On the oldest reason why cryptographers should love lattice problems

Thales Bandiera Paiva
tpaiva@ime.usp.br

Instituto de Matemática e Estatística
Universidade de São Paulo

27 November 2017

Computational Number Theory
Professor Sinai Robins

Abstract. In 1994, Shor published two algorithms for quantum computers that solve efficiently two critical problems in modern cryptography: the discrete logarithm and integer factorization problems. With the engineering progress on building larger quantum computers, the main cryptographic schemes used today. Lattice-based cryptographic primitives, together with cryptographic schemes based on error correcting codes, hashes, elliptic curves isogenies, and multivariate equations, is one of the candidates for providing secure primitives against quantum computers. The history of the use of lattices in cryptography is rich and with some interesting twists. At first, their use for cryptography was though to be weak, since problems in integer lattices are usually related to knapsack problems, which are known to be broken. Then the famous LLL algorithm for lattice basis reduction became a standard tool in cryptanalysis, but cryptographic schemes based on lattices were out of question. After 10 years since the LLL publication, Ajtai showed in his breakthrough work how some lattice problems allow worst to average case hardness reduction. This property was used by Ajtai and Dwork to develop a public-key cryptographic scheme based on lattices. The scheme have not suffered major threats and remains secure, but has the major drawback of having too large keys, which can be of the order of gigabytes. This essay reviews some basic concepts about lattices, focusing on the importance of the worst to average case reduction of some lattice problems. It ends with a review of the reduction from worst to average case hardness discovered by Regev in 2007, which introduces some important tools used today for proving facts about lattices.

Keywords: Lattices, cryptography, worst-case to average-case

Table of Contents

1 Introduction

In 1994, Shor [27] published two algorithms for quantum computers that solve efficiently two critical problems in modern cryptography: the discrete logarithm and integer factorization problems. With the engineering progress on building larger quantum computers, the main cryptographic schemes used today, such as the RSA [26] and elliptic curves [23], become more vulnerable. This seeds the quest for finding cryptographic primitives that do not rely on the hardness of these problems, which is an active research area known as post-quantum cryptography [5]. The need for post-quantum secure standards is recognized by the National Institute of Standards and Technology (NIST), which is calling for proposals [7]. The main cryptographic schemes believed secure against quantum computers are based on lattices, error correcting codes, multivariate equations, hashes, and isogenies among elliptic curves [5]. It is important to note that there are no proofs that these schemes are secure against quantum computers, there are only some justifications for this belief that are similar to the ones for the $\mathcal{P} \neq \mathcal{NP}$ conjecture.

Lattice-based cryptography uses the hardness of some lattice problems to build cryptographic primitives, such as hash functions and one-way trapdoor functions. A lattice is a discrete set of points in $\mathbb{R}^n$ with a periodic structure, which is closed under subtraction of elements. The closure under subtraction of elements is captured by a construction similar to that of linear subspaces, with basis vectors and linear combinations of them. The periodic nature is obtained by considering only integer linear combinations of the basis vectors. Figure 1 illustrates a lattice generated by integer linear combinations of the vectors $\mathbf{b}_1$ and $\mathbf{b}_2$, which are highlated in the figure.

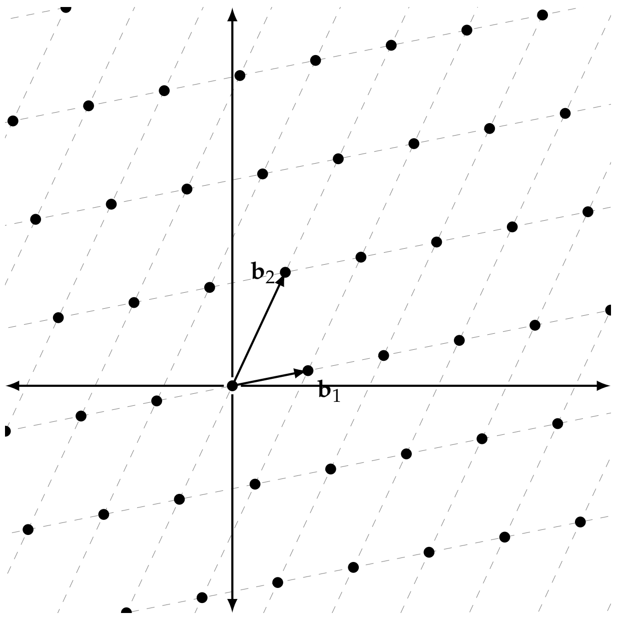

Fig. 1. A lattice in $\mathbb{R}^2$ with basis $\{\mathbf{b}_1, \mathbf{b}_2\}$.

The history of the use of lattices in cryptography is rich and with some interesting twists. At first, their use for cryptography was though to be weak, since problems in integer lattices are usually related to knapsack problems [20], which are known to be broken [1, 25]. Then the famous LLL algorithm [17] for lattice basis reduction became a standard tool in cryptanalysis, but cryptographic schemes based on lattices were out of question. After 10 years since the LLL publication, Ajtai showed in his breakthrough work [2] how some lattice problems allow worst-case to average-case hardness reduction. This property was used by Ajtai and Dwork to develop a public-key cryptographic scheme based on lattices [3]. This was the first cryptographic scheme to have its security based on a worst-case to average-case reduction, but has the major drawback of having too large keys, which can be of the order of gigabytes.

For dealing with the huge sizes of the Ajtai-Dwork construction, some alternatives have been proposed, such as the NTRU [15] and the GGH [14]. Even though there are no known attacks that break the GGH cryptosystem asymptotically, this scheme was broken for reasonable parameters [24] and is considered insecure. In contrast, the NTRU cryptosystem remains secure today, and its variant proposed by Stehlé and Steinfield [28] is considered a promising candidate for a standard in post-quantum secure cryptography [7]. Lattice-based cryptosystems also are the first to support fully homomorphic cryptography [13], which is an important and very active research area.

2 Public-key cryptography

2.1 Secure functions

A family of functions is a set $\{f_k : X \to Y : k \in K\}$. The set K is called the key space, X is called the domain, and Y is called the range of the family. Note that X and Y do not depend on the key k. If $|X| > |Y|$, the functions f_k compress the input, and the functions in this family are called hash functions.

We show next two basic security notions for function families.

Definition 1 (One-way functions). *A family $\{f_k\}$ is one way if, for any $k \in K$, it is infeasible to find some preimage of $f_k(x)$ for some randomly chosen $x \in X$.*

Definition 2 (Collision resistance). *A family $\{f_k\}$ is collision resistant if it is infeasible to find a collision, that is, $x_1 \neq x_2 \in X$ such that $f_k(x_1) = f_k(x_2)$, when the key k is randomly chosen from K.*

In cryptography, the hardness of finding collisions is described as a function of the number of bits of the key. A possible definition for "infeasible" can then be that the expected number of operations to find a collision is proportional to $2^{\log_2 k}$.

It is not known whether one-way functions and collision resistant functions exist or not. The main arguments for proving some function is collision resistant is based on harness reductions.

2.2 Encryption schemes

Cryptography studies how to establish secure communication in a medium shared with malicious adversaries. One can define several security objectives for communication, and the main ones aimed by public-key cryptographic constructions are the following.

1. Confidentiality: only the intended recipient of a message must be allowed to read it.
2. Authentication: one or more recipients of a message must be able to verify that the message was generated by a trusted sender.
3. Integrity: one or more recipients of a message must be able to detect if the message was tampered during transmission.
4. Non-repudiation: the real sender of a message cannot deny authorship of a message.

Confidentiality of communication is achieved by encryption schemes, and the other three objectives are usually obtained by signature schemes. In this work, we are mainly interested in encryption schemes.

Every encryption schemes uses secret information that must be combined to the plaintext to encrypt it, or to the ciphertext to decrypt it. These secret information are called the keys. There are two types of encryption schemes: secret-key and public-key schemes. Using secret-key schemes, the sender ant the recipient must share the same key, which is used for encrypting and decrypting. Therefore, these schemes have the disadvantage of requiring that sender and recipient share, at least once, have access to a secure communication channel for deciding a secret key, unless some public-key cryptographic primitives are used, such as key-exchanging protocols [10]. Despite being conceptually older than public-key schemes, there are several sophisticated secret-key schemes. Some of the main ones are the DES [9] and the AES [8].

In public-key encryption schemes, each user has two keys, one secret, which the user keeps to himself, and on public, which the user shares with anyone who wants to send him private messages. These keys are mathematically related in such a way that it is computationally infeasible to recover the secret key from the public key. To send a message Alice a message, we use her public key to encrypt the message, and Alice uses her private key to decrypt the message. These schemes were envisioned by Diffie and Hellman [10], who showed, supposing the difficulty on average of the discrete log problem, how users of a network could decide on a secret key to be used, for example, with the AES. The main encryption scheme are the RSA by Rivest, Shamir, e Adleman [26], which was shortly after Diffie and Hellman protocol, and elliptic curves [23]. A formal definition of an ecryption scheme is given next.

Definition 3 (Encryption scheme [11]). *Let λ be a positive integer. Consider the following spaces, all dependent on λ. $\mathcal{M}_\lambda$ is the set of possible messages. $\mathcal{PK}_\lambda$ is the set of possible public keys. $\mathcal{SK}_\lambda$ is the set of possible secret keys. $\mathcal{C}_\lambda$ is the set of possible ciphertexts. We say that the triple of algorithms*

$$(\textsc{KeyGen}, \textsc{Encrypt}, \textsc{Decrypt})$$

is a public-key encryption scheme over the spaces $\mathcal{PK}_\lambda, \mathcal{SK}_\lambda, \mathcal{C}_\lambda$ if they are all of expected polynomial time in λ, and the following conditions hold.

1. *The key generating algorithm* KeyGen *is a probabilistic algorithm that takes λ as input, and outputs K_{SEC} and K_{PUB}, such that $K_{\text{SEC}} \in \mathcal{SK}_\lambda$ and $K_{\text{PUB}} \in \mathcal{PK}_\lambda$.*
2. *The encrypting algorithm* Encrypt *is a probabilistic algorithm that takes $\mathbf{m} \in \mathcal{M}_\lambda$ and $K_{\text{PUB}} \in \mathcal{PK}_\lambda$ as inputs, and outputs $\mathbf{c} \in \mathcal{C}_\lambda$.*
3. *The decrypting* Decrypt *is an algorithm that takes $\mathbf{c} \in \mathcal{C}_\lambda$ and $K_{\text{SEC}} \in \mathcal{SK}_\lambda$ as inputs, and ouputs $\mathbf{m} \in \mathcal{M}_\lambda$, or the symbol $\perp$. The symbol $\perp$ means that a decryption error occurred, possibly because of an invalid input.*
4. *The three algorithms are related in the following way. If K_{PUB} and K_{SEC} are a valid output of* KeyGen(λ) *for some λ, then*

$$\textsc{Decrypt}(\textsc{Encrypt}(\mathbf{m}, K_{\text{PUB}}), K_{\text{SEC}}) = \mathbf{m},$$

for all, except possibly for a negligible portion, $\mathbf{m} \in \mathcal{M}$*.*

The algorithms and spaces must be defined in such a way that the best known algorithm for the attack takes more than 2^{λ_0} *steps to successfully attack the scheme when the scheme is instantiated with* $\lambda = \lambda_0$*. For this reason,* λ *is known as the security parameter of the scheme.*

3 The mathematics of lattices

3.1 Fundamentals

Lattices are discrete subspaces of $\mathbb{R}^n$ that are closed under subtraction of elements.

Definition 4 (Lattice). *Let* $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ *be a set of n independent vectors in* $\mathbb{R}^m$*. The lattice in* $\mathbb{R}^m$ *generated by this set of vectors is the set*

$$\mathcal{L}(\mathbf{b}_1, \dots, \mathbf{b}_n) = \left\{ \sum_{i=1}^{n} x_i \mathbf{b}_i : x_i \in \mathbb{Z} \right\}.$$

If we join the vectors $\mathbf{b}_1, \dots, \mathbf{b}_n$ *in a matrix as* $\mathbf{B} = [\mathbf{b}_1 | \dots | \mathbf{b}_n]$*, we can write more compactly*

$$\mathcal{L}(\mathbf{B}) = \{\mathbf{Bx} : \mathbf{x} \in \mathbb{Z}\}.$$

The set of vectors $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ *is called a basis of the lattice. Similarly, the matrix* $\mathbf{B}$ *is called a basis matrix of the lattice. The integers n and m are called the rank and dimension of the lattice, respectively.*

In this paper we are interested in cryptography, which assumes that the basis matrices of a lattice can be efficiently represented by finite precision machine. Because of their finite precision, these machines cannot deal with the real numbers and effectively deal with integers. As such, from now on we assume that all basis matrices have integral coefficients.

Each set of independent vectors of a lattice $\mathcal{L}$ clearly generate the same linear space, which is denoted by span($\mathcal{L}$). But not all set of independent vectors in $\mathcal{L}$ generate the $\mathcal{L}$. In general, a set of independent vectors of $\mathcal{L}$ generate a sublattice of $\mathcal{L}$, that is, a lattice $\mathcal{L}' \subseteq \mathcal{L}$. It is not difficult to prove that that two matrices $\mathbf{B}$ and $\mathbf{B}'$ generate the same lattice if and only if

$$\mathbf{B} = \mathbf{B}'\mathbf{U},$$

for some is a unimodular[1] matrix $\mathbf{U}$.

For some of the algorithms on lattices, it is useful to define equivalence classes for vectors in the space span($\mathcal{L}$) with respect to the periodic structure of $\mathcal{L}$. For that, we define define the half open parallelepiped as the set

$$P(\mathbf{B}) = \{\mathbf{Bx} : x_i \in [0, 1)^n\}.$$

[1] An integer matrix $\mathbf{U}$ such that $\det(\mathbf{U}) = \pm 1$.

This set contains, for a given basis matrix, an n-dimensional parallelepiped which *partitions* the space span$(\mathcal{L})$. That is, it can cover the linear space generated by $\mathbf{B}$ without overlays. The only point in $P(\mathbf{B})$ that is in the lattice is the origin. To reduce a vector point $\mathbf{v}$ in span$(\mathcal{L})$ to a point in $P(\mathbf{B})$, one first write v as a linear combination of the basis, that is, solve for $\mathbf{Bx} = \mathbf{v}$. Then, the point in the $P(\mathbf{B})$ corresponding to to $\mathbf{v}$ is $\mathbf{By}$, where $y_i = x_i - \lfloor x_i \rfloor$ for each $1 \leq i \leq n$. We say $\mathbf{v}$ was reduced to $\mathbf{y}$ modulo $P(\mathbf{B})$. Figure 2 shows a fundamental parallelepiped and a reduction from a point in span$(\mathcal{L})$ to the respective point inside the fundamental parallelepiped.

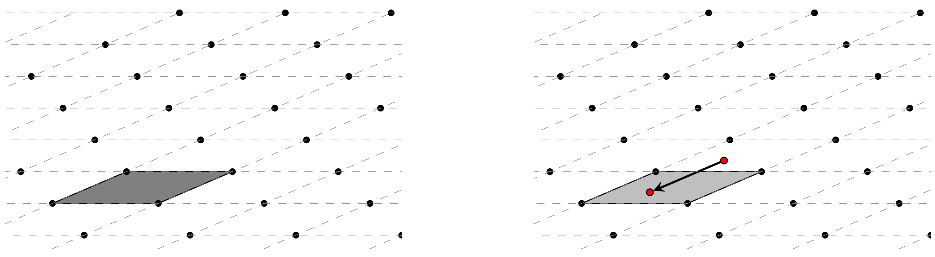
Fig. 2. On the left, a fundamental parallelepiped for the lattice. On the right, a reduction of a point modulo the fundamental parallelepiped.

One important quantity associated to a lattice and invariant with respect to the choice of basis is its determinant, denoted det$(\mathcal{L}(\mathbf{B}))$. Its importance comes from the fact that it is used to bound some other important quantities of the lattice, and it can be efficiently computed. For any fixed basis $\mathbf{B}$, det$(\mathcal{L}(\mathbf{B}))$ is the volume of its fundamental parallelepiped. It can be computed as

$$\det(\mathcal{L}(\mathbf{B})) = \sqrt{\det(\mathbf{B}^T\mathbf{B})}.$$

Using this expression and the fact that any basis can be written as the product of $\mathbf{B}$ and a unimodular matrix, one can see that the determinant is in fact invariant with respect to the basis.

Lattice bases are usually classified as good or bad. This classification is based on the algorithmic use of a basis, that is, a good basis is one which can be used to efficiently solve some lattice problem. In contrast, a bad basis is one which no known algorithm can use to efficiently solve a problem. In general, good bases consist of small and almost orthogonal vectors, while bad bases consist of long and close vectors.

This motivates the definition of the series of numbers $\lambda_1, \ldots, \lambda_n$, where each λ_i is the least radius of an m-dimensional ball which contains at least i linear independent vectors in the a fixed lattice with rank n and dimension n. The successive minima of a lattice can be defined using any norm, but we are mainly interested in the euclidean norm. The definition of the successive minima might mislead the reader into thinking that the set of independent vectors $\{\mathbf{v}_1, \ldots, \mathbf{v}_1\}$, where $\|\mathbf{v}_i\| = \lambda_i$ for each $\mathbf{v}_i$, is a basis of the lattice, but this is not always the case [21]. To answer how large can the successive minima be, a result by Minkowski can be used. Note that this bound is independent of the basis, which makes it more useful.

Theorem 5 (Minkowski's theorem [21]). *Let $\lambda_1, \ldots, \lambda_n$ be the successive minima of some rank n lattice $\mathcal{L}(\mathbf{B})$. Then*

$$\left(\prod_{i=1}^{n} \lambda_i\right)^{1/n} < \sqrt{n}\det(\mathcal{L}(\mathbf{B}))^{1/n}.$$

3.2 Some lattice problems

There are many interesting problems defined over lattices. Some of the most well known are the following.

1. The shortest vector problem (SVP) is: given a basis $\mathbf{B}$ of a lattice, find the shortest nonzero vector in $\mathcal{L}(\mathbf{B})$.
2. The closest vector problem (CVP) is: given a basis $\mathbf{B}$ of a lattice and a target vector $\mathbf{t}$, find the closest lattice point to $\mathbf{t}$ in $\mathcal{L}(\mathbf{B})$.
3. The shortest independent vectors problem vector problem (SIVP) is: given a basis $\mathbf{B}$ of a lattice, find a set S of n linear independent vectors in $\mathcal{L}(\mathbf{B})$ such that, for all $\mathbf{v} \in S$, it holds that $\|\mathbf{v}\| \leq \lambda_n$.

The first two problems are illustrated in Figure 3.

The SVP is $\mathcal{NP}$-hard for randomized reductions, and it can be shown to be not harder than the CVP. Both the CVP and the SIVP are $\mathcal{NP}$-hard. There a simple reduction from the CVP to the subset sum problem, which is known to be $\mathcal{NP}$-hard. The subset sum problem is: given a set A of n integers and an integer target s, find a subset of A that adds up to s.

Next we show that the CVP is $\mathcal{NP}$-complete. There are two reasons why this proof is shown here. The first one is that it is simple and can be used to warm up the reader for the more sophisticated worst to average case reduction shown in the next section. The second is that it is related to an algorithm presented by Lagarias and Odlyzko [16]. to solve the subset sum using the LLL for lattice basis reduction, which was one important step to the use of lattice algorithms in cryptanalysis.

For the reduction we consider the decisional CVP is: given a basis $\mathbf{B}$ of a lattice, a target vector $\mathbf{t}$ possibly outside $\mathcal{L}(\mathbf{B})$, and a distance d, decide if there exists find a point in the lattice $\mathcal{L}(\mathbf{B})$ within distance d from $z\mathbf{t}$.

Theorem 6. *The decisional CVP is $\mathcal{NP}$-complete.*

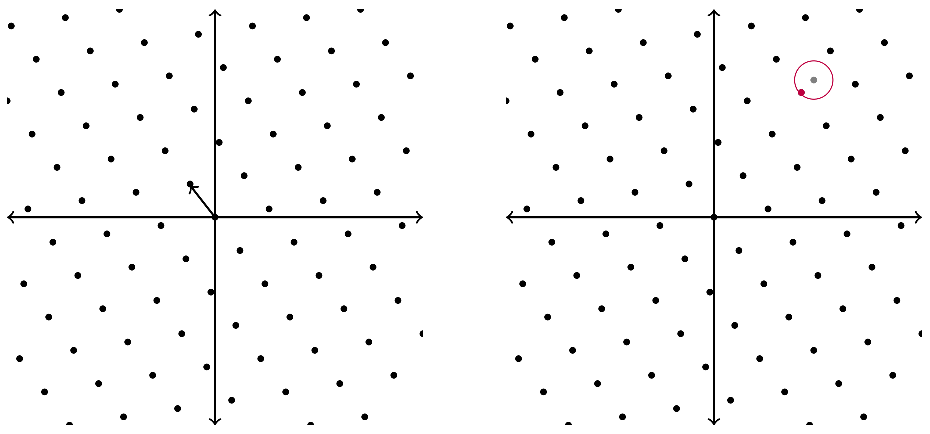

Fig. 3. The figure on the left shows one of the shortest vectors of the lattice. The figure on the right shows the lattice point that is the closest to a fixed point outside the lattice.

Proof [21]. We reduce the decisional subset sum problem to the decisional CPV. Let the set A and a the integer target s be an instance of the subset sum problem. Build the following target vector

$$\mathbf{t} = [s\,1\,1\,\ldots\,1],$$

which is an encoding of the target s. Now build the matrix

$$\mathbf{B} = \begin{bmatrix} a_1 & a_2 & \ldots & a_n \\ 2 & 0 & \ldots & 0 \\ 0 & 2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \ldots & \ldots & 2 \end{bmatrix},$$

which encodes the elements in A. Later that these 2 entries in matrix **B** Set the target distance to be $d = \sqrt{n}$.

We now show that this reduction maps YES (NO) instances of the decisional subset sum problem are mapped to YES (resp. NO) instances of the decisional CVP.

Suppose (A, s) is a YES instance of the subset sum problem, that is, there exists a subset of A which adds up to s. Then there exists a vector $\mathbf{v} = [v_1, \ldots, v_n] \in 0, 1^n$ such that

$$\sum_{i=1}^{n} v_i a_i = s.$$

We claim that the vector $\mathbf{Bv} = [\sum_{i=1} v_i a_i \quad 2v_1 \quad \dots \quad 2v_n]^T$ is a point in the lattice close to $\mathbf{t}$. Consider the norm of their difference

$$\|\mathbf{Bv} - \mathbf{t}\| = \sqrt{\left(\sum_{i=1} v_i a_i - s\right)^2 + \sum_{i=1}^{n} (2v_i - 1)^2} = \sqrt{0 + \sum_{i=1}^{n} (2v_i - 1)^2}.$$

Since $v_i = 0$ or 1, then $2v_i - 1 = \pm 1$, and we get $\|\mathbf{Bv} - \mathbf{t}\| = \sqrt{\sum_{i=1}^{n} 1} = \sqrt{n}$, which equals to the target distance d. Therefore $(\mathbf{B}, \mathbf{t}, d)$ is a YES instance for CVP.

Now suppose $(\mathbf{B}, \mathbf{t}, d)$ is a YES instance. We show that the associated subset sum problem must be also a YES instance. By hypothesis, there is a vector $\mathbf{v}$ such that $\|\mathbf{Bv} - \mathbf{t}\| \leq d$. That is

$$\sqrt{\left(\sum_{i=1} v_i a_i - s\right)^2 + \sum_{i=1}^{n} (2v_i - 1)^2} \leq \sqrt{n}. \qquad (1)$$

But $(2v_i - 1)^2 \geq n$, because $2_v i - 1 = \pm 1$. Therefore, the only possibility satisfying Equation 1 is $\sum_{i=1} v_i a_i - s = 0$, making (A, s) a YES instance to the decisional subset sum problem, with one possible a certificate given by $\mathbf{v}$. □

In cryptography we are particularly interested in the approximation version of these problems, described next.

1. The approximate shortest vector problem (γ-SVP) is: given a basis $\mathbf{B}$ of a lattice, find a nonzero vector $\mathbf{v}$ in $\mathcal{L}(\mathbf{B})$ such that $\|\mathbf{v}\| \leq \gamma \lambda_1$.
2. The approximate closest vector problem (γ-CVP) is: given a basis $\mathbf{B}$ of a lattice and a target vector $\mathbf{t}$, find a vector $\mathbf{v}$ such that $\|\mathbf{v} - \mathbf{t}\| \leq \|\mathbf{w} - \mathbf{t}\|$, where $\mathbf{w}$ is the closest lattice point to $\mathbf{t}$.
3. The approximate shortest independent vectors problem vector problem (γ-SIVP) is: given a basis $\mathbf{B}$ of a lattice, find a set S of n linear independent vectors in $\mathcal{L}(\mathbf{B})$ such that, for all $\mathbf{v} \in S$, it holds that $\|\mathbf{v}\| \leq \gamma \lambda_n$.

These problems are believed to be intractable for polynomial approximation factors γ. The best known algorithms either run efficiently but have exponential approximation factors, or solve for polynomial approximation factors in exponential running time. But interestingly, these problems are not $\mathcal{NP}$-hard for the approximation factors considered for cryptography.

3.3 A lattice-based cryptosystem: GGH

The GGH [14] public-key cryptosystem was one of the first lattice-based encryption schemes. It is analogous to the McEliece scheme [19] for lattices. The scheme is based on the closet vector problem, while McEliece's scheme is based on the similar problem of correcting errors on a linear code. Both of these problems are known to be $\mathcal{NP}$-hard.

The private key is a good almost orthogonal basis $\mathbf{B}_s$ of some lattice $\mathcal{L}$ which allows its holder to efficiently and reliably solve the closest vector problem. The public key is a bad basis $\mathbf{B}_p$ of $\mathcal{L}$ which allows anyone to generate points in the lattice.

Let Alice be the public key holder. To encrypt a message represented by the vector $\mathbf{m}$, we perform

$$\mathbf{c} = \mathbf{B}_p\mathbf{m} + \mathbf{e},$$

where $\mathbf{e}$ is a small error vector and $\mathbf{c}$ is the ciphertext. Note that $\mathbf{c}$ is not a point in the lattice, with overwhelming probability, and. Further, if the $\mathbf{e}$ is small enough, the closest point to $\mathbf{c}$ is $\mathbf{B}_p\mathbf{m}$, from which Alice can recover $\mathbf{m}$.

To find the closest lattice point to $\mathbf{c}$, the authors propose the use of Babai's round-off algorithm [4]. This algorithm writes $\mathbf{c}$ as a linear combination $\mathbf{y}$ of the rows of the secret matrix $\mathbf{B}_s$, that is $\mathbf{B}_s\mathbf{y} = \mathbf{c}$. Then each of the real values in $\mathbf{y}$ is rounded to its nearest integer obtaining a vector denoted by $\lfloor \mathbf{y} \rceil$, as an approximation for the closest vector to $\mathbf{c}$. By solving $\mathbf{B}_s\mathbf{m} = \lfloor \mathbf{y} \rceil$ for $\mathbf{m}$, the message is recovered.

The authors had to deal with two contradicting objectives:

1. the entries of the error vector $\mathbf{e}$ should be large enough so that it is difficult for an attacker to find the closest vector to $\mathbf{c}$;
2. the entries of the error vector $\mathbf{e}$ should be short enough so that Babai's algorithm finds the closes vector to $\mathbf{c}$.

The authors propose to sample each entry of the vector $\mathbf{e}$ uniformly at random from the set $\{-\sigma, +\sigma\}$, for some real value parameter σ. To set this parameter, they prove a theorem that upper bounds the probability of failing to decrypt a message, which is stated next. One then can use this theorem to find how large should σ be so that the one can decrypt successfully with high probability.

Theorem 7. *Let $\mathbf{B}_s$ be the secret basis matrix, and let $\gamma/\sqrt{n}$ be the maximum norm $\|\cdot\|_\infty$ of the rows in $\mathbf{B}_s^{-1}$. Then the probability of decryption errors is upper bounded by*

$$\Pr(\textit{Decryption error}) \le 2n \exp\left(-\frac{1}{8\sigma^2\gamma^2}\right).$$

Unfortunately, shortly after the GGH scheme was proposed, Nguyen showed how to break this scheme for reasonable parameters [24]. The attack is based on the fact the decoding problem associated with the scheme can be reduced to a closest vector problem much easier than the instances of the general problem.

4 Worst-case to average-case reduction

4.1 How to believe on the hardness of a problem

Possibly the main objective in computer science is to study how to solve problems algorithmically. It is known for a long time that there are problems which cannot be solved algorithmically, but we want to give bounds on the hardness of the ones that can be. This is one of the fundamental problems in complexity theory.

The vast majority of the classes of problems studied in complexity theory are defined in terms of the worst case complexity, that is, the complexity of the best possible algorithm for one problem running on the hardest instances of the problem. This is quite useful if we want conservative estimates on what problems we can expect to solve. But when doing cryptography, it is very important to be able to consider not the worst case hardness of a problem, but the hardness of the problem in typical instances. For example, when designing a public-key encryption scheme, one wants to prove that, to recover the secret key from the public key, any possible attacker would have to perform a very large number of operations. These kind of results offering a lower bound on the hardness of a problem are very scarce and difficult to obtain. We can point two different approaches to study the average-case hardness of a problem, which are discussed next.

The first one is average-case complexity theory [18], which consider not only the problems but also the distribution on the set of its instances. To reduce a problem A to another problem B, one is only allowed to map instances of A to instances in B when the probability associated with these instances are somewhat similar. The theory elegant and potentially very useful, but it is not clear how to use it to prove for natural problems together with natural distributions over their instances [6].

The second one is the random self-reducibility of problems. Intuitively, a problem is random self-reducible if, using an oracle that solves random instances of the problem, one can use this oracle to solve any instance of the problem. For example, consider the case of inverting the RSA function. Suppose we have access to an oracle which inverts the RSA function for a small fraction ϵ of the possible ciphertexts. Now suppose we are given a ciphertext $c = m^{K_{\text{PUB}}} \bmod N$, where K_{PUB} is the recipient's public key, N is the product of two large primes, and m is a secret number. We can generate around $1/\epsilon$ random messages $\overline{m}$, creating random-looking ciphertexts $c' = m^{K_{\text{PUB}}}\overline{m}^{K_{\text{PUB}}} = (m\overline{m})^{K_{\text{PUB}}} \bmod N$ and ask our oracle to invert them. If ϵ is not negligible, with high probability we will eventually obtain a valid decryption for one c', that is, the value of $m\overline{m} \bmod N$, from which we can recover m. This indeed shows that **for a fixed** N, the average-case hardness of inverting the RSA function is the same as the worst-case hardness.

The main and critical problem of of this random self-reductions for the RSA function is that it tells nothing on how to choose an N that give us hard instance, and each possible N cannot be used by more than one person. The difference between the RSA worst-case to average-case reduction and The idea behind Ajtai's worst-case to average-reduction for lattice problems is similar to the reduction showed for the RSA function. The important difference is that, every natural number n, it shows an average-case problem that, if it can be solved efficiently, then a collection of lattice problems believed to be hard can be solved efficiently for **all** lattices of dimension n. The dimension n for lattices can be seen, doing an analogy with the RSA, as the approximate number of bits of the two prime numbers that form N. The RSA key generation algorithm has the problem that there are a lot of n bits numbers, even some primes, that are not suitable for use in the RSA, and have to be discarded. But for lattices, the reduction is valid for all n-dimensional lattices, and the number of possible lattices, which usually correspond to keys, grow fast with respect to their dimension n.

4.2 The smoothing lemma

Many results in reductions between lattice problems use sampling from discrete Gaussian distributions over lattices. To define this distribution, we first introduce Gaussian functions.

Definition 8 (Gaussian function ρ)**.** *Let n be a positive integer and real $s > 0$. The Gaussian function $\rho_s : \mathbb{R}^n \to \mathbb{R}^+$ of width s is*

$$\rho_s(\mathbf{x}) = \exp\left(\frac{-\pi\|\mathbf{x}\|^2}{s^2}\right).$$

Definition 9 (Discrete Gaussian distribution $\mathcal{N}_{\mathbf{c},s}$)**.** *Let $\mathcal{L}$ be a lattice,* $\mathbf{c}$ *be point in $\mathcal{L}$ and s a positive real denoting the width of a Gaussian function. The discrete Gaussian probability distribution $\mathcal{N}_{\mathbf{c},s}$ is a probability distribution over the points in $\mathcal{L}$ such that*

$$\mathcal{N}_{\mathbf{c},s}(\mathbf{x}-\mathbf{c}) \propto \begin{cases} \rho_s(\mathbf{x}-\mathbf{c}) \text{ if } \mathbf{x} \in \mathcal{L}, \\ 0, \text{ otherwise.} \end{cases}$$

The proportionality constant is chosen to guarantee that $\mathcal{N}_{\mathbf{c},s}$ is a probability distribution, that is, the sum of all masses is equal to 1. The distribution depends on the lattice $\mathcal{L}$, but we omit this dependency because usually the lattice in question should be clear from the context.

The more general definition defines the probability distribution using lattice cosets, which are nothing more than shifts of a lattice by some offset vector. The presented definition is slightly simpler, only because we do not need the more general one in this paper.

To compare distributions, the following measure of statistical distance is considered.

Definition 10. *Let X and Y be two discrete random variables over a countable set A. The statistical distance between X and Y is defined as*

$$\Delta(X,Y) = \frac{1}{2}\sum_{a\in A} |\Pr(X=1) - \Pr(Y=a)|.$$

The next definition introduces the smoothing parameter η_ϵ. Its name comes from the fact that when the standard deviation of the Gaussian is around η_ϵ, the points points sampled from the Gaussian, when reduced modulo the fundamental parallelepiped $P(\mathbf{B})$, are almost uniformly distributed over the $P(\mathbf{B})$. We usually want ϵ to be a negligible function of n, e.g. 2^{-n}.

Definition 11 (Smoothing parameter). *Let $s > 0$ and $\mathbf{c} \in \mathbb{R}^n$ be the scale factor and the center of the Gaussian distribution $\mathcal{N}_{s,\mathbf{c}}$, respectively. Consider the lattice $\mathcal{L}(\mathbf{B})$, and its fundamental parallelepiped $P(\mathbf{B})$. Then smoothing parameter of the lattice is defined as*

$$\eta_\epsilon = \inf\left\{s : \Delta\left(\mathcal{N}_{s,\mathbf{c}} \bmod P(\mathbf{B}), U(P(\mathbf{B}))\right) \leq \epsilon/2\right\}.$$

From this definition only, it is not clear if the smoothing parameter is well defined. The next lemma gives an upper bound on the smoothing parameter, which is related to λ_n, the n-th successive minima. This lemma is illustrated by Figure 4.

Lemma 12. *For any n-dimensional lattice $\mathcal{L}$ and positive real $\epsilon > 0$, the smoothing parameter η_ϵ is upper bounded by*

$$\eta_\epsilon \leq \lambda_n(\mathcal{L})\sqrt{\frac{\ln(2n(1+1/\epsilon))}{\pi}}.$$

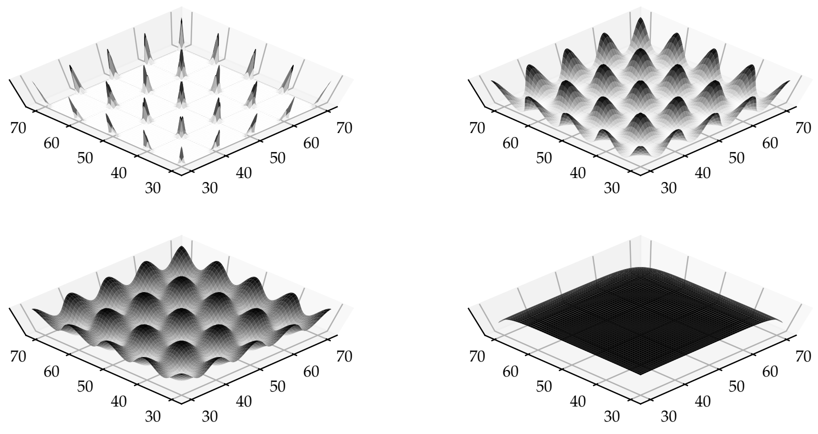

Fig. 4. The sum of Gaussian distributions centered on lattice points getting increasingly closer to the uniform distribution when the standard deviation s approaches the smoothing parameter η_ϵ for a small ϵ. The lattice considered is the simple grid with square of size 10, which implies $\lambda_1 = \lambda_2 = 10$. The values of s are given next. Upper left: 0.2. Upper right: 2. Bottom left: 5. Bottom right: 10.

The importance of the smoothing parameter is subtle, but it plays a central role in Micciancio and Regev's worst to average case reduction [22]. We now give a simple explanation of the importance of the smoothing parameter η_ϵ of a lattice.

Suppose we can sample random lattice points, and pick a random lattice point $\mathbf{v}$ from some lattice Λ. Consider Λ_q the q-ary lattice induced by Λ, that is

$$\Lambda_q = \{\mathbf{v}/q : \mathbf{v} \in \Lambda\},$$

which is a supper lattice of Λ. This construction can be seen in Figure 5.

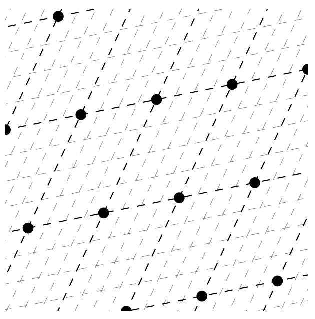

Fig. 5. A subdivided lattice Λ_4.

Now, sample a point $\mathbf{x}$ from Λ_q from the Gaussian distribution centered in $\mathbf{v}$ and with scaling parameter $s > \eta_\epsilon$, that is $\rho_{\mathbf{v},s}$. Then, s being greater than smoothing parameter guarantees that $\mathbf{x}$ is almost uniformly distributed over all possible points in Λ_q.

One can ask why didn't we simply picked a random point from Λ_q, and went through the trouble of sampling from Gaussian distributions. The answer, which is exactly the importance of this sampling procedure, is that we have a uniformly distributed $\mathbf{x}$ in Λ_q for which we know a close lattice point in Λ, given by $\mathbf{v}$. In other words, the whole procedure gives us uniformly random points from Λ_q, but the one who sampled the point knows a close vector to it in Λ. The next lemma tells us how close we can expect $\mathbf{x}$ and $\mathbf{v}$ to be.

Lemma 13 ([22], adapted)**.** *Let Λ be an n-dimensional lattice of rank k, $\mathbf{c}$ be a vector in span(Λ), and s be a scaling factor greater than $\eta_\epsilon(\Lambda)$ for some ϵ in $(0,1)$. Sample the vector $\mathbf{x}$ from Λ_q using the Gaussian $\rho_{\mathbf{c},s}$. Then*

$$\Pr\left(\|\mathbf{x}-\mathbf{c}\| > s\sqrt{n}\right) \le \frac{1+\epsilon}{1-\epsilon}2^{-k}.$$

In words, the distance between $\mathbf{x}$ and the center of the distribution $\mathbf{c}$ is less than $s\sqrt{n}$ with overwhelming probability.

Some important notes on sampling from discrete Gaussian distributions over lattices are given next. The first one is that, to simplify our discussion in end of this section, we assumed that it is possible to pick random lattice points. To be more precise, one would have to define where the points are to be taken and present an efficient algorithm for that. Luckily, this is not necessary for the reduction, and everything can be made modulo the fundamental parallelepiped. The second, which is also an algorithmic matter, is that we said nothing about how to sample from discrete Gaussian over lattices. It turns out that there are simple sampling algorithms which run efficiently for any basis such as the ones considered by Gentry, Peikert, and Vaikuntanathan [12]. One interesting algorithm they show works even with sets of independent vectors that do not form a basis for the lattice. The third note is that the lemma bounding smoothing parameter of a lattice depends on the parameter λ_n, which is possibly hard to bound. One can argue that, using Minkowski's lemma, we can use a binary search to find a good estimate for the smoothing parameter, supposing we know how to efficiently test if a candidate parameter is a good smoothing parameter. A simpler solution is to use Gentry, Peikert, and Vaikuntanathan's [12] bound on the smoothing parameter given by $\eta_\epsilon \leq \|\hat{\mathbf{B}}\|\omega\left(\sqrt{\log(n)}\right)$, where $\hat{\mathbf{B}}$ is the maximum norm of the Gram-Schmidt orthogonalized vectors.

4.3 A collision resistant hash function

In this brief section we introduce the hash function used for the worst to average case reduction. Before defining the function in question, we first define a natural problem on which the function is based.

Definition 14 (SIS). *Given an integer q, a q-ary matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a positive real β, the short integer solution problem* (SIS) *asks to find a nonzero vector $\mathbf{z} \in \mathbb{Z}^m$ such that $\mathbf{A}\mathbf{z} = \mathbf{0} (\bmod q)$ and $\|\mathbf{z}\| \leq \beta$.*

Note that this problem is the lattice problem of finding a nonzero short vector in the lattice $\mathcal{L}^\perp(\mathbf{A}) = \{\mathbf{z} : \mathbf{A}\mathbf{z} \equiv 0 \bmod q\}$. Next, we show a lemma on how to choose β so that a solution to SIS is guaranteed to exist. Unless stated otherwise, this is the parameter β considered for SIS instances.

Lemma 15. *Let q be any integer, and $\mathbf{A}$ be any matrix in $\mathbb{Z}_q^{n\times m}$. If $\beta \geq \sqrt{m} q^{n/m}$, then there exists a nonzero vector $\mathbf{z}$ such that $\mathbf{A}\mathbf{z} \equiv 0 \mod q$ and $\|z\| \leq \beta$.*

Proof. Let S be the set of all m-dimensional vectors with all coordinates less than or equal to $\lceil q^{n/m} \rceil$. Then $|S| = (q^{n/m}+1)^m > q^n$. So there must exist two different vectors $\mathbf{z}_1$ and $\mathbf{z}_2$ such that $\mathbf{A}\mathbf{z}_1 = \mathbf{A}\mathbf{z}_2$. Consider $\mathbf{z}$, the difference between the two, that is $\mathbf{z} = \mathbf{z}_1 - \mathbf{z}_2$. By the restriction on the coordinates of $\mathbf{z}_1$ and $\mathbf{z}_2$, the norm of every coordinate in $\mathbf{z}$ must be less than or equal to $q^{n/m}$. Therefore, $\|\mathbf{z}\| \leq \sqrt{mq^{2n/m}} = \sqrt{m}q^{n/m} \leq \beta$. □

Definition 16 (SIS-based hash). *Let q be any integer, and $\mathbf{A}$ be any matrix in $\mathbb{Z}_q^{n\times m}$, for some n and m such that $m \geq n \log q$. The SIS-based hash function*

$$h_{\mathbf{A}} : \{0,1\}^m \to \mathbb{Z}_q^n$$

is defined as

$$h_{\mathbf{A}}(\mathbf{r}) = \mathbf{A}\mathbf{r} \pmod q.$$

The next lemma shows that, if one can efficiently find collisions for the hash function $h_{\mathbf{A}}$, then there is an efficient algorithm that approximates SIS for polynomial factors.

Lemma 17. *Given a matrix* $\mathbf{A}$ *and a target norm* β*, finding collisions in* $h_{\mathbf{A}}$ *implies in finding a vector* $\mathbf{z}$ *such that* $\|\mathbf{z}\| \leq \gamma\beta$*, and* $\mathbf{A}\mathbf{z} \equiv 0 \mod q$*, for an approximation factor* γ *polynomial on* n*.*

Proof. Let $(\mathbf{r}, \mathbf{r}')$ be a collision for $h_{\mathbf{a}}$, that is, $h_{\mathbf{A}}(\mathbf{r}) \equiv h_{\mathbf{A}}(\mathbf{r}')(\text{mod} q)$. Then $h_{\mathbf{A}}(\mathbf{r} - \mathbf{r}') \equiv 0(\text{mod} q)$. But since $\mathbf{r}$ and $\mathbf{r}'$ are vectors with binary entries[2], their difference $\mathbf{r} - \mathbf{r}'$ has entries in $-1, 0, 1$. Therefore $\|\mathbf{r} - \mathbf{r}'\| \leq \sqrt{n} \leq \beta\gamma$. Since β is not negligible, γ is polynomial on n, and $\|\mathbf{r} - \mathbf{r}'\|$ is an approximation solution for the γ-SIS instance $(\mathbf{A}, \beta)$.

□

4.4 The reduction

In 1996, Ajtai [2] showed the first reduction form a a reduction from a set of approximation lattice problems, which are believed to be hard in the worst-case, to the average case of a problem. This construction was improved by Micciancio and Regev in 2007 [22], who introduced some important tools to analyze lattices.

In this section, we review one of Micciancio and Regev's worst to average case reduction. In their reduction, they introduce an artificial problem on lattices that is more suitable for their presentation. This problem was used as an intermediate between the reductions.

In this presentation, we do not consider the intermediate problem and show the description of a direct reduction from the wort-case hard problem γ-SIVP, for an approximation factor $\gamma = \mathcal{O}(n \log n)$, to the average-case problem of finding collisions for the SIS-based hash function.

Overview of the reduction. We want to show that, if we are given an oracle $\mathcal{F}$ that finds collisions for a non-negligible fraction of the functions in the family $\{h_{\mathbf{A}}\}$ of SIS-based hash functions, then we can efficiently solve γ-SIVP with overwhelming probability for some $\gamma = \beta n^c$, where c is a constant. At first, we are given an instance of γ-SIVP, consisting of a a basis matrix $\mathbf{B}$ and an approximation factor $\gamma = \beta n^c$. The idea is to progressively obtain sets of short vectors by using the oracle to find collisions for a number of random SIS-based hash functions, *picked uniformly at random*, and which are related, in a secret way only known to us, to the SIVP instance we want to solve. The fact that the relation must be oblivious to the oracle is very important, because this is what makes the solutions returned by the oracle not problematically distributed[3]. Finally, by combining the collisions for these hash functions we hopefully can obtain a solution for the γ-SIVP.

[2] I'm avoiding the term binary vectors because here sums are not modulo 2.

[3] Intuitively, we consider the oracle algorithm as an adversary. The oracle is a very powerful machine, which can solve hard problems. So we can assume it knows everything that can be learned from our input, and will use it to give solutions to SIS which we cannot combine to obtain a solution to γ-SIVP. Our goal, then, is to make it impossible, in the information-theoretic sense, for the oracle to extract sufficient information about the secret γ-SIVP instance we want to solve.

Theorem 18. *For $\gamma = \mathcal{O}(n^3)$, there is a polynomial time reduction from γ-SIVP to the average-case problem of finding collision in the SIS-based hash with parameters n, m, q, where q is polynomial on n and $m = \Theta(n \log q)$.*

Proof. Let $\mathbf{B} \in \mathbb{Z}^{n\times n}$ be the given basis matrix of a lattice Λ, and let $s > \eta_\epsilon(\Lambda)$. Consider For each point $\mathbf{v}_i$, sample an error vector $\mathbf{e}_i$ from Λ_q following $\rho_{\mathbf{0},s}$ over the super-lattice $\mathcal{L}(\mathbf{B}/q)$. This process is illustrated by Figure 6. Build the following matrix

$$\mathbf{A} = [\mathbf{a}_1|\mathbf{a}_2|\dots|\mathbf{a}_m],$$

where each $\mathbf{a}_i = q\mathbf{B}^{-1}(\mathbf{v}_i + \mathbf{e}_i \mod P(\mathbf{B}))$. This operation can be seen as a reduction mod q of each coordinate of $\mathbf{a}_i$, by considering a bijection between the vectors $\{0,1,2,\dots,q-1\}^n$ and the points in $P(\mathbf{B}) \cap \mathcal{L}(\mathbf{B}/q)$. Because of the smoothing parameter, the vectors $\mathbf{y}_i + \mathbf{e}_i$ mod $P(\mathbf{B})$ are very close to uniformly distributed over $P(\mathbf{B})$. Then by the reversible nature of the construction of $\mathbf{a}_i$, each $\mathbf{a}_i$ is very close to uniformly distributed over the vectors in $\mathbb{Z}_q^n$. Therefore, the matrix $\mathbf{A}$ is close to uniformly distributed over the set $\mathbb{Z}_q^{n\times m}$.

Choosing the hash function $h_\mathbf{A}$ uniformly at random. We first note that this is equivalent to pick $\mathbf{A}$ uniformly at random. The problem here is to pick a matrix that really is uniformly distributed, but that also encodes some information about the SIVP problem we have to solve. The idea is to use the sampling procedure discussed in Section 4.2, but now we state how to do it without picking random points in the infinite lattice. Consider the supper lattice $\Lambda_q = \{\mathbf{v}/q : \mathbf{v} \in \Lambda_q\}$. Let s be an approximation for smoothing factor of Λ, for some negligible ϵ. Sample a set of m points $\{\mathbf{x}_1,\dots,\mathbf{x}_m\}$ from Λ_q using the discrete Gaussian distribution $D_{\mathbf{0},s}$. By Lemma 13, $\|\mathbf{x}_i\| \le s\sqrt{n}$ with overwhelming probability. Now for each $\mathbf{x}_i$, let $\mathbf{y}_i = \mathbf{x}_i \mod P(\mathbf{B})$. By our discussion on the properties of Λ_q in Section 4.2, the vectors $\mathbf{y}_i$ are randomly distributed in the set $P(\mathbf{B}) \cap \Lambda_q$. These properties make the vectors $\mathbf{y}$ would the perfect candidates for forming the columns of matrix $\mathbf{A}$. The problem is that their coordinates are not q-ary. To solve this problem, we can simply identify each coordinate of the set of the tiled fundamental parallelepiped tiled in q^n parts, that is $P(\mathbf{B}) \cap \Lambda_q$, which one natural vector from $\mathbb{Z}_q^n$, by considering for each vector the fraction of each basis element that constitute this vector. In symbols, the vector $\mathbf{v}$ is associated with the vector $q\mathbf{B}^{-1}\mathbf{v}(\mod q)$. Note that a vector is in Λ if and only if $\mathbf{B}^{-1}\mathbf{v}$ is an integer, which implies that its reduction equals to 0. Perform this reduction procedure for each $\mathbf{y}_i$, obtaining a vector $\mathbf{a}_i \in \mathbb{Z}_q^n$. Therefore, the matrix $\mathbf{A} = [\mathbf{a}_1 \dots \mathbf{a}_m]$ is uniformly distributed over $\mathbb{Z}_q^{n\times m}$, and this determines the hash function to which we want the oracle to find collisions. This construction can be seen in Figure 6.

Obtaining short vector from collisions. Take the matrix $\mathbf{A}$ obtained by the sampling procedure, and ask the oracle for a collision for $h_\mathbf{A}$. If the oracle successfully gives us a collision $(\mathbf{r}, \mathbf{r}')$, from the proof of Lemma 17, we have a vector $\mathbf{e} = \mathbf{r} - \mathbf{r}' \in \{-1,0,1\}^m$ such that $\mathbf{A}\mathbf{e} \equiv 0 \mod q$. That is, $\mathbf{v} = \sum_{i=1}^m \mathbf{a}_i e_i$ is a lattice

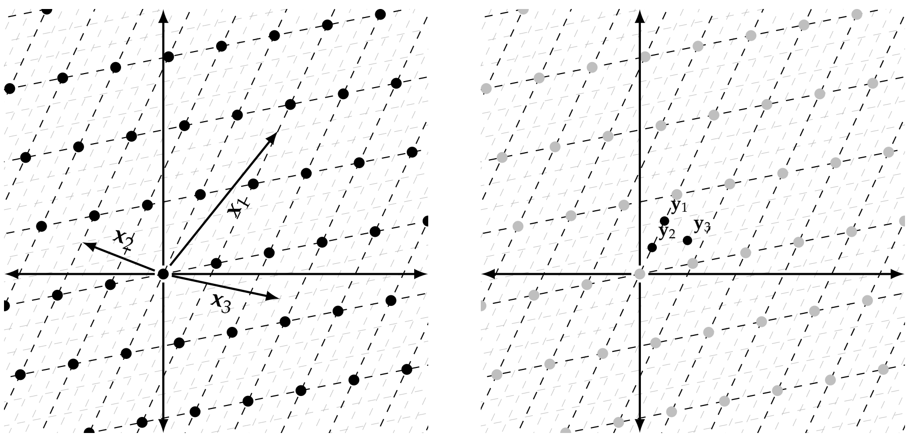

Fig. 6. On the left, three samples from the discrete Gaussian centered on the origin. On the right, there three points reduced modulo the fundamental parallelepiped. The indexes are such that $\mathbf{y}_i = \mathbf{x}_i \bmod P(\Lambda)$. An appropriate scaling factor for the discrete Gaussian, with respect to the smoothing parameter, gives some guarantees that the vectors $\mathbf{y}_i$ are distributed uniformly inside the fundamental parallelepiped, and also that the vectors $\mathbf{x}_i$ are not too large ($\|x\| \approx \lambda_n$).

vector. Remember that each $\mathbf{a}_i$ is the reduction $\bmod P(\mathbf{B})$ of the lattice vector $\mathbf{x}_i$, and therefore $\mathbf{a}_i = \mathbf{x}_i + \mathbf{u}_i$, where $\mathbf{u}_i$ is also a lattice vector. Therefore

$$\mathbf{v} = \sum_{i=1}^{m} \mathbf{x}_i e_i + \sum_{i=1}^{m} \mathbf{u}_i e_i.$$

Since both $\mathbf{v}$ and $\sum_{i=1}^{m} \mathbf{u}_i$ are in Λ, so should be $\mathbf{c} = \sum_{i=1}^{m} \mathbf{x}_i e_i$. This is exactly the short vector we want. Since $\mathbf{e} \in -1, 0, 1^m$, vector $\mathbf{c}$ is the sum of at most m vectors of length less than or equal to $s\sqrt{n}$. This means

$$\|\mathbf{c}\| \le ms\sqrt{n} \approx m\eta_\epsilon\sqrt{n} \le m\lambda_n\sqrt{n\log n} = \mathcal{O}(n^3\lambda_n).$$

Solving the γ-SIVP and bounding the probability of failure. The value of the parameter q controls the density of the sampling sets, which grows exponentially in n. This makes unnecessary for it to be large, and is taken to be polynomial on n. The presented procedure can find short vectors given an oracle that finds collisions in random SIS-based hash function. But the γ-SIVP asks for n independent short vectors. The obvious solution is perform the procedure iteratively until n independent short vectors are found. The problem we now face is that of showing that the probability that the solutions the oracle finds are not in the same $n-1$ dimensional space is non-negligible. The proof is rather technical and is omitted here, but we explain why we should expect the algorithm to solve SIVP. The main intuition behind why this happens is that, even though the oracle can learn the vectors $\mathbf{y}_i$, it cannot learn sufficient information about the vectors

$\mathbf{x}_i$. Looking at the lattice in Figure 6 we can see that the representatives in the fundamental parallelepiped have a maximum likelihood $\mathbf{x}$ that reduces to them. So the oracle can indeed obtain information on $\mathbf{x}_i$. But this information cannot be complete, in the information theoretic sense, and the oracle will always have some uncertainty on the set of the vectors $\mathbf{x}_i$. This uncertainty grows fast with respect to the dimension n.

□

We note that the reduction presented is simpler than the original one. As a consequence, the factors we obtain here are not optimal, and can be improved [12, 22].

References

1. Adleman, L.M.: On breaking the iterated merkle-hellman public-key cryptosystem. In: Advances in Cryptology. pp. 303–308. Springer (1983)
2. Ajtai, M.: Generating hard instances of lattice problems. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 99–108. ACM (1996)
3. Ajtai, M., Dwork, C.: A public-key cryptosystem with worst-case/average-case equivalence. In: Proceedings of the twenty-ninth annual ACM symposium on Theory of computing. pp. 284–293. ACM (1997)
4. Babai, L.: On lovász'lattice reduction and the nearest lattice point problem. Combinatorica 6(1), 1–13 (1986)
5. Bernstein, D.J., Buchmann, J., Dahmen, E.: Post-quantum cryptography. Springer Science & Business Media (2009)
6. Bogdanov, A., Trevisan, L., et al.: Average-case complexity. Foundations and Trends in Theoretical Computer Science 2(1), 1–106 (2006)
7. Chen, L., Chen, L., Jordan, S., Liu, Y.K., Moody, D., Peralta, R., Perlner, R., Smith-Tone, D.: Report on post-quantum cryptography. US Department of Commerce, National Institute of Standards and Technology (2016)
8. Daemen, J., Rijmen, V.: The design of Rijndael: AES-the advanced encryption standard. Springer Science & Business Media (2013)
9. DES: Data encryption standard. In: FIPS PUB 46, Federal Information Processing Standards Publication. pp. 46–2 (1977)
10. Diffie, W., Hellman, M.: New directions in cryptography. IEEE transactions on Information Theory 22(6), 644–654 (1976)
11. Galbraith, S.D.: Mathematics of public key cryptography. Cambridge University Press (2012)
12. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the fortieth annual ACM symposium on Theory of computing. pp. 197–206. ACM (2008)
13. Gentry, C., et al.: Fully homomorphic encryption using ideal lattices. In: STOC. vol. 9, pp. 169–178 (2009)
14. Goldreich, O., Goldwasser, S., Halevi, S.: Public-key cryptosystems from lattice reduction problems. In: Advances in Cryptology-CRYPTO'97: 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 1997. Proceedings. p. 112. Springer (1997)

15. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: International Algorithmic Number Theory Symposium. pp. 267–288. Springer (1998)
16. Lagarias, J.C., Odlyzko, A.M.: Solving low-density subset sum problems. Journal of the ACM (JACM) 32(1), 229–246 (1985)
17. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Mathematische Annalen 261(4), 515–534 (1982)
18. Levin, L.A.: Average case complete problems. SIAM Journal on Computing 15(1), 285–286 (1986)
19. McEliece, R.J.: A public-key cryptosystem based on algebraic coding theory. Deep Space Network Progress Report 44, 114–116 (1978)
20. Merkle, R., Hellman, M.: Hiding information and signatures in trapdoor knapsacks. IEEE transactions on Information Theory 24(5), 525–530 (1978)
21. Micciancio, D., Goldwasser, S.: Complexity of lattice problems: a cryptographic perspective, vol. 671. Springer Science & Business Media (2012)
22. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM Journal on Computing 37(1), 267–302 (2007)
23. Miller, V.: Use of elliptic curves in cryptography. Advances in Cryptology (CRYPTO85) pp. 417–426 (1986)
24. Nguyen, P.: Cryptanalysis of the goldreich-goldwasser-halevi cryptosystem from crypto'97. In: Annual International Cryptology Conference. pp. 288–304. Springer (1999)
25. Odlyzko, A.M.: The rise and fall of knapsack cryptosystems. Cryptology and computational number theory 42, 75–88 (1990)
26. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
27. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: Proceedings of the 35th Annual Symposium on Foundations of Computer Science. pp. 124–134. IEEE (1994)
28. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 27–47. Springer (2011)