Better Algorithms for LWE and LWR

Alexandre Duc*, Florian Tramèr, and Serge Vaudenay

EPFL, 1015 Lausanne, Switzerland

Abstract. The Learning With Error problem (LWE) is becoming more and more used in cryptography, for instance, in the design of some fully homomorphic encryption schemes. It is thus of primordial importance to find the best algorithms that might solve this problem so that concrete parameters can be proposed. The BKW algorithm was proposed by Blum et al. as an algorithm to solve the Learning Parity with Noise problem (LPN), a subproblem of LWE. This algorithm was then adapted to LWE by Albrecht et al.

In this paper, we improve the algorithm proposed by Albrecht et al. by using multidimensional Fourier transforms. Our algorithm is, to the best of our knowledge, the fastest LWE solving algorithm. Compared to the work of Albrecht et al. we greatly simplify the analysis, getting rid of integrals which were hard to evaluate in the final complexity. We also remove some heuristics on rounded Gaussians. Some of our results on rounded Gaussians might be of independent interest. Moreover, we also analyze algorithms solving LWE with discrete Gaussian noise.

Finally, we apply the same algorithm to the Learning With Rounding problem (LWR) for prime q, a deterministic counterpart to LWE. This problem is getting more and more attention and is used, for instance, to design pseudorandom functions. To the best of our knowledge, our algorithm is the first algorithm applied directly to LWR. Furthermore, the analysis of LWR contains some technical results of independent interest.

1 Introduction

The Learning With Error problem (LWE) was introduced by Regev in [43] and can be seen as an extension of the Learning (from) Parity with Noise problem (LPN). Roughly, the adversary is given queries from an LWE oracle, which returns uniformly random vectors $\boldsymbol{a}_j$ in $\mathbb{Z}_q$ and their inner-product with a fixed secret vector $\boldsymbol{s} \in \mathbb{Z}_q^k$ to which some noise was added (typically some discrete Gaussian noise). The goal of the adversary is then to recover the secret $\boldsymbol{s}$. In LPN, $q = 2$ and the noise follows a Bernoulli distribution. In his seminal paper [43], Regev shows a quantum reduction from some well-known Lattice problems like the decisional shortest vector problem (Gap-SVP) or the short independent vector problem (SIVP) to the LWE problem. Later, Peikert and Brakerski et al. showed how to make this reduction classical [16,42]. The LWE problem was then used to design a wide range of cryptographic primitives. For

* Supported by a grant of the Swiss National Science Foundation, 200021_143899/1

instance, Gentry et al. showed how to construct a trapdoor function based on LWE and created an identity-based cryptosystem [26]. Applebaum et al. used LWE to design encryption schemes with strong security properties [4]. However, the biggest breakthrough that uses LWE is its use in the design of (fully) homomorphic encryption schemes (FHE). FHE was first introduced by Gentry in his PhD thesis [25]. While the initial construction was not using the LWE problem, most of the recent designs are, e.g., [17,15,27].

The Learning With Rounding problem (LWR) was introduced by Banerjee, Peikert, and Rosen to construct pseudorandom functions [8]. LWR can be seen as a derandomization of LWE where the random noise is replaced by a rounding modulo $p < q$. This rounding introduces a deterministic error which makes the problem hard to solve. Banerjee et al. showed that the hardness of the LWE problem can be reduced to the hardness of LWR, when $q/p = k^{\omega(1)}$, where k is the length of the secret. The LWR problem was later revisited by Alwen et al. to get rid of this exponential blowup [3]. However, the number of LWR samples given to the adversary is limited in this case. LWR finds new applications every year. Among them, there is the design of pseudorandom functions [8], lossy trapdoor functions and reusable extractors [3], or key-homomorphic PRFs [13].

When designing a new cryptosystem, one critical part is to propose some concrete parameters so that the new scheme can be used in practice. Regarding the LWE problem, there was no such algorithmic analysis before the work of Albrecht et al. [1]. This lack of concrete complexity analysis implied that most of the constructions based on LWE propose only asymptotic parameters. Hence, it is of primary importance to study algorithms that solve the hard problems on which our cryptosystems rely.

Previous Work. Algorithms solving LWE can be divided into two categories: those finding short vectors in a lattice using, e.g., Regev's [43] or Brakerski et al.'s [16] reduction and those attacking the LWE problem directly. The first type of algorithms is extensively studied (see, e.g., [9,36,21,40,31,30,23,41]). However, there is still no precise complexity analysis for large dimensions. In this paper, we focus only on the second type of algorithms the study of which started with the LPN problem and the Blum—Kalai—Wasserman algorithm (BKW) [11] with complexity $2^{O(k/\log k)}$ where k is the length of the secret vector. The idea of BKW is to add queries together, such that the vectors $\boldsymbol{a}_j$ are zero in all but one positions. Then, using a majority rule, one can recover the corresponding bit of the secret with good probability.

In [35], Levieil and Fouque proposed an optimization of the BKW algorithm for LPN, denoted LF1, which recovers a full block of b bits of the secret $\boldsymbol{s}$ at once by cleverly applying a Walsh-Hadamard transform. Compared to the original BKW algorithm, their method has the advantage of making use of all the available samples after reduction, instead of having to discard those with more than one non-zero position. Instead of an exhaustive search, they use a fast Walsh-Hadamard transform to recover the most likely secret block in time $O\left(m + b2^b\right)$ (where m is the number of samples left after reduction). The analysis of their

algorithm shows that it clearly outperforms the standard BKW, although their asymptotic complexities remain the same. In the same paper, they also proposed to apply some heuristics to reduce the query complexity (LF2 algorithm).

In parallel, Fossorier et al. also improved the original BKW algorithm using techniques taken from fast correlation attacks [22]. Later, Bernstein and Lange [10] combined both the LF algorithms and Fossorier et al.'s work to attack Lapin [32], an authentication protocol based on a version of LPN over a ring (ring-LPN). However, all these results achieve the same asymptotic complexity of $2^{O(k/\log k)}$.

Many cryptographic applications of LPN make sure that the number of queries to the LPN oracle is limited. However, all the algorithms based on BKW initially require a sub-exponential number of LPN samples. In [37], Lyubashevsky proposed a clever way to combine queries using a universal hash function. He could, thus, obtain an algorithm using less queries (the minimal being $k^{1+2/\log k}$ for a worse time complexity).

In ICALP 2011, Arora and Ge publish the first algorithm targeting a specific version of LWE, namely when the Gaussian noise is low [5]. Using BKW for LWE was first mentioned by Regev [43]. However, it is only in 2013 that the first detailed analysis of a generic algorithm targeting LWE is published by Albrecht et al. [1]. It is an adaptation of the original BKW algorithm with some clever improvements of the memory usage and achieves complexity $2^{O(k)}$. Their analysis is extremely detailed and we present their results in Section 3. Finally, Albercht et al. presented in PKC 2014 an algorithm targeting LWE when the secret vector has small components (typically binary). Using BKW along with modulus switching techniques, they managed to reduce the complexity for solving the LWE problem in these cases [2].

Our Contribution. We contributed in the following:

- First we propose a *new algorithm for LWE*, which is better than the current state of the art. Our new algorithm replaces the log-likelihood part from [1] by a multidimensional Fourier transform. We also propose a heuristic adapted from LF2 [35] to reduce the number of oracle queries even further.
- Albrecht et al. in [1] were relying on the heuristic that the sum of rounded Gaussian variables remain rounded Gaussians. We *remove this heuristic* by a careful analysis. In particular, we give good bounds on the expected value of the cosine of the rounded Gaussian distribution. Our algorithm relies solely on the common heuristic stating that after having performed all the XORs in the BKW algorithm, all the noises are independent. This heuristic is already used in most of the LPN-solving algorithms (e.g. [35,22,1]).
- In [1], they consider only the rounded Gaussian distribution for the noise in LWE. Whereas this is the noise used initially by Regev, more recent papers tend to use the discrete Gaussian distribution instead. We perform our analysis for both distributions.
- Albrecht et al.'s complexity is rather difficult to estimate when $\sqrt{2^a}\sigma > q/2$ [1, Theorem 2]. Indeed, their result contains a parameter which they

could express only using an integral and the erf function. Our detailed analysis allows us to bound the Fourier coefficients of the rounded Gaussian distribution in all the cases and, hence, all our complexities are simple to evaluate.
- We adapt Lyubashevsky's idea to LWE and show that for LWE (and LWR), the minimum number of queries required using his method is $k^{1+(\log q+1)/\log k}$.
- We propose the *first algorithmic analysis of the LWR problem* when q is prime. While our proposal requires a subexponential number of samples, our detailed analysis contains many results of independent interest.

Organization. In Section 2, we introduce the LWE and LWR problems and give basic results about Gaussians and Fourier transforms. In Section 3, we present the BKW algorithm as it was done in [1]. We detail our algorithm and apply it to LWE in Section 4. We adapt it to LWR in Section 5. Finally, we conclude in Section 6.

2 Preliminaries

2.1 Notations

Given a vector $\boldsymbol{a}$ we denote by $\boldsymbol{a}_j$ its j-th component. We write $\boldsymbol{a}^{(j)}$ to say that we access the j-th vector of a set. We let $\lceil . \rfloor \colon \mathbb{R} \to \mathbb{Z}$ be the rounding function that rounds to the closest integer.[1] We define $\sqrt{-1} = i \in \mathbb{C}$. Finally, for a predicate $\pi(x)$, we denote by $\mathbb{1}_{\{\pi(x)\}}$ the function which is 1 when $\pi(x)$ is true and 0 otherwise.

2.2 The LWE Problem

In this section, we define the LWE problem.

Definition 1 (LWE Oracle). *Let k, q be positive integers. A* Learning with Error (LWE) *oracle $\Pi_{\boldsymbol{s},\chi}$ for a hidden vector $\boldsymbol{s} \in \mathbb{Z}_q^k$ and a probability distribution χ over $\mathbb{Z}_q$ is an oracle returning*

$$\left(\boldsymbol{a} \xleftarrow{U} \mathbb{Z}_q^k \, , \, \langle \boldsymbol{a}, \boldsymbol{s} \rangle + \nu\right) \, ,$$

where $\nu \leftarrow \chi$.

Definition 2 (Search-LWE). *The* Search-LWE *problem is the problem of recovering the hidden secret $\boldsymbol{s}$ given n queries $(\boldsymbol{a}^{(j)}, c^{(j)}) \in \mathbb{Z}_q^k \times \mathbb{Z}_q$ obtained from $\Pi_{\boldsymbol{s},\chi}$.*

In typical schemes based on LWE, the parameter q is taken to be polynomial in k, and χ follows a discretized Gaussian distribution (see next section).

[1] In case of equality, we take the floor.

2.3 Gaussian Distributions

Let $\mathcal{N}(0, \sigma^2)$ denote the Gaussian distribution of mean 0 and standard deviation σ. We denote its probability density function by $\phi \mapsto p(\phi; \sigma)$, for $\phi \in \mathbb{R}$. Consider the *wrapped* Gaussian distribution $\Psi_{\sigma,q}$ resulting from wrapping the Gaussian distribution around a circle of circumference $q > 0$. Its probability density function $g(\theta; \sigma, q)$ is given by

$$g(\theta; \sigma, q) \coloneqq \sum_{\ell=-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left[\frac{-(\theta + \ell q)^2)}{2\sigma^2}\right], \quad \text{for} \quad \theta \in \left]-\frac{q}{2}, \frac{q}{2}\right] . \tag{1}$$

Note that $\Psi_{\sigma,2\pi}$ is the standard wrapped normal distribution obtained by wrapping $\mathcal{N}(0, \sigma^2)$ around the unit circle, used for instance in directional statistics [39].

LWE schemes use a discretization of a Gaussian over $\mathbb{Z}_q$. There are two variants of LWE that we will consider in this paper. We will see that we obtain similar results for both distributions. In the initial version by Regev [43], the noise in LWE was a *rounded Gaussian distribution*. This is also what is considered in [1,29]. Such a distribution can be obtained by sampling from $\Psi_{\sigma,q}$ and rounding the result to the nearest integer in the interval $]\frac{-q}{2}, \frac{q}{2}]$. We denote this distribution by $\bar{\Psi}_{\sigma,q}$. Its probability mass function is given by

$$\Pr[x \leftarrow \bar{\Psi}_{\sigma,q}] = \int_{x-\frac{1}{2}}^{x+\frac{1}{2}} g(\theta; \sigma, q)\, \mathrm{d}\theta \ , \tag{2}$$

for x an integer in the interval $]\frac{-q}{2}, \frac{q}{2}]$.
The LWE problem is believed to be hard when $\sigma \geq \sqrt{k}$ and $q \in \mathsf{poly}(k)$.

The second Gaussian distribution used for LWE is the *discrete Gaussian distribution* $D_{\sigma,q}$. This distribution is used in most of the applications and in the classical LWE reduction [16]. This distribution is, for x an integer in $]-\frac{q}{2}, \frac{q}{2}]$:

$$\Pr[x \leftarrow D_{\sigma,q}] = \frac{\exp(x^2/(2\sigma^2))}{\sum_{y \in]-\frac{q}{2}, \frac{q}{2}]} \exp(y^2/(2\sigma^2))} \ . \tag{3}$$

2.4 The LWR problem

In this section, we define the LWR problem.

Definition 3 (Rounding Function). *Let $q \geq p \geq 2$ be positive integers. The LWR problem uses the rounding function from $\mathbb{Z}_q = \{0, \ldots, q-1\}$ to $\mathbb{Z}_p =$*

$\{0, \ldots, p-1\}$, *given by*[2]

$$\lceil \cdot \rfloor_p \ : \ \mathbb{Z}_q \to \mathbb{Z}_p \ : \ x \mapsto \left\lceil \left(\frac{p}{q}\right) \cdot x \right\rfloor .$$

Definition 4 (LWR Oracle). *Let k and $q \geq p \geq 2$ be positive integers. A* Learning with Rounding (LWR) *oracle $\Lambda_{\boldsymbol{s},p}$ for a hidden vector $\boldsymbol{s} \in \mathbb{Z}_q^k$, $\boldsymbol{s} \neq 0$ is an oracle returning*

$$\left(\boldsymbol{a} \xleftarrow{U} \mathbb{Z}_q^k \ , \ \lceil \langle \boldsymbol{a}, \boldsymbol{s} \rangle \rfloor_p\right) .$$

Definition 5 (Search-LWR). *The* Search-LWR *problem is the problem of recovering the hidden secret $\boldsymbol{s}$ given n queries $(\boldsymbol{a}^{(j)}, c^{(j)}) \in \mathbb{Z}_q^k \times \mathbb{Z}_p$ obtained from $\Lambda_{\boldsymbol{s},p}$.*

Two reductions from LWE to LWR exist: one with exponential parameters and another with a limited number of samples.

Theorem 6 (Theorem 3.2 in [8]). *Let $\beta \in \mathbb{R}_+$ and let χ be any efficiently sampleable distribution over $\mathbb{Z}$ such that $\Pr_{x \leftarrow \chi}[|x| > \beta]$ is negligible. Let $q \geq p \cdot \beta \cdot k^{\omega(1)}$. Then, solving decision-LWR with secrets of size k and parameters p and q is at least as hard as solving decision-LWE over $\mathbb{Z}_q$ with secret of size k and noise distribution χ.*

The second result reduces this explosion in the parameters but limits the number of samples the adversary is allowed to get from the LWR oracle.

Theorem 7 (Theorem 4.1 from [3]). *Let λ be the security parameter. Let k, ℓ, m, p, γ be positive integers, $p_{\max}$ be the largest prime divisor of q, and $p_{\max} \geq 2\beta\gamma kmp$. Let χ be a probability distribution over $\mathbb{Z}$ such that $\mathbb{E}[|\chi|] \leq \beta$. Then, if $k \geq (\ell + \lambda + 1)\log(q)/\log(2\gamma) + 2\lambda$ and if $\gcd(q, q/p_{\max}) = 1$, the decision-LWR with secret of size k, parameters p and q and limited to m queries is at least as hard as solving decision-LWE over $\mathbb{Z}_q$ with secrets of size ℓ, noise distribution χ and limited to m queries.*

2.5 Discrete Fourier Transform

Let $p_1, \cdots, p_b$ be integers and let $\theta_{p_j} \coloneqq \exp(2\pi i/p_j)$, for $1 \leq j \leq b$ and where $i = \sqrt{-1}$. Define the group $G \coloneqq \mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_b}$. We may write an element $x \in G$ as $(x_1, \cdots, x_b)$. The discrete Fourier transform (DFT) of a function $f \colon G \to \mathbb{C}$ is a function $\widehat{f} \colon G \to \mathbb{C}$ defined as

$$\widehat{f}(\alpha) \coloneqq \sum_{x \in G} f(x) \theta_{p_1}^{-\alpha_1 x_1} \cdots \theta_{p_b}^{-\alpha_b x_b} . \tag{4}$$

The discrete Fourier transform can be computed in time $O\left(|G| \log(|G|)\right) =: C_{\mathsf{FFT}} \cdot |G| \log(|G|)$ for a small constant C_{FFT}.

[2] For the second component returned by the LWR oracle, we decided to return the rounding of $\langle \boldsymbol{a}, \boldsymbol{s} \rangle$ instead of the usual $\lfloor \langle \boldsymbol{a}, \boldsymbol{s} \rangle \rfloor$. The problem is equivalent (see, e.g., [3]). However, if we would use the floor operation, the noise in Lemma 19 would not have zero mean but mean $(1/2 - \gcd(p, q)/2q)$ and we would have to introduce tedious correcting terms in (32).

2.6 Hoeffding's Inequality

We will use the following Hoeffding bound.

Theorem 8 ([33]). *Let $X_1, X_2, \ldots, X_n$ be n independent random variables such that $\Pr[X_j \in [\alpha_j, \beta_j]] = 1$ for $1 \leq j \leq n$. We define $X = X_1 + \ldots + X_n$ and $\mathbb{E}[X]$ to be the expected value of X. We have that*

$$\Pr[X - \mathbb{E}[X] \geq t] \leq \exp\left(\frac{-2t^2}{\sum_{j=1}^{n}(\beta_j - \alpha_j)^2}\right)$$

and

$$\Pr[X - \mathbb{E} \leq -t] \leq \exp\left(\frac{-2t^2}{\sum_{j=1}^{n}(\beta_j - \alpha_j)^2}\right),$$

for any $t > 0$.

3 The BKW Algorithm

The BKW algorithm [11], introduced by Blum et al., was the first sub-exponential algorithm given for solving the *Learning Parity with Noise* (LPN) problem. Asymptotically, it has a time and samples complexity of $2^{O(k/\log k)}$. Since LPN can be seen as a special case of LWE where we work over $\mathbb{Z}_2$, the BKW algorithm can be adapted to solve Search-LWE over $\mathbb{Z}_q$ with an asymptotic sample and time complexity of $q^{O(k/\log(k))} = 2^{O(k)}$ when the modulus q is polynomial in k [1,43,44].

The BKW algorithm can be described as a variant of the standard Gaussian elimination procedure, where a row addition results in the elimination of a whole block of elements instead of a single element. The main idea is that by using 'few' row additions and no row multiplications, we limit the size of the noise at the end of the reduction, allowing us to recover a small number of elements of $\boldsymbol{s}$ with high probability through maximum likelihood. The main complexity drawback of the algorithm comes from finding samples colliding on a block of elements such that their addition eliminates multiple elements at once.

The BKW algorithm takes two integer parameters, usually denoted a and b, such that $a = \lceil k/b \rceil$. The algorithm repeatedly eliminates blocks of up to b elements per row addition, over a rounds, to obtain the samples used for recovering elements of $\boldsymbol{s}$. Minimizing the complexity of the algorithm requires a tradeoff between the two parameters. For small a, the reduced samples have low noise and the complexity of recovering elements of $\boldsymbol{s}$ with high probability is reduced. For large b however, the complexity of finding colliding samples increases.

In [1], Albrecht et al. view the BKW algorithm as a linear system solving algorithm consisting of three stages, denoted *sample reduction*, *hypothesis testing* and *back substitution*. For convenience, we briefly describe each of these stages below.

Sample Reduction. Given an LWE oracle $\Pi_{\boldsymbol{s},\chi}$, the goal of this stage is to construct a series of oracles $\mathcal{A}_{\boldsymbol{s},\chi,\ell}$, each of which produces samples $(\boldsymbol{a}, c)$, where the first $b \cdot \ell$ elements of $\boldsymbol{a}$ are zero. To create the oracles $\mathcal{A}_{\boldsymbol{s},\chi,\ell}$ for $0 < \ell < a$, Albrecht et al. make use of a set of tables T^ℓ, which are maintained throughout the execution of the algorithm. To sample from $\mathcal{A}_{\boldsymbol{s},\chi,1}$, we query the oracle $\mathcal{A}_{\boldsymbol{s},\chi,0}$ (which is the original LWE oracle) to obtain samples $(\boldsymbol{a}, c)$ to be stored in table T^1. If T^1 already contains a sample $(\boldsymbol{a}', c')$ such that $\boldsymbol{a}$ and $\pm\boldsymbol{a}'$ agree on their first b coordinates, we do not store $(\boldsymbol{a}, c)$ but instead output $(\boldsymbol{a} \mp \boldsymbol{a}', c \mp c')$. If a sample from $\mathcal{A}_{\boldsymbol{s},\chi,0}$ already has its first b elements to be zero, we directly output it as a sample from $\mathcal{A}_{\boldsymbol{s},\chi,1}$.

For $1 < \ell < a$, we proceed recursively by populating a table T^ℓ of non-zero samples from $\mathcal{A}_{\boldsymbol{s},\chi,\ell-1}$ and outputting a query as soon as we get a collision in the table.

Exploiting the symmetry of $\mathbb{Z}_q$ and the fact that we do not need to store queries which are already all-zero on a block, a table T^ℓ contains at most $(q^b - 1)/2$ samples. Then, to create m samples from $\mathcal{A}_{\boldsymbol{s},\chi,\ell}$, we will need at most $m + \frac{q^b-1}{2}$ calls to $\mathcal{A}_{\boldsymbol{s},\chi,\ell-1}$. Furthermore, since there is no use in storing the zero elements from reduced samples, table T^ℓ stores samples of size $n - (\ell - 1) \cdot b + 1$ elements from $\mathbb{Z}_q$. The description of the oracles $\mathcal{A}_{\boldsymbol{s},\chi,\ell}$ is given in Algorithm 1.

In the original BKW algorithm (see [11,35]), one would then take samples from $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$, i.e., samples with zeros everywhere except in the first b positions, and delete any sample $(\boldsymbol{a}, c)$ with more than one non-zero coordinate $\boldsymbol{a}_i$. The remaining samples would be used to recover one bit of $\boldsymbol{s}$ at a time.

Albrecht et al. generalized a bit the result. Instead of keeping only one single element of the secret vector, they select a parameter $d \leq k - (a-1) \cdot b$ and create a final oracle $\mathcal{A}_{\boldsymbol{s},\chi,a}$, which produces samples with d non-zero entries at fixed positions in $\boldsymbol{a}$. These samples are used to recover d bits of s through exhaustive search over q^d values. The oracle $\mathcal{A}_{\boldsymbol{s},\chi,a}$ is defined similarly as above, making use of a final table T^a. It samples from $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$ and adds (or subtracts) queries $(\boldsymbol{a}, c)$, $(\boldsymbol{a}', c')$, for which $\boldsymbol{a}$ and $\pm\boldsymbol{a}'$ agree on coordinates $(a-1) \cdot b + 1$ through $k - d - 1$. Albrecht et al. note that they obtain the best results when choosing d equal to 1 or 2. Note that $d = 1$ corresponds to the BKW algorithm.[3]

Hypothesis testing. After the reduction phase, Albrecht et al. are left with samples $(\boldsymbol{a}, c)$ from $\mathcal{A}_{\boldsymbol{s},\chi,a}$, where $\boldsymbol{a}$ has d non-zero elements. We can view $\mathcal{A}_{\boldsymbol{s},\chi,a}$ as outputting samples in $\mathbb{Z}_q^d \times \mathbb{Z}_q$. Let $\boldsymbol{s}'$ denote the d first elements of $\boldsymbol{s}$. Since $\boldsymbol{a}$ was obtained by summing or subtracting up to 2^a samples from the LWE oracle $\Pi_{\boldsymbol{s},\chi}$ (and considering the fact that χ is symmetric around 0), the noise $(c - \langle \boldsymbol{a}, \boldsymbol{s}' \rangle)$ of the reduced samples follows the distribution of the sum of 2^a noise samples. The problem of recovering $\boldsymbol{s}'$ can then be seen as a problem of distinguishing between the noise distributions for $\boldsymbol{s}'$ and $\boldsymbol{v} \neq \boldsymbol{s}'$.

[3] The only difference between the two algorithms is that the original BKW algorithm restarts every time $\mathcal{A}_{\boldsymbol{s},\chi,a}$ outputs something.

Algorithm 1 Oracle $\mathcal{A}_{\boldsymbol{s},\chi,\ell}$, for $0 < \ell < a$

State: A table T^ℓ (initially empty)
Output: An LWE tuple $(\boldsymbol{a}, c)$ such that $\boldsymbol{a}$ has the first $b \cdot \ell$ elements set to 0.

loop
  Let $(\boldsymbol{a}, c) \leftarrow \mathcal{A}_{\boldsymbol{s},\chi,\ell-1}$.
  if $\boldsymbol{a}$ has the first $b \cdot \ell$ elements set to 0 **then**
    return $(\boldsymbol{a}, c)$.
  end if
  if there is $(\boldsymbol{a}', c') \in T^\ell$ such that $\boldsymbol{a}$ and $\pm\boldsymbol{a}'$ are equal on the first $b \cdot \ell$ positions **then**
    return $(\boldsymbol{a} \mp \boldsymbol{a}', c \mp c')$
  end if
  Add $(\boldsymbol{a}, c)$ to T^ℓ.
end loop

By performing an exhaustive search over $\mathbb{Z}_q^d$ and making use of the log-likelihood ratio, Albrecht et al. determine the number m of samples from $\mathcal{A}_{\boldsymbol{s},\chi,a}$ which should be required to recover $\boldsymbol{s}'$ with high enough probability.

As already mentioned, the analysis of the solving phase from [1] makes use of the heuristic assumption that the noise contributions of the samples from $\mathcal{A}_{\boldsymbol{s},\chi,a}$ are independent and that the sum of rounded Gaussians also follows a rounded Gaussian distribution.

Back substitution. This stage was not part of the original BKW algorithm for LPN [11,35] (which does not make use of the set of tables T defined previously either). It is analogous to the back substitution typically used in Gaussian elimination and is a clever way of reducing the size of the LWE problem after part of the secret $\boldsymbol{s}$ has been recovered.

Indeed, once d elements of $\boldsymbol{s}$ are recovered with high probability, we can perform a back substitution over the set of tables T, zeroing-out d elements in each sample. To recover the next d elements from $\boldsymbol{s}$, we query m new samples from $\Pi_{\boldsymbol{s},\chi}$ and reduce them through the tables T (which are already filled) to obtain samples for hypothesis testing. Note that as soon as we recover all the bits at positions $(\ell-1)\cdot b$ through $\ell \cdot b - 1$, the oracle $\mathcal{A}_{\boldsymbol{s},\chi,\ell}$ and its corresponding table T^ℓ become superfluous and further samples will need one reduction phase less.

4 The LWE-solving Algorithm

In this section, we present our new LWE-solving algorithm. Following the structure from [1], our algorithm will also consist of the sample reduction, hypothesis testing and back substitution phases. However, we change the hypothesis testing phase with an idea similar to the LF1 algorithm [35]. Indeed, since the Walsh-Hadamard transform can be seen as a multidimensional discrete Fourier

transform in $\mathbb{Z}_2$, it would seem plausible that a similar optimization could be achieved over $\mathbb{Z}_q$ for LWE. As we have seen, the BKW algorithm for LWE from [1] differs slightly from the original BKW algorithm in its reduction phase. Recall that after reducing samples to a block of size $k' \leq b$, Albrecht et al. further reduce the samples to d elements. Our idea is to remove this last reduction to d elements and recover directly the k' elements of $\boldsymbol{s}$ using a DFT. Thus, the samples we use for the DFT would have noise sampled from the sum of 2^{a-1} discretized Gaussians instead of 2^a, which might also lead to a significant improvement. As for most other works on LPN or LWE solving algorithms, we will make use of an heuristic assumption of independence for the noise of the reduced samples.

Finally, note that the LF1 algorithm uses the exact same reduction phase as the original BKW. Similarly, our algorithm will use (nearly) the same reduction phase as in [1], combined with a different hypothesis testing phase. The major differences in our reduction phase will be that we perform one reduction round less, and that we decide to store and re-use samples for solving successive blocks of $\boldsymbol{s}$.

4.1 Sample reduction

As mentioned previously, our algorithm uses the same reduction phase as the BKW algorithm from [1], except that we always stop the reduction as soon as we reach a block of $k' \leq b$ non-zero elements. We will construct the oracles $\mathcal{A}_{\boldsymbol{s},\chi,\ell}$ and the tables T^ℓ only for $1 \leq \ell \leq a-1$. It is thus fairly trivial to adapt the results from [1] to bound the complexity of our algorithm's reduction phase.

Lemma 9 (Lemma 2 and 3 from [1]). *Let k, q be positive integers and $\Pi_{\boldsymbol{s},\chi}$ be an LWE oracle, where $\boldsymbol{s} \in \mathbb{Z}_q^k$. Let $a \in \mathbb{Z}$ with $1 \leq a \leq k$, let b be such that $ab \leq k$, and let $k' = k - (a-1)b$. The worst case cost of obtaining m samples $(\boldsymbol{a}_i, c_i)$ from the oracle $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$, where the $\boldsymbol{a}_i$ are zero for all but the first k' elements, is upper bounded by*

$$\left(\frac{q^b-1}{2}\right)\left(\frac{(a-1)\cdot(a-2)}{2}(k+1) - \frac{ab\cdot(a-1)\cdot(a-2)}{6}\right) + m\left(\frac{a-1}{2}(k+2)\right)$$

additions in $\mathbb{Z}_q$ and $(a-1)\cdot\frac{q^b-1}{2} + m$ calls to $\Pi_{\boldsymbol{s},\chi}$.

The memory required in the worst case to store the set of tables T^1 through T^{a-1}, expressed in elements of $\mathbb{Z}_q$ is upper bounded by

$$\left(\frac{q^b-1}{2}\cdot(a-1)\cdot\left(k+1-b\frac{a-2}{2}\right)\right) .$$

Proof. The proof follows exactly the one from [1], with the exception that we do not use any table T^a. □

4.2 Hypothesis testing

At the end of the reduction phase, we are left with m samples $(\boldsymbol{a}^{(j)}, c^{(j)})$ from the oracle $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$, where each $\boldsymbol{a}^{(j)}$ has all elements equal to zero except for a block of size $k' = k-(a-1)\cdot b$. Let $\boldsymbol{s}'$ denote the corresponding block of the secret $\boldsymbol{s}$. We can view the oracle $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$ as returning samples in $\mathbb{Z}_q^{k'} \times \mathbb{Z}_q$. We will consider that each such sample is the sum of 2^{a-1} samples (or their negation) from the LWE oracle $\Pi_{\boldsymbol{s},\chi}$. Then, the noise $\langle \boldsymbol{a}^{(j)}, \boldsymbol{s}' \rangle - c^{(j)}$ will correspond to the sum of 2^{a-1} independent samples from the distribution χ, multiplied by ± 1, and taken modulo q. We perform our analysis when χ is the discrete Gaussian distribution (3) and when χ is the rounded Gaussian distribution (2) which are used in most of the LWE research, i.e., we let $\chi = D_{\sigma,q}$ or $\chi = \bar{\Psi}_{\sigma,q}$.

We represent our m samples as a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{m \times k'}$ with rows $\boldsymbol{A}_j$ and a vector $\boldsymbol{c} \in \mathbb{Z}_q^m$. Recall that $\theta_q \coloneqq \exp(2\pi i/q)$. Let us consider the function

$$f(\boldsymbol{x}) \coloneqq \sum_{j=1}^{m} \mathbb{1}_{\{\boldsymbol{A}_j = \boldsymbol{x}\}} \, \theta_q^{\boldsymbol{c}_j} \ , \quad \forall \boldsymbol{x} \in \mathbb{Z}_q^{k'} \ . \tag{5}$$

The discrete Fourier transform of f is

$$\widehat{f}(\boldsymbol{\alpha}) \coloneqq \sum_{\boldsymbol{x} \in \mathbb{Z}_q^{k'}} f(\boldsymbol{x}) \theta_q^{-\langle \boldsymbol{x}, \boldsymbol{\alpha} \rangle} = \sum_{\boldsymbol{x} \in \mathbb{Z}_q^{k'}} \sum_{j=1}^{m} \mathbb{1}_{\{\boldsymbol{A}_j = \boldsymbol{x}\}} \, \theta_q^{\boldsymbol{c}_j} \theta_q^{-\langle \boldsymbol{x}, \boldsymbol{\alpha} \rangle} = \sum_{j=1}^{m} \theta_q^{-(\langle \boldsymbol{A}_j, \boldsymbol{\alpha} \rangle - \boldsymbol{c}_j)} \ .$$

In particular, note that

$$\widehat{f}(\boldsymbol{s}') = \sum_{j=1}^{m} \theta_q^{-(\langle \boldsymbol{A}_j, \boldsymbol{s}' \rangle - c_j)} = \sum_{j=1}^{m} \theta_q^{-(\nu_{j,1} \pm \cdots \pm \nu_{j,2^{a-1}})} \ , \tag{6}$$

where the $\nu_{j,l}$ are independent samples from χ. Note that we dropped the reduction of the sum of the ν modulo q, since $\theta_q^{kq} = 1$, for $k \in \mathbb{Z}$.

We will now show, through a series of lemmas, that for appropriate values for m and a, the maximum value of the function $\mathrm{Re}(\widehat{f}(\boldsymbol{\alpha}))$ is reached by $\boldsymbol{s}'$ with high probability. Our algorithm for recovering $\boldsymbol{s}'$ will thus consist in finding the highest peak of the real part of the DFT of $f(\boldsymbol{x})$.

We start first with two technical lemmas regarding Gaussian distributions which might be of independent interest.

Lemma 10. *For q an odd integer, let $X \sim \bar{\Psi}_{\sigma,q}$ and let $Y \sim 2\pi X/q$. Then*

$$\mathbb{E}[\cos(Y)] \geq \frac{q}{\pi} \sin\left(\frac{\pi}{q}\right) e^{-2\pi^2\sigma^2/q^2} \quad \textit{and} \quad \mathbb{E}[\sin(Y)] = 0 \ .$$

Proof. Let S_ℓ be the set of integers in $]-q/2+\ell q, q/2+\ell q]$. Using (1) and (2), we can write

$$\mathbb{E}[\cos(Y)] = \sum_{x\in S_0} \cos\left(\frac{2\pi}{q}x\right) \sum_{\ell=-\infty}^{\infty} \int_{x-1/2}^{x+1/2} p(\theta+\ell q;\sigma)\,\mathrm{d}\theta \tag{7}$$

$$= \sum_{\ell=-\infty}^{\infty} \sum_{x\in S_0} \cos\left(\frac{2\pi}{q}x + 2\pi\ell\right) \int_{x-1/2}^{x+1/2} p(\theta+\ell q;\sigma)\,\mathrm{d}\theta \tag{8}$$

$$= \sum_{\ell=-\infty}^{\infty} \sum_{x\in S_0} \cos\left(\frac{2\pi}{q}(x+\ell q)\right) \int_{x-1/2+\ell q}^{x+1/2+\ell q} p(\theta;\sigma)\,\mathrm{d}\theta \tag{9}$$

$$= \sum_{\ell=-\infty}^{\infty} \sum_{x'\in S_\ell} \cos\left(\frac{2\pi}{q}x'\right) \int_{x'-1/2}^{x'+1/2} p(\theta;\sigma)\,\mathrm{d}\theta \tag{10}$$

$$= \sum_{x'=-\infty}^{\infty} \cos\left(\frac{2\pi}{q}x'\right) \int_{x'-1/2}^{x'+1/2} p(\theta;\sigma)\,\mathrm{d}\theta \tag{11}$$

$$= \sum_{\chi=-\infty}^{\infty} \mathcal{F}\left(\cos\left(x\frac{2\pi}{q}\right) \int_{x-1/2}^{x+1/2} p(\theta;\sigma)\,\mathrm{d}\theta\right)(\chi)\,, \tag{12}$$

where, for (10), we used $x' \coloneqq x+\ell q$ and, for (12), we used the Poisson summation formula (Lemma 25 in Appendix A). Basics about continuous Fourier transforms can be found in Appendix A. The Fourier transforms of $\cos(2\pi x/q)$ and $1/(\sigma\sqrt{2\pi})\exp[-x/(2\sigma^2)]$ can be found in Appendix A. We are now ready to prove the lemma (we drop some (χ) for readability). For integer values of χ, we have

$$\mathcal{F}\left(\cos\left(x\frac{2\pi}{q}\right)\int_{x-1/2}^{x+1/2}\frac{1}{\sigma\sqrt{2\pi}}e^{-\theta^2/(2\sigma^2)}\,\mathrm{d}\theta\right) \tag{13}$$

$$= \mathcal{F}\left(\cos\left(x\frac{2\pi}{q}\right)\right) * \left(\mathcal{F}\left(\int_{-\infty}^{x+\frac{1}{2}}\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{\theta^2}{2\sigma^2}}\,\mathrm{d}\theta\right) - \mathcal{F}\left(\int_{-\infty}^{x-\frac{1}{2}}\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{\theta^2}{2\sigma^2}}\,\mathrm{d}\theta\right)\right) \tag{14}$$

$$= \mathcal{F}\left(\cos\left(x\frac{2\pi}{q}\right)\right) * \left(\left(e^{\pi i\chi} - e^{-\pi i\chi}\right)\mathcal{F}\left(\int_{-\infty}^{x}\frac{1}{\sigma\sqrt{2\pi}}e^{-\theta^2/(2\sigma^2)}\,\mathrm{d}\theta\right)\right) \tag{15}$$

$$= \frac{1}{2}\left(\delta\left(\chi-\frac{1}{q}\right)+\delta\left(\chi+\frac{1}{q}\right)\right) * \left(\left(e^{\pi i\chi} - e^{-\pi i\chi}\right)\left(\frac{1}{2\pi i\chi}e^{-2\pi^2\sigma^2\chi^2} + \frac{1}{2}\delta(\chi)\right)\right) \tag{16}$$

$$= \frac{1}{2}\left(\delta\left(\chi-\frac{1}{q}\right)+\delta\left(\chi+\frac{1}{q}\right)\right) * \left(\sin(\pi\chi)\left(\frac{1}{\pi\chi}e^{-2\pi^2\sigma^2\chi^2}\right)\right) \tag{17}$$

$$= \frac{q}{2\pi}\sin\left(\frac{\pi}{q}\right)(-1)^\chi\left(\frac{e^{-2\pi^2\sigma^2(q\chi+1)^2/q^2}}{q\chi+1} - \frac{e^{-2\pi^2\sigma^2(q\chi-1)^2/q^2}}{q\chi-1}\right)\,, \tag{18}$$

where (14) is the convolution property of the FT, (15) comes from the translation property of the FT, (16) comes from the integration property of the FT, and

(17) holds since $\delta(\chi \pm 1/q) * \delta(\chi) = 0$ for integer values of χ. We can write (12) as

$$\frac{q}{\pi}\sin\left(\frac{\pi}{q}\right)\exp^{-2\pi^2\sigma^2/q^2} + \sum_{\chi=1}^{\infty}\frac{q}{\pi}\sin\left(\frac{\pi}{q}\right)(-1)^{\chi}\left(\frac{e^{-2\pi^2\sigma^2(q\chi+1)^2/q^2}}{q\chi+1} - \frac{e^{-2\pi^2\sigma^2(q\chi-1)^2/q^2}}{q\chi-1}\right) .$$

Notice that the sum term in this equation is alternating and decreasing in absolute value when χ grows (derivative is negative). Notice also that the first term (when $\chi = 1$) is positive. Hence this sum is greater than 0 and we get our result for $\mathbb{E}[\cos(Y)]$.

For $\mathbb{E}[\sin(Y)]$, note that when q is odd, X and Y are perfectly symmetric around 0. The result then follows trivially from the symmetry of the sine function. □

Lemma 11. *For q an odd integer, let $X \sim D_{\sigma,q}$ and let $Y \sim 2\pi X/q$. Then*

$$\mathbb{E}[\cos(Y)] \geq 1 - \frac{2\pi^2\sigma^2}{q^2} \quad \textit{and} \quad \mathbb{E}[\sin(Y)] = 0 .$$

Proof. Using [7, Lemma 1.3] with $a = 1/(2\sigma^2)$, we have that $\mathbb{E}[X^2] \leq \sigma^2$. Hence, using $\cos(x) \geq 1 - x^2/2$,

$$\mathbb{E}[\cos(2\pi X/q)] \geq 1 - 2\pi^2\mathbb{E}[X^2]/q^2 = 1 - 2\pi^2\sigma^2/q^2 .$$

For $\mathbb{E}[\sin(Y)]$, note that when q is odd, X and Y are perfectly symmetric around 0. The result then follows trivially from the symmetry of the sine function. □

Definition 12 ($R_{\sigma,q,\chi}$). *In the following, let $R_{\sigma,q,\chi} \coloneqq \mathbb{E}[\cos(\chi)]$, i.e.,*

$$R_{\sigma,q,\chi} \coloneqq \begin{cases} \frac{q}{\pi}\sin\left(\frac{\pi}{q}\right)e^{-2\pi^2\sigma^2/q^2} & \textit{when } \chi = \bar{\Psi}_{q,\sigma} \\ 1 - \frac{2\pi^2\sigma^2}{q^2} & \textit{when } \chi = D_{q,\sigma} \end{cases}$$

Lemma 13. $\mathbb{E}\left[\mathrm{Re}(\widehat{f}(\boldsymbol{s}'))\right] \geq m \cdot \left(R_{\sigma,q,\chi}\right)^{2^{a-1}}$.

Proof. From (6), we get

$$\mathbb{E}\left[\mathrm{Re}(\widehat{f}(\boldsymbol{s}'))\right] = \mathrm{Re}\left(\sum_{j=1}^{m}\mathbb{E}\left[\theta_q^{-(\nu_{j,1}\pm\cdots\pm\nu_{j,2^{a-1}})}\right]\right) = \mathrm{Re}\left(\sum_{j=1}^{m}\mathbb{E}\left[\cos\left(\frac{2\pi}{q}\nu_{j,1}\right)\right]^{2^{a-1}}\right) ,$$

using the independence of the noise samples $\nu_{j,\ell}$ and $\mathbb{E}[\theta_q^{\pm\nu_{j,\ell}}] = \mathbb{E}[\cos(2\pi\nu_{j,\ell}/q)]$ (which follows from Lemmas 10 and 11). Using Lemmas 10 and 11 again, we have that $\mathbb{E}[\cos(2\pi\nu_{j,\ell}/q)] \geq R_{\sigma,q,\chi}$. Hence, we get that

$$\mathbb{E}\left[\mathrm{Re}(\widehat{f}(\boldsymbol{s}'))\right] > \sum_{j=1}^{m}\left(R_{\sigma,q,\chi}\right)^{2^{a-1}} = m \cdot \left(R_{\sigma,q,\chi}\right)^{2^{a-1}} . \tag{19}$$

□

Lemma 14. *Let $G \subseteq \mathbb{Z}_q$ be a subgroup of $\mathbb{Z}_q$, let $X \xleftarrow{U} G$ and let $e \in \mathbb{Z}_q$ be independent from X. Then, $\mathbb{E}\left[\theta_q^{X+e}\right] = 0$.*

Proof. Define $Y = \frac{2\pi}{q}X$. Then Y is a random variable following a discrete uniform distribution on the unit circle. Then $\mathbb{E}[\theta_q^X] = 0$ follows from the analysis of discrete circular uniform distributions (see e.g. [6]). Now, since X and e are independent, $\mathbb{E}[\theta_q^{X+e}] = \mathbb{E}[\theta_q^X]\mathbb{E}[\theta_q^e] = 0$. □

Lemma 15. $\arg\max_{\boldsymbol{\alpha}} \mathrm{Re}(\widehat{f}(\boldsymbol{\alpha})) = \boldsymbol{s}'$ *with probability greater than*

$$1 - q^{k'} \cdot \exp\left(-\frac{m}{8} \cdot (R_{\sigma,q,\chi})^{2^a}\right) .$$

Proof. A similar proof is proposed for LPN in [12]. We are looking to upper bound the probability that there is some $\boldsymbol{\alpha} \neq \boldsymbol{s}'$ such that $\mathrm{Re}(\widehat{f}(\boldsymbol{\alpha})) \geq \mathrm{Re}(\widehat{f}(\boldsymbol{s}'))$. Using a union bound, we may upper bound this by $q^{k'}$ times the probability that $\mathrm{Re}(\widehat{f}(\boldsymbol{\alpha})) \geq \mathrm{Re}(\widehat{f}(\boldsymbol{s}'))$ for some fixed vector $\boldsymbol{\alpha} \in \mathbb{Z}_q^{k'}$, $\boldsymbol{\alpha} \neq \boldsymbol{s}'$ which is the probability that

$$\sum_{j=1}^{m} \left(\mathrm{Re}\left(\theta_q^{-(\langle \boldsymbol{A}_j, \boldsymbol{s}'\rangle - \boldsymbol{c}_j)}\right) - \mathrm{Re}\left(\theta_q^{-(\langle \boldsymbol{A}_j, \boldsymbol{\alpha}\rangle - \boldsymbol{c}_j)}\right)\right) \leq 0 .$$

Let $\boldsymbol{y} = \boldsymbol{\alpha} - \boldsymbol{s}' \in \mathbb{Z}_q^{k'}$. Also, define $e_j \coloneqq \langle \boldsymbol{A}_j, \boldsymbol{s}'\rangle - \boldsymbol{c}_j$, for $1 \leq j \leq m$. Then, $\langle \boldsymbol{A}_j, \boldsymbol{\alpha}\rangle - \boldsymbol{c}_j = \langle \boldsymbol{A}_j, \boldsymbol{y}\rangle + e_j$. Note that since $\boldsymbol{A}_j$ is uniformly distributed at random, independently from e_j, and $\boldsymbol{y}$ is fixed and non-zero, $\langle \boldsymbol{A}_j, \boldsymbol{y}\rangle$ is uniformly distributed in a subgroup of $\mathbb{Z}_q$, and thus so is $\langle \boldsymbol{A}_j, \boldsymbol{\alpha}\rangle - \boldsymbol{c}_j$. Hence, we can apply Lemma 14.

From our heuristic assumption, we will consider $X_1, X_2, \ldots, X_m$ to be independent random variables with $X_j = u_j - v_j$, where

$$u_j = \mathrm{Re}\left(\theta_q^{-(\langle \boldsymbol{A}_j, \boldsymbol{s}'\rangle - \boldsymbol{c}_j)}\right) \qquad \text{and} \qquad v_j = \mathrm{Re}\left(\theta_q^{-(\langle \boldsymbol{A}_j, \boldsymbol{\alpha}\rangle - \boldsymbol{c}_j)}\right) . \tag{20}$$

Note that $X_j \in [-2, 2]$ for all j. Furthermore, let $X = \sum_{j=1}^{m} X_j$. Using Lemmas 13 (for the u_j's) and 14 (for the v_j's), we get that

$$\mathbb{E}[X] \geq m \cdot (R_{\sigma,q,\chi})^{2^{a-1}} . \tag{21}$$

We will bound the probability that $X \leq 0$ using Hoeffding's inequality (Theorem 8). Let $t = \mathbb{E}[X] > 0$. Then,

$$\begin{aligned} \Pr[X \leq 0] = \Pr\left[\,(X - \mathbb{E}[X]) \leq -\mathbb{E}[X]\right] &\leq \exp\left(\frac{-2(\mathbb{E}[X])^2}{16m}\right) \\ &\leq \exp\left(-\frac{m}{8} \cdot (R_{\sigma,q,\chi})^{2^a}\right) . \end{aligned} \tag{22}$$

Applying the aforementioned union-bound, we get the desired result. □

We are now ready to derive the number of samples m required to recover the correct secret block $\boldsymbol{s}'$ with high probability.

Theorem 16. *Let k, q be positive integers and $\Pi_{\boldsymbol{s},\chi}$ be an LWE oracle, where $\boldsymbol{s} \in \mathbb{Z}_q^k$. Let $a \in \mathbb{Z}$ with $1 \leq a \leq k$, let b be such that $ab \leq k$, and let $k' = k - (a-1)b$. Let $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$ be the oracle returning samples $(\boldsymbol{a}_i, c_i)$ where the $\boldsymbol{a}_i$ are zero for all but the first k' elements. Denote the vector consisting of the first k' elements of $\boldsymbol{s}$ as $\boldsymbol{s}'$. Fix an $\epsilon \in (0,1)$. Then, the number of independent samples m^{LWE} from $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$, which are required such that we fail to recover the secret block $\boldsymbol{s}'$ with probability at most ϵ satisfies*

$$m^{LWE} \geq \begin{cases} 8 \cdot k' \cdot \log\left(\frac{q}{\epsilon}\right) \cdot \left(\frac{q}{\pi} \sin\left(\frac{\pi}{q}\right) e^{-2\pi^2\sigma^2/q^2}\right)^{-2^a} & \text{when } \chi = \bar{\Psi}_{\sigma,q} \\ 8 \cdot k' \cdot \log\left(\frac{q}{\epsilon}\right) \cdot \left(1 - \frac{2\pi^2\sigma^2}{q^2}\right)^{-2^a} & \text{when } \chi = D_{\sigma,q}\ . \end{cases}$$

Furthermore, the hypothesis testing phase (the FFT phase in Algorithm 2) that recovers $\boldsymbol{s}'$ requires $2m^{LWE} + C_{\mathsf{FFT}} \cdot k' \cdot q^{k'} \cdot \log q$ operations in $\mathbb{C}$ and requires storage for $q^{k'}$ complex numbers, where C_{FFT} is the small constant in the complexity of the FFT.[4]

Proof. For a fixed m, we get

$$\epsilon = \Pr\left[\exists\ \boldsymbol{\alpha} \neq \boldsymbol{s}'\ :\ \mathrm{Re}(\widehat{f}(\boldsymbol{\alpha})) \geq \mathrm{Re}(\widehat{f}(\boldsymbol{s}'))\right] < q^{k'} \cdot \exp\left(-\frac{m}{8} \cdot (R_{\sigma,q,\chi})^{2^a}\right)\ .$$

Solving for m, we get the desired result.

Concerning the algorithmic and memory complexities, we need to store the values of the function $f(\boldsymbol{x})$ as $q^{k'}$ elements from $\mathbb{C}$. For each of the m^{LWE} samples we receive from $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$, we compute an exponentiation and an addition in $\mathbb{C}$ to update $f(\boldsymbol{x})$ and then discard the sample. Finally, computing the discrete Fourier transform of f can be achieved with $C_{\mathsf{FFT}} \cdot k' \cdot q^{k'} \cdot \log q$ complex operations, and no additional memory, using an in-place FFT algorithm. □

The hypothesis testing part of the algorithm is summarized in Algorithm 2.

4.3 Back substitution

We use a similar back substitution mechanism as the one described in [1]. Note that we have to apply back substitution on one table less, since we performed only $a-1$ reductions. Furthermore, since we recovered a complete block of $\boldsymbol{s}$, the table T^{a-1} would be completely zeroed-out by back substitution and can

[4] One might comment on the required precision needed to compute the DFT. For this, we set our precision to $O\left(\log(m(R_{\sigma,q,\chi})^{2^a})\right)$ bits which is the expected size of our highest peak in the DFT. Using this result along with some standard results about the exact complexity to compute a DFT with a given precision (see, e.g., [18]), the ratio between our (binary) complexities and the binary complexities of [1] remain the same.

Algorithm 2 Hypothesis testing algorithm for LWE.

Input: m independent LWE samples with only $k' \coloneqq k-(a-1)b$ non-zero components in $\boldsymbol{a}$. We represent our samples as a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{m \times k'}$ and a vector $\boldsymbol{c} \in \mathbb{Z}_q^m$.

Output: A vector consisting of the k' elements of $\boldsymbol{s}$ that are at the non-zero positions of $\boldsymbol{a}$

1: Compute the fast Fourier Transform $\widehat{f}(\boldsymbol{\alpha})$ of the function $f(\boldsymbol{x})\colon \sum_{j=1}^{m} \mathbb{1}_{\boldsymbol{A}_j=\boldsymbol{x}} \theta_q^{\boldsymbol{c}_j}$

2: **return** $\arg\max_{\boldsymbol{\alpha} \in \mathbb{Z}_q^{k'}} \widehat{f}(\boldsymbol{\alpha})$

therefore simply be dropped after the hypothesis testing phase. Finally, we do not discard the m^{LWE} queries from $\Pi_{\boldsymbol{s},\chi}$, which were reduced and then used for the solving phase. Instead, we store these m^{LWE} original queries and re-use $m' < m^{\mathrm{LWE}}$ of these queries for the next block of $\boldsymbol{s}$.

4.4 Complexity of BKW with multidimensional DFT

We now have all the results we need in order to state the total complexity of solving SEARCH-LWE with our algorithm. For ease of notation, we will consider from here on that the parameters a and b are chosen such that $k = a \cdot b$. Note that the general case, where $k = (a-1) \cdot b + k'$, follows similarly from our previous results.

Theorem 17 (Complexity of SEARCH-LWE). *Let k, q be positive integers and $\Pi_{\boldsymbol{s},\chi}$ be an LWE oracle, where $\boldsymbol{s} \in \mathbb{Z}_q^k$. Let $a, b \in \mathbb{N}$ be such that $a \cdot b = k$. Let C_{FFT} be the small constant in the complexity of the fast Fourier transform computation. Let $0 < \epsilon < 1$ be a targeted success rate and define $\epsilon' \coloneqq (1-\epsilon)/a$. For $0 \le j \le a-1$, let*

$$m_{j,\epsilon}^{LWE} \coloneqq \begin{cases} 8 \cdot b \cdot \log\left(\frac{q}{\epsilon}\right) \cdot \left(\frac{q}{\pi} \sin\left(\frac{\pi}{q}\right) e^{-2\pi^2\sigma^2/q^2}\right)^{-2^{a-j}} & \text{when } \chi = \bar{\Psi}_{\sigma,q} \\ 8 \cdot b \cdot \log\left(\frac{q}{\epsilon}\right) \cdot \left(1 - \frac{2\pi^2\sigma^2}{q^2}\right)^{-2^{a-j}} & \text{when } \chi = D_{\sigma,q}\ . \end{cases}$$

Under the standard heuristic that all the samples after reduction are independent (which was also used in the previous work), the time complexity of our algorithm to recover the secret $\boldsymbol{s}$ with probability at least ϵ is $c_1 + c_2 + c_3 + c_4$, where

$$c_1 \coloneqq \left(\frac{q^b - 1}{2}\right) \cdot \left(\frac{(a-1)\cdot(a-2)}{2}(k+1) - \frac{b}{6}(a \cdot (a-1) \cdot (a-2))\right) \tag{23}$$

is the number of additions in $\mathbb{Z}_q$ to produce all tables T^j, $0 \le j \le a-1$,

$$c_2 \coloneqq \sum_{j=0}^{a-1} m_{j,\epsilon'}^{LWE} \cdot \frac{a-1-j}{2} \cdot (k+2) \tag{24}$$

is the number of additions in $\mathbb{Z}_q$ to produce the samples required to recover all blocks of $\boldsymbol{s}$ with probability ϵ,

$$c_3 \coloneqq 2\left(\sum_{j=0}^{a-1} m_{j,\epsilon'}^{LWE}\right) + C_{\mathsf{FFT}} \cdot k \cdot q^b \cdot \log(q) \tag{25}$$

is the number of operations in $\mathbb{C}$ to prepare and compute the DFTs,and

$$c_4 \coloneqq (a-1)\cdot(a-2)\cdot b \cdot \frac{q^b-1}{2} \tag{26}$$

is the number of operations in $\mathbb{Z}_q$ for back substitution.

The number of calls to the oracle $\Pi_{\boldsymbol{s},\chi}$ is

$$(a-1)\cdot\frac{q^b-1}{2} + m_{0,\epsilon}^{LWE}\,. \tag{27}$$

Finally, the memory complexity in number of elements from $\mathbb{Z}_q$ and $\mathbb{C}$ are respectively

$$\left(\frac{q^b-1}{2}\cdot(a-1)\cdot\left(k+1-b\frac{a-2}{2}\right)\right) + m_{0,\epsilon}^{LWE} \quad \textit{and} \quad q^b\,. \tag{28}$$

Proof. To recover $\boldsymbol{s}$, we need to recover each block of $\boldsymbol{s}$ successfully. Since we are making use of the same set of tables T and reduced queries for each block, these events are not independent. Using a union bound, and a failure probability bounded by $(1-\epsilon)/a$ for each of the a blocks thus leads to a overall success probability of at least ϵ.

- The cost of constructing the set of tables T in (23) is given by Lemma 9. Note that theses tables are constructed only once and maintained throughout the execution of the algorithm.
- As per Lemma 9, the cost of obtaining m samples from the oracle $\mathcal{A}_{\boldsymbol{s},\chi,a-1}$ is upper bounded by $m\cdot\frac{a-1}{2}\cdot(k+2)$. Noting that after solving the jth block, the table T^j is dropped, the result in (24) follows.
- The DFT has to be applied a times, for each block of size b. Since the number of samples required is updated for each block, we get equation (25).
- After solving the first block, back substitution has to be applied to $a-2$ tables (table T^{a-1} can be dropped). Per table, the substitution has cost $2b$ for each of the $\frac{q^b-1}{2}$ rows. In total, we get a cost of $\sum_{j=1}^{a-2} 2\cdot b\cdot\left(i\cdot\frac{q^b-1}{2}\right)$, as in (26).
- The required number of oracle samples follows from Lemma 9. Note that the samples needed to fill up the tables are required only once and that the $m_{0,\epsilon}^{\mathrm{LWE}}$ additional queries are stored and can be reused for each block of $\boldsymbol{s}$ since $m_{0,\epsilon}^{\mathrm{LWE}} > m_{j,\epsilon}^{\mathrm{LWE}}$ for $j>0$. This gives us the total from (27).

– Finally, the storage cost for the tables follows from Lemma 9. In addition, we need an array of size q^b to store the complex function on which we apply the DFT (we assume an in-place DFT algorithm requiring no extra storage). We also store the $m_{0,\epsilon}^{\mathrm{LWE}}$ samples queried to solve the first block. Combining these results gives us (28).

□

4.5 Using Fewer Samples

If the number of queries to the LWE oracle is limited we can use an idea introduced by Lyubashevsky [37]. The idea is to use a universal family of hash function to combine samples and create new ones. However, these new samples will have higher noise.

Theorem 18. *Let $\epsilon \geq (\log q+1)/\log k$. Then, one can convert an LWE instance $\Pi_{\boldsymbol{s},\chi}$ where χ is $\bar{\Psi}_{\sigma,q}$ (resp. $D_{\sigma,q}$) and using $k^{1+\epsilon}$ samples into an LWE instance $\Pi_{\boldsymbol{s},\chi'}$ where χ' is $\bar{\Psi}_{\sigma\lceil(\log q+1)k/(\epsilon\log k)\rceil,q}$ (resp. $D_{\sigma\lceil(\log q+1)k/(\epsilon\log k)\rceil}$) without any sample limit.*

Proof (sketch). The proof is exactly the same as in [37] except for few differences that we state here. We let our samples be $A = \boldsymbol{a}^{(1)}, \cdots \boldsymbol{a}^{(k^{1+\epsilon})} \in \mathbb{Z}_q^k$. Let also $X \subset \{0,1\}^{k^{1+\epsilon}}$ with $x \in X$ if $\sum_j x_j = \lceil(\log(q)+1)k/(\epsilon\log k)\rceil$. We use the following universal family of hash function $H \coloneqq \left\{h_A \colon X \leftarrow \mathbb{Z}_q^k\right\}$ where A is defined above and $h_A(x) \coloneqq x_1 a^{(1)} + \cdots + x_{k^{1+\epsilon}} a^{(k^{1+\epsilon})}$. By the Leftover Hash Lemma [34], when A and x are uniformly distributed, with probability greater than $1-2^{-k/4}$, $\Delta(h_A(x), U) \leq 2^{-n/4}$, where U is the uniform probability distribution over Z_q^k. Note that the Leftover Hash Lemma holds since

$$|X| \geq \left(\frac{k^{1+\epsilon}}{\lceil(\log q+1)k/(\epsilon\log k)\rceil}\right)^{\lceil(\log q+1)k/(\epsilon\log k)\rceil} \geq q^k \ ,$$

when $\epsilon \geq (\log q+1)/\log k$. □

The LF2 Heuristic. In [35], Levieil and Fouque propose LF2, an heuristic improvement for the reduction phase of their LPN solving algorithm LF1. The main idea of LF2 is to compute the sum (or difference) of *any* pair of samples $(\boldsymbol{a}, c)$ and $(\boldsymbol{a}', c')$, which agree on b particular coordinates. Thus, in an entry of a reduction table T^i, we would store not only one, but all samples agreeing (up to negation) on b coordinates. Then, when reducing a sample $(\boldsymbol{a}, c)$, we could output $(\boldsymbol{a} \pm \boldsymbol{a}', c \pm c')$ for *each* sample $(\boldsymbol{a}', c')$ in the corresponding table entry. Note that if we have x samples agreeing on b positions, we can output $\binom{x}{2}$ reduced samples.

An interesting case arises when we take exactly $3 \cdot q^b/2$ oracle samples. In the worst case, we get exactly 3 samples per entry in table T^1. Then, applying

all the pairwise reductions, we again get $3 \cdot q^b/2$ samples to be stored in table T^2 and so forth. Hence, if we take

$$\max\left\{m_{0,\epsilon'}^{\mathrm{LWE}}, 3 \cdot q^b/2\right\} \tag{29}$$

oracle queries, we are ensured to have enough samples for the Fourier transform. We could thus solve the LWE problem using fewer oracle samples than in Theorem 17 and with a similar time complexity, at the expense of a higher memory complexity (to store multiple samples per table entry).

4.6 Results

We computed the number of operations needed in $\mathbb{Z}_q$ to solve the LWE problem for various values of k when the parameters are chosen according to Regev's cryptosystem [43] and $\epsilon = 0.99$. In this scheme, q is a prime bigger than k^2 and $\sigma = q/(\sqrt{k}\log^2(k)\sqrt{2\pi})$. For our table, we took q to be the smallest prime greater than k^2. Our results are displayed in Table 1.[5] To simplify our result, we considered operations over $\mathbb{C}$ to have the same complexity as operations over $\mathbb{Z}_q$. We also took $C_{\mathsf{FFT}} = 1$ which is the best one can hope to obtain for a FFT. Regarding the noise distribution, we obtained the same results for both $D_{\sigma,q}$ and $\bar{\Psi}_{\sigma,q}$. If we compare our results with [1, Table1], we see that we are better in all the cases.[6] This improvement with respect to log likelihood comes from the fact that we do one reduction less in our reduction phase as we recover a full block instead of a single element in $\mathbb{Z}_q$. This implies that our noise is going to be smaller and, hence, we will need a lower number of queries. However, we still achieve the same asymptotic complexity.

5 Applying our Algorithm to LWR

In this section, we try to apply a similar algorithm to LWR. In the following, we will always consider q to be *prime*.

Lemma 19. *Let k and $q > p \geq 2$ be positive integers, q prime. Let $(\boldsymbol{a}, c)$ be a random sample from an LWR oracle $\Lambda_{\boldsymbol{s},p}$. Then, the "rounding error", given by $\xi = (p/q)\langle \boldsymbol{a}, \boldsymbol{s}\rangle - c$, follows a uniform distribution in a discrete subset of $[-1/2, 1/2]$ with mean zero.*

Furthermore, for $\gamma \in \mathbb{R}_{\neq 0}$,

$$\mathbb{E}\left[e^{\pm i\xi\gamma}\right] = \frac{1}{q} \cdot \frac{\sin(\frac{\gamma}{2})}{\sin(\frac{\gamma}{2q})} \,. \tag{30}$$

[5] The code used to compute these value is available on our website `lasec.epfl.ch/lwe/`

[6] Albrecht et al. simplified their complexity by considering non-integer a which explains why the difference between our results varies depending on k.

k	q	a	$\log(\#\mathbb{Z}_q)$	$\log(m)$	$\log(m)$ for LF2	$\log(\#\mathbb{Z}_q)$ in [1]
64	4 099	19	52.62	43.61	41.01	54.85
80	6 421	20	63.23	53.85	51.18	65.78
96	9 221	21	73.72	63.95	61.98	76.75
112	12 547	21	85.86	75.94	73.20	87.72
128	16 411	22	95.03	84.86	82.05	98.67
160	25 601	23	115.87	105.33	102.46	120.43
224	50 177	24	160.34	149.26	146.32	163.76
256	65 537	25	178.74	167.43	164.43	185.35
384	147 457	26	269.18	257.23	254.17	–
512	262 147	27	357.45	345.03	341.92	–

Table 1. We write $\#\mathbb{Z}_q$ for the worst case cost (in operations over $\mathbb{Z}_q$) of solving Search-LWE for various parameters for the Regev cryptosystem [43] when $\epsilon = 0.99$ according to Theorem 17. We provide also the value of a that minimizes the complexity, the number of queries (m) according to (27), and the number of queries (m) when we apply the LF2 heuristic (29).

Proof. We first prove the first part of the lemma. We will prove that for any $\alpha \in [\frac{-q+1}{2}, \ldots, \frac{q-1}{2}]$, ξ takes the value α/q with probability $1/q$. We have $p \cdot \langle \boldsymbol{a}, \boldsymbol{s} \rangle \equiv \xi q \pmod q$. So $\alpha = \xi q = ((p \cdot \langle \boldsymbol{a}, \boldsymbol{s} \rangle + (q-1)/2) \bmod q) - (q-1)/2$. Since $\langle \boldsymbol{a}, \boldsymbol{s} \rangle$ is uniform in $\mathbb{Z}_q$ (for $\boldsymbol{s} \neq 0$), α is uniform in $-(q+1)/2, \cdots, (q-1)/2$ and has mean zero. Hence, so has ξ.

We now prove the second part of our lemma. Let $X = q \cdot \xi$ be a random variable following a discrete uniform distribution on the set of integers $\{(-q+1)/2, \ldots, (q-1)/2\}$. Then, from the characteristic function of X, for any $t \in \mathbb{R}$ we have

$$\mathbb{E}\left[e^{itX}\right] = \frac{e^{-it(q-1)/2} - e^{it(q+1)/2}}{q \cdot (1 - e^{it})} \,. \tag{31}$$

By simple arithmetic, we obtain

$$\mathbb{E}\left[e^{i\xi\gamma}\right] = \mathbb{E}\left[e^{i\gamma q^{-1} X}\right] = \frac{e^{i\gamma/(2q)}\left(e^{-i\gamma/2} - e^{i\gamma/2}\right)}{q\left(1 - e^{i\gamma/q}\right)} = \frac{-\sin(\gamma/2) \cdot 2i}{q\left(e^{-\gamma i/(2q)} - e^{\gamma i/(2q)}\right)}$$

which gives our result. □

In our case, q is an odd prime and different from p. Hence, $\mathbb{E}[e^{i\xi\gamma}]$tends to $\frac{2}{\gamma}\sin(\gamma/2)$ as q grows to infinity. We will be interested in the value $\gamma = 2\pi/p$. Then, for small $p = \{2, 3, 4, 5, \ldots\}$, $\mathbb{E}[e^{i\xi\gamma}]$ is $\{0.6366, 0.8270, 0.9003, 0.9355, \ldots\}$.

5.1 The LWR-solving Algorithm

From the similarity of the LWR and LWE problems, it should not seem surprising that we would use the same sample reduction and back substation phases, but

we need an alternative "hypothesis testing phase" (which we call *solving phase*) to account for the difference in error distributions.

As for LWE, we choose some $a, b \leq k$ such that $ab \leq k$ and we let $k' = k - (a-1)b$. We will view the reduction phase of our algorithm as producing a series of oracles $\mathcal{B}_{\boldsymbol{s},p,\ell}$ for $0 \leq \ell \leq a-1$, where $\mathcal{B}_{\boldsymbol{s},p,0}$ is the original LWR oracle $\Lambda_{\boldsymbol{s},p}$. The final oracle $\mathcal{B}_{\boldsymbol{s},p,a-1}$ produces samples $(\boldsymbol{a}, c)$ where $\boldsymbol{a}$ is non-zero only on the first k' elements.

Solving Phase. We consider the samples from $\mathcal{B}_{\boldsymbol{s},p,a-1}$ as belonging to $\mathbb{Z}_q^{k'} \times \mathbb{Z}_p$. We assume we have m such samples and represent them as a matrix $\boldsymbol{A} \in \mathbb{Z}_q^{m \times k'}$ with rows $\boldsymbol{A}_i$ and a vector $\boldsymbol{c} \in \mathbb{Z}_p^m$. The corresponding block of k' elements of the secret $\boldsymbol{s}$ is denoted $\boldsymbol{s}'$.

Additionally, we assume that each sample $(\boldsymbol{a}^{(j)}, c^{(j)})$ from $\mathcal{B}_{\boldsymbol{s},p,a-1}$ is the sum of 2^{a-1} samples (or their negation) from the LWR oracle. The 'noise' $\langle \boldsymbol{a}^{(j)}, \boldsymbol{s}' \rangle \frac{p}{q} - c^{(j)}$ will then correspond to the sum of 2^{a-1} independent "rounding errors" (or their negation) from the original samples.

For $\theta_u \coloneqq \exp(2\pi i/u)$, we consider the function

$$f_{\mathsf{lwr}}(\boldsymbol{x}) \coloneqq \sum_{j=1}^{m} \mathbb{1}_{\{\boldsymbol{A}_j = \boldsymbol{x}\}} \, \theta_p^{\boldsymbol{c}_j} \ , \quad \forall \boldsymbol{x} \in \mathbb{Z}_q^{k'} \ . \tag{32}$$

The discrete Fourier transform of f_{lwr} is

$$\widehat{f}_{\mathsf{lwr}}(\boldsymbol{\alpha}) \coloneqq \sum_{\boldsymbol{x} \in \mathbb{Z}_q^{k'}} f_{\mathsf{lwr}}(\boldsymbol{x}) \theta_q^{-\langle \boldsymbol{x}, \boldsymbol{\alpha} \rangle} = \sum_{j=1}^{m} \theta_p^{-(\langle \boldsymbol{A}_j, \boldsymbol{\alpha} \rangle \frac{p}{q} - \boldsymbol{c}_j)} \ . \tag{33}$$

In particular, note that

$$\widehat{f}_{\mathsf{lwr}}(\boldsymbol{s}') = \sum_{j=1}^{m} \theta_p^{-(\langle \boldsymbol{s}', \boldsymbol{\alpha} \rangle \frac{p}{q} - \boldsymbol{c}_j)} = \sum_{j=1}^{m} \theta_p^{-(\pm \xi_{j,1} \pm \cdots \pm \xi_{j,2^{a-1}})} \ , \tag{34}$$

where the $\xi_{j,\ell}$ are independent rounding errors from the original LWR samples. Note that it is irrelevant whether the noise has been reduced modulo p, since $\theta_p^{-up} = 1$ for $u \in \mathbb{Z}$.

As for LWE, we can now derive an explicit formula for the number of samples m, which are required to recover $\boldsymbol{s}'$ with high probability.

Lemma 20. *For $q > p \geq 2$, q prime,* $\mathbb{E}\left[\mathrm{Re}(\widehat{f}_{\mathsf{lwr}}(\boldsymbol{s}'))\right] = m \cdot \left(\frac{1}{q} \cdot \frac{\sin(\frac{\pi}{p})}{\sin(\frac{\pi}{pq})}\right)^{2^{a-1}}$.

Proof. Let ξ be the random variable defined in Lemma 19. Since the original rounding errors are independent, using Lemma 19, we may write

$$\mathbb{E}\left[\mathrm{Re}(\widehat{f}_{\mathsf{lwr}}(\boldsymbol{s}'))\right] = m \cdot \mathrm{Re}\left(\mathbb{E}\left[e^{\mp i \xi \frac{2\pi}{p}}\right]^{2^{a-1}}\right) = m \cdot \left(\frac{1}{q} \cdot \frac{\sin(\frac{\pi}{p})}{\sin(\frac{\pi}{pq})}\right)^{2^{a-1}} \ . \tag{35}$$

□

We need also to bound the values of $\widehat{f}$ when not evaluated at $\boldsymbol{s}'$.

Lemma 21. *Let $\boldsymbol{\alpha} \neq \boldsymbol{s}'$. Then*

$$\mathbb{E}\left[\mathrm{Re}(\widehat{f}_{\mathsf{lwr}}(\boldsymbol{\alpha}))\right] \leq m\left(\frac{2}{p} + \frac{1}{p}\cos\left(\frac{\pi}{p}\right)\right)^{2^{a-1}} \leq m\left(\frac{3}{p}\right)^{2^{a-1}} .$$

Proof. Like in the previous lemma, we can write, for $\boldsymbol{a}$ uniformly distributed,

$$\mathbb{E}\left[\mathrm{Re}(\widehat{f}_{\mathsf{lwr}}(\boldsymbol{\alpha}))\right] = m \cdot \mathrm{Re}\left(\mathbb{E}\left[e^{\mp i(2\pi\langle \boldsymbol{a},\boldsymbol{\alpha}\rangle/q - 2\pi \boldsymbol{c}/p)}\right]^{2^{a-1}}\right) . \tag{36}$$

However, unlike in the LWE case, we cannot use the independence of $\boldsymbol{a}$ and the noise to obtain a zero expected value. This occurs because the errors are computed deterministically from the vectors $\boldsymbol{a}$ in LWR. In fact, experiments showed that the error is strongly correlated to $\boldsymbol{a}$ and that the expected value is not zero. Thus, we will instead bound this expected value. To do this, we write

$$\mathbb{E}\left[e^{\mp i(2\pi\langle \boldsymbol{a},\boldsymbol{\alpha}\rangle/q - 2\pi c/p)}\right] = \mathbb{E}\left[\cos\left(\frac{2\pi\langle \boldsymbol{a},\boldsymbol{\alpha}\rangle}{q} - \frac{2\pi c}{p}\right)\right] \pm i\cdot\mathbb{E}\left[\sin\left(-\frac{2\pi\langle \boldsymbol{a},\boldsymbol{\alpha}\rangle}{q} + \frac{2\pi c}{p}\right)\right]$$

and we bound both the sine and the cosine term.

- We first show that the contribution of the sine is zero, i.e., that for $\boldsymbol{\alpha} \neq \boldsymbol{s}'$ fixed,[7]
$$\mathbb{E}\left[\sin\left(2\pi\langle \boldsymbol{a},\boldsymbol{\alpha}\rangle/q - 2\pi c/p\right)\right] = 0 . \tag{37}$$
Let $w(\boldsymbol{a}) \coloneqq \sin\left(2\pi\langle \boldsymbol{a},\boldsymbol{\alpha}\rangle/q - 2\pi\lceil\langle \boldsymbol{a},\boldsymbol{s}'\rangle(p/q)\rfloor/p\right)$. First, note that for $\boldsymbol{a} = \boldsymbol{0}$, $c = 0$. For $\boldsymbol{a} \neq \boldsymbol{0}$, the contribution in the expected value is $w(\boldsymbol{a})$. We have
$$\begin{aligned} w(-\boldsymbol{a}) &= \sin\left(2\pi\langle -\boldsymbol{a},\boldsymbol{\alpha}\rangle/q - 2\pi\lceil\langle -\boldsymbol{a},\boldsymbol{s}'\rangle(p/q)\rfloor/p\right) \\ &= \sin\left(-2\pi\langle \boldsymbol{a},\boldsymbol{\alpha}\rangle/q - 2\pi\lceil -\langle \boldsymbol{a},\boldsymbol{s}'\rangle(p/q)\rfloor/p\right) = -w(\boldsymbol{a}) . \end{aligned}$$
Since q is odd, $-\boldsymbol{a} \neq \boldsymbol{a}$ and, thus, in the expected value, the contribution of any $\boldsymbol{a} \neq 0$ is cancelled. Hence, the result.
- For the cosine, as in Lemma 15, we let $\boldsymbol{y} = \boldsymbol{\alpha} - \boldsymbol{s}' \in \mathbb{Z}_q^{k'}$. We get,
$$\begin{aligned} \cos\left(\frac{2\pi\langle \boldsymbol{a},\boldsymbol{\alpha}\rangle}{q} - \frac{2\pi c}{p}\right) &= \cos\left(\frac{2\pi\langle \boldsymbol{a},\boldsymbol{y}\rangle}{q} + \frac{2\pi(\langle \boldsymbol{a},\boldsymbol{s}'\rangle p/q - c)}{p}\right) \\ &= \cos\left(\frac{2\pi\langle \boldsymbol{a},\boldsymbol{y}\rangle}{q} + \frac{2\pi\xi}{p}\right) , \end{aligned} \tag{38}$$
where $\xi \in [-1/2, 1/2]$ is the rounding error from Lemma 19. We are looking for an upper-bound and, hence, we assume that $\xi \in [-1/2, 1/2]$ will always

[7] This is where the round function instead of the floor function in the definition of LWR becomes handy.

be such that $\cos(2\pi\langle \boldsymbol{a}, \boldsymbol{y}\rangle/q + 2\pi\xi/p)$ is maximized. Figure 1 might help with the reading. We divide the circle into sets of the form

$$\mathcal{S}_\ell \coloneqq \left[\frac{\ell\pi}{p}, \frac{(\ell+1)\pi}{p}\right] \cup \left[\frac{-\ell\pi}{p}, \frac{-(\ell+1)\pi}{p}\right] , \quad \ell \in [0, p-1] .$$

Note that this covers the whole circle. The hashed surface in Figure 1 is such a set.
When $2\pi\langle \boldsymbol{a}, \boldsymbol{y}\rangle/q \in \mathcal{S}_\ell$ for $\ell \neq 0$, we upper-bound (38) by $\cos((\ell-1)\pi/p)$ (the bold line in Figure 1). Indeed, $|2\pi\xi/p| \leq \pi/p$. When $2\pi\langle \boldsymbol{a}, \boldsymbol{y}\rangle/q \in \mathcal{S}_0$, we upper-bound (38) by $\cos(0) = 1$.

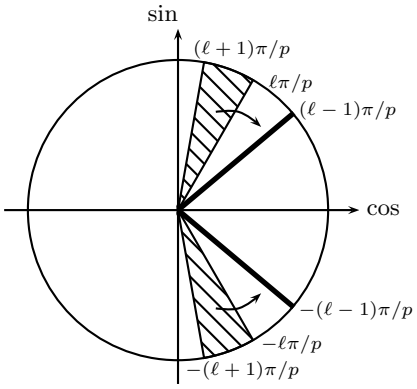

Fig. 1. Figure for the proof of Lemma 21.

Note that $\Pr[2\pi\langle \boldsymbol{a}, \boldsymbol{y}\rangle/q \in \mathcal{S}_\ell]] = 1/p$ since $\langle \boldsymbol{a}, \boldsymbol{y}\rangle$ is uniformly distributed in $\mathbb{Z}_q$ and $p \leq q$. Hence,

$$\begin{aligned}\mathbb{E}\left[\cos\left(2\pi\langle \boldsymbol{a}, \boldsymbol{y}\rangle + 2\pi\xi/p\right)\right] &\leq \frac{1}{p} + \frac{1}{p}\sum_{\ell=1}^{p-1} \cos\left(\frac{(\ell-1)\pi}{p}\right) \\ &= \frac{1}{p} + \frac{1}{p}\cos(0) - \frac{1}{p}\cos\left(\frac{(p-1)\pi}{p}\right) = \frac{2}{p} + \frac{1}{p}\cos\left(\frac{\pi}{p}\right) \leq \frac{3}{p} . \qquad (39)\end{aligned}$$

Plugging the values of the sine and the upper-bound for the cosine in (36) finishes the proof. □

Lemma 22. *When $q > p \geq 4$ and q is prime,* $\arg\max_{\boldsymbol{\alpha}} \mathrm{Re}(\widehat{f}_{\mathsf{lwr}}(\boldsymbol{\alpha})) = \boldsymbol{s}'$ *with probability greater than*

$$1 - q^{k'} \cdot \exp\left(-\frac{m}{8} \cdot \left(\left(\frac{1}{q} \cdot \frac{\sin(\frac{\pi}{p})}{\sin(\frac{\pi}{pq})}\right)^{2^{a-1}} - \left(\frac{3}{p}\right)^{2^{a-1}}\right)^2\right) .$$

Proof. We first want the probability that $\mathrm{Re}(\widehat{f}(\boldsymbol{x})) \geq \mathrm{Re}(\widehat{f}(\boldsymbol{s}'))$ for some fixed vector $\boldsymbol{x} \in \mathbb{Z}_q^{k'}$, $\boldsymbol{x} \neq \boldsymbol{s}'$. Applying the same heuristic argument as for LWE, we consider $X_1, X_2, \ldots, X_m$ to be independent random variables with $X_j = u_j - v_j$, where

$$u_j = \mathrm{Re}\left(\theta_p^{-(\langle \boldsymbol{A}_j, \boldsymbol{s}' \rangle \frac{p}{q} - \boldsymbol{c}_j)}\right) \qquad \text{and} \qquad v_j = \mathrm{Re}\left(\theta_p^{-(\langle \boldsymbol{A}_j, \boldsymbol{x} \rangle \frac{p}{q} - \boldsymbol{c}_j)}\right) . \tag{40}$$

Note that $X_j \in [-2, 2]$ for all j. Furthermore, let $X = \sum_{j=1}^{m} X_j$. Using Lemmas 20 and 21, we get that

$$\mathbb{E}[X] \geq m \cdot \left(\left(\frac{1}{q} \cdot \frac{\sin(\frac{\pi}{p})}{\sin(\frac{\pi}{pq})} \right)^{2^{a-1}} - \left(\frac{3}{p} \right)^{2^{a-1}} \right) \geq 0 . \tag{41}$$

We will again bound the probability that $X \leq 0$ using Hoeffding's inequality. Let $t = \mathbb{E}[X] > 0$. Then,

$$\begin{aligned} \Pr[X \leq 0] = \Pr\left[(X - \mathbb{E}[X]) \leq -\mathbb{E}[X] \right] &\leq \exp\left(\frac{-2(\mathbb{E}[X])^2}{16m} \right) \\ &\leq \exp\left(-\frac{m}{8} \cdot \left(\left(\frac{1}{q} \cdot \frac{\sin(\frac{\pi}{p})}{\sin(\frac{\pi}{pq})} \right)^{2^{a-1}} - \left(\frac{3}{p} \right)^{2^{a-1}} \right)^2 \right) . \end{aligned} \tag{42}$$

The final result follows by applying a union bound over all possible values of $\boldsymbol{x}$. □

As for LWE, we may now deduce the number m of reduced samples that are required to recover a block $\boldsymbol{s}'$.

Theorem 23. *Let k and $q > p \geq 4$ be positive integers, q prime, and $\Lambda_{\boldsymbol{s},p}$ be an LWR oracle, where $\boldsymbol{s} \in \mathbb{Z}_q^k$. Let $a \in \mathbb{Z}$ with $1 \leq a \leq k$, let b be such that $ab \leq k$, and let $k' = k - (a-1)b$. Let $\mathcal{B}_{\boldsymbol{s},p,a-1}$ be the oracle returning samples $(\boldsymbol{a}_i, c_i)$ where the $\boldsymbol{a}_i$ are zero for all but the first k' elements. Denote the vector consisting of the first k' elements of $\boldsymbol{s}$ as $\boldsymbol{s}'$. Fix an $\epsilon \in (0, 1)$. Then, the number of samples m from $\mathcal{B}_{\boldsymbol{s},p,a-1}$, which are required such that we fail to recover the secret block $\boldsymbol{s}'$ with probability at most ϵ satisfies*

$$m^{LWR} \geq 8 \cdot k' \cdot \log\left(\frac{q}{\epsilon}\right) \cdot \left(\left(\frac{1}{q} \cdot \frac{\sin(\frac{\pi}{p})}{\sin(\frac{\pi}{pq})} \right)^{2^{a-1}} - \left(\frac{3}{p} \right)^{2^{a-1}} \right)^{-2} .$$

Furthermore, recovering $\boldsymbol{s}'$ in the solving phase (the FFT phase) requires $2m^{LWR} + C_{\mathsf{FFT}} \cdot k' \cdot q^{k'} \cdot \log q$ operations in $\mathbb{C}$, as well as storage for $q^{k'}$ complex numbers.

We now summarize the complexity of our algorithm in the following theorem (the proof of which is analogous to the proof of Theorem 17).

Theorem 24 (Complexity of SEARCH-LWR). *Let $k, q > p \geq 4$ be positive integers, q prime, and $\Pi_{\boldsymbol{s},\chi}$ be an LWE oracle, where $\boldsymbol{s} \in \mathbb{Z}_q^k$. Let $a, b \in \mathbb{Z}_q$ be such that $a \cdot b = k$. For $0 \leq j \leq a-1$, let*

$$m_{j,\epsilon}^{LWR} \coloneqq 8 \cdot b \cdot \log\left(\frac{q}{\epsilon}\right) \cdot \left(\left(\frac{1}{q} \cdot \frac{\sin(\frac{\pi}{p})}{\sin(\frac{\pi}{pq})}\right)^{2^{a-1-j}} - \left(\frac{3}{p}\right)^{2^{a-1-j}}\right)^{-2} .$$

Let $0 < \epsilon < 1$ be a targeted success rate and define $\epsilon' \coloneqq (1-\epsilon)/a$. The (time, memory and query) complexities to recover the LWR secret $\boldsymbol{s}$ with probability ϵ are the same as in Theorem 17 where we replace $m_{j,\epsilon}^{LWE}$ by $m_{j,\epsilon}^{LWR}$.

5.2 Results

The current hardness results for LWR require either a parameter q exponential in k or a bound m on the number of oracle samples that an adversary may query. It is an open problem ([3]) to assess the hardness of LWR with polynomial parameters when the adversary has no sample limit. In such a case, for $a = O(\log k)$ and $b = \lceil k/a \rceil$, our algorithm would solve LWR in time $2^{O(k)}$, as for LWE.

However, the bound on the number of oracle samples in Theorem 7 is much lower than the amount of samples required by our algorithm. Using an idea from Lyubashevsky [37] we can generate additional samples with higher noise (see Theorem 18). Yet, even this method requires at least $k^{1+\epsilon}$ samples for $\epsilon \geq (\log q + 1)/\log k$, which is incompatible with the constraints of Theorem 7, for a q polynomial in k.

in [3, Corollary 4.2], two types of parameters are proposed: parameters minimizing the Modulus/Error ratio (a) and parameters maximizing efficiency (b). For completeness, we show in Table 2 the complexity of our algorithm applied to these parameters. More precisely, we took for the underlying LWE problem Regev's parameters and *ignored the constrains on the number of samples*. For the type (a) parameters, we took

$$\sigma = \frac{k^2}{\sqrt{k}\log^2(k)\sqrt{2\pi}} \qquad q = \mathsf{nextprime}(\lceil (2\sigma k)^3 \rceil) \qquad p = \mathsf{nextprime}(\lceil \sqrt[3]{q} \rceil)$$

and for the type (b) parameters

$$\sigma = \frac{k^2}{\sqrt{k}\log^2(k)\sqrt{2\pi}} \qquad p = 13 \qquad q = \mathsf{nextprime}(\lceil 2\sigma k p \rceil) .$$

Table 2 shows that the parameters proposed in [3] seem secure even if we remove the constrain on the number of samples as the complexities are still quite high.

6 Conclusion

To summarize, we propose an algorithm which is currently the best algorithm for solving the LWE problem. Our algorithm uses Fourier transforms and we

k	q	p	a	$\log(\#\mathbb{Z}_q)$	$\log(m)$	type
64	383 056 211	733	23	92.20	82.80	(a)
80	1 492 443 083	1 151	25	110.91	101.11	(a)
96	$\approx 2^{32}$	1 663	26	132.26	122.15	(a)
112	$\approx 2^{33}$	2 287	28	148.08	137.68	(a)
128	$\approx 2^{34}$	3 023	29	167.52	156.87	(a)
64	9 461	13	12	81.61	72.90	(b)
80	14 867	13	12	103.89	94.86	(b)
96	21 611	13	12	126.97	117.66	(b)
112	29 717	13	13	140.21	130.60	(b)
128	39 241	13	13	162.63	152.84	(b)

Table 2. Worst case cost (in operations over $\mathbb{Z}_q$) of solving Search-LWR for various parameters for the Regev cryptosystem [43] when $\epsilon = 0.99$ according to Theorem 24. We provide also the value of a that minimizes the complexity, the number of queries (m) according to (27).

propose a careful analysis of the rounded Gaussian distribution which can be of independent interest. In particular, we study its variance and the expected value of its cosine. We also adapt our algorithm to the LWR problem when q is prime. This algorithm is the first LWR-solving algorithm.

Further work includes the study of the Ring variants of LWE and LWR [38,8] and the study of variants of LWE, e.g., when the secret follows a non-uniform distribution (like in [2]) or when the noise follows a non Gaussian distribution. It would also be interesting to see if our LWR algorithm can be extended for q non prime.

Acknowledgments. We are grateful to Dimitar Jetchev and Adeline Langlois for helpful discussions and pointers. We thank the Eurocrypt 2015 reviewers for their fruitful comments.

References

1. Albrecht, M.R., Cid, C., Faugère, J.C., Fitzpatrick, R., Perret, L.: On the complexity of the BKW algorithm on LWE. Designs, Codes and Cryptography pp. 1–30 (2013)
2. Albrecht, M.R., Faugère, J.C., Fitzpatrick, R., Perret, L.: Lazy Modulus Switching for the BKW Algorithm on LWE. In: Krawczyk, H. (ed.) Public Key Cryptography. Lecture Notes in Computer Science, vol. 8383, pp. 429–445. Springer (2014)
3. Alwen, J., Krenn, S., Pietrzak, K., Wichs, D.: Learning with rounding, revisited - new reduction, properties and applications. In: Canetti and Garay [19], pp. 57–74
4. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems. In: Halevi, S. (ed.) Advances in Cryptology - CRYPTO 2009. Lecture Notes in Computer Science, vol. 5677, pp. 595–618. Springer (2009)