Quantum algorithm for Discrete Gaussian Sampling

Clémence Chevignard, Yixin Shen, and André Schrottenloher

Univ Rennes, Inria, CNRS, IRISA, Rennes, France

Abstract. Discrete Gaussian Sampling on lattices is a fundamental problem in lattice-based cryptography. It appears both in basic cryptographic primitives such as digital signatures and as an important cryptanalysis building block for solving hard lattice problems. In this paper, we show a quantum algorithm based on the quantum rejection sampling technique whose complexity is asymptotically quadratically faster than its classical counterpart in [Wang & Ling, IEEE Trans. Inf. Theory 2019]. Our sampler outputs a quantum state which can either be measured to get the desired distribution or be used directly as such in other quantum algorithms. By doing so, we derive two versions of quantum dual attacks that improve upon the previous ones in [Pouly & Shen, EUROCRYPT 2024]. The two versions are incomparable, each having distinct advantages (speed vs memory requirement). The second version is particularly interesting as it requires only polynomial classical and quantum memory, excluding the classical memory used in the preprocessing step of the Discrete Gaussian sampler. Our quantum Discrete Gaussian sampler can also be used to speed up the algorithm for solving the Short Integer Solution problem, in any norm, of [Bollauf, Pouly & Shen, ePrint 2026/225].

Keywords: Quantum cryptanalysis · Lattice-based cryptography · Quantum rejection sampling · Discrete Gaussian Sampling

1 Introduction

Discrete Gaussian Sampling on lattices is a fundamental problem in lattice-based cryptography. Given a lattice, a vector $\boldsymbol{c}$ and a parameter σ, it asks to sample a lattice vector $\boldsymbol{x}$ with probability proportional to $e^{-\frac{\pi\|\boldsymbol{x}-\boldsymbol{c}\|^2}{\sigma^2}}$. The problem arises both in basic cryptographic primitives such as "hash-and-sign" digital signature schemes [27,44], and in cryptanalysis, where it serves as a fundamental tool for solving hard problems like the Shortest Vector Problem [3], the Learning with Errors problem [42,45] and the Short Integer Solution problem [12].

The width of the distribution is controlled by σ. The smaller the width, the harder it is to sample from this distribution. A particularly important threshold is known as the smoothing parameter [37] which marks the transition between the distribution looking more like a continuous Gaussian, and the distribution being more heavily concentrated on the closest lattice vector to the center $\boldsymbol{c}$.

There are several algorithms to sample from the discrete Gaussian distribution with various trade-offs. The most well-known algorithm, known as the Klein sampler, is very efficient (polynomial time) but the width depends on the quality of the input basis [27]. This makes this algorithm useful for relatively large values of σ, much larger than the smoothing parameter, such as in cryptographic primitives. At the other end of the spectrum is the algorithm by [3] which can sample the (centered) discrete Gaussian for any value of σ in time $2^{n+o(n)}$ and above the smoothing parameter in time $2^{n/2+o(n)}$. While this algorithm has some theoretical applications [1,2], its complexity is too high for practical attacks on cryptosystems. The algorithm of [1] provides a time-space tradeoff ranging from exponential to super-exponential time, and exponential to polynomial space. This algorithm only works for a width above the smoothing parameter. Finally, the algorithms of [49] and [16], which are essentially equivalent despite using a very different presentation[1], work for any σ and provide an interesting trade-off where the complexity depends on both the quality of the input basis and σ. This algorithm has been used in several attacks [42,43,45,12] in the regime where its complexity is exponential but much smaller than 2^n.

There is no known quantum speedup on the above algorithms. The Klein sampler, which only requires polynomial space, can easily be turned into a quantum sampler (as shown in [11], which we reuse in Section 3) but does not provide a speed-up per se and still suffers from the strong constraint on the width. The algorithm of [3] requires exponential memory and cannot be turned into a quantum sampler without using quantum RAM. Finally, the algorithm of [49,16] can also be turned into a quantum sampler with a polynomial number of qubits with no particular speedup.

Contributions. In this paper, we present a quantum algorithm allowing to sample vectors according to a discrete Gaussian distribution, with an (asymptotically) quadratic speed-up on the complexity of the algorithm of [49]. Our sampler works for any width σ and its complexity depends on the width and the quality of the input basis.

Our algorithm is based on *quantum rejection sampling* [41], i.e., transforming the amplitudes of a quantum state from a given distribution to a target distribution – in a way analogous to classical rejection sampling. Consider that we want to sample vectors in a lattice $\mathcal{L}$, according to a Gaussian lattice distribution D, then our objective is to build a quantum state (close to) $\sum_{\boldsymbol{x}\in\mathcal{L}} \sqrt{D(\boldsymbol{x})}\,|\boldsymbol{x}\rangle$, since measuring this state gives us a vector $\boldsymbol{x}$ sampled according to D.

In order to build this target state, we start from an algorithm which prepares another state $\sum_{\boldsymbol{x}\in\mathcal{L}} \sqrt{D_1(\boldsymbol{x})}\,|\boldsymbol{x}\rangle$, which in our case is a quantum version of Klein's sampling algorithm, previously given in [11]. We modify the target amplitudes, and then use quantum amplitude amplification over these steps to reach the target state.

[1] See Remark 1 in [43].

Application to SIS and Dual Attacks on LWE. We use our quantum sampling algorithm in three distinct algorithms solving the Learning with Errors (LWE) and the Short Integer Solution (SIS) problems. Our first application consists in replacing the first step of the quantum dual attack of Pouly and Shen [42], which asks to sample vectors according to a discrete Gaussian. Using our algorithm, we can obtain a quadratic speedup on this first step (excluding the preprocessing step for BKZ reduction), while the second step (which uses the data produced inside an FFT-based distinguisher) remains unchanged.

Our second application is a new quantum dual attack, in which the classical data used in the second step of the previous quantum dual attack [42] is replaced by our quantum sampling algorithm. This algorithm has a higher time complexity, but uses only polynomial quantum memory.

Our third application consists in quantizing the algorithm of Bollauf, Pouly and Shen [12] for solving the SIS problem for any norm. Using our quantum sampling algorithm and quantum amplitude amplification, we can construct a quantum state that provides a solution of the SIS problem with high probability. Excluding the BKZ basis reduction preprocessing step, which itself allows for a quantum speedup, the remaining step achieves a quadratic speedup over its classical counterpart. Ignoring polynomial factors in the complexity, we provide rough estimates of the quantum complexity of the SIS^∞ problem in the Dilithium signature scheme.

Like many works in lattice-based quantum cryptanalysis [20,8,13], we focus on the asymptotic complexity of algorithms and on theoretical quantum speedups against classical algorithms. It has been acknowledged before that such asymptotic speedups may be difficult to ever realize in practice, especially if they require memory, due to the inherent cost in maintaining quantum RAM and the sequentiality of Grover's quantum search [7]. While these issues would also present themselves for the algorithms that we study in this paper, our goal is primarily to expand the quantum cryptanalysis toolbox. We leave detailed considerations for implementations, such as depth and non-asymptotic gate counts, for future work.

Organization of the Paper. In Section 2, we present several useful notions: lattices and Gaussian distributions on lattices, the LWE and the SIS problems, and some useful quantum algorithms. In Section 3, we describe our quantum algorithm for sampling from discrete Gaussian distributions: Section 3.4 shows that this algorithm essentially obtains a quadratic speedup compared to the Markov Chain Monte Carlo (MCMC) algorithm [49].

We then move to applications. In Section 4, we apply our quantum sampling algorithm to the "modern" dual attack framework on LWE of [42]; first by introducing the quantum sampler in the first step of the algorithm, second by introducing the quantum sampler in the second step. In Section 5, we apply our quantum sampling algorithm to quantize the algorithm of [12] for solving the SIS problem. We give some rough estimates on the quantum complexity of SIS^∞ used in the Dilithium signature scheme.

2 Preliminaries

Throughout this paper, we assume familiarity with basic quantum computing notions such as quantum states, qubits, the ket notation $|\cdot\rangle$, etc. We write quantum algorithms as quantum circuits, which is the most common approach in quantum cryptanalysis; a full overview of the quantum circuit model can be found in [40]. The gate count (i.e., number of operations) is our main metric of interest, but since we consider asymptotic algorithms, we do not specify explicitly a universal gate set.

The algorithms that we consider have polynomial qubit count, but they may require exponential-size *quantum RAM* (qRAM). The model used is *classical memory with quantum random access* (QRACM). It extends the quantum circuit model with the operation: $|i\rangle|x\rangle \mapsto |i\rangle|x \oplus m_i\rangle$, where $(m_0, \ldots, m_{\ell-1})$ is an exponential-size classical memory and i is an index register. When using QRACM, we make the assumption that a specific physical component allows to implement this operation in time $O(1)$, since the circuit model cannot. qRAM has been part of all quantum speedups for lattice sieving to date [35,20,13]. A recent paper [21] showed that under some reasonable hypothesis, there is no quantum speed-up possible for lattice sieving without using qRAM. This is opposed to lattice enumeration [9], since it runs with polynomial memory. Indeed, any quantum algorithm using polynomial space overall does not need qRAM.

For Ω a set and $\alpha_j > 0$, the state $|\psi\rangle = \sum_{j\in\Omega} \alpha_j |j\rangle$ represents a probability distribution Π over Ω, defined by $\Pi(j) = \alpha_j^2$. When this is clear from context, we will denote this state as $|\Pi\rangle$. To measure the distance between two probability distributions μ_1 and μ_2, we use the *total variation distance*, defined as

$$d_{TV}(\mu_1, \mu_2) = \frac{1}{2} \sum_{x\in\Omega} |\mu_1(x) - \mu_2(x)| \ .$$

2.1 Lattices

A $\mathbb{R}$-*lattice* $\mathcal{L}$ is a discrete subgroup of $\mathbb{R}^n$ spanned by a set of independent vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n \in \mathbb{R}^m$ over $\mathbb{Z}$, meaning that $\mathcal{L} = \{\sum_{i=1}^n a_i \boldsymbol{b}_i, a_i \in \mathbb{Z}\}$. When $n = m$, we say that the lattice $\mathcal{L}$ is full rank. The vectors $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$ form a *basis* of $\mathcal{L}$. Usually, to represent this basis, we concatenate these column vectors into a matrix B, and we write $\mathcal{L} = \mathcal{L}(B)$. The matrix $\widetilde{B}$ denotes the Gram-Schmidt decomposition of B, and we denote by $\widetilde{\boldsymbol{b}}_i$ its column vectors. The length of the shortest nonzero vector of $\mathcal{L}(B)$ is denoted by $\lambda_1(\mathcal{L}(B))$.

The *fundamental parallelepiped* of $\mathcal{L}(B)$ is $P(\mathcal{L}(B)) := \{\sum_{i=1}^n a_i \boldsymbol{b}_i, a_i \in [0, 1)\}$. The volume of the lattice $\mathcal{L}(B)$ is the volume $\mathrm{vol}(P(\mathcal{L}(B)))$ of $P(\mathcal{L}(B))$. It also equals $\sqrt{\det(B^t B)}$, or simply $|\det(B)|$ when B is square and full rank.

Define, in the same fashion, $\mathrm{span}_{\mathbb{R}}(\mathcal{L}(B)) := \{\sum_{i=1}^n a_i \boldsymbol{b}_i, a_i \in \mathbb{R}\}$. The dual $\hat{\mathcal{L}}$ of the lattice $\mathcal{L}$ is defined as

$$\hat{\mathcal{L}} := \{\boldsymbol{y} \in \mathrm{span}_{\mathbb{R}}(\mathcal{L}) \text{ such that } \forall \boldsymbol{x} \in \mathcal{L}, \langle \boldsymbol{x}, \boldsymbol{y}\rangle \in \mathbb{Z}\} \ .$$

Let $1 \leqslant n \leqslant m$, and let q be a prime power. A lattice $\mathcal{L}$ is a m-dimensional q-ary lattice if $q\mathbb{Z}^m \subset \mathcal{L} \subset \mathbb{Z}^m$. Given a matrix $A \in \mathbb{Z}^{m\times n}$, we can define two m-dimensional q-ary lattices

$$\mathcal{L}_q(A) := \{\boldsymbol{x} \in \mathbb{Z}^m \text{ such that } \exists \boldsymbol{s} \in \mathbb{Z}^n \text{ such that } A\boldsymbol{s} = \boldsymbol{x} \bmod q\},$$
$$\mathcal{L}_q^{\perp}(A) := \{\boldsymbol{x} \in \mathbb{Z}^m \text{ such that } A^T\boldsymbol{x} = 0 \bmod q\}.$$

There exists matrices A and B such that $\mathcal{L} = \mathcal{L}_q(A) = \mathcal{L}_q^{\perp}(B)$. Moreover, $\widehat{\mathcal{L}_q^{\perp}(A)} = \frac{1}{q}\mathcal{L}_q(A)$, and $\det(\mathcal{L}_q(A)) = q^{m-rk(A)}$, $\det(\mathcal{L}_q^{\perp}(A)) = q^{rk(A)}$. From now on, we will always consider that A has full rank.

2.2 Discrete Gaussian Distributions on Lattices and Variants

Discrete Gaussian Distributions. Consider vectors $\boldsymbol{c}, \boldsymbol{x} \in \mathbb{R}^m$ and a real $\sigma > 0$. The *Gaussian function centered at* $\boldsymbol{c}$ for standard deviation σ is

$$\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x}) := e^{-\frac{\pi||\boldsymbol{x}-\boldsymbol{c}||^2}{\sigma^2}} \quad .$$

From this function, we define *Discrete Gaussian distributions* on lattices. Let us fix $\boldsymbol{c} \in \mathbb{R}^m$, and consider a lattice $\mathcal{L} \in \mathbb{R}^m$, some $\boldsymbol{x} \in \mathcal{L}$, and $\sigma > 0$. The Gaussian lattice distribution defined on $\mathcal{L}$, centered in $\boldsymbol{c}$ and with standard deviation σ, is

$$D_{\mathcal{L},\sigma,\boldsymbol{c}}(\boldsymbol{x}) := \frac{\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})}{\rho_{\sigma,\boldsymbol{c}}(\mathcal{L})} = \frac{e^{-\frac{\pi}{\sigma^2}||\boldsymbol{x}-\boldsymbol{c}||^2}}{\sum_{\boldsymbol{y}\in\mathcal{L}} e^{-\frac{\pi}{\sigma^2}||\boldsymbol{y}-\boldsymbol{c}||^2}} \quad .$$

The denominator $\rho_{\sigma,\boldsymbol{c}}(\mathcal{L})$ is a scaling factor so that $D_{\mathcal{L},\sigma,\boldsymbol{c}}(\mathcal{L}) = 1$. When $\boldsymbol{c} = 0$, we simply write $D_{\mathcal{L},\sigma}$ instead of $D_{\mathcal{L},\sigma,\boldsymbol{c}}$. We use the following lemmas to bound Gaussian masses and their tails.

Lemma 1 ([38, Lemma 2.9]). *Let $\mathcal{L}$ be a lattice in $\mathbb{R}^m$. Then, for any $\sigma > 0$ and $\boldsymbol{c} \in \mathbb{R}^m$, we have $\rho_{\sigma,\boldsymbol{c}}(\mathcal{L}) \leqslant \rho_{\sigma}(\mathcal{L})$.*

Lemma 2. *[From Lemma 2 in [43]] For any $\sigma > 0$, $\rho_\sigma(\mathbb{Z}) \leq 3\sigma + 4$.*

Lemma 3. *Consider an integer $M \in \mathbb{N}^*$, and real numbers $\sigma > 0$ and $\boldsymbol{c} \in \mathbb{R}$. Then, we have*

$$\rho_{\sigma,\boldsymbol{c}}(\mathbb{Z}) \leqslant \frac{\rho_\sigma([\![-2^M, 2^M]\!])}{1 - 2^M\sqrt{2\pi e} \times e^{-\pi 2^{2M}}} \text{ and } \rho_{\sigma,\boldsymbol{c}}(\mathbb{Z}\backslash[\![-2^M, 2^M]\!]) \leqslant O(\sigma \times 2^M e^{-\pi 2^{2M}}) \quad .$$

The proof is given in Appendix B.

Periodic Gaussian Function. The periodic Gaussian function is defined as follows:

$$f_{\mathcal{L},\sigma}(\boldsymbol{x} \in \mathbb{R}^m) := \frac{\rho_\sigma(\mathcal{L}+\boldsymbol{x})}{\rho_\sigma(\mathcal{L})} \tag{1}$$

This function is clearly periodic over $\mathcal{L}$. Therefore, one can define its Fourier coefficients as follows:

$$\text{For } w \in \hat{\mathcal{L}},\ \hat{f}_{\mathcal{L},\sigma}(w) = \frac{1}{\text{vol}(\mathcal{L})} \int_{z \in P(\mathcal{L})} f_{\mathcal{L},\sigma}(z) e^{-2\pi i \langle w,z \rangle} dz \ .$$

Moreover, the Fourier series of $f_{\mathcal{L},\sigma}$ at $\boldsymbol{x}$ is defined as:

$$\sum_{w \in \hat{\mathcal{L}}} \hat{f}_{\mathcal{L},\sigma}(w) e^{2\pi i \langle \boldsymbol{x},\boldsymbol{w} \rangle} \ .$$

Lemma 4 ([4, Fact 2.3]). *For any sufficiently smooth function $f : \mathbb{R}^n \to \mathbb{R}$ that is periodic over some lattice $\mathcal{L}$ and any $x \in \mathbb{R}^n$, the Fourier series of f at x is equal to $f(x)$.*

The function $f_{\mathcal{L},\sigma}$ is sufficiently smooth for this lemma to apply, for any $\mathcal{L}(B)$ and $\sigma > 0$. We then have the following result.

Lemma 5 ([4, Claim 4.1]). *For any lattice $\mathcal{L} \subset \mathbb{R}^m$, and $\sigma > 0$, we have $\hat{f}_{\mathcal{L},\sigma} = D_{\hat{\mathcal{L}},1/\sigma}$, which is a probability measure over the dual lattice $\hat{\mathcal{L}}$.*

Later on, we will need to estimate the distance of an element to a lattice $\mathcal{L}$ efficiently. One of the results used in [42] to do so is the following Lemma, which is based on the Chernoff-Hoeffding bound.

Lemma 6 (Pointwise approximation [4, Lemma 1.3],[42, Lemma 6]). *Consider $\mathcal{L}$ a full-rank lattice in $\mathbb{R}^m$, and $h : \mathbb{R}^m \to \mathbb{R}$ that is periodic over $\mathcal{L}$ and whose Fourier series $\hat{h}$ is a probability measure over $\hat{\mathcal{L}}$. Let N be an integer, $\delta > 0$, and $X \subset \mathbb{R}^m$ a finite set. Let $W = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N)$ be a list of vectors in the dual lattice chosen randomly and independently from the distribution $\hat{h}$. Then, with probability at least $1 - |X| 2^{-\Omega(N\delta^2)}$, for all $\boldsymbol{x} \in \mathcal{L} + X$:*

$$|h_W(\boldsymbol{x}) - h(x)| \leqslant \delta, \ \textit{where } h_W(\boldsymbol{x}) := \frac{1}{N} \sum_{i=1}^{N} \cos(2\pi \langle \boldsymbol{w}_i, \boldsymbol{x} \rangle) \ .$$

By Lemma 5, the pointwise approximation lemma applies to $f_{\mathcal{L},\sigma}$.

Variants of Discrete Gaussian Distributions for Finite Probability Spaces. In the rest of the paper, we will build registers $|s\rangle$ representing probability distributions close to either $D_{\mathcal{L},\sigma}$, or the "Klein distribution" (that we will define later). However, a register contains only finitely many qubits, so it can only represent a probability distribution over a finite space, while $D_{\mathcal{L},\sigma}$ is

defined over an infinite space. Let Ω be a finite subset of $\mathcal{L}$, we define the *finite discrete gaussian distribution* with standard deviation σ as

$$D_{\Omega,\sigma}(\boldsymbol{x} \in \Omega) = \frac{\rho_\sigma(\boldsymbol{x})}{\rho_\sigma(\Omega)} = \frac{e^{-\frac{\pi}{\sigma^2}||\boldsymbol{x}||^2}}{\sum_{\boldsymbol{y}\in\Omega} e^{-\frac{\pi}{\sigma^2}||\boldsymbol{y}||^2}} \ .$$

We have:

$$\begin{aligned} 2d_{TV}(D_{\Omega,\sigma}, D_{\mathcal{L},\sigma}) &= \sum_{\boldsymbol{x}\in\Omega} |D_{\Omega,\sigma}(\boldsymbol{x}) - D_{\mathcal{L},\sigma}(\boldsymbol{x})| + \sum_{\boldsymbol{x}\in\mathcal{L}\setminus\Omega} |D_{\Omega,\sigma}(\boldsymbol{x}) - D_{\mathcal{L},\sigma}(\boldsymbol{x})| \\ &= \sum_{\boldsymbol{x}\in\Omega} D_{\Omega,\sigma}(\boldsymbol{x}) \left(1 - \frac{\rho_\sigma(\Omega)}{\rho_\sigma(\mathcal{L})}\right) + \sum_{\boldsymbol{x}\in\mathcal{L}\setminus\Omega} D_{\mathcal{L},\sigma}(\boldsymbol{x}) \\ &= \left(1 - \frac{\rho_\sigma(\Omega)}{\rho_\sigma(\mathcal{L})}\right) + \rho_\sigma(\mathcal{L}\setminus\Omega) \\ &= \rho_\sigma(\mathcal{L}\setminus\Omega)(1 + 1/\rho_\sigma(\mathcal{L})) \leqslant 2\rho_\sigma(\mathcal{L}\setminus\Omega) \ . \end{aligned} \tag{2}$$

Proposition 1. *Consider a full rank lattice $\mathcal{L}(B) \in \mathbb{R}^m$, along with a finite subset $\Omega \supset B\prod_{i=1}^m [\![-2^{M-1}, 2^{M-1}]\!]$, for some integer $M > 0$. We have:*

$$\rho_\sigma(\mathcal{L}\setminus\Omega) \leqslant 2\left(2^{M-1}\sqrt{2\pi e}\right)^m \times e^{-\pi\cdot m\cdot 2^{2M-2}} (3\sigma/\min_i \|\widetilde{\boldsymbol{b}}_i\| + 4)^m \ .$$

The proof is given in Appendix B.

Now, we define the (finite) *Klein distribution* which arises from a variant of the Klein sampling algorithm [32]. Indeed, the probability space of the original Klein distribution of [32] is $\mathcal{L}(B)$, which is an infinite space. Therefore we define a variant that samples from a finite set Ω of $\mathcal{L}(B)$.

Algorithm 1 Klein's sampling algorithm variant

Input: B, σ, $\boldsymbol{c}$, and an integer $M > 1$

Output: $B\boldsymbol{x} \in \Omega$

Write B as $B = QR$, with Q orthogonal and R upper triangular, with $|R_{i,i}| = ||\widetilde{\boldsymbol{b}}_i||$
$\boldsymbol{c}' \leftarrow Q^{-1}\boldsymbol{c}$
for $i = m \to 1$ **do**
    $\sigma_i \leftarrow \sigma/|R_{i,i}|$
    $\widetilde{x}_i \leftarrow \frac{c'_i - \sum_{j=i+1}^m R_{i,j}x_j}{R_{i,i}}$
    Sample x_i from $D_{[\![-2^M+\lfloor\widetilde{x}_i\rceil, 2^M+\lfloor\widetilde{x}_i\rceil]\!],\sigma_i,\widetilde{x}_i}$
Return $B\boldsymbol{x}$

Here $\boldsymbol{c}$ is, again, the center of the distribution. Let us denote by q the Klein distribution yielded by Algorithm 1. We have, since Q is orthogonal:

$$q(B\boldsymbol{x}) = \prod_{i=1}^m \mathbb{P}(x_{m+1-i}|(x_{m-i}, \ldots, x_1)) = \prod_{i=m}^1 \frac{\rho_{\sigma_i,\widetilde{x}_i}(x_i)}{\rho_{\sigma_i,\widetilde{x}_i}([\![-2^M + \lfloor\widetilde{x}_i\rceil, 2^M + \lfloor\widetilde{x}_i\rceil]\!])}$$

$$= \prod_{i=m}^{1} \frac{e^{-\frac{\pi}{\sigma^2}((Q^{-1}\boldsymbol{c})_i - \sum_{j=i+1}^{m} R_{i,j}x_j - x_i R_{i,i})^2}}{\rho_{\sigma_i,\widetilde{x}_i}([\![-2^M + \lfloor \widetilde{x}_i \rceil, 2^M + \lfloor \widetilde{x}_i \rceil]\!])} = \frac{e^{-\frac{\pi}{\sigma^2}\sum_{i=m}^{1}\left((Q^{-1}\boldsymbol{c})_i - \sum_{j=i}^{m} R_{i,j}x_j\right)^2}}{\prod_{i=m}^{1} \rho_{\sigma_i,\widetilde{x}_i}([\![-2^M + \lfloor \widetilde{x}_i \rceil, 2^M + \lfloor \widetilde{x}_i \rceil]\!])}$$

$$= \frac{\rho_{\sigma,\boldsymbol{c}}(B\boldsymbol{x})}{\prod_{i=m}^{1} \rho_{\sigma_i,\widetilde{x}_i}([\![-2^M + \lfloor \widetilde{x}_i \rceil, 2^M + \lfloor \widetilde{x}_i \rceil]\!])} \tag{3}$$

The probability space Ω is a bit hard to fully describe. One can however note that it contains the space $B \times \prod_{i=1}^{m} [\![-2^{M-1}, 2^{M-1}]\!]$, which, when M is chosen large enough, ensures that $D_{\Omega,\sigma}$ is close to $D_{\mathcal{L},\sigma}$. Later (in the proof of Theorem 5) we give an upper bound on the size of the vectors sampled by the algorithm.

2.3 LWE and SIS

The LWE problem was first introduced by Regev in [46]. Recent NIST standardized lattice-based public key cryptosystems, Kyber [14] and Dilithium [23], both rely on the classical and quantum computational hardness of (module version of) LWE.

Let $n, m, q \in \mathbb{N}$, and let χ_e be a probability distribution over $\mathbb{Z}_q$, which is called the *noise distribution*. For any vector $\boldsymbol{s} \in \mathbb{Z}_q^n$, we denote by $LWE(m, \boldsymbol{s}, \chi_e)$ the probability distribution obtained by sampling a matrix $A \in \mathbb{Z}_q^{m \times n}$ uniformly at random, sampling a vector $\boldsymbol{e} \in \mathbb{Z}_q^m$ according to χ_e^m, and outputting $(A, \boldsymbol{b} := A\boldsymbol{s} + \boldsymbol{e})$. We denote by A_i the i-th row of A, e_i the i-th coefficient of e, and so $b_i = \langle A_i, \boldsymbol{s} \rangle + \boldsymbol{e}_i \in \mathbb{Z}_q$. The search-LWE problem is, given oracle access to a sampler $LWE(m, \boldsymbol{s}, \chi_e)$, to find $\boldsymbol{s}$.

There exists two main strategies to solve LWE: the *primal attack* and the *dual attack*. The latter has been studied both in the classical and quantum setting, and will be detailed in Section 4.

The short integer solution (SIS) problem is an important problem in lattice-based cryptography. It asks to find a short non-zero vector in a randomly generated q-ary lattice of a certain form under the ℓ_p norm.

Definition 1. *Let $m \geqslant n \geqslant 1$, q be an integer, $\ell > 0$ and $p \in (0, \infty]$. The search problem $\mathrm{SIS}^p_{n,m,q,\ell}$ (Short Integer Solution) is defined as follows. The input is a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. The goal is to output a vector $\boldsymbol{x} \in \mathbb{Z}^m$ such that $\mathbf{A}\boldsymbol{x} = 0 \bmod q$ and $0 < \|\boldsymbol{x}\|_p \leqslant \ell$.*

The ℓ_2-norm case was first introduced by Ajtai [5] and admits worst-case to average-case reduction for certain parameter regimes. It has found numerous applications in cryptography, including the construction of collision-resistant hash functions, digital signatures, identification schemes, and more recently, zero-knowledge proofs. The hardness of the recently NIST standardized signature scheme Dilithium [22] relies on the hardness of the (module version of the) SIS problem under the ℓ_∞-norm.

2.4 Quantum Search

A fundamental building block of many algorithms (including several of this paper) is quantum amplitude amplification (QAA).

Theorem 1 ([17], adapted). *Let $\mathcal{A}$ be a quantum algorithm with no measurements constructing a quantum state:*

$$\mathcal{A}|0\rangle = \alpha|\psi\rangle|1\rangle + \sqrt{1-\alpha^2}|\phi\rangle|0\rangle$$

where α is a real number and $|\psi\rangle, |\phi\rangle$ are normalized quantum states. Let $QAA(\mathcal{A}) = -\mathcal{A}O_0\mathcal{A}^\dagger O$ be the QAA iterate*, where O flips the phase of $|\psi\rangle$ (basis states marked by 1) only, and O_0 flips the phase of the all-0 basis state only. Then, for all k:*

$$QAA(\mathcal{A})^k\mathcal{A}|0\rangle = \sin((2k+1)\arcsin\alpha)|\psi\rangle|1\rangle + \cos((2k+1)\arcsin\alpha)|\phi\rangle|0\rangle \quad .$$

This theorem tells us that if the algorithm $\mathcal{A}$ "succeeds" (output marked by 1) with probability $\alpha^2 = p$, then by making $O(1/\alpha) = O\big(1/\sqrt{p}\big)$ iterations, we will measure a success with constant probability. This can be increased arbitrarily close to 1 if p is exactly known when running the algorithm [17].

In our paper, we will also need to succeed when only an upper bound on the probability p is given, but it is not exactly known. In that case one can rely on fixed-point quantum search [50]. It modifies the iteration operators to ensure a probability of success $1-\delta^2$ with $O\left(\log(1/\delta)\frac{1}{\sqrt{p}}\right)$ iterations, where p is a lower bound on the probability of success of $\mathcal{A}$. Therefore, we can also have a probability of success exponentially close to 1 at the cost of a polynomial factor in the complexity.

2.5 Quantum Algorithm for Finding the Maximum

Within the quantum attack on LWE of [42], one has to be able, at some point, to find and measure the *maximum* in a superposition, in the following sense. We have a function: $s : \{0,1\}^\ell \to \mathbb{R}$ and a unitary operator computing s in superposition: $|i\rangle|0\rangle \mapsto |i\rangle|s(i)\rangle$. The goal is to find $i = \text{argmax}_{\{0,1\}^\ell} s$. An algorithm that allows to do so is presented in [25]. However, in the setting in which we will have to use this algorithm, we will have to deal with a function s that is probabilistic. More precisely, we will use a function s that, if called several times upon the same input i, will most of the time return the "right" value $s(i)$, and will sometime return a "wrong" value that differs from $s(i)$. To do so, we rely on a combination of maximum-finding with bounded-error quantum search [29], proving the following result in Appendix A. This is essentially a generalization of Theorem 8 in [42].

Theorem 2. *[[25,29]] Let s be an efficient randomized algorithm that, on input $i \in \{0,1\}^\ell$, returns $s(i)$, where $s(i)$ equals a value s_i ("good") with probability at least 9/10, and s returns a "faulty" output with probability at most 1/10. Algorithm 6, parametrised by $k > 0$, finds the i such that s_i is maximal, with probability at least $1 - 1/2^k$. This algorithms calls s a number of times in $O(k\sqrt{2^\ell})$, and uses $O(k\ell\sqrt{2^\ell})$ additional gates, and* poly(ℓ) *qubits.*

2.6 Quantum Algorithm for Computing the Mean

Let $\mathcal{C}$ be a quantum algorithm without measurements constructing a superposition over ℓ qubits: $\mathcal{C}\left|0\right\rangle^{\otimes \ell} = \left|\psi\right\rangle$. Let $\phi : \{0,1\}^{\ell} \to \mathbb{R}$ be a function. Let Y be the random variable obtained by measuring the output of $\mathcal{C}\left|0\right\rangle^{\otimes \ell}$, and applying ϕ on it. We use a result of [39] to estimate the mean value of Y, assuming that $\mathrm{Var}(Y) \leq v^2$ for some value v.

Theorem 3 ([39, Theorem 5 and Algorithm 3, adapted]). *For any $\epsilon > 0$, there exists a quantum algorithm that calls $\mathcal{C}$ (and its inverse) a total of $O((v/\epsilon)\log^{3/2}(v/\epsilon)\log\log(v/\epsilon))$ times, a quantum implementation of ϕ a number $O(\log(v/\epsilon)\log\log(v/\epsilon))$ of times, and that estimates the mean value of Y up to additive error ϵ, with success probability at least $2/3$.*

Remark 1. The algorithm of Theorem 3 heavily relies on the quantum amplitude estimation algorithm of [17]. The number of qubits is $O(\log(1/\epsilon))$ summed with the number of qubits used by ϕ and $\mathcal{C}$.

Given a full rank lattice $\mathcal{L} \in \mathbb{R}^n$, an element $\boldsymbol{x} \in \mathbb{R}^n$, and a real number $\sigma > 0$, we apply this theorem to estimate $f_{\mathcal{L},\sigma}(\boldsymbol{x})$. Indeed, we saw in Section 2.2 that, by Lemma 4 and 5, for all $\boldsymbol{x} \in \mathbb{R}^n$,

$$f_{\mathcal{L},\sigma}(\boldsymbol{x}) = \sum_{w \in \hat{\mathcal{L}}} D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) e^{2\pi i \langle \boldsymbol{x},\boldsymbol{w}\rangle} = \sum_{w \in \hat{\mathcal{L}}} D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) \cos(2\pi\langle \boldsymbol{x},\boldsymbol{w}\rangle) \ . \quad (4)$$

(the second equality is obtained by regrouping $e^{2\pi i\langle \boldsymbol{x},\boldsymbol{w}\rangle}$ and $e^{2\pi i\langle \boldsymbol{x},-w\rangle}$). We can then bound the variance $\mathcal{V}$ of $\cos(2\pi\langle \boldsymbol{x},\boldsymbol{w}\rangle)$ when $\boldsymbol{w}$ is sampled from $D_{\hat{\mathcal{L}},1/\sigma}$:

$$\begin{aligned}\mathcal{V} &= -\left(\sum_{w \in \hat{\mathcal{L}}} D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) \cos(2\pi\langle \boldsymbol{x},\boldsymbol{w}\rangle)\right)^2 + \sum_{w \in \hat{\mathcal{L}}} D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) \cos(2\pi\langle \boldsymbol{x},\boldsymbol{w}\rangle)^2 \\ &\leqslant \sum_{w \in \hat{\mathcal{L}}} D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) \cos(2\pi\langle \boldsymbol{x},\boldsymbol{w}\rangle)^2 \leqslant \sum_{w \in \hat{\mathcal{L}}} D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) = 1 \ .\end{aligned}$$

Therefore, to estimate $f_{\mathcal{L},\sigma}(\boldsymbol{x})$, we can use Theorem 3 with $v = 1$. We also replace $\hat{\mathcal{L}}$ by a finite subset, ensuring a good approximation by a total variation distance bound. Finally, we use the powering lemma [30] to increase the success probability of the algorithm, so that it can be guaranteed to work for all inputs $\boldsymbol{x}$ simultaneously.

Corollary 1. *Consider a full rank lattice $\mathcal{L} \in \mathbb{R}^n$, a finite set $X \subseteq \mathbb{R}^n$, and real numbers $\epsilon, \sigma > 0$. Let Ω be a finite subset of $\hat{\mathcal{L}}$ such that $d_{TV}(D_{\hat{\mathcal{L}},1/\sigma}, D_{\Omega,1/\sigma}) \leqslant \gamma$ for some $\gamma > 0$. Let $\mathcal{C}$ be a quantum algorithm without measurements that does: $\mathcal{C} : \left|0\right\rangle^{\otimes \ell} \mapsto \left|D_{\Omega,1/\sigma}\right\rangle$ and $\phi : \boldsymbol{w} \mapsto \cos(2\pi\langle \boldsymbol{x},\boldsymbol{w}\rangle)$.*

There exists a quantum algorithm that calls $\mathcal{C}$ a number $\widetilde{O}((1/\epsilon)\log(1/\delta))$ times, calls a quantum implementation of ϕ a number $\widetilde{O}(\log(1/\epsilon)\log(1/\delta))$ of times, and that for all $\boldsymbol{x} \in X$, estimates the value of $f_{\mathcal{L},\sigma}(\boldsymbol{x})$ up to additive error $\epsilon + 2\gamma$, with success probability at least $1 - |X|\delta$. If $\mathcal{C}$ and ϕ use $\mathrm{poly}(\ell, \log(1/\epsilon))$ qubits, then this algorithm uses $\mathrm{poly}(\ell, \log(1/\epsilon))$ qubits.

Proof. Let us first assume that $\boldsymbol{x} \in X$ is fixed. By Theorem 3, the mean value that the algorithm outputs is an estimation Y' of the value

$$\mu(\boldsymbol{x}) := \sum_{w \in \Omega} D_{\Omega,1/\sigma}(\boldsymbol{w}) \cos(2\pi \langle \boldsymbol{x}, \boldsymbol{w} \rangle) \ ,$$

such that with probability at least $2/3$, we have $|Y' - \mu(\boldsymbol{x})| \leqslant \epsilon$.

Next, we use [30, Powering Lemma 6.1] to increase the accuracy of the estimate. Given an error parameter $\delta > 0$, we run the estimation algorithm $O(\log(1/\delta))$ times. Each run is a sample of a random variable Y' such that $|Y' - \mu(\boldsymbol{x})| \leqslant \epsilon$ with probability $2/3$. By taking the median $\widetilde{\mu}(\boldsymbol{x})$ of all samples, we have $|\widetilde{\mu}(\boldsymbol{x}) - \mu(\boldsymbol{x})| \leqslant \epsilon$ with probability at least $1 - \delta$. The probability that this holds for all $\boldsymbol{x} \in X$ simultaneously is at least $(1-\delta)^{|X|} \geqslant 1 - |X|\delta$.

Then, by Equation 4, we have

$$\begin{aligned} f_{\mathcal{L},\sigma}(\boldsymbol{x}) &= \sum_{w \in \hat{\mathcal{L}}} D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) \cos(2\pi \langle \boldsymbol{x}, \boldsymbol{w} \rangle) \\ \implies f_{\mathcal{L},\sigma}(\boldsymbol{x}) - \mu(\boldsymbol{x}) &= \sum_{w \in \hat{\mathcal{L}}} (D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) - D_{\Omega,1/\sigma}(\boldsymbol{w})) \cos(2\pi \langle \boldsymbol{x}, \boldsymbol{w} \rangle) \\ \implies |f_{\mathcal{L},\sigma}(\boldsymbol{x}) - \mu(\boldsymbol{x})| &\leq \sum_{w \in \hat{\mathcal{L}}} \left| D_{\hat{\mathcal{L}},1/\sigma}(\boldsymbol{w}) - D_{\Omega,1/\sigma}(\boldsymbol{w}) \right| \leqslant 2\gamma \ . \end{aligned}$$

By the triangle inequality, we have $|f_{\mathcal{L},\sigma}(\boldsymbol{x}) - \widetilde{\mu}(\boldsymbol{x})| \leqslant \epsilon + 2\gamma$ with probability at least $1 - |X|\delta$, for all $\boldsymbol{x}$. From Theorem 3 and Remark 1, we get the complexities for a single call to the estimation algorithm, which we multiply by $\log(1/\delta)$ (the number of calls). This concludes the proof. □

We note that in the algorithm of Corollary 1, which contains multiple independent runs of Montanaro's algorithm, all measurements can be deferred to the end. Doing this only increases the qubit count by a multiplicative factor $\log(1/\delta)$ (since all runs must be performed coherently).

3 Quantum Gaussian Sampling

In this section we explain how to sample elements from the Gaussian lattice distribution quantumly, by combining a quantum Klein sampler with quantum rejection sampling. Pouly and Shen [42, Section 5.2] also noticed that the Klein sampler could be turned into a quantum algorithm, which they could use after a BKZ preprocessing of the lattice basis – though they did not give any details on the quantum implementation. de Boer and Felderhoff [11] described a detailed quantum version of the Klein sampler (Algorithm 6.1 in [11]), which we reproduce in Section 3.2 for completeness, with slightly different, somewhat simplified complexity estimates for our case. Afterwards, the remainder of this section is dedicated to the rejection sampling step.

Throughout this section, we work with approximate distribution states:

$$|D \pm 2^{-\nu}\rangle := \sum_{x\in X} \alpha_x |x\rangle\,,\ \alpha_x \in \left[\sqrt{D(x)} - 2^{-\nu}, \sqrt{D(x)} + 2^{-\nu}\right]$$

where $|D\rangle = \sum_{x\in X} \sqrt{D(x)}\,|x\rangle$ would have been our wanted state. The total variation distance between the two probability distributions can be bounded as:

$$\frac{1}{2}\sum_{x\in X} \left|(\sqrt{D(x)} \pm 2^{-\nu})^2 - D(x)\right| \leq \frac{3}{2}|X|2^{-\nu}\ .$$

Thus, it suffices to choose $\nu \gg \log_2 |X|$ to obtain a negligible total variation distance with respect to the target distribution.

3.1 Preliminary: Estimating a Gaussian Mass

An important operation, both for the quantum Klein sampler and quantum rejection sampling, is to estimate a Gaussian mass over a finite subset of $\mathbb{Z}$, i.e., for $\widetilde{x} \in \mathbb{R}$:

$$\rho_{\sigma,\widetilde{x}}([\![-2^M, 2^M]\!]) = \sum_{k=-2^M}^{2^M} e^{-\frac{\pi}{\sigma^2}(k-\widetilde{x})^2}\ .$$

For this estimation we can follow Lemma A.30 in [10], which gives the explicit complexity of a method proposed by Kitaev and Webb [31]. In a nutshell, one starts from an approximate quantum implementation of $x \mapsto e^x$ using fixed-point arithmetic, a summation over a smaller interval, and uses the Poisson formula to switch between the cases of a small or a large σ. The only technicality in our result is that we switch from a relative precision to an absolute one. In the rest of the paper, M will always be large enough to satisfy the condition of Corollary 2.

Corollary 2. *There exists a universal constant c such that, for any ν, σ, M with $\nu \geq \log\sigma$ and $\log\sigma + M\log(2) + \nu \leq c2^{2M}$, the following holds.*

There exists a quantum algorithm that has a gate count $\widetilde{O}(\nu^{3/2})$ and a qubit count $\widetilde{O}(\nu)$, and that computes:

$$|\widetilde{x}\rangle\,|0\rangle \mapsto |\widetilde{x}\rangle\,|\rho_{\sigma,\widetilde{x}}([\![-2^M, 2^M]\!]) \pm 2^{-\nu}\rangle\ .$$

Proof. We start by computing an estimation $\widetilde{\rho}$ of $\rho_{\sigma,\widetilde{x}}(\mathbb{Z}) = \sum_{k=-\infty}^{\infty} e^{-\frac{\pi}{\sigma^2}(k-\widetilde{x})^2}$ with relative precision $2^{-\nu'}$, for some ν', using Lemma A.30 in [10]. The algorithm of [10] is unitary, has gate count $\widetilde{O}((\nu')^{3/2})$ and qubit count $\widetilde{O}(\nu')$. We have:

$$\begin{aligned}
|\widetilde{\rho} - \rho_{\sigma,\widetilde{x}}([\![-2^M, 2^M]\!])| &\leq |\widetilde{\rho} - \rho_{\sigma,\widetilde{x}}| + |\rho_{\sigma,\widetilde{x}} - \rho_{\sigma,\widetilde{x}}([\![-2^M, 2^M]\!])| \\
|\widetilde{\rho} - \rho_{\sigma,\widetilde{x}}([\![-2^M, 2^M]\!])| &\leq 2^{-\nu'}\rho_{\sigma,\widetilde{x}} + |\rho_{\sigma,\widetilde{x}} - \rho_{\sigma,\widetilde{x}}([\![-2^M, 2^M]\!])| \text{ By [10]} \\
&\leq 2^{-\nu'}\rho_{\sigma,\widetilde{x}} + O(\sigma \times 2^M e^{-\pi 2^{2M}}) \text{ By Lemma 3} \\
&\leq 2^{-\nu'}(3\sigma + 4) + O(\sigma 2^M e^{-\pi 2^{2M}}) \text{ By Lemma 1 and 2.}
\end{aligned}$$

We take $\nu' = O(\nu)$ and bound the second term thanks to the condition on M, concluding the proof. □

3.2 Step 1: Constructing a Gaussian State

In the following, we give the quantum Klein sampler algorithm from [11].

One-dimensional Gaussian State. The algorithm starts by approximating a one-dimensional quantum Gaussian state, using the method of Kitaev and Webb [31], whose complexity is detailed by de Boer [10]. We adapt Theorem A.29 from de Boer [10], who constructs a one-dimensional centered Gaussian state. In our case we just need to shift the Gaussian by a variable quantity $\widetilde{x}$, which is also made possible by Algorithm 10 in [10].

The input $\widetilde{x}$, which is real, is stored in qubits using a fixed-point approximation with a sufficient precision. This is taken into account by the $O(M+\nu)$ memory complexity. The idea of the algorithm is to produce the Gaussian state bit by bit, recursively. At each bit, one computes a Gaussian sum to determine the left and right weight, then produces a superposition of $|0\rangle$ / $|1\rangle$ with appropriate amplitudes, and uncomputes the sum. The entire algorithm runs without measurements.

Theorem 4 (A.29 in [10], adapted to our notation). *For $M \in \mathbb{N}$, $\nu \in \mathbb{N}$ and σ, there exists a quantum algorithm that computes:*

$$|\widetilde{x}\rangle\,|0\rangle \mapsto |\widetilde{x}\rangle\,|D_{\sigma,\widetilde{x}}([\![-2^M + \lfloor\widetilde{x}\rceil, 2^M + \lfloor\widetilde{x}\rceil]\!])\rangle$$

within trace distance $e^{-2^{2M}/(2\sigma^2)} + M2^{-\nu}$, using $O\big(M\nu^{3/2}\mathrm{polylog}(\nu)\big)$ quantum gates and $O(M+\nu)$ qubits.

Recall that the trace distance between two (pure) states $|\varphi\rangle$ and $|\psi\rangle$ is $\sqrt{1-|\langle\varphi|\psi\rangle|^2}$, and is higher than the total variation distance of the probability distributions resulting from a measurement of the states.

By adapting the precision used inside the algorithm, we can bound the trace distance exactly as $2^{-\nu}$ if the complexity formula is slightly changed. It suffices to choose M large enough compared to σ and ν. This very mild constraint will easily be satisfied in our algorithms.

Corollary 3. *There exists a universal constant c such that, for any σ, for any ν, for any $M \geq c(\log\sigma + \log\nu)$, there exists a quantum algorithm that uses using $O(M+\nu)$ qubits and $\widetilde{O}\big(M\nu^{3/2}\big)$ quantum gates, and that computes:*

$$|\widetilde{x}\rangle\,|0\rangle \mapsto |\widetilde{x}\rangle\,|D_{\sigma,\widetilde{x}}([\![-2^M + \lfloor\widetilde{x}\rceil, 2^M + \lfloor\widetilde{x}\rceil]\!]) \pm 2^{-\nu}\rangle \quad .$$

Proof. We run the algorithm of Theorem 4 using a slightly larger precision $\nu' = O(\nu + \log M)$. Using $M \geq c\log\sigma$ we can ensure that $e^{-2^{2M}/(2\sigma^2)}$ is dominated by $e^{-2^{2M-1}}$. Using ν' we also ensure that the second term $M2^{-\nu'}$ in Theorem 4 is dominated by $2^{-\nu}$. The distance becomes: $e^{-2^{2M-1}} + 2^{-\nu}$. Finally since $M \geq c\log\nu$ the first term can be bounded by $2^{-\nu}$ as well. The asymptotic gate and qubit counts are unchanged, except for the dependency in $\log M$, which is absorbed by the $\widetilde{\mathcal{O}}$ notation. □

Algorithm 2 Quantum Klein sampling (variant of Algorithm 6.1 in [11]).

Input: $B = QR$, σ, $\boldsymbol{c}' = Q^{-1}\boldsymbol{c}$, $M \in \mathbb{N}$, $M > 1$

Output: $|q\rangle$

1: Initialize a quantum state $|\mathbf{0}\rangle = |0\rangle\,|0\rangle \dots |0\rangle$ with m registers
2: **for** $i = m \to 1$ **do**
3: $\quad \sigma_i \leftarrow \sigma / |R_{i,i}|$
4: $\quad$ Let the current state be $\sum |0\rangle\,|0\rangle \dots |0\rangle\, \alpha_{x_{i+1},\dots,x_m}\, |x_{i+1}\rangle \dots |x_m\rangle$
5: $\quad$ Compute:

$$\sum |0\rangle\,|0\rangle \dots |0\rangle\, \alpha_{x_{i+1},\dots,x_m}\, |x_{i+1}\rangle \dots |x_m\rangle\, |\widetilde{x}_i\rangle$$

where $\widetilde{x}_i = \frac{c'_i - \sum_{j=i+1}^{m} R_{i,j} x_j}{R_{i,i}}$

6: $\quad$ Construct a Gaussian state in the register i:

$$\sum |0\rangle\,|0\rangle \dots \Big| D_{\sigma_i, \widetilde{x}_i}(\llbracket -2^M + \lfloor \widetilde{x}_i \rceil, 2^M + \lfloor \widetilde{x}_i \rceil \rrbracket) \Big\rangle\, \alpha_{x_{i+1},\dots,x_m}\, |x_{i+1}\rangle \dots |x_m\rangle$$

7: Multiply by B in-place on the quantum state

Many-dimensional Gaussian State. We now detail the algorithm of [11]. Recall that the Klein distribution q parametrised by σ, $B \in GL_m(\mathbb{R})$, $\boldsymbol{c} \in \mathbb{R}^m$ and $M \in \mathbb{N}$, $M > 1$, is the distribution of lattice vectors obtained by running Algorithm 1 classically. In the quantum setting, Algorithm 2 allows to construct an approximation of it, as a quantum state: $|q \pm 2^{-\nu}\rangle$. The idea is to run the same procedure as classically, where one-dimensional Gaussian samples are replaced by the construction of Gaussian quantum states. Note that we assume that the QR decomposition of B and $\boldsymbol{c}' = Q^{-1}\boldsymbol{c}$ are precomputed classically.

Note that in Algorithm 2, we can compress the space by storing $(x_i - \lfloor \widetilde{x}_i \rfloor)$ in Step 5, which is within $\llbracket -2^M, 2^M \rrbracket$. Then at each loop iterate, we can recompute the center of the previous distribution. We choose to simplify the notations in order to make the algorithm easier to understand.

Theorem 5. *There exists a universal constant c such that the following holds.*

Let q be the Klein distribution resulting from Algorithm 1, which is parametrised by σ, $B \in GL_m(\mathbb{R})$, $\boldsymbol{c}$ and $M \in \mathbb{N}$, $M > 1$. Let $r = \max \mathsf{size}(R_{ij}, c'_i, \sigma)$. Let $\nu \geq c(r + \log m)$ and $M \geq c(\log r + \log \nu + m)$. There exists a quantum circuit of qubit count $\widetilde{O}(m(M + mr + \nu))$ and gate count $\widetilde{O}\big(m^2(mr + M + \nu) + mM\nu^{3/2}\big)$ that prepares $|q \pm 2^{-\nu}\rangle$.

Here size denotes the bit-size required to represent the constants (R_{ij}, c'_i, σ) as rational numbers.

Proof. The quantum circuit this theorem refers to is Algorithm 2. We bound precisely its cost and precision.

Within the algorithm, while x_i are always integers, each $\widetilde{x}_i$ is computed using a fixed-point approximation to precision $\nu' = O(\nu)$. Since each $R_{i,j}$ and c'_i are smaller than 2^r, at step i, the numerator of $\widetilde{x}_i$ can be bounded by: $|c'_i| +$

$2^r(\max_{j\le i}|x_j|)(m-i) \le 2^r + 2^r(m-1)\max_{j\le i}|x_j|$, while the denominator can be bounded by 2^r. Since $|x_m| \le 2^M$, we can unroll this recurrence relation and obtain: $\forall i, 2^r|\widetilde{x_i}| \le (2^r m)^i 2^M + O\big((2^r m)^{i-1}2^r\big) = O\big((2^r m)^i 2^M\big)$. As a consequence, each coordinate x_i requires at most $M + m\log(2^r m) + \nu = \widetilde{O}(M + mr + \nu)$ bits of space to be represented. This means that multiplications required to compute $\widetilde{x_i}$ can be implemented within $\widetilde{O}(M + mr + \nu)$ gates, and there are $O\big(m^2\big)$ of them. All operations in Step 4 therefore cost $\widetilde{O}\big(m^2(mr + M + \nu)\big)$ gates.

Finally, the result of Corollary 3 is used m times, with the different σ_i (using fixed-point approximations again). Each step runs without measurements. The cost of this part is $\widetilde{O}\big(mM\nu^{3/2}\big)$, and the qubit count $O(\nu + M)$ is dominated by the rest of the algorithm.

At the end of Algorithm 2, the amplitudes in the quantum state are the product of m approximations of Gaussian amplitudes. The final error is approximately the sum of m errors. In order to have a final precision ν, we use a precision $\nu' = \nu + \log_2 m$ internally in the Gaussian state creation and arithmetic. Since $\nu \ge c\log m$ we still have $\nu' = O(\nu)$. Thus, we can reach a precision $2^{-\nu}$ with a gate count $\widetilde{O}\big(m^2(mr + M + \nu) + mM\nu^{3/2}\big)$ and a qubit count $\widetilde{O}(m(M + mr + \nu))$. □

This result is similar to Lemma L.5 in [11], as the two algorithms are equivalent (only the order of operations in [11] is performed in a slightly different way, as the quantum state is rescaled between each iteration of the for loop).

3.3 Step 2: Transforming the Distribution

Let Ω be a finite probability space within $\mathcal{L}(B)$. Given a quantum circuit that produces quantum samples for a distribution D_1 over Ω:

$$|0\rangle \xmapsto{\mathcal{A}_1} |D_1\rangle := \sum_{\boldsymbol{x}\in\Omega} \sqrt{D_1(\boldsymbol{x})}\,|\boldsymbol{x}\rangle$$

we use *quantum rejection sampling* [41] to construct quantum samples for another distribution D_2 over Ω, up to some approximation. This section starts with a generic statement (Theorem 6) which is essentially from [41], and continues with the details in our case.

Quantum Rejection Sampling. The main building block from quantum rejection sampling is *amplitude transduction*. We use the method of [48] to implement the following unitary with $O(\nu)$ basic gates:

$$\big|\alpha \pm 2^{-\nu}\big\rangle\,|0\rangle \mapsto \big|\alpha \pm 2^{-\nu}\big\rangle\,\big((\alpha \pm 2^{-\nu})\,|0\rangle + |*\rangle\big) \tag{5}$$

where $|*\rangle$ is a superposition of non-zero basis vectors, and $\alpha \pm 2^{-\nu}$ is a fixed-point encoding, at precision of ν bits, of an amplitude between 0 and 1. We then have the following result.

Theorem 6. *Let $\mathcal{A}_1$ be a quantum circuit (without measurement) that produces an approximation of $|D_1\rangle$: $\mathcal{A}_1|0\rangle = |D_1 \pm 2^{-\nu}\rangle$. Let $w \geq 0$ be such that for all $\boldsymbol{x} \in \Omega$, $D_2(\boldsymbol{x})/D_1(\boldsymbol{x}) \leq w$. Let $\mathcal{B}$ be a quantum circuit that computes a fixed-point approximation of $\sqrt{\frac{1}{w}\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}}$:*

$$\mathcal{B}|\boldsymbol{x}\rangle|0\rangle = |\boldsymbol{x}\rangle\left|\sqrt{\frac{1}{w}\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}} \pm 2^{-\nu}\right\rangle$$

then there exists a quantum algorithm that outputs $\left|D_2 \pm 2^{-\nu+1}\right\rangle$ after making an average $O(\sqrt{w})$ calls to $\mathcal{A}_1$ and $\mathcal{B}$.

Proof. From the output of $\mathcal{A}_1$, we apply $\mathcal{B}$, and obtain:

$$\sum_{\boldsymbol{x}\in\Omega}(\sqrt{D_1(\boldsymbol{x})} \pm 2^{-\nu})|\boldsymbol{x}\rangle\left|\frac{1}{w}\sqrt{\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}} \pm 2^{-\nu}\right\rangle \quad . \tag{6}$$

Since $\sqrt{\frac{1}{w}\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}} \pm 2^{-\nu}$ is a ν-bit fixed-point approximation of a number between 0 and 1, we apply the algorithm of [48] to an ancilla register $|0\rangle$:

$$\begin{aligned}
&\sum_{\boldsymbol{x}\in\Omega}(\sqrt{D_1(\boldsymbol{x})}\pm 2^{-\nu})|\boldsymbol{x}\rangle\left|\sqrt{\frac{1}{w}\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}} \pm 2^{-\nu}\right\rangle\left(\left(\sqrt{\frac{1}{w}\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}} \pm 2^{-\nu}\right)|0\rangle + |*\rangle\right) \\
&\quad = \left(\left(\frac{1}{\sqrt{w}}\sum_{\boldsymbol{x}}\sqrt{D_2(\boldsymbol{x})} \pm 2^{-\nu+1}\right)|\boldsymbol{x}\rangle\left|\sqrt{\frac{1}{w}\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}} \pm 2^{-\nu}\right\rangle\right)|0\rangle + |*\rangle
\end{aligned}$$

We uncompute $\mathcal{B}$. Then, we use quantum amplitude amplification to project on the component $|0\rangle$. The total amplitude over this component being $1/w$, using quantum amplitude amplification we can succeed with $O(w)$ iterations. □

Our Application of Rejection Sampling. We use quantum rejection sampling with the following setup. The initial distribution D_1 is the Klein distribution q. The target distribution D_2 is a Gaussian lattice distribution $D_{\Omega,\sigma}$. The domain Ω is the probability space of the Klein Algorithm (Algorithm 1) parametrised by σ, $\boldsymbol{c} = 0$, and the integer $M > 1$.

We need to determine a bound w on the ratio of the two distributions: $w := \max_{\boldsymbol{x}\in\Omega}\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}$, and we need to be able to compute $\sqrt{\frac{1}{w}\frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}}$ on input $\boldsymbol{x}$ (though it is not necessary to know w itself to run the algorithm). To achieve this, recall that the Gaussian lattice distribution that we are targeting has expression:

$$\forall \boldsymbol{x} \in \Omega, D_{\Omega,\sigma}(\boldsymbol{x}) = \frac{\rho_\sigma(\boldsymbol{x})}{\rho_\sigma(\Omega)} = \frac{e^{-\frac{\pi}{\sigma^2}||\boldsymbol{x}||^2}}{\sum_{\boldsymbol{y}\in\Omega} e^{-\frac{\pi}{\sigma^2}||\boldsymbol{y}||^2}}$$

Following Equation 3, the Klein distribution has expression:

$$\forall \boldsymbol{x} \in B^{-1}\Omega, q(B\boldsymbol{x}) = \frac{\rho_\sigma(B\boldsymbol{x})}{\prod_{i=1}^m \rho_{\sigma_i,\widetilde{x}_i}(\llbracket -2^M + \lfloor\widetilde{x}_i\rceil, 2^M + \lfloor\widetilde{x}_i\rceil\rrbracket)}$$

Given a vector $\boldsymbol{x} \in \mathbb{R}^n$, denote by $\widetilde{x}_{B,i}$ the coefficient $\widetilde{y}_i = \frac{-\sum_{j=i+1}^m R_{i,j} y_j}{R_{i,i}}$, where $\boldsymbol{y} = B^{-1}\boldsymbol{x}$. We rewrite the expression of q as

$$\begin{aligned}\forall \boldsymbol{x} \in \Omega, q(\boldsymbol{x}) &= \frac{\rho_\sigma(\boldsymbol{x})}{\prod_{i=1}^m \rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)} \ . \\ \Rightarrow \frac{D_{\Omega,\sigma}(\boldsymbol{x})}{q(\boldsymbol{x})} &= \frac{\prod_{i=1}^m \rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)}{\rho_\sigma(\Omega)} \ .\end{aligned}$$

According to Lemma 3, we always have

$$\begin{aligned}\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil) &\leqslant C\rho_{\sigma_i,\lfloor \widetilde{x}_{B,i} \rceil}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil) \\ &\leqslant C\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket),\end{aligned}$$

where $C := (1 - 2^M\sqrt{2\pi e} \times e^{-\pi 2^{2M}})^{-1}$. So, for all $\boldsymbol{x} \in \Omega$, we have

$$\frac{D_{\Omega,\sigma}(\boldsymbol{x})}{q(\boldsymbol{x})} \leqslant \frac{C^m \prod_{i=1}^m \rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)}{\rho_\sigma(\Omega)} := w \ .$$

This gives our definition of w. Note that the constant C is easy to compute, and gets closer to 1 as M increases. Computing w itself would be difficult, but it is not necessary. For all $\boldsymbol{x} \in \Omega$, we have:

$$\begin{aligned}\frac{1}{w} \times \frac{D_{\Omega,\sigma}(\boldsymbol{x})}{q(\boldsymbol{x})} &= \frac{\rho_\sigma(\Omega)}{C^m \prod_{i=1}^m \rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)} \times \frac{\prod_{i=1}^m \rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)}{\rho_\sigma(\Omega)} \\ &= \frac{\prod_{i=1}^m \rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)}{C^m \prod_{i=1}^m \rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)} \ .\end{aligned}$$

Let us then explain how to compute this quantity quantumly.

Lemma 7. *Let $r = \max \mathsf{size}(R_{ij}, c'_i, \sigma)$ as in Theorem 5. Assume that the constant C is precomputed classically and exactly, and $\nu \geq r$. There exists an implementation of the quantum circuit $\mathcal{B} : |\boldsymbol{x}\rangle\,|0\rangle \to |\boldsymbol{x}\rangle \left|\sqrt{\frac{1}{w} \times \frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}} \pm 2^{-\nu}\right\rangle$ that has a gate count $\widetilde{O}\big(m^2(mr + M + \nu) + m\nu^2\big)$ and uses $\widetilde{O}(mr + M + \nu)$ qubits.*

Proof. In order to compute $\frac{1}{w} \times \frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}$, we need to estimate $2m$ Gaussian masses, using Corollary 2. Then, we note that, since $\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket) \geqslant 1$, we have

$$\begin{aligned}\frac{1}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket) + 2^{-\nu''}} &= \frac{1 + 2^{-\nu''}/\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket) - 2^{-\nu''}/\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket) + 2^{-\nu''}} \\ &= \frac{1}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)} - \frac{2^{-\nu''}}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)(\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket) + 2^{-\nu''})} \\ &\geqslant \frac{1}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)} - 2^{-\nu''}\end{aligned}$$

and likewise $$\frac{1}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket) - 2^{-\nu''}} \leqslant \frac{1}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)} + 2^{-\nu''} \ .$$

Therefore, assuming $\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)$ and $\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)$ are computed with precision $2^{-\nu''}$, we can compute $\frac{\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)}$ up to precision

$$2^{-\nu''}\left(\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil) + 1/\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)\right) + 2^{-2\nu''} \ .$$

We already know that we have $1/\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket) \leqslant 1$, and by Lemma 1 and 2, we have:

$$\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil) \leqslant \rho_{\sigma_i,\widetilde{x}_{B,i}}(\mathbb{Z}) \leqslant \rho_{\sigma_i}(\mathbb{Z}) \leqslant 3\sigma_i + 4 \ .$$

So we can compute $\frac{\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)}$ up to precision $2^{-\nu''}(3\sigma_i+4)+2^{-2\nu''}$. Setting $\nu'' = \nu' + O(\log(\sigma_i + 1))$ allows to get an additive precision of $2^{-\nu'}$. Finally, since $\sigma_i = \sigma/R_{i,i}$ and since we defined $r \geq \max \mathsf{size}(R_{i,i}, \sigma)$, we can bound $\nu'' = \nu' + O(r)$. By Lemma 3, we have

$$\frac{\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)}{C\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)} \leqslant 1 \ . \tag{7}$$

Therefore, if we can compute each $\frac{\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)}{\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)}$ up to precision $2^{-\nu'}$, then we can compute $\frac{1}{w} \times \frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}$ up to precision $m2^{-\nu'}$. Setting $\nu' = \nu + \log(m)$ allows precision $2^{-\nu}$. This means that $\nu'' = \log m + \nu + O(r)$ is the precision required for estimating the Gaussian masses.

For each i, we then apply transduction separately:

$$|\boldsymbol{x}\rangle\,|0\rangle \mapsto |\boldsymbol{x}\rangle \left(\sqrt{\frac{\rho_{\sigma_i,\widetilde{x}_{B,i}}(\llbracket -2^M, 2^M \rrbracket + \lfloor \widetilde{x}_{B,i} \rceil)}{C\rho_{\sigma_i}(\llbracket -2^M, 2^M \rrbracket)}}\,|0\rangle + |*\rangle\right) \ . \tag{8}$$

The quantity $\frac{1}{w} \times \frac{D_2(\boldsymbol{x})}{D_1(\boldsymbol{x})}$ then appears naturally as the amplitude over the 0-state. The gate and qubit count can be summarized as follows:

- We need to recompute the $\widetilde{x}_{B,i}$, similarly to Algorithm 2. The gate count will be $\widetilde{O}\big(m^2(mr + M + \nu)\big)$ gates. We also need $\widetilde{O}(mr + M + \nu)$ qubits to store the x_i, but this is the input of the algorithm.
- We need to compute $2m$ times a Gaussian mass at precision $\log m + \nu + O(r)$. By Corollary 2, this takes $\widetilde{O}\big(m(\log m + \nu + r)^{3/2}\big) = \widetilde{O}\big(m(\nu + r)^{3/2}\big)$ gates and $\widetilde{O}(\log m + \nu + r)$ qubits.
- We need to compute m times a square root of a number smaller than 1, with additive precision $2^{-\nu}$.

For the latter, a simple method which uses $O(\nu)$ space and $O\big(\nu^2\big)$ gates is to use an iterative integer square root algorithm [33]. Since we assumed that $\nu \geq r$, we can simplify the gate count as follows:

$$\widetilde{O}\Big(m^2(mr + M + \nu) + m(\nu + r)^{3/2} + m\nu^2\Big) = \widetilde{O}\big(m^2(mr + M + \nu) + m\nu^2\big)$$

and the qubit count is $\widetilde{O}(mr + M + \nu)$ (dominated by the storage of x_i). □

Constructing a New Gaussian State. We can now combine the quantum Klein sampler (Theorem 5), and the unitary of Lemma 7 as components of the quantum rejection sampling algorithm (Theorem 6).

Theorem 7. *Consider a lattice $\mathcal{L}$, and Ω a finite subset of $\mathcal{L}$ that corresponds to the probability space of the output distribution of Algorithm 1, parametrised by σ, $\boldsymbol{c} = 0$ and $M \in \mathbb{N}$. Let $r = \max \mathsf{size}(R_{ij}, c'_i, \sigma)$ as in Theorem 5 and assume $M \geq c(\log r + \log \nu)$ for some universal constant c.*

There exists a quantum algorithm of gate count:

$$\sqrt{\frac{C^m \prod_{i=1}^m \rho_{\sigma_i}([\![-2^M, 2^M]\!])}{\rho_\sigma(\Omega)}} \times \widetilde{O}\big(mM(\nu + mM + m^2 r)^2\big)$$

and of qubit count $\widetilde{O}\big(m(\nu + mM + m^2 r)\big)$ that outputs a distribution D such that $d_{TV}(D, D_{\mathcal{L},\sigma}) \leqslant 2^{-\nu}$.

Proof. According to Theorem 6, we need $O(\sqrt{w})$ calls to the algorithms of Theorem 5 and Lemma 7, in order to produce the wanted distribution, where: $w = \frac{C^m \prod_{i=1}^m \rho_{\sigma_i}([\![-2^M, 2^M]\!])}{\rho_\sigma(\Omega)}$ (Section 3.3).

We set the precision of the unitaries to $\nu' = \nu + \log_2(|\Omega|)$, where $\log_2 |\Omega| = m \log_2((2^r m)^m 2^M) = \widetilde{O}\big(mM + m^2 r\big)$ due to the bounding of x_i given previously.

Due to this, the qubit count is: $\widetilde{O}(m(mr + M + \nu')) = \widetilde{O}(m\nu')$. The gate count of the quantum Klein sampler is $\widetilde{O}\big(m^2\nu' + mM(\nu')^{3/2}\big) = \widetilde{O}\big(mM(\nu')^{3/2}\big)$ (observe that ν' is bigger than m^2). The gate count of the circuit $\mathcal{B}$ is given by $\widetilde{O}\big(m^2\nu' + m(\nu')^2\big) = \widetilde{O}\big(m(\nu')^2\big)$. Summing both gives us the gate count for the iterate of the rejection sampling algorithm, which we upper bound by $\widetilde{O}\big(mM(\nu')^2\big)$ for simplicity.

The precision chosen ensures that $|\Omega| 2^{-(\nu + \log_2(|\Omega|))} = 2^{-\nu}$, so $d_{TV}(D, D_{\Omega,\sigma})$ is upper bounded by $2^{-\nu}$. Then, by Equation 2 we have:

$$d_{TV}(D, D_{\mathcal{L},\sigma}) \leqslant d_{TV}(D, D_{\Omega,\sigma}) + d_{TV}(D_{\Omega,\sigma}, D_{\mathcal{L},\sigma}) \leqslant 2^{-\nu} + \rho_\sigma(\mathcal{L} \backslash \Omega) \ .$$

Finally, we use Proposition 1 to bound $d_{TV}(D, D_{\mathcal{L},\sigma})$ as follows:

$$d_{TV}(D, D_{\mathcal{L},\sigma}) \leq 2 \left(2^{M-1}\sqrt{2\pi e}\right)^m \times e^{-\pi \cdot m \cdot 2^{2M-2}} (3\sigma / \min_i \|\widetilde{\boldsymbol{b}}_i\| + 4)^m \qquad (9)$$

Choosing M a multiple of $\log r + \log \nu$ (at least) we can make this smaller than $2^{-\nu}$, finishing the proof. □

3.4 Comparison with Classical Sampling

On a classical computer, the Klein sampler can be used to sample for large values of σ (which depends on the qualitiy of the basis) in polynomial time. For smaller values of σ, the Markov Chain Monte Carlo (MCMC) algorithm [49] allows to

sample at the expense of a larger time complexity. The cost of approximately sampling from $D_{\mathcal{L}(B),\sigma}$ using this algorithm is

$$\mathrm{poly}(m) \times \nu \times \frac{1}{\Delta}, \quad \Delta = \frac{\rho_\sigma(\mathcal{L}(B))}{\prod_{i=1}^m \rho_{\sigma/||\widetilde{\boldsymbol{b}}_i||}(\mathbb{Z})} \ ,$$

where $2^{-\nu}$ indicates the distance to the discrete Gaussian we target. The $\widetilde{\boldsymbol{b}}_i$ are again the columns of the Gram-Schmidt transform of B.

With Theorem 7, we proved that we can sample quantumly from a distribution that can be made arbitrarily close to $D_{\mathcal{L}(B),\sigma}$, in time:

$$\sqrt{\frac{C^m \prod_{i=1}^m \rho_{\sigma_i}([\![-2^M, 2^M]\!])}{\rho_\sigma(\Omega)}} \times \widetilde{O}\big(mM(\nu + mM + m^2 r)^2\big) \, ,$$
$$\text{where } C := \frac{1}{1 - 2^M\sqrt{2\pi e} \times e^{-\pi 2^{2M}}} \ .$$

Suppose that $M = km$ where $k > 1$ is a constant. We can notice that, as m grows, C^m is very close to 1, and therefore has a negligible influence on the complexity:

$$C^m = \left(1 - 2^M\sqrt{2\pi e} \times e^{-\pi 2^{2M}}\right)^{-m} \simeq 1 + m2^M\sqrt{2\pi e} \times e^{-\pi 2^{2M}} = 1 + o(1) \ .$$

Moreover, since $\Omega \supset B\prod_{i=1}^m [\![-2^{M-1}, 2^{M-1}]\!]$, if $M = km$ with a constant $k > 1$, we can reuse Lemma 3 and Proposition 1 to show that:

$$w = \frac{\prod_{i=1}^m \rho_{\sigma_i}([\![-2^M, 2^M]\!])}{\rho_\sigma(\Omega)} \simeq \frac{\prod_{i=1}^m \rho_{\sigma_i}(\mathbb{Z})}{\rho_\sigma(\mathcal{L}(B))} = \frac{1}{\Delta} \ ,$$

up to a polynomial factor. Therefore, if we sample from $D_{\mathcal{L}(B),\sigma}$ using Theorem 7, we roughly gain a quadratic speedup with respect to the MCMC algorithm, since the complexity was $1/\Delta$ (up to polynomial factors in ν, m and $r = \max \mathsf{size}(R_{ij}, c'_i, \sigma)$) and becomes $1/\sqrt{\Delta}$.

4 Application to Dual Attacks on LWE

In this section, we apply our quantum discrete Gaussian sampling algorithm to dual attacks on the LWE problem. Recall from Section 2.3 that the search-LWE problem is to find $\boldsymbol{s} \in \mathbb{Z}_q^n$, given a pair $(A, \boldsymbol{b})$, where A is a matrix in $\mathbb{Z}_q^{m\times n}$, and $\boldsymbol{b} := A\boldsymbol{s} + \boldsymbol{e}$, with $\boldsymbol{e}$ a small error vector in $\mathbb{Z}_q^m$.

The general idea behind dual attacks goes as follows. One splits the matrix A column-wise, and writes $A = [A_{\text{guess}}||A_{\text{dual}}]$. One also writes $\boldsymbol{s} = [\boldsymbol{s}_{\text{guess}}|\boldsymbol{s}_{\text{dual}}]$ accordingly, so that we have $\boldsymbol{b} = A_{\text{guess}}\boldsymbol{s}_{\text{guess}} + A_{\text{dual}}\boldsymbol{s}_{\text{dual}} + \boldsymbol{e}$. Then, one samples short vectors according to a discrete Gaussian distribution, in the dual of the lattice generated by A_{dual}. Finally, one guesses $\boldsymbol{s}_{\text{guess}}$. Whether the guess is correct is checked with the help of the short vectors sampled just before.

For the first step, the typical approach is to use sieving algorithms in order to obtain short vectors in the desired lattice [6,26]. For the second step, the most efficient approach is to use a Fast Fourier Transform (FFT) to speed up the test of candidates for $\boldsymbol{s}_{\text{guess}}$ [28,36,19,24,18]. These two strategies are combined to obtained the current most efficient versions of the attack. While these are classical algorithms, quantum speedups are known for sieving [35,34,20], which is an important building blocks of all attacks on LWE, and quantum dual attacks were studied in [8].

However, Ducas and Pulles [24] pointed out a problem in several dual attacks using sieving algorithms along with the FFT. One of the ground statistical assumptions supporting the correctness of these dual attacks, is false in a lot of cases. This especially affects [28,36], and prompted further scrutiny into the correctness of dual attacks. As a result, Pouly and Shen [42] presented a new statistical analysis to justify that a revisited dual attack works – which they call a "modern dual attack", that allows for more rigorous correctness analysis. They also gave a quantum variant of this modern dual attack.

In this section, we first describe the "modern dual attack", both classical and quantum, of Pouly and Shen [42]. We then propose two variants of this quantum algorithm. The first variant naively plugs the quantum Gaussian sampling algorithm of Section 3 into the first step (sampling short vectors). This improves the time complexity of the entire attack, but similarly to [42], requires qRAM in the second step. The second variant combines the quantum Gaussian sampling algorithm with the quantum algorithm for mean estimation of Corollary 1, directly into the second step. This algorithm improves over the classical complexity of [42] without using qRAM.

4.1 The Modern Dual Attack on LWE

Here we present the idea of the "modern dual attack" of Pouly and Shen [42], first the classical algorithm (Algorithm 3), then its quantum version.

The Classical Version and its Complexity. Assume that $(A, \boldsymbol{b})$ is sampled from $LWE(m, \boldsymbol{s}, \chi_e)$. The secret $\boldsymbol{s}$ is in $\mathbb{Z}_q^n$. Let us write $n = n_{\text{guess}} + n_{\text{dual}}$. We split $\boldsymbol{s}$ into two parts $\boldsymbol{s}_{\text{guess}} \in \mathbb{Z}_q^{n_{\text{guess}}}$ and $\boldsymbol{s}_{\text{dual}} \in \mathbb{Z}_q^{n_{\text{dual}}}$, and we also split $A := [A_{\text{guess}} \| A_{\text{dual}}]$ accordingly. Now, we have $\boldsymbol{b} = A_{\text{guess}}\boldsymbol{s}_{\text{guess}} + A_{\text{dual}}\boldsymbol{s}_{\text{dual}} + \boldsymbol{e}$. We guess the coefficients of $\boldsymbol{s}_{\text{guess}}$. Let us write this guess $\overline{\boldsymbol{s}}_{\text{guess}}$ to avoid confusion. The intuition that underlies the attack is that the good $\boldsymbol{s}_{\text{guess}}$ minimizes $\text{dist}(\boldsymbol{b} - A_{\text{guess}}\overline{\boldsymbol{s}}_{\text{guess}}, \mathcal{L}_q(A_{\text{dual}}))$. The goal is therefore to estimate this distance for all $\overline{\boldsymbol{s}}_{\text{guess}}$ with sufficient precision to spot the best candidate. Fortunately, the periodic Gaussian function $f_{\mathcal{L}_q(A_{\text{dual}}),1/\sigma}(\boldsymbol{y})$, defined in Equation 1, can serve as a measure of the distance of a point $\boldsymbol{y}$ from the lattice $\mathcal{L}_q(A_{\text{dual}})$: closer points will have a larger value. More precisely, we have

Lemma 8. *[[42, Lemma 8]] Consider $\mathcal{L} \subset \mathbb{R}^m$ a lattice and $\sigma > 0$. Then, for any $\boldsymbol{x} \in \mathbb{R}^m$*

Algorithm 3 Classical modern dual attack [42, Algorithm 2]

Input: Integers m, $n = n_{\text{guess}} + n_{\text{dual}}$, q, $N \in \mathbb{N}^*$. A LWE sample $(A, \boldsymbol{b})$, a list $W = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\}$ of vectors sampled from $D_{\mathcal{L}_q^\perp(A_{\text{dual}}),q\sigma}$

Output: A guess of n_{guess} coordinates of $\boldsymbol{s}$, or $\perp$

$\boldsymbol{s}_{\text{guess}} \leftarrow \perp$
$S_{max} \leftarrow 0$
for $\overline{\boldsymbol{s}}_{\text{guess}} \in \mathbb{Z}_q^{n_{\text{guess}}}$ **do**
  Compute each $y_1, \ldots, y_N$, where $y_j := \boldsymbol{w}_j^T(b - A_{\text{guess}}\overline{\boldsymbol{s}}_{\text{guess}})$
  $S \leftarrow \sum_{j=1}^N \cos(2\pi y_j/q)$
  if $S \geqslant S_{max}$ **then**
    $S_{max} \leftarrow S$
    $\boldsymbol{s}_{\text{guess}} \leftarrow \overline{\boldsymbol{s}}_{\text{guess}}$
Return $\boldsymbol{s}_{\text{guess}}$

- $f_{\mathcal{L},1/\sigma}(\boldsymbol{x}) \geqslant \rho_{1/\sigma}(\text{dist}(\boldsymbol{x}, \mathcal{L}))$,
- *if* $\text{dist}(\boldsymbol{x}, \mathcal{L}) \geqslant \tau := \frac{1}{\sigma}\sqrt{m/2\pi}$, *then* $f_{\mathcal{L},1/\sigma}(\boldsymbol{x}) \leqslant \rho_{1/\sigma}(\text{dist}(\boldsymbol{x}, \mathcal{L}) - \tau)$.

which can be loosely interpreted as:

$$f_{\mathcal{L}_q(A_{\text{dual}}),1/\sigma}(\boldsymbol{y}) \simeq \rho_{1/\sigma}(\text{dist}(\boldsymbol{y}, \mathcal{L}_q(A_{\text{dual}}))) \ .$$

To approximate this function, we build a set $W = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\}$ of vectors sampled from $D_{\mathcal{L}_q^\perp(A_{\text{dual}}),q\sigma}$. Then, we compute

$$g_W(\boldsymbol{y}) := \frac{1}{N}\sum_{j=1}^N \cos(2\pi\langle \boldsymbol{y}, \boldsymbol{w}_j\rangle/q), \ \boldsymbol{y} := \boldsymbol{b} - A_{\text{guess}}\overline{\boldsymbol{s}}_{\text{guess}} \ .$$

The function g_W is close to the function $f_{\mathcal{L}_q(A_{\text{dual}}),1/\sigma}$ (Lemma 6). This gives the following algorithm (note that a Discrete Fourier Transform is used to accelerate the computation of scores S).

Theorem 8 ([42, Theorem 6]). *Let $A \in \mathbb{Z}_q^{m\times n}$, $\boldsymbol{e} \in \mathbb{Z}^m$, $\boldsymbol{s} \in \mathbb{Z}_q^n$, $\sigma, \delta > 0$, and $N \in \mathbb{N}$. Let $\tau := \frac{1}{\sigma}\sqrt{m/2\pi}$. Assume that $m \geqslant n$, A has full rank, $\lambda_1(\mathcal{L}_q(A)) \geqslant \tau + \|e\|$, and*

$$\rho_{1/\sigma}(\boldsymbol{e}) - \rho_{1/\sigma}(\lambda_1(\mathcal{L}_q(A)) - \|e\| - \tau) > 2\delta.$$

Let $\boldsymbol{b} = A\boldsymbol{s}+\boldsymbol{e} \mod q$. Let $W = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\}$ be vectors sampled from $D_{\mathcal{L}_q^\perp(A_{\text{dual}}),q\sigma}$. Then, Algorithm 3 with inputs $(m, n = n_{\text{guess}} + n_{\text{dual}}, q, N, (A, \boldsymbol{b}), W)$ runs in time

$$\text{poly}(m, n) \times (N + q^{n_{\text{guess}}})$$

and returns $\boldsymbol{s}_{\text{guess}}$ with probability at least $1 - q^m \times 2^{-\Omega(N\delta^2)}$ over the choice of W.

Algorithm 4 Quantum modern dual attack

Input: Integers m, $n = n_{\text{guess}} + n_{\text{dual}}$, q, $N \in \mathbb{N}^*$. A real number $\eta > 0$. A LWE sample $(A, \boldsymbol{b})$, an oracle O_W for a set $W = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\}$ of vectors sampled from $D_{\mathcal{L}_q^\perp(A_{\text{dual}}),q\sigma}$

Output: A guess of n_{guess} coordinates of $\boldsymbol{s}$, of nothing ($\perp$).

1: Set $\mathcal{A}$ as in planned by Theorem 9, with $\delta = 1/10$, $\epsilon = \eta$, and q
2: $|s\rangle \leftarrow \sum_{\overline{\boldsymbol{s}}_{\text{guess}} \in \mathbb{Z}_q^{n_{\text{guess}}}} |\overline{\boldsymbol{s}}_{\text{guess}}\rangle \, |\mathcal{A}^{O_W}(\boldsymbol{b} - A_{\text{guess}}\overline{\boldsymbol{s}}_{\text{guess}})\rangle$
3: Run the algorithm of Theorem 2 to find $\overline{\boldsymbol{s}}_{\text{guess}}$ such that $\mathcal{A}^{O_W}(\boldsymbol{b} - A_{\text{guess}}\overline{\boldsymbol{s}}_{\text{guess}})$ is maximal
4: **Return** $\boldsymbol{s}_{\text{guess}} = \overline{\boldsymbol{s}}_{\text{guess}}$

The Quantum Version and its Complexity. The algorithm presented above is an exhaustive search over the set of possible $\overline{\boldsymbol{s}}_{\text{guess}}$, where we look for $\overline{\boldsymbol{s}}_{\text{guess}}$ maximizing the score S. The quantum version uses two tools. The first is an approximation of the score S (Theorem 9 below).

Theorem 9 ([8, Theorem 5],[42, Theorem 9]). *Let $N \in \mathbb{N}^*$, and let $W = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\} \subset \mathbb{Z}^m$. Let $g_W(\boldsymbol{y}) := \frac{1}{N}\sum_{i=1}^{N} \cos(2\pi\langle \boldsymbol{w}_i, \boldsymbol{y}\rangle/q)$. Define O_W to be an oracle such that*

$$O_W : |j\rangle\,|0\rangle \to |j\rangle\,|\boldsymbol{w}_j\rangle\,. \tag{10}$$

For any $\epsilon, \delta > 0$, there exists a quantum algorithm $\mathcal{A}$ that, given $\boldsymbol{y}$ and oracle access to O_W, outputs $\mathcal{A}^{O_W}(\boldsymbol{y})$ such that $|\mathcal{A}^{O_W}(\boldsymbol{y}) - g_W(\boldsymbol{y})| \leqslant \epsilon$, with probability $1 - \delta$. The algorithm makes $O(\epsilon^{-1}\log(1/\delta))$ queries to O_W, and requires $O(\log(1/\epsilon) + \text{poly}(\log(m)))$ qubits.

The second tool is a quantum maximum-finding algorithm that is robust to errors (since the computation of S makes such errors). For this purpose, we can directly use Theorem 2 (generalization of [42, Theorem 8]) with a constant probability of error. The combination of these two tools gives Algorithm 4.

Theorem 10 ([42, Theorem 9]). *Let $A \in \mathbb{Z}_q^{m\times n}$, $\boldsymbol{e} \in \mathbb{Z}^m$, $\boldsymbol{s} \in \mathbb{Z}_q^n$, $\sigma, \delta, \eta > 0$, and $N \in \mathbb{N}$. Let $\tau := \frac{1}{\sigma}\sqrt{m/2\pi}$. Assume that $m \geqslant n$, A has full rank, $\lambda_1(\mathcal{L}_q(A)) \geqslant \tau + ||e||$, and*

$$\rho_{1/\sigma}(\boldsymbol{e}) - \rho_{1/\sigma}(\lambda_1(\mathcal{L}_q(A)) - ||e|| - \tau) > 2\delta + \eta.$$

Let $\boldsymbol{b} = A\boldsymbol{s}+\boldsymbol{e} \mod q$. Let $W = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\}$ be vectors sampled from $D_{\mathcal{L}_q^\perp(A_{\text{dual}}),q\sigma}$, and let O_W be an oracle for W as defined in Equation 10. Then, Algorithm 4 with inputs $(m, n = n_{\text{guess}} + n_{\text{dual}}, q, N, \eta/2, (A, \boldsymbol{b}), W, O_W)$ makes $O(\eta^{-1} \times q^{n_{\text{guess}}/2})$ calls to O_W. It has a qubit cost in $O(\log(\eta^{-1}) + \text{poly}(\log(N)))$ and returns $\boldsymbol{s}_{\text{guess}}$ with probability at least $1 - q^m \times 2^{-\Omega(N\delta^2)}$.

The cost of sampling the necessary $\boldsymbol{w}_j$ should also be included in the cost of the attack. Pouly and Shen [42] proposed to do so by computing a BKZ-reduced

basis B of $\mathcal{L}_q^\perp(A_{\text{dual}})$, and using the Markov Chain Monte Carlo (MCMC) algorithm N times to sample N vectors in $\mathcal{L}_q^\perp(A_{\text{dual}})$ [49]. As recalled in Section 3.4, the cost of the MCMC algorithm is

$$N \times \text{poly}(n) \times \nu \times \frac{1}{\Delta}, \quad \Delta = \frac{\rho_{q\sigma}(\mathcal{L}_q^\perp(A_{\text{dual}}))}{\prod_{i=1}^m \rho_{q\sigma/||\tilde{\boldsymbol{b}}_i||}(\mathbb{Z})} \ ,$$

where $2^{-\nu}$ is the distance between the distribution $D_{\mathcal{L}_q^\perp(A_{\text{dual}}),q\sigma}$, and the one we actually sample from using the MCMC algorithm.

Pouly and Shen show ([42, Section 4.3]) that one can take $\eta = \delta/100$ and $\eta^{-1} = \text{poly}(\log m))\sqrt{N}$. They propose two methods for the quantum attack.

- Without qRAM: preprocessing the basis with BKZ and directly using the sampler in the algorithm O_W (instead of using precomputed data). The preprocessing step admits a quantum speedup[2]. We denotes its time by $T_{\text{Q-BKZ}}$. The quantum time complexity of the attack becomes:

$$T_{\text{Q-BKZ}} + \frac{1}{\Delta}\sqrt{N}q^{n_{\text{guess}}/2}\text{poly}(\log m) \ . \tag{11}$$

  They note that there is no quantum speedup on the sampler (whether one uses Klein's algorithm directly or MCMC).
- With qRAM: using the same sampler to precompute the $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N$, they need to be stored in classical memory with quantum random access (QRACM) in order to implement the O_W algorithm efficiently. The quantum time complexity becomes:

$$T_{\text{Q-BKZ}} + \frac{1}{\Delta}N\text{poly}(\log m) + \sqrt{N}q^{n_{\text{guess}}/2}\text{poly}(\log m) \ . \tag{12}$$

4.2 Improvement with qRAM

A first simple way to improve the quantum modern dual attack of [42], is to sample the N vectors in $\mathcal{L}_q^\perp(A_{\text{dual}})$ using the algorithm of Theorem 7, instead of the MCMC algorithm. As we have shown in Section 3.3, we gain a quadratic speedup on the MCMC sampler (up to polynomial factors). As a consequence, the quantum time complexity becomes:

$$T_{\text{Q-BKZ}} + \frac{1}{\sqrt{\Delta}}N\text{poly}(\log m) + \sqrt{N}q^{n_{\text{guess}}/2}\text{poly}(\log m) \ , \tag{13}$$

which is always better than Equation 12. The time complexity of the second step (maximum-finding) remains unchanged, as well as the quantum memory complexity N.

We do not provide concrete numbers for the attack on the Kyber cryptosystem because our analysis, which follows that of [42] does not include modulus switching which is required to obtain a competitive complexity.

[2] BKZ relies on solving the shortest vector problem (SVP) in smaller dimensions. This can be done either by sieving or enumeration, both of which admit a quantum speedup [9,13]. In most crytographic applications, lattice sieving is always faster than enumeration.

Algorithm 5 New quantum modern dual attack

Input: Integers m, $M > 1$, $n = n_{\text{guess}} + n_{dual}$, q. Real numbers $\epsilon, \eta_1, \eta_2 > 0$, $\sigma > 0$. A LWE sample $(A, \boldsymbol{b})$.

Output: A guess of n_{guess} coordinates of $\boldsymbol{s}$.

1: Set ϕ as the quantum algorithm : $|\boldsymbol{x}\rangle|\boldsymbol{w}\rangle \to |\cos(2\pi\langle \boldsymbol{x}, \boldsymbol{w}\rangle/q)\rangle$
2: Set $\mathcal{GS}$ as the sampling algorithm resulting from Theorem 7, which performs: $|0\rangle \mapsto |D\rangle$ where D is a distribution close to $D_{\Omega,q\sigma}$
3: Set $\mathcal{A}^{\mathcal{GS},\phi}$ the mean estimation algorithm of Corollary 1 with parameters ϵ and $\delta = 1/(20 \times q^m)$ which performs: $|\boldsymbol{x}\rangle|0\rangle \mapsto |\boldsymbol{x}\rangle|\mathcal{A}^{\mathcal{GS},\phi}(\boldsymbol{x})\rangle$ with ϕ the function $\boldsymbol{w} \to \cos(2\pi\langle \boldsymbol{x}, \boldsymbol{w}\rangle)$ and $\mathcal{C}$ the circuit $\mathcal{GS}$
4: Run the algorithm of Theorem 2 to find $\overline{s}_{guess}$ such that $\mathcal{A}^{\mathcal{GS},\phi}(\boldsymbol{b} - A_{\text{guess}}\overline{s}_{guess})$ is maximal
5: **Return** $\boldsymbol{s}_{\text{guess}} = \overline{s}_{guess}$

4.3 Improvement without qRAM

Denote by B a basis of the lattice $\mathcal{L}_q^{\perp}(A_{\text{dual}})$, as before, and a finite set $\Omega \subset \mathcal{L}(B)$. Now, instead of classically sampling N short vectors according to a distribution $D_{\mathcal{L}_q^{\perp}(A_{\text{dual}}),q\sigma}$, as in [42], we use the algorithm of Theorem 7, along with Corollary 1, to estimate the value $f_{\hat{\mathcal{L}}_q(A_{\text{dual}}),1/q\sigma}$. The resulting dual attack on LWE is presented in Algorithm 5. Recall that our approximate distribution $D_{\Omega,q\sigma}$ replaces the distribution $D_{\mathcal{L}_q^{\perp}(A_{\text{dual}}),q\sigma}$ (Lemma 8).

Lemma 9. *Let $B \in \mathbb{Z}_q^{m\times n}$, $\mathcal{L} \subset \mathbb{R}^m$ a lattice, $\boldsymbol{e} \in \mathbb{Z}^m$, $\epsilon, \sigma, \delta, \eta_1 > 0$, and $N \in \mathbb{N}$. Let $\tau := \frac{1}{\sigma}\sqrt{m/2\pi}$, $\eta_2 \geqslant 0$, and assume that $\lambda_1(\mathcal{L} + \mathcal{L}_q(B)) \geqslant \tau + ||\boldsymbol{e}||$, and*

$$\rho_{1/\sigma}(\boldsymbol{e}) - \rho_{1/\sigma}(\lambda_1(\mathcal{L} + \mathcal{L}_q(B)) - ||\boldsymbol{e}|| - \tau) > 2\epsilon + 2\eta_1 + \eta_2.$$

Let D be a probability distribution over a finite subset Ω of $\mathcal{L}_q^{\perp}(B)$, such that $d_{TV}(D, D_{\mathcal{L}_q^{\perp}(B),q\sigma}) \leqslant \eta_1/2$. Let $\mathcal{GS}$ be a quantum circuit such that $\mathcal{GS} : |0\rangle \to |D\rangle$, and let $\phi : |\boldsymbol{x}\rangle|\boldsymbol{w}\rangle \to |\cos(2\pi\langle \boldsymbol{x}, \boldsymbol{w}\rangle/q)\rangle$. Denote by $\widetilde{f}(\boldsymbol{x})$ the output of $\mathcal{A}^{\mathcal{GS},\phi}$ on input $\boldsymbol{x}$ and with parameters ϵ and δ.

Then, with probability at least $1 - q^m\delta$, for all $\boldsymbol{x} \in \mathcal{L}\backslash\mathcal{L}_q(B)$, we have

$$|\widetilde{f}(\boldsymbol{x}) - f_{\mathcal{L}_q(B),1/\sigma}(\boldsymbol{x})| \leqslant \epsilon + \eta_1,$$

$$\widetilde{f}(\boldsymbol{e}) \geqslant \rho_{1/\sigma}(\boldsymbol{e}) - \epsilon - \eta_1 > \rho_{1/\sigma}(\lambda_1(\mathcal{L} + \mathcal{L}_q(B)) - ||\boldsymbol{e}|| - \tau) + \epsilon + \eta_1 + \eta_2 \geqslant \widetilde{f}(\boldsymbol{e} + \boldsymbol{x}) + \eta_2 \ .$$

The proof is given in Section B.

We now have the necessary tools to prove the following theorem.

Theorem 11. *Consider a matrix $A \in \mathbb{Z}_q^{m\times n}$, vectors $\boldsymbol{e} \in \mathbb{Z}^m$, $\boldsymbol{s} \in \mathbb{Z}_q^n$, real numbers $\epsilon, \sigma, \eta_1, \eta_2 > 0$, and $\delta = 1/(20 \times q^m)$. Let $\tau := \frac{1}{\sigma}\sqrt{m/2\pi}$ and denote by $\mathcal{L}$ the lattice $\mathcal{L}_q(A)$. Assume that $m \geqslant n$, A has full rank, $\lambda_1(\mathcal{L}_q(A)) \geqslant \tau + ||e||$, and*

$$\rho_{1/\sigma}(\boldsymbol{e}) - \rho_{1/\sigma}(\lambda_1(\mathcal{L}_q(A)) - ||e|| - \tau) > 2\epsilon + 2\eta_1 + \eta_2.$$

Let $\boldsymbol{b} = A\boldsymbol{s}+\boldsymbol{e} \mod q$. Let D be a probability distribution over a finite subset Ω of $\mathcal{L}_q^\perp(A_{\text{dual}})$, such that $d_{TV}(D, D_{\mathcal{L}_q^\perp(A_{\text{dual}}),q\sigma}) \leqslant \eta_1/2$. Let $\mathcal{GS}$ and ϕ be the quantum circuits as defined in Algorithm 5.

Then, Algorithm 5 with inputs $(m, M, n = n_{\text{guess}} + n_{dual}, q, \epsilon, \eta_1, \eta_2, \sigma, (A, \boldsymbol{b}))$ makes

$$O\left(\text{poly}(\log q, m, \log(1/\epsilon))\sqrt{q^{n_{\text{guess}}}}\right),$$

calls to ϕ, and

$$O\left(\text{poly}(\log q, m, \log(1/\epsilon))\sqrt{q^{n_{\text{guess}}}} \times \frac{1}{\epsilon}\right)$$

calls to $\mathcal{GS}$. It returns $\boldsymbol{s}_{\text{guess}}$ with probability at least $15/16$. It has an additional qubit count in $O(\log(1/\epsilon)) + \text{poly}(\log q, m)$.

Proof. **Correctness.** Recall that $\widetilde{f}(\boldsymbol{x})$ is the output of the probabilistic algorithm $\mathcal{A}^{\mathcal{GS},\phi}$ on input $\boldsymbol{x}$ and with parameters ϵ and δ. This output is an estimation of $f_{\mathcal{L}_q(A_{\text{dual}}),1/q\sigma}(\boldsymbol{x})$. Assume that $\delta = 1/(20\times q^m)$. In this case, by Lemma 9, with probability at least $1 - 1/20$, we have for all $\boldsymbol{x} \in \mathcal{L}\backslash\mathcal{L}_q(A_{\text{dual}})$,

$$\widetilde{f}(\boldsymbol{e}) > \widetilde{f}(\boldsymbol{e}+\boldsymbol{x}) + \eta_2. \tag{14}$$

We say that in this case, $\mathcal{A}^{\mathcal{GS},\phi}$ returns a correct value, and otherwise it returns a faulty value. We can then apply Theorem 2 to say that, with probability at least $15/16$, the output $\overline{s}'_{guess}$ of Algorithm 5 maximizes $\widetilde{f}(\boldsymbol{b} - A_{\text{guess}}\overline{s}_{guess})$.

Let us now show that, in this case, $\overline{s}'_{guess} = \boldsymbol{s}_{\text{guess}}$. Assume that this is not true, for the sake of contradiction.

Note that A has full rank, and $m \geqslant n$, so the columns of A are linearly independent over $\mathbb{Z}_q$ and

$$\mathcal{L}\backslash\mathcal{L}_q(A_{\text{dual}}) = \mathcal{L}_q(A_{\text{guess}})\backslash\mathcal{L}_q(A_{\text{dual}}) = \mathcal{L}_q(A_{\text{guess}})\backslash q\mathbb{Z}^m.$$

Moreover, since $\boldsymbol{b} = A\boldsymbol{s} + \boldsymbol{e}$, we have

$$\boldsymbol{b} - A_{\text{guess}}\overline{s}'_{guess} = A_{\text{guess}}(\boldsymbol{s}_{\text{guess}} - \overline{s}'_{guess}) + \boldsymbol{e} \ .$$

So, if $\overline{s}'_{guess} \neq \boldsymbol{s}_{\text{guess}}$, there is some $\boldsymbol{x} \in \mathcal{L}_q(A_{\text{guess}})$ such that $\boldsymbol{b} - A_{\text{guess}}\overline{s}'_{guess} = \boldsymbol{e} + \boldsymbol{x}$. Here, Lemma 9 applies. Namely, with probability $1 - q^m\delta$, we have

$$\widetilde{f}(\boldsymbol{e}) > \widetilde{f}(\boldsymbol{e}+\boldsymbol{x}) + \eta_2. \tag{15}$$

This is in contradiction with the fact that $\overline{s}'_{guess}$ maximizes $\widetilde{f}(\boldsymbol{b} - A_{\text{guess}}\overline{s}_{guess})$. Therefore, $\overline{s}'_{guess} = \boldsymbol{s}_{\text{guess}}$.

Performances. Computing $\boldsymbol{b} - A_{\text{guess}}\overline{s}_{guess}$ on a quantum computer involves products of $m \times n_{\text{guess}}$ dimensional matrices with integers modulo q, so this can be done with a gate and qubit count in $\text{poly}(\log q, m)$.

By Corollary 1, and since $\delta = 1/(20 \times q^m)$, the operator $\mathcal{A}^{\mathcal{GS},\phi}$ calls ϕ a number $\widetilde{O}\big(m\log(q)\log\left(\frac{1}{\epsilon}\right)\big)$ of times. Likewise, it calls $\mathcal{GS}$ a number of times in $\widetilde{O}\big(m\log(q)\frac{1}{\epsilon}\big)$. The qubit count is in $O(\log(1/\epsilon) + m\log(q))$.

By Theorem 2, to find the $\overline{s}_{guess}$ that maximises $\widetilde{f}(\boldsymbol{b} - A_{\text{guess}}\overline{s}_{guess})$, we need to call $\mathcal{A}^{\mathcal{GS},\phi}$ a number of times in $O(n_{\text{guess}}\log(q)\sqrt{q^{n_{\text{guess}}}})$, and to use $O(n_{\text{guess}}\log(q)\sqrt{q^{n_{\text{guess}}}})$ additional gates, and $\text{poly}(n_{\text{guess}}\log(q))$ qubits. By multiplying this (and using $n_{\text{guess}} < m$) we obtain the total number of calls to $\mathcal{GS}$ and ϕ, with an additional gate count accounting for a polynomial factor. □

Concrete Complexity and Comparison. Similarly to [42, Section 4.3], we can take a concrete value for ϵ and express the complexity in terms of the parameter N of the previous attack. Indeed, one should choose ϵ to be smaller than the smallest possible value of $\rho_{1/\sigma}(\boldsymbol{e})$, and this corresponds (up to a polynomial factor) to $\epsilon = \frac{1}{\sqrt{N}}$ where N is the amount of samples required in the first variant of the attack.

Therefore, by combining our new sampling with Theorem 11 we obtain a complexity:

$$T_{\text{Q-BKZ}} + \frac{1}{\sqrt{\Delta}}\sqrt{N}q^{n_{\text{guess}}/2}\text{poly}(m, \log q) \ , \tag{16}$$

which improves by a factor $\sqrt{\Delta}$ over Equation 11.

Comparison with sieving-based dual attacks. The main advantage of this improvement is that it does not require a qRAM and only a polynomial number of qubits. This should be compared to dual attacks based on sieving which, although more efficient in the classical setting[3], would require very large qRAMs in the quantum setting.

5 Quantum Algorithm for solving SIS in any norm

In [12], the authors gave a simple algorithm that solves the SIS problem for any norm. The generic framework of the algorithm is the following: given access to a discrete Gaussian sampler, they estimate the probability p for the sampler to produce a short enough vector that corresponds to the solution of the SIS problem, using averaging results on q-ary lattices [42]. They then sample enough vectors so that the probability that one of them is the solution of the SIS problem is constant. The complexity of the algorithm is then $O(1/p \cdot \text{T}_{\text{sampler}})$ where $\text{T}_{\text{sampler}}$ is the complexity of producing one sample. Note that this ignores any potential preprocessing required for the sampler, such as reducing the basis.

The discrete Gaussian sampler that they used was precisely the MCMC sampler in [49] for which we have a quantum quadratic speedup (see Section 3.4). Note that the overall attack starts by reducing the basis using the BKZ algorithm. This preprocessing step also admits a quantum speedup, albeit far from a quadratic one when we use lattice sieving for solving the shortest vector problem in smaller dimensions [13].

[3] Sieving based dual attacks are more efficient but require heuristic assumptions [18], whereas the "modern" dual attack in [42] using a discrete gaussian sampler is provably correct.

In our quantum algorithm, we constructed a quantum state $\sum_{\boldsymbol{x}\in\Omega}\sqrt{D(\boldsymbol{x})}\,|\boldsymbol{x}\rangle$ where D is (very close to) the discrete Gaussian over the lattice L of interest, and Ω is a large domain. From this state and given ℓ, the length parameter of the SIS problem, we can easily build

$$|\varphi\rangle := \sum_{\boldsymbol{x}\in\Omega}\sqrt{D(\boldsymbol{x})}\,|\boldsymbol{x}\rangle\,|f(\boldsymbol{x})\rangle\,.$$

where f is the indicator function of the ℓ_p-ball of radius ℓ minus the zero vector. The crux of the analysis in [12] lies in the analysis of the probability p that a vector $\boldsymbol{x}$ sampled from D falls into this ball. In other words,

$$p = \sum_{\boldsymbol{x}\in\Omega} D(\boldsymbol{x})f(\boldsymbol{x}). \tag{17}$$

By doing an amplitude amplification on $|\varphi\rangle$, we can keep only[4] the vectors $\boldsymbol{x}$ such that $f(\boldsymbol{x}) = 1$. The remaining state will exactly contain all vectors in the ball minus the zero vector, so we can measure it and obtain a solution to the SIS problem. The complexity of this quantum algorithm is therefore $O(\mathrm{T}_{\text{Q-sample}}\mathrm{T}_{\text{QAA}})$ where $\mathrm{T}_{\text{Q-sample}}$ is the time complexity of the quantum sampler and T_{QAA} the time of the quantum amplitude amplification. We have shown in Section 3.4 that $\mathrm{T}_{\text{Q-sample}} = \mathrm{poly}(m)\sqrt{\mathrm{T}_{\text{sample}}}$. Finally, by (17), we have $\mathrm{T}_{\text{QAA}} = O(\frac{1}{\sqrt{p}})$. Hence, the overall complexity of our quantum algorithm is

$$T_{\text{attack}} = \mathrm{T}_{\text{Q-BKZ}} + \mathrm{poly}(m)\sqrt{\frac{\mathrm{T}_{\text{sample}}}{p}}\,.$$

In Table 1, we show the estimated classical time cost of the attack on SIS$^\infty$ for the Dilithium signature scheme [22] given in [12]. We modified the estimator of [12] to use the quantum complexity above (neglecting polynomial factors). It uses [43] to estimate the complexity of the sampler for a BKZ reduced basis and optimizes the choice of β, the BKZ block size, and s to find the optimal set of parameters. We applied this modified estimator to the parameters of SIS$^\infty$ for Dilithium. The results can be found in Table 2. We can see that, while the complexity is much better than the classical attack, it remains limited by the BKZ step. It should also be noted that the method for solving SIS$^\infty$ in [12] is provable but still quite far from the heuristic estimates underlying the security of Dilithium.

[4] Technically, we only obtain a state very close to the wanted one which means that the final measurement only gives a solution with constant probability.

Table 1. Estimated classical time cost of the attack on SIS^∞ for Dilithium in [12]. The T_{sample} column corresponds to the cost of the MCMC sampler (to produce one sample) and T_{BKZ} is the cost of reducing the basis. N is the number of Gaussian samples used by the attack. Columns β and s give the parameters of the attacks (BKZ block size and width of the Gaussian sampler).

Scheme	$\log_2 T_{\mathrm{attack}}$	β	s	$\log_2 N$	$\log_2 T_{\mathrm{sample}}$	$\log_2 T_{\mathrm{BKZ}}$
NIST Level 2	202	609	346702	39	159	201
NIST Level 3	289	919	723749	55	230	288
NIST Level 5	400	1314	769533	74	323	400

Table 2. Estimated quantum time cost of our attack on SIS^∞ for Dilithium. $T_{\text{Q-sample}}$ is the cost of our quantum sampler and $T_{\text{Q-BKZ}}$ is the cost of reducing the basis using quantum lattice sieving algorithm in [13]. Columns β and s give the parameters of the attacks (BKZ block size and width of the Gaussian sampler).

Scheme	$\log_2 T_{\mathrm{attack}}$	β	s	$-\log_2 p$	$\log_2 T_{\text{Q-sample}}$	$\log_2 T_{\text{Q-BKZ}}$
NIST Level 2	146	571	358953	50	119	146
NIST Level 3	219	854	752706	72	179	219
NIST Level 5	312	1215	804127	100	259	312

Acknowledgments. The authors thank the anonymous reviewers of CRYPTO 2026 for their comments which helped improve the paper. This work has been supported by the French Agence Nationale de la Recherche through the France 2030 program under grant agreement No. ANR-22-PETQ-0008 PQ-TLS.

References

1. Aggarwal, D., Chen, Y., Kumar, R., Shen, Y.: Improved (provable) algorithms for the shortest vector problem via bounded distance decoding. In: STACS. LIPIcs, vol. 187, pp. 4:1–4:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPICS.STACS.2021.4
2. Aggarwal, D., Chen, Y., Kumar, R., Shen, Y.: Improved classical and quantum algorithms for the shortest vector problem via bounded distance decoding. SIAM J. Comput. **54**(2), 233–278 (2025). https://doi.org/10.1137/22M1486959, https://doi.org/10.1137/22m1486959
3. Aggarwal, D., Dadush, D., Regev, O., Stephens-Davidowitz, N.: Solving the shortest vector problem in 2^n time using discrete gaussian sampling: Extended abstract. In: STOC. pp. 733–742. ACM (2015). https://doi.org/10.1145/2746539.2746606
4. Aharonov, D., Regev, O.: Lattice problems in NP cap conp. J. ACM **52**(5), 749–765 (2005). https://doi.org/10.1145/1089023.1089025, https://doi.org/10.1145/1089023.1089025

5. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: STOC. pp. 99–108. ACM (1996). https://doi.org/10.1145/237814.237838
6. Albrecht, M.R.: On dual lattice attacks against small-secret LWE and parameter choices in helib and SEAL. In: EUROCRYPT (2). Lecture Notes in Computer Science, vol. 10211, pp. 103–129 (2017). https://doi.org/10.1007/978-3-319-56614-6_4
7. Albrecht, M.R., Gheorghiu, V., Postlethwaite, E.W., Schanck, J.M.: Estimating quantum speedups for lattice sieves. In: ASIACRYPT (2). pp. 583–613. Lecture Notes in Computer Science, Springer (2020). https://doi.org/10.1007/978-3-030-64834-3_20
8. Albrecht, M.R., Shen, Y.: Quantum augmented dual attack. IACR Cryptol. ePrint Arch. p. 656 (2022), https://eprint.iacr.org/2022/656
9. Aono, Y., Nguyen, P.Q., Shen, Y.: Quantum lattice enumeration and tweaking discrete pruning. In: ASIACRYPT (1). Lecture Notes in Computer Science, vol. 11272, pp. 405–434. Springer (2018). https://doi.org/10.1007/978-3-030-03326-2_14
10. de Boer, K.: Random Walks on Arakelov Class Groups. Ph.D. thesis, Leiden University (2022), https://hdl.handle.net/1887/3463719, phD thesis, Institutional Repository of Leiden University
11. de Boer, K., Felderhoff, J.: Quantumly computing s-unit groups in quantified polynomial time and space. IACR Cryptol. ePrint Arch. p. 1825 (2025), https://eprint.iacr.org/2025/1825
12. Bollauf, M.F., Pouly, A., Shen, Y.: Solving SIS in any norm via gaussian sampling. Cryptology ePrint Archive, Paper 2026/225 (2026), https://eprint.iacr.org/2026/225
13. Bonnetain, X., Chailloux, A., Schrottenloher, A., Shen, Y.: Finding many collisions via reusable quantum walks - application to lattice sieving. In: EUROCRYPT (5). Lecture Notes in Computer Science, vol. 14008, pp. 221–251. Springer (2023). https://doi.org/10.1007/978-3-031-30589-4_8
14. Bos, J.W., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In: EuroS&P. pp. 353–367. IEEE (2018). https://doi.org/10.1109/EUROSP.2018.00032
15. Boyer, M., Brassard, G., Høyer, P., Tapp, A.: Tight bounds on quantum searching. Fortschritte der Physik: Progress of Physics **46**(4-5), 493–505 (1998). https://doi.org/10.1002/(SICI)1521-3978(199806)46:4/5<493::AID-PROP493>3.0.CO;2-P
16. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: STOC. pp. 575–584. ACM (2013). https://doi.org/10.1145/2488608.2488680
17. Brassard, G., Hoyer, P., Mosca, M., Tapp, A.: Quantum amplitude amplification and estimation. Contemporary Mathematics **305**, 53–74 (2002). https://doi.org/10.1090/conm/305/05215
18. Carrier, K., Meyer-Hilfiger, C., Shen, Y., Tillich, J.: Assessing the impact of a variant of matzov's dual attack on kyber. In: CRYPTO (1). Lecture Notes in Computer Science, vol. 16000, pp. 444–476. Springer (2025). https://doi.org/10.1007/978-3-032-01855-7_15
19. Carrier, K., Shen, Y., Tillich, J.: Faster dual lattice attacks by using coding theory. IACR Cryptol. ePrint Arch. p. 1750 (2022)
20. Chailloux, A., Loyer, J.: Lattice sieving via quantum random walks. In: ASIACRYPT (4). Lecture Notes in Computer Science, vol. 13093, pp. 63–91. Springer (2021). https://doi.org/10.1007/978-3-030-92068-5_3

21. Cho, B., Hhan, M., Kim, T., Lee, J., Shen, Y.: Does quantum lattice sieving require quantum RAM? IACR Cryptol. ePrint Arch. p. 1700 (2024), https://eprint.iacr.org/2024/1700
22. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-dilithium algorithm specifications and supporting documentation (2017), https://api.semanticscholar.org/CorpusID:198994007
23. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: Crystals-dilithium: A lattice-based digital signature scheme. IACR Trans. Cryptogr. Hardw. Embed. Syst. **2018**(1), 238–268 (2018). https://doi.org/10.13154/TCHES.V2018.I1.238-268, https://doi.org/10.13154/tches.v2018.i1.238-268
24. Ducas, L., Pulles, L.N.: Does the dual-sieve attack on learning with errors even work? In: CRYPTO (3). Lecture Notes in Computer Science, vol. 14083, pp. 37–69. Springer (2023). https://doi.org/10.1007/978-3-031-38548-3_2
25. Dürr, C., Høyer, P.: A quantum algorithm for finding the minimum. CoRR **quant-ph/9607014** (1996), http://arxiv.org/abs/quant-ph/9607014
26. Espitau, T., Joux, A., Kharchenko, N.: On a dual/hybrid approach to small secret LWE - A dual/enumeration technique for learning with errors and application to security estimates of FHE schemes. In: INDOCRYPT. Lecture Notes in Computer Science, vol. 12578, pp. 440–462. Springer (2020). https://doi.org/10.1007/978-3-030-65277-7_20
27. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC. pp. 197–206. ACM (2008). https://doi.org/10.1145/1374376.1374407
28. Guo, Q., Johansson, T.: Faster dual lattice attacks for solving LWE with applications to CRYSTALS. In: ASIACRYPT (4). Lecture Notes in Computer Science, vol. 13093, pp. 33–62. Springer (2021). https://doi.org/10.1007/978-3-030-92068-5_2
29. Høyer, P., Mosca, M., de Wolf, R.: Quantum search on bounded-error inputs. In: ICALP. Lecture Notes in Computer Science, vol. 2719, pp. 291–299. Springer (2003). https://doi.org/10.1007/3-540-45061-0_25
30. Jerrum, M.R., Valiant, L.G., Vazirani, V.V.: Random generation of combinatorial structures from a uniform. Theor. Comput. Sci. **43**(2–3), 169–188 (Jul 1986)
31. Kitaev, A., Webb, W.A.: Wavefunction preparation and resampling using a quantum computer. arXiv preprint arXiv:0801.0342 (2008)
32. Klein, P.N.: Finding the closest lattice vector when it's unusually close. In: SODA. pp. 937–941. ACM/SIAM (2000). https://doi.org/DBLP:conf/soda/Klein00
33. Knuth, D.E.: The Art of Computer Programming, Volume 2: Seminumerical Algorithms. Addison-Wesley, 4 edn. (2011)
34. Laarhoven, T.: Search problems in cryptography: From fingerprinting to lattice sieving. Ph.D. thesis, Eindhoven University of Technology (2015)
35. Laarhoven, T., Mosca, M., van de Pol, J.: Solving the shortest vector problem in lattices faster using quantum search. In: PQCrypto. Lecture Notes in Computer Science, vol. 7932, pp. 83–101. Springer (2013). https://doi.org/10.1007/978-3-642-38616-9_6
36. MATZOV: Report on the security of LWE: Improved dual lattice attack (Apr 2022). https://doi.org/10.5281/zenodo.6412487, https://doi.org/10.5281/zenodo.6412487
37. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. vol. 37, pp. 267–302 (2007). https://doi.org/10.1137/S0097539705447360, https://doi.org/10.1137/S0097539705447360

38. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM J. Comput. **37**(1), 267–302 (2007). https://doi.org/10.1137/S0097539705447360, https://doi.org/10.1137/S0097539705447360
39. Montanaro, A.: Quantum speedup of monte carlo methods. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences **471**(2181), 20150301 (09 2015). https://doi.org/10.1098/rspa.2015.0301, https://doi.org/10.1098/rspa.2015.0301
40. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press, USA, 10th edn. (2011)
41. Ozols, M., Roetteler, M., Roland, J.: Quantum rejection sampling. ACM Trans. Comput. Theory **5**(3), 11:1–11:33 (2013). https://doi.org/10.1145/2493252.2493256, https://doi.org/10.1145/2493252.2493256
42. Pouly, A., Shen, Y.: Provable dual attacks on learning with errors. In: EUROCRYPT (6). Lecture Notes in Computer Science, vol. 14656, pp. 256–285. Springer (2024). https://doi.org/10.1007/978-3-031-58754-2_10
43. Pouly, A., Shen, Y.: Discrete gaussian sampling for BKZ-reduced basis. In: PQCrypto (2). Lecture Notes in Computer Science, vol. 15578, pp. 63–88. Springer (2025). https://doi.org/10.1007/978-3-031-86602-9_3
44. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: Falcon. Post-Quantum Cryptography Project of NIST (2020)
45. Qu, H., Xu, G.: On the provable dual attack for LWE by modulus switching. In: ASIACRYPT (3). Lecture Notes in Computer Science, vol. 16247, pp. 34–64. Springer (2025). https://doi.org/10.1007/978-981-95-5099-9_2
46. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. vol. 56, pp. 34:1–34:40 (2009). https://doi.org/10.1145/1568318.1568324, https://doi.org/10.1145/1568318.1568324
47. Regev, O., Stephens-Davidowitz, N.: A reverse minkowski theorem. In: STOC. pp. 941–953. ACM (2017). https://doi.org/DBLP:conf/stoc/0001S17
48. Sanders, Y.R., Low, G.H., Scherer, A., Berry, D.W.: Black-box quantum state preparation without arithmetic. Phys. Rev. Lett. **122**, 020502 (Jan 2019). https://doi.org/10.1103/sanders2019black, https://link.aps.org/doi/10.1103/sanders2019black
49. Wang, Z., Ling, C.: Lattice gaussian sampling by markov chain monte carlo: Bounded distance decoding and trapdoor sampling. IEEE Trans. Inf. Theory **65**(6), 3630–3645 (2019). https://doi.org/10.1109/TIT.2019.2901497, https://doi.org/10.1109/TIT.2019.2901497
50. Yoder, T.J., Low, G.H., Chuang, I.L.: Fixed-point quantum search with an optimal number of queries. Physical review letters **113**(21), 210501 (2014)

A Finding the maximum in a register

In this section, we prove Theorem 2 from Section 2.5.

Theorem 2. *[[25,29]] Let s be an efficient randomized algorithm that, on input $i \in \{0,1\}^\ell$, returns $s(i)$, where $s(i)$ equals a value s_i ("good") with probability at least 9/10, and s returns a "faulty" output with probability at most 1/10. Algorithm 6, parametrised by $k > 0$, finds the i such that s_i is maximal, with probability at least $1 - 1/2^k$. This algorithms calls s a number of times in $O(k\sqrt{2^\ell})$, and uses $O(k\ell\sqrt{2^\ell})$ additional gates, and* $\mathrm{poly}(\ell)$ *qubits.*

This is essentially a "bounded-error" version of quantum maximum-finding, which combines the results of [25] (for maximum-finding) and [29, Section 3] (for bounded-error quantum search).

We start from the quantum minimum-finding from [25], adapted for maximum-finding, assuming for now that the function s is exact. The algorithm runs as follows:

1. Choose $j \in \{0,1\}^\ell$ uniformly at random.
2. Apply quantum exponential search [15] over the space $\{0,1\}^\ell$ to find $i \in \{0,1\}^\ell$ such that $s(i) > s(j)$. Let $j \leftarrow i$.
3. Go back to Step 2.

One should repeat this loop until "the total running time is less than $22.5\sqrt{2^\ell} + 1.4\log_2(2^\ell)$", as per [25]. Indeed, after some time, there will not be any new i such that $s(i) > s(j)$, and the exponential search step will just run forever (before we stop it).

Theorem 12 ([25, Theorem 1, Lemma 2]). *Consider a function s with quantum oracle access. The algorithm presented in [25] finds $i \in \{0,1\}^\ell$ such that $s(i)$ is maximal with probability at least $1/2$, in time (and number of calls to s) $O(\sqrt{2^\ell})$.*

In Step 3, *exponential search* is a sequence of Grover searches with an increasing number of iterations, which allows to produce the solution state in some guaranteed average time even though the probability of success of the search itself is unknown.

Like in [42], we adapt this algorithm to the case where the function s has a non-zero error rate (and only returns the "true" value s_i with probability $\geq 9/10$). The idea is to replace the quantum search steps in the exponential search layers, which are instances of Grover's search, by the bounded-error quantum search presented in [29].

Theorem 13 ([29, Section 3]). *Consider a set of $\{F_i\}_{i\in\{0,1\}^\ell}$ functions, such that each F_i is destined to be evaluated on i, and to return a value f_i, that equals 0 or 1. Consider that each F_i can return a faulty result with probability at most $1/10$, and that there are t solutions. There exists a quantum algorithm that finds a solution in average time $O(\sqrt{2^\ell/t})$.*

The method proposed in [29, Section 3] is a combination of exponential search with their bounded-error search. The bounded-error search consists in nested quantum amplification subroutines, in which one amplifies the elements evaluating to 1, and progressively refines the test by using a majority vote on several instances of F_i.

In our case, we can immediately plug in the result of Theorem 13 into the algorithm above. Indeed, the test f_i is exactly: $s(i) > s(j)$, where j is the current constant. While $s(j)$ is evaluated once before running the search, and so can be evaluated with certainty, the error comes from the evaluation of $s(i)$ in the

search. Thus the probability of error for $s(i)$ only needs to be $1/10$. This gives the following algorithm.

Algorithm 6 Finding the maximum

Input: An efficient randomized function s defined on $\{0,1\}^\ell$, that returns a result $s(i)$ that is supposed to be equal to a value s_i, and that is faulty with probability at most $1/10$. An integer $k > 0$.

Output: An index $i \in \{0,1\}^\ell$

1: Choose $j \in \{0,1\}^\ell$ uniformly at random
2: $j_{max} \leftarrow j$
3: **for** $k_1 = 1 \to k$ **do**
4: **repeat**
5: Apply bounded-error quantum exponential search over the space $\{0,1\}^\ell$ to find $i \in \{0,1\}^\ell$ such that $s(i) > s(j)$.
6: **until** the loop has expended $22.5\sqrt{2^\ell} + 1.4\log_2(2^\ell)$ time
7: If $s(j) > s(j_{max})$, set $j_{max} = j$
8: Return j_{max}

We can now give the proof of Theorem 2.

Proof. By Theorem 13, the exponential search loop finds a solution in $O(\sqrt{2^\ell/t})$ average time, even under bounded-error inputs. As we said above, by assumption, the probability of error of s is exactly the one required by the theorem.

By Theorem 12, the loop succeeds in finding the maximum in time $O\left(\sqrt{2^\ell}\right)$ and with probability of success $1/2$. By repeating the algorithm k times and taking the maximum of the obtained values, we amplify the success probability to $1 - 1/2^k$. Finally, this algorithm is efficient in space, as we only require ℓ qubits to represent i, and enough space to compute s in superposition. Since s is assumed to be efficient this gives a $\mathrm{poly}(\ell)$ space. There is an additional count of $O(\ell)$ gates per iterate of Grover's search, which gives a total of $O\left(k\ell\sqrt{2^\ell}\right)$ additional gates. □

B Proofs of Some Results

B.1 Justification of Lemma 2

Lemma 2. *[From Lemma 2 in [43]] For any $\sigma > 0$, $\rho_\sigma(\mathbb{Z}) \leq 3\sigma + 4$.*

This follows immediately from Lemma 2 in [43], which we restate here for completeness. We simply bound together the case of small and large σ.

Lemma 10 ([43, Lemma 2]). *Consider a real number $\sigma > 0$ and define*

$$\tilde{\rho}(\sigma) = \begin{cases} 1 + 2e^{-\pi/\sigma^2} & \text{if } \sigma \leqslant 1 \\ \sigma(1 + 2e^{-\pi\sigma^2}) & \text{otherwise.} \end{cases}$$

Then $\tilde{\rho}$ is a continuously increasing function, and for any $\sigma > 0$,

$$0 < \rho_\sigma(\mathbb{Z}) - \tilde{\rho}(\sigma) \leqslant 2\sum_{k=2}^{\infty} e^{-\pi k^2} \leqslant \gamma := 6.974685811 \times 10^{-6}$$

B.2 Proof of Lemma 3

In order to prove Lemma 3, we use the following.

Lemma 11 ([38, Lemma 2.10]). *Let $\mathcal{L}$ be a full rank lattice in $\mathbb{R}^m$, and $\sigma > 0$, $t > 1/\sqrt{2\pi}$, and $\boldsymbol{c} \in \mathbb{R}^m$. Let $\mathcal{B}$ be the ball of $\mathbb{R}^m$ of center 0 and radius 1, according to the euclidean distance. Then, we have*

$$\rho_\sigma(\mathcal{L} \backslash t\sqrt{m}\mathcal{B}) < C^m \rho_\sigma(\mathcal{L})$$
$$\rho_{\sigma,\boldsymbol{c}}(\mathcal{L} \backslash t\sqrt{m}\mathcal{B}) < 2C^m \rho_\sigma(\mathcal{L})$$

where $C := t\sqrt{2\pi e} \times e^{-\pi t^2}$.

Lemma 3. *Consider an integer $M \in \mathbb{N}^*$, and real numbers $\sigma > 0$ and $\boldsymbol{c} \in \mathbb{R}$. Then, we have*

$$\rho_{\sigma,\boldsymbol{c}}(\mathbb{Z}) \leqslant \frac{\rho_\sigma(\llbracket -2^M, 2^M \rrbracket)}{1 - 2^M\sqrt{2\pi e} \times e^{-\pi 2^{2M}}} \text{ and } \rho_{\sigma,\boldsymbol{c}}(\mathbb{Z} \backslash \llbracket -2^M, 2^M \rrbracket) \leqslant O(\sigma \times 2^M e^{-\pi 2^{2M}}) \ .$$

Proof. Let us prove the first inequality. By Lemma 1, one has

$$\rho_{\sigma,\boldsymbol{c}}(\llbracket -2^M, 2^M \rrbracket) \leqslant \rho_{\sigma,\boldsymbol{c}}(\mathbb{Z}) \leqslant \rho_\sigma(\mathbb{Z}).$$

Note that, since $M > 0$, we have $2^M > 1/\sqrt{2\pi}$. Then, we can apply Lemma 11 with $t = 2^M$ and $m = 1$, which gives:

$$\begin{aligned}\rho_\sigma(\mathbb{Z}) &= \rho_\sigma(\llbracket -2^M, 2^M \rrbracket) + \rho_\sigma(\mathbb{Z} \backslash \llbracket -2^M, 2^M \rrbracket) \\ &\leqslant \rho_\sigma(\llbracket -2^M, 2^M \rrbracket) + C\rho_\sigma(\mathbb{Z}) \ ,\end{aligned}$$

where $C := 2^M\sqrt{2\pi e} \times e^{-\pi 2^{2M}} < 1$. Therefore, we have

$$\begin{aligned}\rho_\sigma(\mathbb{Z}) &\leqslant \rho_\sigma(\llbracket -2^M, 2^M \rrbracket) \times \frac{1}{1-C} \\ &= \rho_\sigma(\llbracket -2^M, 2^M \rrbracket) \times \frac{1}{1 - 2^M\sqrt{2\pi e} \times e^{-\pi 2^{2M}}} \ .\end{aligned}$$

Let us then prove the second inequality. By Lemma 11 with $t = 2^M$, we also have:

$$\rho_{\sigma,\boldsymbol{c}}(\mathbb{Z} \backslash \llbracket -2^M, 2^M \rrbracket) \leqslant 2C\rho_\sigma(\mathbb{Z}) \ .$$

But then, by Lemma 2, we have $\rho_\sigma(\mathbb{Z}) \leqslant 3\sigma + 4$, therefore,

$$\rho_{\sigma,\boldsymbol{c}}(\mathbb{Z} \backslash \llbracket -2^M, 2^M \rrbracket) \leqslant 2^{M+1}\sqrt{2\pi e} \times e^{-\pi 2^{2M}}(3\sigma + 4) = O(\sigma \times 2^M e^{-\pi 2^{2M}}) \ .$$

This concludes the proof of the second inequality. □

B.3 Proof of Proposition 1

Proposition 1. *Consider a full rank lattice $\mathcal{L}(B) \in \mathbb{R}^m$, along with a finite subset $\Omega \supset B\prod_{i=1}^m [\![-2^{M-1}, 2^{M-1}]\!]$, for some integer $M > 0$. We have:*

$$\rho_\sigma(\mathcal{L}\backslash\Omega) \leqslant 2\left(2^{M-1}\sqrt{2\pi e}\right)^m \times e^{-\pi\cdot m\cdot 2^{2M-2}}(3\sigma/\min_i \|\widetilde{\boldsymbol{b}}_i\| + 4)^m \ .$$

Proof. Note $B = QR$ the QR decomposition of B. The matrix Q is orthonormal, R is upper triangular, and each diagonal coefficient $R_{i,i}$ of R equals $||\widetilde{\boldsymbol{b}}_i||$, where the $\widetilde{\boldsymbol{b}}_i$'s are the columns of the Gram-Schmidt transform of B. Then, we have

$$\rho_\sigma(\mathcal{L}\backslash\Omega) \leqslant \sum_{\boldsymbol{x}\in\mathbb{Z}^m\backslash\prod_{i=1}^m [\![-2^{M-1},2^{M-1}]\!]} e^{-\frac{\pi}{\sigma^2}||B\boldsymbol{x}||^2} = \sum_{\boldsymbol{x}\in\mathbb{Z}^m\backslash\prod_{i=1}^m [\![-2^{M-1},2^{M-1}]\!]} e^{-\frac{\pi}{\sigma^2}||R\boldsymbol{x}||^2}.$$

Since R is upper triangular, it is straightforward to see that its eigenvalues are precisely the $R_{i,i} = ||\widetilde{\boldsymbol{b}}_i||$. We then have:

$$\begin{aligned}\rho_\sigma(\mathcal{L}\backslash\Omega) &\leqslant \sum_{\boldsymbol{x}\in\mathbb{Z}^m\backslash\prod_{i=1}^m [\![-2^{M-1},2^{M-1}]\!]} e^{-\frac{\pi\min(||\widetilde{\boldsymbol{b}}_i||)^2}{\sigma^2}||\boldsymbol{x}||^2} \\ &= \rho_{\sigma/\min(||\widetilde{\boldsymbol{b}}_i||)}\left(\mathbb{Z}^m\backslash\prod_{i=1}^m [\![-2^{M-1},2^{M-1}]\!]\right) \ .\end{aligned}$$

We notice that $\mathbb{Z}^m\backslash\prod_{i=1}^m [\![-2^{M-1},2^{M-1}]\!] = \left(\mathbb{Z}\backslash[\![-2^{M-1},2^{M-1}]\!]\right)^m$. We can then apply Lemma 11 in dimension 1, and obtain:

$$\rho_\sigma(\mathcal{L}\backslash\Omega) \leqslant \left(2^{M-1}\sqrt{2\pi e}\right)^m \times e^{-\pi\cdot m\cdot 2^{2M-2}}(\rho_{\sigma/\min(||\widetilde{\boldsymbol{b}}_i||)}(\mathbb{Z}))^m \ .$$

Finally, by Lemma 2, we have $\rho_{\sigma/\min(||\widetilde{\boldsymbol{b}}_i||)}(\mathbb{Z}) \leqslant 3(\sigma/\min(||\widetilde{\boldsymbol{b}}_i||)) + 4$, so we conclude.

This concludes the proof. □

B.4 Proof of Lemma 9

Lemma 9. *Let $B \in \mathbb{Z}_q^{m\times n}$, $\mathcal{L} \subset \mathbb{R}^m$ a lattice, $\boldsymbol{e} \in \mathbb{Z}^m$, $\epsilon, \sigma, \delta, \eta_1 > 0$, and $N \in \mathbb{N}$. Let $\tau := \frac{1}{\sigma}\sqrt{m/2\pi}$, $\eta_2 \geqslant 0$, and assume that $\lambda_1(\mathcal{L} + \mathcal{L}_q(B)) \geqslant \tau + ||\boldsymbol{e}||$, and*

$$\rho_{1/\sigma}(\boldsymbol{e}) - \rho_{1/\sigma}(\lambda_1(\mathcal{L} + \mathcal{L}_q(B)) - ||\boldsymbol{e}|| - \tau) > 2\epsilon + 2\eta_1 + \eta_2.$$

Let D be a probability distribution over a finite subset Ω of $\mathcal{L}_q^\perp(B)$, such that $d_{TV}(D, D_{\mathcal{L}_q^\perp(B),q\sigma}) \leqslant \eta_1/2$. Let $\mathcal{GS}$ be a quantum circuit such that $\mathcal{GS} : |0\rangle \to |D\rangle$, and let $\phi : |\boldsymbol{x}\rangle|\boldsymbol{w}\rangle \to |\cos(2\pi\langle\boldsymbol{x},\boldsymbol{w}\rangle/q)\rangle$. Denote by $\widetilde{f}(\boldsymbol{x})$ the output of $\mathcal{A}^{\mathcal{GS},\phi}$ on input $\boldsymbol{x}$ and with parameters ϵ and δ.

Then, with probability at least $1 - q^m\delta$, for all $\boldsymbol{x} \in \mathcal{L}\backslash\mathcal{L}_q(B)$, we have

$$|\widetilde{f}(\boldsymbol{x}) - f_{\mathcal{L}_q(B),1/\sigma}(\boldsymbol{x})| \leqslant \epsilon + \eta_1,$$

$$\widetilde{f}(\boldsymbol{e}) \geqslant \rho_{1/\sigma}(\boldsymbol{e}) - \epsilon - \eta_1 > \rho_{1/\sigma}(\lambda_1(\mathcal{L} + \mathcal{L}_q(B)) - ||\boldsymbol{e}|| - \tau) + \epsilon + \eta_1 + \eta_2 \geqslant \widetilde{f}(\boldsymbol{e} + \boldsymbol{x}) + \eta_2 \ .$$

Proof. Let us set $X = \{0, \dots, q-1\}^m$. Then, by Corollary 1, with probability at least $1-q^m\delta$, we have $|\tilde{f}(\boldsymbol{x}) - f_{\mathcal{L}_q(B),1/\sigma}(\boldsymbol{x})| \leqslant \delta + \eta_1$, for all $\boldsymbol{x} \in X$. Additionally, $\mathcal{L} = \mathcal{L}_q(B)$ is a q-ary lattice, which means $q\mathbb{Z}^m \subset \mathcal{L}$. So, $\mathcal{L} + X \supset q\mathbb{Z}^m + \{0, \dots, q-1\}^m = \mathbb{Z}^m$. Therefore, this means that with probability at least $1 - q^m\delta$, we have $|\tilde{f}(\boldsymbol{x}) - f_{\mathcal{L}_q(B),1/\sigma}(\boldsymbol{x})| \leqslant \epsilon + \eta_1$, for all $\boldsymbol{x} \in \mathbb{Z}^m$.

By Lemma 8, we have

$$\tilde{f}(\boldsymbol{e}) \geqslant f_{\mathcal{L}_q(B),1/\sigma}(\boldsymbol{e}) - \delta - \eta_1 \geqslant \rho_{1/\sigma}(\boldsymbol{e}) - \epsilon - \eta_1.$$

Let $\boldsymbol{x} \in \mathcal{L}\backslash\mathcal{L}_q(B)$, then $\boldsymbol{z} - \boldsymbol{x} \in \mathcal{L} + \mathcal{L}_q(B)$, and $\boldsymbol{z} - \boldsymbol{x} \neq 0$ for any $\boldsymbol{z} \in \mathcal{L}_q(B)$. As a result, $\mathcal{L}_q(B) - \boldsymbol{x} \subset (\mathcal{L} + \mathcal{L}_q(B))\backslash\{0\}$. Hence

$$\mathrm{dist}(\boldsymbol{x}, \mathcal{L}_q(B)) = \min_{\boldsymbol{z}\in\mathcal{L}_q(B)} ||\boldsymbol{x}+\boldsymbol{z}|| \geqslant \min_{\boldsymbol{y}\in(\mathcal{L}+\mathcal{L}_q(B))\backslash\{0\}} ||\boldsymbol{y}|| = \lambda_1(\mathcal{L}+\mathcal{L}_q(B)) \geqslant \tau + ||\boldsymbol{e}||.$$

Then

$$\mathrm{dist}(\boldsymbol{e}+\boldsymbol{s}, \mathcal{L}_q(B)) \geqslant \mathrm{dist}(\boldsymbol{x}, \mathcal{L}_q(B)) - ||\boldsymbol{e}|| \geqslant \tau.$$

We can apply Lemma 8 to get that for any $\boldsymbol{x} \in \mathcal{L}\backslash\{0\}$,

$$\tilde{f}(\boldsymbol{e}+\boldsymbol{x}) \leqslant f_{\mathcal{L}_q(B),1/\sigma}(\boldsymbol{e}+\boldsymbol{x}) + \eta_1 + \epsilon \leqslant \rho_{1/\sigma}(\mathrm{dist}(\boldsymbol{e}+\boldsymbol{x}, \mathcal{L}_q(B)) - \tau) + \epsilon + \eta_1.$$

Since $\rho_{1/\sigma} : [0, \infty[\to \mathbb{R}$ is decreasing, and reusing the two previous inequalities, we have moreover

$$\begin{aligned}\rho_{1/\sigma}(\mathrm{dist}(\boldsymbol{e}+\boldsymbol{x}, \mathcal{L}_q(B)) - \tau) &\leqslant \rho_{1/\sigma}(\mathrm{dist}(\boldsymbol{x}, \mathcal{L}_q(B)) - ||\boldsymbol{e}|| - \tau) \\ &\leqslant \rho_{1/\sigma}(\lambda_1(\mathcal{L} + \mathcal{L}_q(B)) - ||\boldsymbol{e}|| - \tau).\end{aligned}$$

Putting everything together, we have

$$\tilde{f}(\boldsymbol{e}) - \tilde{f}(\boldsymbol{e}+\boldsymbol{x}) \geqslant \rho_{1/\sigma}(\boldsymbol{e}) - \rho_{1/\sigma}(\lambda_1(\mathcal{L} + \mathcal{L}_q(B)) - ||\boldsymbol{e}|| - \tau) - 2\epsilon - 2\eta_1 > \eta_2.$$

by assumption. □