in the subset of the k-th product by seeing which of the exponents have a 1 as the k-th most significant bit.

Now we can build the final answer, the exponents are written in binary to better show what is happening in each step. We are building each exponent bit for bit, starting at the most significant bit. On the left we write the numbers we are multiplying with in a way such that it is easy to see which exponents get incremented by the multiplication, while on the right we show what this product actually is (modulo 43).

$$
\begin{aligned}
(1 \cdot 1 \cdot 1 \cdot 7) \times \Big(& \quad 3^{0} \cdot 5^{0} \cdot 6^{0} \cdot 7^{0} \equiv 1 \bmod 43 \Big) \times 7 \\
& \quad 3^{0} \cdot 5^{0} \cdot 6^{0} \cdot 7^{1} \equiv 7 \bmod 43 \Big) \textit{ square} \\
(3 \cdot 5 \cdot 1 \cdot 1) \times \Big(& \quad 3^{00} \cdot 5^{00} \cdot 6^{00} \cdot 7^{10} \equiv 6 \bmod 43 \Big) \times 15 \\
& \quad 3^{01} \cdot 5^{01} \cdot 6^{00} \cdot 7^{10} \equiv 4 \bmod 43 \Big) \textit{ square} \\
(3 \cdot 5 \cdot 6 \cdot 1) \times \Big(& \quad 3^{010} \cdot 5^{010} \cdot 6^{000} \cdot 7^{100} \equiv 16 \bmod 43 \Big) \times 4 \\
& \quad 3^{011} \cdot 5^{011} \cdot 6^{001} \cdot 7^{100} \equiv 21 \bmod 43 \Big) \textit{ square} \\
(1 \cdot 5 \cdot 6 \cdot 7) \times \Big(& \quad 3^{0110} \cdot 5^{0110} \cdot 6^{0010} \cdot 7^{1000} \equiv 11 \bmod 43 \Big) \times 38 \\
& \quad 3^{0110} \cdot 5^{0111} \cdot 6^{0011} \cdot 7^{1001} \equiv 31 \bmod 43 \Big) \textit{ square} \\
(1 \cdot 1 \cdot 1 \cdot 7) \times \Big(& \quad 3^{01100} \cdot 5^{01110} \cdot 6^{00110} \cdot 7^{10010} \equiv 15 \bmod 43 \Big) \times 7 \\
& \quad 3^{01100} \cdot 5^{01110} \cdot 6^{00110} \cdot 7^{10011} \equiv 19 \bmod 43 \Big)
\end{aligned}
$$

So we have calculated

$$3^{12} \cdot 5^{14} \cdot 6^6 \cdot 7^{19} \equiv 19 \bmod 43.$$

We still need to calculate the products of the subsets of $\{a_1, \ldots, a_d\}$, and here comes the trick: this can be done very fast if we force all of $\{a_1, \ldots, a_d\}$ to be very small numbers. The only restriction we placed on them is that we wanted them to be squares, additionally it would be nice if they are all pairwise coprime, to make the most out of every dimension. Perfect candidates would be to choose d distinct squares of $\mathcal{O}(\log d)$-bit primes. The following lemma shows that the product can now be efficiently computed.

LEMMA 5.3.2. The product of d integers with $\mathcal{O}(\log d)$ bits can be calculated in $\mathcal{O}(d \log^3 d)$ time (using Fast Integer Multiplication).

Proof. Let $M(k)$ be the cost of multiplying two k-bit numbers, and $T_d(k)$ be the cost of computing the product of k numbers with $\mathcal{O}(\log d)$ bits. If we split our k numbers into two groups of (rougly) equal size and multiply them together we get the recurrence

$T_d(k) = 2T_d(k/2) + M(k \log d)$. The $k \log d$ comes from the fact that the two smaller products we are multiplying are the product of at most $k/2$ numbers with $\mathcal{O}(\log d)$ bits, so these are $\mathcal{O}(k \log d)$-bit integers. If we use Fast Integer Multiplication we get $M(k \log d) = \mathcal{O}(k \log d(\log k + \log \log d))$. Since we need to multiply d numbers at most, we can use $k \leq d$ to get a complexity of $\mathcal{O}(d \log^2 d)$. Then we can use the Master Theorem to determine that $T_d(d) = \mathcal{O}(d \log^3 d)$. □

We need to do this just as often as the squaring operation, so $\mathcal{O}(t)$ times. In total calculating all the products of the subsets costs $\mathcal{O}(td \log^3 d)$ time. Adding this to the cost of the large multiplications, which is $\mathcal{O}(tn \log n)$, we get a total complexity of $\mathcal{O}(t(n \log n + d \log^3 d))$. We can summarize the proven results of this section in the following theorem:

THEOREM 5.3.3. Let $a \in \mathbb{Z}^d$ be a vector of $(\log d)$-bit integers and $z \in \mathbb{Z}^d_{\geq 0}$ be a vector of $\mathcal{O}(t)$-bit integers. We can compute $a^z \bmod N$ in

$$\mathcal{O}(t(n \log n + d \log^3 d))$$

time.

5.4. The quantum subroutine

We first give an overview of the quantum subroutine, we also state the output of the subroutine and give a useful property of this output, which we will prove later in this section.

The subroutine starts by approximating a discrete Gaussian superposition restricted to the integer points in a hypercube around the origin in $\mathbb{Z}^d$. This Gaussian has a parameter R which determines how much of its mass is concentrated around the origin, a higher R value means points further from the origin contribute more. We choose the side length of the hypercube as some power of 2, say 2^t, which should be somewhat larger than the parameter R to ensure that our state contains a large enough share of the total Gaussian mass over $\mathbb{Z}^d$. This means we only consider vector whose coordinates can be stored as t-bit signed integers. See Figure 5.1 for what this distribution looks like in the case that $d = 2$.

REMARK 5.4.1. Note that we do not take a uniform superposition, as we did in Shor's algorithm. This is because in Shor's algorithm we take a superposition over the entire multiplicative group $(\mathbb{Z}/N\mathbb{Z})^\times$, while in Regev's algorithm we only take a superposition over a small subset close to 0. Since the mass of a Gaussian is negligible far from the origin, restricting the Gaussian to a small grid will be a good approximation for taking the Gaussian over all of $\mathbb{Z}^d$. It is possible that the algorithm still works if you take a uniform superposition restricted to a grid, but this would give less convenient quantum Fourier transform calculations.

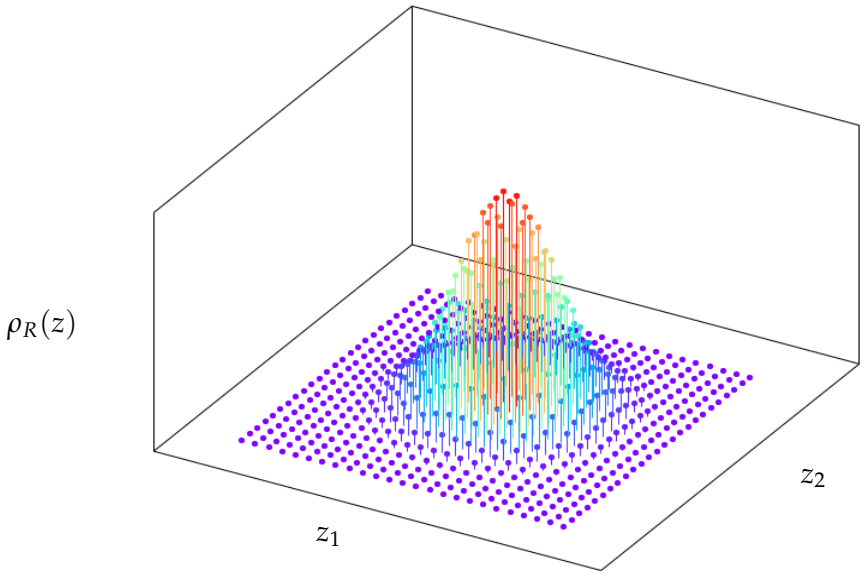


Figure 5.1.: Discrete Gaussian in 2 dimensions, restricted to a square around the origin.

Now we apply modular exponentiation to this state superposition, so we apply the reversible computation $|z\rangle|1\rangle \mapsto |z\rangle|a^z\rangle$, where $a \in \mathbb{Z}^d$ is a vector of small, pairwise coprime, square integers.

After this we have two registers, an input register containing our initial Gaussian superposition, and an output register containing the result of modular exponentiation performed on the input register. Now we apply a quantum Fourier transform over $\mathbb{Z}/2^t\mathbb{Z}$ to the input register. Then we measure both registers. We discard the measurement of the output register, since we're not interested in the values produced by the modular exponentation function, only its periodicity. The measurement of the input register should produce d t-bit binary numbers. We can interpret each of those as a binary fraction in the interval $[0, 1)$ (or, equivalently, we interpret it as an unsigned integer and divide it by 2^t). Using this interpretation, the output of the quantum subroutine is a vector in $[0, 1)^d$. More precisely if we define for $k \in \mathbb{Z}_{>1}$ the cyclic subgroup of $\mathbb{R}/\mathbb{Z}$

$$\mathcal{C}_k = \{0, 1/k, \ldots, (k-1)/k\}, \tag{5.3}$$

the output will be an element of $\mathcal{C}_{2^t}^d$. We will show that the output of the quantum subroutine will be very close to a vector in the dual lattice $\mathcal{L}^*$ of the lattice $\mathcal{L}$ defined in (5.1). The output of the quantum subroutine will be within statistical distance $1/\operatorname{poly}(d)$ of the distribution of choosing a uniformly random $v \in \mathcal{L}^*/\mathbb{Z}^d$ and then taking a sample

from the distribution

$$Q_v(w) = \frac{\rho_{1/(\sqrt{2}R)}(v - w + \mathbb{Z}^d)}{\rho_{1/(\sqrt{2}R)}(v - 2^{-t}\mathbb{Z}^d)} \tag{5.4}$$

supported on $\mathcal{C}^d_{2^t}$. The probability distribution $\rho_{1/(\sqrt{2}R)}$ is a Gaussian centered on v. Since we will choose R to be quite large, the Gaussian will be very steep, so there is a high probability that the sample is close to v. The denominator in (5.4) is simply a normalization factor, making Q_v a probability distribution.

The main goal of this section is proving the following theorem about the output of the quantum subroutine:

THEOREM 5.4.2. The distribution of the output of the quantum subroutine described above will be within ℓ_1 statistical distance $1/\operatorname{poly}(d)$ of the probability distribution

$$Q(w) = (\det \mathcal{L})^{-1} \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} Q_v(w), \tag{5.5}$$

where Q_v is as defined in (5.4), supported on $\mathcal{C}^d_{2^t}$ as defined in (5.3).

Moreover, if we choose a parameter $R > \sqrt{2d}$ and set $t = 1 + \lceil \log_2(\sqrt{d}R) \rceil$, then we can create a quantum circuit which produces the results of the quantum subroutine described above using

$$\mathcal{O}(t(n \log n + d \log^3 d + d \log t) + d \operatorname{poly}(\log d))$$

quantum gates.

As discussed, a sample from the quantum subroutine is a good approximation of a uniform sample from $\mathcal{L}^*/\mathbb{Z}^d$. We make this more precise with the following theorem, that we will prove later in this section.

THEOREM 5.4.3. [Reg24, Claim A.7] For any $v \in \mathbb{R}^d/\mathbb{Z}^d$, if w is chosen from the distribution Q_v as defined in (5.4), then the probability that $\|w - v\|_{\mathbb{R}^d/\mathbb{Z}^d} > \sqrt{d}/(\sqrt{2}R)$ is at most $\mathcal{O}(2^{-d})$.

The remainder of this section is dedicated to showing how we can efficiently implement this quantum subroutine and proving these two theorems.

5.4.1. Preparing the initial state

We choose parameters $R > \sqrt{2d}$ and $t = 1 + \lceil \log_2(\sqrt{d}R) \rceil$. This ensures that $2^t \in [2\sqrt{d}R, 4\sqrt{d}R)$. Define the d-dimensional grid

$$\mathcal{G} = \{-2^t, -2^t + 1, \ldots, 2^t - 2, 2^t - 1\}^d.$$

The algorithm starts by approximating a state that is proportional to a Gaussian distribution of parameter R supported on the grid $\mathcal{G}$. So we want to approximate the state proportional to

$$\sum_{z \in \mathcal{G}} \rho_R(z)|z\rangle. \tag{5.6}$$

In Figure 5.1 we show what this distribution looks like in the case that $d = 2$.

Note that this is simply the tensor product of d one-dimensional Gaussians, so we only need to consider the one-dimensional case. We use the technique described in [GR02] to prepare the initial state. We give a very brief overview of how this technique works. We prepare each qubit in order of most significant to least significant. First we split the domain into a left and right half, and let the first qubit be the superposition of $|0\rangle$ and $|1\rangle$ that encodes the probability of landing in the left and right regions. Then each iteration we split each region in half again, and condition the next qubit on the qubit we just prepared. Doing this for t iterations, which is as many qubits as we need, will cost $\text{poly}(t)$ quantum gates, which is too expensive. Fortunately, if we apply this technique for just $\mathcal{O}(\log d)$ operations, and then simply set the remaining qubits to the $|+\rangle$ state, which can be done with a Hadamard gate per qubit, the resulting state will still be a good enough approximation and can be prepared using only $\mathcal{O}(\text{poly}(\log d) + t)$ quantum gates.

The domain of the Gaussian we start with has length 2^t, and since $t = \mathcal{O}(\log d + \log R)$ we only need $\mathcal{O}(\log d)$ halvings of the regions such that each region has length at most $R/\text{ poly}(d)$. Now we have, for $z \in \mathcal{G}$,

$$\rho_R(z + R/\text{ poly}(d)) = e^{-\pi(z+R/\text{ poly}(d))^2/R^2} = \rho_R(z) \cdot e^{z/(R\,\text{poly}(d))}.$$

The exponent $z/(R\,\text{poly}(d))$ gets further from 0 the larger z gets in absolute value, which is at most $2^{t-1} \leq 2\sqrt{d}R$, so the exponent is bounded in absolute value by $2\sqrt{d}/\text{ poly}(d) = 1/\text{ poly}(d)$. With the Taylor Series[1] of e^x, we can see that $\rho_R(z + R/\text{ poly}(d))$ is within $1/\text{ poly}(d)$ of 1, so after having done sufficient halvings (at most $\mathcal{O}(\log d)$), the ratio of the Gaussians of two points within a region is within $1 \pm 1/\text{ poly}(d)$. Since the number of regions is t and the number of dimensions is t, if we assume that t is at most polynomial in d the probability of landing in the left of the region is within $1/(\text{poly}(d))$ of $1/2$, meaning the state $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ approximates the actual probability within $1/\text{ poly}(d)$ accuracy.

We summarize this subsection with the following theorem:

THEOREM 5.4.4. Let $R > \sqrt{2d}$, $t = 1 + \lceil \log_2(\sqrt{d}R) \rceil$ and $\mathcal{G} = \{-2^t, \ldots, 2^t - 1\}^d$. If t is at most polynomial in d, we can prepare to within ℓ_2-distance $1/\text{ poly}(d)$ the state proportional to

$$\sum_{z \in \mathcal{G}} \rho_R(z)|z\rangle$$

[1] $e^{1/\text{ poly}(x)} = 1 + 1/\text{ poly}(x)$.

using $\mathcal{O}(d \operatorname{poly}(\log d) + dt)$ quantum gates.

5.4.2. Modular exponentiation in superposition

Let $h\colon \mathbb{Z}^d \to (\mathbb{Z}/N\mathbb{Z})^\times$ be the modular exponentiation function for some fixed vector $a \in \mathbb{Z}^d$, so $h(z) = a^z$. We want to calculate the state proportional to

$$\sum_{z \in \mathcal{G}} \rho_R(z)|z\rangle|h(z)\rangle.$$

We can rewrite this sum to group all the vectors in $\mathcal{G}$ that get sent to the same value

$$\sum_{u \in \operatorname{im} h} \sum_{z \in h^{-1}(u) \cap \mathcal{G}} \rho_R(z)|z\rangle|u\rangle. \tag{5.7}$$

This state can be rewritten further using the following lemma:

LEMMA 5.4.5. There exists a group isomorphism $\psi\colon \operatorname{im} h \xrightarrow{\sim} \mathbb{Z}^d/\mathcal{L}$, given by $u \mapsto u + \mathcal{L}$. Moreover, for $u \in \mathbb{Z}^d/\mathcal{L}$ the set $h^{-1}(\psi^{-1}(u))$ is the coset $u + \mathcal{L}$.

Proof. First note that h is a group homomorphism, since

$$h(z_1 + z_2) = a^{z_1+z_2} = a^{z_1} \cdot a^{z_2}.$$

The kernel of h is by definition

$$\ker h = \{z \in \mathbb{Z}^d \mid a^z \equiv 1 \bmod N\},$$

and this is just equal to our lattice $\mathcal{L}$. Now the first isomorphism theorem gives an isomorphism $\operatorname{im} h \xrightarrow{\sim} \mathbb{Z}^d/\mathcal{L}$ given by $u \mapsto u + \mathcal{L}$. The inverse image $h^{-1}(u)$ is the set of vectors in $\mathbb{Z}^d$ that get sent to u by h, this corresponds to the set of vectors in $\mathbb{Z}^d$ that get sent to $u + \mathcal{L}$ by the canonical surjective homomorphism from $\mathbb{Z}^d$ to $\mathbb{Z}^d/\mathcal{L}$, which is of course the coset $u + \mathcal{L}$. □

Now we can rewrite the state described in (5.7) to

$$\sum_{u \in \mathbb{Z}^d/\mathcal{L}} \sum_{z \in (u+\mathcal{L}) \cap \mathcal{G}} \rho_R(z)|z\rangle|u\rangle.$$

Here u is a representative of the coset $u + \mathcal{L}$, but it actually doesn't matter which one, since we are not at all interested in what the register $|u\rangle$ contains. We don't need $|u\rangle$ to be the correct value, we could also have used any other $\mathcal{L}$-periodic function. In particular we can multiply with $a^{2^{t-1}}$ (which is a bijection, since this is a group multiplication) after applying h, which gives the new function g given by

$$g(z) = a^{z+2^{t-1}},$$

which is also $\mathcal{L}$-periodic. The reason we do this is that now every exponent is non-negative, so we can use the fast modular exponentiation algorithm as described in Section 5.3. Theorem 5.3.3 states that we can apply g like so:

$$\sum_{z\in\mathcal{G}} \rho_R(z)|z\rangle \xmapsto{g} \sum_{z\in\mathcal{G}} \rho_R(z)|z\rangle|g(z)\rangle, \tag{5.8}$$

using $\mathcal{O}(t(n\log n + d\log^3 d))$ quantum gates. Moreover, since we don't care about the value of the $|u\rangle$ register, we can view this as being equivalent up to unitaries only affecting the second register to the state

$$|\varphi_1\rangle := Z_1^{-1} \sum_{u\in\mathbb{Z}^d/\mathcal{L}} \sum_{z\in(u+\mathcal{L})\cap\mathcal{G}} \rho_R(z)|z\rangle|u\rangle, \tag{5.9}$$

where $Z_1 > 0$ is the normalization factor.

The results of this section can be summarized in the following theorem:

THEOREM 5.4.6. Let $h\colon \mathbb{Z}^d \to (\mathbb{Z}/N\mathbb{Z})^\times$ be the modular exponentation function, where N is an n-bit integer. Let $\mathcal{L}$ be the d-dimensional lattice containing all the periods of h. Let $R, t \in \mathbb{Z}_{>0}$ and $\mathcal{G} = \{-2^{t-1}, \dots, 2^{t-1}-1\}$. Then the state proportional to

$$\sum_{z\in\mathcal{G}} \rho_R(z)|z\rangle|h(z)\rangle$$

is – up to unitary operations that only affect the second register – equivalent to the state $|\varphi_1\rangle$ defined in (5.9). Moreover, if we are given the state proportional to

$$\sum_{z\in\mathcal{G}} \rho_R(z)|z\rangle,$$

we can compute (a state equivalent to) $|\varphi_1\rangle$ using $\mathcal{O}(t(n\log n + d\log^3 d))$ quantum gates.

5.4.3. Applying the quantum Fourier transform

We are interested in the output of the following procedure: given (a state equivalent to) $|\varphi_1\rangle$ as defined in (5.9), apply the quantum Fourier transform to the input register, measure and discard the output register, and then measure the remaining state, read the coordinates as unsigned integers and divide by 2^t. We define $P_1(w)$ as the probability of $w \in \mathcal{C}_{2^t}^d$ being the output of this procedure. The main goal of this subsection is to prove the following theorem:

THEOREM 5.4.7. Let Q be the probability distribution as defined in (5.5). We have that

$$\|P_1 - Q\|_1 = \mathcal{O}(2^{-d}).$$

In the calculations in this subsection we will not directly apply the quantum Fourier transform to the state $|\varphi_1\rangle$. Instead, we will apply it to a state $|\varphi_2\rangle$ that is a very close approximation of $|\varphi_1\rangle$ and for which the quantum Fourier transform calculations are nicer. The state $|\varphi_2\rangle$ is similar to $|\varphi_1\rangle$, but we extend the grid $\mathcal{G}$ out to infinity. More precisely, instead of each basis state $|z\rangle$ only getting the Gaussian mass from z itself, it will get all the Gaussian mass from the coset[2] in $\mathbb{Z}^d/2^t\mathbb{Z}^d$ containing z. So we define the state

$$|\varphi_2\rangle := Z_2^{-1} \sum_{u \in \mathbb{Z}^d/\mathcal{L}} \sum_{z \in \mathcal{G}} \rho_R((z + 2^t\mathbb{Z}^d) \cap (u + \mathcal{L}))|z\rangle|u\rangle, \tag{5.10}$$

where $Z_2 > 0$ is the normalization factor. Define P_2 just as P_1, except using this new state $|\varphi_2\rangle$ instead of $|\varphi_1\rangle$. The reason we do this is that we can now see the coordinates of vectors in $\mathcal{G}$ as cosets of $\mathbb{Z}/2^t\mathbb{Z}$, meaning we can apply the quantum Fourier transform over $\mathbb{Z}/2^t\mathbb{Z}$. Intuitively it can be justified that $|\varphi_2\rangle$ cannot differ too much from $|\varphi_1\rangle$, since we are only adding Gaussian mass very far from the origin, which should be a negligible amount. To actually prove this fact requires some calculation, we follow the proof of [Reg24, Claim A.5].

CLAIM 5.4.8. We have that

$$\| |\varphi_1\rangle - |\varphi_2\rangle \|_2 \leq 2^{-d+1}.$$

Moreover, $Z_1/Z_2 \in [1 \pm 2^{-d}]$.

Instead of directly proving this claim we start with some smaller lemmas.

LEMMA 5.4.9. If we identify a pair of d-dimensional vectors with a $2d$-dimensional vector in the obvious way, the set

$$\mathcal{M} := \{(x, y) \in \mathbb{Z}^{2d} \mid x - y \in \mathcal{L} \cap 2^t\mathbb{Z}^d\}.$$

forms a $2d$-dimensional lattice and we have that

$$Z_2^2 = \rho_R(\mathcal{M}).$$

Proof. The intersection of two subgroups of the same group is again a group, so if we take $(x_1, y_1), (x_2, y_2) \in \mathcal{M}$, we have

$$(x_1 - x_2) - (y_1 - y_2) = (x_1 - y_1) + (x_2 - y_2) \in \mathcal{L} \cap 2^t\mathbb{Z}^d$$

by closure. Thus we have $(x_1, y_1) - (x_2, y_2) \in \mathcal{M}$, so $\mathcal{M}$ is a subgroup of $\mathbb{Z}^{2d}$, meaning it is a $2d$-dimensional lattice. We can calculate what it means to take the Gaussian of a vector in this lattice

$$\rho_R(x, y) = e^{-\pi/R^2\left(\sum_{k=1}^d x_k^2 + \sum_{k=1}^d y_k^2\right)} = e^{-\pi\|x/R\|^2} \cdot e^{-\pi\|y/R\|^2} = \rho_R(x) \cdot \rho_R(y).$$

[2] Each $z \in \mathcal{G}$ is a unique representative of a coset of $\mathbb{Z}^d/2^t\mathbb{Z}^d$, since $\mathcal{G}$ is a hypercube of sidelength 2^t.

So we have

$$\rho_R(\mathcal{M}) = \sum_{x \in \mathbb{Z}^d} \sum_{y \in x + (\mathcal{L} \cap 2^t \mathbb{Z}^d)} \rho_R(x) \cdot \rho_R(y). \tag{5.11}$$

We can write out the value of the normalization factor Z_2 :

$$Z_2^2 = \sum_{u \in \mathbb{Z}^d / \mathcal{L}} \sum_{z \in \mathcal{G}} \rho_R((z + 2^t \mathbb{Z}^d) \cap (u + \mathcal{L}))^2. \tag{5.12}$$

We are taking the square of a sum, so we can write out all the cross-terms. The inner sum will range over every vector in $\mathbb{Z}^d$, since every vector $x \in \mathbb{Z}^d$ is an element of a coset of $2^t\mathbb{Z}^d$ and a coset of $\mathcal{L}$. Then the Gaussian of this vector gets multiplied by every $\rho_R(y)$ such that y is both in the same coset of $2^t\mathbb{Z}^d$ and in the same coset of $\mathcal{L}$ as x, this perfectly coincides with what we saw in (5.11), so we have $Z_2^2 = \rho_R(\mathcal{M})$, as desired. □

LEMMA 5.4.10. We have that

$$\| |\varphi_1\rangle - (Z_1/Z_2)|\varphi_2\rangle \|_2 \leq 2^{-d}.$$

Proof. We can calculate $\|Z_2|\varphi_2\rangle - Z_1|\varphi_1\rangle\|_2^2$ like so:

$$\begin{aligned} \|Z_2|\varphi_2\rangle - Z_1|\varphi_1\rangle\|_2^2 &= \sum_{u \in \mathbb{Z}^d / \mathcal{L}} \sum_{z \in \mathcal{G}} \Big(\rho_R((z + 2^t\mathbb{Z}^d) \cap (u + \mathcal{L})) - \rho_R(\{z\} \cap (u + \mathcal{L}))\Big)^2 \\ &= \sum_{u \in \mathbb{Z}^d / \mathcal{L}} \sum_{z \in \mathcal{G}} \Big(\rho_R((z + 2^t\mathbb{Z}^d) \cap (u + \mathcal{L}) \setminus \mathcal{G})\Big)^2 \end{aligned}$$

If we compare with (5.12), this removes every term with a vector (x, y) such that either x or y is in $\mathcal{G}$. So we may define the subset $\mathcal{M}' := \{(x, y) \in \mathcal{M} \mid x, y \notin \mathcal{G}\} \subset \mathcal{M}$ to get the equality

$$\|Z_2|\varphi_2\rangle - Z_1|\varphi_1\rangle\|_2^2 = \rho_R(\mathcal{M}'). \tag{5.13}$$

Since $\mathcal{G}$ contains all the small vectors, we know that every vector in $\mathcal{M}'$ must have norm at least[3] $2^t/\sqrt{2} \geq \sqrt{2d}R$. We can use Lemma 4.5.4 to get the inequality

$$\rho_R(\mathcal{M}') \leq \rho_R(\mathcal{M} \setminus \overline{\mathcal{B}_{\sqrt{2d}R}\}}) \leq 2^{-2d}\rho_R(\mathcal{M}).$$

Using (5.13) and Lemma 5.4.9 we get

$$\|Z_2|\varphi_2\rangle - Z_1|\varphi_1\rangle\|_2^2 \leq (2^{-d}Z_2)^2,$$

Taking the square root and dividing by Z_2 gives the desired upper bound. □

Now we can prove Claim 5.4.8:

[3]Since for every $(x, y) \in \mathcal{M}'$, we have $\|x\|, \|y\| \geq 2^{t-1}$, so $\|(x, y)\| \geq \sqrt{2 \cdot 2^{2t-2}} = \sqrt{2^{2t}/2}$.

Proof of Claim 5.4.8. Since both $|\varphi_1\rangle$ and $|\varphi_2\rangle$ are normalized, we have $\||\varphi_2\rangle\|_2 = 1$ and $\|(Z_1/Z_2)|\varphi_1\rangle\|_2 = Z_1/Z_2$. Now we can use the (reverse) triangle inequality to find

$$|1 - Z_1/Z_2| = \Big| \||\varphi_2\rangle\|_2 - \|(Z_1/Z_2)|\varphi_1\rangle\|_2 \Big| \leq \||\varphi_2\rangle - (Z_1/Z_2)|\varphi_1\rangle\|_2 .$$

By Lemma 5.4.10 this is bounded from above by 2^{-d}. Thus

$$Z_1/Z_2 \in [1 \pm 2^{-d}].$$

Then we use the triangle inequality to get

$$\||\varphi_1\rangle - |\varphi_2\rangle\|_2 \leq \||\varphi_1\rangle - (Z_1/Z_2)|\varphi_1\rangle\|_2 + \|(Z_1/Z_2)|\varphi_1\rangle - |\varphi_2\rangle\|_2 \leq 2^{-d+1},$$

as desired. □

Now we prove that the square of the normalization factor Z_1 is very close to $(R/\sqrt{2})^d$, we follow the proof of [Reg24, Claim A.4].

CLAIM 5.4.11. We have that

$$Z_1^2 \in (R/\sqrt{2})^d[1 \pm 2^{-d+1}].$$

Proof. We have

$$Z_1^2 = \sum_{u \in \mathbb{Z}^d/\mathcal{L}} \rho_R(\{(u + \mathcal{L}) \cap \mathcal{G}\})^2$$

Note u ranges over every coset in $\mathbb{Z}^d/\mathcal{L}$, so we can simply range over all of $\mathbb{Z}^d \cap \mathcal{G} = \mathcal{G}$.

$$Z_1^2 = \sum_{z \in \mathcal{G}} \rho_R(z)^2 = \rho_{R/\sqrt{2}}(\mathcal{G}).$$

Where the last equality holds since squaring a Gaussian is the same as diving its parameter by $\sqrt{2}$. We again use that $\mathcal{G}$ contains every small vector, so the Gaussian mass outside of $\mathcal{G}$ is negligible. We can use Lemma 4.5.4 to get the inequality

$$\rho_{R/\sqrt{2}}(\mathbb{Z}^d \setminus \mathcal{G}) \leq \rho_{R/\sqrt{2}}(\mathbb{Z}^d \setminus \overline{\mathcal{B}_{\sqrt{d}R/\sqrt{2}}}) \leq 2^{-d}\rho_{R/\sqrt{2}}(\mathbb{Z}^d).$$

Clearly the Gaussian mass of all of $\mathbb{Z}^d$ will give an upper bound for the Gaussian mass of $\mathcal{G}$, so we can bound Z_1^2 from both sides:

$$Z_1^2 \in \rho_{R/\sqrt{2}}(\mathbb{Z}^d)[1 - 2^{-d}, 1]. \tag{5.14}$$

Now we can use the Poisson Summation Formula (Corollary 4.5.3) to get

$$\rho_{R/\sqrt{2}}(\mathbb{Z}^d) = (R/\sqrt{2})^d \rho_{\sqrt{2}/R}(\mathbb{Z}^d). \tag{5.15}$$

It remains to show that $\rho_{\sqrt{2}/R}(\mathbb{Z}^d)$ is close to 1. We assumed that $R > \sqrt{2d}$, so we have $\sqrt{d} \cdot \sqrt{2}/R < 1$, meaning we don't have any lattice vectors in $\mathbb{Z}^d$ of length at most $\sqrt{d} \cdot \sqrt{2}/R$, so we can use Corollary 4.5.5 to get

$$\rho_{\sqrt{2}/R}(\mathbb{Z}^d \setminus \{0\}) \in [0, 2^{-d+1}].$$

Simply adding $\rho_{\sqrt{2}/R}(0) = 1$ gives

$$\rho_{\sqrt{2}/R}(\mathbb{Z}^d) \in [1, 1 + 2^{-d+1}].$$

Now we use the equality we found in (5.15) to get

$$\rho_{R/\sqrt{2}} \in (R/\sqrt{2})^d [1, 1 + 2^{-d+1}].$$

Substituting this into (5.14) gives the desired bounds. □

Now we can begin to calculate what happens after applying the quantum Fourier transform and measuring.

LEMMA 5.4.12. The probability distribution P_2 is given by

$$P_2(w) = \frac{R^{2d}}{Z_2^2 \cdot 2^{td} (\det \mathcal{L})^2} \sum_{u \in \mathbb{Z}^d/\mathcal{L}} \Big| \sum_{v \in \mathcal{L}^*} e^{2\pi i \langle v,u \rangle} \rho_{1/R}(v - w) \Big|^2,$$

where $w \in \mathcal{C}_{2^t}^d$. In particular this implies that $P_2(w)$ is higher the closer w is to a dual lattice vector v.

Proof. Recall the state $|\varphi_2\rangle$ from (5.10)

$$|\varphi_2\rangle = Z_2^{-1} \sum_{u \in \mathbb{Z}^d/\mathcal{L}} \sum_{z \in \mathcal{G}} \rho_R((z + 2^t\mathbb{Z}^d) \cap (u + \mathcal{L}))|z\rangle|u\rangle.$$

We apply the quantum Fourier transform over $\mathbb{Z}/2^t\mathbb{Z}$ to the input register of $|\varphi_2\rangle$. Here we see a basis state $|k\rangle$ as corresponding to the coset in $\mathbb{Z}/2^t\mathbb{Z}$ containing k. Since $\mathcal{G}$ forms a set of representatives of $(\mathbb{Z}/2^t\mathbb{Z})^d$, we can see the coordinates of its vectors as cosets of $\mathbb{Z}/2^t\mathbb{Z}$. This gives the state

$$\frac{1}{Z_2\sqrt{2^{td}}} \sum_{u \in \mathbb{Z}^d/\mathcal{L}} \sum_{w,z \in (\mathbb{Z}/2^t\mathbb{Z})^d} e^{2\pi i \langle w,z \rangle / 2^t} \rho_R(z + 2^t\mathbb{Z}^d \cap (u + \mathcal{L}))|w\rangle|u\rangle.$$

Now if we measure a vector $w \in (\mathbb{Z}/2^t\mathbb{Z})^d$, then for each of the d dimensions it should give a t-bit integer representing a coset of $\mathbb{Z}/2^t\mathbb{Z}$, which we can naturally interpret as an integer in $\{0, \ldots, 2^t - 1\}$. We can let w range over $\mathcal{C}_{2^t}^d$ by multiplying each occurence of w by 2^t, this gives

$$\frac{1}{Z_2\sqrt{2^{td}}} \sum_{u \in \mathbb{Z}^d/\mathcal{L}} \sum_{w \in \mathcal{C}_{2^t}^d} \sum_{z \in (\mathbb{Z}/2^t\mathbb{Z})^d} e^{2\pi i \langle w,z \rangle} \rho_R(z + 2^t\mathbb{Z}^d \cap (u + \mathcal{L}))|2^t w\rangle|u\rangle.$$

Note the cancellation of the 2^t factor in the exponent. Now since in the Gaussian we sum over the coset containing z for each coset, we sum over $\mathbb{Z}^d \cap (u + \mathcal{L}) = u + \mathcal{L}$. We still have that for each $z \in u + \mathcal{L}$ the Gaussian $\rho_R(z)$ has a factor $e^{2\pi i\langle w,z\rangle}$, and if $z \notin u + \mathcal{L}$ the Gaussian is an empty sum, so 0. Using this we can simplify the state by letting z range over $u + \mathcal{L}$ directly. This gives

$$\frac{1}{Z_2\sqrt{2^{td}}} \sum_{u\in\mathbb{Z}^d/\mathcal{L}} \sum_{w\in\mathcal{C}^d_{2^t}} \sum_{z\in u+\mathcal{L}} e^{2\pi i\langle w,z\rangle}\rho_R(z)\big|2^t w\big\rangle|u\rangle.$$

Now notice how the innermost sum is $\sum_{z\in\mathcal{L}} f(z)$, where $f(x) = e^{2\pi i\langle w,x+u\rangle}\rho_R(z+u)$. Applying the Poisson Summation Formula (Theorem 4.4.4) we see that this equals $\frac{1}{\det\mathcal{L}}\sum_{v\in\mathcal{L}^*}\widehat{f}(v)$, meaning we just need to find $\widehat{f}$. By Proposition 4.4.5 we know the Fourier transform of $x \mapsto \rho_R(x+u)$ is $e^{2\pi i\langle x,u\rangle}\widehat{\rho_R}(x)$. Now by Proposition 4.4.6 the Fourier transform of $f(x)$ is $e^{2\pi i\langle x,u\rangle}\widehat{\rho_R}(x-w)$. By simply filling in $\widehat{\rho_R}(x) = R^d\rho_{1/R}(x)$ we can rewrite the state as

$$\frac{R^d}{Z_2\sqrt{2^{td}}\det\mathcal{L}} \sum_{u\in\mathbb{Z}^d/\mathcal{L}} \sum_{w\in\mathcal{C}^d_{2^t}} \sum_{v\in\mathcal{L}^*} e^{2\pi i\langle v,u\rangle}\rho_{1/R}(v-w)\big|2^t w\big\rangle|u\rangle.$$

If we now discard the result of measuring the second register and divide the result of measuring the first register by 2^t, the probability of measuring $w \in \mathcal{C}^d_{2^t}$ is

$$\frac{R^{2d}}{Z_2^2\cdot 2^{td}(\det\mathcal{L})^2} \sum_{u\in\mathbb{Z}^d/\mathcal{L}} \Big|\sum_{v\in\mathcal{L}^*} e^{2\pi i\langle v,u\rangle}\rho_{1/R}(v-w)\Big|^2,$$

as desired. □

LEMMA 5.4.13. The probability distribution P_2 can be bounded from below by the probability measure

$$P_2'(w) := \frac{R^{2d}}{Z_2^2\cdot 2^{td}\det\mathcal{L}} \sum_{v\in\mathcal{L}^*/\mathbb{Z}^d} \rho_{1/\sqrt{2}R}(v-w+\mathbb{Z}^d),$$

where $w \in \mathcal{C}^d_{2^t}$.

Proof. For ease of notation let $\gamma := \frac{R^{2d}}{Z_2^2\cdot 2^{td}(\det\mathcal{L})^2}$. We can rewrite $P_2(w)$ (as obtained in Lemma 5.4.12) as follows:

$$\begin{aligned} P_2(w) &= \gamma \sum_{u\in\mathbb{Z}^d/\mathcal{L}} \sum_{v_1,v_2\in\mathcal{L}^*} e^{2\pi i\langle v_1,u\rangle}\rho_{1/R}(v_1-w)\cdot\overline{e^{2\pi i\langle v_2,u\rangle}\rho_{1/R}(v_2-w)} \\ &= \gamma \sum_{v_1,v_2\in\mathcal{L}^*} \left(\sum_{u\in\mathbb{Z}^d/\mathcal{L}} e^{2\pi i\langle v_1-v_2,u\rangle}\right)\rho_{1/R}(v_1-w)\rho_{1/R}(v_2-w). \end{aligned}$$

We can use character theory[4] to show that

$$\sum_{u \in \mathbb{Z}^d/\mathcal{L}} e^{2\pi i \langle v_1 - v_2, u\rangle} = \begin{cases} \det \mathcal{L} & \text{if } v_1 - v_2 \in \mathbb{Z}^d; \\ 0 & \text{else.} \end{cases} \tag{5.16}$$

This is proven in Claim A.2.2. We can use this to simplify $P_2(w)$ to

$$\gamma \det \mathcal{L} \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} \rho_{1/R}(v - w + \mathbb{Z}^d)^2.$$

Here we take the square of a sum of Gaussians, which we can bound from below with a sum of squares of Gaussians, where we use that taking the square of a Gaussian is the same as dividing the parameter by $\sqrt{2}$:

$$P_2(w) \geq \gamma \det \mathcal{L} \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} \rho_{1/\sqrt{2}R}(v - w + \mathbb{Z}^d) = P_2'(w),$$

which is the desired bound. □

If we define

$$\alpha_v = \frac{R^{2d}}{Z_2^2 \cdot 2^{td} \cdot \det \mathcal{L}} \cdot \rho_{1/(\sqrt{2}R)}(v - 2^{-t}\mathbb{Z}^d) \tag{5.17}$$

for $v \in \mathcal{L}^*/\mathbb{Z}^d$, then we have

$$P_2'(w) = \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} \alpha_v Q_v(w),$$

where Q_v is defined in (5.4).

LEMMA 5.4.14. We have that

$$\alpha_v \in (\det \mathcal{L})^{-1}[1 \pm \mathcal{O}(2^{-d})].$$

In particular this is not very much dependent on v.

Proof. Using Proposition 4.4.5 we have that the Fourier transform of $w \mapsto \rho_s(v - w)$ is $y \mapsto e^{2\pi i \langle v, y\rangle} \widehat{\rho}_{1/s}(y)$. We use the Poisson Summation Formula (Theorem 4.4.4), using

[4] In short, notice that for $v \in \mathcal{L}^*/\mathbb{Z}^d$ the function $\chi_v(x) = e^{2\pi i \langle v, x\rangle}$ is a character of $\mathbb{Z}^d/\mathcal{L}$, then the sum equals $\det \mathcal{L} \cdot \langle \chi_{v_1}, \chi_{v_2}\rangle$ and the result follows from the orthogonality relations of characters.

that $2^t\mathbb{Z}^d$ is the dual lattice of $2^{-t}\mathbb{Z}^d$ to get

$$\begin{aligned}
\alpha_v &= \frac{R^{2d}}{Z_2^2 \cdot 2^{td} \cdot \det \mathcal{L}} \cdot \rho_{1/(\sqrt{2}R)}(v - 2^{-t}\mathbb{Z}^d) \\
&= \frac{R^{2d}}{Z_2^2 \cdot 2^{td} \cdot \det \mathcal{L}} \cdot \frac{1/(\sqrt{2}R)^d}{\det(2^{-t}\mathbb{Z}^d)} \sum_{y \in 2^t\mathbb{Z}^d} e^{2\pi i \langle v, y \rangle} \rho_{\sqrt{2}R}(y) \\
&= \frac{(R^2/\sqrt{2}R)^d}{Z_2^2 \cdot 2^{td} \cdot \det \mathcal{L} \cdot 2^{-td}} \cdot \sum_{y \in 2^t\mathbb{Z}^d} e^{2\pi i \langle v, y \rangle} \rho_{\sqrt{2}R}(y) \\
&= \frac{(R/\sqrt{2})^d}{Z_2^2 \cdot \det \mathcal{L}} \cdot \sum_{y \in 2^t\mathbb{Z}^d} e^{2\pi i \langle v, y \rangle} \rho_{\sqrt{2}R}(y).
\end{aligned}$$

We can bound the absolute value of the sum (ignoring the Gaussian mass at 0 for now):

$$\begin{aligned}
\Big| \sum_{y \in 2^t\mathbb{Z}^d \setminus \{0\}} e^{2\pi i \langle v, y \rangle} \rho_{\sqrt{2}R}(y) \Big| &\leq \sum_{y \in 2^t\mathbb{Z}^d \setminus \{0\}} |e^{2\pi i \langle v, y \rangle}| \cdot |\rho_{\sqrt{2}R}(y)| \\
&= \rho_{\sqrt{2}R}(2^t\mathbb{Z}^d \setminus \{0\}).
\end{aligned}$$

Note that $2^t > \sqrt{2d}R$, so we may use Corollary 4.5.5 to get that this is at most 2^{-d+1}. Then adding $e^{2\pi i \langle v, 0 \rangle} \rho_{\sqrt{2}R}(0) = 1$, we get

$$\alpha_v \in \frac{(R/\sqrt{2})^d}{Z_2^2 \cdot \det \mathcal{L}} \cdot [1 \pm 2^{-d+1}].$$

Now we can use the bounds found in Claim 5.4.11 to get

$$\alpha_v \in (\det \mathcal{L})^{-1}[1 \pm \mathcal{O}(2^{-d})],$$

as desired. □

LEMMA 5.4.15. We have that

$$\left\| P_2' - Q \right\|_1 = \mathcal{O}(2^{-d}).$$

Proof. We have

$$\begin{aligned}
\left\|P_2' - Q\right\|_1 &= \sum_{w \in 2^{-t}\mathcal{G}} \Big| \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} (\alpha_v - (\det \mathcal{L})^{-1}) Q_v(w) \Big| \\
&\leq \sum_{w \in 2^{-t}\mathcal{G}} \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} |\alpha_v - (\det \mathcal{L})^{-1}| Q_v(w) \\
&= (\det \mathcal{L})^{-1} \mathcal{O}(2^{-d}) \sum_{w \in 2^{-t}\mathcal{G}} \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} Q_v(w) && \text{(Lemma 5.4.14)} \\
&= (\det \mathcal{L})^{-1} \mathcal{O}(2^{-d}) \sum_{v \in \mathcal{L}^*/\mathbb{Z}^d} 1 \\
&= (\det \mathcal{L})^{-1} \mathcal{O}(2^{-d}) \det \mathcal{L} \\
&= \mathcal{O}(2^{-d}).
\end{aligned}$$

□

We are finally ready to prove Theorem 5.4.7, the main theorem of this subsection.

Proof of Theorem 5.4.7. We know that Q is a probability distribution, so its ℓ^1 norm is 1, so the ℓ^1 norm of P' is at least $1 - \mathcal{O}(2^{-d})$. We can use this to show that P_2' is very close to P_2. Using $P_2'(w) \leq P_2(w)$ for all $w \in 2^{-t}\mathcal{G}$ (Lemma 5.4.13), we get

$$\left\|P_2 - P_2'\right\|_1 = \|P_2\|_1 - \left\|P_2'\right\| \leq 1 - (1 - \mathcal{O}(2^{-d})) = \mathcal{O}(2^{-d}).$$

Now using Lemma 5.4.15 the triangle inequality gives

$$\|P_2 - Q\|_1 \leq \left\|P_2 - P_2'\right\|_1 + \left\|P_2' - Q\right\|_1 = \mathcal{O}(2^{-d}).$$

Now the ℓ_1 distance between P_1 and P_2 is simply the square of the ℓ_2 distance between the states obtained after applying the quantum Fourier transform to $|\varphi_1\rangle$ and $|\varphi_2\rangle$ respectively. Since the quantum Fourier transform is unitary, it is norm-preserving, so by Claim 5.4.8 this (squared) ℓ^2 distance is $\mathcal{O}(2^{-d})$, so $\|P_1 - P_2\|_1 = \mathcal{O}(2^{-d})$. Finally the triangle inequality gives

$$\|P_1 - Q\|_1 \leq \|P_1 - P_2\|_1 + \|P_2 - Q\|_1 = \mathcal{O}(2^{-d}),$$

as desired. □

We finish this section by proving Theorems 5.4.2 and 5.4.3.

Proof of Theorem 5.4.2. By Theorem 5.4.4 we can prepare the initial state to within an ℓ_2 distance of $1/\operatorname{poly}(d)$ using $\mathcal{O}(d \operatorname{poly}(\log d) + dt)$ quantum gates. By Theorem 5.4.6, we can apply (5.8) in $\mathcal{O}(t(n \log n + d \log^3 d))$ quantum gates. Since unitary operations preserve ℓ_2-distance, this gives a state within $1/\operatorname{poly}(d)$ distance of a state equivalent[5] to $|\varphi_1\rangle$ as defined in (5.9). We then apply the approximate quantum Fourier transform

[5] up to unitary operations that only affect the second register

with error $1/\operatorname{poly}(d)$ to the first register. This costs $\mathcal{O}(dt(\log t + \log d))$ quantum gates. Now if we measure the state and interpret the first register as a vector of binary fraction, this gives a distribution within $1/\operatorname{poly}(d)$ statistical distance of P_1, since our state had an error of at most $1/\operatorname{poly}(d)$. Now by Theorem 5.4.7 and the triangle inequality the distribution of the output of the quantum subroutine is within ℓ_1 statistical distance of Q, as desired. We can add all the complexities to get

$$\mathcal{O}(d\operatorname{poly}(\log d) + dt) + \mathcal{O}(t(n\log n + d\log^3 d)) + \mathcal{O}(dt(\log t + \log d)),$$

which simplifies to

$$\mathcal{O}(t(n\log n + d\log^3 d + d\log t) + d\operatorname{poly}(\log d)).$$

□

Proof of Theorem 5.4.3. We will bound the Gaussian mass

$$\rho_{1/(\sqrt{2}R)}(\{x \in v - 2^{-t}\mathbb{Z}^d \mid \operatorname{dist}_{\mathbb{R}^d/\mathbb{Z}^d}(x,0) > \sqrt{d}/(\sqrt{2}R)\})$$

from above. The $\mathbb{R}^d/\mathbb{Z}^d$ distance from 0 is always at most the standard norm in $\mathbb{R}^d$, so we may bound it from above by

$$\rho_{1/(\sqrt{2}R)}((v - 2^{-t}\mathbb{Z}^d) \setminus \overline{\mathcal{B}_{\sqrt{d}/(\sqrt{2}R)}}).$$

Now we can use Lemma 4.5.4 to bound this from above by

$$2^{-d}\rho_{1/(\sqrt{2}R)}(2^{-t}\mathbb{Z}^d).$$

If we take α_v as defined in (5.17), then we have

$$\frac{\rho_{1/(\sqrt{2}R)}(2^{-t}\mathbb{Z}^d)}{\rho_{1/(\sqrt{2}R)}(v - 2^t\mathbb{Z}^d)} = \frac{\alpha_0}{\alpha_v}.$$

By Lemma 5.4.14 we must have $\alpha_0/\alpha_v \in [1 \pm \mathcal{O}(2^{-d})]$. So we have

$$\begin{aligned} 2^{-d}\rho_{1/(\sqrt{2}R)}(2^{-t}\mathbb{Z}^d) &< 2^{-d}(1 + \mathcal{O}(2^{-d}))\rho_{1/(\sqrt{2}R)}(v - 2^t\mathbb{Z}^d) \\ &= \mathcal{O}(2^{-d})\rho_{1/(\sqrt{2}R)}(v - 2^t\mathbb{Z}^d). \end{aligned}$$

Now dividing this by $\rho_{1/(\sqrt{2}R)}(v - 2^t\mathbb{Z}^d)$ gives an the desired bound. □

5.5. Recovering a lattice vector

In this section we show that given access to the quantum subroutine described in Section 5.4 we can perform a polynomial-time classical algorithm to find a vector in $\mathcal{L} \setminus \mathcal{L}_0$.

If we take m independent samples $w_1, \ldots, w_m$ from our subroutine, we get d-dimensional vectors that are very close to uniformly random vectors $v_1, \ldots, v_m \in \mathcal{L}^*/\mathbb{Z}^d$. Say that for each $1 \leq i \leq m$ we have $\text{dist}_{\mathbb{R}^d/\mathbb{Z}^d}(v_i, w_i) < \delta$ for some constant $\delta > 0$. The trick is to create a basis of a $(d+m)$-dimensional lattice $\mathcal{L}'$ defined by the columns of the following matrix:

$$B = \left(\begin{array}{c|c} I_{d\times d} & 0 \\ \hline \begin{array}{c} \delta^{-1} \cdot w_1^T \\ \vdots \\ \delta^{-1} \cdot w_m^T \end{array} & \delta^{-1} \cdot I_{m\times m} \end{array}\right). \tag{5.18}$$

Let b_i be the columns of B. This lattice has some interesting properties. Take some arbitrary $u = (u_1, \ldots, u_d) \in \mathbb{Z}^d$, we can define a vector

$$\widehat{u} = \sum_{i=1}^{d} u_i b_i \in \mathcal{L}'.$$

The $(d+i)$-th coordinate of $\widehat{u}$ is

$$\begin{aligned} \widehat{u}_{d+i} &= \sum_{j=1}^{d} u_j b_{j,d+i} \\ &= \delta^{-1} \cdot \sum_{j=1}^{d} u_i w_{ij} \\ &= \delta^{-1} \cdot \langle u, w_i \rangle. \end{aligned}$$

Now let k_i be the nearest integer to $\langle u, w_i \rangle$ for $1 \leq i \leq m$ and define the vector

$$u' = \widehat{u} - \sum_{j=1}^{m} k_j b_{d+j},$$

now each of the last m coordinates have the minimal absolute value, namely $\delta^{-1} \cdot \|\langle u, w_i \rangle\|_{\mathbb{R}/\mathbb{Z}}$ for $1 \leq i \leq m$, given that the first d coordinates must form the vector $u \in \mathbb{Z}^d$, so u' is the vector with the smallest norm in $\mathcal{L}'$ that has u as its first d coordinates. We record this fact in a Lemma for later use.

LEMMA 5.5.1. Let $\mathcal{L} \subset \mathbb{Z}^d$ be some d-dimensional lattice. Let $v_1, \ldots, v_m \in \mathcal{L}^*/\mathbb{Z}^d$, $w_1, \ldots, w_m \in \mathbb{R}^d/\mathbb{Z}^d$ and $\delta > 0$. Define the lattice $\mathcal{L}'$ with basis B as defined in (5.18). For each vector $u \in \mathbb{Z}^d$ there exists vectors in $\mathcal{L}'$ whose first d coordinates are u. The

shortest vector in $u' \in \mathcal{L}'$ with this property has (squared) norm

$$\|u'\|^2 = \|u\|^2 + \delta^{-2} \cdot \sum_{i=1}^{m} \|\langle u, w_i \rangle\|^2_{\mathbb{R}/\mathbb{Z}}.$$

Choosing δ small enough (making δ^{-1} large) will make the norm of u' large if $\langle u, w_i \rangle$ isn't close to an integer for every $1 \leq i \leq m$.

Now we explain why we want the vectors w_i to be close approximations of dual lattice vectors. For now say that our approximations are perfect, so we have $w_i = v_i$ for all $1 \leq i \leq m$. Then every w_i is an element of the dual of $\mathcal{L}$, so every $\langle u, w_i \rangle = 0$ if and only if $u \in \mathcal{L}$, since $\mathcal{L}$ is the dual of $\mathcal{L}^*$. If every vector $u \notin \mathcal{L}$ would make at least some $\langle u, v_i \rangle$ differ from an integer by at least some large enough amount, say ϵ, then we can show that every vector in $\mathcal{L}'$ whose first d coordinates are not a vector in $\mathcal{L}$ becomes large, and as such every small vector in $\mathcal{L}'$ must have a vector in $\mathcal{L}$ as its first d coordinates. We can then simply use the LLL-algorithm to find a small vector in $\mathcal{L}'$, since we have an explicit basis for it, and this gives us a vector in $\mathcal{L}$. Now back to the case we actually have, where we have some maximal error $\delta > 0$ such that $\text{dist}_{\mathbb{R}^d/\mathbb{Z}^d}(v_i, w_i) < \delta$ for all $1 \leq i \leq m$. We can use the Cauchy-Schwartz inequality to show that $\langle u, w_i \rangle$ is a good approximation of $\langle u, v_i \rangle$ for small u.

LEMMA 5.5.2. Let $u \in \mathbb{Z}^d$ and $v_1, \ldots v_m, w_1, \ldots, w_m \in \mathbb{R}^d/\mathbb{Z}^d$. Let $\delta > 0$ such that $\|v_i - w_i\|_{\mathbb{R}^d/\mathbb{Z}^d} < \delta$ for all $1 \leq i \leq m$. We have

$$\|\langle u, v_i \rangle - \langle u, w_i \rangle\|_{\mathbb{R}/\mathbb{Z}} < \delta \|u\|$$

for all $1 \leq i \leq m$

Proof. Fix some $1 \leq i \leq m$. By properties of the inner product we have $\langle u, v_i \rangle - \langle u, w_i \rangle = \langle u, v_i - w_i \rangle$. Let $y \in \mathbb{R}^d$ such that $y \in v_i - w_i + \mathbb{Z}^d$ and $\|y\| = \|v_i - w_i\|_{\mathbb{R}^d/\mathbb{Z}^d}$. Since $u \in \mathbb{Z}^d$ we know that $\|\langle u, v_i - w_i \rangle\|_{\mathbb{R}/\mathbb{Z}}$ is well-defined, so we have $\|\langle u, y \rangle\|_{\mathbb{R}/\mathbb{Z}} = \|\langle u, v_j - w_j \rangle\|_{\mathbb{R}/\mathbb{Z}}$. The remainder of the proof follows from the Cauchy-Schwartz inequality:

$$\|\langle u, v_i \rangle - \langle u, w_i \rangle\|_{\mathbb{R}/\mathbb{Z}} = \|\langle u, y \rangle\|_{\mathbb{R}/\mathbb{Z}} \leq |\langle u, y \rangle| \leq \|u\| \cdot \|y\| \leq \delta \|u\|. \qquad \square$$

We want to show that for some fixed $u \in \mathbb{Z}^d \setminus \mathcal{L}$, there exists an index $1 \leq i \leq d$ such that $\langle u, v_i \rangle$ gets far enough from an integer, which would imply the same fact for $\langle u, w_i \rangle$ for small enough u. First we show that for a fixed $u \in \mathbb{Z}^d \setminus \mathcal{L}$, the values $\langle u, v_i \rangle$ mod 1 are uniformly distributed (if we choose v_i uniformly random from $\mathcal{L}^*/\mathbb{Z}^d$) and evenly spaced on the interval $[0, 1)$.

LEMMA 5.5.3. Let $\mathcal{L} \subset \mathbb{Z}^d$ be a d-dimensional lattice. Fix some non-zero $u \in \mathbb{Z}^d/\mathcal{L}$. The distribution of $\langle u, v \rangle$ mod 1 where v is uniformly chosen from $\mathcal{L}^*/\mathbb{Z}^d$ is the uniform distribution over the cyclic group $\mathcal{C}_k = \{0, 1/k, 2/k, \ldots, (k-1)/k\}$ for some $k \geq 2$.

Moreover $h(v) = \langle u, v \rangle \bmod 1$ is a group homomorphism.

Proof. The fact that h is a group homomorphism is clear by properties of the inner product. Thus by the First Isomorphism Theorem we have

$$\mathcal{I} := \left\{ \langle u, v \rangle \bmod 1 \mid v \in \mathcal{L}^*/\mathbb{Z}^d \right\} \cong (\mathcal{L}^*/\mathbb{Z}^d)/\ker h.$$

If you choose a uniformly random element from $\mathcal{L}^*/\mathbb{Z}^d$ it is in a uniformly random coset of $\ker h$, so the distribution over $\mathcal{I}$ is uniform. Note that $\mathcal{I}$ is a finite abelian group, we will show that this group is in fact cyclic. We prove this using the following fact: If $x_1, x_2 \in \mathcal{I}$ with order r_1 and r_2 respectively, then $1/\operatorname{lcm}(r_1, r_2) \in \mathcal{I}$, this means that $x_1, x_2 \in \langle 1/\operatorname{lcm}(r_1, r_2) \rangle$, so you can find an element in $\mathcal{I}$ that generates the entire group.

Say we have $\langle u, v \rangle = a/r$ with $\gcd(a, r) = 1$, this element has order r and each of its multiples can be written as a fraction with denominator r. This implies that every fraction in $[0, 1)$ with denominator r is in $\mathcal{I}$. Now if we have elements of order r_1 and r_2 in the group, call them x_1 and x_2 respectively, the element $x' = x^{\gcd(r_1, r_2)}$ has order $r_1/\gcd(r_1, r_2)$. Since r_2 and $r_1/\gcd(r_1, r_2)$ are coprime the order of $x' + y$ is $r_1 r_2/\gcd(r_1, r_2) = \operatorname{lcm}(r_1, r_2)$, meaning $1/\operatorname{lcm}(r_1, r_2) \in \mathcal{I}$.

Thus we have $\mathcal{I} = \mathcal{C}_k$ for some k. We must have $k \geq 2$, since otherwise h sends everything to 0, so $\langle u, v \rangle \in \mathbb{Z}^d$ for all $v \in \mathcal{L}^*$, which means $u \in \mathcal{L}$, but this is not possible because it would be zero in $\mathbb{Z}^d/\mathcal{L}$. □

We can show that it if $m \geq d + 4$ it suffices to take $\epsilon = (2 \det \mathcal{L})^{-1/m}/3$ to get a probability of at least $1/2$ that for each $u \in \mathbb{Z}^d \setminus \mathcal{L}$ there exists an index $1 \leq i \leq m$ such that $\langle u, v_i \rangle \bmod 1 \notin [\pm\epsilon]$. To prove this we need the fact that if you take at least $d + 4$ independent uniform samples from $\mathcal{L}^*/\mathbb{Z}^d$, there is a probability of at least $1/2$ that these vectors generate $\mathcal{L}^*$.

THEOREM 5.5.4. [Pom01]. Suppose G is a finite abelian group with minimal number of generators r. The expected number of elements from G (chosen independently and with the uniform distribution) so that the elements chosen generate G is at most $r + \sigma$, where $\sigma \approx 2.12$.

COROLLARY 5.5.5. In a finite abelian group G that can be generated with r elements, $r + 4$ uniformly random elements of G generate G with probability at least $1/2$.

Proof. Assume, for a contradiction, that with probability at least $1/2$ you need at least $r + 5$ uniformly random elements to generate G. We know there is 0 probability that you can generate G in less than r elements, so the expectation is at least $r + 5/2 > r + \sigma$, which contradicts Theorem 5.5.4. □

LEMMA 5.5.6. Let $\mathcal{L} \subset \mathbb{Z}^d$ be a d-dimensional lattice and let $m \geq d+4$. Assume $v_1, \ldots, v_m$ are uniformly chosen from $\mathcal{L}^*/\mathbb{Z}^d$. With probability at least $1/2$, we have that for every $u \in \mathbb{Z}^d \setminus \mathcal{L}$, there exists an i such that $\langle u, v_i \rangle \notin [\pm\epsilon] \bmod 1$, where $\epsilon = (2 \det \mathcal{L})^{-1/m}/3$.

Proof. By Lemma 5.5.3 the distribution of $\langle u, v \rangle \bmod 1$ where v is chosen uniformly from $\mathcal{L}^*/\mathbb{Z}^d$ is the uniform distribution over the cyclic group $\mathcal{C}_k$ for some $k \geq 2$.

First assume $k \geq 1/\epsilon$, we calculate the probability that $\langle u, v_j \rangle \in [\pm\epsilon] \bmod 1$ for fixed j. The element v_j was uniformly chosen from $\mathcal{L}^*/\mathbb{Z}^d$, so $\langle u, v_j \rangle \bmod 1$ is uniformly chosen from $\mathcal{S}_k$. Since $a/k \leq \epsilon \Leftrightarrow a \leq \lfloor \epsilon k \rfloor$, we have $\lfloor \epsilon k \rfloor + 1$ values within distance ϵ of 0 (including 0). Furthermore, by the same logic, we have $\lfloor \epsilon k \rfloor$ values within ϵ distance of 1 (excluding 1), so in total $1 + 2\lfloor \epsilon k \rfloor$ of the k elements of $\mathcal{S}_k$ are within ϵ distance of an integer, so the probablity that $\langle u, v_j \rangle \in [\pm\epsilon] \bmod 1$ is

$$\frac{1 + 2\lfloor \epsilon k \rfloor}{k}.$$

Since we assumed $k \geq 1/\epsilon$, we have $1/k \leq \epsilon$, giving

$$\frac{1 + 2\lfloor \epsilon k \rfloor}{k} \leq \frac{1}{k} + \frac{2\epsilon k}{k} \leq 3\epsilon.$$

Since every v_j is independent the probability that each $\langle u, v_j \rangle$ is in $[\pm\epsilon]$ is at most $(3\epsilon)^m$. Now applying the union bound, the probability that any non-zero $u \in \mathbb{Z}^d/\mathcal{L}$ does not have a i such that $\langle u, v_i \rangle \notin [\pm\epsilon]$ is at most

$$(\det \mathcal{L} - 1)(3\epsilon)^m = \frac{\det \mathcal{L} - 1}{2 \det \mathcal{L}} \leq \frac{1}{2}$$

So the probability that for all non-zero $u \in \mathbb{Z}^d/\mathcal{L}$, there exists a j such that $\langle u, v_j \rangle \notin [\pm\epsilon]$ is at least $1/2$.

Now for the case where $k < 1/\epsilon$. We would like to prove that for every non-zero $u \in \mathbb{Z}^d/\mathcal{L}$ there exists an $1 \leq i \leq m$ such that $\langle u, v_i \rangle \bmod 1 \neq 0$. Because of $1/k > \epsilon$ this would imply that $\langle u, v_i \rangle \bmod 1 \notin [\pm\epsilon]$. Since u is not in the dual of $\mathcal{L}^*$, there must be a $v \in \mathcal{L}^*/\mathbb{Z}^d$ such that we have $\langle u, v \rangle \bmod 1 \neq 0$. If for some non-zero $u \in \mathbb{Z}^d/\mathcal{L}$ we have for each $1 \leq i \leq m$ that $\langle u, v_i \rangle \in \mathbb{Z}$, then each integer combination of $v_1, \ldots, v_m$ also has an integral inner product with u. Thus if $v_1, \ldots, v_m$ generate the group $\mathcal{L}^*/\mathbb{Z}^d$, then for each non-zero $u \in \mathbb{Z}^d/\mathcal{L}$ there must be an $1 \leq i \leq m$ such that $\langle u, v_i \rangle \bmod 1 \notin [\pm\epsilon]$. According to Corollary 5.5.5 this happens with probability at least $1/2$ over the choice of $v_1, \ldots, v_m$. □

We are now ready to prove that the every "small" vector in the lattice $\mathcal{L}'$ with basis as defined in (5.18) has a vector in $\mathcal{L}$ as its first d vectors (with probability at least $1/2$)

THEOREM 5.5.7. Let $\mathcal{L} \subset \mathbb{Z}^d$ be a d-dimensional lattice and let $m \geq d+4$. Assume $v_1, \ldots, v_m$ are uniformly chosen vectors from $\mathcal{L}^*/\mathbb{Z}^d$. For some $\delta > 0$ let $w_1, \ldots, w_m \in \mathbb{R}^d/\mathbb{Z}^d$ be such that for each $1 \leq i \leq m$ we have $\mathrm{dist}_{\mathbb{R}^d/\mathbb{Z}^d}(w_i, v_i) < \delta$. Define the lattice $\mathcal{L}'$ as the $(d+m)$-dimensional lattice generated by the basis matrix B as defined in (5.18). Then, for any lattice vector $u \in \mathcal{L}$, there exists a vector $u' \in \mathcal{L}'$ whose first d coordinates are u, and whose norm is less than $\|u\| \cdot \sqrt{m+1}$.

Moreover, with probability at least $1/2$ we have that any non-zero vector $u' \in \mathcal{L}'$ of norm less than $\delta^{-1} \cdot \epsilon/2$ has a non-zero vector $u \in \mathcal{L}$ as its first d coordinates, where $\epsilon = (2 \det \mathcal{L})^{-1/m}/3$.

Proof. Let $u \in \mathcal{L}$. By Lemma 5.5.1 there exists a vector $u' \in \mathcal{L}$ whose first d coordinates are u such that

$$\|u'\|^2 = \|u\|^2 + \delta^{-2} \cdot \sum_{i=1}^{m} \|\langle u, w_i \rangle\|_{\mathbb{R}/\mathbb{Z}}^2.$$

Now Lemma 5.5.2 states that the distance from $\langle u, w_i \rangle$ to $\langle u, v_i \rangle$ is less than $\delta\|u\|$, since $\langle u, v_i \rangle$ is an integer we have that $\|\langle u, w_i \rangle\|_{\mathbb{R}/\mathbb{Z}} < \delta\|u\|$, giving the bound

$$\begin{aligned} \|u'\|^2 &< \|u\|^2 + \delta^{-2} \cdot \sum_{i=1}^{m} \delta^2 \|u\|^2 \\ &= \|u\|^2 (m+1). \end{aligned}$$

Taking the square root gives the desired bound.

By Lemma 5.5.6 there is a probability of at least $1/2$ over the choice of $v_1, \ldots, v_m$ that for every non-zero $u \in \mathbb{Z}^d/\mathcal{L}$ there exists an i such that $\langle u, v_i \rangle \notin [\pm\epsilon] \bmod 1$. Assuming this is the case, we show that any non-zero vector in $\mathcal{L}'$ which does not have a non-zero vector in $\mathcal{L}$ as its first d coordinates must have norm at least $\delta^{-1} \cdot \epsilon/2$. This implies that any non-zero vector in $\mathcal{L}'$ with norm less than $\delta^{-1} \cdot \epsilon/2$ has a non-zero vector in $\mathcal{L}$ as its first d coordinates, as desired.

First let $u' \in \mathcal{L}'$ be a non-zero vector which has 0 as its first d coordinates. The last m coordinates form a non-zero vector in $\delta^{-1}\mathbb{Z}^d$, which has norm at least $\delta^{-1} > \delta^{-1} \cdot \epsilon/2$, as desired.

Now let $u' \in \mathcal{L}'$ be a (non-zero) vector such that its first d coordinates form a (non-zero) vector $u \in \mathbb{Z}^d \setminus \mathcal{L}$, such that $\|u\| \leq \delta^{-1} \cdot \epsilon/2$ (otherwise we immediately have $\|u'\| \geq \|u\| > \delta^{-1} \cdot \epsilon/2$). By Lemma 5.5.1 we have

$$\|u'\|^2 \geq \|u\|^2 + \delta^{-2} \cdot \sum_{i=1}^{m} \|\langle u, w_i \rangle\|_{\mathbb{R}/\mathbb{Z}}^2. \tag{5.19}$$

Choose an i such that $\|\langle u, v_i \rangle\|_{\mathbb{R}/\mathbb{Z}} > \epsilon$. By Lemma 5.5.2 the distance between $\langle u, v_i \rangle$ and $\langle u, w_i \rangle$ is less than $\delta\|u\|$. This implies that

$$\|\langle u, w_i \rangle\|_{\mathbb{R}/\mathbb{Z}} > \epsilon - \delta\|u\| \geq \epsilon - \epsilon/2 = \epsilon/2.$$

Together with the bound in (5.19) this gives the desired bound $\|u'\| > \delta \cdot \epsilon/2$. □

Now all we have to do is find a small enough lattice vector in $\mathcal{L}'$ using the LLL-algorithm, then its first d coordinates must be a vector in $\mathcal{L}$. However, we actually want a vector in $\mathcal{L} \setminus \mathcal{L}_0$. If we have some upper bound for the smallest vector in $\mathcal{L} \setminus \mathcal{L}_0$, we can use the LLL-algorithm to find a list of vectors that generate every vector in $\mathcal{L}$ up to that norm bound. Since this is a vector that generates at least one vector in $\mathcal{L} \setminus \mathcal{L}_0$, it must contain a vector in $\mathcal{L} \setminus \mathcal{L}_0$.

LEMMA 5.5.8. Given a basis of a lattice $\mathcal{L} \subset \mathbb{R}^d$ and some norm bound $M > 0$. There exists a polynomial-time algorithm that can compute a list of $\ell \leq d$ vectors $z_1, \ldots, z_\ell \in \mathcal{L}$ of norm at most $\sqrt{d}2^{d/2}M$ such that every lattice vector in $\mathcal{L}$ of norm less than M is an integer combination of $z_1, \ldots, z_\ell$.

Proof. By Theorem 4.6.4 we can obtain an LLL-reduced basis of $\mathcal{L}$ in polynomial time. Let $z_1, \ldots, z_d \in \mathcal{L}$ be an LLL-reduced basis and let $\widetilde{z}_1, \ldots, \widetilde{z}_d$ be its Gram-Schmidt orthogonalization, which can be computed in polynomial time. Let ℓ be the smallest index such that $\|\widetilde{z}_{\ell+1}\| \geq 2^{d/2}M$, or let it equal d if such an index does not exist. We will show that $z_1, \ldots, z_\ell$ are the desired lattice vectors.

For all $1 \leq i < d$ we have that $\|\widetilde{z}_{i+1}\| \geq \|\widetilde{z}_i\|/\sqrt{2}$ By Proposition 4.6.5. Now for each $i > \ell$ we have

$$\|\widetilde{z}_i\| \geq (1/\sqrt{2})^{i-\ell-1}\|\widetilde{z}_{\ell+1}\| \geq 2^{d/2} \cdot 2^{-(i-\ell)/2}M \geq M.$$

Take a vector $v \in \mathcal{L}$, we can write it in terms of the basis $z_1, \ldots, z_d$ as

$$v = \sum_{i=1}^{d} a_i z_i,$$

with $a_i \in \mathbb{Z}$. Then the norm of v is at least as large as the basis vector z_i with the largest norm such that $a_i \neq 0$. From Proposition 4.6.6 we have $\|z_i\| \geq \|\widetilde{z}_i\|$, so if $\|v\| < M$ we must have $a_i = 0$ for $i > \ell$. This shows that every lattice vector in $\mathcal{L}$ of norm less than M is an integer combination of $z_1, \ldots, z_\ell$. Proposition 4.6.6 gives for $1 \leq i \leq \ell$

$$\|z_i\| \leq \sqrt{\sum_{j=1}^{i} \|\widetilde{z}_j\|^2} < \sqrt{d2^d M^2} = \sqrt{d}2^{d/2}M,$$

as desired. □

THEOREM 5.5.9. Let $\mathcal{L}_0$ be a sublattice of the d-dimensional lattice $\mathcal{L}$. Let T be an upper bound on the norm of the smallest vector in $\mathcal{L} \setminus \mathcal{L}_0$. Say we have uniformly random $v_1, \ldots, v_m \in \mathcal{L}^*/\mathbb{Z}^d$ and $w_1, \ldots, w_m \in \mathbb{R}^d/\mathbb{Z}^d$. Let $\delta > 0$ be such that for all $1 \leq i \leq m$ we have $\mathrm{dist}_{\mathbb{R}^d/\mathbb{Z}^d}(w_i, v_i) < \delta$. If we have

$$T \cdot 2^{(d+m)/2} \cdot \sqrt{(m+1)(d+m)} < \delta^{-1} \cdot \epsilon/2,$$

with $\epsilon = (2 \det \mathcal{L})^{-1/m}/3$, then with probability at least $1/2$ over the choice of $v_1, \ldots, v_m$

we can obtain a vector in $\mathcal{L} \setminus \mathcal{L}_0$ in polynomial time.

Proof. Apply the algorithm described in Lemma 5.5.8 to the basis B as defined in (5.18) with norm bound $T\sqrt{m+1}$. This returns a list of vectors $z_1, \ldots, z_\ell \in \mathbb{Z}^d$ of norm at most

$$T \cdot 2^{(d+m)/2} \cdot \sqrt{(m+1)(d+m)}$$

such that the first d coordinates of every vector in $\mathcal{L}'$ of norm less than $T\sqrt{m+1}$ is an integer combination of $z_1, \ldots, z_\ell$. Now by Theorem 5.5.7, with probability at least $1/2$ over the choice of $v_1, \ldots, v_m$, we have that $z_1, \ldots, z_\ell \in \mathcal{L} \setminus \{0\}$.

There exists a $u \in \mathcal{L} \setminus \mathcal{L}_0$ with norm less than T, by Theorem 5.5.7 we know that there must exist a lattice vector $u' \in \mathcal{L}'$ of norm less than $T\sqrt{m+1}$ which has u as its first d coordinates. This vector must be in the span of $z_1, \ldots, z_\ell$, and since $\mathcal{L}_0$ is a sublattice of $\mathcal{L}$, there exists an index $1 \leq i \leq \ell$ such that $z_\ell \in \mathcal{L} \setminus \mathcal{L}_0$. □

5.6. Putting it all together

LEMMA 5.6.1. For an n-bit integer N and some vector of d primes b and the vector of its squares a, define the d-dimensional lattice and sublattice $\mathcal{L}$ and $\mathcal{L}_0$ as in (5.1) and (5.2) respectively. Let T be an upper bound on the norm of the smallest vector in $\mathcal{L} \setminus \mathcal{L}_0$. If we set the parameter

$$R = 6T \cdot 2^{(d+m)/2+(n+1)/m} \cdot \sqrt{d(m+1)(d+m)/2}, \tag{5.20}$$

Then after making $m \geq d+4$ calls to the quantum subroutine described in Section 5.4 we can, with probability at least $1/2 - \mathcal{O}(1/\operatorname{poly}(d))$, obtain a vector in $\mathcal{L} \setminus \mathcal{L}_0$ in polynomial time.

Proof. By Theorem 5.4.2, within statistical distance $1/\operatorname{poly}(d)$, the output of the m calls to the quantum subroutine will be samples from the distributions Q_{v_i}, as defined in (5.4), for $1 \leq i \leq m$, where $v_1, \ldots, v_m$ are independent uniform samples from $\mathcal{L}^*/\mathbb{Z}^d$. Set $\delta = \sqrt{d}/(\sqrt{2}R)$. By Theorem 5.4.3 we obtain vectors $w_1, \ldots, w_m \in \mathbb{R}^d$ such that, with all but probability $\mathcal{O}(2^{-d})$, we have $\operatorname{dist}_{\mathbb{R}^d/\mathbb{Z}^d}(w_i, v_i) < \delta$, for all $1 \leq i \leq m$. Set $\epsilon = (2 \det \mathcal{L})^{-1/m}/3$. Now we have

$$\begin{aligned}
\delta^{-1} \cdot \epsilon/2 &= \frac{\sqrt{2}R}{\sqrt{d}} \cdot (2 \det \mathcal{L})^{-1/m}/6 \\
&= T \cdot 2^{(d+m)/2+(n+1)/m} \cdot \sqrt{(m+1)(d+m)} \cdot (2 \det \mathcal{L})^{-1/m} \\
&> T \cdot 2^{(d+m)/2+(n+1)/m} \cdot \sqrt{(m+1)(d+m)} \cdot 2^{-(n+1)/m} \\
&= T \cdot 2^{(d+m)/2} \cdot \sqrt{(m+1)(d+m)},
\end{aligned}$$

where the inequality follows from $\det \mathcal{L} < 2^n$. The proof now follows from Theorem 5.5.9. □

Now we can prove the main theorem of this thesis:

THEOREM 5.6.2. Assuming that Assumption 5.1.2 is true, there exists a polynomial-time classical algorithm that can find (with high probability) a non-trivial factor of an n-bit integer N using $\mathcal{O}(\sqrt{n})$ calls to a quantum circuit of size $\mathcal{O}(n^{3/2} \log n)$.

Proof. If N is even or a prime power we already know how to efficiently find a non-trivial factor, so we may assume N is odd and not a prime power. Set parameters $d \approx \sqrt{n}$ and R as defined in (5.20), note that $R > \sqrt{2d}$. Choose a vector of $\mathcal{O}(\log n)$-bit primes $b = (b_1, \ldots, b_d)$ and set $a = (b_1^2, \ldots, b_d^2)$. Define the d-dimensional lattice and sublattice $\mathcal{L}$ and $\mathcal{L}_0$ as in (5.1) and (5.2) respectively. Set $m = d + 4$. By Lemma 5.6.1 we can make $m = \mathcal{O}(\sqrt{n})$ calls to the quantum subroutine to get a vector in $\mathcal{L} \setminus \mathcal{L}_0$ with probability at least $1 - \text{poly}(d)$. We can repeat this a constant number of times to get a probability of success arbitrarily close to 1. Now by Theorem 5.2.1 we can find a non-trivial factor of N in polynomial time.

Recall that t is the highest exponent we need to calculate in superposition, and that we have $t = 1 + \lceil \log_2(\sqrt{d}R) \rceil$. We have that

$$\begin{aligned} t &= \mathcal{O}(\log d + \log R) \\ &= \mathcal{O}(\log d + \log T + d + m + n/m + \log(d^2 m)) \\ &= \mathcal{O}(\log T + d + n/d). \end{aligned}$$

If Assumption 5.1.2 is true, we have $\mathcal{O}(t) = \mathcal{O}(d + n/d)$. Now filling in $d \approx \sqrt{n}$, we get $t = \mathcal{O}(\sqrt{n})$. Now filling in $t = \mathcal{O}(\sqrt{n})$ and $d \approx \sqrt{n}$ into Theorem 5.4.2 we find that the quantum subroutine can be implemented with a quantum circuit of size

$$\mathcal{O}(n^{3/2} \log n),$$

as desired. □

5.6.1. Recap

We finish this chapter by giving a step-by-step recap of the entire algorithm. We want to find a non-trivial factor of an n-bit integer N. We may assume that N is odd and not a prime power. Let $d \approx \sqrt{n}$ be an integer. Choose a vector of d distinct random $\mathcal{O}(\log n)$-bit primes $b = (b_1, \ldots, b_d)$ and define $a = (b_1^2, \ldots, b_d^2)$. This defines the d-dimensional lattice

$$\mathcal{L} = \left\{ z \in \mathbb{Z}^d \mid a^z \equiv 1 \bmod N \right\}$$

and the sublattice

$$\mathcal{L}_0 = \left\{ z \in \mathbb{Z}^d \mid b^z \equiv \pm 1 \bmod N \right\} \subset \mathcal{L}.$$

Let T be an upper bound on the norm of the smallest vector in $\mathcal{L} \setminus \mathcal{L}_0$. Choose

$$R = 6T\sqrt{(d+5)(2d+4)(d/2)} \cdot 2^{(n+1)/(d+4)+d+2}$$

and $t = 1 + \lceil \log_2(\sqrt{d}R) \rceil$. Let $\mathcal{G} = \{-2^t, \ldots, 2^t - 1\}^d$.

Then, on a quantum computer, we approximate (within $1/\operatorname{poly}(d)$) the state proportional to

$$\sum_{z \in \mathcal{G}} \rho_R(z)|z\rangle|1\rangle,$$

and apply $|z\rangle|1\rangle \mapsto |z\rangle|a^{z+2^{t-1}}\rangle$ to this state. We apply a quantum Fourier transform over $\mathbb{Z}/2^t\mathbb{Z}$ to the first register. Then we measure both registers, read the output of the first register as a vector of unsigned integers and divide it by 2^t. If Assumption 5.1.2 is true, this quantum circuit can be made with $\mathcal{O}(n^{3/2}\log n)$ quantum gates. The output of the quantum subroutine is equivalent to choosing a uniform sample $v \in \mathcal{L}^*/\mathbb{Z}^d$ and returning a vector $w \in \mathbb{R}^d/\mathbb{Z}^d$ that is within distance $\delta = \sqrt{d}/(\sqrt{2}R)$ of v (with all but probability $1/\operatorname{poly}(d)$). We run this circuit $d+4$ times independently, yielding $d+4$ vectors $w_1, \ldots, w_{d+4} \in \mathbb{R}^d$. Let $\delta = \sqrt{d}/(\sqrt{2}R)$. We then apply the LLL-algorithm to the columns of the matrix

$$\left(\begin{array}{c|c} I_{d\times d} & 0 \\ \hline \begin{array}{c} \delta^{-1} \cdot w_1^T \\ \vdots \\ \delta^{-1} \cdot w_{d+4}^T \end{array} & \delta^{-1} \cdot I_{d+4\times d+4} \end{array}\right).$$

This gives an LLL-reduced basis and we return all the vectors of this basis up to but not including the first vector whose corresponding vector in the Gram-Schmidt orthogonalization has norm at least $T\sqrt{d+5}$. This results in $\ell \leq 2d+4$ vectors $z'_1, \ldots z'_\ell \in \mathbb{Z}^d \oplus \mathbb{R}^{d+4}$. Then taking the first d coordinates of each of these vectors yields ℓ vectors $z_1, \ldots, z_\ell \in \mathbb{Z}^d$ such that (with probability at least $1/2$) at least one of them lies in $\mathcal{L} \setminus \mathcal{L}_0$. Now we simply calculate b^{z_i} for $1 \leq i \leq \ell$, then if this is not ± 1, we calculate $\gcd(N, b^{z_i} - 1)$ and $\gcd(N, b^{z_i} + 1)$, and with probability at least $1/2 - 1/\operatorname{poly}(d)$ this will be a non-trivial factor of N.

6. Discussion

Although Regev's algorithm has an asymptotically faster quantum subroutine than Shor's algorithm, this does not necessarily imply that it is the better choice for practical problems. The main use for factoring is breaking encryption schemes that assume that integer factoring is a hard problem, these schemes generally use 2048 or 4096-bit integers. A recent paper states that in its current form, Regev's algorithm does not seem to have an advantage when factoring integers of these sizes [EG24]. Thus further optimization would be needed for Regev's algorithm to break such encryption schemes before the best known optimized versions of Shor's algorithm would.

Further research would be necessary to show that Assumption 5.1.2 is true, which would make sure we do not need to use the slightly slower algorithm given in [Pil24]. Another possible improvement is in the preparation of the initial state, it might also work to use a uniform superposition instead of a Gaussian superposition, although the calculations might not be as nice. This wouldn't make the algorithm any faster asymptotically, but it might reduce the cost for values of N low enough to be relevant in practice, and it would also be easier to implement.

Ethics

At the time of writing, a quantum computer that is even remotely close to being powerful enough to run either Shor's algorithm or Regev's algorithm on cryptographically relevant numbers does not exist. However, developing algorithms like Regev's algorithm has the potential to make the ability to factor large numbers possible sooner. The first quantum computer powerful enough (if it will ever be created) would be able to decrypt any data that has been encrypted using any scheme dependent on the assumption that factoring is hard. This would be a disaster for security and privacy.

Despite the risks that the development of these algorithms pose, it is important to know what possibilities more powerful quantum computers would bring. Currently different encryption schemes that should be robust against quantum computers are being considered, we need to know the capabilities of quantum algorithms to know if these new schemes are secure.

7. Conclusion

In this thesis we presented and compared Shor's and Regev's quantum factoring algorithms. We proved that, assuming a natural statement about existance of certain elements in lattices, the quantum subroutine in Regev's algorithm is asymptotically cheaper in terms of the number of quantum gates than that of Shor's algorithm. We adapted the proofs of the claims made in Regev's paper in such a way that the reader does not need any familiarity with lattices. We gave the prerequisite knowledge of lattices that is necessary to understand the proof of the correctness and efficiency of Regev's algorithm.

Combined with the improvements given in [RV24], which reduces the size complexity to near that of Shor's algorithm and allows some of the calls to the quantum subroutine to fail, it is quite possible that Regev's algorithm requires a less powerful quantum computer to factor integers of sizes used in practice than Shor's algorithm.

While a slightly slower version of Regev's algorithm is known to work [Pil24] without any unproven assumptions, it is still expected that the assumption used in Regev's algorithm is true.

Further research is required to find out how high the constants are in the complexity of Regev's algorithm, which would indicate how fast is should run in practice and if it is actually better than Shor's algorithm for breaking encryption schemes relying on the hardness of factoring.

In conclusion, while Shor's algorithm might still be the best quantum factoring algorithm for practical problems, Regev's algorithm gives a promising alternative and shows that Shor's algorithm may not be as optimal as previously thought.

Bibliography

[Ban93] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(4):625–636, 1993.

[Cop02] D. Coppersmith. An approximate fourier transform useful in quantum factoring, 2002.

[Dad18] D. Dadush. Introduction to lattices algorithms and cryptography. https://homepages.cwi.nl/~dadush/teaching/lattices-2018/, 2018.

[EG24] Martin Ekerå and Joel Gärtner. A high-level comparison of state-of-the-art quantum algorithms for breaking asymmetric cryptography, 2024.

[GR02] L. Grover and T. Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions, 2002.

[HvdH21] D. Harvey and J. van der Hoeven. Integer multiplication in time $\mathcal{O}(n \log n)$. *Ann. of Math.*, 2021.

[HW79] G. H. Hardy and E. M. Wright. *An Introduction to the Theory of Numbers*. Oxford University Press, 1979.

[LLL82] A.K. Lenstra, H.W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 1982.

[NC00] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[Pil24] Cédric Pilatte. Unconditional correctness of recent quantum algorithms for factoring and computing discrete logarithms, 2024.

[Pom01] C. Pomerance. The expected number of random elements to generate a finite abelian group, 2001.

[Reg04] O. Regev. Lecture notes on lattices in computer science. https://cims.nyu.edu/~regev/teaching/lattices_fall_2004/ln/, 2004.

[Reg24] O. Regev. An efficient quantum factoring algorithm. https://arxiv.org/abs/2308.06572, 2024.

[RV24] Seyoon Ragavan and Vinod Vaikuntanathan. Space-efficient and noise-robust quantum factoring, 2024.

[Sho97] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997.

[Ste12] B. Steinberg. *Representation Theory of Finite Groups: An Introductory Approach*. Springer, 2012.

Popular summary

The ability to encrypt data in a way that nobody without the "key" can read it relies on having operations that can be easily done in one direction, but that are very hard to do in the opposite direction, even with a powerful computer. One of these operations is multiplying two numbers. If you have two large number x and y, then we can easily compute the product $x \cdot y$, but if you are given just the (very large) number $x \cdot y$, there is currently no fast way to find out what x and y are on a classical (non-quantum) computer. In other words, finding the factors of general large numbers is hard.

However, quantum computers can do some specific things much faster than a regular computer. Using these properties Peter Shor showed an efficient algorithm to find the factors of an integer in 1994. Meaning a quantum computer doesn't have to be nearly as powerful as a regular computer to factor large numbers. However, current quantum computers are still not even close to powerful enough to compute even relatively small numbers, let alone large enough numbers to break encryption. This thesis describes a new algorithm, developed by Oded Regev in 2023, that is a variation on Shor's algorithm, that in theory would require an even less powerful quantum computer. This means that if quantum computers continue to get better in the future, encryption that relies on the assumption that finding the factors of large numbers is hard to do could be broken sooner than we previously would have thought. However, it is still unclear if Regev's algorithm will actually be faster for the particular numbers that would need to be factored to break encryption. All we know is that as the numbers to be factored get larger and larger, there is some point at which Regev's algorithm will become faster than Shor's algorithm. We do not yet know exactly where this point is, and it could be that Regev's algorithm is only faster for numbers much larger than are used in practice.

A. Character theory

A.1. Background

Definition A.1.1 (Character of a finite abelian group). Let A be a finite abelian group, then a group homomorphism $\chi\colon A \to \mathbb{C}^\times$ is called a *character* of A.

We define an inner product on complex-valued functions from a finite group

Definition A.1.2 (Inner product). [Ste12, Def. 4.2.1]. Let G be a finite group, with function $f_1, f_2\colon G \to \mathbb{C}$, then the following is an inner product on the space of complex-valued functions from G:

$$\langle f_1, f_2\rangle := \frac{1}{|G|}\sum_{g\in G} f_1(g)\overline{f_2(g)}.$$

We will need the following basic fact about characters:

Theorem A.1.3 (First orthogonality relations). [Ste12, Thm. 4.3.9]. The characters of a finite abelian group form an orthonormal set under the inner product defined in A.1.2.

A.2. Proof of (5.16)

Lemma A.2.1. Take $v \in \mathcal{L}^*/\mathbb{Z}^d$. The function $\chi_v\colon \mathbb{Z}^d/\mathcal{L} \to \mathbb{C}^*$ given by

$$\chi_v(x) := e^{2\pi i\langle v,x\rangle},$$

is a character of $\mathbb{Z}^d/\mathcal{L}$. Moveover, the function $v \mapsto \chi_v$ is injective.

Proof. We show that $\chi_{v_1} = \chi_{v_2}$ if and only if $v_1 - v_2 \in \mathbb{Z}^d$. If assume $v_1 - v_2 \in \mathbb{Z}^d$, then

$$\chi_{v_1}(x)\cdot\chi_{v_2}(x)^{-1} = e^{2\pi i\langle v_1-v_2,x\rangle},$$

now both $v_1 - v_2$ and x are vectors in $\mathbb{Z}^d$, so $\langle v_1 - v_2, x\rangle \in \mathbb{Z}$, meaning the exponent becomes an integer multiple of $2\pi i$, so $\chi_{v_1}(x) = \chi_{v_2}(x)$. If we assume $\chi_{v_1} = \chi_{v_2}$, we have $\chi_{v_1-v_2}(x) \equiv 1$, so $\langle v_1 - v_2, x\rangle \in \mathbb{Z}$ for all $x \in \mathbb{Z}^d$, so we must have $v_1 - v_2 \in \mathbb{Z}^d$.

Now we show that the function $\chi_v(x)$ is well-defined. If we take x_1, x_2 from the same coset of $\mathcal{L}$, then

$$\chi_v(x_1) \cdot \chi_v(x_2)^{-1} = e^{2\pi i \langle v, x_1 - x_2 \rangle},$$

this is 1 since $x_1 - x_2 \in \mathcal{L}$ and $v \in \mathcal{L}$, so by definition of the dual lattice we have $\langle v, x_1 - x_2 \rangle \in \mathbb{Z}$.

Finally we show that χ_v is a homomorphism:

$$\chi_v(x_1 + x_2) = e^{2\pi i \langle v, x_1 + x_2 \rangle} = e^{2\pi i \langle v, x_1 \rangle} \cdot e^{2\pi i \langle v, x_2 \rangle} = \chi_v(x_1) \cdot \chi_v(x_2).$$

□

CLAIM A.2.2. We have that

$$\sum_{u \in \mathbb{Z}^d / \mathcal{L}} e^{2\pi i \langle v_1 - v_2, u \rangle} = \begin{cases} \det \mathcal{L} & \text{if } v_1 - v_2 \in \mathbb{Z}^d; \\ 0 & \text{else.} \end{cases}$$

Proof. Using Lemma A.2.1 and Theorem A.1.3 we get

$$\begin{aligned} \sum_{u \in \mathbb{Z}^d / \mathcal{L}} e^{2\pi i \langle v_1 - v_2, u \rangle} &= \sum_{u \in \mathbb{Z}^d / \mathcal{L}} e^{2\pi i \langle v_1, u \rangle} \cdot \overline{e^{2\pi i \langle v_2, u \rangle}} \\ &= \det \mathcal{L} \cdot \langle \chi_{v_1}, \chi_{v_2} \rangle \\ &= \begin{cases} \det \mathcal{L} & \text{if } \chi_{v_1} = \chi_{v_2}; \\ 0 & \text{else.} \end{cases} \\ &= \begin{cases} \det \mathcal{L} & \text{if } v_1 - v_2 \in \mathbb{Z}^d \\ 0 & \text{else.} \end{cases} \end{aligned}$$

□