Estimating Key Sizes For High Dimensional Lattice-Based Systems

J. van de Pol and N.P. Smart

Dept. Computer Science,
University of Bristol,
United Kingdom.
joop.vandepol@bristol.ac.uk, nigel@cs.bris.ac.uk

Abstract. We revisit the estimation of parameters for use in applications of the BGV homomorphic encryption system, which generally require high dimensional lattices. In particular, we utilize the BKZ-2.0 simulator of Chen and Nguyen to identify the best lattice attack that can be mounted using BKZ in a given dimension at a given security level. Using this technique, we show that it should be possible to work with lattices of smaller dimensions than previous methods have recommended, while still maintaining reasonable levels of security. As example applications we look at the evaluation of AES via FHE operations presented at Crypto 2012, and the parameters for the SHE variant of BGV used in the SPDZ protocol from Crypto 2012.

1 Introduction

Estimating parameters for lattice-based cryptographic systems is a major problem. Such systems are becoming increasingly of interest since, to the best of our knowledge, they offer resistance to attacks that arise from the future development of a quantum computer; and in addition can offer functionality not found in traditional public key systems. This problem of parameter estimation becomes more pronounced when one considers the lattice-based schemes underlying Fully Homomorphic Encryption (FHE) [7]. This is particularly tricky as the lattice dimension in such schemes needs to be very large, so large in fact that it is unclear whether our existing methods for parameter estimation even apply. It is to this task that the current paper is focused.

The traditional measure of security of a lattice is the estimated root Hermite value δ_B (see later for a definition), for a lattice basis B output by a lattice basis reduction algorithm. In the literature one sees statements such as a δ_B of 1.05 as being "not secure", but a value of δ_B of 1.005 as being "secure". These values are given, and evidence is presented for the correctness of such statements, when in the context of relatively low lattice dimension. It is then assumed that such statements also hold when applied to large dimensional lattices, since the overall lattice dimension is not assumed to affect the difficulty of lattice reduction too much. However, such an extrapolation is clearly not valid; lattice basis reduction will be harder in higher dimension. Hence, it is not realistic to expect the same value of δ_B to be achieveable in high dimension as it is in low dimension.

In various works on FHE, for example [8], a method to produce parameter estimates which extrapolates the run time of existing lattice basis reduction implementations is used. This extrapolation is needed so as to obtain security estimates for high dimensional lattices, which are out of the reach of existing software. In particular this line of approach follows from the analysis of Lindner and Peikert [10], where an extrapolation of the performance of the Block Korkine Zolotarev (BKZ) [14] algorithm in NTL is performed. This itself poses some problems as the implementation of BKZ within NTL is very old (dating from the 1990's in some respects) and does not take into account the various optimizations and improvements which have been introduced over the years.

It turns out that on one hand the analysis in Lindner and Peikert extrapolates the run times of an implementation which does not use modern techniques, whilst on the other hand we show that the parameter estimates are too conservative. This could be explained by the fact that Lindner and Peikert look at a decoding attack, as opposed to our examination of a distinguishing attack. The decoding attack is slightly more powerful than the distinguishing attack. The decoding attack could benefit from the application of extreme pruning techniques, and the type of analysis conducted here, but it is unclear how one could analytically analyse the application of extreme pruning to decoding.

The BKZ algorithm, as one would implement it today, has a number of parameters which one can set to obtain different run-times and output qualities. Such parameters include the block size β, the number of rounds R of BKZ one runs (where each round consists of $d - \beta$ applications of finding short vectors in β dimensional projected lattices), and so-called *pruning parameters* for the search in the projected lattices. Fortunately, in [2], Chen and Nguyen present a simulation algorithm for their improved variant of BKZ. This simulation algorithm allows one to estimate the output quality of a lattice produced by the BKZ algorithm when performing R rounds with block size β. They also provide an estimate for the number of basic operations needed to perform each search, for varying values of the block size β. The term basic operation is deliberately fuzzy, but in this paper we shall take it to mean the number of nodes visited in all of the searches in the projected lattices.

Using the simulation algorithm in [2] one obtains the following "standard" method of determining the hardness of a *given* set of lattice security parameters. One first estimates the value of δ_B one would need to obtain so as to break the system, one then uses the BKZ simulator to determine how many operations this would require, and then one can deem the parameters to be secure or not. However, this in itself implies that the parameters have already been chosen, which have probably been done via appealing to the above rule of thumb in relation to "secure" values of δ_B, and by extrapolation of the runtime of existing software.

We start this work with the idea of achieving a more rational method of obtaining suitable parameters for lattice-based systems in high dimension; with a special focus on FHE systems. We will still be utilizing the simulation algorithm of [2], but in a way to *generate* parameters as opposed to testing them. In FHE systems the underlying hard problem is essentially the bounded distance decoding problem associated to LWE based lattices. This in effect has three parameters the dimension n (i.e. the ring dimension when considering ring-LWE based schemes such as the BGV system [1]), the modulus q and the distance between a lattice vector and the target vector. In LWE systems, this last quantity is essentially given by the standard deviation r chosen in the Gaussian sampling of the error vector. For fixed n we know that as the ratio r/q becomes larger the problem becomes harder to solve.

In BGV it is common to fix the value of r, and hence the only parameters one can play with are q and n. On one hand we would like q to be large so as to allow deeper circuits to be evaluated by the FHE scheme, but a large q implies low security by the above rule of thumb. To compensate for this one also selects large values of n, as can be seen in [8] where rings of dimension over 60000 are considered. Thus there is a tension in selecting q and n, between the evaluation power and the security of the resulting scheme.

In this paper we adopt the following approach . We first select a security parameter sec. This is a value, such as 80, 128 or 256, for which we feel that visiting 2^{sec} nodes in a BKZ algorithm is infeasible. Then, for a particular lattice dimension d (which for reasons we will explain later satisfies $d \geq n$) we determine the best δ_B one could obtain via a BKZ algorithm limited to visiting 2^{sec} nodes. This step is performed by using the BKZ 2.0 simulator from [2] called with various values of β and R on the estimated Gram-Schmidt lengths of an LLL-reduced basis of a random, d dimensional lattice. The notion of a random lattice will be explained in the next section. In this way the δ_B we obtain is not a fixed value (such as 1.005) but is in essence a function of d and sec. We then utilize this δ_B value in the distinguishing attack analysis of Micciancio and Regev [11], so as to obtain an equation linking n and q, in a way which guarantees 2^{sec} security. This equation can then be combined with any equation linking q and n needed to obtain evaluation of circuits of the correct depth, so as to then obtain a given set of parameters for a given specific application and/or system.

It should be noted first and foremost that things change over time. The available computing power increases as time passes by and new algorithms or attacks can be discovered. Furthermore, it is tricky to make claims about the security of lattice schemes, because it is often unclear how the behaviour of attacks in low dimensions extrapolates to higher dimensions. This work analyses one attack, which is currently believed to be the best generic attack against lattice-based schemes. It is currently unknown whether generic attacks are the best attack in every setting. In structured lattices, such as ideal or symplectic lattices, there may exist better attacks that are not yet known to the cryptographic community. Finally, in order to have confidence in any cryptographic scheme, there should be a reasonably large margin between parameters that are trivially broken and recommended secure ones. It is important to take this into account, especially when selecting parameters for lattice-based schemes.

2 Lattice Background

In this section we present the basics on lattices which we will require, and in addition present our notation.

A (full rank) lattice of dimension d is the discrete subgroup of $\mathbb{R}^d$ generated (over $\mathbb{Z}$) by a set of vectors $[\mathbf{b}_1, \ldots, \mathbf{b}_d]$ in $\mathbb{R}^d$ called the basis. It is common to represent the basis as a matrix B in which row i of the matrix B is given by the vector $\mathbf{b}_i$ (all vectors will be row vectors). Note that this is mathematically not so nice as we then always deal with row vectors, but from a programming point of view it is nicer due to being able to deal with swapping rows (i.e. basis vectors) via pointer arithmetic. Therefore, this convention is common in the literature on lattice basis reduction. We write

$$\mathcal{L}(B) = \{\mathbf{z} \cdot B : \mathbf{z} \in \mathbb{Z}^d\}.$$

A lattice basis is not unique and each basis is related to another via the relation $B' = Z \cdot B$ where $Z \in \mathsf{GL}_d(\mathbb{Z})$, i.e. Z is an integer matrix with determinant ± 1. We often use the shorthand $\mathcal{L}$ for $\mathcal{L}(B)$ if the underlying basis (which of course does not really matter) is clear.

On vectors in $\mathbb{C}^d$ we can define the following norms

$$\|\mathbf{x}\|_p = \begin{cases} \left(\sum_{i=1}^d |x_i|^p\right)^{1/p} & p \neq \infty \\ \max_{i=1}^d |x_i| & p = \infty. \end{cases}$$

Being a discrete structure there is a well defined quantity of a non-zero minimum of the lattice, which we denote by

$$\lambda_1^{(p)}(\mathcal{L}) := \min\{\|\mathbf{x}\|_p : \mathbf{x} \in \mathcal{L}, \mathbf{x} \neq \mathbf{0}\}.$$

We can also define the successive minima $\lambda_i^{(p)}(\mathcal{L})$, which are defined as the smallest radius r such that the d-dimensional ball of radius r centred on the origin contains i linearly independent lattice points. To ease notation, and because we will be mainly working with the 2-norm, we write $\lambda_i(\mathcal{L}) = \lambda_i^{(2)}(\mathcal{L})$.

For any basis B we define the fundamental region as the set

$$\mathcal{P}(B) = \left\{ \sum_{1 \leq i \leq d} x_i \cdot \mathbf{b}_i : x_i \in [0, 1) \right\}.$$

The d-dimensional volume, $\Delta(\mathcal{L}) = \mathsf{Vol}(\mathcal{P}(B))$, is called the fundamental volume, and can be computed via $\Delta(\mathcal{L}) = |\det(B)|$. It is clear that this quantity is an invariant of the lattice, and does not depend on the precise basis chosen. The *dual* $\mathcal{L}^*$ of a lattice $\mathcal{L}$ is the set of all vectors $\mathbf{y} \in \mathbb{R}^d$ such that $\mathbf{y} \cdot \mathbf{x}^\mathsf{T} \in \mathbb{Z}$ for all $\mathbf{x} \in \mathcal{L}$. Given a basis matrix B of $\mathcal{L}$ we can compute the basis matrix B^* of $\mathcal{L}^*$ via $B^* = (B^{-1})^\mathsf{T}$. Hence we have $\Delta(\mathcal{L}^*) = 1/\Delta(\mathcal{L})$.

The classic result in lattice theory (a.k.a. geometry of numbers), is that of Minkowski, which relates the minimal distance to the fundamental volume.

Theorem 1 (Minkowski's Theorem). *For any d dimensional lattice $\mathcal{L}$ we have*

$$\lambda_1(\mathcal{L}) \leq \sqrt{d} \cdot \Delta(\mathcal{L})^{1/d}.$$

The notion of a random lattice stems from work by Goldstein and Mayer [9]. Consider lattices with a prime determinant p. For large p the vast majority of these lattices are of the following type:

$$\begin{pmatrix} p & & & \\ x_1 & 1 & & \\ \vdots & & \ddots & \\ x_{d-1} & & & 1 \end{pmatrix}.$$

Goldstein and Mayer show that lattices generated by taking p at random and taking x_i independently and uniformly at random in $\{0, \ldots, p-1\}$ are in some (natural) sense random. These lattices are often studied when considering the behaviour of basis reduction algorithms [12,5].

For such random lattices the first minimum is approximated by the *Gaussian Heuristic*, which states that for a random lattice we have

$$\lambda_1(\mathcal{L}) \approx \sqrt{\frac{d}{2 \cdot \pi \cdot e}} \cdot \Delta(\mathcal{L})^{1/d}.$$

Hermite showed that there is an absolute constant γ_d, depending only on d, such that

$$\lambda_1(\mathcal{L}) \leq \sqrt{\gamma_d} \cdot (\Delta(\mathcal{L}))^{1/d}.$$

The value of γ_d (called "Hermite's constant") is, however, only known for $1 \leq d \leq 8$ and $d = 24$.

A specific basis B is said to have *Hermite factor* δ_B^d, or *root Hermite factor* δ_B, if

$$\|\mathbf{b}_1\|_2 = \delta_B^d \cdot \Delta(\mathcal{L})^{1/d}.$$

The root Hermite factor of the *lattice* is said to be the constant δ_L such that

$$\lambda_1(\mathcal{L}) = \delta_L^d \cdot \Delta(\mathcal{L})^{1/d}.$$

In lattice basis reduction algorithms we are trying to determine an output lattice basis such that $\delta_B = \delta_L$, i.e. the first vector in the basis is the shortest vector.

By the Gaussian heuristic we have for a random lattice

$$\delta_L \approx \left(\sqrt{\frac{d}{2 \cdot \pi \cdot e}}\right)^{1/d}.$$

3 Estimating BKZ

In this section we provide an overview of the prior work on analysing the BKZ algorithm and then present our results on estimating the output δ_B from BKZ, for a specific dimension and with an explicit limit on the number of nodes evaluated. In later sections we will use this analysis to estimate parameters for the LWE based systems used in FHE schemes.

BKZ Overview. Throughout the paper we assume the input basis to the BKZ algorithm has been LLL reduced (i.e., reduced by the LLL algorithm). The BKZ algorithm, as modified in [2] and called BKZ 2.0, is parameterized by two parameters R and β and operates as follows. The algorithm executes the following round function R times. In each round we iterate the index i from one to $d - \beta$, and for each value of i we take the β-dimensional projected lattice generated by the basis vectors $\mathbf{b}_i, \ldots, \mathbf{b}_{i+\beta-1}$ projected onto the orthogonal space spanned by the first $i - 1$ basis vectors. A small vector is obtained in the projection of this lattice, and the resulting vector is inserted into the main lattice basis at the ith position. The search for the small vector in the projected lattice is performed by an enumeration method using a heuristic called *extreme pruning* [6].

Historical Background. The line of work aimed at assessing the behaviour of basis reduction algorithms in practice was started by Gama and Nguyen [5]. They considered this behaviour from an experimental point of view and tried to extrapolate it to higher dimensions (although not the astronomical dimensions required in FHE schemes). Specifically, they analyse the behaviour of basis reduction algorithms when applied to solving various lattice problems, such as Hermite-SVP, Approximate SVP and Unique SVP. However, since BKZ 2.0 did not exist at the time, they analysed the original BKZ which did not use extreme pruning and did not abort after a fixed number of rounds, but would instead run until termination.

The most interesting result from these experiments was that basis reduction algorithms output a basis B which appeared to solve Hermite-SVP, i.e. finding a short basis vector, with Hermite Factor δ_B^d. The interesting part is that on average, the δ_B observed in practice was much smaller than theoretical worst-case bounds obtained from analysing the reduction algorithms theoretically. It should be noted that this worst-case behaviour was tied to the basis of the particular lattice, rather than to the lattice itself. Applying the basis reduction algorithms to a 'randomized' basis of the same lattice resulted in average-case rather than worst-case behaviour. Gama and Nguyen conjectured that the value of δ_B of the output basis depends mostly on the basis reduction algorithm that was used and not on the input lattice (unless this lattice has special structure). The value also depended on the dimension d but appeared to converge quickly as d increases.

Gama and Nguyen drew several conclusions. Most importantly, they concluded that with the basis reduction algorithms available at that time, $\delta_B = 1.01$ was the best reachable root-Hermite factor. They also examined the run-time of exact SVP solvers and concluded that up to dimension 60 the shortest vector problem could be solved within an hour, whereas dimension 100 seemed out of reach. They also observed that BKZ with block sizes much higher than 25 was not realistic in higher dimensions due to run-time constraints. Once again, these observations were before the discovery of extreme pruning and before the adoption of aborting BKZ after a fixed number of rounds R.

It should also be noted that this work was not aimed at cryptography, but only at basis reduction algorithms in a general setting. Hence, Gama and Nguyen did not experiment specifically with lattices that arise from a cryptographic setting, but instead with random lattices from the Goldstein Mayer distribution [9] (as described in Section 2) and some specially structured lattices for the unique shortest vector problem.

Gama, Nguyen and Regev in 2010 [6] proposed improved heuristics for solving SVP using enumeration via a technique called extreme pruning. Potentially, this technique could be used with the enumeration of the β dimensional projected lattices within the BKZ algorithm. However, this heuristic technique requires a pretty good estimate of the length of the shortest vector. But Gama and Nguyen had already observed that the projected lattices that occur in BKZ with low block size (say $\beta < 50$) do not follow the distribution of random lattices. More specifically, these projected lattices did not adhere to the Gaussian Heuristic, which would have given a good approximation to the length of the shortest vector. Thus, extreme pruning cannot trivially be applied to BKZ with low block size.

But then Chen and Nguyen [2] made the observation that the projected lattices that appear in BKZ for higher blocksizes (say $\beta > 50$) behave like random lattices as far as the Gaussian Heuristic is concerned. This enables the introduction of extreme pruning and several other heuristic improvements to BKZ, resulting in the BKZ 2.0 algorithm outlined above. The BKZ 2.0 algorithm is able to reduce lattices with much higher block sizes in practice than the original BKZ. This observation about the projected lattices and Gaussian Heuristic also allowed Chen and Nguyen to create a simulator for BKZ 2.0, which simulates the behaviour of the algorithm on the lengths of the Gram-Schmidt vectors of the basis. This makes it much easier to heuristically explain (for large enough block size) the behaviour of BKZ in practice and the associated output δ_B, even for block sizes that we might not be able to run in practice.

Chen and Nguyen use the simulator to estimate the approximate security of the NTRU encryption scheme and the Gentry-Halevi FHE challenges. Specifically for the challenges by Gentry and Halevi they reason as follows. From the parameters of the scheme they can derive that they require a root-Hermite factor of δ_B. They use the simulation to estimate that this requires R rounds of BKZ with block size β (starting from an LLL-reduced basis). Using an upper bound for the cost of a block size β enumeration derived from experiments, they convert the R rounds into the number of enumeration nodes (given that each round consists of $d - \beta$ enumerations where d is the dimension of the lattice). This number of nodes gives a rough estimate for the bit-security of the specific parameters of the scheme.

Our Approach. In the heuristic approach by Chen and Nguyen (and others), an estimated secrity level is essentially derived from a system with given parameters. However, we would like to choose our parameters according to a given security level. Thus, we reverse the analysis by Chen and Nguyen and try to answer the question: Given a security level of sec such that the adversary can only perform 2^{sec} operations, how should we choose our parameters such that our system is secure against this adversary?

Say we choose the dimension d of a Goldstein Mayer lattice and a security level sec. Now, an adversary can attempt to run BKZ with block size β, for varying β. For each β, we can approximate the cost of a single enumeration using the tables from Chen and Nguyen [2]. Then, we can compute how many enumerations we could maximally perform with this block size without exceeding 2^{sec} nodes. This bound on the number of enumerations gives us a bound on the number of rounds R, say $R(\beta, d, \mathsf{sec})$, for the dimension d as well. Now we can simulate the behaviour of $R(\beta, d, \mathsf{sec})$ rounds of BKZ with block size β on a random LLL-reduced basis of a d-dimensional Goldstein Mayer lattice, using the simulation algorithm from [2]. This allows us to predict the root-Hermite factor δ_B of the output basis from BKZ. Thus, on input of d, sec and β, we obtain a value of δ_B. If we perform this procedure for all block sizes β, we find an estimated value of δ_B (one for each β). Taking the minimum of all such δ_B we obtain an estimate for the best value of δ_B which can be obtained by an adversary which is limited to enumerating at most 2^{sec} nodes.

Doing this for a number of increasing dimensions we find the data in Table 1 for the estimate of the best δ_B an adversary can obtain in a given dimension d. Unsurprisingly we see that as the dimension increases the best value of δ_B that one can obtain also increases, although the increase is not too pronounced. This can be explained as follows. If we allow BKZ with block size β to run indefinitely, so for unbounded R, the simulation suggests that δ_B of the output basis converges to some value that seems to only depend on β (consistent with the observations from [5]). However, as the dimension increases, performing a round of BKZ becomes more costly. Furthermore, the simulation also indicates that in higher dimensions it converges more slowly to this value δ, i.e., it takes a larger number of rounds R to reach it. In higher dimensions, BKZ with block size β reaches a worse δ_B in $R(\beta, d, \mathsf{sec})$ rounds than BKZ with block size $\beta' < \beta$ in $R(\beta', d, \mathsf{sec})$ rounds. The results in Table 1 assume that the estimated number of nodes visited during an enumeration reported in [2] cannot be improved by further algorithmic improvements.

	d							
sec	1024	2048	4096	8192	16384	32768	65536	131072
80	1.0081	1.0081	1.0084	1.0084	1.0088	1.0088	1.0092	1.0092
128	1.0067	1.0067	1.0067	1.0069	1.0069	1.0069	1.0069	1.0072
256	1.0055	1.0055	1.0055	1.0055	1.0055	1.0055	1.0055	1.0055

Table 1. Smallest achievable δ_B by BKZ in dimension d and evaluating at most 2^{sec} nodes.

For $d > 2^{17}$, the BKZ simulator is rather slow, but for the applications in Section 5 dimensions up to 2^{17} are sufficient. Therefore, only dimensions up to d^{17} were considered here. The value of δ_B achievable when evaluating at most 2^{256} nodes is achieved by performing BKZ with block size 250. Since Chen and Nguyen only give the cost of enumerations up to block size 250, it is possible that an attacker could use BKZ with a higher block size and achieve a better δ_B, while evaluating no more than 2^{256} nodes. Because it was not possible to reproduce the costs for varying block sizes and because it is unclear how to realistically extrapolate the costs to higher block sizes, the value of δ_B here corresponds to block size 250.

4 Estimating LWE Parameters

Our goal is to provide estimates for LWE parameters for specific cryptographic systems in large dimensions, given the estimates in the previous section. Before proceeding we recap a little on notation and prior analysis so as to fix notation. The LWE problem, and hence to the best of our knowledge the ring-LWE problem, is based upon arithmetic in q-ary lattices.

q-ARY LATTICES. A q-ary lattice $\mathcal{L}$ of dimension n is one such that $q\mathbb{Z}^n \subset \mathcal{L} \subset \mathbb{Z}^n$ for some integer q. Note that all integer lattices are q-ary lattices for a value of q which is an integer multiple of $\Delta(\Lambda)$. Our interest will be in special lattices which are q-ary for a value of q much smaller than the determinant. Much of our discussion follows that in [11].

Suppose we are given a matrix $A \in \mathbb{Z}_q^{n\times d}$, with $d \geq n$, we then define the following two d-dimensional q-ary lattices.

$$\begin{aligned}\Lambda_q(A) &= \left\{\mathbf{y} \in \mathbb{Z}^d : \mathbf{y} = \mathbf{z}\cdot A \pmod q \text{ for some } \mathbf{z}\in\mathbb{Z}^n\right\},\\ \Lambda_q^{\perp}(A) &= \left\{\mathbf{y}\in\mathbb{Z}^d : \mathbf{y}\cdot A^{\mathsf{T}} = 0 \pmod q\right\}.\end{aligned}$$

Suppose we have $\mathbf{y} \in \Lambda_q(A)$ and $\mathbf{y}' \in \Lambda_q^{\perp}(A)$ then we have $\mathbf{y} = \mathbf{z}\cdot A$ and $\mathbf{y}'\cdot A^{\mathsf{T}} = 0 \pmod q$. This implies that

$$\mathbf{y}\cdot\mathbf{y}'^{\mathsf{T}} = (\mathbf{z}\cdot A)\cdot\mathbf{y}'^{\mathsf{T}} = \mathbf{z}\cdot(\mathbf{y}'\cdot A^{\mathsf{T}})^{\mathsf{T}} \in q\cdot\mathbb{Z}.$$

Hence, the two lattices are, up to normalisation, duals of each other. We have $\Lambda_q(A) = q\cdot\Lambda_q^{\perp}(A)^*$ and $\Lambda_q^{\perp}(A) = q\cdot\Lambda_q(A)^*$.

To fix ideas consider the following example; Let $n = 2$, $m = d = 3$, $q = 1009$ and set

$$A = \begin{pmatrix}1 & 2 & 3\\ 3 & 5 & 6\end{pmatrix}.$$

To define a basis B of $\Lambda_q(A)$ we can take the row-HNF of the 5×3 matrix $\begin{pmatrix}A\\ q\cdot I_3\end{pmatrix}$ to obtain

$$B = \begin{pmatrix}1009 & 0 & 0\\ 1 & 1 & 0\\ 336 & 0 & 1\end{pmatrix}.$$

The basis of $\Lambda_q^{\perp}(A)$ is given by

$$B^* = q\cdot((B^{\mathsf{T}})^{-1}) = \begin{pmatrix}1 & -1 & -336\\ 0 & 1009 & 0\\ 0 & 0 & 1009\end{pmatrix}.$$

The properties of the above example hold in general; namely if q is prime and (in general) if d is a bit larger than n then we have $\Delta(\Lambda_q(A)) = q^{d-n}$ and $\Delta(\Lambda_q^{\perp}(A)) = q^n$.

We now turn to discussing how short the vectors are that one can find in q-ary lattices. Let us focus on the lattice $\Lambda_q^{\perp}(A)$, which will be more important for our analysis. We know that this contains vectors of length q (since it is a q-ary lattice), we assume that lattice reduction will output a basis B with root Hermite factor δ_B for some value of δ_B. This means that computationally the shortest vector we can produce in the lattice $\Lambda_q^{\perp}(A)$ will be of size

$$\min(q, \delta_B^d\cdot q^{n/d})$$

since $\Delta(\Lambda_q^{\perp}(A)) = q^n$.

LWE PROBLEM. The LWE problem is parametrized by four parameters n, d, q and $r = s/\sqrt{2\pi}$. To define the problem we introduce the Gaussian distribution in one variable with parameter s (and mean zero) as the distribution with probability distribution function proportional to

$$f(x) = \frac{1}{s}\exp\left(-\frac{\pi\cdot x^2}{s^2}\right).$$

Thus we have that the standard deviation is given by $r = s/\sqrt{2\cdot\pi}$. The (spherical) multivariate normal distribution on $\mathbb{R}^n$, with Gaussian parameter s (resp. standard deviation r) is given by

$$f(\mathbf{x}) = \frac{1}{s}\exp\left(-\frac{\pi\cdot\|\mathbf{x}\|_2^2}{s^2}\right) = \frac{1}{r\cdot\sqrt{2\cdot\pi}}\exp\left(-\frac{\|\mathbf{x}\|_2^2}{2\cdot r^2}\right).$$

Sampling from this distribution is performed by simply sampling each component of the vector $\mathbf{x}$ independently from $N(0, r)$.

The discrete Gaussian distribution, with support on the lattice $\mathcal{L}$ with Gaussian parameter s (equivalently standard deviation $r = s/\sqrt{2 \cdot \pi}$, denoted $D_{\mathcal{L},s}$, is the probability distribution on $\mathcal{L}$ which selects $\mathbf{x} \in \mathcal{L}$ with probability proportional to $\exp(-\pi \cdot \|\mathbf{x}\|_2^2/s^2)$.

Definition 1 (LWE Decision Problem). *Given $(A, \mathbf{v})$ where $A \in \mathbb{Z}_q^{n \times d}$ and $\mathbf{v} \in \mathbb{Z}_q^d$ determine which of the following distributions $\mathbf{v}$ is from:*

1. *$\mathbf{v}$ is chosen uniformly at random from $\mathbb{Z}_q^d$.*
2. *$\mathbf{v} = \mathbf{s} \cdot A + \mathbf{e}$ where $\mathbf{e}, \mathbf{s} \leftarrow D_{\mathbb{Z}^n,s}$.*

The link between LWE and q-ary lattices is then immediately obvious. Given A and $\mathbf{v}$ the decision problem is to determine whether $\mathbf{v}$ is a random point or an element which is close to a point in the lattice $\Lambda_q(A)$.

The natural "attack" against the decision LWE problem is to first find a short vector $\mathbf{w}$ in the dual lattice $\Lambda_q(A)^*$ and then check whether $\mathbf{w} \cdot \mathbf{v}^\mathsf{T}$ is close to an integer. If it is, one concludes that the input vector is an LWE sample, whereas if it is not one concludes that the input vector is random. Thus to ensure security, following the argument in [11, Section 5.4.1], we require

$$r \geq \frac{1.5}{\|\mathbf{w}\|_2}.$$

Now from earlier, we deduce that when applying lattice reduction to the lattice $\Lambda_q(A)^* = \frac{1}{q}\Lambda_q^\perp(A)$ we will obtain a vector $\mathbf{w}$ with

$$\|\mathbf{w}\|_2 \approx \frac{1}{q} \min(q, \delta_B^d \cdot q^{n/d}).$$

The point is that we have some freedom in choosing d here, since it is related to the number of LWE samples we take. In the traditional analysis [11] one assumes δ_B is already given and one then applies calculus to minimize the above estimate for $\|\mathbf{w}\|_2$ by picking d as a function of q, n and δ_B. But as we presented in Section 3 the value of δ_B is essentially a function of d and sec.

Our Analysis. We make the heuristic assumption that the behaviour of applying the BKZ lattice basis reduction technique to the d dimensional lattice $\Lambda_q(A)^*$ performs roughly the same as the application to the Goldstein Mayer lattices in Section 3. For the above distinguishing attack to fail to work we require

$$q^{n/d-1} \geq \frac{1.5}{r \cdot \delta_B^d} = c_{r,d,\mathsf{sec}}.$$

For fixed values of r we can derive, using the method in Section 3, values of $c_{r,d,\mathsf{sec}}$ for any value of sec and d that we require. We therefore require, to ensure security, that for all $d \geq n$ we have

$$n \log_2 q - d \log_2 q \geq d \cdot \log_2 c_{r,d,\mathsf{sec}}.$$

Note that, as a sanity check, for fixed n this means we have an upper bound on $\log_2 q$ of

$$\log_2 q \leq \min_{d>n} \frac{-d \cdot \log_2 c_{r,d,\mathsf{sec}}}{d-n}. \qquad (1)$$

We end this section by discussing what this means for a simple LWE based system at the security level of 80 bits; in particular we determine what the maximum value of q could be when we fix $n = 4000$ and $r = 3.2$. We first derive a more detailed version of Table 1 and use linear interpolation to determine estimated δ_B values for dimensions not in our table; this needs to be done once and for all, in all of our analysis. Retuning to considering our specific values of n and r: We enumerate all $d > n$ up to 2^{17}, and use the linear interpolation of Table 1 to determine a value of δ_B for reducing a lattice of dimension d at this security level. This enables us to obtain an upper bound on $\log_2 q$, over all values of d, from Equation 1. Indeed we obtain an upper bound of $\log_2 q$ of 195 and the "best" value of d for the distinguishing attack comes out as $d = 8045$ with $\delta_B \approx 1.0084$. We compare this with the traditional analysis which assumes δ_B given and then computed d as $d = \sqrt{n \cdot \log(q)/\log(\delta_B)}$, which would give us a value of $d \approx 8045$ as well, as expected. However, we reiterate that this traditional method of obtaining d comes from somehow estimating the value of δ_B one would obtain in performing BKZ on lattices of (an as yet unknown) dimension d.

5 Application of our method to two examples

As a first application we re-evaluate the parameters in (the full version of) [8]. The authors of [8] determine parameters for their SHE scheme so as to homomorphically evaluate large circuits, including the AES circuit. They select a security level equivalent to 80 bits of security and derive sizes for the resulting parameters to evaluate circuits of multiplicative depth L, for various values of L. In order to compare the results, we will consider the same security level.

In [8, Appendix C], they use the security analysis by Lindner and Peikert [10] to derive a lower bound on the approximate ring dimension $n = \phi(m)$ depending on the largest modulus Q, standard deviation r and the security level sec, which guarantees the security of the scheme. In particular the lower bound is

$$n \geq \frac{\log(Q/r)(\mathsf{sec} + 110)}{7.2}. \tag{2}$$

To guarantee the functionality of the L-leveled homomorphic scheme, they then derive an estimate on the size of Q needed to evaluate a circuit of depth L, this is given by

$$Q \approx 2^{22.5 \cdot L - 3.6} \cdot r \cdot n^L.$$

The individual moduli in the SHE scheme are given by

$$p_0 \approx 2^{23.9} \cdot n, \qquad p_i \approx 2^{11.3}\sqrt{n} \text{ for } i = 1, \ldots, L-2, \qquad p_{L-1} \approx \sqrt{n} + 11,$$

and

$$P \approx 2 \cdot 308^L \cdot \zeta^{L-2} \cdot r \cdot n^{L/2}.$$

Combining the two equations for Q, setting $\mathsf{sec} = 80$, $\zeta = 8$ and $r = 3.2$ they derive values of n and Q for various values of L.

In our analysis, we replace the security-related lower bound (2) on n by the equivalent upper bound from Equation (1) on Q, given n. Now, we increase n, in steps of 100 from a given starting value, until the upper bound on Q is above the estimate for Q needed to ensure correct evaluation of a circuit of multiplicative depth L. We present our results, and the comparison with those in [8] in Table 2. As one can see the methodology for choosing parameters in this paper results in roughly the same values for the moduli, but also produces significantly smaller lattice dimensions. In practice, this will translate into faster overall performance figures for the SHE scheme.

L	Estimates from [8]					Our Estimates				
	n	$\ell_2(p_0)$	$\ell_2(p_i)$	$\ell_2(p_{L-1})$	$\ell_2(P)$	n	$\ell_2(p_0)$	$\ell_2(p_i)$	$\ell_2(p_{L-1})$	$\ell_2(P)$
10	9326	37.1	17.9	7.5	177.3	7100	36.7	17.7	6.6	163.3
20	19434	38.1	18.4	8.1	368.8	14300	37.7	18.2	7.0	369.0
30	29749	38.7	18.7	8.4	564.2	21600	38.3	18.5	7.3	550.6
40	40199	39.2	18.9	8.6	762.2	28500	38.7	18.7	7.5	743.3
50	50748	39.5	19.1	8.7	962.1	35500	39.0	18.6	7.6	937.9
60	61376	39.8	19.2	8.9	1163.5	42900	39.3	18.9	7.7	1134.3
70	72071	40.0	19.3	9.0	1366.1	50400	39.5	19.1	7.9	1332.1
80	82823	40.2	19.4	9.1	1569.8	57900	39.7	19.2	7.9	1530.9
90	93623	40.4	19.5	9.2	1774.5	65500	39.9	19.3	8.0	1730.6

Table 2. Table comparing the estimates from [8] with our estimates. Here $\ell_2(x) = \log_2(x)$.

As another example we look at the example parameters used in the SPDZ MPC protocol, see [3,4]. In [3] parameters are given for instantiating the SPDZ MPC protocol over fields of prime characteristic of size 32, 64 and 128 bits. The resulting parameter sets have lattice dimensions 8192, 16384 and 32768 respectively. In the prime characteristic case greater efficiency is obtained in the protocol if one has lattices of dimension a power of two. If one performs the same analysis as in [3] for the case of characteristic two one finds that the resulting dimension will have size *roughly* 8192.

By using our analysis we find that we can securely use dimensions of size roughly 4096 (for characteristic two), 8192 (for prime characteristic of size roughly 2^{32}) 16384 (for prime characteristic of size roughly 2^{64}) 16384 (for prime characteristic of size roughly 2^{128}). Thus we obtain a more efficient scheme for the case of characteristic two and for very large prime characteristic only. The reason for the lack of a general improvement is that for odd prime characteristic, the dimensions are restricted to a power of two due to scheme specific efficiency.

6 Acknowledgements

The authors would like to acknowledge the partial support of of the ERC (Advanced Grant ERC-2010-AdG-267188-CRIPTO), the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL) (under agreement number FA8750-11-2-0079)[1], and by EPSRC (Grant EP/I03126X). The second author has been supported in part by a Royal Society Wolfson Merit Award. The authors also thank the anonymous reviewers for their helpful comments and insights.

References

1. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS*, pages 309–325. ACM, 2012.
2. Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2011.
3. Ivan Damgård, Marcel Keller, Enrique Larraia, Valerio Pastro, Peter Scholl, and Nigel P. Smart. Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. *IACR Cryptology ePrint Archive*, 2012:642, 2012.
4. Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In Safavi-Naini and Canetti [13], pages 643–662.
5. Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 31–51. Springer, 2008.
6. Nicolas Gama, Phong Q. Nguyen, and Oded Regev. Lattice enumeration using extreme pruning. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 257–278. Springer, 2010.
7. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *STOC*, pages 169–178. ACM, 2009.
8. Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the AES circuit. In Safavi-Naini and Canetti [13], pages 850–867.
9. Daniel Goldstein and Andrew Mayer. On the equidistribution of Hecke points. *Forum Math.*, 15:165–189, 2003.
10. Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *CT-RSA*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer, 2011.
11. Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-Quantum Cryptography*, pages 147–192. Springer, 2009.
12. Phong Q. Nguyen and Damien Stehlé. Lll on the average. In Florian Hess, Sebastian Pauli, and Michael E. Pohst, editors, *ANTS*, volume 4076 of *Lecture Notes in Computer Science*, pages 238–256. Springer, 2006.
13. Reihaneh Safavi-Naini and Ran Canetti, editors. *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*. Springer, 2012.

[1] The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Defense Advanced Research Projects Agency (DARPA) or the U.S. Government.

14. C.P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. In L. Budach, editor, *Fundamentals of Computation Theory*, volume 529 of *Lecture Notes in Computer Science*, pages 68–85. Springer Berlin Heidelberg, 1991.