Report on the Security of LWE: Improved Dual Lattice Attack

The Center of Encryption and Information Security – MATZOV[*†]
IDF

Abstract

Many of the leading post-quantum key exchange and signature schemes rely on the conjectured hardness of the Learning With Errors (LWE) and Learning With Rounding (LWR) problems and their algebraic variants, including 3 of the 6 finalists in NIST's PQC process. The best known cryptanalysis techniques against these problems are primal and dual lattice attacks, where dual attacks are generally considered less practical.

In this report, we present several algorithmic improvements to the dual lattice attack, which allow it to exceed the efficiency of primal attacks. In the improved attack, we enumerate over more coordinates of the secret and use an improved distinguisher based on FFT. In addition, we incorporate improvements to the estimates of the cost of performing a lattice sieve in the RAM model, reducing the gate-count of random product code decoding and performing less inner product calculations.

Combining these improvements considerably reduces the security levels of Kyber, Saber and Dilithium, the LWE/LWR based finalists, bringing them below the thresholds defined by NIST.

1 Introduction

In this report, the Center of Encryption and Information Security (MATZOV) presents an overview of results of an inner audit project regarding leading post-quantum cryptographic schemes. This work does not aim to provide a complete analysis of all candidate post-quantum encryption schemes, nor to recommend cryptographic algorithms to use. Rather, this publication is meant to share advances in the cryptanalysis of lattices which we believe to be relevant to academic research in the field. Some of the additional results of the audit, regarding quantum security ramifications, possible extension of the claims to NTRU-based lattice schemes and additional improvements

[*]matzov@idf.il

[†]MATZOV is a unit of the IDF, exclusively responsible for all aspects of Defensive Information Security and Encryption in Israel, providing security solutions to Israeli security establishments and to critical state infrastructures.

to the presented attack are mentioned in Section 8, and might be published at a later date.

1.1 Motivation

Since the 1990s, the threat of quantum computers to modern security protocols has been well known, with ongoing research regarding applications of the Shor [Sho94, HRS17, RNSL17, HJN$^+$20, BBvHL21, GE21] and Grover [Gro96, Ber10, GLRS16, BNS19, JNRV20] algorithms to cryptanalysis of cryptographic schemes. It is heavily debated by experts whether and when a quantum computer capable of executing those algorithms for relevant cryptanalytic problems will be feasible. However, the growing interest in academic cycles, followed by governments [BPS21, EHH$^+$21, BDH$^+$21, ANS22, Dir22] and industry powers [CCD$^+$15, Ant20, Gri21] - led to accelerated development of quantum-resistant countermeasures, such as Quantum Key Distribution (QKD) and Post-Quantum Cryptography (PQC).

The main initiative for determining new cryptographic schemes is led by NIST – NIST PQC Standardization Process [CJL$^+$16, Nat16]. Starting with 69 candidates in 2017, the process now consists of 7 finalists which are being considered for standardization [AASA$^+$19, MAA$^+$20]. The majority of the finalists are based on structured lattices, and rely on the hardness of various lattice problems [Ajt98, Mic98, Kho04, HR07].

Lattice-based cryptography was first proposed by Ajtai and Dwork [AD97], and later expanded by the introduction of the NTRU cryptosystem [HPS98]. Over the years, lattice-based cryptography has become one of the most promising solutions for the quantum computing threat. A large fraction of lattice-based cryptographic mechanisms is built upon the Learning With Errors (LWE) problem [Reg09] and variations of it, such as Ring-LWE (RLWE) [LPR13], Module-LWE (MLWE) [LS15] and Learning With Rounding (LWR) [BPR12]. LWE and its variations benefit from a series of worst-case to average-case reductions – breaking certain random instantiations of these problems is at least as hard as solving worst case lattice problems [Reg09, Pei08, BLP$^+$13, LPR13, LS15]. This work is focused on lattice schemes whose hardness relies on the LWE/R problems – CRYSTALS-KYBER [BDK$^+$18, ABD$^+$17], CRYSTALS-DILITHIUM [DKL$^+$18, DKL$^+$21] and SABER [DKRV18, BMD$^+$20].

Understanding the exact hardness level of the underlying lattice problems is an active and rapidly changing research area. As such, we took an interest in understanding and verifying the current assumptions regarding the strength of lattice cryptosystems, starting with LWE/R problems. Various publications have shown substantial improvements in recent years [Laa15, Laa16, AGVW17, Duc18, LM18, PHS19, ABF$^+$20, BR20, DLdW20, BLL$^+$21, CL21, Hei21] in the classical and quantum analysis of lattice problems. We therefore assumed further improvements are imminent – and our security evaluation of the different cryptosystems takes this into consideration.

1.2 Contributions

In this work we present several improvements to the dual lattice attack which reduce the security level of Kyber [BDK$^+$18], Saber [DKRV18], and Dilithium [DKL$^+$18] below the required security level specified by NIST [Nat16]. In particular, the security of Kyber is reduced to between 4 and 14 bits below the security cutoff requirement. This reduction in security is independent of the refined BKZ strategies suggested in [ADH$^+$19] and the further reduction in security foreseen in the Kyber submission document [BDK$^+$18].

Our improvements consist of three main results.

1. We present an *improved enumeration strategy over the secret*, combined with a *more efficient distinguishing algorithm*. The standard dual lattice attack performs an exhaustive search on the first coordinate of the secret, and for each guess invokes a distinguishing algorithm. Our attack enumerates over several coordinates of the secret simultaneously, and uses a more efficient distinguishing algorithm. Its efficiency stems from a tailored usage of the Fast-Fourier-Transform (FFT) combined with modulus switching. Our attack requires reducing lattices of much smaller dimension than was believed to be necessary [ACD$^+$18, BMD$^+$20, ABD$^+$21, DKL$^+$21] which translates to a more efficient attack in total.

2. We *improve the estimated costs of sieving in the* RAM *model* compared to previous estimates [AGPS20]. The main improvement is the random product code decoding algorithm which requires a smaller number of gates. Our algorithm requires one addition, one xor, and three comparisons per legal codeword, which translate to $\approx$430 gates for a lattice of rank 400, as opposed to [AGPS20] which requires a super-constant number of *inner products* per legal codeword, which translate to $\approx$3,500,000 gates for a lattice of rank 400.

   We also optimize over a larger set of parameters than considered in [AGPS20], which leads to further improvements. Overall, the cost of sieving is reduced by $\approx$6 bits in rank 400.

3. We propose a *faster short vectors sampling procedure*. We first run the BKZ algorithm using a sieve as the relevant SVP oracle. As observed previously, in such cases one enjoys the so-called "dimensions-for-free" trick [Duc18]. Then, we run a final sieve on the first block of the reduced basis to find many short vectors (this time without using dimensions-for-free, as we use many resulting vectors from the output of the sieve). However, we use a different block size for this task, which allows us to optimize the overall cost by balancing the costs of the two parts.

We optimize our proposed attack and review the security level of various NIST candidates. We analyze Kyber [BDK$^+$18], Dilithium [DKL$^+$18], and Saber [DKRV18] and present the results in Table 1. We compare the security level, as specified by the Call for Proposals [Nat16], the security estimates by the candidate's authors, and our proposed attack.

We remark that our research primarily focused on Kyber. As such, this candidate is the one that is most affected by our proposed attack. Our attack can also be applied to Dilithium and Saber, and so we analyze the cost of our attack on these candidates. We believe that certain adaptations can be made to improve our attack for the other specific candidates, but have not yet completed such research. We mention some of these ideas in Section 8.

Table 1: Security estimations for Dilithium, Kyber and Saber. Comparison between the security level required by NIST, the authors' claimed security, and the security level as shown in this work. All of the costs are $\log_2$ in the gate-count metric.

Candidate	Required Security Level By NIST [Nat16]	Estimated Security Level [DKL+21] [ABD+21] [BMD+20]	This Work
Kyber512	143	151.5	**137.5**
Kyber768	207	215.1	**193.5**
Kyber1024	272	287.3	**257.8**
Dilithium2	146	159	**146.3**
Dilithium3	207	217	**202.0**
Dilithium5	272	285	**263.6**
LightSaber	143	Unspecified	**138.4**
Saber	207	Unspecified	**202.7**
FireSaber	272	Unspecified	**264.9**

We see that for almost all candidates and security levels, with the sole exception of Dilithium with Security Level 2, our proposed attack's cost is below the required cost.

A recent paper by Guo and Johansson [GJ21] presented a similar attack for Kyber and Dilithium, and achieved the best known results to date. Their attack also uses an FFT-based distinguishing algorithm in a small modulus, which allows them to reduce the error of some of the equations while keeping the running time of the FFT low. Our attack calls for a larger modulus compared to theirs, and allows us to use a lattice reduction algorithm on a lattice of a much smaller rank. Guo and Johansson also presented a "two-step lattice reduction strategy" which is the same as our improved short vectors sampling procedure. In Section 1.4 we compare our attack with the results of [GJ21] and show that it presents a significant improvement over them. In particular, a comparison between the results of the two techniques is given in Table 2.

1.3 Technical Overview

We present the main techniques used in the different components of our proposed attack. We first describe the attack from a high-level point of view, listing the core improvements over the standard dual lattice attack. We then describe our short vectors sampling algorithm, and conclude by discussing the reduced cost of the underlying sieving algorithms.

1.3.1 Overview of the Dual Attack

We first recall the dual lattice attack, which is a distinguishing attack. Given an input sample $(A, b) \in (\mathbb{Z}/q\mathbb{Z})^{m\times n} \times (\mathbb{Z}/q\mathbb{Z})^m$, the attacker needs to decide whether the pair (A, b) was sampled uniformly at random, or is of the form $b = As + e \mod q$, for some short $(s, e) \in (\mathbb{Z}/q\mathbb{Z})^n \times (\mathbb{Z}/q\mathbb{Z})^m$. The attack consists of two parts. First, one uses A to find many vectors such that when (A, b) is sampled from the LWE distribution, their inner products with b tend to be small. On the other hand, when (A, b) is sampled from a uniform distribution, the inner products are distributed uniformly. The second part of the attack boils down to distinguishing between a modular (discrete) Gaussian and the uniform distribution. We remark that the first part is done by finding short vectors in the lattice $\Lambda = \{(x, y) \in \mathbb{Z}^m \times \mathbb{Z}^n \mid x^T A = y^T\}$. The second part employs statistical tools to distinguish between two similar distributions.

The distinguishing attack can then be used to recover the secret as follows. The attacker generates short vectors $(x, y) \in \Lambda$ as before. They then iterate over the coordinates of the secret. For each one, the attacker guesses the value of the secret in that coordinate and test the guess using the distinguishing algorithm, until they find the correct guess. In each step, one utilizes the parts of the secret already recovered. As such, the complexity of the entire attack is dominated by the complexity of recovering the first coordinate of the secret.

1.3.2 FFT-Based Distinguisher

Our first observation is that we can easily improve the attack by enumerating over *several* coordinates of the secret simultaneously. Note that this not only lets us recover more coordinates of the secret, but also *reduces the dimension* of the lattice. Indeed, as we enumerate over some coordinates of the secret, we can drop the constraint that the corresponding coordinates of $y^T = x^T A$ should be small. This decreases the running time by a corresponding exponential factor.

Our second and main observation is that the Fast Fourier Transform can be used as an efficient distinguishing algorithm. Instead of enumerating over several coordinates of the secret, and invoking the distinguishing algorithm for each candidate, the FFT algorithm allows us to check all of the guesses simultaneously at the cost of a single FFT and a single iteration over the vectors (x_j, y_j). A-priori, the use of FFT is limited because the cost of the FFT algorithm on multiple dimensions with modulus q is very expensive. Instead, we pass to a smaller modulus p where the cost of the FFT is smaller. Since the secret is short, using modulus switching introduces only a small error.

Finally, we note that we can further reduce the total running time by first enumerating over some coordinates of the secret, and only then applying the FFT distinguisher on some other coordinates of the secret.

To summarize, our attack consists of three stages. We first generate many pairs of vectors (x, y), where x and only a part of y are required to be short. Then, we enumerate over some coordinates of the secret. Finally, for each guess we apply an improved distinguishing algorithm that involves modulus switching and FFT over the new modulus.

1.3.3 Improved Short Vectors Sampling Procedure

The first part of our attack, as described above, consists of finding a list of short vectors in a lattice. This is usually done by running a BKZ reduction on the lattice, using sieving as an SVP oracle. To generate many short vectors, one simply outputs all the vectors found by the sieve on the final invocation on the first block.

We propose a more efficient algorithm for this task. Note that the last sieve, as described above, is inherently different from the rest of the sieves in the BKZ algorithm. When performing BKZ, lattice sieving is used as an SVP oracle, and only the shortest vector found by the sieving is used. In the last sieve, however, we use all of the resulting vectors. Therefore, we can use the "dimensions-for-free" optimization [Duc18] for all the sieves but the last. Our improved algorithm uses different block sizes for these two kinds of sieves – the first block size is used for obtaining a BKZ reduced basis; and in this case the "dimensions-for-free" optimization can be used. The second block size is used only for the last sieve, in which we use all of the output vectors.

1.3.4 Sieve Costs

Lattice sieving is a major component of our attack. The asymptotically best lattice sieving algorithm uses Locality Sensitive Filtering (LSF) [BDGL16]. In [AGPS20], Albrecht et al. calculate the cost of lattice sieving in the RAM model, a calculation used by following works for estimating the security of lattice based schemes [ABD$^+$21, DKL$^+$21].

We have performed a careful analysis of the sieve's running time, which reduced its cost in the RAM model by about 6 bits for cryptographically relevant block sizes. These improvements mainly stem from an improved random product code decoding algorithm and better parameter optimization. We also take into account the probability that close vector pairs are not detected by the LSF. This requires increasing the list sizes used, and means that the running time estimate taking this into account is slightly larger and more accurate.

Our improved random product code decoding algorithm is described in Section 6. The exact sieve cost analysis is beyond the scope of this paper, and will be published in future works.

1.4 Comparison to GJ21

In a recent paper, Guo and Johansson [GJ21] proposed an improvement of the dual lattice attack which is similar in spirit to our improvements. Despite the similarities, our attack still gives further improvements, as illustrated in Table 2. We explain some of the differences between our attacks. We use some of the notations from Section 1.3 above.

Table 2: Comparison between [GJ21] and our proposed attack in different models, see Section 7.1 for the definitions of the different models. All the costs are given in $\log_2$ of the gate-count metric.

	[GJ21]	This Work	
Asymptotic Dimensions-for-Free Model [Duc18]			
Sieve Cost Model	[AGPS20]	[AGPS20]	Section 6
Kyber512	148.3	**143.8**	**138.2**
Kyber768	207.3	**200.5**	**194.5**
Kyber1024	275.4	**266.0**	**259.3**
Dilithium-2	155.4	**153.4**	**147.3**
Dilithium-3	212.9	**210.5**	**203.7**
Dilithium-5	278.1	**273.3**	**266.2**
G6K Dimensions-for-Free Model [ADH$^+$19]			
Sieve Cost Model	[AGPS20]	[AGPS20]	Section 6
Kyber512	147.1	**143.1**	**137.5**
Kyber768	205.2	**199.5**	**193.5**
Kyber1024	272.3	**264.4**	**257.8**
Dilithium-2	153.8	**152.2**	**146.3**
Dilithium-3	210.4	**208.9**	**202.0**
Dilithium-5	274.4	**270.9**	**263.6**

In [GJ21], the authors partition the secret into three parts – a part of the secret over which they enumerate; a part of the secret on which an FFT-based distinguishing algorithm is used; and a part of the secret which is attacked similarly to the dual attack. Our attack uses the same partitioning. Moreover, their "two-step lattice reduction strategy" is the same as our improved short vectors sampling procedure. However, there are two key differences between the two attacks, which have significant effects on the running time.

First, although both attacks use FFT-based distinguishers, their inputs and algorithmic consequences are different. [GJ21] use FFT to efficiently enumerate over the second part of the secret (defined above) mod p, and modify the equations so that given that information, the equation error is reduced by a factor of $\frac{1}{p}$. We, however, use modulus switching to reduce the modulus of the equations to p. After the modulus switching, enumeration over the mod p part of the secret is equivalent to enumeration on the whole secret coordinate. The effect is dramatic, as it decreases the dimension of the lattice we reduce significantly (the change in the dimension can be as large as 80 in Kyber1024, for instance).

Second, [GJ21] do not aim to improve the cost of sieving, which as shown in the tables is an additional significant improvement in our attack.

1.5 Organization

The paper is organized as follows. In Section 2, we present some notations and preliminaries. In Section 3, we present the improved dual attack. In Section 4, we present the improved short vectors sampling procedure. In Section 5, we present the exact calculation of the attack parameters, and analyze the running time as a function of the parameters. In Section 6, we present the improved analysis of sieving in the RAM model. In Section 7, we present the exact parameters and running times of the algorithm in the RAM model when applied to Kyber, Dilithium, and Saber. In Section 8, we present future directions that might further reduce the running time of the attack.

2 Preliminaries

2.1 Notations

We denote matrices by uppercase letters, e.g. A, and vectors by lowercase letters, e.g. v. We treat vectors as column matrices. For vectors and matrices, x^T denotes the transpose of the vector or matrix. For a random variable χ, the notation $\chi(x)$ stands for the value of the probability distribution at x, $\mathbb{E}(\chi)$ denotes its expectation, $\mathrm{Var}(\chi)$ denotes its variance, and $H(\chi)$ denotes its Shannon entropy. We let $\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}\mathrm{d}t$ be the cumulative distribution function (cdf) of the standard normal distribution, and $\Phi^{-1} : [0,1) \to \mathbb{R}$ its inverse. For a value $x \in \mathbb{R}$, we denote $[x] = \mathrm{argmin}_{z\in\mathbb{Z}}\left(|x - z|\right)$ and $\{x\} = x - [x]$. We define $\left[n + \frac{1}{2}\right] = n + 1$ for $n \in \mathbb{Z}$, since $\frac{1}{2}$ could be rounded both ways. As part of lattice algorithms, one often enumerates over the values of a variable x. We denote the guesses of the value of x by $\tilde{x}$.

2.2 Lattices and Lattice Algorithms

We use standard definitions for lattices, lattice problems, and lattice algorithms. See e.g. [ACD+18, BSW18] for reference. We list here notations used in the paper which may not be standard as well as several standard assumptions.

Recall that lattice sieving is a method for obtaining a list of short vectors in a given lattice. For an integer β, we denote by $N_{\text{sieve}}(\beta)$ the number of vectors output by a sieve on a lattice of dimension β. We provide explicit estimates for this quantity when analyzing our attack in Section 7. We make the heuristic assumption that the sieve returns the $N_{\text{sieve}}(\beta)$ shortest vectors in the lattice.

For integers d and β, we denote by $\mathrm{BKZ}_{d,\beta}$ the BKZ algorithm for lattices of dimension d and block-size β. We recall the Geometric Series Assumption.

Assumption 2.1 (Geometric Series Assumption (GSA) [Sch03])**.** *Let Λ be some lattice of dimension d, and let $2 \leq \beta \leq d$ be some integer. Denote by $\{b_i^*\}_{i=1}^{d}$ the Gram-Schmidt's orthogonalization of the basis of Λ after running $\mathrm{BKZ}_{d,\beta}$. Assuming that $\beta \gg 200$ and $k < d - 2\beta$, then the first k vectors $(b_1^*, \ldots, b_k^*)$ follow the* Geometric-

Series-Assumption (GSA). *That is,*

$$\delta(\beta) := \|b_i^*\| \,/\, \|b_{i+1}^*\| \approx \left(\frac{\beta}{2\pi e} \cdot (\pi\beta)^{1/\beta}\right)^{\frac{1}{\beta-1}}$$

The GSA is considered to be inaccurate for some range of parameters due to the "head" and "tail" phenomena [CN11]. However, for our range of parameters, where $\beta \gg 200$, the head phenomenon is considered insignificant [BSW18]. The size of the tail is usually estimated to be around the block size. In our attack we rely on the GSA for the $O(\beta) < d - 2\beta$ first vectors, so we assume that we can ignore the tail phenomenon.

We recall the Gaussian Heuristic.

Assumption 2.2 (Gaussian Heuristic)**.** *Let Λ be a "random" lattice of dimension d and determinant 1. Let S be a centrally symmetric convex set. Then with high probability,*

$$\#\left(\Lambda \cap S\right) \approx \mathrm{Vol}\left(S\right)$$

In particular, the length of its shortest vector is

$$\lambda_1(\Lambda) \approx \sqrt{\frac{d}{2\pi e} \cdot (\pi d)^{1/d}}\ ,$$

and for every $c > 1$,

$$\#\left\{v \in \Lambda \mid \|v\| \leq c \cdot \lambda_1(\Lambda)\right\} \approx c^d\ .$$

The two conclusions follow by choosing S to be a ball of certain radius. The latter conclusion implies that the vectors returned by a sieve in a lattice of dimension β, which are the $N_{\text{sieve}}(\beta)$ shortest lattice vectors, have length at most $N_{\text{sieve}}(\beta)^{1/\beta}$ times the length of the shortest lattice vector.

We do not define formally what is a "random lattice", rather assume that unstructured lattices encountered during the algorithm follow the Gaussian Heuristic. Specifically, we assume that the projected lattices corresponding to block in the BKZ algorithm follow the Gaussian Heuristic. As in Assumption 2.1, and the discussion following it, we assume the Gaussian Heuristic for blocks in the head and body parts, but not the tail.

2.3 LWE

The Learning with Errors (LWE) problem, introduced by Regev [Reg09], is a computational problem whose presumed hardness gives rise to several cryptographic constructions.

Definition 2.3 (LWE)**.** Let $n, m, q \in \mathbb{N}$, and let χ_s, χ_e be distributions over $\mathbb{Z}/q\mathbb{Z}$. Denote by $\mathrm{LWE}_{n,m,q,\chi_s,\chi_e}$ the probability distribution on $(\mathbb{Z}/q\mathbb{Z})^{m\times n} \times (\mathbb{Z}/q\mathbb{Z})^m$ obtained by sampling the coordinates of the matrix $A \in (\mathbb{Z}/q\mathbb{Z})^{m\times n}$ independently and uniformly over $\mathbb{Z}/q\mathbb{Z}$, sampling the coordinates of $s \in (\mathbb{Z}/q\mathbb{Z})^n, e \in (\mathbb{Z}/q\mathbb{Z})^m$ independently from χ_s and χ_e respectively, and outputting $(A, As + e)$.

We define two problems:

- *Decision-LWE.* Distinguish the uniform distribution over $(\mathbb{Z}/q\mathbb{Z})^{m\times n} \times (\mathbb{Z}/q\mathbb{Z})^m$ from $\mathrm{LWE}_{n,m,q,\chi_s,\chi_e}$.
- *Search-LWE.* Given a sample from $\mathrm{LWE}_{n,q,\chi_s,\chi_e}$, recover s.

2.4 Discrete Fourier Transform

The Fourier Transform is an operation that, given a function $f : G \to \mathbb{C}$ on an abelian group G, evaluates $\hat{f}(\chi) := \sum_{g\in G} f(g)\chi(g)$ for all characters $\chi \in \hat{G}$ in the dual group $\hat{G}$ of G. For a function $f : (\mathbb{Z}/q\mathbb{Z})^n \to \mathbb{C}$, its Fourier transform $\hat{f}$ is given by

$$\hat{f}(v) = \sum_u e^{\frac{2\pi i}{q} v^T u} f(u).$$

The Fourier transform of a function $f : (\mathbb{Z}/q\mathbb{Z})^n \to \mathbb{C}$ can be calculated in time $O(nq^n)$ using the Fast Fourier Transform (FFT) algorithm.

2.5 Modular Gaussian Distribution

Let $\sigma > 0$. For all $x \in \mathbb{R}$, the density of the centered Gaussian distribution with standard deviation σ is defined as $\rho_\sigma(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$. The Modular Gaussian distribution mod q is a probability distribution over $\mathbb{Z}/q\mathbb{Z}$, defined as:

$$\rho_{q,\sigma}(t) = \frac{1}{C_{q,\sigma}} \sum_{x\in t+q\mathbb{Z}} e^{-\frac{x^2}{2\sigma^2}} ,$$

where $C_{q,\sigma} = \sum_{t\in\mathbb{Z}} e^{-\frac{x^2}{2\sigma^2}}$ is a normalizing factor.

Theorem 2.4 ([SW71])**.** *The discrete Fourier transform of $\rho_{q,\sigma}$ is $\widehat{\rho_{q,\sigma}} = \frac{1}{\rho_{q,\sigma'}(0)} \rho_{q,\sigma'}$ where $\sigma' = \frac{q}{2\pi\sigma}$.*

We state here bounds on the first and second Fourier coefficient which will later be useful.

Lemma 2.5. *Let $q \in \mathbb{N}$ be some modulus, and $\sigma > 0$ some real number. Then the following inequality holds*

$$\widehat{\rho_{q,\sigma}}(1) \geq e^{-2\left(\frac{\pi\sigma}{q}\right)^2} .$$

If further $\sigma \geq \sqrt{\frac{q\log 2}{8\pi^2}}$, then

$$\widehat{\rho_{q,\sigma}}(2) \leq 2e^{-8\left(\frac{\pi\sigma}{q}\right)^2} .$$

Proof. Fix q and σ. By Theorem 2.4, we have that

$$\widehat{\rho_{q,\sigma}}(1) = \frac{\rho_{q,\sigma'}(1)}{\rho_{q,\sigma'}(0)} , \qquad \sigma' = \frac{q}{2\pi\sigma} .$$

Hence,

$$\widehat{\rho_{q,\sigma}}\,(1) = \frac{\rho_{q,\sigma'}\,(1)}{\rho_{q,\sigma'}\,(0)} = \frac{\sum_{x\in 1+q\mathbb{Z}} e^{-\frac{x^2}{2\sigma'^2}}}{\sum_{x\in q\mathbb{Z}} e^{-\frac{x^2}{2\sigma'^2}}} = \frac{\sum_{r\in\mathbb{Z}} e^{-\frac{(1+rq)^2}{2\sigma'^2}}}{\sum_{r\in\mathbb{Z}} e^{-\frac{(rq)^2}{2\sigma'^2}}}\ .$$

Note that

$$e^{-\frac{(rq)^2}{2\sigma'^2}} = e^{\frac{1}{2\sigma'^2}} e^{-\frac{(rq)^2+1}{2\sigma'^2}} = e^{\frac{1}{2\sigma'^2}} \sqrt{e^{-\frac{(rq)^2+2rq+1}{2\sigma'^2}} \cdot e^{-\frac{(rq)^2-2rq+1}{2\sigma'^2}}} =$$
$$= e^{\frac{1}{2\sigma'^2}} \sqrt{e^{-\frac{(1+rq)^2}{2\sigma'^2}} \cdot e^{-\frac{(1-rq)^2}{2\sigma'^2}}} \le \frac{e^{\frac{1}{2\sigma'^2}}}{2}\left(e^{-\frac{(1+rq)^2}{2\sigma'^2}} + e^{-\frac{(1-rq)^2}{2\sigma'^2}}\right)\ .$$

Therefore,

$$\frac{\sum_{r\in\mathbb{Z}} e^{-\frac{(1+rq)^2}{2\sigma'^2}}}{\sum_{x\in q\mathbb{Z}} e^{-\frac{(rq)^2}{2\sigma'^2}}} \ge 2e^{-\frac{1}{2\sigma'^2}} \frac{\sum_{r\in\mathbb{Z}} e^{-\frac{(1+rq)^2}{2\sigma'^2}}}{\sum_{r\in\mathbb{Z}}\left(e^{-\frac{(1+rq)^2}{2\sigma'^2}} + e^{-\frac{(1-rq)^2}{2\sigma'^2}}\right)} =$$
$$= 2e^{-\frac{1}{2\sigma'^2}} \frac{\sum_{r\in\mathbb{Z}} e^{-\frac{(1+rq)^2}{2\sigma'^2}}}{\sum_{r\in\mathbb{Z}} 2e^{-\frac{(1+rq)^2}{2\sigma'^2}}} = e^{-\frac{1}{2\sigma'^2}} = e^{-2\left(\frac{\pi\sigma}{q}\right)^2}\ .$$

For the second coefficient we similarly have

$$\frac{\rho_{q,\sigma'}\,(2)}{\rho_{q,\sigma'}\,(0)} = \frac{\sum_{x\in 2+q\mathbb{Z}} e^{-\frac{x^2}{2\sigma'^2}}}{\sum_{x\in q\mathbb{Z}} e^{-\frac{x^2}{2\sigma'^2}}} \le \frac{\sum_{r\in\mathbb{Z}} e^{-\frac{(2+rq)^2}{2\sigma'^2}}}{1} \le$$
$$\le e^{-\frac{4}{2\sigma'^2}} \sum_{r=0}^{\infty} e^{-\frac{4rq}{2\sigma'^2}} = e^{-\frac{4}{2\sigma'^2}} \frac{1}{1-e^{-\frac{2q}{\sigma'^2}}} \le$$
$$\le 2e^{-8\left(\frac{\pi\sigma}{q}\right)^2}$$

where the last inequality follows since

$$\sigma' \le \sqrt{\frac{2q}{\log 2}} \Leftrightarrow \sigma \ge \sqrt{\frac{q\log 2}{8\pi^2}}\ .$$

□

3 FFT-Based Dual Lattice Attack

In this section we describe the main idea of the FFT based dual lattice attack. We begin with an informal sketch of its main components in Section 3.1, and in Section 3.2 we present a more thorough description of the attack. The analysis of the attack is presented in Section 5.

3.1 Motivation

We now present the main idea of the FFT-based dual attack presented in Section 3.2. Our starting point is a generalization of the dual attack, along the lines of [EJK20].

The central part of the dual attack is a method for distinguishing between a pair $(A, b) \in (\mathbb{Z}/q\mathbb{Z})^{m\times n} \times (\mathbb{Z}/q\mathbb{Z})^m$ sampled from the uniform distribution over $(\mathbb{Z}/q\mathbb{Z})^{m\times n} \times (\mathbb{Z}/q\mathbb{Z})^m$, and a pair sampled from an LWE distribution, that is $b \equiv As+e \mod q$ for some $s \in (\mathbb{Z}/q\mathbb{Z})^n$ sampled from χ_s^n and $e \in (\mathbb{Z}/q\mathbb{Z})^m$ sampled from χ_e^m. We suppose that χ_s and χ_e are supported on small values with high probability. We find many short vectors $x_j \in (\mathbb{Z}/q\mathbb{Z})^m$ such that $y_j^T = x_j^T A \in (\mathbb{Z}/q\mathbb{Z})^n$ are also short, and calculate the list of values $\left(x_j^T b\right)$. For a random pair (A, b), these values are distributed uniformly. For an LWE pair, we have $x_j^T b = y_j^T s + x_j^T e$ which are approximately distributed according to a modular Gaussian distribution. Given sufficiently many samples, we can distinguish between the two distributions.

In order to perform a key recovery attack, we partition s into two components: $s^T = \left(s_1^T \| s_2^T\right)$. We partition the matrix A analogously: $A = (A_1 \| A_2)$, so that

$$b = As + e = A_1 s_1 + A_2 s_2 + e.$$

If s_1 were known, we could create a new LWE problem

$$b' = A_2 s_2 + e \ ,$$

where $b' = b - A_1 s_1$. However, s_1 is unknown. Nonetheless, we may enumerate over s_1 and use the distinguishing attack on the pairs $(A_2, b - A_1\tilde{s}_1)$ for every guess $\tilde{s}_1$ of s_1 to determine the correct one, for which the pairs come from an LWE distribution. In the standard dual attack, s_1 consists of a single coordinate of s. [EJK20] suggested letting s_1 consist of an arbitrary number of coordinates. This generalized algorithm is described in Algorithm 1.

Algorithm 1 Generalized Dual Attack

```
Input: LWE parameters $(n, m, q, \chi_s, \chi_e)$, integers $k_1, k_2$ such that $k_1 + k_2 = n$, and an LWE
    pair $(A, b) \in (\mathbb{Z}/q\mathbb{Z})^{m\times n} \times (\mathbb{Z}/q\mathbb{Z})^m$.
Output: The first $k_1$ coordinates of $s$.
1: Find sufficiently many vectors $(x_j, y_j)$ such that $x_j^T A = \left(y_{j,1}^T \| y_{j,2}^T\right)$ and the pair $(x_j, y_{j,2})$
   is short.
2: for every value $\tilde{s}_1$ of $s_1$ do
3:     Calculate the list of values $\left(x_j^T (b - A_1\tilde{s}_1)\right)$.
4:     if the distribution of these values is Gaussian rather than uniform then
5:         return $\tilde{s}_1$.
6:     end if
7: end for
```

Our main improvement to this generalized algorithm is a method of using the FFT algorithm to test whether the values of $\left(x_j^T (b - A_1\tilde{s}_1)\right)$ are sampled from a modular

Gaussian distribution for all possible values of s_1 at the same time. As stated in [EJK20], an efficient way to determine if a list of values $w_1, w_2, \ldots, w_D \in \mathbb{Z}/q\mathbb{Z}$ has been sampled from a modular Gaussian distribution or a uniform distribution is by calculating the quantity $\sum_j e^{2\pi i w_j/q}$ whose real value is expected to be large when w_j are distributed according to a centered Gaussian distribution, and close to zero when w_j are distributed uniformly, provided sufficiently many samples.

In our case, the values are $w_j = x_j^T \left(b - A_1 \tilde{s}_1\right)$, which we write as $w_j = u_j^T \tilde{s}_1 + c_j$ where $u_j = -y_{j,1}$ and $c_j = x_j^T b$ are known. The aforementioned quantity equals

$$\sum_j e^{w_j \frac{2\pi i}{q}} = \sum_j e^{\left(u_j^T \tilde{s}_1 + c_j\right) \frac{2\pi i}{q}} \tag{1}$$

We recall that the FFT algorithm, given function $f : (\mathbb{Z}/q\mathbb{Z})^n \to \mathbb{C}$, computes $\sum_j e^{\left(u_j^T v\right) \frac{2\pi i}{q}} f\left(u_j\right)$ for all vectors $v \in (\mathbb{Z}/q\mathbb{Z})^d$. This allows us to evaluate Eq. (1) for all values of $\tilde{s}_1$ simultaneously, by setting $f(u) = \sum_{j : u_j = u} e^{\frac{2\pi i}{q} c_j}$.

However, this would take $O\left(k_1 q^{k_1}\right)$ operations, and quickly become infeasible. Additionally, it would calculate Eq. (1) for all values $\tilde{s}_1 \in (\mathbb{Z}/q\mathbb{Z})^{k_1}$, while the secret distribution likely permits significantly fewer possible values.

To accelerate the FFT we transform the problem from distinguishing between uniform and Gaussian distributions in $\mathbb{Z}/q\mathbb{Z}$ to distinguishing between those distributions in $\mathbb{Z}/p\mathbb{Z}$, for some $p < q$. We transfer the problem to $\mathbb{Z}/p\mathbb{Z}$ by rounding the equations: We multiply each equation by $\frac{p}{q}$, and round $\frac{p}{q} u_j$ to the closest integer. Recall the rounding notations $[a], \{a\}$ as defined in Section 2.1. The original samples are defined in $(\mathbb{Z}/q\mathbb{Z})$:

$$w_j \equiv_q u_j^T \tilde{s}_1 + c_j \ .$$

We explicitly write the modular equation, so it is defined in $\mathbb{Z}$:

$$w_j = u_j^T \tilde{s}_1 + c_j + aq \ .$$

When multiplying by $\frac{p}{q}$, the equations are defined in $\frac{p}{q}\mathbb{Z}$:

$$\frac{p}{q} w_j = \frac{p}{q} u_j^T \tilde{s}_1 + \frac{p}{q} c_j + ap \ .$$

The FFT algorithm works on functions $(\mathbb{Z}/p\mathbb{Z})^n \to \mathbb{C}$. Since the coordinates of $\frac{p}{q} u_j^T$ are not integral, they cannot be used directly by the algorithm. We separate it to the integer and the fractional components, and write:

$$\frac{p}{q} w_j = \left[\frac{p}{q} u_j\right]^T \tilde{s}_1 + \left\{\frac{p}{q} u_j\right\}^T \tilde{s}_1 + \frac{p}{q} c_j + ap \tag{2}$$

The vectors $\left[\frac{p}{q} u_j\right]$ are defined in $(\mathbb{Z}/p\mathbb{Z})^{k_1}$. Thus, we can perform FFT to calculate the values

$$\sum_j e^{\left(\left[\frac{p}{q} u_j\right]^T \tilde{s}_1 + \frac{p}{q} c_j\right) \frac{2\pi i}{p}} = \sum_j e^{\left(u_j^T \tilde{s}_1 + c_j\right) \frac{2\pi i}{q} - \left(\left\{\frac{p}{q} u_j\right\}^T \tilde{s}_1\right) \frac{2\pi i}{p}} \tag{3}$$

which is the same as Eq. (1) up to a small factor involving the fractional part $\left\{\frac{p}{q}u_j\right\}^T \tilde{s}_1$, which essentially becomes part of the Gaussian error term. This is the essence of our improved algorithm.

Additional algorithmic improvements include externally enumerating over a third part of the secret, as well as an improved sampling method for short vectors $(x_j, y_{j,2})$ described in section 4. The full algorithm is described below in detail.

3.2 Formal Description

In this section, we give a formal description of the algorithm, while deferring the analysis to Section 5. We first describe the algorithm formally, then explain what each part does. We remark that, by ignoring some of the LWE equations, we may replace the number of equations m with $m' \leq m$ of our choice. Without loss of generality, we present the algorithm for $m' = m$, but optimize the parameters in the general setting.

The parameters our algorithm uses are

- D: The number of vectors used in the dual attack.
- C: The cutoff used to differentiate between incorrect and correct guesses to s_{enum} and s_{fft}.
- β_1, β_2: Parameters used by the short vectors sampling algorithm.
- k_{enum}: The number of secret coordinates over which we enumerate directly.
- k_{fft}: The number of secret coordinates over which we enumerate using FFT.
- k_{lat}: The number of remaining secret coordinates.
- p: The modulus in which we perform FFT.

We are given a matrix $A \in (\mathbb{Z}/q\mathbb{Z})^{m \times n}$, and a vector $b \in (\mathbb{Z}/q\mathbb{Z})^m$ that satisfies $b = As + e$, where the coordinates of s and e are sampled according to some known distributions χ_s and χ_e respectively, which have small variances σ_s^2 and σ_e^2 respectively. We partition s into three components:

$$s = \begin{pmatrix} s_{\text{enum}} \\ s_{\text{fft}} \\ s_{\text{lat}} \end{pmatrix},$$

where s_{enum} has k_{enum} coordinates, s_{fft} has k_{fft} coordinates, and s_{lat} has $k_{\text{lat}} = n - k_{\text{enum}} - k_{\text{fft}}$ coordinates. We similarly partition A into three components:

$$A = \begin{pmatrix} A_{\text{enum}} & A_{\text{fft}} & A_{\text{lat}} \end{pmatrix},$$

so that $As = A_{\text{enum}}s_{\text{enum}} + A_{\text{fft}}s_{\text{fft}} + A_{\text{lat}}s_{\text{lat}}$.

We define the matrix:

$$B = \begin{pmatrix} \alpha I_m & 0 \\ A_{\text{lat}}^T & qI_{k_{\text{lat}}} \end{pmatrix},$$

where α is a constant equal to $\frac{\sigma_e}{\sigma_s}$ and is used for normalization in the case that s, e have different distributions. We find D short vectors in the column space of B using

Algorithm 2 Improved Dual Attack

```
Input: LWE parameters $(n, m, q, \chi_s, \chi_e)$, integers $\beta_1, \beta_2 \le d$, integers $k_{\text{enum}}, k_{\text{fft}}, k_{\text{lat}}$ such
    that $k_{\text{enum}} + k_{\text{fft}} + k_{\text{lat}} = n$, an integer $p \le q$, an integer $D$, a real number $C$, and an
    LWE pair $(A, b) \in (\mathbb{Z}/q\mathbb{Z})^{m \times n} \times (\mathbb{Z}/q\mathbb{Z})^m$.
Output: The first $k_{\text{enum}}$ coordinates of $s$.
 1: Compute $A_{\text{lat}}$ composed of the last $k_{\text{lat}}$ columns of $A$.
 2: Compute the matrix $B = \begin{pmatrix} \alpha I_m & 0 \\ A_{\text{lat}}^T & qI_{k_{\text{lat}}} \end{pmatrix}$, where $\alpha = \frac{\sigma_e}{\sigma_s}$.
 3: Run the short vectors sampling algorithm (Algorithm 3) on the basis $B$ with parameters
    $\beta_1, \beta_2, D$, to get a list $L$ of $D$ short vectors.
 4: for every value of $\tilde{s}_{\text{enum}}$, in descending order of probability according to secret distribution
    do
 5:     Initialize a table $T$ of dimensions $\underbrace{p \times p \times \cdots \times p}_{k_{\text{fft}} \text{times}}$
 6:     for every short vector $(\alpha x_j, y_{\text{lat}})$ in $L$ do
 7:         Compute $y_{j,\text{fft}} = x_j^T A_{\text{fft}}$.
 8:         Compute $y_{j,\text{enum}} = x_j^T A_{\text{enum}}$.
 9:         Add $e^{\left(x_j^T b - y_{j,\text{enum}}^T \tilde{s}_{\text{enum}}\right)\frac{2\pi i}{q}}$ to cell $\left[\frac{p}{q} y_{j,\text{fft}}\right]$ of T.
10:     end for
11:     Perform FFT on $T$.
12:     if for any $\tilde{s}_{\text{fft}}$, the real part of $\frac{1}{\psi(\tilde{s}_{\text{fft}})} T[s_{\text{fft}}]$ is larger than $C$ then
13:         return $\tilde{s}_{\text{enum}}$.
14:     end if
15: end for
```

our improved short vectors sampling procedure (Algorithm 3). Each such vector can be partitioned to two parts:

$$v \equiv_q \begin{pmatrix} \alpha x \\ A_{\text{lat}}^T x \end{pmatrix} .$$

For each such vector, we have $x^T b = x^T A s + x^T e$.
Writing $x^T A = y^T = \left(y_{\text{enum}}^T \| y_{\text{fft}}^T \| y_{\text{lat}}^T\right)$, we get

$$y_{\text{enum}}^T s_{\text{enum}} + y_{\text{fft}}^T s_{\text{fft}} - x^T b = -y_{\text{lat}}^T s_{\text{lat}} - x^T e \ .$$

To perform the modulus switching, we multiply the equation by $\frac{p}{q}$ and round the y_{fft}. We get

$$\left[\frac{p}{q} y_{\text{fft}}\right]^T s_{\text{fft}} + \frac{p}{q} y_{\text{enum}}^T s_{\text{enum}} - \frac{p}{q} x^T b = -\frac{p}{q} y_{\text{lat}}^T s_{\text{lat}} - \frac{p}{q} x^T e - \left\{\frac{p}{q} y_{\text{fft}}\right\}^T s_{\text{fft}} \ .$$

The right hand term is approximately distributed according to a modular Gaussian distribution. Thus, for the correct guess $(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}}) = (s_{\text{enum}}, s_{\text{fft}})$, the term $\left[\frac{p}{q} y_{\text{fft}}\right]^T \tilde{s}_{\text{fft}} + \frac{p}{q} y_{\text{enum}}^T \tilde{s}_{\text{enum}} - \frac{p}{q} x^T b$ is distributed according to a modular Gaussian distribution, while for an incorrect guess $(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}}) \neq (s_{\text{enum}}, s_{\text{fft}})$ it is distributed uniformly.

The modular Gaussian distribution induced by the correct guess is not always centered. To center it, we use a normalization constant $\psi(\tilde{s}_{\text{fft}})$ with $|\psi(\tilde{s}_{\text{fft}})| = 1$, which will be calculated in Lemma 5.4.

Therefore, the expectation of the quantity

$$\frac{1}{\psi(\tilde{s}_{\text{fft}})} \sum_j \exp\left(\left(\left[\frac{p}{q} y_{j,\text{fft}}\right]^T \tilde{s}_{\text{fft}} + \frac{p}{q} y_{j,\text{enum}}^T \tilde{s}_{\text{enum}} - \frac{p}{q} x_j^T b\right) \cdot \frac{2\pi i}{p}\right) \tag{4}$$

has a large real value when $(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}}) = (s_{\text{enum}}, s_{\text{fft}})$, and is close to zero otherwise. We thus define the score of a pair $(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}})$ as

$$F(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}}) = \Re\left(\frac{1}{\psi(\tilde{s}_{\text{fft}})} \sum_j \exp\left(\left(\left[\frac{p}{q} y_{j,\text{fft}}\right]^T \tilde{s}_{\text{fft}} + \frac{p}{q} y_{j,\text{enum}}^T \tilde{s}_{\text{enum}} - \frac{p}{q} x_j^T b\right) \cdot \frac{2\pi i}{p}\right)\right) \tag{5}$$

We use this to determine the correct value of s_{enum}: For each $\tilde{s}_{\text{enum}}$ we calculate $F(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}})$ for all values of s_{fft} via FFT. If for a given $\tilde{s}_{\text{enum}}$, the score of any $(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}})$ is larger than the cutoff C, we determine that $\tilde{s}_{\text{enum}} = s_{\text{enum}}$, and otherwise $\tilde{s}_{\text{enum}} \neq s_{\text{enum}}$.

Having determined the first k_{enum} coordinates of the key, we continue to recover the complete key by recursively solving the resulting LWE problem, which is significantly smaller.

To simplify the analysis, we do not assume the recovery of s_{fft}. Although the above sum is expected to have large real value when s_{fft} is guessed correctly, it may also be large for certain wrong guesses of s_{fft}, provided s_{enum} is guessed correctly. This does not concern us since we only wish to recover s_{enum}.

4 Short Vectors Sampling Procedure

In this section we present and analyze our improved procedure for sampling multiple short lattice vectors, a key part of the dual attack. In an independent and concurrent work, Guo and Johansson [GJ21] proposed the same sampling procedure. Our presentation and analysis is slightly different, and is presented here for completeness.

Informally, to find many short vectors in the lattice Λ, we first run the BKZ algorithm using a sieve with block size β_1 as the SVP oracle to find a reduced basis of Λ. Since the output of this stage of the algorithm is a reduced basis, we can employ all known BKZ optimizations, such as the "dimensions-for-free" trick [Duc18]. To find many short vectors, we perform lattice sieving on the sublattice spanned by the first β_2 vectors of the reduced basis, this time without using dimensions-for-free, as we use all the vectors found by the sieve. Performing lattice sieving on a lattice of rank β_2 returns $N_{\text{sieve}}(\beta_2)$ vectors. To sample $D > N_{\text{sieve}}(\beta_2)$ vectors, we repeat the algorithm $\left\lceil \frac{D}{N_{\text{sieve}}(\beta_2)} \right\rceil$ times, using random initial bases. Heuristically, running BKZ on random initial bases returns random reduced bases, and new vectors are found in each iteration. The advantage of the improved short vectors sampling procedure over standard BKZ is that we use different block sizes for lattice sieving where dimensions-for-free can be employed and where it cannot be, and can optimize both to reduce the overall cost of the algorithm.

The algorithm is described formally in Algorithm 3. We analyze its quality by computing the number of vectors found by the algorithm and their length.

Recall the notations from Section 2.2 for $N_{\text{sieve}}(\beta)$, the number of sieve results, and $\delta(\beta)$, the ratio in the Geometric Series Assumption.

Algorithm 3 Short Vectors Sampling Procedure

Input: A basis $B = (b_1, \ldots, b_d)$ for a lattice and integers $\beta_1, \beta_2 \leq d$ and D.
Output: A list of length at least D of vectors from the lattice.

```
1: for $i = 1, \ldots, \left\lceil \frac{D}{N_{\text{sieve}}(\beta_2)} \right\rceil$ do
2:     Randomize the basis $B$.
3:     Run $\text{BKZ}_{d,\beta_1}$ on $B$ to obtain a reduced basis $(b'_1, \ldots, b'_d)$.
4:     Run a sieve of dimension $\beta_2$ on the sublattice $(b'_1, \ldots, b'_{\beta_2})$ to obtain a list of vectors,
       and add them to $L$.
5: end for
6: return $L$.
```

Lemma 4.1 (Short Vectors Sampling Complexity)**.** *Let B be a basis for a d-dimensional lattice, and (β_1, β_2, D) be integer parameters for Algorithm 3. Then the running time of the algorithm is*

$$\left\lceil \frac{D}{N_{\text{sieve}}(\beta_2)} \right\rceil \cdot \left(T_{\text{BKZ}}(d, \beta_1) + T_{\text{sieve}}(\beta_2)\right) .$$

The proof of the lemma is clear from the description of the algorithm. We give a fine-grained analysis of the running time in the RAM model in Section 7.

Lemma 4.2 (Short Vectors Sampling Procedure's Quality and Correctness)**.** *Let Λ be a d-dimensional lattice. Let β_1, β_2 be the basis reduction block size, and the vectors sampling block size, respectively. Then Algorithm 3 outputs at least D vectors of length*

$$\det(\Lambda)^{1/d} \cdot N_{\text{sieve}}(\beta_2)^{1/\beta_2} \cdot \sqrt{\frac{\beta_2}{2\pi e} \cdot (\pi\beta_2)^{1/\beta_2} \cdot \delta(\beta_1)^{\frac{d-\beta_2}{2}}}\ .$$

Proof. Consider the lattice Λ' given by the basis $(b'_1, \ldots, b'_{\beta_2})$. Its volume is given by

$$\det\left(\Lambda'\right) = \prod_{i=1}^{\beta_2} \|b_i^*\| = \det(\Lambda)^{\beta_2/d} \cdot \delta(\beta_1)^{\beta_2 \frac{d-\beta_2}{2}}\ ,$$

where the last equality follows from the GSA (Assumption 2.1). Therefore, by the Gaussian Heuristic (Assumption 2.2),

$$\lambda_1(\Lambda') = \det(\Lambda)^{1/d} \cdot \sqrt{\frac{\beta_2}{2\pi e} \cdot (\pi\beta_2)^{1/\beta_2} \cdot \delta(\beta_1)^{\frac{d-\beta_2}{2}}}\ .$$

Step 4 in the algorithm is a sieve on Λ'. The sieve outputs the $N_{\text{sieve}}(\beta_2)$ shortest lattice vectors, so their length is at most $N_{\text{sieve}}(\beta_2)^{1/\beta_2} \cdot \lambda_1(\Lambda')$ (see Assumption 2.2 and the discussion following it). In each iteration we add $N_{\text{sieve}}(\beta_2)$ new vectors, therefore, after $\left\lceil \frac{D}{N_{\text{sieve}}(\beta_2)} \right\rceil$ iterations, the list L contains at least D vectors of the required length. $\square$

Remark 4.3 (Comments on randomization)**.** *We note that for the range of parameters relevant for the NIST candidates considered, which are given in Section 7.3, $N_{\text{sieve}}(\beta_2) \approx D$. That means that a single iteration yields enough short vectors, and we do not need to randomize the basis. See also Section 8.3 for further relevant future directions.*

We make the following assumption regarding the output of the algorithm.

Assumption 4.4 ([CN11, EJK20])**.** *Let ℓ be the expected length of the vectors returned by Algorithm 3, as described in Lemma 4.2. Then the coordinates of the returned vectors have approximately independent Gaussian distributions with mean 0 and standard deviation $\ell/\sqrt{d}$.*

5 Analysis

In this section we analyze the time complexity of our algorithm. The algorithm, as presented in Section 3, has several parameters. We first describe the complexity of the attack in terms of these parameters. Then, we show how to set the parameters (specifically the number of samples required for distinguishing) to guarantee its success with high probability. We note that our analysis is exact, and not asymptotic, as we use it for the fine-grained RAM model. Finally, we state the asymptotic cost of the attack as a function of the desired success probability.

We follow the notations from Section 3.

5.1 Running Time

We now describe the number of operations performed by the algorithm, as a function of its parameters.

Recall that the algorithm iterates over parts of the secret in descending order of probabilities. We therefore analyze the expected time complexity (expectation over the choice of the secret). The algorithm can be converted into a Monte Carlo Algorithm using the standard transformation.

Theorem 5.1. *Let $(n, m, q, \chi_s, \chi_e)$ be LWE parameters and $(\beta_1, \beta_2, k_{\text{enum}}, k_{\text{fft}}, k_{\text{lat}}, p, D, C)$ be parameters for Algorithm 2. The expected running time is*

$$T_{\text{sample}_{\beta_1,\beta_2,D}}(m + k_{\text{lat}}) + 2^{k_{\text{enum}} H(\chi_s)} \cdot (T_{\text{fft}}(k_{\text{fft}}, p) + T_{\text{table}}) \ ,$$

where

- *$T_{\text{sample}_{\beta_1,\beta_2,D}}(d)$ is the time complexity of Algorithm 3 with parameters (β_1, β_2, D) on an input basis of dimension d.*
- *$T_{\text{fft}}(k, p)$ is the time complexity of FFT of dimension k with modulus p.*
- *T_{table} is the time complexity of updating the table in each iteration.*

Proof. The attack has three stages:

1. Finding short vectors using the short vectors sampling procedure.
2. Enumerating over possible values of s_{enum}.
3. Calculating the score for all values of $\tilde{s}_{\text{fft}}$ given the enumerated $\tilde{s}_{\text{enum}}$.

We find D short vectors using our short vectors sampling algorithm (Algorithm 3) on a lattice of dimension $m + k_{\text{lat}}$, taking $T_{\text{sample}_{\beta_1,\beta_2,D}}(m + k_{\text{lat}})$ operations.

After finding short vectors, we enumerate over guesses for s_{enum}. It has k_{enum} coordinates, each with entropy $H(\chi_s)$. Therefore, the expected number of steps before reaching s_{enum} is at most $2^{k_{\text{enum}} H(\chi_s)}$.

For each possible value of s_{enum}, we perform an FFT on a table of size $p^{k_{\text{fft}}}$, taking $T_{\text{fft}}(k_{\text{fft}}, p) = O(k_{\text{fft}} p^{k_{\text{fft}}})$ operations. Searching the table for an $\tilde{s}_{\text{fft}}$ with value above the threshold takes only $O(p^{k_{\text{fft}}})$ operations, and is insignificant compared to the FFT calculation. We then update the table for the next iteration. This update is performed according to Section 5.4 and takes T_{table} operations.

Combining all the terms, the total number of operations performed by the algorithm is

$$T_{\text{sample}_{\beta_1,\beta_2,D}}(m + k_{\text{lat}}) + 2^{k_{\text{enum}} H(\chi_s)} \cdot (T_{\text{fft}}(k_{\text{fft}}, p) + T_{\text{table}}) \ .$$

□

5.2 Required Number of Samples

In this section we calculate the parameters D, the number of samples required to launch the dual attack, and C, the cutoff used to distinguish between correct and incorrect guesses, as a function of the desired success probability.

Our calculation is exact and not asymptotic, which allows accurate estimations. We first determine the value of the parameters as a function of the secret and the error. We then analyze how to set the parameters as a function of the distributions and not specific values.

Notations. For the sake of clarity, we introduce several notations that will make the contribution of different factors to the final expression clearer. In what follows, let $(n, m, q, \chi_s, \chi_e)$ be LWE parameters, and let $(\beta_1, \beta_2, k_{\text{enum}}, k_{\text{fft}}, k_{\text{lat}}, p)$ be a partial tuple of parameters for Algorithm 2. We fix a secret vector and an error term $(s, e) \in (\mathbb{Z}/q\mathbb{Z})^n \times (\mathbb{Z}/q\mathbb{Z})^m$. We denote by ℓ the expected length of the vectors returned by Algorithm 3. We first define

$$D_{\text{eq}} = e^{4\left(\frac{\pi\tau}{q}\right)^2} ,$$

where

$$\tau^2 = \frac{\alpha^{-2} \|e\|^2 + \|s_{\text{lat}}\|^2}{m + k_{\text{lat}}} \ell^2 ,$$
$$\alpha = \frac{\sigma_e}{\sigma_s} .$$

Informally, D_{eq} is an exponential factor which comes from the clean Fourier coefficient of the error in the dual attack equations. Moreover, we define

$$D_{\text{round}} = \left(\prod_{\substack{t=1 \\ s_t \neq 0}}^{k_{\text{fft}}} \left(\frac{\sin\left(\frac{\pi s_t}{p}\right)}{\frac{\pi s_t}{p}} \right) \right)^{-2} ,$$

where we write $s_{\text{fft}} = (s_1, \ldots, s_{k_{\text{fft}}})$. Informally, D_{round} is an exponential factor which accounts for the effect of rounding on the Fourier coefficient after performing a modulus switch. Furthermore, we define

$$D_{\text{arg}} = \frac{1}{2} + e^{-8\left(\frac{\pi\tau}{q}\right)^2} ,$$

where $\tau^2 = \frac{\alpha^{-2}\|e\|^2 + \|s_{\text{lat}}\|^2}{m + k_{\text{lat}}} \ell^2$ as above. Informally, $D_{\text{arg}} \approx \frac{1}{2}$ is a constant improvement factor obtained by considering the complex argument of the Fourier coefficient rather than just its magnitude.

We denote by $N_{\text{enum}}(s_{\text{enum}})$ the number of guesses $\tilde{s}_{\text{enum}}$ with probabilities larger than the probability of s_{enum}, according to the distribution χ_s. This is the number of candidates $\tilde{s}_{\text{enum}}$ that will be enumerated upon before the correct guess is reached.

Lastly, suppose that $0 < \mu < 1$ is a target failure probability for the algorithm. We define

$$D_{\text{fpfn}}(\mu) = (\phi_{\text{fp}}(\mu) + \phi_{\text{fn}}(\mu))^2 ,$$
$$\phi_{\text{fp}}(\mu) = \Phi^{-1}\left(1 - \frac{\mu}{2 \cdot N_{\text{enum}}(s_{\text{enum}}) \cdot p^{k_{\text{fft}}}}\right) ,$$
$$\phi_{\text{fn}}(\mu) = \Phi^{-1}\left(1 - \frac{\mu}{2}\right) ,$$

where Φ^{-1} is the inverse of the cumulative distribution function of the standard normal distribution. Informally, D_{fpfn} is a polynomial factor that ensures small false positive and false negative rates. That is, ensures detecting the correct value of s_{enum}, and only the correct one, with probability $1-\mu$.

5.2.1 Parameters as a Function of the Secret

Having defined several relevant notations, we state an upper bound on the required number of samples D for the algorithm to succeed with high probability.

Theorem 5.2. *Let $(n, m, q, \chi_s, \chi_e)$ be LWE parameters, $(\beta_1, \beta_2, k_{\text{enum}}, k_{\text{fft}}, k_{\text{lat}}, p)$ be a partial tuple of parameters for Algorithm 2, and let $0 < \mu < 1$ be the desired failure probability.*

Fix $(s, e) \in (\mathbb{Z}/q\mathbb{Z})^n \times (\mathbb{Z}/q\mathbb{Z})^m$. Then Algorithm 2 with parameters

$$D \geq D_{\text{eq}} \cdot D_{\text{round}} \cdot D_{\text{arg}} \cdot D_{\text{fpfn}}(\mu) \ ,$$
$$C = \phi_{\text{fp}}(\mu)\sqrt{D_{\text{arg}} \cdot D} \ ,$$

outputs the correct s_{enum} with probability at least $1-\mu$.

We note that Theorem 5.2 gives bounds for a given value of (s, e). In order to achieve an advantage of $\frac{1}{2}$ for the attacker, we use parameters for which the attack succeeds for at least half of the values of s. Since $\log(D)$ is approximately normally distributed, the logarithmic mean value of D satisfies this condition. These results are formalized in Theorem 5.9.

We now prove the Theorem.

Proof of Theorem 5.2. Recall that for every guess $(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}})$ we calculate the sum

$$F(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}}) = \Re\left(\frac{1}{\psi(s_{\text{fft}})}\sum_j e^{\left(\frac{p}{q}x_j^T b - \frac{p}{q}y_{j,\text{enum}}^T \tilde{s}_{\text{enum}} - \left[\frac{p}{q}y_{j,\text{fft}}\right]^T \tilde{s}_{\text{fft}}\right)\frac{2\pi i}{p}}\right) \ ,$$

via FFT. When $(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}}) = (s_{\text{enum}}, s_{\text{fft}})$ we have

$$F(s_{\text{enum}}, s_{\text{fft}}) = \Re\left(\frac{1}{\psi(s_{\text{fft}})}\sum_j e^{\left(\frac{p}{q}y_{j,\text{lat}}^T s_{\text{lat}} + \frac{p}{q}x_j^T e + \left\{\frac{p}{q}y_{j,\text{fft}}\right\}^T s_{\text{fft}}\right)\frac{2\pi i}{p}}\right) \ .$$

By Assumption 5.8 (see below), the value $F(s_{\text{enum}}, s_{\text{fft}})$ is the sum of independent identically distributed variables

$$F(s_{\text{enum}}, s_{\text{fft}}) = \sum_j \Re\left(\frac{1}{\psi(s_{\text{fft}})} e^{\left(\frac{p}{q}y_{j,\text{lat}}^T s_{\text{lat}} + \frac{p}{q}x_j^T e + \left\{\frac{p}{q}y_{j,\text{fft}}\right\}^T s_{\text{fft}}\right)\frac{2\pi i}{p}}\right)$$
$$= \sum_j \Re\left(\varepsilon_{j,\text{eq}} \cdot \varepsilon_{j,\text{round}}\right) \ ,$$

where $\varepsilon_{j,\mathrm{eq}} = e^{\left(y_{j,\mathrm{lat}}^T s_{\mathrm{lat}} + x_j^T e\right)\frac{2\pi i}{q}}$ and $\varepsilon_{j,\mathrm{round}} = \frac{1}{\psi(s_{\mathrm{fft}})} e^{\left(\left\{\frac{p}{q} y_{j,\mathrm{fft}}\right\}^T s_{\mathrm{fft}}\right)\frac{2\pi i}{p}}$. When D is large, the distribution of $F\left(s_{\mathrm{enum}}, s_{\mathrm{fft}}\right)$ is approximately normal with mean and variance determined by the distributions of $\varepsilon_{j,\mathrm{eq}}$ and $\varepsilon_{j,\mathrm{round}}$. We briefly describe results regarding the mean and variance that will be proven later.

Lemma 5.3. *We have*

$$\mathbb{E}\left(\varepsilon_{j,\mathrm{eq}}\right) \geq e^{-2\left(\frac{\pi\tau}{q}\right)^2} = D_{\mathrm{eq}}^{-\frac{1}{2}} \ ,$$
$$\mathbb{E}\left(\varepsilon_{j,\mathrm{eq}}^2\right) \leq 2e^{-8\left(\frac{\pi\tau}{q}\right)^2} \ ,$$

where $\tau^2 = \frac{\alpha^{-2}\|e\|^2 + \|s_{\mathrm{lat}}\|^2}{m + k_{\mathrm{lat}}}\ell^2$ *and* ℓ *is the average length of the vectors given by the short vectors sampling algorithm.*

Lemma 5.4. *Writing* $s_{\mathrm{fft}} = (s_1, \ldots, s_{k_{\mathrm{fft}}})$, *we have*

$$\mathbb{E}\left(\varepsilon_{j,\mathrm{round}}\right) \geq D_{\mathrm{round}}^{-\frac{1}{2}} \ .$$

Lemma 5.5. *We have* $\mathbb{E}\left(F\left(s_{\mathrm{enum}}, s_{\mathrm{fft}}\right)\right) = D\,\mathbb{E}\left(\varepsilon_{j,\mathrm{eq}}\right)\mathbb{E}\left(\varepsilon_{j,\mathrm{round}}\right)$.

Lemma 5.6. *When* $\tilde{s}_{\mathrm{enum}} \neq s_{\mathrm{enum}}$, *we have* $\mathbb{E}\left(F\left(\tilde{s}_{\mathrm{enum}}, \tilde{s}_{\mathrm{fft}}\right)\right) = 0$.

Lemma 5.7. *For all* $(\tilde{s}_{\mathrm{enum}}, \tilde{s}_{\mathrm{fft}})$, *we have*

$$\mathrm{Var}\left(F\left(\tilde{s}_{\mathrm{enum}}, \tilde{s}_{\mathrm{fft}}\right)\right) \leq D\left(\frac{1}{2} + e^{-8\left(\frac{\pi\tau}{q}\right)^2}\right) = D \cdot D_{\mathrm{arg}} \ .$$

The score $F\left(\tilde{s}_{\mathrm{enum}}, \tilde{s}_{\mathrm{fft}}\right)$ is the sum of many independent and identically distributed variables, and is thus approximately distributed according to a normal distribution. According to Lemmas 5.3-5.7, the score of the correct guess, $F\left(s_{\mathrm{enum}}, s_{\mathrm{fft}}\right)$, is normally distributed with mean $\geq D_{\mathrm{eq}}^{-\frac{1}{2}} \cdot D_{\mathrm{round}}^{-\frac{1}{2}} \cdot D$ and variance at most $D_{\mathrm{arg}} \cdot D$. Therefore

$$\begin{aligned}
&\Pr\left[F\left(s_{\mathrm{enum}}, s_{\mathrm{fft}}\right) \leq C\right] \leq \\
&\qquad \leq \Pr\left[\mathcal{N}\left(D_{\mathrm{eq}}^{-\frac{1}{2}} \cdot D_{\mathrm{round}}^{-\frac{1}{2}} \cdot D, D_{\mathrm{arg}} \cdot D\right) \leq C\right] = \\
&\qquad = 1 - \Phi\left(\frac{D_{\mathrm{eq}}^{-\frac{1}{2}} \cdot D_{\mathrm{round}}^{-\frac{1}{2}} \cdot D - C}{\sqrt{D_{\mathrm{arg}} \cdot D}}\right) \leq \\
&\qquad \leq 1 - \Phi\left(\sqrt{D_{\mathrm{fpfn}}} - \frac{C}{\sqrt{D_{\mathrm{arg}} \cdot D}}\right) = \\
&\qquad = 1 - \Phi\left(\phi_{\mathrm{fp}} + \phi_{\mathrm{fn}} - \phi_{\mathrm{fp}}\right) = 1 - \Phi\left(\phi_{\mathrm{fn}}\right) = \frac{\mu}{2} \ ,
\end{aligned}$$

where the third transition follows from our choice of $D \geq D_{\mathrm{eq}} \cdot D_{\mathrm{round}} \cdot D_{\mathrm{arg}} \cdot D_{\mathrm{fpfn}}$, the fourth transition follows from the definition of $D_{\mathrm{fpfn}} = \left(\phi_{\mathrm{fp}} + \phi_{\mathrm{fn}}\right)^2$ and choice of $C = \phi_{\mathrm{fp}}\sqrt{D_{\mathrm{arg}} \cdot D}$, and the last transition follows from the definition of $\phi_{\mathrm{fn}} = \Phi^{-1}\left(1 - \frac{\mu}{2}\right)$.

Moreover, for any fixed $(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}})$ where $\tilde{s}_{\text{enum}} \neq s_{\text{enum}}$, the score $F(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}})$ is normally distributed with mean 0 and variance at most $D_{\text{arg}} \cdot D$ according to Lemmas 5.6 and 5.7. Therefore

$$\begin{aligned}\Pr\left[F(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}}) > C\right] \leq \Pr\left[\mathcal{N}(0, D_{\text{arg}} \cdot D) > C\right] &= \\ &= 1 - \Phi\left(\frac{C}{\sqrt{D_{\text{arg}} \cdot D}}\right) = \\ &= 1 - \Phi(\phi_{\text{fp}}) = \\ &= \frac{\mu}{2N_{\text{enum}}(s_{\text{enum}})\, p^{k_{\text{fft}}}} ,\end{aligned}$$

where the third transition follows from the choice of C, and the fourth transition follows from the definition of $\phi_{\text{fp}} = \Phi^{-1}\left(1 - \frac{\mu}{2 \cdot N_{\text{enum}}(s_{\text{enum}}) \cdot p^{k_{\text{fft}}}}\right)$.

The number of guesses $\tilde{s}_{\text{enum}}$ before reaching s_{enum} is $N_{\text{enum}}(s_{\text{enum}})$, so the number of pairs we check before reaching s_{enum} is $N_{\text{enum}}(s_{\text{enum}})\, p^{k_{\text{fft}}}$. By the union bound, we get

$$\Pr\left[\text{Algorithm 2 does not return } s_{\text{enum}}\right] \leq \frac{\mu}{2} + \frac{\mu N_{\text{enum}}(s_{\text{enum}})\, p^{k_{\text{fft}}}}{2N_{\text{enum}}(s_{\text{enum}})\, p^{k_{\text{fft}}}} = \mu .$$

□

We now prove Lemmas 5.3-5.7. In the proofs we use the following extension of Assumption 4.4 regarding the distribution of the output of the short vectors sampling algorithm.

Assumption 5.8. *The vectors $x_j, y_{j,\text{lat}}, y_{j,\text{enum}}, y_{j,\text{fft}}$ are approximately independent. The distributions of $y_{j,\text{enum}}, y_{j,\text{fft}}$ are approximately uniform mod q, while the distributions of $\alpha x_j, y_{j,\text{lat}}$ are as described in Assumption 4.4.*

Recall that $y_{j,\text{enum}}$ and $y_{j,\text{fft}}$ are determined as a linear function of x_j, namely $A_{\text{enum}}^T x_j$ and $A_{\text{fft}}^T x_j$ respectively. However, since the entries of $A_{\text{enum}}, A_{\text{fft}}$ are uniformly random mod q, the dependency is negligible for our purposes.

Proof of Lemma 5.3. We recall that $\varepsilon_{j,\text{eq}} = \mathbb{E}\left(e^{\left(y_{j,\text{lat}}^T s_{\text{lat}} + x_j^T e\right)\frac{2\pi i}{q}}\right)$. The exponent is the sum of two inner products. By Assumption 5.8, each inner product is the sum of multiple independent modular Gaussian random variables. Hence it is also a modular Gaussian, namely, $y_{j,\text{lat}}^T s_{\text{lat}} + x_j^T e \sim \rho_{q,\tau}$ for some $\tau > 0$ we will later compute. When X is distributed according to $\rho_{q,\tau}$, Theorem 2.4 along with Lemma 2.5 gives

$$\mathbb{E}\left(e^{\frac{2\pi i}{q}X}\right) = \sum_{j=0}^{q-1} e^{\frac{2\pi i j}{q}} \rho_{q,\tau}(j) = \widehat{\rho_{q,\tau}}(1) \geq e^{-2\left(\frac{\pi\tau}{q}\right)^2} .$$

Similarly, we have

$$\mathbb{E}\left(\left(e^{\frac{2\pi i}{q}X}\right)^2\right) = \widehat{\rho_{q,\tau}}(2) \leq 2e^{-8\left(\frac{\pi\tau}{q}\right)^2} .$$

Here we assume that $\tau \geq \sqrt{\frac{q \log 2}{8\pi^2}}$, which holds for our range of parameters (see Section 7.3 for the parameters used)[1].

Finally, we calculate the variance τ^2 of $y_{j,\text{lat}}^T s_{\text{lat}} + x_j^T e$. Recall that Algorithm 3 is applied to the column space of

$$B = \begin{pmatrix} \alpha I_m & 0 \\ A_{\text{lat}}^T & q I_{k_{\text{lat}}} \end{pmatrix} ,$$

where $\alpha = \frac{\sigma_e}{\sigma_s}$ is chosen in order to balance the error terms in our equations. Let $v = (\alpha x, y_{\text{lat}})$ be a vector in $\mathbb{R}^{m+k_{\text{lat}}}$ output by Algorithm 3, and let ℓ its expected length. By Assumption 4.4, each coordinate of v has variance $\frac{\ell^2}{m+k_{\text{lat}}}$ and mean 0. Therefore

$$\begin{aligned} \tau^2 = \operatorname{Var}\left(\sum_t x_t e_t + \sum_t (y_{\text{lat}})_t (s_{\text{lat}})_t\right) &= \\ = \mathbb{E}\left(\frac{1}{\alpha^2}\sum_t v_t^2 e_t^2 + \sum_t v_t^2 (s_{\text{lat}})_t^2\right) &= \\ = \frac{1}{\alpha^2}\sum_t e_t^2\, \mathbb{E}\left(v_t^2\right) + \sum_t (s_{\text{lat}})_t^2\, \mathbb{E}\left(v_t^2\right) &= \\ = \left(\alpha^{-2}\|e\|^2 + \|s_{\text{lat}}\|^2\right)\frac{\ell^2}{m+k_{\text{lat}}} &, \end{aligned}$$

as required. □

Proof of Lemma 5.4. Recall that $\varepsilon_{j,\text{round}} = \frac{1}{\psi(s_{\text{fft}})} e^{\left\{\frac{p}{q} y_{j,\text{fft}}\right\}^T s_{\text{fft}} \frac{2\pi i}{p}}$. Note that the exponent $e^{\left\{\frac{p}{q} y_{j,\text{fft}}\right\}^T s_{\text{fft}} \frac{2\pi i}{p}}$ is the product of independent identically distributed variables. Denoting a single coordinate of y_{fft} as y_t and a single coordinate of s_{fft} as s_t, we get $\mathbb{E}\left(\varepsilon_{j,\text{round}}\right) = \frac{1}{\psi(s_{\text{fft}})} \prod_t \mathbb{E}\left(e^{\left\{\frac{p}{q} y_t\right\} s_t \frac{2\pi i}{p}}\right)$.

Let $q' := \frac{q}{\gcd(p,q)}$. When q' is odd, $\left\{\frac{p}{q} y_t\right\}$ is distributed uniformly over the set $\left\{\frac{-q'+1}{2q'}, \frac{-q'+3}{2q'}, \ldots, \frac{q'-1}{2q'}\right\}$. When q' is even, it is distributed uniformly over the set $\left\{\frac{1}{q'} - \frac{1}{2}, \frac{2}{q'} - \frac{1}{2}, \ldots, \frac{1}{2}\right\}$. The second set is the same as the first set up to a constant added to all of the values. In order to treat both cases at the same time, we use the set $\left\{\frac{-q'+1}{2q'} + c_{q'}, \frac{-q'+3}{2q'} + c_{q'}, \ldots, \frac{q'-1}{2q'} + c_{q'}\right\}$, where $c_{q'} = 0$ when q' is odd and $\frac{1}{2q'}$ when q' is even.

When $s_t = 0$, the expectation $\mathbb{E}\left(e^{\left\{\frac{p}{q} y_t\right\} s_t \frac{2\pi i}{p}}\right)$ is 1. Otherwise, we write

$$\mathbb{E}\left(e^{\left\{\frac{p}{q} y_t\right\}^T s_t \frac{2\pi i}{p}}\right) = \frac{1}{q'} \sum_{k=0}^{q'-1} e^{\frac{2\pi i s_t}{p}\left(\frac{-q'+1+2k}{2q'} + c_{q'}\right)} = \frac{e^{\frac{2\pi i s_t c_{q'}}{p}}}{q'} \sum_{k=0}^{q'-1} e^{\frac{2\pi i s_t(-q'+1+2k)}{2pq'}} .$$

[1]We remark that for a range of parameters where the inequality doesn't hold, namely that $\tau < \sqrt{\frac{q \log 2}{8\pi^2}}$, then $\varepsilon_{j,\text{eq}}$ can be shown to be large, which improves the running time of the attack.

The sum $\sum_{k=0}^{q'-1} e^{\frac{2\pi i s_t(-q'+1+2k)}{2pq'}}$ is the sum of a geometric series with ratio $e^{\frac{2\pi i s_t}{pq'}}$. When $s_t \neq 0$, it equals

$$\frac{e^{\frac{2\pi i s_t(q'+1)}{2pq'}} - e^{\frac{2\pi s_t(-q'+1)}{2pq'}}}{e^{\frac{2\pi i s_t}{pq'}} - 1} = \frac{\sin\left(\frac{\pi s_t}{p}\right)}{\sin\left(\frac{\pi s_t}{pq'}\right)} ,$$

Thus the total expectation equals

$$\begin{aligned}\mathbb{E}\left(\varepsilon_{j,\text{round}}\right) &= \frac{1}{\psi\left(s_{\text{fft}}\right)} \prod_{t=1}^{k_{\text{fft}}} \mathbb{E}\left(e^{\left\{\frac{p}{q} y_t\right\} s_t \frac{2\pi i}{p}}\right) = \\ &= \frac{1}{\psi\left(s_{\text{fft}}\right)} \prod_{\substack{t=1 \\ s_t \neq 0}}^{k_{\text{fft}}} \frac{e^{\frac{2\pi i s_t c_{q'}}{p}} \sin\left(\frac{\pi s_t}{p}\right)}{q' \sin\left(\frac{\pi s_t}{pq'}\right)} = \\ &= \frac{e^{\frac{2\pi i c_{q'}}{p} \sum_t s_t}}{\psi\left(s_{\text{fft}}\right)} \prod_{\substack{t=1 \\ s_t \neq 0}}^{k_{\text{fft}}} \frac{\sin\left(\frac{\pi s_t}{p}\right)}{q' \sin\left(\frac{\pi s_t}{pq'}\right)} .\end{aligned}$$

Defining $\psi\left(s_{\text{fft}}\right) = e^{\frac{2\pi i c_{q'}}{p} \sum_t s_t}$, we get

$$\mathbb{E}\left(\varepsilon_{j,\text{round}}\right) = \prod_{\substack{t=1 \\ s_t \neq 0}}^{k_{\text{fft}}} \frac{\sin\left(\frac{\pi s_t}{p}\right)}{q' \sin\left(\frac{\pi s_t}{pq'}\right)} \geq \prod_{\substack{t=1 \\ s_t \neq 0}}^{k_{\text{fft}}} \frac{\sin\left(\frac{\pi s_t}{p}\right)}{\frac{\pi s_t}{p}} = D_{\text{round}}^{-\frac{1}{2}} ,$$

which completes the proof. □

Proof of Lemma 5.5. We have

$$\begin{aligned}&\mathbb{E}\left(F\left(s_{\text{enum}}, s_{\text{fft}}\right)\right) = \mathbb{E}\left(\Re\left(\varepsilon_{j,\text{eq}} \cdot \varepsilon_{j,\text{round}}\right)\right) = \Re\left(\mathbb{E}\left(\varepsilon_{j,\text{eq}} \cdot \varepsilon_{j,\text{round}}\right)\right) = \\ &\Re\left(\mathbb{E}\left(\varepsilon_{j,\text{eq}}\right) \mathbb{E}\left(\varepsilon_{j,\text{round}}\right)\right) = \mathbb{E}\left(\varepsilon_{j,\text{eq}}\right) \mathbb{E}\left(\varepsilon_{j,\text{round}}\right) ,\end{aligned}$$

since $\varepsilon_{j,\text{eq}}$ and $\varepsilon_{j,\text{round}}$ are independent and have real expectations by Lemmas 5.3 and 5.4. □

Proof of Lemma 5.6. We write $\tilde{s}_{\text{enum}} = s_{\text{enum}} + \Delta s_{\text{enum}}$, and $\tilde{s}_{\text{fft}} = s_{\text{fft}} + \Delta s_{\text{fft}}$. We have

$$\begin{aligned}&F\left(\tilde{s}_{\text{enum}}, \tilde{s}_{\text{fft}}\right) = \\ &= \Re\left(\frac{1}{\psi\left(s_{\text{fft}}\right)} \sum_j e^{\left(\frac{p}{q} y_{\text{lat}}^T s_{\text{lat}} + \frac{p}{q} x^T e + \left\{\frac{p}{q} y_{j,\text{fft}}\right\}^T s_{\text{fft}} - \frac{p}{q} y_{j,\text{enum}}^T \Delta s_{\text{enum}} - \left[\frac{p}{q} y_{j,\text{fft}}\right]^T \Delta s_{\text{fft}}\right) \frac{2\pi i}{p}}\right) = \\ &= \sum_j \Re\left(\frac{1}{\psi\left(s_{\text{fft}}\right)} e^{\left(-y_{j,\text{enum}}^T \Delta s_{\text{enum}}\right) \frac{2\pi i}{q}} e^{\left(\frac{p}{q} y_{\text{lat}}^T s_{\text{lat}} + \frac{p}{q} x^T e + \left\{\frac{p}{q} y_{j,\text{fft}}\right\}^T s_{\text{fft}} - \left[\frac{p}{q} y_{j,\text{fft}}\right]^T \Delta s_{\text{fft}}\right) \frac{2\pi i}{p}}\right) = \\ &\qquad = \sum_j \Re\left(e^{\left(-y_{j,\text{enum}}^T \Delta s_{\text{enum}}\right) \frac{2\pi i}{q}} e^{2\pi i W_j}\right) ,\end{aligned}$$

where W_j is independent of $y_{j,\mathrm{enum}}$ by Assumption 5.8. When Δs_{enum} is nonzero, $y_{j,\mathrm{enum}}^T \Delta s_{\mathrm{enum}}$ is uniform mod q, which implies

$$\mathbb{E}\left(F\left(\tilde{s}_{\mathrm{enum}}, \tilde{s}_{\mathrm{fft}}\right)\right) = \sum_j \Re\left(\mathbb{E}\left(e^{\left(-y_{j,\mathrm{enum}}^T \Delta s_{\mathrm{enum}}\right)\frac{2\pi i}{q}}\right)\mathbb{E}\left(e^{2\pi i W_j}\right)\right) = 0$$

as required. □

Proof of Lemma 5.7. For any $(\tilde{s}_{\mathrm{enum}}, \tilde{s}_{\mathrm{fft}})$, we have

$$\begin{aligned}
&F\left(\tilde{s}_{\mathrm{enum}}, \tilde{s}_{\mathrm{fft}}\right) = \\
&= \Re\left(\frac{1}{\psi\left(s_{\mathrm{fft}}\right)} \sum_j e^{\left(\frac{p}{q} y_{\mathrm{lat}}^T s_{\mathrm{lat}} + \frac{p}{q} x^T e + \left\{\frac{p}{q} y_{j,\mathrm{fft}}\right\}^T s_{\mathrm{fft}} - \frac{p}{q} y_{j,\mathrm{enum}}^T \Delta s_{\mathrm{enum}} - \left[\frac{p}{q} y_{j,\mathrm{fft}}\right]^T \Delta s_{\mathrm{fft}}\right)\frac{2\pi i}{p}}\right) = \\
&= \sum_j \Re\left(\frac{1}{\psi\left(s_{\mathrm{fft}}\right)} e^{\left(y_{\mathrm{lat}}^T s_{\mathrm{lat}} + x^T e\right)\frac{2\pi i}{q}} e^{\left(\left\{\frac{p}{q} y_{j,\mathrm{fft}}\right\}^T s_{\mathrm{fft}} - \frac{p}{q} y_{j,\mathrm{enum}}^T \Delta s_{\mathrm{enum}} - \left[\frac{p}{q} y_{j,\mathrm{fft}}\right]^T \Delta s_{\mathrm{fft}}\right)\frac{2\pi i}{p}}\right) = \\
&\qquad = \sum_j \Re\left(\varepsilon_{j,\mathrm{eq}} e^{2\pi i Z_j}\right) \ ,
\end{aligned}$$

where Z_j is independent of $\varepsilon_{j,\mathrm{eq}}$ by Assumption 5.8. Therefore,

$$\begin{aligned}
&\mathrm{Var}\left(F\left(\tilde{s}_{\mathrm{enum}}, \tilde{s}_{\mathrm{fft}}\right)\right) = D \cdot \mathrm{Var}\left(\Re\left(\varepsilon_{j,\mathrm{eq}} e^{2\pi i Z_j}\right)\right) \leq \\
&\quad \leq D \cdot \mathbb{E}\left(\Re\left(\varepsilon_{j,\mathrm{eq}} e^{2\pi i Z_j}\right)^2\right) = D \cdot \mathbb{E}\left(\frac{\varepsilon_{j,\mathrm{eq}}^2 e^{4\pi i Z_j} + 2 + \overline{\varepsilon_{j,\mathrm{eq}}^2 e^{4\pi i Z_j}}}{4}\right) = \\
&\quad = \frac{D}{4}\left(2 + 2\,\mathbb{E}\left(\Re\left(\varepsilon_{j,\mathrm{eq}}^2 e^{4\pi i Z_j}\right)\right)\right) = \frac{D}{2}\left(1 + \Re\left(\mathbb{E}\left(\varepsilon_{j,\mathrm{eq}}^2\right)\mathbb{E}\left(e^{4\pi i Z_j}\right)\right)\right) \leq \\
&\qquad \leq \frac{D}{2}\left(1 + \left|\mathbb{E}\left(\varepsilon_{j,\mathrm{eq}}^2\right)\mathbb{E}\left(e^{4\pi i Z_j}\right)\right|\right) \leq \frac{D}{2}\left(1 + 2e^{-8\left(\frac{\pi\tau}{q}\right)^2}\right) = D_{\mathrm{arg}} \cdot D \ ,
\end{aligned}$$

where the second-to-last transition follows from $\left|e^{4\pi i Z_j}\right| = 1$ and Lemma 5.3. □

5.2.2 Parameters for Advantage $\frac{1}{2}$

In this section we give concrete parameters (D, C) for which the attacker gains advantage close to $\frac{1}{2}$. Whereas Theorem 5.2 gives parameters that are dependent on the secret (s, e), the following theorem gives parameters for which the attack succeeds for at least half of the vectors (s, e).

Theorem 5.9. *Let $(n, m, q, \chi_s, \chi_e)$ be LWE parameters, $(\beta_1, \beta_2, k_{\mathrm{enum}}, k_{\mathrm{fft}}, k_{\mathrm{lat}}, p)$ be a partial tuple of parameters for Algorithm 2, let ℓ be the expected length of the vectors returned by Algorithm 3, and let $0 < \mu < 1$. Then Algorithm 2 with parameters*

$$D \geq \tilde{D}_{\mathrm{eq}} \cdot \tilde{D}_{\mathrm{round}} \cdot \tilde{D}_{\mathrm{arg}} \cdot \tilde{D}_{\mathrm{fpfn}}(\mu)$$
$$C = \tilde{\phi}_{\mathrm{fp}}(\mu)\sqrt{\tilde{D}_{\mathrm{arg}} \cdot D}$$

where

$$\begin{aligned}
\tilde{D}_{\mathrm{eq}} &= e^{4\left(\frac{\pi\sigma_s\ell}{q}\right)^2} \\
\tilde{D}_{\mathrm{round}} &= \prod_{\bar{s}\neq 0}\left(\frac{\sin\left(\frac{\pi\bar{s}}{p}\right)}{\frac{\pi\bar{s}}{p}}\right)^{-2k_{\mathrm{fft}}\chi_s(\bar{s})} \\
\tilde{D}_{\mathrm{arg}} &= \frac{1}{2}\cdot e^{2\left(\chi_e(0)+e^{-\frac{8\pi^2\alpha^{-2}\ell^2}{q^2(m+k_{\mathrm{lat}})}}\right)^m\left(\chi_s(0)+e^{-\frac{8\pi^2\ell^2}{q^2(m+k_{\mathrm{lat}})}}\right)^{k_{\mathrm{lat}}}} \approx \frac{1}{2} \\
\tilde{D}_{\mathrm{fpfn}}(\mu) &= \left(\tilde{\phi}_{\mathrm{fp}}(\mu)+\tilde{\phi}_{\mathrm{fn}}(\mu)\right)^2 \\
\tilde{\phi}_{\mathrm{fp}}(\mu) &= \Phi^{-1}\left(1-\frac{\mu}{2\cdot 2^{k_{\mathrm{enum}}H(\chi_s)}\cdot p^{k_{\mathrm{fft}}}}\right) \\
\tilde{\phi}_{\mathrm{fn}}(\mu) &= \Phi^{-1}\left(1-\frac{\mu}{2}\right)
\end{aligned}$$

outputs the correct s_{enum} *with probability at least* $\frac{1-\mu}{2}$.

See the beginning of Section 5.2 for informal explanations of these different factors, and Corollary 5.10 for simpler asymptotic expressions.

Proof. By Theorem 5.2, the attack succeeds with probability $1-\mu$ given that $D \geq D_{\mathrm{eq}}\cdot D_{\mathrm{round}}\cdot D_{\mathrm{arg}}\cdot D_{\mathrm{fpfn}}(\mu)$ and C is chosen accordingly.
Note that $\log\left(D_{\mathrm{eq}}\cdot D_{\mathrm{round}}\cdot D_{\mathrm{arg}}\cdot D_{\mathrm{fpfn}}(\mu)\right)$ is approximately normally distributed (over the randomness of (s,e)), and so it is greater than its expectation with probability $\approx \frac{1}{2}$. It remains to show that the logarithmic expectation of $D_{\mathrm{eq}}\cdot D_{\mathrm{round}}\cdot D_{\mathrm{arg}}\cdot D_{\mathrm{fpfn}}(\mu)$ is at most $\tilde{D}_{\mathrm{eq}}\cdot\tilde{D}_{\mathrm{round}}\cdot\tilde{D}_{\mathrm{arg}}\cdot\tilde{D}_{\mathrm{fpfn}}(\mu)$.

Recall that $\log(D_{\mathrm{eq}}) = \frac{4\pi^2}{q^2}\tau^2$ where $\tau^2 = \frac{\alpha^{-2}\|e\|^2+\|s_{\mathrm{lat}}\|^2}{m+k_{\mathrm{lat}}}\ell^2$ and $\alpha = \frac{\sigma_e}{\sigma_s}$. Since each coordinate of e, s_{lat} is independently distributed, $\log(D_{\mathrm{eq}})$ is approximately normal with expectation

$$\mathbb{E}\left(\log\left(D_{\mathrm{eq}}\right)\right) = \frac{4\pi^2}{q^2}\frac{\alpha^{-2}\sigma_e^2 m+\sigma_s^2 k_{\mathrm{lat}}}{m+k_{\mathrm{lat}}}\ell^2 = \frac{4\pi^2}{q^2}\sigma_s^2\ell^2 = \log\left(\tilde{D}_{\mathrm{eq}}\right) .$$

Recall further that $\log(D_{\mathrm{round}}) = -2\sum_t \log\left(\frac{\sin(\pi s_t/p)}{\pi s_t/p}\right)$, where the sum is taken over all nonzero coordinates s_t of s_{fft}. As above, since each coordinate of s_{fft} is independently distributed, $\log(D_{\mathrm{round}})$ is approximately normal with expectation

$$\mathbb{E}\left(\log\left(D_{\mathrm{round}}\right)\right) = -2k_{\mathrm{fft}}\sum_{\bar{s}\neq 0}\chi_s(\bar{s})\log\left(\frac{\sin(\pi\bar{s}/p)}{\pi\bar{s}/p}\right) = \log\left(\tilde{D}_{\mathrm{round}}\right) .$$

Next, we have $\log(D_{\mathrm{arg}}) = \log\left(\frac{1}{2}+e^{-\frac{8\pi^2\tau^2}{q^2}}\right) \leq \log\left(\frac{1}{2}\right)+2e^{-\frac{8\pi^2\tau^2}{q^2}}$ which is very close to $\log\left(\frac{1}{2}\right)$ for relevant parameter sets. For concreteness, we have

$$\begin{aligned}
\mathbb{E}\left(\log\left(D_{\text{arg}}\right)\right) &\leq \log\left(\frac{1}{2}\right) + \mathbb{E}\left(2e^{-\frac{8\pi^2\tau^2}{q^2}}\right) = \\
&= \log\left(\frac{1}{2}\right) + 2\left(\mathbb{E}\left(e^{-\frac{8\pi^2\alpha^{-2}e_j^2\ell^2}{q^2(m+k_{\text{lat}})}}\right)\right)^m \left(\mathbb{E}\left(e^{-\frac{8\pi^2 s_j^2\ell^2}{q^2(m+k_{\text{lat}})}}\right)\right)^{k_{\text{lat}}} \leq \\
&\leq \log\left(\frac{1}{2}\right) + 2\left(\chi_e(0) + e^{-\frac{8\pi^2\alpha^{-2}\ell^2}{q^2(m+k_{\text{lat}})}}\right)^m \left(\chi_s(0) + e^{-\frac{8\pi^2\ell^2}{q^2(m+k_{\text{lat}})}}\right)^{k_{\text{lat}}} = \log\left(\tilde{D}_{\text{arg}}\right) .
\end{aligned}$$

We emphasize that this expression is very close to $\log\left(\frac{1}{2}\right)$ for all reasonable parameter sets.

Lastly, we have

$$\begin{aligned}
D_{\text{fpfn}} &= \left(\Phi^{-1}\left(1 - \frac{\mu}{2}\right) + \Phi^{-1}\left(1 - \frac{\mu}{2 \cdot N_{\text{enum}}(s_{\text{enum}}) \cdot p^{k_{\text{fft}}}}\right)\right)^2 = \\
&= \left(a + \Phi^{-1}\left(1 - \frac{b}{N_{\text{enum}}(s_{\text{enum}})}\right)\right)^2 .
\end{aligned}$$

for certain constants $a, b > 0$ not dependent on (s, e). Since $x \mapsto \left(a + \Phi^{-1}\left(1 - \frac{b}{x}\right)\right)^2$ is a concave function in the relevant domain and since $\mathbb{E}\left(N_{\text{enum}}(s_{\text{enum}})\right) \leq 2^{k_{\text{enum}} H(\chi_s)}$, we have

$$\mathbb{E}\left(\log\left(D_{\text{fpfn}}\right)\right) \leq \log\left(\mathbb{E}\left(D_{\text{fpfn}}\right)\right) \leq \log\left(a + \Phi^{-1}\left(1 - \frac{b}{2^{k_{\text{enum}} H(\chi_s)}}\right)\right)^2 = \log\left(\tilde{D}_{\text{fpfn}}\right) ,$$

by applying Jensen's inequality twice. We remark again that since D_{fpfn} is only a polynomially large factor, its contribution to the overall complexity is relatively negligible.

In total, we get

$$\begin{aligned}
&\mathbb{E}\left(\log\left(D_{\text{eq}} \cdot D_{\text{round}} \cdot D_{\text{arg}} \cdot D_{\text{fpfn}}\right)\right) = \\
&\quad = \mathbb{E}\left(\log\left(D_{\text{eq}}\right)\right) + \mathbb{E}\left(\log\left(D_{\text{round}}\right)\right) + \mathbb{E}\left(\log\left(D_{\text{arg}}\right)\right) + \mathbb{E}\left(\log\left(D_{\text{fpfn}}\right)\right) \leq \\
&\quad\quad \leq \log\left(\tilde{D}_{\text{eq}}\right) + \log\left(\tilde{D}_{\text{eq}}\right) + \log\left(\tilde{D}_{\text{eq}}\right) + \log\left(\tilde{D}_{\text{eq}}\right) = \\
&\quad\quad\quad = \log\left(\tilde{D}_{\text{eq}} \cdot \tilde{D}_{\text{round}} \cdot \tilde{D}_{\text{arg}} \cdot \tilde{D}_{\text{fpfn}}\right) ,
\end{aligned}$$

as required. □

5.3 Asymptotic Complexity

In this section we give simpler asymptotic expressions for the time complexity of Algorithm 2. The goal of this section is to give a clearer view of the asymptotic contributions of the different parameters. We begin by calculating an asymptotic expression for D, the required number of short vectors to be sampled. We then combine this expression with Theorem 5.1 to state the asymptotic complexity of the entire algorithm.

5.3.1 Asymptotic Number of Samples

Theorem 5.9 gives an exact bound on the required number of samples. The following corollary gives an asymptotic bound as a function of the parameters for the algorithm.

Corollary 5.10. *Let $(n, m, q, \chi_s, \chi_e)$ be LWE parameters, $(\beta_1, \beta_2, k_{\text{enum}}, k_{\text{fft}}, k_{\text{lat}}, p)$ be a partial tuple of parameters for Algorithm 2, let ℓ be the expected length of the vectors returned by Algorithm 3, and let $0 < \mu < 1$. Then the number of samples D can be chosen such that*

$$D = O\left(e^{4\left(\frac{\ell\sigma_s\pi}{q}\right)^2} \cdot e^{\frac{k_{\text{fft}}}{3}\left(\frac{\sigma_s\pi}{p}\right)^2} \cdot (k_{\text{enum}} \cdot H(\chi_s) + k_{\text{fft}} \cdot \log(p) + \log(1/\mu))\right) ,$$

where

$$\ell\sigma_s = \sigma_e^{\frac{m}{m+k_{\text{lat}}}} \cdot (\sigma_s q)^{\frac{k_{\text{lat}}}{m+k_{\text{lat}}}} \cdot \sqrt{4/3} \cdot \sqrt{\frac{\beta_2}{2\pi e}} \cdot \delta(\beta_1)^{\frac{m+k_{\text{lat}}-\beta_2}{2}} \cdot (1 + o(1)) ,$$

and such that Algorithm 2 outputs the correct s_{enum} with probability at least $\frac{1-\mu}{2}$.

Proof. Recall that by Theorem 5.9, we need to sample $D \geq \tilde{D}_{\text{eq}} \cdot \tilde{D}_{\text{round}} \cdot \tilde{D}_{\text{arg}} \cdot \tilde{D}_{\text{fpfn}}(\mu)$ vectors in order to achieve advantage $\frac{1-\mu}{2}$. We give asymptotic expressions for each of these four factors, and combine them to get an asymptotic estimate for the required D.

Recall that the first term, $\tilde{D}_{\text{eq}} = \exp\left(4\left(\frac{\pi\sigma_s\ell}{q}\right)^2\right)$, comes from the Fourier coefficient of the error of the original dual attack equations, and is exponentially large. Here ℓ is the expected length of the vectors generated by Algorithm 3, which by Lemma 4.2 equals

$$\ell = \det(\Lambda)^{1/(m+k_{\text{lat}})} \cdot N_{\text{sieve}}(\beta_2)^{1/\beta_2} \cdot \sqrt{\frac{\beta_2}{2\pi e} \cdot (\pi\beta_2)^{1/\beta_2}} \cdot \delta(\beta_1)^{\frac{d-\beta_2}{2}} =$$
$$= \left(\frac{\sigma_e}{\sigma_s}\right)^{\frac{m}{m+k_{\text{lat}}}} \cdot q^{\frac{k_{\text{lat}}}{m+k_{\text{lat}}}} \cdot \sqrt{4/3} \cdot \sqrt{\frac{\beta_2}{2\pi e}} \cdot \delta(\beta_1)^{\frac{d-\beta_2}{2}} \cdot (1 + o(1)) .$$

The second term, $\tilde{D}_{\text{round}} = \prod_{\bar{s}\neq 0}\left(\frac{\sin(\pi s_t/p)}{\pi s_t/p}\right)^{-2k_{\text{fft}}\chi_s\bar{s}}$, comes from the rounding error from the modulus switch, and is also exponentially large. To approximate $\log\left(\tilde{D}_{\text{round}}\right) = -2k_{\text{fft}}\sum_{\bar{s}\neq 0}\chi_s(\bar{s})\log\left(\frac{\sin(\pi\bar{s}/p)}{\pi\bar{s}/p}\right)$, we approximate $\sin(x)$ as $x(1 - \frac{1}{6}x^2)$ using its Taylor series around $x = 0$, and $\log(x)$ as $x - 1$ using its Taylor series around $x = 1$. This approximation is accurate when $\bar{s}$ is close to 0, and is inaccurate when $\bar{s}$ is close to p. For values of $\bar{s}$ in $\left[-\frac{p}{2}, \frac{p}{2}\right]$, the approximation error is less than 10%. In the analyzed cryptographic schemes, the error introduced by the approximation is small close to the optimal parameter sets, and using the approximation induces very minor errors in the optimization results.

We get:

$$\begin{aligned}
\log\left(\tilde{D}_{\text{round}}\right) &= -2k_{\text{fft}} \sum_{\bar{s}\neq 0} \chi_s\left(\bar{s}\right) \log\left(\frac{\sin\left(\pi\bar{s}/p\right)}{\pi\bar{s}/p}\right) \approx \\
&\approx -2k_{\text{fft}} \sum_{\bar{s}\neq 0} \chi_s\left(\bar{s}\right) \log\left(1-\frac{\pi^2\bar{s}^2}{6p^2}\right) \approx \\
&\approx 2k_{\text{fft}} \sum_{\bar{s}\neq 0} \chi_s\left(\bar{s}\right) \frac{\pi^2\bar{s}^2}{6p^2} = \\
&= \frac{k_{\text{fft}}}{3}\frac{\pi^2}{p^2} \sum_{\bar{s}\neq 0} \chi_s \bar{s}^2 = \frac{k_{\text{fft}}}{3}\left(\frac{\sigma_s \pi}{p}\right)^2
\end{aligned}$$

and in total,

$$\tilde{D}_{\text{round}} = O\left(\exp\left(\frac{k_{\text{fft}}}{3}\left(\frac{\sigma_s\pi}{p}\right)^2\right)\right) .$$

The third term, $\tilde{D}_{\text{arg}} \approx \frac{1}{2}$, is a constant improvement factor obtained by considering the complex argument of the Fourier coefficient. We have $\tilde{D}_{\text{arg}} = O\left(1\right)$, and in fact it is very close to $\frac{1}{2}$ for all reasonable parameter sets.

The fourth and final term, $\tilde{D}_{\text{fpfn}} = \left(\tilde{\phi}_{\text{fp}}\left(\mu\right) + \tilde{\phi}_{\text{fn}}\left(\mu\right)\right)^2$ where $\tilde{\phi}_{\text{fp}}\left(\mu\right) = \Phi^{-1}\left(1 - \frac{\mu}{2\cdot 2^{k_{\text{enum}} H(\chi_s)}\cdot p^{k_{\text{fft}}}}\right)$ and $\tilde{\phi}_{\text{fn}}\left(\mu\right) = \Phi^{-1}\left(1-\frac{\mu}{2}\right)$, is a factor which ensures small false and false negative probabilities, and is only polynomially large. By the bound $\Phi^{-1}(1-x) = O\left(\sqrt{\log\left(\frac{1}{x}\right)}\right)$ for $x < 1/2$, we get

$$\begin{aligned}
\tilde{D}_{\text{fpfn}} &= \left(O\left(\sqrt{\log\left(2/\mu\right)}\right) + O\left(\sqrt{\log\left(2\cdot 2^{k_{\text{enum}}H(\chi_s)}\cdot p^{k_{\text{fft}}}/\mu\right)}\right)\right)^2 \\
&= O\left(\log\left(2^{k_{\text{enum}}H(\chi_s)}\cdot p^{k_{\text{fft}}}/\mu\right)\right) = O\left(k_{\text{enum}}H(\chi_s) + k_{\text{fft}}\log\left(p\right) + \log\left(1/\mu\right)\right) .
\end{aligned}$$

□

5.3.2 Asymptotic Time Complexity

The following corollary gives an asymptotic expression for the time complexity of Algorithm 2.

Corollary 5.11. *Let $(n, m, q, \chi_s, \chi_e)$ be LWE parameters, $(\beta_1, \beta_2, k_{\text{enum}}, k_{\text{fft}}, k_{\text{lat}}, p)$ be a partial tuple of parameters for Algorithm 2, and let $0 < \mu < 1$. Choosing the parameters C, D according to Theorem 5.2, Algorithm 2 outputs s_{enum} with probability at least $\frac{1-\mu}{2}$ in time*

$$O\left(\left\lceil \frac{D}{\left(\sqrt{4/3}\right)^{\beta_2 + o(\beta_2)}} \right\rceil \cdot \left(T_{\text{BKZ}}(d, \beta_1) + T_{\text{sieve}}(\beta_2)\right) + 2^{k_{\text{enum}}H(\chi_s)} \cdot \left(k_{\text{fft}} p^{k_{\text{fft}}} + D\right)\right) ,$$

where

$$D = O\left(e^{4\left(\frac{\ell\sigma_s\pi}{q}\right)^2} \cdot e^{\frac{k_{\text{fft}}}{3}\left(\frac{\sigma_s\pi}{p}\right)^2} \cdot \left(k_{\text{enum}} \cdot H(\chi_s) + k_{\text{fft}} \cdot \log(p) + \log(1/\mu)\right)\right) \ ,$$

$$\ell\sigma_s = \sigma_e^{\frac{m}{m+k_{\text{lat}}}} \cdot (\sigma_s q)^{\frac{k_{\text{lat}}}{m+k_{\text{lat}}}} \cdot \sqrt{4/3} \cdot \sqrt{\frac{\beta_2}{2\pi e}} \cdot \delta(\beta_1)^{\frac{m+k_{\text{lat}}-\beta_2}{2}} \cdot (1+o(1)) \ ,$$

and $T_{\text{BKZ}}, T_{\text{sieve}}$ are the running times of the BKZ algorithm and lattice sieving algorithm, respectively.

Proof. By Theorem 5.1, the running time of the algorithm is

$$T_{\text{sample}_{\beta_1,\beta_2,D}}(m+k_{\text{lat}}) + 2^{k_{\text{enum}}H(\chi_s)} \cdot \left(T_{\text{fft}}(k_{\text{fft}}, p) + T_{\text{table}}\right) \ ,$$

where $T_{\text{sample}_{\beta_1,\beta_2,D}}(m+k_{\text{lat}})$ is the cost of Algorithm 3, $T_{\text{fft}}(k_{\text{fft}}, p)$ is the cost of FFT, and T_{table} is the cost of the updating the table in each iteration. By Lemma 4.1,

$$T_{\text{sample}_{\beta_1,\beta_2,D}}(m+k_{\text{lat}}) = O\left(\left\lceil \frac{D}{N_{\text{sieve}}(\beta_2)} \right\rceil \cdot \left(T_{\text{BKZ}}(d,\beta_1) + T_{\text{sieve}}(\beta_2)\right)\right) =$$
$$= O\left(\left\lceil \frac{D}{\left(\sqrt{4/3}\right)^{\beta_2+o(\beta_2)}} \right\rceil \cdot \left(T_{\text{BKZ}}(d,\beta_1) + T_{\text{sieve}}(\beta_2)\right)\right) \ .$$

Moreover, it is well known that FFT can be implemented in $T_{\text{fft}}(k_{\text{fft}}, p) = O\left(k_{\text{fft}} p^{k_{\text{fft}}}\right)$ operations (see e.g. [Knu98]). Lastly, the table updating can be performed in $T_{\text{table}} = O(D)$ operations according to Lemma 5.12 below. Combining the above with the formula for D from Corollary 5.10, we get the desired result. □

5.4 Efficient Updating of the FFT Input

Recall that in Algorithm 2, we enumerate on the value of s_{enum}. Under that enumeration, we calculate a table T, and for every short vector pair (x, y) we add the phase $e^{\left(x_j^T b - y_{j,\text{enum}}^T \tilde{s}_{\text{enum}}\right)\frac{2\pi i}{q}}$ to cell $\left[\frac{p}{q} y_{j,\text{fft}}\right]$ of T. Then, we use the FFT algorithm to calculate the Fourier transform of T. Naively, updating the tables requires computation of the inner product $y_{j,\text{enum}}^T \tilde{s}_{\text{enum}}$ for each pair, which costs $O(k_{\text{fft}} D)$. We can improve this algorithmically.

Lemma 5.12. *Preparing the table T when iterating over the possible values of s_{enum} in descending order of probabilities can be done in amortized $4D$ addition operations.*

Proof. We describe an efficient algorithm for updating the table T while iterating over all possible values of s_{enum} with log-likelihood above some constant C. We first note that instead of calculating each inner product $y_{j,\text{enum}}^T \tilde{s}_{\text{enum}}$ from scratch, we can store the inner products and use them to calculate subsequent inner products faster. If we have already calculated the inner product $y_{j,\text{enum}}^T \tilde{s}_{\text{enum}}$, we can calculate the inner product $y_{j,\text{enum}}^T \tilde{s}'_{\text{enum}}$ as $y_{j,\text{enum}}^T \tilde{s}_{\text{enum}} + y_{j,\text{enum}}^T (\tilde{s}'_{\text{enum}} - \tilde{s}_{\text{enum}})$. The update operations requires only as many multiplications as there are non-zero coordinates in $(\tilde{s}'_{\text{enum}} - \tilde{s}_{\text{enum}})$.

Thus, to calculate all inner products $y_{j,\text{enum}}^T \tilde{s}_{\text{enum}}$ for all values of s_{enum}, we just have to find an efficient iteration order where only one coordinate changes in every iteration step. To efficiently iterate over the possible values of s_{enum} with log-likelihood above some cutoff, we use the following algorithm:

1. We use the most likely $\tilde{s}_{\text{enum}}$ as the starting point for the iteration.
2. We then enumerate over all values for the first coordinate which do not lower the log-likelihood below the cutoff, and for each we recursively iterate over the remaining coordinates using the same algorithm.

We note that for some $\tilde{s}_{\text{enum}}$ to be reached, the algorithm has to perform a series of recursive calls with intermediate possible values of s_{enum}. In the i'th intermediate value, the first i coordinates are equal to those of $\tilde{s}_{\text{enum}}$, and the last $n-i$ coordinates have the most likely value of χ_s. Since the coordinates are sampled independently, the log-likelihood of every intermediate is larger than that of $\tilde{s}_{\text{enum}}$. Thus, if the log-likelihood of $\tilde{s}_{\text{enum}}$ is above the cutoff, the likelihoods of all necessary intermediates are also above the cutoff, and it will be reached by the algorithm. It follows that all possible values of $\tilde{s}_{\text{enum}}$ are found by the algorithm.

This iteration order does not guarantee, however, that the iteration proceeds in order of decreasing probability. To iterate over the values of the secret in descending order of probabilities, we perform a series of iterations, with decreasing cutoffs for the minimal log-likelihood of a value of $\tilde{s}_{\text{enum}}$ found by the iteration. We pick the cutoffs so that in the i'th iteration, we iterate over 2^i values for $\tilde{s}_{\text{enum}}$. This ensures that we reach the j'th most likely value for s_{enum} in step $\lceil \log_2(j) \rceil$, and do $1+2+4+\cdots+2^{\lceil \log_2 j \rceil} < 4j$ iterations. Thus, during our iteration, when we reached the j'th value for the first time, we will have updated the table no more than $4j$ times. $\square$

6 Lattice Sieving Cost Model

In this section we review our model for the cost of lattice sieving in the RAM model. Our model largely follows [AGPS20], apart from a number of changes. We start by describing the model used in [AGPS20], then describe the differences in our model.

The costs of our algorithm are calculated according to our model, as well as the model in [AGPS20], in Section 7.3.

6.1 General Overview of the Model

As stated above, our cost model is based on the model presented in [AGPS20]. For the purposes of this work, we limit ourselves to the classical model and leave the quantum model for future work. In this section we review basic properties of the model, as present in [AGPS20].

Recall that lattice sieving is an algorithm for finding a list of short vectors in a lattice. The algorithm starts by generating a list of long lattice vectors. It then repeatedly applies the “AllPairSearch” routine, finding short combinations of vectors in the list and adding them to the list. Heuristically, after some number of repetitions of this routine, the list contains the $N_{\text{sieve}}(d)$ shortest vectors in the lattice. The cost of a sieve in this model is the number of basic logic gates required in total for implementing the “AllPairSearch” routines.

Progressive Lattice Sieving. Following [AGPS20, ABD$^+$21], our model uses progressive lattice sieving [LM18], which is a variant of lattice sieving in which sieves on sublattices of increasing dimensions are performed. This model assumes that running a sieve in dimension d requires performing one “AllPairSearch” in dimension d' for every $d' \leq d$.

Locality Sensitive Filtering. The implementation of the “AllPairSearch” routine in this model uses Locality Sensitive Filtering (LSF) [BDGL16]. In order to find close pairs of vectors in a list, the algorithm constructs buckets according to certain filters, such that vectors in the same bucket are more likely to be close to each other. The algorithm then iterates over all such pairs to find all close pairs in the list.

Locality Sensitive Filtering uses Random Product Codes for the construction of the buckets. The core part of the algorithm is the decoding routine, which efficiently finds the set of buckets in which a given vector lies.

Popcount Filter. The “popcount-filter” is a practical improvement of Locality Sensitive Filtering introduced in [AGPS20]. The core idea is that instead of directly checking whether a pair of vectors is close by calculating their inner product, the algorithm first performs a cheaper test which is more likely to succeed for close pairs of vectors. If the test is passed, the heavier inner product calculation is performed.

6.2 Modifications of the Model

In this section we list the main differences between the model from [AGPS20] and our model.

- The main difference is the cost of the random product code decoding algorithm. Our algorithm requires one addition, one xor, and three comparisons per legal codeword, which translate to 433 gates for a lattice of rank 400, as opposed to [AGPS20] which requires a super-constant number of *inner products* per legal codeword, which translate to 3,540,524 gates for a lattice of rank 400. This algorithm is described in Section 6.2.1.
- Additionally, we optimize the popcount filters over a larger set of parameters than the one considered in [AGPS20]. Popcount filters allow skipping the calculation of the inner product of two vectors by checking if the sign bits of their coordinates tend to be the same (or, more generally, the sign bits of their projections to one dimensional vector spaces). When two vectors are close to each other, their sign bits are the same with probability $\approx \frac{2}{3}$, while two random vectors are expected to share sign bits with probability $\frac{1}{2}$. When checking if two vectors are close, before

calculating the inner product, we calculate how many bits differ among their sign bits. If the number of differing bits is above some threshold, we conclude that the two vectors are likely not close without calculating their inner product.

The popcount filters have two parameters – the number of sign bits and the threshold. In [AGPS20], the threshold is fixed to be 1/3 of the number of sign bits. With these parameters, roughly half of the close vector pairs are rejected by the filter. We optimized both of these parameters. For a lattice of rank 400, we use the sign bits of 576 projections, and require that no more than 208 sign bits are different. This allows $\approx 91\%$ of the close vector pairs to pass the filter.

- Similarly to [ADH+19], we treat every lattice vector v as $\pm v$. This reduces the number of list vectors used in the sieve by a factor of 2. Additionally, since the number of queries to the LSF data structure becomes twice the number of vectors used in its construction, the LSF parameters involved in the construction and querying are optimized independently.
- Finally, our model considers two additional factors which make it more realistic and pessimistic for the attacker.
  - Originally, the number of filters in the LSF construction is determined by the probability of two close vectors to lie in the same bucket. Namely, if this probability is p, then one constructs $\frac{1}{p}$ filters such that each pair has probability $1-\frac{1}{e}$ to be found. However, this means that a constant proportion of the close pairs will not be found. Since the size of the list is the same as the number of close pairs in it, the list size will shrink by a constant factor after one iteration, and the next iterations will not continue properly.
    In order to solve this problem, we increase the size of the list by a constant factor such that only a constant proportion of the close pairs is required for the iterations to continue. We then multiply the number of filters by an appropriate constant such that the required proportion of the close pairs is found. We optimize these constant factors to minimize the cost of the algorithm.
  - The cost of "AllPairSearch" in [AGPS20] is mainly composed of two parts: the total cost of "popcount-filter" calculations, and the total cost of inner product calculations. The cost of the inner product calculations is ignored, but is ensured to be smaller than the cost of the popcount calculations. In our model, we add the two costs together.

Despite these pessimistic factors, our model predicts significantly lower costs for lattice sieving, mainly due to the improved random product code decoding algorithm.

6.2.1 Random Product Code Decoding Algorithm

In this section, we describe our Random Product Code decoding algorithm, which is a slightly optimized version of the algorithm presented in [BDGL16]. Since the algorithm is outside the scope of this paper, we describe it in general terms and do not give a formal analysis of its running time.

We start by describing the Random Product Code decoding problem. Let $v \in \mathbb{R}^{l \cdot k}$ be a given vector, and let $V_1, \dots, V_k \subseteq \mathbb{R}^{l \cdot k}$ be l-dimensional vector spaces such that $V_i \perp V_j$ for all i, j. Let $L_i \subseteq V_i$ for $i = 1, \dots, k$ be lists of vectors of length $\frac{1}{\sqrt{k}}$ called *filters*. A *codeword* is a k-tuple of filters, $(f_1, \dots, f_k) \in L_1 \times \cdots \times L_k$, which represents the vector $\sum_i f_i$. A *legal codeword* is a codeword for which $\sum_i \langle f_i, v \rangle \geq C$ for a given constant $C \in \mathbb{R}$. We refer to this sum as the inner product of a codeword and v. The problem is to efficiently find all legal codewords.

Our general strategy is, given a vector, to define an iteration tree over all legal codewords. The algorithm is similar to the algorithm for updating the FFT table presented in Section 5.4. The nodes of the tree represent legal codewords. For each codeword, we define several codewords as possible child nodes, and the child nodes are the subset of those that are legal. When iterating, we store only the current codeword and its inner product with the query vector.

The iteration order must satisfy three properties:

1. For every legal codeword except for the root codeword, there exists a legal parent node with a larger inner product with the query vector.
2. Given the inner product of a codeword and the query vector, the inner products between its child codewords and the query vector can be calculated using a constant number of operations.
3. For every node, only a constant number of possible child codewords are checked and found to be illegal codewords.

Property 1 ensures that all legal codewords are visited by the iteration. Properties 2 and 3 ensure that the iteration requires a constant number of operations per legal codeword.

We preprocess the lists L_i to allow quick iteration. Each list L_i is sorted by descending order of inner product with the query. Then, the lists are relabeled so that the inner product of the difference between first two vectors and the query vector is descending: $\langle L_i[1] - L_i[0], v \rangle \geq \langle L_{i+1}[1] - L_{i+1}[0], v \rangle$.

We now define the iteration order. Each codeword is represented as a k-tuple of indices in the L_i: the codeword w is represented by a tuple of indices $(w_1, w_2, \dots, w_k)$, and represents the vector $\sum_i L_i [w_i]$. The root node is the codeword $(0, 0, \dots, 0)$. For every codeword, we denote by n_w its largest nonzero index, or -1 for the root. The possible child nodes of the node w are obtained by incrementing $w[j]$ for any $j \geq n_w$.

For every child node, its parent node can be obtained by decrementing the last nonzero index. Since each list L_i is sorted by decreasing inner product with v, decrementing any index in the codeword increases the inner product with v, so property 1 is satisfied.

A parent node $w^{(p)}$ and a child node $w^{(c)}$ differ in a single filter, where $w_i^{(c)} = w_i^{(p)} + 1$. Thus, the difference between the vectors they represent is $L_i [j+1] - L_i [j]$ for some j, and the difference between their inner products with v is $\langle L_i [j+1] - L_i [j], v \rangle$. This term can be precomputed so that the calculation takes a constant number of operations, so property 2 is satisfied.

The second term can be precomputed so that the calculation takes a constant number of operations, so property 2 is satisfied. All child nodes of the codeword

w are obtained by incrementing w_j for some $j \geq n_w$. Except for w_{n_w}, the values of w_j are all zero, by the definition of n_w. Thus, the inner products of v with the child nodes of w are all equal to the inner product of v with w, plus a term of the form $\langle L_j[1] - L_j[0] \rangle$. In the preprocessing stage, the lists L_i were relabeled so that $\langle L_j[1] - L_j[0] \rangle \geq \langle L_{j+1}[1] - L_{j+1}[0] \rangle$. Therefore, if for some $j > n_w$ the child node obtained by incrementing j is illegal, it implies that the child nodes obtained by incrementing any $j' > j$ are also illegal. This means that the iteration can stop considering child nodes after the first illegal child node in which $j > n_w$. Thus, up to two illegal child nodes are considered, and property 3 is satisfied.

A careful analysis shows that the number of operation per legal codeword is a single addition, three comparisons, and a single xor.

As a further optimization, we note that the inner product calculations do not require floating-point operations, and can be performed using 16-bit integers instead. The sum of k b-bit numbers has an error of no more than $k2^{-b}$. When the error is smaller than $O(\frac{1}{n})$, the effect on the accuracy of the LSF is insignificant. Thus, for relevant lattice ranks and values of k, 16 bits are more than sufficient.

7 Analysis in the RAM Model

In this section we perform a detailed analysis of our attack in the RAM model. We start off by stating the different assumptions we make when analyzing lattice algorithms. Afterwards, we describe our refined analysis of the cost of sieving. We then determine the cost of our attack under these assumptions. Finally, we present the concrete costs of our attack for various LWE-based candidates.

7.1 Description of the Model and Assumptions

We now state the exact costs and assumptions on the behaviour of the various lattice algorithms in the RAM model. Our assumptions are standard and follow e.g. [AGPS20, ABD+21, DKL+21]. We list them here for completion and clarity.

Recall the Geometric Series Assumption (Assumption 2.1), and the Gaussian Heuristic (Assumption 2.2) from Section 2.2. Both are used in the analysis of the Short Vectors Sampling Algorithm (Algorithm 3) in Lemma 4.2 which we use in this section. Also recall Assumption 4.4 and its extension, Assumption 5.8, regarding the distribution of the outputs of Algorithm 3. We use these assumptions in the analysis of the attack in Section 5.

Our additional assumptions are as follows.

Assumption 7.1 (Sieve Quality [AGPS20])**.** *Let Λ be some lattice of dimension β. The number of lattice points produced by a sieve is*

$$N_{\text{sieve}}(\beta) = \frac{1}{\mathsf{Caps}(\beta, \pi/3)} \;,$$

where $\mathsf{Caps}(d, \theta)$ is the probability that a vector sampled uniformly from the unit d-sphere has angle at most θ with some fixed unit vector.

Assumption 7.2 ("Dimensions-for-Free" [Duc18, ADH$^+$19])**.** *Solving SVP for lattices in dimension β can be done using a sieve of dimension β^{eff}.*
In [Duc18], β^{eff} is estimated as

$$\beta^{\text{eff}} = \beta - \frac{\beta \log(4/3)}{\log(\beta/(2\pi e))} \ .$$

We refer to this model as the Asymptotic Model.
In [ADH$^+$19], β^{eff} is estimated as

$$\beta^{\text{eff}} = \beta - (11.46 + 0.0757 \cdot \beta) \ .$$

for a certain range of dimensions. We refer to this model as the G6K Model.

In Section 7.3, we analyze our attack for different candidates both under the asymptotic and the G6K models. Skipping ahead, the $\log_2$ of the running time is smaller by no more than 2 (and the difference is is usually smaller, around 1) when assuming the G6K model over the asymptotic one. Compared to the effect of the proposed attack itself, and the suggested sieving costs, the effect of the choice of the dimensions-for-free model is not dramatic.

We note that the G6K Model is an extrapolation of experimental results in [ADH$^+$19], and the authors recommend against using this model for higher dimensions than presented in the paper. Nevertheless, we choose to present our results in the G6K Model as well, mainly for purposes of comparison with previous results in [GJ21]. The difference between the two models has a small effect on the estimated complexity of the attack.

Assumption 7.3 (Sieve and BKZ Cost [LM18, ABD$^+$21])**.** *The cost of lattice sieving in a β-dimensional lattice using progressive sieving is*

$$T_{\text{sieve}}(\beta) = C_{\text{prog}} \cdot T_{\mathsf{NNS}}(\beta) \ ,$$

and the cost of progressive $\text{BKZ}_{d,\beta}$ *is*

$$T_{\text{BKZ}}(d, \beta) = C_{\text{prog}}^2 \cdot (d - \beta + 1) \cdot T_{\mathsf{NNS}}(\beta^{\text{eff}}) \ .$$

*Here $T_{\mathsf{NNS}}(\beta)$ is the cost of an "*AllPairSearch*" routine in dimension β. Moreover, $C_{\text{prog}} = 1/\left(1 - 2^{-0.292}\right)$ is the sum of a geometric series which accounts for the total number of calls to "*AllPairSearch*", and β^{eff} is the effective sieving dimension due to the "dimensions-for-free" (see Assumption 7.2).*

We consider two cost models for T_{NNS} – the one by [AGPS20] and our refined model described in Section 6. Our results are optimized for each model and discussed in Section 7.3.

Finally, let us make some assumptions about the cost of the FFT operation.

Assumption 7.4 (FFT Cost [Knu98])**.** *Given a table T of dimensions p^n for some integers n, p, the cost of performing FFT on T is*

$$T_{\text{fft}}(n, p) = C_{\text{mul}} \cdot n \cdot p^{n+1} \ ,$$

where C_{mul} is the cost of a single multiplication. We assume that $C_{\text{mul}} = |\text{word}|^2$, using naive multiplication.

We work with wordsize of 4 bytes (32 bits). This is sufficient precision for the floating-point arithmetics involved for the candidates considered.

7.2 Complexity Analysis

We now turn to analyze the complexity of Algorithm 3 and Algorithm 2 in the RAM model. Our analysis relies on the assumptions stated in Section 7.1, and follows the same notations.

Lemma 7.5 (Cost of Short Vectors Sampling Procedure)**.** *Let Λ be a d-dimensional lattice, let β_1, β_2 be the basis reduction block size, and vectors sampling block size respectively, and let D be the required number of vectors. Recall the notations from Assumption 7.3. The cost of Algorithm 3 is*

$$T_{\text{sample}_{\beta_1,\beta_2,D}}(d) = \left\lceil \frac{D}{N_{\text{sieve}}(\beta_2)} \right\rceil \cdot \left(C_{\text{prog}}^2 \cdot (d - \beta_1 + 1) \cdot T_{\mathsf{NNS}}(\beta_1^{\text{eff}}) + C_{\text{prog}} \cdot T_{\mathsf{NNS}}(\beta_2) \right) .$$

Proof. Algorithm 3 consists of $\left\lceil \frac{D}{N_{\text{sieve}}(\beta_2)} \right\rceil$ iterations, and each iteration has two steps. The first is a progressive BKZ on a lattice of dimension d using block size β_1. The second is a progressive sieve of dimension β_2. By Assumption 7.3, the cost of the former is

$$C_{\text{prog}}^2 \cdot (d - \beta_1 + 1) \cdot T_{\mathsf{NNS}}(\beta_1^{\text{eff}}) .$$

Similarly, the cost of the latter is

$$C_{\text{prog}} \cdot T_{\mathsf{NNS}}(\beta_2) .$$

□

We turn to analyze our attack. The analysis, as formally stated below, is straightforward – it follows from Theorem 5.1 and the supporting lemmas and assumptions referenced in the statement. Recall the parameters of the algorithm, as described in Section 3.2.

Theorem 7.6. *Let $(n, m, q, \chi_s, \chi_e)$ be LWE parameters, $(\beta_1, \beta_2, k_{\text{enum}}, k_{\text{fft}}, k_{\text{lat}}, p)$ be a partial tuple of parameters for Algorithm 2, and $\frac{1-\mu}{2}$ the desired success probability. Set C, D to be integers determined by Theorem 5.9. Then the cost of Algorithm 2 in the* RAM *model is*

$$T_{\text{sample}_{\beta_1,\beta_2,D}}(k_{\text{lat}} + m) + 2^{k_{\text{enum}} H(\chi_s)} \left(T_{\text{fft}}(k_{\text{fft}}, p) + T_{\text{table}}(D) \right) ,$$

where

- *The sampling time is:*

$$T_{\text{sample}_{\beta_1,\beta_2,D}}(k_{\text{lat}} + m) = \\ = \left\lceil \frac{D}{N_{\text{sieve}}(\beta_2)} \right\rceil \cdot \left(C_{\text{prog}}^2 \cdot (k_{\text{lat}} + m - \beta_1) \cdot T_{\mathsf{NNS}}(\beta_1^{\text{eff}}) + C_{\text{prog}} \cdot T_{\mathsf{NNS}}(\beta_2) \right)$$

  by Lemma 7.5.

- *The FFT time is:* $T_{\text{fft}}(k_{\text{fft}}, p) = C_{\text{mul}} \cdot k_{\text{fft}} \cdot p^{k_{\text{fft}}+1}$ *by Assumption 7.4.*
- *The required number of samples is:*

$$D = \tilde{D}_{\text{eq}} \cdot \tilde{D}_{\text{round}} \cdot \tilde{D}_{\text{arg}} \cdot \tilde{D}_{\text{fpfn}}(\mu) \ ,$$

where

$$\tilde{D}_{\text{eq}} = e^{4\left(\frac{\pi \sigma_s \ell}{q}\right)^2}$$

$$\tilde{D}_{\text{round}} = \prod_{\bar{s} \neq 0} \left(\frac{\sin\left(\frac{\pi \bar{s}}{p}\right)}{\frac{\pi \bar{s}}{p}} \right)^{-2k_{\text{fft}}\chi_s(\bar{s})}$$

$$\tilde{D}_{\text{arg}} = \frac{1}{2} \cdot e^{2\left(\chi_e(0)+e^{-\frac{8\pi^2\alpha^{-2}\ell^2}{q^2(m+k_{\text{lat}})}}\right)^m \left(\chi_s(0)+e^{-\frac{8\pi^2\ell^2}{q^2(m+k_{\text{lat}})}}\right)^{k_{\text{lat}}}} \approx \frac{1}{2}$$

$$\tilde{D}_{\text{fpfn}}(\mu) = \left(\tilde{\phi}_{\text{fp}}(\mu) + \tilde{\phi}_{\text{fn}}(\mu)\right)^2$$

$$\tilde{\phi}_{\text{fp}}(\mu) = \Phi^{-1}\left(1 - \frac{\mu}{2 \cdot 2^{k_{\text{enum}}H(\chi_s)} \cdot p^{k_{\text{fft}}}}\right)$$

$$\tilde{\phi}_{\text{fn}}(\mu) = \Phi^{-1}\left(1 - \frac{\mu}{2}\right)$$

by Theorem 5.9,

$$\ell\sigma_s = \sigma_e^{\frac{m}{m+k_{\text{lat}}}} \cdot (\sigma_s q)^{\frac{k_{\text{lat}}}{m+k_{\text{lat}}}} \cdot \sqrt{4/3} \cdot \sqrt{\frac{\beta_2}{2\pi e}} \cdot \delta(\beta_1)^{\frac{m+k_{\text{lat}}-\beta_2}{2}} \cdot (1+o(1))$$

by Lemma 4.2.

- *The table generation time is* $T_{\text{table}}(D) = 4 \cdot C_{\text{add}} \cdot D$ *by Lemma 5.12.*

7.3 Application to NIST Candidates

We now present the complexity of our attack on various NIST candidates. We optimized our attack for CRYSTALS-Kyber [BDK+18], CRYSTALS-Dilithium [DKL+18], and Saber [DKRV18] and the results are presented below. We compare the estimated security parameters by the candidates' authors, the security level as defined in the Call for Proposals [Nat16] and our proposed attack.

Our research primarily focused on CRYSTALS-Kyber. As our results also apply to CRYSTALS-Dilithium and Saber, we analyzed the proposed attack for these candidates as well. We note that the largest gap between the previously estimated security level, and the one arising from this work is obtained for CRYSTALS-Kyber, which was our main focus.

The results are given in Tables 3-6. The tables differ by their exact cost models. As detailed in Section 7.1 above, we consider both the asymptotic model [Duc18] and the G6K [ADH+19] model for the effect of "dimensions-for-free", see Assumption 7.2 and the discussion following it. We also consider two cost models for the cost of sieving – the one suggested by [AGPS20], and our estimations, see Assumption 7.3. The tables present the costs in every combination of these models.

Table 3: Evaluation of the security level using [AGPS20] sieve costs and the asymptotic model [Duc18].

Candidate	Required Security Level [Nat16]	Estimated Security Level [DKL+21] [ABD+21] [BMD+20]	This Work	Parameters					
				m	p	β_1	β_2	k_{enum}	k_{fft}
Kyber512	143	151.5	**143.8**	474	5	377	380	19	34
Kyber768	207	215.1	**200.5**	659	4	576	570	30	59
Kyber1024	272	287.3	**266.0**	836	4	809	790	40	82
Dilithium2	146	159	**153.4**	1002	6	409	406	21	31
Dilithium3	207	217	**210.5**	1280	12	609	601	21	34
Dilithium5	272	285	**273.3**	1679	6	832	814	36	64
LightSaber	143	Unspecified	**144.8**	512	7	380	383	16	29
Saber	207	Unspecified	**210.4**	712	6	612	603	25	48
FireSaber	272	Unspecified	**273.4**	885	5	835	816	36	72

Table 4: Evaluation of the security level using [AGPS20] sieve costs and the G6K model [ADH+19].

Candidate	Required Security Level [Nat16]	Estimated Security Level [DKL+21] [ABD+21] [BMD+20]	This Work	Parameters					
				m	p	β_1	β_2	k_{enum}	k_{fft}
Kyber512	143	151.5	**143.1**	474	6	380	378	19	30
Kyber768	207	215.1	**199.5**	655	4	580	566	31	58
Kyber1024	272	287.3	**264.4**	839	4	816	786	39	82
Dilithium2	146	159	**152.2**	1002	6	409	406	21	31
Dilithium3	207	217	**208.9**	1280	12	613	594	20	34
Dilithium5	272	285	**270.9**	1707	5	837	805	36	71
LightSaber	143	Unspecified	**144.1**	512	7	383	381	17	27
Saber	207	Unspecified	**209.3**	708	6	617	599	24	49
FireSaber	272	Unspecified	**271.7**	897	5	843	811	35	72

Table 5: Evaluation of the security level using the sieve costs from Section 6 and the asymptotic model [Duc18].

Candidate	Required Security Level [Nat16]	Estimated Security Level [DKL+21] [ABD+21] [BMD+20]	This Work	Parameters					
				m	p	β_1	β_2	k_{enum}	k_{fft}
Kyber512	143	151.5	**138.2**	485	5	379	383	17	33
Kyber768	207	215.1	**194.5**	652	5	580	574	27	51
Kyber1024	272	287.3	**259.3**	834	4	813	794	36	82
Dilithium2	146	159	**147.3**	989	6	409	411	17	33
Dilithium3	207	217	**203.7**	1279	11	611	603	18	35
Dilithium5	272	285	**266.2**	1663	6	834	816	32	65
LightSaber	143	Unspecified	**139.1**	512	7	382	386	15	27
Saber	207	Unspecified	**203.9**	714	6	614	605	23	48
FireSaber	272	Unspecified	**266.5**	878	5	840	818	32	73

Table 6: Evaluation of the security level using the sieve costs from Section 6 and the G6K model [ADH+19].

Candidate	Required Security Level [Nat16]	Estimated Security Level [DKL+21] [ABD+21] [BMD+20]	This Work	Parameters					
				m	p	β_1	β_2	k_{enum}	k_{fft}
Kyber512	143	151.5	**137.5**	492	5	381	381	17	33
Kyber768	207	215.1	**193.5**	651	5	585	571	27	50
Kyber1024	272	287.3	**257.8**	844	4	820	789	37	80
Dilithium2	146	159	**146.3**	1008	7	410	409	17	30
Dilithium3	207	217	**202.0**	1274	12	615	594	18	34
Dilithium5	272	285	**263.6**	1680	6	840	804	32	64
LightSaber	143	Unspecified	**138.4**	512	7	385	383	15	27
Saber	207	Unspecified	**202.7**	714	6	619	602	23	47
FireSaber	272	Unspecified	**264.9**	905	5	846	814	32	72

Finally, we compare the results across the different models in Table 7. It can be seen that the proposed attack itself has the most dominant effect on the security level, decreasing it by roughly 7 bits. Our proposed sieve costs yield a similar effect, though slightly smaller. Finally, the choice of the dimensions-for-free model has a less significant effect of roughly 1 bit.

Table 7: Evaluation of the security level across models.

Candidate	Required Security Level [Nat16]	Estimated Security Level [DKL+21] [ABD+21] [BMD+20]	This Work			
			G6K [ADH+19]		Asymptotic [Duc18]	
			[AGPS20]	Section 6	[AGPS20]	Section 6
Kyber512	143	151.5	**143.1**	**137.5**	**143.8**	**138.2**
Kyber768	207	215.1	**199.5**	**193.5**	**200.5**	**194.5**
Kyber1024	272	287.3	**264.4**	**257.8**	**266.0**	**259.3**
Dilithium2	146	159	**152.2**	**146.3**	**153.4**	**147.3**
Dilithium3	207	217	**208.9**	**202.0**	**210.5**	**203.7**
Dilithium5	272	285	**270.9**	**263.6**	**273.3**	**266.2**
LightSaber	143	Unspecified	**144.1**	**138.4**	**144.8**	**139.1**
Saber	207	Unspecified	**209.3**	**202.7**	**210.4**	**203.9**
FireSaber	272	Unspecified	**271.7**	**264.9**	**273.5**	**266.5**

8 Future directions

The ideas presented in this paper are ones that have been analyzed carefully as part of our audit. We note there are further improvements possible, concrete ideas which require additional analysis to determine their impact. We hope to be able to publish some of these improvements in the future. In addition, our audit raised more general conjectural ideas that have broader applications, and which require further research. This section is meant to outline some of these ideas that we find relevant to future PQC research. Our goal with sharing these is to both point to where we believe our analysis and algorithms can be improved, and to stimulate further research into these fields and advance the understanding of these cryptosystems.

8.1 Partial Enumeration

Recall that in Algorithm 2, we first enumerate over possible values of s_{enum}, then use FFT to enumerate over possible values of s_{fft}. There are two problems with this scheme:

- When we enumerate over k coordinates, we perform $2^{kH(\chi_s)}$ guesses, while when we use FFT we use p^k guesses. When $2^{H(\chi_s)} < p$, this means the FFT is highly inefficient, and tests values of s_{fft} which are very unlikely to be the real secret.

- The rounding error induced by nonzero coordinate $\bar{s}$ of s_{fft} is $\frac{p}{\pi\bar{s}}\sin\left(\frac{\pi\bar{s}}{p}\right)$ (see Lemma 5.4). Note that it has two components: $\frac{1}{\bar{s}}$, dependent on the size of $\bar{s}$, and $\sin\left(\frac{\pi\bar{s}}{p}\right)$, dependent on the value of $\bar{s} \bmod p$. If the probability of values of $\bar{s}$ very close to $\pm p$ were lower, then the rounding error would be significantly lower.

To fix these inefficiencies, we could divide the secret distribution into two dependent random variables: $s = s^{(1)} + s^{(2)}$, enumerate directly over the value of $s^{(1)}$, and enumerate over the value of $s^{(2)}$ using FFT. If given a value of $s^{(1)}$, the distribution $s^{(2)}$ is close to uniform on $\left[-\frac{p}{2}, \frac{p}{2}\right]$, then p can be close to $2^{H\left(s^{(2)}\right)}$ and the rounding error is smaller, making the enumeration more efficient.

8.2 Partial FFT

Recall that in Algorithm 2, we enumerate over the values of s_{fft} using FFT in $(\mathbb{Z}/p\mathbb{Z})^{k_{\text{fft}}}$. We choose p to balance the rounding error, $\varepsilon_{\text{round}}$, and the FFT cost, $k_{\text{fft}}p^{k_{\text{fft}}}$. To better balance the terms, we could choose two moduli, p_1 and p_2, and perform FFT in $(\mathbb{Z}/p_1\mathbb{Z})^{k_1} \oplus (\mathbb{Z}/p_2\mathbb{Z})^{k_2}$. This would allow fine-grained control of the trade-off between $\varepsilon_{\text{round}}$ and the FFT work.

8.3 Improved Short Vectors Sampling Procedure

Recall that in Algorithm 3, we first perform BKZ to obtain a reduced basis, then perform lattice sieving to get multiple short vectors. The second application of lattice sieving cannot benefit from the "dimensions-for-free" optimization, as it has to return all sieve vectors. However, the number of vectors found when using the "dimensions-for-free" algorithm is actually a parameter, which we can optimize. To find short vectors in the lattice Λ, we choose a sublattice of small volume $\Lambda' \subset \Lambda$, and perform lattice sieving in Λ/Λ'. Usually, The rank of Λ' is chosen to be as large as possible while maintaining that the shortest vector of Λ is found by the lattice sieving in Λ/Λ'. We can choose Λ' of smaller rank, and perform lattice sieving on a lattice of higher rank, to find not only the shortest vector of Λ, but many short vectors. This could potentially accelerate the second lattice sieve performed in Algorithm 3.

8.4 Adaptations for NTRU

NTRU-based cryptosystems are among the leading candidates for lattice-based post-quantum cryptography. In this work, we propose improvements to the dual attack on LWE, and as such our attack is not immediately applicable to NTRU-based cryptosystems. It is an interesting question whether ideas from this work can be adapted to similar improvements to attacks on NTRU. Specifically, there appear to be similarities between the dual attack on LWE and the so-called "hybrid attack" [How07, Wun16] on NTRU. The hybrid attack also involves enumerating over parts of the secret, and then invoking some distinguisher to determine whether a resulting vector is close to a certain constant lattice. It seems reasonable to the writers of this paper that ideas similar to those presented here can be used to reduce the running time of such attacks as well, though anything definitive would of course require further research.

8.5 Quantum Speedups to Enumeration and FFT

The attack described in this paper can be further accelerated in the quantum setting. Recall that the attack enumerates on possible values of s_{enum} directly, and enumerates over possible values of s_{fft} using the FFT algorithm. The enumeration can be accelerated using Grover Boosting, reducing the enumeration component in the running time from $O\left(2^{k_{\text{enum}}H(\chi_s)}\right)$ to $O\left(2^{\frac{k_{\text{enum}}H(\chi_s)}{2}}\right)$, and allowing us to enumerate over twice as many coordinates of the secret at the same cost. In addition, the FFT could be replaced with QFT, which takes only polynomial time. However, after performing QFT we do not possess a table storing all the outputs, but a weighted superposition over all values of $\tilde{s}_{\text{fft}}$. Efficiently recovering s_{fft} from this superposition is a subject of further research.

8.6 Quantum Improvements to Dimensions-for-Free

It appears possible to use Grover's search [Gro96], to further increase Ducas' [Duc18] "dimensions-for-free" reduction in lattice size, given a quantum computer.

In the original "dimensions-for-free" algorithm, we search for the projection of the shortest vector in the lattice to a quotient lattice of a smaller rank. This projection is a moderately short vector in the quotient lattice. The rank of the quotient lattice is chosen such that this vector is $\sqrt{4/3}$ times longer than the shortest vector in the quotient lattice, and thus can be found by enumerating over the output of a lattice sieve in the quotient.

The basic idea of the quantum variant is to use Grover's algorithm to perform this enumeration in square root of the number of iterations. Because the cost of this enumeration is normally dominated by the sieve cost, simply adding Grover's algorithm does not improve the overall complexity. Therefore, it is necessary to enumerate over more vectors in the quotient lattice and gain more dimensions for free.

When decreasing the rank of the quotient lattice, the projection of the shortest vector becomes even longer compared to the shortest vector in the quotient lattice, such that it is not found by a lattice sieve. To find it, we first perform a lattice sieve. We then employ an algorithm for sampling moderately short vectors in the quotient lattice, which first samples a long vector, then successively shortens it using the output of a sieve until it is of moderate size. We use Grover search to find the projection of the shortest vector in the original lattice among these samples. This allows us to reduce the rank of the quotient lattice and enumerate over longer vectors in it.

Further improvements may be obtained by observing that the sampling algorithm does not need the entire output of the sieve, but rather only the vectors shorter than $c < \sqrt{4/3}$ times the shortest vector. This subset may be found by performing a sieve and discarding some of the vectors, but a more efficient way to do so is recursively applying the previous algorithm – sampling moderately short vectors in a quotient of the quotient, and searching for projections of short vectors in the quotient among them. This procedure can be continued recursively several times.

We estimate that this recursive procedure allows to significantly reduce the dimension of the lattice in which we eventually perform a full sieve, and further reduce the running time of a quantum attack on the level 1 parameters of most candidates by roughly ≈ 20 bits of security.

References

[AASA⁺19] Gorjan Alagic, Jacob Alperin-Sheriff, Daniel Apon, David Cooper, Quh Dang, Yi-Kai Liu, Carl Miller, Dustin Moody, Rene Peralta, Ray Perlner, Angela Robinson, and Daniel Smith-Tone. Status report on the first round of the NIST post-quantum cryptography standardization process. Technical report, National Institute of Standards and Technology, January 2019. doi:10.6028/nist.ir.8240. 1.1

[ABD⁺17] Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber algorithm specifications and supporting documentation, 2017. URL: https://pq-crystals.org/kyber/data/kyber-specification.pdf. 1.1

[ABD⁺21] Roberto Avanzi, Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber (version 3.02) – submission to round 3 of the nist post-quantum project, 2021. URL: https://pq-crystals.org/kyber/data/kyber-specification-round3-20210804.pdf. 1, 1, 1.3.4, 6.1, 7.1, 7.3, 3, 4, 5, 6, 7

[ABF⁺20] Martin R. Albrecht, Shi Bai, Pierre-Alain Fouque, Paul Kirchner, Damien Stehlé, and Weiqiang Wen. Faster enumeration-based lattice reduction: Root hermite factor $\mathrm{k}^{1/(2\mathrm{k})}$ time $\mathrm{k}^{\mathrm{k}/8+\mathrm{o}(\mathrm{k})}$. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology - CRYPTO 2020 - 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17-21, 2020, Proceedings, Part II*, volume 12171 of *Lecture Notes in Computer Science*, pages 186–212. Springer, 2020. doi:10.1007/978-3-030-56880-1_7. 1.1

[ACD⁺18] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. Estimate all the {LWE, NTRU} schemes! In Dario Catalano and Roberto De Prisco, editors, *Security and Cryptography for Networks - 11th International Conference, SCN 2018, Amalfi, Italy, September 5-7, 2018, Proceedings*, volume 11035 of *Lecture Notes in Computer Science*, pages 351–367. Springer, 2018. doi:10.1007/978-3-319-98113-0_19. 1, 2.2

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 284–293. ACM, 1997. doi:10.1145/258533.258604. 1.1

[ADH$^+$19] Martin R. Albrecht, Léo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn W. Postlethwaite, and Marc Stevens. The general sieve kernel and new records in lattice reduction. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 717–746. Springer, 2019. doi:10.1007/978-3-030-17656-3_25. 1.2, 2, 6.2, 7.2, 7.1, 7.3, 4, 6, 7

[AGPS20] Martin R. Albrecht, Vlad Gheorghiu, Eamonn W. Postlethwaite, and John M. Schanck. Estimating quantum speedups for lattice sieves. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II*, volume 12492 of *Lecture Notes in Computer Science*, pages 583–613. Springer, 2020. doi:10.1007/978-3-030-64834-3_20. 2, 1.3.4, 2, 6, 6.1, 6.1, 6.1, 6.2, 7.1, 7.1, 7.1, 7.3, 3, 4, 7

[AGVW17] Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer. Revisiting the expected cost of solving usvp and applications to LWE. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, volume 10624 of *Lecture Notes in Computer Science*, pages 297–322. Springer, 2017. doi:10.1007/978-3-319-70694-8_11. 1.1

[Ajt98] Miklós Ajtai. The shortest vector problem in L_2 is *NP*-hard for randomized reductions (extended abstract). In Jeffrey Scott Vitter, editor, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 10–19. ACM, 1998. doi:10.1145/276698.276705. 1.1

[ANS22] ANSSI. Anssi views on the post-quantum cryptography transition. Technical report, ANSSI, January 2022. URL: https://www.ssi.gouv.fr/en/publication/anssi-views-on-the-post-quantum-cryptography-transition. 1.1

[Ant20] Sophia Antipolis. Migration strategies and recommendations to quantum safe schemes. Technical report, European Telecommunications Standards

Institute, August 2020. URL: https://www.etsi.org/images/files/ETSIWhitePapers/QuantumSafeWhitepaper.pdf. 1.1

[BBvHL21] Gustavo Banegas, Daniel J. Bernstein, Iggy van Hoof, and Tanja Lange. Concrete quantum cryptanalysis of binary elliptic curves. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(1):451–472, 2021. doi:10.46586/tches.v2021.i1.451-472. 1.1

[BDGL16] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 10–24. SIAM, 2016. doi:10.1137/1.9781611974331.ch2. 1.3.4, 6.1, 6.2.1

[BDH+21] Ward Beullens, Jan-Pieter D'Anvers, Andreas Hülsing, Tanja Lange, Lorenz Panny, Cypriende Saint Guilhem, and Nigel P. Smart. Post-quantum cryptography - current state and quantum mitigation. Technical report, European Union Agency for Cybersecurity, May 2021. URL: https://www.enisa.europa.eu/publications/post-quantum-cryptography-current-state-and-quantum-mitigation, doi:10.2824/92307. 1.1

[BDK+18] Joppe W. Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - kyber: A cca-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy, EuroS&P 2018, London, United Kingdom, April 24-26, 2018*, pages 353–367. IEEE, 2018. doi:10.1109/EuroSP.2018.00032. 1.1, 1.2, 1.2, 7.3

[Ber10] Daniel J. Bernstein. Grover vs. mceliece. In Nicolas Sendrier, editor, *Post-Quantum Cryptography, Third International Workshop, PQCrypto 2010, Darmstadt, Germany, May 25-28, 2010. Proceedings*, volume 6061 of *Lecture Notes in Computer Science*, pages 73–80. Springer, 2010. doi:10.1007/978-3-642-12929-2_6. 1.1

[BLL+21] Lei Bi, Xianhui Lu, Junjie Luo, Kunpeng Wang, and Zhenfei Zhang. Hybrid dual attack on LWE with arbitrary secrets. *IACR Cryptol. ePrint Arch.*, page 152, 2021. URL: https://eprint.iacr.org/2021/152. 1.1

[BLP+13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 575–584. ACM, 2013. doi:10.1145/2488608.2488680. 1.1

[BMD+20] Andrea Basso, Jose Maria Bermudo Mera, Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Michiel Van Beirendonck, and Frederik Vercauteren. Saber: Mod-lwr based kem (round 3 submission), 2020. URL: https://www.esat.kuleuven.be/cosic/pqcrypto/saber/files/saberspecround3.pdf. 1.1, 1, 1, 3, 4, 5, 6, 7

[BNS19] Xavier Bonnetain, María Naya-Plasencia, and André Schrottenloher. Quantum security analysis of AES. *IACR Trans. Symmetric Cryptol.*, 2019(2):55–93, 2019. doi:10.13154/tosc.v2019.i2.55-93. 1.1

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology - EUROCRYPT 2012 - 31st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cambridge, UK, April 15-19, 2012. Proceedings*, volume 7237 of *Lecture Notes in Computer Science*, pages 719–737. Springer, 2012. doi:10.1007/978-3-642-29011-4_42. 1.1

[BPS21] William Barker, William Polk, and Murugiah Souppaya. Getting ready for post-quantum cryptography: Exploring challenges associated with adopting and using post-quantum cryptographic algorithms. Technical report, National Institute of Standards and Technology, April 2021. doi:10.6028/nist.cswp.04282021. 1.1

[BR20] Olivier Bernard and Adeline Roux-Langlois. Twisted-phs: Using the product formula to solve approx-svp in ideal lattices. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II*, volume 12492 of *Lecture Notes in Computer Science*, pages 349–380. Springer, 2020. doi:10.1007/978-3-030-64834-3_12. 1.1

[BSW18] Shi Bai, Damien Stehlé, and Weiqiang Wen. Measuring, simulating and exploiting the head concavity phenomenon in BKZ. In Thomas Peyrin and Steven D. Galbraith, editors, *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part I*, volume 11272 of *Lecture Notes in Computer Science*, pages 369–404. Springer, 2018. doi:10.1007/978-3-030-03326-2_13. 2.2, 2.2

[CCD$^+$15] Matthew Campagna, Lidong Chen, Özgür Dagdelen, Jennifer K. Fernick, Nicolas Gisin, Donald Hayford, Thomas Jennewein, Norbert Lütkenhau, Michele Mosca, Brian Neill, Mark Pecen, Ray Perlner, Grégoire Ribordy, John M. Schanck, Douglas Stebila, Nino Walenta, William Whyte, and Zhenfei Zhang. Post-quantum cryptography - current state and quantum mitigation. Technical report, European Telecommunications Standards Institute, June 2015. URL: https://www.etsi.org/images/files/ETSIWhitePapers/QuantumSafeWhitepaper.pdf. 1.1

[CJL$^+$16] Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray Perlner, and Daniel Smith-Tone. Report on {Post-Quantum} cryptography. Technical report, National Institute of Standards and Technology, April 2016. doi:10.6028/NIST.IR.8105. 1.1

[CL21] André Chailloux and Johanna Loyer. Lattice sieving via quantum random walks. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part IV*, volume 13093 of *Lecture Notes in Computer Science*, pages 63–91. Springer, 2021. doi:10.1007/978-3-030-92068-5_3. 1.1

[CN11] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2011. doi:10.1007/978-3-642-25385-0_1. 2.2, 4.4

[Dir22] Israel National Cyber Directorate. Best practices - organizational cyber readiness to the post-quantum age. Technical report, Israel National Cyber Directorate, January 2022. URL: https://www.gov.il/he/Departments/General/quantum_computing/. 1.1

[DKL$^+$18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2018(1):238–268, 2018. doi:10.13154/tches.v2018.i1.238-268. 1.1, 1.2, 1.2, 7.3

[DKL$^+$21] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium – algorithm specifications and supporting documentation (version 3.1), 2021. URL: https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf. 1.1, 1, 1, 1.3.4, 7.1, 3, 4, 5, 6, 7

[DKRV18] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. Saber: Module-lwr based key exchange, cpa-secure encryption and cca-secure KEM. In Antoine Joux, Abderrahmane Nitaj, and Tajjeeddine Rachidi, editors, *Progress in Cryptology - AFRICACRYPT 2018 - 10th International Conference on Cryptology in Africa, Marrakesh, Morocco, May 7-9, 2018, Proceedings*, volume 10831 of *Lecture Notes in Computer Science*, pages 282–305. Springer, 2018. doi:10.1007/978-3-319-89339-6_16. 1.1, 1.2, 1.2, 7.3

[DLdW20] Emmanouil Doulgerakis, Thijs Laarhoven, and Benne de Weger. Sieve, enumerate, slice, and lift: - hybrid lattice algorithms for SVP via CVPP. In Abderrahmane Nitaj and Amr M. Youssef, editors, *Progress in Cryptology - AFRICACRYPT 2020 - 12th International Conference on Cryptology in Africa, Cairo, Egypt, July 20-22, 2020, Proceedings*, volume 12174 of *Lecture Notes in Computer Science*, pages 301–320. Springer, 2020. doi:10.1007/978-3-030-51938-4_15. 1.1

[Duc18] Léo Ducas. Shortest vector from lattice sieving: A few dimensions for free. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 125–145. Springer, 2018. doi:10.1007/978-3-319-78381-9_5. 1.1, 3, 1.3.3, 2, 4, 7.2, 7.3, 3, 5, 7, 8.6

[EHH$^+$21] Stephan Ehlen, Heike Hagemeier, Tobias Hemmert, Stavros Kousidis, Manfred Lochter, Stephanie Reinhardt, and Thomas Wunderer. Kryptografie quantensicher gestalten. Technical report, Bundesamts für Sicherheit in der Informationstechnik, December 2021. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Broschueren/Kryptografie-quantensicher-gestalten.pdf. 1.1

[EJK20] Thomas Espitau, Antoine Joux, and Natalia Kharchenko. 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 Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, *Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings*, volume 12578 of *Lecture Notes in Computer Science*, pages 440–462. Springer, 2020. doi:10.1007/978-3-030-65277-7_20. 3.1, 3.1, 4.4

[GE21] Craig Gidney and Martin Ekerå. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum*, 5:433, 2021. doi:10.22331/q-2021-04-15-433. 1.1

[GJ21] Qian Guo and Thomas Johansson. Faster dual lattice attacks for solving LWE with applications to CRYSTALS. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Singapore, December 6-10, 2021, Proceedings, Part IV*, volume 13093 of *Lecture Notes in Computer Science*, pages 33–62. Springer, 2021. doi:10.1007/978-3-030-92068-5_2. 1.2, 1.4, 2, 1.4, 4, 7.1

[GLRS16] Markus Grassl, Brandon Langenberg, Martin Roetteler, and Rainer Steinwandt. Applying grover's algorithm to AES: quantum resource estimates. In Tsuyoshi Takagi, editor, *Post-Quantum Cryptography - 7th International Workshop, PQCrypto 2016, Fukuoka, Japan, February 24-26, 2016, Proceedings*, volume 9606 of *Lecture Notes in Computer Science*, pages 29–43. Springer, 2016. doi:10.1007/978-3-319-29360-8_3. 1.1

[Gri21] Roger Grimes. Practical preparations for the post-quantum world. Technical report, Cloud Security Alliance, October 2021. URL: https://cloudsecurityalliance.org/artifacts/practical-preparations-for-the-post-quantum-world. 1.1

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 212–219. ACM, 1996. doi:10.1145/237814.237866. 1.1, 8.6

[Hei21] Max Heiser. Improved quantum hypercone locality sensitive filtering in lattice sieving. *IACR Cryptol. ePrint Arch.*, page 1295, 2021. URL: https://eprint.iacr.org/2021/1295. 1.1

[HJN^{+}20] Thomas Häner, Samuel Jaques, Michael Naehrig, Martin Roetteler, and Mathias Soeken. Improved quantum circuits for elliptic curve discrete logarithms. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020, Paris, France, April 15-17, 2020, Proceedings*, volume 12100 of *Lecture Notes in Computer Science*, pages 425–444. Springer, 2020. doi:10.1007/978-3-030-44223-1_23. 1.1

[How07] Nick Howgrave-Graham. A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In Alfred Menezes, editor, *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, volume 4622 of *Lecture Notes in Computer Science*, page 150–169. Springer, 2007. doi:10.1007/978-3-540-74143-5_9. 8.4

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In Joe Buhler, editor, *Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998. doi:10.1007/BFb0054868. 1.1

[HR07] Ishay Haviv and Oded Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In *Proceedings of the 39th Annual ACM Symposium on the Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 469–477. ACM, 2007. doi:10.1145/1250790.1250859. 1.1

[HRS17] Thomas Häner, Martin Roetteler, and Krysta M. Svore. Factoring using \$2n+2\$ qubits with toffoli based modular multiplication. *Quantum Inf. Comput.*, 17(7&8):673–684, 2017. doi:10.26421/QIC17.7-8-7. 1.1

[JNRV20] Samuel Jaques, Michael Naehrig, Martin Roetteler, and Fernando Virdia. Implementing grover oracles for quantum key search on AES and lowmc. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 280–310. Springer, 2020. doi:10.1007/978-3-030-45724-2_10. 1.1

[Kho04] Subhash Khot. Hardness of approximating the shortest vector problem in lattices. In *45th Annual Symposium on Foundations of Computer Science, FOCS 2004, 17-19 October 2004, Rome, Italy, Proceedings*, pages 126–135. IEEE Computer Society, 2004. doi:10.1109/FOCS.2004.31. 1.1

[Knu98] Donald Erwin Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms*. Addison-Wesley, 1998. URL: https://www.worldcat.org/oclc/312898417. 5.3.2, 7.4

[Laa15] Thijs Laarhoven. Sieving for shortest vectors in lattices using angular locality-sensitive hashing. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2015. doi:10.1007/978-3-662-47989-6_1. 1.1

[Laa16] Thijs Laarhoven. *Search problems in cryptography: from fingerprinting to lattice sieving*. PhD thesis, Mathematics and Computer Science, February 2016. Proefschrift. URL: https://research.tue.nl/en/publications/search-problems-in-cryptography-from-fingerprinting-to-lattice-si. 1.1

[LM18] Thijs Laarhoven and Artur Mariano. Progressive lattice sieving. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018, Fort Lauderdale, FL, USA, April 9-11, 2018, Proceedings*, volume 10786 of *Lecture Notes in Computer Science*, pages 292–311. Springer, 2018. doi:10.1007/978-3-319-79063-3_14. 1.1, 6.1, 7.3

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6):43:1–43:35, 2013. doi:10.1145/2535925. 1.1

[LS15] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.*, 75(3):565–599, 2015. doi:10.1007/s10623-014-9938-4. 1.1

[MAA$^+$20] Dustin Moody, Gorjan Alagic, Daniel C Apon, David A Cooper, Quynh H Dang, John M Kelsey, Yi-Kai Liu, Carl A Miller, Rene C Peralta, Ray A Perlner, Angela Y Robinson, Daniel C Smith-Tone, and Jacob Alperin-Sheriff. Status report on the second round of the NIST post-quantum cryptography standardization process. Technical report, National Institute of Standards and Technology, July 2020. doi:10.6028/nist.ir.8309. 1.1

[Mic98] Daniele Micciancio. The shortest vector in a lattice is hard to approximate to within some constants. In *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8-11, 1998, Palo Alto, California, USA*, pages 92–98. IEEE Computer Society, 1998. doi:10.1109/SFCS.1998.743432. 1.1

[Nat16] National Institute of Standards and Technology. Submission requirements and evaluation criteria for the post-quantum cryptography standardization process, 2016. URL: https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf. 1.1, 1.2, 1.2, 1, 7.3, 3, 4, 5, 6, 7

[Pei08] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In Ran Canetti, Shafi Goldwasser, Günter Müller, and Rainer Steinwandt, editors, *Theoretical Foundations of Practical Information Security, 30.11. - 05.12.2008*, volume 08491 of *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, 2008. URL: http://drops.dagstuhl.de/opus/volltexte/2009/1892/. 1.1

[PHS19] Alice Pellet-Mary, Guillaume Hanrot, and Damien Stehlé. Approx-svp in ideal lattices with pre-processing. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part II*, volume 11477 of *Lecture Notes in Computer Science*, pages 685–716. Springer, 2019. doi:10.1007/978-3-030-17656-3_24. 1.1

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, 2009. URL: http://doi.acm.org/10.1145/1568318.1568324, doi:10.1145/1568318.1568324. 1.1, 2.3

[RNSL17] Martin Roetteler, Michael Naehrig, Krysta M. Svore, and Kristin E. Lauter. Quantum resource estimates for computing elliptic curve discrete logarithms. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part II*, volume 10625 of *Lecture Notes in Computer Science*, pages 241–270. Springer, 2017. doi:10.1007/978-3-319-70697-9_9. 1.1

[Sch03] Claus-Peter Schnorr. Lattice reduction by random sampling and birthday methods. In Helmut Alt and Michel Habib, editors, *STACS 2003, 20th Annual Symposium on Theoretical Aspects of Computer Science, Berlin, Germany, February 27 - March 1, 2003, Proceedings*, volume 2607 of *Lecture Notes in Computer Science*, pages 145–156. Springer, 2003. doi:10.1007/3-540-36494-3_14. 2.1

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 124–134. IEEE Computer Society, 1994. doi:10.1109/SFCS.1994.365700. 1.1

[SW71] Elias M Stein and Guido Weiss. *Introduction to Fourier analysis on Euclidean spaces (PMS-32)*, volume 32. Princeton University Press, 1971. 2.4

[Wun16] Thomas Wunderer. Revisiting the hybrid attack: Improved analysis and refined security. *IACR Cryptol. ePrint Arch.*, page 733, 2016. URL: https://eprint.iacr.org/2016/733. 8.4