A quantum algorithm for computing the unit group of an arbitrary degree number field

Kirsten Eisenträger[*]
Department of Mathematics
The Pennsylvania State University
eisentra@math.psu.edu
and Harvard University

Sean Hallgren[†]
Dept. of Computer Science and Engineering
The Pennsylvania State University
hallgren@cse.psu.edu

Alexei Kitaev
Kavli Institute for Theoretical Physics
University of California, Santa Barbara
kitaev@kitp.ucsb.edu
and California Institute of Technology

Fang Song
Department of Combinatorics & Optimization
and Institute for Quantum Computing
University of Waterloo
fang.song@uwaterloo.ca

ABSTRACT

Computing the group of units in a field of algebraic numbers is one of the central tasks of computational algebraic number theory. It is believed to be hard classically, which is of interest for cryptography. In the quantum setting, efficient algorithms were previously known for fields of constant degree. We give a quantum algorithm that is polynomial in the degree of the field and the logarithm of its discriminant. This is achieved by combining three new results. The first is a classical algorithm for computing a basis for certain ideal lattices with doubly exponentially large generators. The second shows that a Gaussian-weighted superposition of lattice points, with an appropriate encoding, can be used to provide a unique representation of a real-valued lattice. The third is an extension of the hidden subgroup problem to continuous groups and a quantum algorithm for solving the HSP over the group $\mathbb{R}^n$.

[*]Partially supported by National Science Foundation grant DMS-1056703 and by the National Security Agency (NSA) under Army Research Office (ARO) contract number W911NF-12-1-0522. Part of this work was done while the first author was visiting Harvard University and MIT.

[†]Partially supported by National Science Foundation awards CCF-0747274 and CCF-1218721, and by the National Security Agency (NSA) under Army Research Office (ARO) contract number W911NF-12-1-0522. Part of this work was done while visiting MIT.

STOC '14, May 31 - June 03 2014, New York, NY, USA
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2710-7/14/05$15.00.
http://dx.doi.org/10.1145/2591796.2591860.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems

General Terms

Algorithms, Theory

Keywords

Quantum Algorithms, Unit Group, Computational Algebraic Number Theory

1. INTRODUCTION

The problems where quantum algorithms have exponential speedups over the best known classical algorithm have mostly been of number theoretic origin. Shor found quantum algorithms for factoring and discrete log [Sho97] and Hallgren found a quantum algorithm for solving Pell's equation [Hal07]. These algorithms were further generalized to finding the unit group of a number field and related problems [Hal05, SV05]. The running time is measured in terms of the discriminant and the degree of the number field. The degree of a number field is its dimension as a vector space over $\mathbb{Q}$, while the discriminant is related to the volume of the fundamental domain of the ring of integers. The algorithms in [Hal05, SV05] are only efficient for constant degree number fields. In this paper we address the arbitrary degree case and give an algorithm that is efficient in both the discriminant and the degree.

A number field K can be defined as a subfield of the complex numbers $\mathbb{C}$ which is generated over the rational numbers $\mathbb{Q}$ by an algebraic number, i.e. $K = \mathbb{Q}(\theta)$, where θ is the root of a polynomial with rational coefficients. If K is a number field, then the subset of K consisting of all elements that are roots of monic polynomials with integer coefficients, forms a ring $\mathcal{O}$, called the ring of integers of K. The ring $\mathcal{O} \subseteq K$ can be thought of as a generalization of $\mathbb{Z} \subset \mathbb{Q}$. In particular, we can ask whether $\mathcal{O}$ is a principal ideal domain, whether elements of $\mathcal{O}$ have unique factorization, and what the set of invertible elements is. The unit group $\mathcal{O}^*$ is the set of invertible algebraic integers inside K, that is, elements $\alpha \in \mathcal{O}$ such that $\alpha^{-1} \in \mathcal{O}$.

Computing the unit group of a number field is an important problem in computational number theory. By Dirichlet's Theorem the group of units $\mathcal{O}^*$ is isomorphic to $\mu(K) \times \mathbb{Z}^{s+t-1}$, where $\mu(K)$ are the roots of unity contained in K and K has s real embeddings and t pairs of complex conjugate embeddings. An elementary version of the problem is Pell's equation: given a positive non-square integer d, find x and y such that $x^2 - dy^2 = 1$. Solutions to this equation are parametrized by the formula $x_k + y_k\sqrt{d} = \bigl(x_1 + y_1\sqrt{d}\bigr)^k$; the numbers $\pm\bigl(x_k + y_k\sqrt{d}\bigr)$ are exactly the units of the quadratic ring $\mathbb{Z}\bigl[\sqrt{d}\bigr]$ (or a subgroup of index 2 if there is a unit that has norm -1). The fundamental solution (x_1, y_1) is difficult to find, or even to write down because it may be exponential in d (i.e., doubly-exponential). Moreover, the computation of the real number $R = \ln\bigl(x_1 + y_1\sqrt{d}\bigr)$ with a polynomial number of precision digits is believed to be a hard problem classically.

A polynomial time quantum algorithm for the computation of R was given in [Hal07]. The approach is to reduce the problem to a hidden subgroup problem (HSP) over the real numbers $\mathbb{R}$, and then to give a quantum algorithm for that hidden subgroup problem. In this context, the HSP amounts to having a periodic function on $\mathbb{R}$ which is 1-1 within the period. The goal is to approximate the period.

For the unit group the corresponding periodic function g takes a real number u to a lattice $g(u) \subset \mathbb{R}^2$. More specifically, we can embed $\mathcal{O}$ as a lattice, and then $g(u)$ is obtained by stretching by a factor of e^u in one direction and squeezing in the other:

$$g(u) = (e^u, e^{-u})\,\mathcal{O} := \bigl\{\bigl(e^u z^{(1)}, e^{-u} z^{(2)}\bigr) : (z^{(1)}, z^{(2)}) \in \mathcal{O}\bigr\},$$
$$\text{where} \quad \mathcal{O} = \bigl\{\bigl(x + y\sqrt{d},\, x - y\sqrt{d}\bigr) : x, y \in \mathbb{Z}\bigr\},$$

for $d \equiv 2, 3 \pmod 4$. The function g is periodic with period R and 1-1 within the period. However, exponential stretching and squeezing of lattices are not computationally trivial. Furthermore, the standard quantum algorithm for the hidden subgroup problem requires a unique representation of the oracle function value (a representation up to an equivalence relation will not work). In [Hal07] these issues were addressed by using an intricate notion of "reduced ideals". This method was extended to constant degree number fields [Hal05, SV05], but it is difficult to generalize this method to rings of higher degree. At a minimum, computing the required reduced ideals seems to require solving the shortest vector problem in ideal lattices of dimension n, and enumerating lattice points also seems necessary. Cryptosystems whose security relies on the hardness of solving problems in ideal lattices have been suggested for cryptography [PR07, LPR10]. Another problem is running the hidden subgroup algorithm for the continuous group $G = \mathbb{R}^m$, where rounding causes errors. Such errors are tolerable when m is fixed, but worsen in higher dimensions.

We propose a different scheme, leading to a quantum reduction from computing the unit group of a number field of arbitrary degree n to solving an Abelian hidden subgroup problem over $\mathbb{R}^m$, where $m = O(n)$. It involves several important ingredients. First, we represent a lattice by a reduced basis (up to some precision). The exponential transformation is performed using repeated squaring of lattices. These lattices can be multiplied because they are also ideals. Having obtained some basis of the lattice $L = f(u)$, we construct a canonical *quantum representation* of L, namely the Gaussian-weighted superposition of lattice points with a sufficiently large dispersion. To ensure stability against rounding errors, each lattice point is represented by a superposition of nearby points in a fine grid. (For example, in one dimension, such a superposition straddles two adjacent grid points.) The initial idea for handling this was using double Gaussian states as in [GKP01], which required a different representation of points. In addition to showing how to classically compute approximate bases for the stretched lattices, we prove that the inner product of Gaussian lattice states has a hidden subgroup property.

One byproduct of this work is a generalization of the HSP to uncountable topological groups such as $\mathbb{R}$. Most exponential speedups by quantum algorithms either use or try to use the HSP [FIM$^+$03, HMR$^+$10]. In the HSP a function $f : G \to S$ is given on a group G to some set S. For an unknown subgroup $H \subseteq G$, the function is constant on cosets of H and distinct on different cosets. The goal is to find a set of generators for H in time polynomial in the appropriate input size, e.g. $\log|G|$. When G is finite Abelian or $\mathbb{Z}^m$ there is an efficient quantum algorithm to solve the problem.

Using the usual definition of the HSP for the group $G = \mathbb{R}$ does not work as can be seen by the following illustration. When the group is discrete the function can be evaluated on any group element. For example, it is possible to verify that a given element h is in H, by testing if $g(0) = g(h)$. Over the reals, if the period is some transcendental number x, then no algorithm could ever even query $g(x)$, and then see that it matches $g(0)$. It is possible to address this by giving an ad-hoc technical definition if we replace $\mathbb{R}$ by a discrete set with rounding, as in the case of constant degree number fields [Hal07, Hal05, SV05]. However, it is not known how to solve the HSP with such a definition. Here we give a cleaner definition using continuous functions which aids us in finding an algorithm to solve the general problem.

Definition 1.1 (The continuous HSP over $\mathbb{R}^m$). *The unknown subgroup $L \subseteq \mathbb{R}^m$ is a full-rank lattice satisfying some promise: the norm of the shortest vector is at least λ and the unit cell volume is at most d. The oracle has parameters (a, r, ε). Let $f : \mathbb{R}^m \to S$ be a function, where S is the set of unit vectors in some Hilbert space. We assume that f hides L in the following way.*

1. *f is periodic on L: for all $v \in L$, $x \in \mathbb{R}^m$, $f(x) = f(x + v)$;*
2. *$\bigl\| |f(x)\rangle - |f(y)\rangle \bigr\| \leqslant a \cdot \operatorname{dist}(x, y)$ for all $x, y \in \mathbb{R}^m$ (Lipschitz);*
3. *If $\min_{v \in L} \|x - y - v\| \geqslant r$, then $\bigl|\langle f(x)|f(y)\rangle\bigr| \leqslant \varepsilon$.*

Given an efficiently computable function with this property, compute a basis for L.

We show that computing the unit group of an arbitrary degree number field can be (quantum) reduced to this definition of the HSP, and we also give a quantum algorithm for solving it. We prove the following main theorem

THEOREM 1.2. *There is an efficient quantum algorithm to compute the unit group of a number field K that is polynomial in the degree of K and polynomial in log of the discriminant of K.*

This follows from Theorem 4.4, Theorem 5.7, Theorem 6.1, and the fact that lattice Gaussians can be computed efficiently given an approximate basis.

Computing the unit group is one of the main computational tasks in algebraic number theory [Coh93]. Two of the others are solving the principal ideal problem and computing the class group. Based on the previous quantum algorithms for solving these three problems in the constant degree case, the unit group seems to be the most difficult part. The other two problems can be solved using the unit group algorithm and general hidden subgroup techniques. We leave the other two problems open for arbitrary degree. The main issue will be proving that the HSP functions constructed to solve them are Lipschitz.

In the context of cryptography, the problem of computing the unit group and solving the principal ideal problem are considered to be hard classically, even over degree two number fields. It was used as a basis in the Buchmann-Williams key exchange problem in an effort to find a system that is harder to break than factoring-based systems. On the other hand, the typical ideal lattice problem, such as finding short vectors over degree two number fields, is easy because the degree is constant.

In the last few years, since the discovery of homomorphic encryption and the ensuing efforts to make the systems more efficient and more secure, assumptions related to number fields have been used. These systems are set up based on high degree number fields. In [GH11], a version of the principal ideal problem where a special generator is the secret was used as the hardness assumption. The Ring-LWE problem which forms the basis in [LPR10, BV11] assumes that finding short vectors in ideal lattices of high degree number fields is hard.

To summarize, the constant degree assumptions are broken by quantum algorithms. The relatively recent high degree number field assumptions about computing short vectors are still open in terms of security against quantum computers. However, in this paper we show that it is now possible to efficiently compute the unit group in these number fields, which could move towards understanding whether the new homomorphic cryptosystems really are secure against quantum computers.

2. NUMBER-THEORETIC BACKGROUND

In the following K will denote a number field of degree n over $\mathbb{Q}$, and $\mathcal{O}$ will denote its ring of integers. When we want to consider $\mathcal{O}$ as a lattice in $E = \mathbb{R}^s \times \mathbb{C}^t$ with $s+2t = n$ (see below), we will write $\underline{\mathcal{O}}$. We use bold letters to designate elements of E and vectors in general.

If $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ is a basis for a lattice $\Lambda \subseteq \mathbb{R}^n$, let B be the matrix $B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ which is composed of column vectors $\boldsymbol{b}_k$. Then $d(\Lambda) := |\det(B)|$ is the unit cell volume of the lattice Λ generated by the basis.

Elements of K can be conveniently represented by using the embeddings of K into the field of complex numbers. In general, there are n such embeddings, which break into s real ones and t complex-conjugate pairs:

$$\tau_1, \ldots, \tau_s : K \to \mathbb{R},$$
$$\tau_{s+1}, \ldots, \tau_{s+t}, \overline{\tau_{s+1}}, \ldots, \overline{\tau_{s+t}} : K \to \mathbb{C} \qquad (s+2t = n).$$

Each element $z \in K$ is mapped to the corresponding *conjugate vector* $\boldsymbol{\tau}(z) = \Big(z^{(1)}, \ldots, z^{(s)}, z^{(s+1)}, \ldots, z^{(s+t)}, \overline{z^{(s+1)}}, \ldots, \overline{z^{(s+t)}}\Big)^T \in \mathbb{R}^s \times \mathbb{C}^{2t}$, where the last t coordinates are redundant. Thus, K is embedded into $E = \mathbb{R}^s \times \mathbb{C}^t$. Conjugate vectors are added and multiplied coordinate-wise. Many useful functions on K extend naturally to E. For example, the algebraic trace and norm are defined for arbitrary conjugate vectors:

$$\operatorname{tr}(\boldsymbol{z}) = \sum_{j=1}^{s} z^{(j)} + \sum_{j=s+1}^{s+t} \big(z^{(j)} + \overline{z^{(j)}}\big),$$

$$\mathcal{N}(\boldsymbol{z}) = \prod_{j=1}^{s} z^{(j)} \prod_{j=s+1}^{s+t} |z^{(j)}|^2.$$

Both these functions take real values.

As far as the additive structure is concerned, the ring E is simply an n-dimensional real space. We can define a Euclidean inner product on E by letting

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \operatorname{tr}(\boldsymbol{x}\overline{\boldsymbol{y}}) = \sum_{j=1}^{s} x^{(j)} y^{(j)} +$$
$$2 \sum_{j=s+1}^{s+t} \Big(\big(\operatorname{Re} x^{(j)}\big)\big(\operatorname{Re} y^{(j)}\big) + \big(\operatorname{Im} x^{(j)}\big)\big(\operatorname{Im} y^{(j)}\big) \Big).$$

The length of a vector with respect to this inner product is denoted by $\|\boldsymbol{z}\|$.

Now let $\{\omega_1, \ldots, \omega_n\}$ be some basis (over $\mathbb{Z}$) for the ring $\mathcal{O}$ of integral elements in K. One way to characterize $\mathcal{O}$ is by its "multiplication table", i.e., the decomposition of $\omega_j\omega_k$ into ω_l with integer coefficients. In the conjugate vector representation, $\mathcal{O}$ becomes a lattice $\underline{\mathcal{O}} \subseteq E$ with basis $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$. From the computational perspective, it is important to have some upper bound on the length of the basis vectors or, equivalently, on the coefficients in the multiplication table. To this end, we use the notion of *discriminant*, which is defined as the determinant of the matrix G with entries $G_{jk} = \operatorname{tr}(\omega_j\omega_k)$. The discriminant $D = D(\mathcal{O})$ depends only on the ring but not the basis. The extension degree, n, and the discriminant, D, constitute a natural set of parameters characterizing the "complexity" of the ring. Our algorithm for finding the group of units is polynomial in $n + \log|D|$.

For various algorithmic tasks, e.g. the computation of the lattice $e^{\boldsymbol{u}}\underline{\mathcal{O}}$, the basis vectors of $\underline{\mathcal{O}}$ must be known with sufficient precision. We will use the fact that the embedding of elements of K can be found to any polynomial number of precision bits in polynomial time [Thi95].

3. OVERVIEW OF THE ALGORITHM

The group of units $\mathcal{O}^*$ consists of elements $z \in \mathcal{O}$ such that $\mathcal{N}(z) = \pm 1$, and they are represented by conjugate vectors of the form $\boldsymbol{z} = e^{\boldsymbol{u}}\boldsymbol{v}$. Here $\boldsymbol{u} = (u^{(1)}, \ldots, u^{(s+t)}) \in \mathbb{R}^{s+t}$ satisfies the condition $\sum_j u^{(j)} = 0$, and the components of $\boldsymbol{v} = (v^{(1)}, \ldots, v^{(s+t)}) \in E = \mathbb{R}^s \times \mathbb{C}^t$ are real or complex numbers of absolute value 1. Thus, the group of units $\mathcal{O}^*$ is contained in

$$G = \mathbb{R}^{s+t-1} \times \big((\mathbb{Z}_2)^s \times (\mathbb{R}/\mathbb{Z})^t\big).$$

More specifically, $\mathcal{O}^*$ is the hidden subgroup in G which corresponds to the following oracle:

$$g : G \to \text{lattices in } E : (\boldsymbol{u}, \boldsymbol{v}) \mapsto e^{\boldsymbol{u}}\boldsymbol{v}\underline{\mathcal{O}}. \quad (3.1)$$

We give an efficient classical realization of this function, where the output (i.e. a lattice $L \subset E$) is represented by some basis with a certain precision. Unfortunately, such a representation is not unique, and therefore g cannot be used as an oracle for a quantum HSP algorithm. To deal with this issue, we compose the function g with another function:

$$\tilde{f} : \text{lattices in } E \to \text{quantum states} : L \mapsto |\tilde{f}(L)\rangle, \quad (3.2)$$

where $|\tilde{f}(L)\rangle$ is a uniquely defined quantum superposition that encodes the lattice L. Thus, we obtain a usable quantum oracle

$$f = \tilde{f} \circ g : G \to \text{quantum states} : (\boldsymbol{u}, \boldsymbol{v}) \mapsto \big|\tilde{f}\big(e^{\boldsymbol{u}}\boldsymbol{v}\underline{\mathcal{O}}\big)\big\rangle.$$

Note: By abuse of notation, we will later denote $\tilde{f}$ as f. For example, we will refer to the quantum state that encodes the lattice L as $|f(L)\rangle$.

Finally, we reduce the HSP problem for G to that for $\mathbb{R}^m$ and apply a general algorithm for finding the hidden subgroup in $\mathbb{R}^m$.

Thus, our algorithm for finding the group of units splits into three self-contained parts:

- A classical algorithm for computing the function $g : (\boldsymbol{u}, \boldsymbol{v}) \mapsto \boldsymbol{v}e^{\boldsymbol{u}}\underline{\mathcal{O}}$. Note that we cannot compute $e^{\boldsymbol{u}}$ because it is an exponentially long number. Instead, we begin with representing $\boldsymbol{u}$ as $2^l\boldsymbol{u}_0$, where $\boldsymbol{u}_0$ is sufficiently small, and compute the lattice $e^{\boldsymbol{u}_0}\underline{\mathcal{O}}$ directly. Then we apply the following procedure l times: given a basis $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$ of the lattice $\Lambda = e^{\boldsymbol{w}}\underline{\mathcal{O}}$ (for some $\boldsymbol{w}$ that does not need to be known), we compute some basis of the lattice $\Lambda^2 = e^{2\boldsymbol{w}}\underline{\mathcal{O}}$. The repeated squaring yields a basis of the lattice $e^{\boldsymbol{u}}\underline{\mathcal{O}}$; then we multiply it by $\boldsymbol{v}$. The ideal squaring procedure is not trivial. We need to compute all products $\boldsymbol{z}_j\boldsymbol{z}_k$ and find some basis of the lattice they generate. This requires the detection of linear dependencies with integer coefficients as well as some way to prevent the vector lengths from growing. The algorithm for computing a basis for $e^{u}\underline{\mathcal{O}}$ is new as far as we know.

- A quantum procedure for the creation of the state $|\tilde{f}(L)\rangle$ representing a lattice $L \subset \mathbb{R}^n$. Let L be given by some basis B. We first make a Gaussian-weighted superposition of coefficient vectors $\boldsymbol{x} \in \mathbb{Z}^n$ that represent the lattice points relative to the basis. Then for each $\boldsymbol{x} \in \mathbb{Z}^n$ we create the corresponding lattice point $\boldsymbol{z} = B\boldsymbol{x} \in L$ as a quantum state. In doing so, we use a straddle encoding $\mathrm{str}_{n,\nu}$ (to be defined in Sect. 5) to account for rounding errors. The original value of $\boldsymbol{x}$ may now be erased (in a reversible way, which requires the reconstruction of $\boldsymbol{x}$ from $\widetilde{\boldsymbol{z}} \approx \boldsymbol{z}$). The resulting state, $|\tilde{f}(L)\rangle = c\sum_{\boldsymbol{z}\in L} e^{-\pi\|\boldsymbol{z}\|^2/s^2}\,|\mathrm{str}_{n,\nu}(\boldsymbol{z})\rangle$, does not depend on the basis (except for small inaccuracies in the preparation of the Gaussian superposition). Assuming a lower bound on the length of a shortest vector, $\lambda_1(L) \geqslant \lambda$, and an upper bound on the unit cell volume, $d(L) \leqslant d$, we find a Lipschitz constant of the function $\tilde{f}$ and estimate the inner product $\langle\tilde{f}(L)|\tilde{f}(L')\rangle$ when the lattices L and L' are far apart.

- An efficient quantum algorithm for finding a hidden subgroup in an elementary Abelian group (as discussed in the introduction). Such groups are quotients of $\mathbb{R}^k \times \mathbb{Z}^l$, therefore it is sufficient to consider this case. The problem is further reduced to the HSP for $G = \mathbb{R}^m$. The algorithm has the usual structure. We create a superposition of points in $\mathbb{R}^m$ with a sufficiently broad wavefunction w, apply the oracle, and measure in the Fourier basis. In the first approximation, the Fourier sampling generates a point u of the reciprocal lattice L^* with the probability distribution
$$q_u = \frac{1}{d(L)^2}\int_{(\mathbb{R}^m/L)^2} \langle f(x')|f(x)\rangle\, e^{2\pi i\langle x-x',u\rangle}\, dx\, dx'.$$
Repeating the procedure sufficiently many times, we obtain a set of vectors that generate L^*. In practice, the Fourier samples deviate from the lattice points by, roughly, the inverse width of the wavefunction w. Then the lattice is reconstructed from an approximate generating set (see Section 4.3).

All of our results are stated for the ring of integers $\mathcal{O}$ of a given number field K, but they can easily be extended to general orders of K.

4. COMPUTING A BASIS FOR $e^t\mathcal{O}$

In this section we show how to compute an approximate basis for the lattice $\boldsymbol{e^t}\underline{\mathcal{O}}$. Because $\boldsymbol{e^t}$ is in general doubly exponential in size and we have to use floating point computations, this is a non-trivial operation. The basic steps are to alternate ideal multiplication with size reduction to compute a short basis for the product of the two ideals that were multiplied. The algebraic numbers that appear in this computation would take exponentially many bits to represent exactly. Instead we show that a polynomial number of bits of precision is sufficient. The idea is to use the fact that we are always using ideal lattices with lower bounds and upper bounds on the vector lengths appearing throughout the computation, so that the precision loss can be bounded at each step. With this we can pick a precision high enough, some polynomial number of bits, so that we still have high precision at the end. The precision we need is that for any vector of length at most $s\sqrt{n}$, the computed vector is within $1/(2N)$ of the actual vector. Here s and N are parameters chosen such that the lattice Gaussian superposition in Section 5 will be a good approximation to the lattice.

Given $\boldsymbol{t} = (t_1, \ldots, t_n) \in \mathbb{R}^n$ such that $\sum t_i = 0$ we will show how to compute a basis of the lattice $f(\boldsymbol{t}) = (e^{t_1}, \ldots, e^{t_n}) \cdot \underline{\mathcal{O}}$.

The function $f : \mathbb{R}^n \to \{\text{real-valued lattices}\}$ is constant and distinct on cosets of the Log embedding of (the free part of) the units. We will later handle the fact that we only have approximations of these lattices, in particular, how to create useful superpositions using the approximations.

The main subroutine needed for computing f computes a basis of the product of two lattices. Lattices A and B can be multiplied in this case since they are always of the form $(a_1, \ldots, a_n) \cdot \underline{\mathcal{O}}$, where $a_i \in \mathbb{R}$, and $\mathcal{O}^2 = \mathcal{O}$. In particular, given the bases of two lattices $A = \langle \boldsymbol{w}_1, \ldots, \boldsymbol{w}_n \rangle$ and $B = \langle \boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \rangle$, all pairwise products $\boldsymbol{w}_i \boldsymbol{v}_j$ of basis vectors are computed giving a generating set of the lattice AB, to which we apply Theorem 4.3 to obtain a size-reduced basis.

To ensure that the entire computation can be done in polynomial time we must give an upper bound for the determinant of each lattice and a lower bound for the shortest vector, which bounds the basis sizes.

4.1 Splitting up the computation

The computation must be split up carefully to avoid ending up with doubly exponential size coefficients. First it is split into two parts. The first part handles the integer part of the t_i's which is complicated by the fact that e^{t_i} will be doubly-exponential in general. The solution will be to compute a sequence of ideals $A_{-1}, A_0, \ldots, A_m$ of bounded determinant such that $A_{-1} \times \prod_{i=0}^{m} A_i^{2^i} = f(\boldsymbol{t})$.

For $1 \leq i \leq n-1$ let $t_i = r_i + s_i$, where $r_i \in \mathbb{Z}$ and $0 \leq s_i < 1$. Let $r_n = -\sum_{i=1}^{n-1} r_i$ and $s_n = t_n - r_n$. Using the fact that $(e^{t_1}, \ldots, e^{t_n}) \cdot \underline{\mathcal{O}} = (e^{r_1}, \ldots, e^{r_n}) \cdot \underline{\mathcal{O}} \cdot (e^{s_1}, \ldots, e^{s_n}) \cdot \underline{\mathcal{O}}$ we will compute these two pieces separately.

Define $A_j = (e^{r_{1j}}, \ldots, e^{r_{(n-1)j}}, (e^{-1})^{\sum_{i=0}^{n-1} r_{ij}}) \cdot \underline{\mathcal{O}}$, where r_{ij} is $\operatorname{sign}(r_i)$ times bit j of $|r_i|$. From the determinant formula it follows that the determinant of A_j is the determinant of $\mathcal{O}$ times $(e^{-1})^{\sum_i r_{ij}} \prod_{i=0}^{n-1} e^{r_{ij}} = e^{\sum_i r_{ij} - \sum_i r_{ij}} = 1$. This also bounds the powers of A_j. The log of the determinant of $\mathcal{O}$ and n define the input size to the problem.

The second part handles the fractional part of the t_i's by directly computing the ideal $A_{-1} = (e^{s_1}, \ldots, e^{s_n}) \cdot \underline{\mathcal{O}}$ using the first polynomially many terms in the formula $e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!}$ to get the desired approximation. From the determinant formula it follows that $\det A_{-1}$ is $\prod_i e^{s_i}$ times the determinant of $\mathcal{O}$. The product $\prod_i e^{s_i}$ is between e^{-2n} and e^{2n}.

To see that $f(\boldsymbol{t}) = A_{-1} \cdot \prod_j A_j^{2^j}$, we compute

$$\begin{aligned}
&A_{-1} \cdot \prod_j A_j^{2^j} \\
&= A_{-1} \cdot \prod_j \left((e^{r_{1j}}, \ldots, e^{r_{(n-1)j}}, (e^{-1})^{\sum_i r_{ij}}) \cdot \underline{\mathcal{O}} \right)^{2^j} \\
&= A_{-1} \cdot (e^{\sum_j r_{1j} 2^j}, \ldots, e^{\sum_j r_{(n-1)j} 2^j}, (e^{-1})^{\sum_{j,i} r_{ij} 2^j}) \cdot \underline{\mathcal{O}} \\
&= (e^{s_1}, \ldots, e^{s_n}) \cdot (e^{r_1}, \ldots, e^{r_{n-1}}, (e^{-1})^{\sum_i r_i}) \cdot \underline{\mathcal{O}} \\
&= (e^{t_1}, \ldots, e^{t_{n-1}}, (e^{-1})^{\sum_i t_i}) \cdot \underline{\mathcal{O}} = f(\boldsymbol{t})
\end{aligned}$$

The algorithm now works as follows. First compute a $\mathbb{Z}$-basis $\omega_1, \ldots, \omega_n$ of $\mathcal{O}$. Next compute the conjugate vector representation $\boldsymbol{z}_i = \underline{\omega_i}$. Compute A_{-1} as described above. Next compute each A_j by first computing $(e^{r_{1j}}, \ldots, e^{r_{nj}})$ and then $(e^{r_{1j}}, \ldots, e^{r_{nj}}) \cdot \boldsymbol{z}_i$ for each i.

Next use repeated squaring of ideals to compute a basis for $A_j = ((e^{t_{1j}}, \ldots, e^{t_{nj}}) \cdot \underline{\mathcal{O}})^{2^j}$. Finally, multiply the $A_j^{2^j}$'s and A_{-1}.

4.2 Computations with approximations

Now we introduce some notation and prove some facts that will allow us to analyze the algorithm for computing an approximate basis of the ideal lattice $e^{\boldsymbol{t}}\mathcal{O}$, which is given in Section 4.4. A real number x is typically approximated by a rational number $\tilde{x} = \frac{p}{2^m}$ such that $|\tilde{x} - x| \leqslant \varepsilon$. The parameter ε is called the *absolute error* and the ratio $\gamma = \varepsilon/|x|$ is called the *relative error*. A vector $\boldsymbol{x} = (x^{(1)}, \ldots, x^{(n)})$ is approximated by another $\widetilde{\boldsymbol{x}} = (\tilde{x}^{(1)}, \ldots, \tilde{x}^{(n)})$ such that $\|\widetilde{\boldsymbol{x}} - \boldsymbol{x}\| \leqslant \varepsilon$; the relative error is defined as $\gamma = \varepsilon/\|\boldsymbol{x}\|$.

We start with a lemma bounding the error in the ideal multiplication step.

LEMMA 4.1. *Suppose a basis $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ for a lattice $\boldsymbol{e^{t_1}}\mathcal{O}$ and a basis $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n$ for $\boldsymbol{e^{t_2}}\mathcal{O}$ are given with relative precision γ, and each $\boldsymbol{e^{t_i}}$ has norm 1, then the resulting products $\boldsymbol{b}_i \boldsymbol{c}_j$ have relative precision $4\gamma\sqrt{n}||\boldsymbol{b}_i||^n$.*

In the next section we show how to compute a basis from a generating set.

4.3 Computing a basis of a lattice from an approximate generating set

To compute in polynomial time a basis $\hat{\boldsymbol{b}}_1, \ldots, \hat{\boldsymbol{b}}_n$ that is bounded in size and δ-close to a basis for $\boldsymbol{e^t}\mathcal{O}$ we need to compute a bounded-length basis from a generating set. This is also used in the hidden subgroup problem algorithm when computing the basis for the unit lattice from a generating set for its dual. The input and output vectors for this are approximate. We need an algorithm that can find integer dependencies among the rounded vectors and also to bound the errors that result from the transformation to the reduced basis. An algorithm for computing a lattice basis from a set of generators was given by Buchmann and Pohst [BP87] and Buchmann and Kessler [BK93]. They do not bound all of the errors, though. In order to bound the errors that result from the transformation to the reduced basis we need better bounds on the coefficient sizes in the transformation than what they give. For that reason we will present their algorithm and improve their analysis. Both [BP87] and [BK93] analyze the same algorithm for computing a basis, so we give an outline of their algorithm and bounds before giving our analysis and proving the error bounds.

The setup is as follows. Suppose that vectors $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k \in \mathbb{R}^n$ generate an r-dimensional lattice L. We are given approximations $\hat{\boldsymbol{a}}_1, \ldots, \hat{\boldsymbol{a}}_k \in \mathbb{Z}^n$ such that $||\boldsymbol{a}_i - \hat{\boldsymbol{a}}_i/2^q|| \leq \sqrt{n}/2^{q+1}$. We want to compute a set of vectors $\hat{\boldsymbol{c}}_1, \ldots, \hat{\boldsymbol{c}}_r$ which approximate a basis $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_r$ of L with some precision q', which will necessarily be smaller than q. The algorithms in [BP87] and [BK93] compute such a basis via a reduction to LLL. We will use their algorithm to solve this problem, but we need a better analysis to make sure that the output accuracy q' is not too much smaller than q. This is done in Theorem 4.3 below.

4.3.1 The Buchmann-Pohst algorithm

The reduction to LLL takes the input vectors $\hat{\boldsymbol{a}}_1, \ldots, \hat{\boldsymbol{a}}_k$ and creates a new lattice with basis defined from the concatenated vectors

$$\tilde{\boldsymbol{a}}_j = (\boldsymbol{e}_j, \hat{\boldsymbol{a}}_j) \quad (1 \leq j \leq k),$$

where $\boldsymbol{e}_j$ denotes the j-th unit vector in $\mathbb{Z}^k$. These vectors $\tilde{\boldsymbol{a}}_1, \ldots \tilde{\boldsymbol{a}}_k \in \mathbb{Z}^{n+k}$ are clearly linearly independent and form a basis of the lattice $\tilde{L} = \bigoplus_{j=1}^{k} \mathbb{Z}\tilde{\boldsymbol{a}}_j$. Note that with this setup, the bottom of the matrix has vectors $\lfloor 2^q \boldsymbol{a}_i \rceil$ as the basis vectors, where $\lfloor \cdot \rceil$ rounds each coordinate of the vector. The LLL-algorithm is then applied to the lattice basis

$\tilde{\boldsymbol{a}}_1, \ldots \tilde{\boldsymbol{a}}_k$ to obtain an LLL-reduced basis $\tilde{\boldsymbol{b}}_1, \ldots, \tilde{\boldsymbol{b}}_k$. For the first $k-r$ of these vectors we will denote the top and bottom components as

$$\tilde{\boldsymbol{b}}_j = (\boldsymbol{m}_j, \hat{\boldsymbol{z}}_j)^T \quad (1 \le j \le k-r),$$

and for the last r vectors as

$$\tilde{\boldsymbol{b}}_{k-r+j} = (\boldsymbol{m}'_j, \hat{\boldsymbol{b}}_j)^T \quad (1 \le j \le r).$$

Note that the vectors $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_{k-r} \in \mathbb{Z}^k$ are the coefficient vectors transforming the vectors $\hat{\boldsymbol{a}}_1, \ldots, \hat{\boldsymbol{a}}_k$ to the vectors $\hat{\boldsymbol{z}}_1, \ldots, \hat{\boldsymbol{z}}_{k-r}$, respectively, and the vector $\boldsymbol{m}'_j$ takes $\hat{\boldsymbol{a}}_1, \ldots, \hat{\boldsymbol{a}}_k$ to $\hat{\boldsymbol{b}}_j$ (for $1 \le j \le r$).

In [BP87] and [BK93] it is shown that the resulting basis has the following two properties. First, the top left $k-r$ columns contain a linearly independent set of relations $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_{k-r} \in \mathbb{Z}^k$ for the exact vectors $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k$. A relation vector $\boldsymbol{m}_j = (m_{1j}, \ldots, m_{kj})^t$ satisfies $\sum_{i=1}^k m_{ij}\boldsymbol{a}_i = 0$. In the exact matrix the vectors $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{k-r}$ would be zero, so the approximate vectors $\hat{\boldsymbol{z}}_i$ will be small. The second property of the resulting basis is that the bottom right of the matrix contains approximations $\hat{\boldsymbol{b}}_1, \ldots, \hat{\boldsymbol{b}}_r$ to a basis for the lattice generated by $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k$.

In [BK93] it is shown that given good enough approximations of $\boldsymbol{a}_,, \ldots, \boldsymbol{a}_k$, the vectors

$$\boldsymbol{b}_j = \sum_{i=1}^{k} m'_{i,j}\boldsymbol{a}_i \ (1 \le j \le r) \tag{4.1}$$

form a basis for L and satisfy

$$||\boldsymbol{b}_j|| \le (\sqrt{kn}+2)2^{\frac{k-1}{2}}\lambda_j(L_r) \quad (1 \le j \le r). \tag{4.2}$$

This holds for every sublattice L_r which is spanned by a subset of r linearly independent vectors of $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k$.

Lemma 4.2. *Choose q as is [BK93, Theorem 4.1]. The square length of the coefficient vectors $\boldsymbol{m}'_j$ $(1 \le j \le r)$ that transform the generators $\boldsymbol{a}_i$ of L into a basis vector $\boldsymbol{b}_j$ of L as in Equation 4.1 is bounded by $(\frac{\alpha^{r-1}}{d(L)} \cdot \sqrt{r}\Delta_{1,j})^2 + \Delta_2^2$. Here $\Delta_{1,j}$ is the right-hand-side of Equation 4.2,*

$$\Delta_{1,j} = (\sqrt{kn}+2) \cdot 2^{\frac{k-1}{2}} \cdot \lambda_j(L_r),$$

and L_r is any sublattice of L which is spanned by a subset of r linearly independent vectors of $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k$. The quantity Δ_2 is

$$\Delta_2 = \sqrt{k-r} \cdot 2^{\frac{k-1}{2}} \left(\frac{k\sqrt{n}}{2} + \sqrt{k}\right) \cdot \frac{\alpha^r}{d(L)}.$$

Lemma 4.2 can be used to prove the following theorem which shows that a basis of bounded length for L exists, for which we can efficiently compute a good approximation.

Theorem 4.3. *Let $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k \in \mathbb{R}^n$ generate a lattice L of rank r, let μ be a lower bound on the shortest vector, let $\alpha = \max ||a_i||$, and let q be such that $2^q \ge (k2^{(k+1)/2} \cdot \max ||a_i||)^r/(\mu \det(L)^2)$.*

Given approximations of $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k \in \mathbb{R}^n$ with q bits of precision, a basis $\hat{\boldsymbol{c}}_1, \ldots, \hat{\boldsymbol{c}}_r$ for L can be computed in time polynomial in q, where the exact vectors $\boldsymbol{c}_j$ satisfy

$$||\boldsymbol{c}_j|| \le (\sqrt{kn}+2)2^{\frac{k-1}{2}} \cdot \lambda_j(L).$$

The absolute error on each output vector $\hat{\boldsymbol{c}}_i$ is bounded by $rk\gamma_1\gamma_3\sqrt{n}/2^{q+1}$.

Here $\gamma_1 = \sqrt{\left(\frac{\alpha^{r-1}}{\det(L)}\sqrt{r}\Delta_{1,n}\right)^2 + \Delta_2^2}$,

and

$$\gamma_3 = \sqrt{\left(\frac{(\alpha')^r}{d(L)} \cdot \sqrt{r} \cdot ||\boldsymbol{b}_j||\right)^2 + \left(\sqrt{k-r}\sqrt{k} \cdot \frac{(\alpha')^r}{d(L)}\right)^2},$$

with $\alpha' = (\sqrt{kn}+2)2^{\frac{k-1}{2}}\alpha$.

4.4 The algorithm for computing $e^{\boldsymbol{t}}\mathcal{O}$

Given t, compute a basis for $e^{\boldsymbol{t}}\mathcal{O}$:

1. Choose a polynomial q.
2. For each bit index j do the following:
3. Compute the diagonal matrix T_j, where $(T_j)_{i,i} = e^{r_{i,j}}$ for $i < n$, and $(T_j)_{n,n} = (e^{-1})^{\sum_{i=1}^{n-1} r_{ij}}$.
4. Compute $A_j := T_j \cdot \underline{\mathcal{O}}$, and compute a short basis for it using Theorem 4.3.
5. Square A_j j times, using j applications of ideal multiplication below.
6. Multiply the resulting ideals together. To multiply two ideals B and C proceed as follows: Let the ideal B have basis $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ and ideal C have basis $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n$.
   (a) Multiply pairwise columns to get n^2 vectors $\boldsymbol{c}_1\boldsymbol{b}_1, \boldsymbol{c}_1\boldsymbol{b}_2, \ldots, \boldsymbol{c}_1\boldsymbol{b}_n, \boldsymbol{c}_2\boldsymbol{b}_1, \ldots \boldsymbol{c}_n\boldsymbol{b}_n$.
   (b) Use Theorem 4.3 to compute a short basis for BC.
   (c) Truncate the precision to q bits.

Theorem 4.4. *There is an algorithm that on input $\boldsymbol{t} \in \mathbb{Q}^n$ and δ computes a basis $\hat{\boldsymbol{b}}_1, \ldots, \hat{\boldsymbol{b}}_n$ that is δ-close to a basis for $e^{\boldsymbol{t}}\mathcal{O}$, has bounded size, and runs in time polynomial in $\log\Delta$, n, $\log ||\boldsymbol{t}||$ and $\log 1/\delta$.*

Proof. We analyze the complexity of the algorithm given above. By Thiel [Thi95] we can take the initial precision as high as we need in Step 3 and Step 4. The main step in the algorithm consists of multiplying ideals B and C, so it is enough to show that each multiplication step BC can be done efficiently, and that we can bound the loss of precision. By Lemma 4.1 we can compute n^2 generators for BC from n generators for B and n from C and bound the loss of precision. By Theorem 4.3, we can compute a basis approximating $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ for BC in polynomial time, with

$$||\boldsymbol{b}_j|| \le (\sqrt{n^3}+2)2^{\frac{n^2-1}{2}} \cdot \lambda_j(BC).$$

The loss of precision for the squaring step is bounded in Theorem 4.3. Therefore we may choose the initial number of precision bits q to be high enough to satisfy δ at the end. □

5. THE QUANTUM REPRESENTATION OF LATTICES

The representation of a lattice by a basis is not unique, which makes it unsuitable for use in an algorithm that deals with quantum superpositions of lattices. To avoid this issue, we will represent each lattice by a unique quantum state, namely, a superposition of the lattice points with certain weights. Let us first discuss some desired properties of such a representation. We want a small deformation of the lattice to result in a small change in the quantum state, whereas substantially different lattices should be mapped to almost orthogonal vectors. These requirements can be formalized as follows.

Definition 5.1. *Let* $\operatorname{dist}(x,y)$ *denote the distance between two points in a metric space* X*, and let* $\mathcal{H}$ *be some Hilbert space. A map* $f : X \to \mathcal{H}$ *is called an* (a, r, ε) quantum encoding *if the following conditions are met:*

1. $\langle f(x)|f(x)\rangle = 1$ *for all* $x \in X$*;*
2. $\big\||f(x)\rangle - |f(y)\rangle\big\| \leqslant a \cdot \operatorname{dist}(x, y)$ *for all* $x, y \in X$*;*
3. *If* $\operatorname{dist}(x, y) \geqslant r$*, then* $\big|\langle f(x)|f(y)\rangle\big| \leqslant \varepsilon$.

Given such an encoding, the vector $|f(x)\rangle$ *is called the* signature state *for* x.

The number a in condition 2 is called a *Lipschitz constant* of the function f. When $X = \mathbb{R}^n$ (or, more generally, when X is a Riemannian manifold), f can be approximated by a smooth function to an arbitrary precision in the sup-norm at cost of an arbitrary small parameter change in the above definition. For smooth functions, a is simply an upper bound on the first derivative of f.

Lemma 5.2. *Let* $f_1 : X_1 \to \mathcal{H}_1$ *and* $f_2 : X_2 \to \mathcal{H}_2$ *take values in unit vectors and satisfy the Lipschitz condition with constants* a_1 *and* a_2*, respectively.*

a) *If* $X_1 = X_2 = X$*, then the function* $f_1 \otimes f_2 : X \to \mathcal{H}_1 \otimes \mathcal{H}_2$ *is* $(a_1 + a_2)$*-Lipschitz.*

b) *If* X_1 *and* X_2 *are Euclidean spaces, then the function* $g : X_1 \times X_2 \to \mathcal{H}_1 \otimes \mathcal{H}_2$ *defined as* $g(x_1, x_2) = f_1(x_1) \otimes f_2(x_2)$ *is* a*-Lipschitz, where* $a = \sqrt{a_1^2 + a_2^2}$.

Example 5.3 (Straddle encoding). *A representation of real numbers by quantum superposition of integers can be defined as follows:*

$$\big|\mathrm{str}_\nu(x)\big\rangle = \cos\left(\tfrac{\pi}{2}t\right)|k\rangle + \sin\left(\tfrac{\pi}{2}t\right)|k+1\rangle,$$

where $k = \lfloor x/\nu \rfloor, \quad t = x/\nu - k.$

The map $\mathrm{str}_\nu : \mathbb{R} \to \mathbb{C}^{\mathbb{Z}}$ *is a* $\left(\frac{\pi}{2\nu}, 2\nu, 0\right)$ *quantum encoding. Applying this map to each coordinate on an* n*-dimensional real vector, we obtain a* $\left(\frac{\pi}{2\nu}\sqrt{n}, 2\nu\sqrt{n}, 0\right)$ *encoding* $\mathrm{str}_{\nu,n} : \mathbb{R}^n \to \left(\mathbb{C}^{\mathbb{Z}}\right)^{\otimes n}$ *(by statement (b) of Lemma 5.2).*

We usually set $\nu = 2^{-q}$. The transformation $|x\rangle \mapsto |x\rangle \otimes |\mathrm{str}_\nu(x)\rangle$ can be implemented efficiently if we assume that x is represented as $2^{-l}\tilde{x}$, where $l \geqslant q$ and $\tilde{x}$ is an integer, which is actually stored in the quantum memory. (In practice, l should be substantially greater than q so that the rounding error in x does not matter.) To construct $|x\rangle \otimes |\mathrm{str}_\nu(x)\rangle$ from $|x\rangle$, we compute k and t, create the state $\cos\left(\frac{\pi}{2}t\right)|0\rangle + \sin\left(\frac{\pi}{2}t\right)|1\rangle$, add k, and reverse the computation of k and t.

A full-rank lattice $L \subseteq \mathbb{R}^n$ may be represented by a superposition of its points with Gaussian amplitudes. Each lattice point is in turn represented using the straddle encoding. Thus,

$$\boxed{|f(L)\rangle = \gamma^{-1/2} \sum_{x \in L} e^{-\pi\|x\|^2/s^2} \big|\mathrm{str}_{n,\nu}(x)\big\rangle.} \tag{5.1}$$

Here $\gamma = \sum_{x \in L} e^{-2\pi\|x\|^2/s^2}$.

We must prove the HSP properties for this state when the lattice L is of the form $e^t\mathcal{O}$. To prove that the states over different ideal lattices have small inner product when $\boldsymbol{t}_1 - \boldsymbol{t}_2$ is not near a lattice point we need the following lemma showing that the two corresponding ideal lattices do not have too many points close to each other.

Lemma 5.4. *Let* e^{t_1} *and* e^{t_2} *be vectors of algebraic norm 1. If some nonzero point of* $e^{t_1}\mathcal{O}$ *is equal to some point of* $e^{t_2}\mathcal{O}$ *and has length at most* R*, then the distance between any unequal pair of points, one from* $e^{t_1}\mathcal{O}$ *and one from* $e^{t_2}\mathcal{O}$*, is at least* $\sqrt{n}/R^n$.

Proof. Suppose $\boldsymbol{e}^{\boldsymbol{t_1}}\boldsymbol{a} = \boldsymbol{e}^{\boldsymbol{t_2}}\boldsymbol{b}$ ($\boldsymbol{a}$, $\boldsymbol{b} \in \underline{\mathcal{O}}$) is a point in the intersection of $\boldsymbol{e}^{\boldsymbol{t_1}}\mathcal{O}$ and $\boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O}$ and has length at most R. Since the length of $\boldsymbol{e}^{\boldsymbol{t_1}}\boldsymbol{a}$ is at most R, each coordinate of $\boldsymbol{e}^{\boldsymbol{t_1}}\boldsymbol{a}$ is bounded by R, and hence the norm of $\boldsymbol{e}^{\boldsymbol{t_1}}\boldsymbol{a}$, which is the product over all coordinates, is bounded by R^n. Since $\boldsymbol{e}^{\boldsymbol{t_1}}$ has norm 1, the norm of $\boldsymbol{a}$ is then bounded by R^n as well. To see how close points in $\boldsymbol{e}^{\boldsymbol{t_1}}\mathcal{O}$ and $\boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O}$ can be (without being equal) we consider the minimum distance of points in the lattice $\boldsymbol{e}^{\boldsymbol{t_1}}\mathcal{O} + \boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O}$.

To compute this lattice we first compute $\boldsymbol{e}^{\boldsymbol{t_1}-\boldsymbol{t_2}}\mathcal{O} + \mathcal{O}$: Since $\boldsymbol{e}^{\boldsymbol{t_1}}\boldsymbol{a} = \boldsymbol{e}^{\boldsymbol{t_2}}\boldsymbol{b}$ we have $\boldsymbol{b}/\boldsymbol{a} = \boldsymbol{e}^{\boldsymbol{t_1}-\boldsymbol{t_2}}$. So $\boldsymbol{e}^{\boldsymbol{t_1}-\boldsymbol{t_2}}\mathcal{O} + \mathcal{O} = (\boldsymbol{b}/\boldsymbol{a})\mathcal{O} + \mathcal{O}$. Let $N(\boldsymbol{a})$ denote the norm of $\boldsymbol{a}$. (I.e., $N(\boldsymbol{a}) = N(a_1) = \prod a_i$, where $\boldsymbol{a} = (a_1, \ldots, a_n)$.)

Claim: $(\boldsymbol{b}/\boldsymbol{a})\mathcal{O} + \mathcal{O} \subseteq \frac{1}{N(\boldsymbol{a})}\mathcal{O}$.

Proof of claim: Clearly $\mathcal{O} \subseteq \frac{1}{N(\boldsymbol{a})}\mathcal{O}$. Let $\boldsymbol{a} = (a_1, \ldots, a_n) \in \underline{\mathcal{O}}$. Since each a_i satisfies the same minimal polynomial as a_1 they are all algebraic integers, and hence so is the product $\prod_{i=2}^n a_i$. Since $a_2 \cdot a_3 \cdot \cdots \cdot a_n = N(a)/a_1$, the product is also in K and hence it is in $\mathcal{O}$. Then $b_1 \cdot a_2 \cdot \cdots \cdot a_n$ is in $\mathcal{O}$ as well. Thus

$$(b_1/a_1)\mathcal{O} = \frac{b_1 \cdot a_2 \cdot \cdots \cdot a_n}{N(a_1)}\mathcal{O} \subseteq \frac{1}{N(a_1)}\mathcal{O}.$$

Hence $(\boldsymbol{b}/\boldsymbol{a})\underline{\mathcal{O}} + \underline{\mathcal{O}} \subseteq \frac{1}{N(\boldsymbol{a})}\underline{\mathcal{O}}$. This proves the claim and shows that $\boldsymbol{e}^{\boldsymbol{t_1}-\boldsymbol{t_2}}\mathcal{O} + \mathcal{O} \subseteq \frac{1}{N(\boldsymbol{a})}\mathcal{O}$.

The shortest vector in $\mathcal{O}$ is $(1, \ldots, 1)$ which has length $\sqrt{n}$. Since we showed that $N(\boldsymbol{a}) \leq R^n$, this implies that the shortest vector in $\frac{1}{N(\boldsymbol{a})}\mathcal{O}$ has length at least $\sqrt{n}/R^n$.

Now we consider $\boldsymbol{e}^{\boldsymbol{t_1}}\mathcal{O} + \boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O}$. By the above argument

$$\boldsymbol{e}^{\boldsymbol{t_1}}\mathcal{O} + \boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O} = \boldsymbol{e}^{\boldsymbol{t_2}}(\boldsymbol{e}^{\boldsymbol{t_1}-\boldsymbol{t_2}}\mathcal{O} + \mathcal{O}) \subseteq \frac{1}{N(\boldsymbol{a})}\boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O}.$$

Since the shortest vector in $\boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O}$ has length at least $\sqrt{n}$, the shortest vector in $\boldsymbol{e}^{\boldsymbol{t_1}}\mathcal{O} + \boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O}$ has length at least $\sqrt{n}/R^n$. Hence points in $\boldsymbol{e}^{\boldsymbol{t_1}}\mathcal{O}$ and $\boldsymbol{e}^{\boldsymbol{t_2}}\mathcal{O}$, which are not equal, are at least $\sqrt{n}/R^n$ apart. $\square$

The following lemma helps bound the overlap between two lattices.

Lemma 5.5. *Suppose we are given lattices* L *and* L'*, and sublattices* $I \subseteq L$ *and* $I' \subseteq L'$*. Assume there is a 1-1 correspondence* $h : I \to I'$ *s.t., for any* $(x, x') \in \tilde{L} \times \tilde{L}'$*,* *1. if $(x,x') \in C := \{(u,v) \in I \times I' : v = h(u)\}$, then $\|x - x'\| \le \varepsilon$,*

2. otherwise $\|x - x'\| \ge 2\nu\sqrt{n}$.

If $I \subsetneq L$ and $I' \subsetneq L'$, then for any $n \ge 5$, $\langle f(L)|f(L')\rangle \le 3/4$ if in addition $s \ge 4\pi n^3 \max\{\lambda_n(L), \lambda_n(L')\}$.

Lemma 5.6. *Let $\nu \leqslant \frac{\lambda}{2\sqrt{n}}$ and $s \geqslant c\,n\left(\sqrt{n}/\lambda\right)^{n-1} d$ for a certain constant c, and let us restrict the encoding $L \mapsto |f(L)\rangle$ to lattices with $d(L) \leqslant d$ and $\lambda_1(L) \geqslant \lambda$. On such lattices, f has a Lipschitz constant*

$$a = \frac{\sqrt{\pi n}\, s}{4\nu} + 1.$$

Next we can choose the parameters to show that the hidden subgroup property holds. Since the inner product is at most a constant < 1, a tensor product of n copies will reduce the inner product to be exponentially small. For simplicity, we state the theorem only for the free part of the unit group Log $\mathcal{O}^* \le \mathbb{R}^{s+t-1}$, and for elements of norm 1.

Theorem 5.7. *Let $s = 2^{2n}\sqrt{nD}$, $\nu = 1/(4n(s\sqrt{n})^{2n})$. Let $\boldsymbol{t}_1, \boldsymbol{t}_2 \in \mathbb{R}^n$. Let $\gamma = (\gamma_1, \ldots, \gamma_n)$ be $\boldsymbol{t}_2 - \boldsymbol{t}_1 - \boldsymbol{u}$, where $\boldsymbol{u}$ is the unit closest to $\boldsymbol{t}_2 - \boldsymbol{t}_1$ in Log $\mathcal{O}^*$. Then the function f is a-Lipschitz with constant $a = \frac{\sqrt{\pi n}\, s}{4\nu} + 1$.*

The inner product $\langle e^{\boldsymbol{t_1}}\mathcal{O}|e^{\boldsymbol{t_2}}\mathcal{O}\rangle$ is at most 3/4 if for some i, we either have $\ln\left(1 - (s\sqrt{n})^{n-1} 2\nu\sqrt{n}\right) \geq \gamma_i$ or $\gamma_i \geq \ln\left(1 + (s\sqrt{n})^{n-1} 2\nu\sqrt{n}\right)$.

The idea is as follows: fix an ideal $e^{\boldsymbol{t_1}}\mathcal{O}$. We want to bound its inner product with ideals $e^{\boldsymbol{t_2}}\mathcal{O}$. First we show that when the two lattices have points inside the ball of radius $s\sqrt{n}$ where they overlap exactly, then their inner product must be small (unless the two lattices coincide, i.e. when $\boldsymbol{t_2} - \boldsymbol{t_1}$ is a unit). Then we show that this inner product does not get bigger when we perturb one of the lattices slightly. Finally, we show that when two lattices are not close to having an exact point of intersection, then their inner product is small as well.

Finally, it can be shown that a good approximation of these states can be computed when evaluated on rational numbers. First an approximate basis is computed using the algorithm in the last section, and then the superposition is created over the points.

6. THE HIDDEN SUBGROUP PROBLEM ON A CONTINUOUS GROUP

The HSP algorithm for $\mathbb{R}^m$ can be thought of in the usual structure, however the analysis is difficult, and we combine phase estimation and our new continuous definition of the HSP to get it to work. The analysis works in continuous space and is discretized in a general way to derive the algorithm. The algorithm creates a superposition of points in $\mathbb{R}^m$ with a sufficiently broad wavefunction w. We cannot measure in the Fourier basis of the continuous group, but we show that phase estimation can be used to approximate this to measure a point u of the reciprocal lattice L^* with the probability distribution

$$q_u = \frac{1}{d(L)^2} \int_{(\mathbb{R}^m/L)^2} \langle f(x')|f(x)\rangle\, e^{2\pi i \langle x - x', u\rangle}\, dx\, dx'. \quad (6.1)$$

This distribution is not close to uniform but we are able to show that the probability of staying in any sublattice is bounded. For rounding, the samples deviate from the lattice points by, roughly, the inverse width of the wavefunction w. We show that the reconstruction of a lattice from an approximate generating set can be done using an improved analysis of [BK93] in Section 4.3. It can be shown that the condition number of a reduced basis is bounded so that the dual lattice, which is the hidden subgroup, can be computed. These provide the main ingredients in the full proof.

Theorem 6.1. *There is a polynomial time quantum algorithm for solving the HSP over $\mathbb{R}^m$.*

In Section 6.2 we outline the steps of the algorithm and indicate how to derive the probability expression. First we show that the known cases of the Abelian HSP can be reduced to the new continuous case in Definition 1.1 over the HSP instance $\mathbb{R}^m$.

6.1 Application: Reduction to $G = \mathbb{R}^m$

Our HSP algorithm is applicable to Abelian groups of the form $\mathbb{R}^k \times \mathbb{Z}^l \times (\mathbb{R}/\mathbb{Z})^s \times H$, where H is finite. We call such groups "elementary". The reduction to $G = \mathbb{R}^k \times \mathbb{Z}^l$ is straightforward. In the case of interest, the hidden subgroup L is a full-rank lattice in $G \subseteq \mathbb{R}^k \times \mathbb{R}^l$ such that $\lambda_1(L \cap \mathbb{R}^k) \geqslant \lambda$ and $d(L) \leqslant d$ for some fixed numbers λ and d. We now describe the further reduction to the group $\tilde{G} = \mathbb{R}^{k+l}$.

The main idea can be illustrated in the one-dimensional case, where the parameter λ has no meaning. We embed $G = \mathbb{Z}$ into $\tilde{G} = \mathbb{R}$ in the standard way, set $\nu = 2^{-q}$ for some $q \geqslant 2$, and define the $\mathbb{R}$-oracle g in terms of the $\mathbb{Z}$-oracle f as follows:

$$\begin{aligned}\big|g(x)\big\rangle = {}& c_0\, \big|\mathrm{str}_\nu(t)\big\rangle \otimes \big|f(s)\big\rangle + \\ & c_1\, \big|\mathrm{str}_\nu(t-1)\big\rangle \otimes \big|f(s+1)\big\rangle, \end{aligned} \quad (6.2)$$

where $s = \lfloor x \rfloor$, $t = x - s$, $c_0 = \cos\left(\frac{\pi}{2}t\right)$, $c_1 = \sin\left(\frac{\pi}{2}t\right)$.

It is clear that g is a continuous function. If f is a periodic function, then g is also periodic with the same period.

To construct the state $|g(x)\rangle$ using the original oracle f, we compute s and t, use them as parameters in the following sequence of operations, and "uncompute" s and t:

$$\begin{aligned} |0\rangle \mapsto \sum_z c_z |z\rangle &\mapsto \sum_z c_z\, |z\rangle \otimes \big|f(s+z)\big\rangle \\ &\mapsto \sum_z c_z\, \big|\mathrm{str}_\nu(t-z)\big\rangle \otimes \big|f(s+z)\big\rangle, \end{aligned}$$

where $z \in \{0,1\}$. The last step, $|z\rangle \mapsto |\,\mathrm{str}_\nu(t-z)\rangle$, requires that we discriminate between the states $|\,\mathrm{str}_\nu(t)\rangle$ and $|\,\mathrm{str}_\nu(t-1)\rangle$. This is easy because the supports of those states on the ν-grid do not overlap.

Let us now consider the general case, $G = \mathbb{R}^k \times \mathbb{Z}^l$. The group G is embedded in $\tilde{G} = \mathbb{R}^{k+l}$ by scaling the $\mathbb{Z}$ factors by λ. This is to guarantee that $\lambda_1(\tilde{L}) \geqslant \lambda$, where $\tilde{L}$ is the image of L under the embedding. The other condition on the new hidden subgroup reads: $d(\tilde{L}) \leqslant \tilde{d}$, where $\tilde{d} = d\lambda^l$. The generalization of Eq. (6.2) is straightforward:

$$\begin{aligned}\big|g(\boldsymbol{x}, x_1, \ldots, x_l)\big\rangle = {}& \sum_{z_1,\ldots,z_l \in \{0,1\}} \left(\bigotimes_{j=1}^{l} |\psi(x_j, z_j)\rangle \right) \\ & \otimes \big|f(\boldsymbol{x},\, s(x_1,z_1), \ldots, s(x_l,z_l))\big\rangle, \end{aligned} \quad (6.3)$$

where $s(x,z) = \lfloor x/\lambda \rfloor + z$, $|\psi(x,z)\rangle = \cos\left(\frac{\pi}{2}t\right) |\mathrm{str}_\nu(t)\rangle$, with $t = x/\lambda - s(x,z)$.

Note that the terms in the above sum are mutually orthogonal vectors. It can be shown that g has parameters $\tilde{a}^2 = a^2 + l\left(\frac{\pi}{2\nu\lambda}(1+\nu)\right)^2$, $\tilde{r}^2 = r^2 + l(2\nu\lambda)^2$ and ϵ.

6.2 An HSP algorithm for the group $\mathbb{R}^m$

Let f be an (a, r, ε) oracle function for some full-rank lattice $L \subseteq \mathbb{R}^m$ such that $\lambda_1(L) \geqslant \lambda$ and $d(L) \leqslant d$ (see Definition 1.1). The core part of our algorithm is a sampling subroutine that generates an approximation to a random point of the reciprocal lattice L^*. It works under certain assumptions about the oracle parameters.

Let $\omega : \mathbb{R} \to \mathbb{C}$ be some Lipschitz function with unit L^2-norm supported on the interval $[0,1]$. For example,

$$\omega(x) = \begin{cases} \sqrt{2}\,\sin(\pi x) & \text{for } x \in [0,1], \\ 0 & \text{otherwise.} \end{cases} \qquad (6.4)$$

Let us also choose a sufficiently large number $\Delta = 2^{q_1}$ and a sufficiently small number $\delta = 2^{-q_2}$. Define

$$w(x_1, \ldots, x_m) = \frac{1}{\Delta^{m/2}} \prod_{j=1}^{m} \omega\left(\frac{x_j}{\Delta}\right), \qquad (6.5)$$

$$w_\delta = w|_{\delta\mathbb{Z}^m} \ \big(\text{restriction of } w \text{ to the lattice } \delta\mathbb{Z}^m\big).$$

In our calculations, we will use the following variables:

Real domain:	$x = \delta\tilde{x} \in \delta\mathbb{Z}^m$	or	$\tilde{x} \in \mathbb{Z}^m$;
Fourier domain:	$y = \delta^{-1}\tilde{y} \in \mathbb{R}^m/\delta^{-1}\mathbb{Z}^m$	or	$\tilde{y} \in \mathbb{R}^m/\mathbb{Z}^m$.

We first create the superposition of points x with the wavefunction w_δ. In the quantum computer, x is actually represented by $\tilde{x}$, therefore the initial state may be written as follows:

$$|w_\delta\rangle = \delta^{m/2} \sum_{\tilde{x} \in \mathbb{Z}^m} w(x) |\tilde{x}\rangle \quad \text{with } x = \delta\tilde{x}.$$

(Our choice of the function ω guarantees the correct normalization on the δ-grid; otherwise we would need to multiply the above expression by some factor that tends to 1 as δ tends to 0.) Then we apply the oracle to get the state

$$|\psi_\delta\rangle = \delta^{m/2} \sum_{\tilde{x} \in \mathbb{Z}^m} w(x)\, |\tilde{x}\rangle \otimes |f(x)\rangle \quad \text{with } x = \delta\tilde{x}. \qquad (6.6)$$

The quantum register containing $f(x)$ may be ignored, and we would like to measure the other register in the Fourier basis,

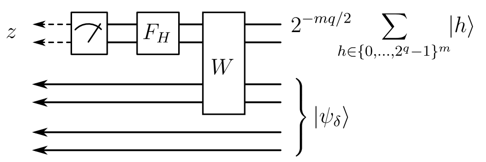
$$|\xi_{\tilde{y}}\rangle = F_{\mathbb{Z}^m}^{-1} |\tilde{y}\rangle = \sum_{\tilde{x} \in \mathbb{Z}^m} e^{-2\pi i \langle \tilde{x}, \tilde{y} \rangle} |\tilde{x}\rangle, \qquad \text{where } \tilde{y} \in (\mathbb{R}/\mathbb{Z})^m.$$

An obvious procedure would be to perform the Fourier transform and measure in the standard basis. However, a quantum computer can only do the Fourier transform over a finite group, and the resulting approximation errors are difficult to analyze. Therefore we use a different method.

Note that $|\xi_{\tilde{y}}\rangle$ is an eigenvector of the mutually commuting translation operators $T_{e_1}, \ldots, T_{e_m}$, where e_j ($j = 1, \ldots, m$) are the generators of the group $\mathbb{Z}^m$. The translation by $h \in G$ on an Abelian group G is defined as follows:

$$T_h : \mathrm{L}^2(G) \to \mathrm{L}^2(G), \qquad (T_h f)(x) = f(x - h). \qquad (6.7)$$

Thus, the Fourier measurement is equivalent to measuring the eigenvalues of the unitary operators T_{e_j}. A general procedure for the eigenvalue measurement (a.k.a. phase estimation) is described in [Kit95]. To illustrate the difference from the direct use of discrete Fourier transform, let us consider a variant of the phase estimation. In the following circuit, the Fourier transform F_H on the group $H = (\mathbb{Z}_{2^q})^m$ acts on a set of ancillary qubits.



Here $W|h, \tilde{x}\rangle = |h,\, \tilde{x} + h\rangle$. For each value of h, the operator W acts on $\tilde{x}$ as $T_1^{h_1} \cdots T_m^{h_m}$. Note that the "+" in the definition of W means the addition of integer vectors, which are *not* reduced modulo 2^q. Thus, W preserves the decomposition of $|\psi_\delta\rangle$ into the vectors $|\xi_{\tilde{y}}\rangle$. The measurement outcome z may be regarded as a random variable conditioned on $\tilde{y}$, and the latter can be inferred from the former with some precision and confidence. The final result of the sampling subroutine, $Y = -\delta^{-1} 2^{-q} z_j$ provides an approximation for y. The error bound for this procedure is pretty standard.

LEMMA 6.2. *For each j, the probability that the inferred value $\tilde{Y}_j = -2^{-q} z_j$ deviates from $\tilde{y}_j$ by $\geqslant \tilde{\nu}$ is at most $2^{-q}/\tilde{\nu}$. Thus, Y approximates y with precision ν in each coordinate, up to an error probability $\mu_{meas} = m2^{-q}/(\delta\nu)$.*

To further analyze the sampling subroutine, we approximate the probability distribution $p_\delta(y)$ of the variable $y = \delta^{-1}\tilde{y}$ by the distribution p that occurs in the $\delta \to 0$ limit. These two distributions are derived from the following quantum states (cf. Eq. (6.6)):

$$|\psi_\delta\rangle = \delta^{m/2} \sum_{\tilde{x} \in \mathbb{Z}^m} |\tilde{x}\rangle \otimes |\psi(x)\rangle, \quad |\psi\rangle = \int_{\mathbb{R}^m} |x\rangle \otimes |\psi(x)\rangle\, dx,$$

where $|\psi(x)\rangle = w(x)|f(x)\rangle$. (The function $w : \mathbb{R}^m \to \mathbb{C}$ is given by Eq. (6.5); the hidden subgroup oracle f and, thus, the function ψ, are vector-valued, i.e. $f, \psi : \mathbb{R}^m \to \mathcal{H}$.) More exactly, p_δ and p are related to the Fourier transform of ψ_δ and ψ, respectively:

$$p_\delta(y) = \langle \widehat{\psi}_\delta(y) | \widehat{\psi}_\delta(y) \rangle \qquad \text{with} \quad \widehat{\psi}_\delta = F_{\delta\mathbb{Z}^m}\psi;$$
$$p(y) = \langle \widehat{\psi}(y) | \widehat{\psi}(y) \rangle \qquad \text{with} \quad \widehat{\psi} = F_{\mathbb{R}^n}\psi. \qquad (6.8)$$

It can be shown that p_δ is close to p.

Let us now focus on the distribution $p(y) = \langle \widehat{\psi}(y) | \widehat{\psi}(y) \rangle$. We have

$$\psi = wf, \qquad \widehat{\psi} = \widehat{w} * (F_{\mathbb{R}^m} f), \qquad \text{where} \quad \widehat{w} = F_{\mathbb{R}^m} w.$$

Since f is a hidden subgroup oracle, we may regard it as a function on $\mathbb{R}^m/L$ and define $\widehat{f} = F_{\mathbb{R}^m/L} f$. That is,

$$\widehat{f}_u = \int_{\mathbb{R}^m/L} e^{2\pi i \langle x, u \rangle} f(x)\, dx \qquad \text{for } u \in L^*,$$

$$f(x) = \frac{1}{d(L)} \sum_{u \in L^*} e^{-2\pi i \langle x, u \rangle} \widehat{f}_u. \qquad (6.9)$$

The Fourier transform over $\mathbb{R}^m$ is

$$\begin{aligned}(F_{\mathbb{R}^m} f)(y) &= \int_{\mathbb{R}^m} e^{2\pi i\langle x,y\rangle} \left(\frac{1}{d(L)} \sum_{u\in L^*} e^{-2\pi i\langle x,u\rangle} \widehat{f}_u \right) dx \\ &= \frac{1}{d(L)} \sum_{u\in L^*} \widehat{f}_u \,\delta(y-u).\end{aligned}$$

It follows that

$$\widehat{\psi}(y) = \big(\widehat{w} * (F_{\mathbb{R}^m} f)\big)(y) = \frac{1}{d(L)} \sum_{u\in L^*} \widehat{f}_u \,\widehat{w}(y-u), \qquad (6.10)$$

$$p(y) = \frac{1}{d(L)^2} \sum_{u,u'\in L^*} \big\langle \widehat{f}_{u'} \big| \widehat{f}_u \big\rangle \,\widehat{w}(y-u)\, \overline{\widehat{w}(y-u')}. \qquad (6.11)$$

The last equation is complicated, but to have a good approximation it is enough to keep the terms with $u = u'$. Let us consider the quantum state $F_{\mathbb{R}^m}|\psi\rangle$ whose wavefunction is given by Eq. (6.10). It consists of identically shaped peaks at the points $u \in L^*$. Each peak has a weight

$$\boxed{q_u = \frac{\langle \widehat{f}_u | \widehat{f}_u \rangle}{d(L)^2} = \frac{1}{d(L)^2} \int e^{2\pi i\langle x-x',u\rangle} \big\langle f(x') \big| f(x) \big\rangle \,dx\,dx',}$$

where the integral is over $(\mathbb{R}^m/L)^2$. The numbers q_u can be interpreted as probabilities because they are nonnegative and add up to 1. Indeed, let us normalize f to make a function of unit norm, $g(x) = d(L)^{-1/2} f(x)$. Then

$$\sum_{u\in L^*} q_u = \frac{1}{d(L)} \sum_{u\in L^*} \langle \widehat{g}_u | \widehat{g}_u \rangle = \langle \widehat{g} \,|\, \widehat{g} \,\rangle = \langle g|g\rangle = 1.$$

7. REFERENCES

[Ban93] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1):625–635, 1993.

[BK93] Johannes Buchmann and Volker Kessler. Computing a reduced lattice basis from a generating system, 1993. Preprint, August 4, 1993.

[BP87] Johannes Buchmann and Michael Pohst. Computing a lattice basis from a system of generating vectors. In *Eurocal'87*, volume 378 of *LNCS*, pages 54–63. Springer-Verlag, June 1987.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *Advances in cryptology—CRYPTO 2011*, volume 6841 of *LNCS*, pages 505–524. Springer, 2011.

[Coh93] Henri Cohen. *A course in computational algebraic number theory.* Springer-Verlag New York, Inc., New York, NY, USA, 1993.

[FIM$^+$03] Katalin Friedl, Gabor Ivanyos, Frederic Magniez, Miklos Santha, and Pranab Sen. Hidden translation and orbit coset in quantum computing. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*, San Diego, CA, 9–11June 2003.

[GH11] C. Gentry and S. Halevi. Implementing gentry's fully-homomorphic encryption scheme. *Eurocrypt 2011*, pages 132–150, 2011.

[GKP01] Daniel Gottesman, Alexei Kitaev, and John Preskill. Encoding a qubit in an oscillator. *Phys. Rev. A*, 64:012310, Jun 2001.

[GVL96] Gene H. Golub and Charles F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

[Hal05] Sean Hallgren. Fast quantum algorithms for computing the unit group and class group of a number field. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 468–474, 2005.

[Hal07] Sean Hallgren. Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem. *Journal of the ACM*, 54(1):1–19, 2007.

[HMR$^+$10] Sean Hallgren, Cristopher Moore, Martin Rötteler, Alexander Russell, and Pranab Sen. Limitations of quantum coset states for graph isomorphism. *J. ACM*, 57:34:1–34:33, November 2010.

[Kit95] Alexei Kitaev. Quantum measurements and the abelian stabilizer problem, 1995. quant-ph/9511026.

[KW08] Alexei Kitaev and William A. Webb. Wavefunction preparation and resampling using a quantum computer, January 2008. arXiv:0801.03422.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Advances in cryptology—EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, 2010.

[Mic08] Daniele Micciancio. Efficient reductions among lattice problems. In *Proceedings of SODA 2008*, pages 84–93, New York, 2008. ACM.

[PR07] Chris Peikert and Alon Rosen. Lattices that admit logarithmic worst-case to average-case connection factors. In *STOC '07: Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 478–487, New York, NY, USA, 2007. ACM Press.

[San91] Jonathan W. Sands. Generalization of a theorem of Siegel. *Acta Arith.*, 58(1):47–57, 1991.

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

[SV05] Arthur Schmidt and Ulrich Vollmer. Polynomial time quantum algorithm for the computation of the unit group of a number field. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 475–480, 2005.

[Thi95] Christoph Thiel. *On the complexity of some problems in algorithmic algebraic number theory.* PhD thesis, Universität des Saarlandes, Saarbrücken, Germany, 1995.