Verifying Solutions to LWE with Implications for Concrete Security

Palash Sarkar and Subhadip Singha
Applied Statistics Unit
Indian Statistical Institute
203, B.T.Road, Kolkata, India - 700108.
{palash, subha_r}@isical.ac.in

June 19, 2019

Abstract

A key step in Regev's (2009) reduction of the Discrete Gaussian Sampling (DGS) problem to that of solving the Learning With Errors (LWE) problem is a statistical test required for verifying possible solutions to the LWE problem. Regev showed that asymptotically, the success probability of this test is exponentially close to 1. In this work, we work out the concrete details of the success probability and its effect on the tightness gap of the reduction. For lattice dimensions from 2 to 187149, the tightness gap is determined entirely by the inverse of the success probability. The actual value of the tightness gap for such lattice dimensions is *huge* showing that the reduction cannot be used for choosing parameters for practical cryptosystems.

Keywords: lattices, learning with errors, discrete gaussian sampling, statistical test, concrete analysis.

1 Introduction

In a seminal work, Regev [10] introduced the learning with errors (LWE) problem and highlighted its role in lattice-based cryptography. The major achievement of the work was to show a reduction from a worst case lattice problem to the breaking of a cryptosystem. This worst-case to average-case reduction has been later claimed to be a major theoretical advantage of cryptosystems based on lattices. The entire analysis in [10] was done in an asymptotic setting where the lattice dimension n is allowed to go to infinity.

Suppose $\mathcal{A}$ is an algorithm which given access to an oracle $\mathcal{O}$ solves a problem $\mathcal{P}$ in time T with success probability P_S. Further, suppose $\mathcal{O}$ takes time T' and has success probability P'_S. Then the tightness gap of the algorithm $\mathcal{A}$ is $(T \cdot P'_S)/(T' \cdot P_S)$. The reduction is said to be tight if the tightness gap is 1 (or small) and is said to be loose if the tightness gap is 'large'.

A later work [6] performed a concrete analysis of the reduction in [10] to determine whether it can be used to choose parameters for practical cryptosystems. This required determining the tightness gap of the reduction in concrete terms as a function of n. It turned out that the tightness gap can indeed be very large casting doubt on the practical usefulness of the reduction in [10]. For example, for $n = 1024$, it was argued that the tightness gap is about 2^{504} and so the worst-case to average-case reduction in [10] cannot be used to argue about the security of cryptosystems with lattice dimension $n = 1024$.

The reduction in [10] is a cascade of three smaller reductions. The first reduction is from the Smallest Independent Vector Problem (SIVP) to the problem of Discrete Gaussing Sampling (DGS). The second reduction is from DGS to (search) LWE while the third reduction is from search-LWE to average case decisional LWE (DLWE$_{ac}$). There is a further reduction from DLWE$_{ac}$ to that of breaking the cryptosystem. We ignore this reduction, since this depends on the actual cryptosystem.

The second reduction, i.e., the one from DGS to LWE is the main contribution of [10]. A key step in this reduction consists of verifying solutions to LWE. This verification is done using a statistical test. It has been argued in [10] that asymptotically the success probability of the statistical test is exponentially close to 1. The statistical test is used many times in the entire reduction and the success probability of the statistical test determines the overall success probability of the complete reduction. We take a close look at the success probability of the statistical test. Using the standard Hoeffding inequality, we determine an upper bound on the error of the statistical test. This in turn leads to an expression for the success probability of the test and then to the success of the entire reduction.

Considering concrete values of the lattice dimension provides very surprising results. For n from 2 to 187149, the overall tightness gap is determined almost entirely by the success probability of the statistical test and the number of oracle calls has a negligible effect on the tightness gap. The values of $1/P_S$ for $n = 2$ and $n = 187149$ are about 2^{2198} and $2^{2^{54}}$ respectively. Surprisingly, for $n = 187150$, the value of $1/P_S$ is essentially 1 and remains so for larger values of n. For $n \geq 187150$, the tightness gap is given by the number of oracles calls.

The effect of the success probability of the statistical test on the tightness gap of the reduction in [10] was not considered in the earlier concrete analysis [6]. Our analysis shows that for values of n of practical interest, the tightness gap is determined almost entirely by the success probability of the statistical test. While the concrete analysis [6] had pointed out that the tightness gap of the reduction in [10] can be quite large, it still held out the possibility that by choosing larger values of n, the reduction in [10] can be used to obtain concrete security guarantees. Our analysis, on the other hand, shows that the reduction in [10] cannot be used to obtain any meaningful practical security guarantee for values of n up to 187149.

The ring-LWE (RLWE) problem was later considered in [9]. This work showed a worst-case to average-case reduction which is analogous to the reduction in [10]. The reduction in [9] refers to the verification lemma used by Regev [10]. So, our concrete analysis of the success probability of the statistical test to verify LWE solutions should also apply to the reduction in [9].

The importance of LWE in the context of lattice based cryptography is underscored by the fact that a number of submissions made to the ongoing NIST process for selecting a new public key standard base their security on the LWE problem and several of its variants. LWE based proposals which are in the second round of the NIST process are Frodo [2], Kyber [3], LAC [8], NewHope [1], Round5 [4] and Saber [7]. A recent work by Bernstein [5] performs a comparative study of the provable security of these and other lattice based proposals. While commenting on the tightness of reduction, Bernstein [5] comments that "the loss of tightness is gigantic" and credits [6] for pointing this out. Our analysis of the reduction in [10] goes beyond that of [6] and shows that for parameters of practical interest it is in fact meaningless.

2 Preliminaries

A full rank lattice L in $\mathbb{R}^n$ is the set of all integer linear combinations of n linearly independent vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ in $\mathbb{R}^n$. Following Regev [10], L will also denote the $n \times n$ matrix whose columns are $\mathbf{v}_1, \ldots, \mathbf{v}_n$. So, given $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{Z}^n$, $L\mathbf{a}$ denotes the lattice point $\mathbf{v} = a_1\mathbf{v}_1 + \cdots + a_n\mathbf{v}_n$ and $L^{-1}(\mathbf{v})$ denotes the integer coefficient vector $\mathbf{a}$ corresponding to $\mathbf{v}$.

The length of a vector is its Euclidean norm. For $i \in \{1, \ldots, n\}$, $\lambda_i(L)$ is the least real number r such that L has i linearly independent vectors with the longest having length r. The dual of a lattice L is denoted as L^* and is defined to be the set of all vectors $\mathbf{y} \in \mathbb{R}^n$ such that $\langle \mathbf{x}, \mathbf{y} \rangle \in \mathbb{Z}$ for all $x \in L$.

The normal distribution with mean μ and standard deviation σ will be denoted as $\mathcal{N}(\mu, \sigma)$. For $\alpha \in (0, 1)$, Ψ_α is the probability distribution obtained by sampling from $\mathcal{N}(0, \alpha/\sqrt{2\pi})$ and reducing the result modulo 1.

Let $p \geq 2$ be an integer. Let χ be a probability distribution on $\mathbb{Z}_p$. Let n be a positive integer and fix $\mathbf{s} \in \mathbb{Z}_p^n$. The distribution $A_{\mathbf{s},\chi}$ on $\mathbb{Z}_n^p \times \mathbb{Z}_p$ is defined as follows. Choose $\mathbf{a}$ uniformly at random from $\mathbb{Z}_p^n$; e from $\mathbb{Z}$ following χ and output $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$. Let ϕ be a probability density function on $\mathbb{T} = (0, 1)$. The distribution $A_{\mathbf{s},\phi}$

is defined as follows. Choose $\mathbf{a}$ uniformly at random from $\mathbb{Z}_p^n$; e from $\mathbb{T}$ following ϕ and output $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s}\rangle/p + e)$, where the addition is performed modulo 1.

Fix a positive integer n and an integer $p \geq 2$. The learning with errors problem $\mathrm{LWE}_{p,\chi}$ is the following. For any $\mathbf{s} \in \mathbb{Z}_p^n$, given samples from $A_{s,\chi}$, it is required to output $\mathbf{s}$. Similarly, for a probability density function ϕ on $\mathbb{T}$, the $\mathrm{LWE}_{p,\phi}$ problem is the following. For any $\mathbf{s} \in \mathbb{Z}_p^n$, given samples from $A_{s,\phi}$, it is required to output $\mathbf{s}$.

For $\mathbf{x} \in \mathbb{R}^n$ and $s > 0$, define $\rho_s(\mathbf{x}) = \exp\left(-\pi ||\mathbf{x}||^2/s^2\right)$. For a lattice L, define $\rho_s(L) = \sum_{\mathbf{x}\in L} \rho_s(\mathbf{x})$. The discrete Gaussian distribution $D_{L,s}$ on a lattice L assigns to a vector $\mathbf{v} \in L$ the probability $D_{L,s}(\mathbf{v}) = \rho_s(\mathbf{v})/\rho_s(L)$. Given a lattice L and a real number r, the discrete Gaussian sampling problem $\mathrm{DGS}_{L,r}$ is to obtain a sample from $D_{L,r}$.

For a lattice L and a real number $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(L)$ is the smallest s such that $\rho_{1/s}(L^* \setminus \{0\}) \leq \epsilon$.

3 From DGS to LWE

The following is a restatement of Theorem 3.1 of [10] which is the main result of [10].

Theorem 1. *Let L be a lattice of dimension n, $p \geq 2$ be an integer and $\alpha \in (0,1)$ be a real number. Given an oracle for $\mathsf{LWE}_{p,\Psi_\alpha}$, it is possible to sample from $D_{L,r}$ where $r \geq \sqrt{2n} \cdot \eta_\epsilon(L)/\alpha$, $\alpha p > 2\sqrt{n}$.*

For $r \geq \sqrt{2n} \cdot \eta_\epsilon(L)/\alpha$, define $r_i = r \cdot (\alpha p/\sqrt{n})^i$ for $i = 1, \ldots, 3n$.

The proof of the theorem is provided in [10] as a sequence of nested oracle calls. In the following, we rewrite the oracle calls and the other computations required for the proof in [10] in an algorithmic form. The required subroutines and data structures are as follows. The quantity c is a constant such that the LWE oracle is provided with n^c samples.

$\mathsf{solveLWE}_{p,\Psi_\alpha}(\mathcal{I})$**:** This is the oracle to solve $\mathsf{LWE}_{p,\Psi_\alpha}$. The list $\mathcal{I}$ consists of n^c samples from $A_{\mathbf{s},\Psi_\beta}$ for some $0 < \beta \leq \alpha$. Note that the oracle is guaranteed to work correctly if $\beta = \alpha$, otherwise it might return an incorrect result.

$\mathsf{verifyLWE}(\mathbf{s}', \mathcal{I})$**:** The input $\mathcal{I}$ contains n^c samples from $A_{\mathbf{s},\Psi_\beta}$. This algorithm returns true if $\mathbf{s} = \mathbf{s}'$, otherwise it returns false.

$\mathsf{solveCVP}^{(p)}(L^*, \mathcal{L}, \mathbf{z})$**:** Here L^* is the dual lattice of L; $\mathcal{L}$ contains n^c samples from D_{L,r_i} for some $i \in \{1, \ldots, 3n\}$; $\mathbf{z}$ is within distance $\lambda_1(L)/2$ of L^*. Returns the coefficient vector modulo p of the vector in L^* which is closest to $\mathbf{z}$.

$\mathsf{solveCVP}(L^*, \mathcal{L}, \mathbf{z})$**:** The inputs $L^*, \mathcal{L}$ and $\mathbf{z}$ are as in the case of $\mathsf{solveCVP}^{(p)}$. Returns a point of L^* which is closest to $\mathbf{z}$.

$\mathsf{quantumSample}()$**:** Uses $\mathsf{solveCVP}(L^*, \mathcal{L}, \cdot)$ as an oracle and some quantum computation to return a sample from $D_{L,r_{i-1}}$. The list $\mathcal{L}$ contains n^c samples from D_{L,r_i}.

$\mathsf{solveDGS}(p, \alpha, r)$**:** Uses the oracle $\mathsf{solveLWE}_{p,\Psi_\alpha}(\cdot)$ to return a sample from $D_{L,r}$ where $r \geq \sqrt{2n} \cdot \eta_\epsilon(L)/\alpha$. Note that the description of the algorithm $\mathsf{solveDGS}$ provides the proof of Theorem 1.

In the algorithm descriptions, we will make use of the following two sub-routines mentioned below. We will not be needing the details of these procedures and so we do not describe these details. They can be found in [10].

1. $\mathsf{bootstrap}(L, r)$**:** Here L is a lattice and $r > \sqrt{2n} \cdot \eta_\epsilon(L)/\alpha$. Returns a list $\mathcal{L}$ containing n^c independent samples from $D_{L,r_{3n}}$ where $r_{3n} = r \cdot ((\alpha p)/(\sqrt{n}))^{3n}$.

2. $\mathsf{reconstruct}(x)$**:** This is used in $\mathsf{solveCVP}$ to reconstruct the closest vector by first applying a nearest neighbour algorithm and then retracing through the results returned by the repeated calls to $\mathsf{solveCVP}^{(p)}$.

Algorithm 1 Algorithm to solve DGS using an LWE oracle.

```
function solveDGS($(p, \alpha)$)
    $\mathcal{L} \leftarrow$ bootstrap$(L, r)$;
    for $i \leftarrow 3n$ down to $1$ do
        $\mathcal{L}' \leftarrow \{\}$;
        for $j \leftarrow 1$ to $n^c$ do
            $\mathbf{y} \leftarrow$ quantumSample() (using solveCVP$(L^*, \mathcal{L}, \cdot)$ as an oracle);
            $\mathcal{L}' \leftarrow \mathcal{L}' \cup \{\mathbf{y}\}$;
        end for
        $\mathcal{L} \leftarrow \mathcal{L}'$; $r_{i-1} = r_i \cdot (\sqrt{n})/(\alpha p)$;
    end for
    return one element from $\mathcal{L}$.
end function.
```

Algorithm 2 Algorithm to solve CVP.

```
function solveCVP($(L^*, \mathcal{L}, z)$)
    $\mathbf{z}_1 \leftarrow \mathbf{z}$;
    for $k \leftarrow 1$ to $n$ do
        $\mathbf{a}_k \leftarrow$ solveCVP$^{(p)}(L^*, \mathcal{L}, \mathbf{z}_k)$;
        $\mathbf{z}_{k+1} \leftarrow (\mathbf{z}_k - L^* \mathbf{a}_k)/p$;
    end for
    $\mathbf{s} \leftarrow$ reconstruct$(\mathbf{z}_{n+1})$;
    return $\mathbf{s}$.
end function.
```

4 Concrete Analysis

The number of times the oracle solveLWE is called is determined by the following factors.

1. The loop in solveDGS has $3n$ iterations. In the i-th iteration n^c samples of D_{L,r_i} are used to generate n^c samples of $D_{L,r_{i-1}}$. Generating each sample of $D_{L,r_{i-1}}$ requires a call to quantumSample which in turn generates a call to solveCVP. So, the sub-routines quantumSample and solveCVP are both called a total of $3n \cdot n^c = 3n^{c+1}$ times.

2. The loop in solveCVP has n iterations and in each iteration, a call to solveCVP$^{(p)}$ is made. So, each call to solveCVP generates n calls to solveCVP$^{(p)}$.

3. In solveCVP$^{(p)}$, the set Z contains about n^{2c} values. So, the loop from Steps 9 to 21 makes about n^{2c+1} calls to solveLWE and to verifyLWE. So, each call to solveCVP$^{(p)}$ generates n^{2c+1} calls to solveLWE and to verifyLWE.

Proposition 2. *Algorithm* solveDGS *has the following properties.*

1. *The* solveLWE *oracle is called* $T = 3n^{3c+3}$ *times.*

2. *Algorithm* verifyLWE *is also called* T *times.*

3. *A total of* $3n^{c+1}$ *quantum computations are required.*

Algorithm 3 Algorithm to solve CVP$^{(p)}$.

```
function solveCVP$^{(p)}$($L^*, \mathcal{L}, \mathbf{z}$)
    $Z \leftarrow$ set of all integer multiples of $n^{-2c}\alpha^2$ in the range $(0, \alpha^2]$;
    $\mathcal{I} \leftarrow \{\}$;
    for $\mathbf{v}$ in $\mathcal{L}$ do
        $\mathbf{a} \leftarrow L^{-1}\mathbf{v} \bmod p$;
        $e \overset{\$}{\leftarrow} \mathcal{N}(0, \alpha/(2\sqrt{\pi}))$;
        $\mathcal{I} \leftarrow \mathcal{I} \cup \{(\mathbf{a}, \langle \mathbf{z}, \mathbf{v} \rangle/p + e \bmod 1)\}$;
    end for
    for $\gamma$ in $Z$ do
        $\mathcal{I}' \leftarrow \{\}$;
        for $i \leftarrow 1$ to $n$ do
            for $(\mathbf{a}, e) \in \mathcal{I}$ do
                $\varepsilon \overset{\$}{\leftarrow} \Psi_{\sqrt{\gamma}}$;
                $\mathcal{I}' \leftarrow \mathcal{I}' \cup \{(\mathbf{a}, e + \varepsilon)\}$;
            end for
            $\mathbf{s}' \leftarrow$ solveLWE$_{p,\Psi_\alpha}(\mathcal{I}')$;
            if verifyLWE($s', \mathcal{I}'$) returns true then
                return $\mathbf{s}'$;
            end if
        end for
    end for
end function.
```

Algorithm 4 Algorithm to verify an LWE solution.

```
 1: function verifyLWE($\mathbf{s}', \mathcal{I}'$)
 2:     $m \leftarrow n$;
 3:     Choose $m$ pairs $(\mathbf{a}_1, x_1), \ldots, (\mathbf{a}_m, x_m)$ from $\mathcal{I}'$;
 4:     $w \leftarrow 0$;
 5:     for $i \leftarrow 1$ to $m$ do
 6:         $y_i \leftarrow x_i - \langle \mathbf{a}_i, \mathbf{s}' \rangle/p$;
 7:         $w \leftarrow w + \cos(2\pi y_i)$;
 8:     end for
 9:     $z \leftarrow w/m$; $t \leftarrow 0.02$;
10:     if $z > t$ then
11:         return true
12:     else
13:         return false
14:     end if
15: end function.
```

Remarks:

1. Regev [10] has shown that each quantum computation is on a state of $n \log R$ qubits where $R \geq 2^{3n}\lambda_n(L^*)$ is an integer.

2. For verifyLWE, the settings of $m = n$ in Step 2 and $t = 0.02$ in Step 9 are the values given by Regev [10].

Note that the set $\mathcal{I}'$ has cardinality n^c and so m is at most n^c.

Algorithm verifyLWE is essentially a test of hypothesis. In verifyLWE, the pairs in $\mathcal{I}'$ are of the form $(a, \langle a, s\rangle/p + e)$ where e follows Ψ_β. The test statistic is the variable z. Let ξ_0 be the distribution of z when $s = s'$ and let μ_0 be the corresponding mean of z; let ξ_1 be the distribution of z when $s \neq s'$ and let μ_1 be the corresponding mean of z. The following have been proved by Regev [10].

- $\xi_0 = \Psi_\alpha$ so that $\mu_0 = \exp(-\pi\alpha^2) \geq 0.04$ for $\alpha < 1$. Note that $\mu_0 > t = 0.02$.
- $\mu_1 = 0$.

The computation performed by verifyLWE is a test of hypothesis between $H_0 : s = s'$ versus $H_1 : s \neq s'$.

Two types of errors are to be considered.

$$\begin{aligned} e_0 &= \Pr[\text{Type-1 error}] = \Pr[\text{reject } H_0 \text{ when it is true}] = \Pr_{z\sim\xi_0}[z \leq t]; & (1)\\ e_1 &= \Pr[\text{Type-2 error}] = \Pr[\text{accept } H_0 \text{ when it is false}] = \Pr_{z\sim\xi_1}[z > t]. & (2)\end{aligned}$$

A Type-1 error will result in the correct value of s' being rejected and so the entire reduction will not succeed. A Type-2 error will result in an incorrect value of s' being accepted. This incorrect value of s' will be passed on to verifyCVP$^{(p)}$ and then on to verifyCVP resulting in an incorrect solution to the CVP problem. So, again, the entire reduction will fail. So, it is required to ensure that both Type-1 and Type-2 errors are small.

For $i = 1, \ldots, m$, let $v_i = \cos(2\pi y_i)$. Then $v_1, \ldots, v_m$ take values in the interval $[-1, 1]$. Applying the Hoeffding inequality (see Appendix A) to $v_1, \ldots, v_m$ and $z = (v_1 + \cdots + v_m)/m$, provides the following upper bounds on e_0 and e_1.

$$\begin{aligned} e_0 &= \Pr_{z\sim\xi_0}[z \leq t] = \Pr_{z\sim\xi_0}[z - \mu_0 \leq -(\mu_0 - t)] \leq \exp\left(-m(\mu_0 - t)^2/2\right); & (3)\\ e_1 &= \Pr_{z\sim\xi_1}[z > t] = \Pr_{z\sim\xi_1}[z - \mu_1 > t - \mu_1] \leq \exp\left(-mt^2/2\right). & (4)\end{aligned}$$

If $s' = s$, then the probability that verifyLWE makes an error is at most e_0; if $s' \neq s$, then the probability that verifyLWE makes an error is at most e_1. So, the probability that verifyLWE makes an error is at most $\max(e_0, e_1)$ and so the probability that verifyLWE is successful is at least $(1 - \max(e_0, e_1)) = (1 - \max(\exp\left(-m(\mu_0 - t)^2/2\right), \exp\left(-mt^2/2\right)))$.

Proposition 2 shows that verifyLWE is called a total of $3n^{3c+3}$ times by solveDGS. The probability that all of these calls are successful is given by

$$P_S = \left(1 - \max(\exp\left(-m(\mu_0 - t)^2/2\right), \exp\left(-mt^2/2\right))\right)^{3n^{3c+3}}. \quad (5)$$

Again from Proposition 2, the number of calls to solveLWE made by solveDGS is $T = 3n^{3c+3}$ and so the tightness gap of the reduction from DGS to LWE is

$$G = T/P_S = 3n^{3c+3} \cdot \left(1 - \max(\exp\left(-m(\mu_0 - t)^2/2\right), \exp\left(-mt^2/2\right))\right)^{-3n^{3c+3}}. \quad (6)$$

For all $\alpha < 1$, we have $\mu_0 > 0.04$. The value of t in Step 9 of verifyLWE is set to 0.02 and so $t < \mu_0 - t$ for all $\alpha < 1$. Further, Step 2 sets $m = n$ as has been done by Regev. So, for $m = n$ and $t = 0.02$, (6) simplifies to the following.

$$G = T/P_S = 3n^{3c+3} \cdot (1 - \exp(-n/5000)))^{-3n^{3c+3}}. \quad (7)$$

Setting $c = 1$, we have evaluated T, P_S and G for various values of n.

1. The dependence of $1/P_S$ on n is quite interesting. The value of $1/P_S$ is about 2^{2198} for $n = 2$ and quickly becomes very high (SAGE [11] was unable to compute the values of $1/P_S$ for a large range of values of n); for $n = 187149$, the value of $1/P_S$ is about $2^{2^{54}}$. Then it sharply decreases to almost 1 for $n = 187150$.

2. The value of T increases with increasing n. Further, $T \approx 2^{106}$ for both $n = 187149$ and $n = 187150$.

3. For $n \leq 187149$, G is determined primarily by $1/P_S$ and the tightness gap G varies from about $7 \cdot 2^{2198}$ for $n = 2$ to about $2^{106} \cdot 2^{2^{54}}$ for $n = 187149$. This gap is **huge**. For $n = 187150$, the tightness gap G is about 2^{106} which is also quite large.

The parameter n is the dimension of the underlying lattice. The above concrete security analysis shows that for lattices with dimension at most 187149, the tightness gap of the reduction is determined primarily by $1/P_S$ and this value is huge. For lattices of dimension at least 187150, the tightness gap of the reduction is determined by T and $1/P_S$ no longer plays a role. If one wishes to choose values of parameters based on the Regev's theorem, then one has to work with lattice dimension at least 187150. The practicality of such a construction is not clear. Further, it is to be noted that the tightness gap for $n \geq 187150$ is itself quite large.

From Proposition 2 and the remark following it, for $c = 1$ the number of quantum computations required is $3n^2$ where each computation is on at least $(3n^2 + n \log \lambda_n(L^*))$ bits. The cost of quantum computation increases quadratically with n.

Remarks:

1. In Step 2 of verifyLWE, the choice of the number of samples m to be equal to n has been taken following Regev [10]. As mentioned earlier, $m \leq n^c$ and it is possible to take $m = n^c$. The determination of the concrete value of the tightness gap done above is for $c = 1$.

2. In Step 9 of verifyLWE, the choice of the value of the threshold t to be equal to 0.02 has been taken following Regev [10]. The statistical test performed by verifyLWE is essentially a test for the means μ_0 and $\mu_1 = 0$ of the distributions ξ_0 and ξ_1 respectively. A natural value of the rejection threshold t is the choice $\mu_0/2 = \exp(-\pi\alpha^2)/2$. Then the expression for the tightness gap given by (6) depends on the value of α. The resulting tightness gap increases with increase in the value of α. A higher value of α makes the LWE problem more difficult but, also results in a worse tightness gap in the reduction from DGS to LWE. Regev used $t = 0.02$ which results in the highest possible value of the tightness gap.

4.1 Comparison to Previous Concrete Analysis

The complete reduction by Regev [10] is from worst case SIVP to average case decisional LWE problem (DLWE$_{ac}$). This reduction consists of three parts.

- SIVP to DGS with tightness gap $2n^3$ [6].
- DGS to LWE with tightness gap G given by (7). In the previous concrete analysis [6], the tightness gap for the reduction from DGS to LWE was obtained as $3n^{c+3}$. The reason is that the loop over γ from Steps 9 to 21 in Algorithm solveCVP$^{(p)}$ was missed.
- LWE to DLWE$_{ac}$ with tightness gap $np \cdot n^{d_1+2d_2+2}$ [6] for positive integers d_1 and d_2.

Our work does not change the tightness gap of the first and the third reductions. The main part of the entire reduction is the second reduction. For this case, we have incorporated into the tightness gap the success probability of the statistical test required by the algorithm for verifying LWE solutions. This was not considered in [6].

In [6], it was argued that for $n = 1024$, it is reasonable to take $d_1 = d_2 = 12.8$ and $c = 1$ resulting in the tightness gap of the reduction from SIVP to DLWE$_{ac}$ to be about 2^{504}. Considering the tightness gap of the reduction from DGS to LWE to be given by (7) for $n = 1024$, SAGE returns the value of $1/P_S$ to be '`+infinity`'. In fact, for $2 \leq n \leq 187149$, the tightness gap is determined solely by $1/P_S$. For this range of n, the number of oracle calls T is insignificant in comparison to $1/P_S$.

5 Conclusion

Our work highlights the role of success probability of a statistical test in determining the tightness gap of the reduction from DGS to LWE. More generally, our work shows that whenever a statistical test is used in an analysis, it is important to determine the success probability of the test for concrete values of the relevant parameters.

References

[1] Erdem Alkim, Roberto Avanzi, Joppe Bos, Leo Ducas, Antonio de la Piedra, Thomas Poppelmann, Peter Schwabe, Douglas Stebila, Martin R. Albrecht, Emmanuela Orsini, Valery Osheter, Kenneth G. Paterson, Guy Peer, and Nigel P. Smart. NewHope: algorithm specifications and supporting documentation. `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`, 2019.

[2] Erdem Alkim, Joppe Bos, Leo Ducas, Patrick Longa, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, and Douglas Stebila. FrodoKEM: Learning With Errors key encapsulation. `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`, 2019.

[3] Roberto Avanzi, Joppe Bos, Leo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Kyber: algorithm specifications and supporting documentation. `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`, 2009.

[4] Hayo Baan, Sauvik Bhattacharya, Scott Fluhrer, Oscar Garcia-Morchon, Thijs Laarhoven, Rachel Player, Ronald Rietman, Markku-Juhani O. Saarinen, Ludo Tolhuizen, Jose-Luis Torre-Arce, and Zhenfei Zhang. Round5: KEM and PKE based on (Ring) Learning With Rounding. `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`, 2019.

[5] Daniel J. Bernstein. Comparing proofs of security for lattice-based encryption. Cryptology ePrint Archive, Report 2019/691, 2019. `https://eprint.iacr.org/2019/691`.

[6] Sanjit Chatterjee, Neal Koblitz, Alfred Menezes, and Palash Sarkar. Another look at tightness II: practical issues in cryptography. In Raphael C.-W. Phan and Moti Yung, editors, *Paradigms in Cryptology - Mycrypt 2016. Malicious and Exploratory Cryptology - Second International Conference, Mycrypt 2016, Kuala Lumpur, Malaysia, December 1-2, 2016, Revised Selected Papers*, volume 10311 of *Lecture Notes in Computer Science*, pages 21–55. Springer, 2016.

[7] Jan-Pieter DAnvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. SABER: Mod-LWR based KEM (round 2 submission). `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`, 2019.

[8] Xianhui Lu, Yamin Liu, Dingding Jia, Haiyang Xue, Jingnan He, Zhenfei Zhang, Zhe Liu, Hao Yang, Bao Li, and Kunpeng Wang. LAC: Lattice-based Cryptosystems. `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`, 2019.

[9] 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.

[10] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, 2009.

[11] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 7.5.1)*, 2019. `https://www.sagemath.org`.

A Hoeffding Inequality

We briefly recall Hoeffding's inequality for the sum of independent random variables.

Theorem 3 (Hoeffding Inequality)**.** *Let, $X_1, X_2, \dots, X_\lambda$ be a finite sequence of independent random variables, such that for all $i = 1, \dots, \lambda$, there exists real numbers $a_i, b_i \in \mathbb{R}$, with $a_i < b_i$ and $a_i \leq X_i \leq b_i$. Let $X = \sum_{i=1}^{\lambda} X_i$. Then for any positive $t > 0$,*

$$\Pr[X - E[X] \geq t] \leq \exp\left(-\frac{2t^2}{D_\lambda}\right) \quad \textit{and} \quad \Pr[X - E[X] \leq -t] \leq \exp\left(-\frac{2t^2}{D_\lambda}\right); \tag{8}$$

where $D_\lambda = \displaystyle\sum_{i=1}^{\lambda}(b_i - a_i)^2$.