




An Experimental Study of Kannan's Embedding Technique for the Search LWE Problem

Yuntao Wang^{1,4}, Yoshinori Aono², and Tsuyoshi Takagi^{3,4}

¹ Graduate School of Mathematics, Kyushu University, Fukuoka, Japan
y-wang@math.kyushu-u.ac.jp

² National Institute of Communication and Technology, Tokyo, Japan
aono@nict.go.jp

³ Institute of Mathematics for Industry, Kyushu University, Fukuoka, Japan
takagi@imi.kyushu-u.ac.jp

⁴ Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan

Abstract. The learning with errors (LWE) problem is considered as one of the most compelling candidates as the security base for the post-quantum cryptosystems. For the application of LWE based cryptographic schemes, the concrete parameters are necessary: the length n of secret vector, the moduli q and the deviation σ . In the middle of 2016, Germany TU Darmstadt group initiated the LWE Challenge in order to assess the hardness of LWE problems. There are several approaches to solve the LWE problem via reducing LWE to other lattice problems. Xu et al.'s group solved some LWE Challenge instances using Liu and Nguyen's adapted enumeration technique (reducing LWE to BDD problem) [14] and they published this result at ACNS 2017 [23]. In this paper, we study Kannan's embedding technique (reducing LWE to unique SVP problem) to solve the LWE problem in the aspect of practice. The lattice reduction algorithm we use is the progressive BKZ [2, 3]. At first, from our experimental results we can intuitively observe that the embedding technique is more efficient with the embedding factor M closer to 1. Then especially for the cases of $\sigma/q = 0.005$, we will give an preliminary analysis for the runtime and give an estimation for the proper size of parameters. Moreover, our experimental results show that for $n \geq 55$ and the fixed $\sigma/q = 0.005$, the embedding technique with progressive BKZ is more efficient than Xu et al.'s implementation of the enumeration algorithm in [21, 23]. Finally, by our parameter setting, we succeeded in solving the LWE Challenge over $(n, \sigma/q) = (70, 0.005)$ using $2^{16.8}$ s (32.73 single core hours).

Keywords: Lattice · LWE Challenge · BDD · Unique SVP
Embedding technique · Lattice reduction · Post-quantum cryptography

1 Introduction

Nowadays many post-quantum cryptographic schemes as fully homomorphic encryption and lattice-based signature schemes base their security on some

lattice hard problems as the learning with errors (LWE) problem, short integer solution (SIS) and so on [9, 16, 18]. LWE problem was introduced by Regev in 2005, which comes from “learning parity with noise” by lifting the moduli value, and concreting the probability distribution of “error” [18]. As an average-case lattice problem, LWE problem is proved as hard as certain worst-case lattice problems such as GapSVP and SIVP [18], which allows to build many provably secure lattice-based cryptographic schemes. The hardness of LWE problem is related to three critical parameters: the length n of secret vector, the moduli q and the deviation σ of error vectors. Some theoretical analysis for the hardness of LWE are given as lattice-based attack [14, 15], and BKW type attack [11], but rarely concrete parameters based on experiments were published. However, for the practical application, it is indispensable to estimate the concrete parameters of LWE from sufficient experiments. In this work, we focus on the more practical lattice-based attack. At first, the LWE problem can be seen as a particular bounded distance decoding (BDD) instance on a q -ary lattice. For a given lattice and a target vector close to the lattice points in a reasonable bound, BDD is to find the closest lattice vector to the target.

There are two main methods to process the BDD instance. One is reducing the lattice basis first and search the secret vector by Babai’s NearestPlane [4] algorithm or its variants [14, 15]. Especially in [14], Liu and Nguyen intermingle the short error vector into an enumeration searching tree, which makes the attack more efficient. Another procedure is to reduce BDD to the unique-shortest vector problem (unique-SVP) by Kannan’s embedding technique [10]. This procedure increase one more lattice dimension by adding the target vector and a so-called embedding factor M into the new basis. By a proper parameter setting, the short error vector is usually a component of the shortest vector in the new lattice. So there is a big gap between the shortest vector and the second shortest vector in the new lattice, which makes a lattice reduction algorithm or a searching algorithm find the shortest one more efficiently. Since both methods call the SVP solver as subroutine, their complexity grow exponentially with the dimension increasing.

In order to assess the hardness of the LWE problem in practice, TU Darmstadt, in alliance with UC San Diego and TU published, a new platform “Darmstadt LWE Challenge” [5, 21]. LWE Challenge provides LWE samples by increasing hardness for researcher to test their solving algorithms.

In this work, we apply the embedding technique on LWE problem, using state-of-the-art progressive BKZ algorithm [2]. The LWE instances used in our experiments are sampled from Darmstadt LWE Challenge. From our experiments, we find that the algorithm can derive a better efficiency if the embedding factor M is closer to 1. We also give an preliminary analysis for the proper parameter as the dimension m of LWE samples should be used in the attack associate to the secret length n . Especially for $n \geq 55$ and the fixed $\sigma/q = 0.005$, our implemented embedding technique with progressive BKZ is more efficient than Xu et al.’s implementation of the enumeration algorithm in [21, 23]. Finally, we got the records of case (70, 0.005) in Darmstadt LWE Challenge, using our extrapolated setting of m , which took 32.73 single core hours.

Roadmap. Section 2 recalls the notations and background on lattice, LWE problem and BKZ reduction algorithms. We introduce Kannan’s embedding technique in Sect. 3. Our experimental results and preliminary analysis on the relevant parameters settings in Kannan’s embedding technique are shown in Sect. 4. Finally we give some conclusions in Sect. 5.

2 Preliminaries

2.1 Lattice Theory

A lattice L is an infinite regular space expanded by a basis $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$, where \mathbf{b}_i ($i = 1, \dots, n$) are a set of linearly independent row vectors in \mathbb{R}^m and n is the dimension of L . Note that even \mathbf{B} is matrix with row vectors, in this paper we use integral lattice for convenience we write the basis in a matrix form as $\mathbf{B} = (\mathbf{b}_1, \dots, \mathbf{b}_n) \in \mathbb{Z}^{n \times m}$, where we will declare the matrix with column vectors if it appears, as the matrix \mathbf{A} in LWE problem. The n -dimensional *volume* of L is denoted by $\text{vol}(L)$, which is computed by the *determinant* of the basis B , i.e. $\text{vol}(L) = \det(B)$. The Euclidean norm of a vector $\mathbf{v} \in \mathbb{R}^m$ is $\|\mathbf{v}\|$. We denote by $V_n(R) = R^n \cdot \frac{\pi^{n/2}}{\Gamma(n/2+1)}$ the volume of n -dim Euclidean ball of radius R .

Shortest Vector. There are at least two non-zero vectors with same minimal Euclidean norm but contrary sign in a lattice L : this norm is called the *first minimum* $\lambda_1(L)$ of L . The *shortest vector* of L refers to one of the vectors whose norms are both $\lambda_1(L)$. Similarly we denote by $\lambda_2(L)$ the length of the second shortest vector, which is linearly independent of the first shortest vectors.

Orthogonalization. We denote by $\mathbf{B}^* = (\mathbf{b}_1^*, \dots, \mathbf{b}_n^*)$ the associated Gram-Schmidt orthogonalization of the given basis $\mathbf{B} = (\mathbf{b}_1, \dots, \mathbf{b}_n)$. Here $\mathbf{b}_1^* = \mathbf{b}_1$ and $\mathbf{b}_i^* = \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{ij} \mathbf{b}_j^*$ for all $2 \leq i \leq n$ while $\mu_{ij} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\|\mathbf{b}_j^*\|^2}$ ($1 \leq j < i \leq n$).

Hermite Factor. To estimate the performance of the algorithm on solving SVP, we usually use the Hermite factor which is defined in [8] as:

$$\text{HF}(\mathbf{b}_1, \dots, \mathbf{b}_n) = \|\mathbf{b}_1\|/\text{vol}(L)^{1/n}.$$

So for a lattice of dimension n , we say the algorithm performs better if the Hermite factor of output is smaller. Also we usually use root Hermite factor convenient for analysis, which is denoted by:

$$\delta = \text{rHF}(\mathbf{b}_1, \dots, \mathbf{b}_n) = (\|\mathbf{b}_1\|/\text{vol}(L)^{1/n})^{1/n}$$

Note that our definition of rHF depending on the given bases and the output $(\mathbf{b}_1, \dots, \mathbf{b}_n)$ of the short vector from lattice algorithms.

Gaussian Heuristic. Given a n -dimensional lattice L and a continuous (and usually convex) set $S \subset \mathbb{R}^n$, Then the *Gaussian heuristic* estimates that the number of points in $S \cap L$ is approximately $\text{vol}(S)/\text{vol}(L)$.

Particularly, taking S as the origin-centered ball of radius R , the number of lattice point is $V_n(R)/\text{vol}(L)$, which derives the length of shortest vector λ_1 so that the volume of ball is equal to that of lattice:

$$\lambda_1(L) \approx \frac{(\Gamma(n/2 + 1)\text{vol}(L))^{1/n}}{\sqrt{\pi}}$$

This is the so-called *Gaussian heuristic of a lattice*, and we denote it by $\text{GH}(L)$. Here the gamma function $\Gamma(s)$ is defined for $s > 0$ by the integral $\Gamma(s) = \int_0^\infty t^{s-1} \cdot e^{-t} dt$.

γ -unique SVP. It is called unique SVP problem, if for a given lattice L which satisfies $\lambda_1(L) \ll \lambda_2(L)$, to find the shortest vector in L . And the γ -unique SVP problem is scaling the bound by a positive multiple as $\gamma\lambda_1(L) < \lambda_2(L)$. The auxiliary condition can be seen as a promised gap between the lengths of the first shortest vector and the second shortest vector. It is known that if the gap is bigger, it is easier to find the shortest vector by a certain algorithm. We abbreviate the γ -unique SVP to γ -uSVP in this paper.

2.2 The Learning With Errors Problem [18]

There are four parameters in LWE problem: the number of samples $m \in \mathbb{Z}$, the length $n \in \mathbb{Z}$ of secret vector, modulo $q \in \mathbb{Z}$ and the standard deviation $\sigma \in \mathbb{R}_{>0}$ for the discrete Gaussian distribution (denoted by D_σ) on \mathbb{Z} . Uniformly sample a matrix $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ and a secret vector $\mathbf{s} \in \mathbb{Z}_q^n$, and randomly sample a relatively small perturbation vector $\mathbf{e} \in \mathbb{Z}_q^m$ from Gaussian distribution D_σ . The LWE distribution Ψ is constructed by pairs $(\mathbf{A}, \mathbf{b} \equiv \mathbf{A}\mathbf{s} + \mathbf{e} \pmod{q}) \in (\mathbb{Z}_q^{m \times n}, \mathbb{Z}_q^m)$ sampled as above. The *search LWE problem* is for a given pair (\mathbf{A}, \mathbf{b}) sampled from LWE distribution Ψ , to compute the pair (\mathbf{s}, \mathbf{e}) .

2.2.1 Darmstadt LWE Challenge

In 2016, TU Darmstadt, in alliance with UC San Diego and TU Eindhoven published a platform for the concrete parameter analysis of LWE problem [5, 21]. In Darmstadt LWE Challenge, the organizers merge the two parameters σ and q into the *relative error size* α , such that $\alpha = \sigma/q$. n is the length of secret vector and q is the minimum prime number bigger than n^2 . For each case of length n , they offer the sampled n column vectors in basis $\mathbf{A}' \in \mathbb{Z}_q^{n^2 \times n}$, and one column target vector $\mathbf{b}' \in \mathbb{Z}_q^{n^2}$. The length n and the relative error size α are arithmetic sequences from 40 and 0.005, with common differences of 5 and 0.005 respectively. To adapt the current lattice algorithms used in the attack algorithm, we should randomly sample $m \ll n^2$ entries of the column vectors in the original basis $\mathbf{A}' \in \mathbb{Z}_q^{n^2 \times n}$ and ample from the target vector $\mathbf{b}' \in \mathbb{Z}_q^{n^2}$ respectively, as from $(\mathbf{A}', \mathbf{b}') \in (\mathbb{Z}_q^{n^2 \times n}, \mathbb{Z}_q^{n^2})$ in Darmstadt LWE Challenge to our instance $(\mathbf{A}, \mathbf{b}) \in (\mathbb{Z}_q^{m \times n}, \mathbb{Z}_q^m)$. This is called sublattice attack, and we will discuss how to choose a suitable m in Sect. 3.4.

2.2.2 Bounded Distance Decoding

In a Euclidean space spanned by a lattice L , there is a target vector $\mathbf{w} \in \mathbb{R}^m$ which is guaranteed to be within a distance $r \leq \alpha \lambda_1(L)$ where $\alpha > 0$. The bounded distance decoding (BDD) output a vector $\mathbf{b} \in L$ such that $\|\mathbf{w} - \mathbf{b}\| \leq r$.

2.2.3 q -ary Lattice

A lattice $L \subset \mathbb{Z}^m$ is a q -ary lattice if $q\mathbb{Z}^m \subset L$ for an integer q . Let $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ ($m > n$) be a matrix with column vectors, we define the following m -dimensional q -ary lattice.

$$L_{(\mathbf{A},q)} = \{\mathbf{y} \in \mathbb{Z}_q^m \mid \mathbf{y} \equiv \mathbf{A}\mathbf{x} \pmod{q} \text{ for some } \mathbf{x} \in \mathbb{Z}^n\}$$

It is the linear code generated by the columns of $\mathbf{A} \pmod{q}$ with $\text{vol}(L_{(\mathbf{A},q)}) \geq q^{m-n}$. $\text{vol}(L_{(\mathbf{A},q)}) = q^{m-n}$ when the columns of \mathbf{A} are linearly independent over \mathbb{Z}_q . We can construct the basis \mathbf{B} of q -ary $L_{(\mathbf{A},q)}$ as follows. $L = \{\mathbf{y} \in \mathbb{Z}_q^m \mid \mathbf{y} \equiv \mathbf{A}\mathbf{x} \pmod{q}, \mathbf{x} \in \mathbb{Z}^n\}$ as

$$\mathbf{B} = \begin{pmatrix} \mathbf{A}^T \\ q\mathbf{I}_m \end{pmatrix} \in \mathbb{Z}^{(m+n) \times m},$$

Then eliminate the linearly dependent vectors by an elementary transformation. In this work, we reduce this basis to a square matrix with Hermite Normal Form, see next paragraph.

2.2.4 Hermite Normal Form

The Hermite Normal Form (HNF) of a basis \mathbf{B} satisfies: (1) \mathbf{B} is lower triangular; (2) the diagonal entries are positive; (3) any entry below the diagonal is a non-negative number strictly less than the diagonal entry in its column. In this work, we use the HNF module in Victor Shoup’s NTL library [17], which uses the Domich et al.’s algorithm [7]. Particularly, a q -ary lattice $L_{(\mathbf{A},q)}$ has this form for some matrix $A'_{n \times (m-n)} \in \mathbb{Z}_q^{n \times (m-n)}$.

$$\mathbf{B}_{\text{HNF}} = \begin{pmatrix} q\mathbf{I}_{m-n} & \mathbf{0} \\ A'_{n \times (m-n)} & \mathbf{I}_n \end{pmatrix} \in \mathbb{Z}^{(m+n) \times m}.$$

2.3 BKZ Reduction Algorithms

The lattice reduction algorithms can make the given basis vectors “better”: relatively more orthogonal to each other with relatively smaller lengths than the given ones. Schnorr and Euchner [20] proposed the BKZ reduction algorithm, which processes the LLL reduction [12] and the enumeration algorithm iteratively with a fixed blocksize. Here the enumeration algorithm is an exhaustive point search algorithm. Refer to [20] for more details about enumeration. The root Hermite Factor of Schnorr and Euchner’s BKZ was considered limited by 1.01 according to Gama and Nguyen [8]. Chen and Nguyen improved the BKZ

algorithm called BKZ 2.0, by inviting “extreme pruning” enumeration in subroutine [6]. The root Hermite Factor of BKZ 2.0 break through the 1.01 limit with a reasonably big blocksize. In 2016, Aono et al. proposed a practical progressive BKZ algorithm [2]. The progressive BKZ algorithm invites some technique from BKZ 2.0. While the significant improvement is that they propose a sharp simulator based on the *Geometric Series Assumption* (GSA) [19], to estimate the runtime for a fixed blocksize β . Then the current local BKZ- β reduction is terminated after this runtime, and increase the blocksize to a simulated optimal larger one or just increase the blocksize step by step, until deriving the expected reduced basis. This progressive BKZ algorithm is shown about 50 times faster than BKZ 2.0 in [2]. Moreover, they also published their progressive BKZ source code in [3]. In this work, we will use the progressive BKZ algorithm to reduce the q -ary lattice bases in solving the LWE problem.

3 Overview of Embedding Technique for Solving LWE Problem

In this section, we recall Kannan’s embedding technique [10], and introduce the parameter settings in our experiments.

3.1 From LWE to BDD

The LWE problem can be reduced to BDD case as follows.

Input: a lattice $L = \{\mathbf{v} \in \mathbb{Z}_q^m \mid \mathbf{v} \equiv \mathbf{A}\mathbf{s} \pmod{q}, \mathbf{s} \in \mathbb{Z}_n\}$ and a target vector \mathbf{t} with bounded distance $\|\mathbf{e}\|$.

Output: a vector $\mathbf{v} \in L$ close to \mathbf{t} , and get \mathbf{s} from $\mathbf{v} \equiv \mathbf{A}\mathbf{s}$ if succeeded.

In 2016, Xu et al.’s group solved some instances of LWE Challenge by reducing LWE to BDD and using Liu-Nguyen’s adapted enumeration algorithm, which can solve BDD directly with a considerable success probability. In this work we focus on solving BDD by embedding technique: further reduce BDD to unique-SVP [10]. The embedding attack is shown in Algorithm 1. We will elaborate on the algorithm as follows.

3.2 Solving LWE via the Embedding Technique

Preprocessing. To solve a given LWE instance, it does not need to use all given samples. For instance, the Darmstadt LWE Challenge supplies the original basis $\mathbf{A}' \in \mathbb{Z}_q^{n^2 \times n}$ for each problem case, thus, a naive construction of matrices in Algorithm 1 requires a lattice reduction of a large number of matrices even for small LWE dimensions. Hence, we can choose m ($m \ll n^2$) vectors as a parameter to optimize the computational time, as from $(\mathbf{A}', \mathbf{b}') \in (\mathbb{Z}_q^{n^2 \times n}, \mathbb{Z}_q^{n^2})$ to $(\mathbf{A}, \mathbf{b}) \in (\mathbb{Z}_q^{m \times n}, \mathbb{Z}_q^m)$. We will discuss the way to compute the optimal m in Sect. 3.4. Also during the random sampling, we should check the independency of vectors to make sure: (1) the correctness of the attack algorithm; (2) the

Algorithm 1. Kannan’s embedding technique to solve LWE problem. [10]

Input: An LWE instance $(\mathbf{A}, \mathbf{b} \equiv \mathbf{A}\mathbf{s} + \mathbf{e} \pmod{q}) \in (\mathbb{Z}_q^{m \times n}, \mathbb{Z}_q^m)$.

Output: The secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ and the short error vector $\mathbf{e} \in \mathbb{Z}_q^m$, s.t. $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$.

Step 1. Construct the basis \mathbf{B} of q -ary lattice

$$L_{(\mathbf{A},q)} = \{\mathbf{v} \in \mathbb{Z}_q^m \mid \mathbf{v} \equiv \mathbf{A}\mathbf{x} \pmod{q}, \mathbf{x} \in \mathbb{Z}^n\}$$

as $\mathbf{B} = \begin{pmatrix} \mathbf{A}^T \\ \mathbf{I}_m \end{pmatrix} \in \mathbb{Z}^{(m+n) \times m}$; and compute the HNF of \mathbf{B} as

$$\mathbf{B}_{\text{HNF}} = \begin{pmatrix} q\mathbf{I}_{m-n} & \mathbf{0} \\ \mathbf{A}'_{n \times (m-n)} & \mathbf{I}_n \end{pmatrix} \in \mathbb{Z}^{m \times m};$$

Step 2. Reduce BDD to unique-SVP by rescaling \mathbf{B}_{HNF}

$$\text{to } \mathbf{B}' = \begin{pmatrix} \mathbf{B}_{\text{HNF}} & \mathbf{0} \\ \mathbf{b} & M \end{pmatrix} \in \mathbb{Z}^{(m+1) \times (m+1)};$$

Step 3. Process \mathbf{B}' using lattice algorithm to derive a short vector \mathbf{w} including the error vector \mathbf{e} ;

Step 4. Use \mathbf{e} to compute the secret vector \mathbf{s} by Gauss elimination in $(\mathbf{b} - \mathbf{e}) = \mathbf{A}\mathbf{s}$.

volume of derived q -ary lattice is q^{m-n} , which will be used in Sect. 3.4. We give explanations for each step in Algorithm 1.

Step 1. We follow the method in Sect. 2.2.3 to construct and compute the HNF basis \mathbf{B}_{HNF} of q -ary lattice $L_{(\mathbf{A},q)} = \{\mathbf{v} \in \mathbb{Z}^m \mid \mathbf{v} \equiv \mathbf{A}\mathbf{x} \pmod{q}, \mathbf{x} \in \mathbb{Z}^n\}$.

Step 2. This step is the key point of embedding technique: expand the q -ary basis $\mathbf{B}_{\text{HNF}} \in \mathbb{Z}^{m \times n}$ by one dimension, and embed the target vector \mathbf{b} and one embedding factor M into the new basis $\mathbf{B}' \in \mathbb{Z}^{(m+1) \times (m+1)}$.

Step 3. At this step, we process the new basis \mathbf{B}' by lattice algorithms. After the reduction, we get the error vector \mathbf{e} from the output shortest vector \mathbf{w} , since $\mathbf{e} = \mathbf{b} - \mathbf{B}\mathbf{u}$ and $\mathbf{w} = \mathbf{B}' \begin{pmatrix} \mathbf{u} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{e} \\ M \end{pmatrix}$ for some $\mathbf{u} \in \mathbb{Z}_q^m$. In our work, we use the progressive BKZ reduction in this step [2].

Step 4. Simply get the secret vector \mathbf{s} by Gauss elimination.

In the following, we explain four discussion points of the algorithm.

- (1) In the embedding procedure of Step 3, if the output vector \mathbf{w} of the lattice algorithm satisfies

$$\|\mathbf{w}\| \leq \sqrt{\|\mathbf{e}\|^2 + M^2} \approx \left(\frac{\sqrt{2m\sigma}}{(Mq^{m-n})^{1/(m+1)}} \right)^{1/(m+1)}, \quad (1)$$

here $\|\mathbf{e}\| \approx \sqrt{m}\sigma$, then the answer is correct with high probability.

- (2) There is a gap between the shortest vector and the linearly independent second shortest vector in $L'(\mathbf{B}')$, namely we have to solve a unique-SVP in this lattice. The size of embedding factor M can affect the gap in some sense and we will discuss it in Sect. 3.3.
- (3) Since we do not know the exact value of $\|\mathbf{e}\|$, we can not terminate by condition (1). $\|\mathbf{w}\| \leq \sqrt{\|\mathbf{e}\|^2 + M^2}$ is the condition for a reduction or point

searching algorithm to terminate in Step 4. However, during the update of lattice reduction of basis $(\mathbf{b}_1, \dots, \mathbf{b}_n)$ in our progressive BKZ, we found that the root Hermite factor δ suddenly drop to a very small value from value around 1. We can set the algorithm to terminate when $\delta < 0.7$ for convenience.

- (4) There is a trade-off between the attack efficiency and success rate, depending on the dimension m of $L_{(\mathbf{A}, q)}(\mathbf{A} \in \mathbb{Z}_q^{m \times n})$ and the embedding factor M of the sampled LWE instances in the embedding algorithm. In this work, our goal is from experiments to get a preliminary analysis of the affect of m and M on the runtime for solving Darmstadt LWE Challenge instances.

3.3 How to Choose M at Step 2

The size of $\|\mathbf{e}\|$ and M intuitively affect the gap of the shortest and the second shortest vector in the unique-SVP of $L(\mathbf{B}') \in \mathbb{Z}^{(m+1) \times (m+1)}$, since the reduction output is $\mathbf{w} = (\frac{\mathbf{e}}{M})$. For the entries of error vector \mathbf{e} are randomly and linearly independently sampled from the discrete Gaussian distribution D_σ , then $\|\mathbf{e}\|^2$ subject to $\sigma^2 \times \chi^2$, where χ means chi distribution. So $\|\mathbf{e}\|^2$ has expectation of $m\sigma^2$ and we can estimate $\|\mathbf{e}\| \approx \sqrt{m}\sigma$. Lyubashevsky and Micciancio [13] suggest that the choice for the embedding factor $M \in \mathbb{N}$ is $\|\mathbf{e}\|$. If M is bigger, then there is a lower chance to solve LWE problems, since the gap in unique-SVP will become smaller. However, if M is too small, there may exist a vector $\mathbf{v} \in L(\mathbf{B}')$ such that $\|\mathbf{v} + c \cdot (\frac{\mathbf{b}}{M})\| < \|\mathbf{w}\| = \|(\frac{\mathbf{e}}{M})\|$ where $c \in \mathbb{Z}$, according to [1]. In our experiments we observe the runtime of attack using increasing M from 1.

3.4 How to Choose m

In this part, we follow the analysis proposed by Micciancio and Regev [16]. With a small Gaussian standard deviation σ , the error vector \mathbf{e} is much shorter than the second shortest vector in the lattice L' , and the latter one can be assumed as the shortest vector in lattice L . According to the Gaussian heuristic, the length of the shortest vector in an m dimensional lattice is $\lambda_1(L) \approx \frac{(\Gamma(m/2+1)\text{vol}(L))^{1/m}}{\sqrt{\pi}}$, approximately $\sqrt{\frac{m}{2\pi e}} q^{(m-n)/m}$. So we can get $\lambda_2(L') \approx \lambda_1(L) \approx \sqrt{\frac{m}{2\pi e}} q^{(m-n)/m}$. In our experiments, we get the result that the attack is more efficient if the embedding factor M is closer to 1 (see Sect. 4.1). We set $M = 1$ here and assume $\lambda_1(L') \approx \|\frac{\mathbf{e}}{M}\| \approx \sqrt{m}\sigma$, for the Gaussian sampled \mathbf{e} has length around $\sqrt{m}\sigma$. So we want to enlarge the following gap in unique-SVP for an efficient attack:

$$\gamma(m) = \frac{\lambda_2(L')}{\lambda_1(L')} \approx \frac{\sqrt{\frac{m}{2\pi e}} q^{(m-n)/m}}{\sqrt{m}\sigma} \tag{2}$$

We need $\sigma \ll q^{\frac{m-n}{m}}$. What's more, for a lattice reduction algorithm with a root Hermite factor δ , the gap should satisfies $\gamma(m) > c\delta^m$ for a proper value c . The constant c is unknown, so we can maximize $q^{(m-n)/m}/\delta^m$, to get the optimal sub-dimension m of LWE sample instances is

$$m = \sqrt{n \log_2(q) / \log_2(\delta)}. \tag{3}$$

This can properly enlarge the gap in γ -unique SVP transformed from BDD, within a reduction algorithm’s capability estimated by the root Hermite factor $\delta = \text{rHF}(\mathbf{b}_1, \dots, \mathbf{b}_n) = (\|\mathbf{b}_1\|/\text{vol}(L)^{1/n})^{1/n}$.

4 Experimental Results and Analysis

In this section, we give the details in our experiments on solving LWE problems using embedding technique (Algorithm 1). All the cases are taken from Darmstadt LWE Challenge [21]. In our experiments, we just observe the hardness of small dimensions from 40 to 60, with the same $\alpha = 0.005$. As a preparing work, we take δ in the range $[1.010, 1.011, \dots, 1.025]$ and randomly sample $m = \sqrt{n \log_2(q)/\log_2 \delta}$ vector entries for $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$ in each LWE case. For each case with parameters (n, δ) , we sample 20 different bases. The progressive BKZ algorithm and its open source code of version 1.1 are used in the Step 3 of Algorithm 1. Our implementation using C language and NTL on Intel(R) Xeon(R) CPU E5-2697 v2 @ 2.70 GHz with 24 cores (over-clocked to 3.5 GHz and hyper-threaded to 48 threads). Xu et al. were using parallel implementation technique and the specifications of hardware are 3.60 GHz Intel Core i7 processor with eight cores and a cluster consisting of 20 c4.8xlarge instances, each equipped by 36 cores (hyper-threaded to 72 threads) [23]. The time unit in the following sections are all **single thread seconds**.

4.1 Efficiency by Changing M

As we discussed in Sect. 3.3, in the Step 2 of Algorithm 1, the embedding factor M in basis $\mathbf{B}' \in \mathbb{Z}^{(m+1) \times (m+1)}$ affect the size of gap in the unique-SVP of $L(\mathbf{B}')$. In this section we will observe what size of M is better for an efficient

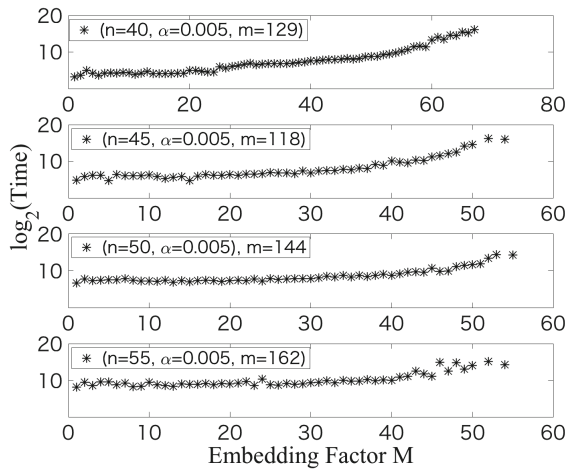


Fig. 1. Runtime for cases (n, α) with fixed bases and increasing embedding factor M .

embedding technique. The fixed dimension m of $L_{(\mathbf{A},q)}$ is referred to Sect. 4.2, and the embedding factor M is from 1 to around 55. For each case of parameters $n = 40, 45, 50, 55$ with fixed $\alpha = 0.005$, we sample a same basis $\mathbf{A} \in \mathbb{Z}^{m \times n}$ from Darmstadt LWE Challenge respectively. Figure 1 shows the runtime of Algorithm 1 for each case with increasing sequence of embedding factor M . We can observe that with growing M , the runtime of Algorithm 1 is gradually increasing. So it is more efficient to solve LWE problem with the embedding factor M closer to 1.

4.2 Optimal Choice of m for Each (n, α)

According to the Eqs. (2) and (3) in Sect. 3.4, the dimension m of q -ary lattice $L_{(\mathbf{A},q)}$ in Step 3 also affects the efficiency of Algorithm 1. A larger dimension m will lead the root Hermite Factor smaller, which makes the lattice algorithm inefficient. While a smaller m will reduce the gap of unique-SVP and make the problem harder to solve. In this section, we observe the affect of size m on the efficiency of Algorithm 1.

At first for each case of $(n, \alpha = 0.005)$, we fix the embedding factor as $M = 1$. We take δ in the range $[1.010, 1.011, \dots, 1.025]$ and for each δ calculate $m = \sqrt{n \log_2(q) / \log_2 \delta}$. We did the experiments for $n = 40, 45, 50, 55, 60, 65$. Note that for case of $(n = 40, \alpha = 0.005)$, since the runtime are close to each other, we ignore it here. We calculate the average runtime for each δ by around 20 random samples of $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$. In Table 1 the ‘‘Average BKZ Runtime’’ shows the minimum of average runtime for each δ . Further, the ‘‘Minimum BKZ Runtime’’ is the minimum data for the relevant δ and m .

From now we will analyze the experimental data in Table 1. We extrapolated the data by curve fitting technique in Fig. 2. The stars are the minimum of Average Runtime in each (n, α) cases as showed in Table 1. Here we get the quadratic function of the average runtime and n with three decimal precision:

$$\text{FittingLog}_2(\text{Average Runtime}) = 0.0153n^2 - 1.17n + 27.6, \tag{4}$$

and plot it in Fig. 2. And the fitting of optimal m and n is the linear function

$$\text{Optimal}(m) = \lceil 4.82n - 98.7 \rceil. \tag{5}$$

Table 1. Experimental runtime for each $(n, \alpha = 0.005)$ cases with parameter δ in range $[1.01, 1.011, \dots, 1.025]$.

(n, α)	δ	m	Average BKZ (\log_2 Runtime (sec))	Minimum BKZ (\log_2 Runtime (sec))
(45, 0.005)	1.025	118	5.99	4.28
(50, 0.005)	1.019	144	7.51	6.49
(55, 0.005)	1.017	162	9.03	8.08
(60, 0.005)	1.013	195	13.13	10.62
(65, 0.005)	1.012	213	16.04	14.65

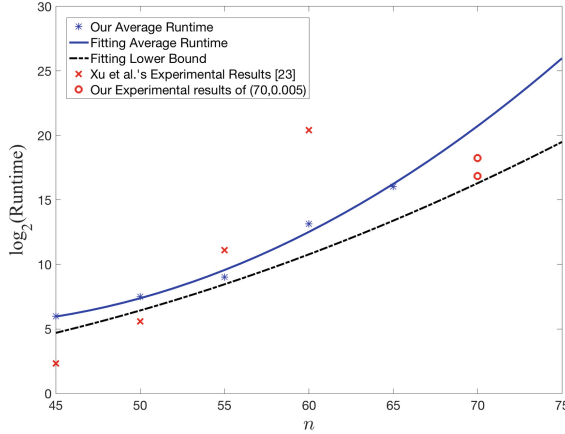


Fig. 2. The runtime for embedding technique on Darmstadt LWE Challenge of $(n, \alpha = 0.005)$ cases: the stars and the full curve denote our average and its fitting of the experimental runtime for the optimal m respectively; the dot line is fitting of the smallest runtime for each optimal m case, which is heuristically seen as the lower bound in our work; the red crosses are Xu et al.’s records at the LWE Challenge website; the hollow circles are our experimental results for $(n, \alpha) = (70, 0.005)$. (Color figure online)

Here the mark $\lceil \dots \rceil$ means taking a rounding number.

Moreover, we also illustrate the minimum runtime (seen as the lower bound heuristically) in all $(n, \alpha = 0.005)$ cases with parameter m from Eq. (4).

$$\text{FittingLog}_2(\text{Lower Bound}) = 0.00584n^2 - 0.208n + 2.21 \tag{6}$$

and we plot the fitting line in Fig. 2. Note that we take the quadratic formulas for the estimation in 4 and 6 since the state-of-the-art extreme pruning enumeration runs in $2^{O(n^2)-0.5n}$ as the subroutine of progressive BKZ.

Furthermore, in Table 2, we estimate the necessary dimension m and the relevant runtime by embedding technique on solving LWE Challenge cases $n \geq 75, \sigma = 0.005$, using progressive BKZ algorithm. Our estimation depending on the fitting function (4) and (5).

Moreover, from Fig. 2 we can see that Xu et al.’s LWE Challenge records of $\alpha = 0.005$ stopped at $n = 65$ for the overwhelming runtime and low success probability [22]. Our implemented embedding technique with progressive BKZ can solve the LWE Challenge instances more efficiently than Xu et al.’s enumeration implementation for $n \geq 55$.

For the cases of $(n = 70, \alpha = 0.005)$, we compute the extrapolated $m \approx 239$ (relevant $\delta = 1.010$) from function (5). Then we use $\delta = 1.010, 1.011, 1.012, 1.013$ and there are just two Darmstadt LWE Challenge cases with $\delta = 1.011, 1.012$ are successfully solved by $m = 233, 223$ in time $2^{16.8}, 2^{18.2}$ s respectively. and we plot it in Fig. 2, which are lying between the two fitting curves and close to the runtime of estimated $\text{FittingLog}_2(\text{Lower Bound})$.

Table 2. Estimation of effective m and runtime on solving ($n \geq 75, \sigma = 0.005$) in the LWE Challenge.

(n, α)	q	δ	m	Estimated BKZ (\log_2 Runtime (sec))
(75, 0.005)	5639	1.009	263	25.91
(80, 0.005)	6421	1.008	287	31.92
(85, 0.005)	7229	1.008	311	38.69
(90, 0.005)	8101	1.007	335	46.23
(95, 0.005)	9029	1.007	359	54.53

5 Conclusions

In this paper, we studied the algorithm to solve LWE problem using Kannan's embedding technique. Especially we randomly sampled LWE instances from Darmstadt LWE Challenge and applied the progressive BKZ algorithm to reduce the embedded bases. From our experiments of fixed relative error size $\alpha = \sigma/q = 0.005$, we observed that the algorithm has a more efficient trend if the embedding factor M is closer to 1. We also illustrated the relation of the dimension m of the q -ary lattice $L_{(\mathbf{A}, q)}$ in LWE instance, the length n of secret vector \mathbf{s} , and the runtime of the algorithm. Furthermore, Xu et al.'s LWE Challenge records of $\alpha = 0.005$ stopped at $n = 55$ for the overwhelming runtime, while our experimental results show that for $n \geq 55$, the embedding technique with progressive BKZ can solve the LWE Challenge instances more efficiently than Xu et al.'s implementation of Liu-Nguyen's enumeration algorithm. Finally our LWE Challenge record of $(n, \alpha) = (70, 0.005)$ cases succeeded in $2^{16.8}$ s (32.73 single core hours), which also lies in the bounds of our fitting curves.

Acknowledgment. This work was supported by JSPS KAKENHI Grant Number JP17J01987, JP26730069 and JST CREST Grant Number JPMJCR14D6, Japan.

References

1. Albrecht, M.R., Fitzpatrick, R., Göpfert, F.: On the efficacy of solving LWE by reduction to unique-SVP. In: Lee, H.-S., Han, D.-G. (eds.) ICISC 2013. LNCS, vol. 8565, pp. 293–310. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-12160-4_18
2. Aono, Y., Wang, Y., Hayashi, T., Takagi, T.: Improved progressive BKZ algorithms and their precise cost estimation by sharp simulator. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9665, pp. 789–819. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49890-3_30
3. The progressive BKZ code. <http://www2.nict.go.jp/security/pbkzcode/>
4. Babai, L.: On Lovász' lattice reduction and the nearest lattice point problem. In: Mehlhorn, K. (ed.) STACS 1985. LNCS, vol. 182, pp. 13–20. Springer, Heidelberg (1985). <https://doi.org/10.1007/BFb0023990>

5. Buchmann, J., Büscher, N., Göpfert, F., Katzenbeisser, S., Krämer, J., Micciancio, D., Siim, S., Vredendaal, C., Walter, M.: Creating cryptographic challenges using multi-party computation: the LWE challenge. In: AsiaPKC 2016, pp. 11–20 (2016)
6. Chen, Y., Nguyen, P.Q.: BKZ 2.0: better lattice security estimates. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 1–20. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_1
7. Domich, P., Kannan, R., Trotter, L.: Hermite normal form computation using modulo determinant arithmetic. *Math. Oper. Res.* **12**, 50–59 (1987)
8. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 31–51. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_3
9. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC 2008, pp. 197–206 (2008)
10. Kannan, R.: Minkowski's convex body theorem and integer programming. *Math. Oper. Res.* **12**(3), 415–440 (1987)
11. Kirchner, P., Fouque, P.-A.: An improved BKW algorithm for LWE with applications to cryptography and lattices. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 43–62. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_3
12. Lenstra, A.K., Lenstra Jr., H.W., Lovász, L.: Factoring polynomials with rational coefficients. *Math. Ann.* **261**(4), 515–534 (1982)
13. Lyubashevsky, V., Micciancio, D.: On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 577–594. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03356-8_34
14. Liu, M., Nguyen, P.Q.: Solving BDD by enumeration: an update. In: Dawson, E. (ed.) CT-RSA 2013. LNCS, vol. 7779, pp. 293–309. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36095-4_19
15. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. In: Kiayias, A. (ed.) CT-RSA 2011. LNCS, vol. 6558, pp. 319–339. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19074-2_21. Decoding Radius and DMT Optimality, ISIT2011, pp. 1106–1110 (2011)
16. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Bernstein, D.J., Buchmann, J., Dahmen, E. (eds.) Post-Quantum Cryptography 2009, pp. 147–191. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-88702-7_5
17. Victor Shoup's NTL library. <http://www.shoup.net/ntl/>
18. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC 2005, pp. 84–93 (2005)
19. Schnorr, C.P.: Lattice reduction by random sampling and birthday methods. In: Alt, H., Habib, M. (eds.) STACS 2003. LNCS, vol. 2607, pp. 145–156. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36494-3_14
20. Schnorr, C.P., Euchner, M.: Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Math. Program.* **66**, 181–199 (1994)
21. TU Darmstadt Learning With Errors Challenge. https://www.latticechallenge.org/lwe_challenge/challenge.php
22. Xu, R.: Private communication (2017)
23. Xu, R., Yeo, S.L., Fukushima, K., Takagi, T., Seo, H., Kiyomoto, S., Henricksen, M.: An experimental study of the BDD approach for the search LWE problem. In: Gollmann, D., Miyaji, A., Kikuchi, H. (eds.) ACNS 2017. LNCS, vol. 10355, pp. 253–272. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-61204-1_13