On the Higher-bit Version of Approximate Inhomogeneous Short Integer Solution Problem

Anaëlle Le Dévéhat * Hiroki Shizuya Shingo Hasegawa

September 1, 2022

Tohoku University, Sendai, Japan
* anaelle.le.devehat.s8@alumni.tohoku.ac.jp

Abstract

We explore a bitwise modification in Ajtai's one-way function. Our main contribution is to define the higher-bit approximate inhomogeneous short integer solution (ISIS) problem and prove its reduction to the ISIS problem. In this new instance, our main idea is to discard low-weighted bits to gain compactness.

As an application, we construct a bitwise version of a hash-and-sign signature in the random oracle model whose security relies on the (Ring)-LWE and (Ring)-ISIS assumptions. Our scheme is built from the hash-and-sign digital signature scheme based on the relaxed notion of approximate trapdoors introduced by Chen, Genise and Mukherjee (2019). Their work can be interpreted as a bitwise optimization of the work of Micciancio and Peikert (2012). We extend this idea and apply our technique to the scheme by discarding low-weighted bits in the public key. Our modification brings improvement in the public key size but also in the signature size when used in the right setting.

However, constructions based on the higher-bit approximate ISIS save memory space at the expense of loosening security. Parameters must be set in regards with this trade-off.

1 Introduction

1.1 Background

Since Peter Shor's breakthrough work in 1994 [25], it became clear that quantum computers are able to break usual cryptographic primitives based on number theory assumptions. For instance, a quantum computer can break factoring-based cryptography in polynomial time of the security parameter. This results threaten usual cryptography and reveal a need for efficient post-quantum secure cryptography. In 2017, NIST launched its still ongoing post-quantum cryptography (PQC) standardization process [22]. It illustrates the necessity of finding

efficient and realistic post-quantum secure cryptographic constructions in order to guarantee the confidentiality and integrity of digital communications.

One high potential candidate for PQC is lattice-based cryptography. It has been an active area of research since Ajtai's groundbreaking work in 1996 which demonstrates strong worst-case to average-case reductions on lattices problems [2, 3]. Worst-case to average-case hardness is very important in cryptographic constructions since it needs to be hard to attack a construction for random instances. Moreover, underlying lattice problems provide strong security even for quantum adversaries (no polynomial attack is known).

The attractiveness of lattice-based cryptography comes from its elegant constructions and efficiency improvements obtained using lattices with algebraic structure [14, 20]. It also enjoys great versatility afforded by the learning with errors (LWE) problem [24]. A lot of lattice-based cryptographic primitives has been studied such as fully homomorphic encryption schemes [12], public-key encryption [14, 15] but also attribute-based encryption and (hierarchical) identity-based encryption [1, 7].

In this work, we focus on lattice-based signatures among lattice-based cryptographic schemes. Even if there has been early attempts at lattice-based digital signatures, it is only in 2008 that the first direct constructions of lattice-based signatures appeared. A "hash-and-sign" signature scheme was constructed by Gentry, Peikert and Vaikuntanathan [13]. At the same time, a provably secure one-time signature using ideal lattices was constructed by Lyubashevsky and Micciancio [17]. Both schemes enjoys security based on the hardness of worst-case lattice problems. Even if both schemes achieved short signatures, they still had several disadvantages. These constructions led the way to two lines of research. First, Lyubashevsky used the Fiat-Shamir transform to improve the one-time signature [17] in several subsequent works [16]. Several of the best candidates in NIST PQC standardization procedure are based on the rejection sampling method [4, 10, 22]. On the other hand, the GPV "hash-and-sign" signature scheme [13] is not very practical. In their work, Gentry, Peikert and Vaikuntanathan show how to sample solutions following a distribution simulatable without knowing the secret to avoid any information leakage. In order to do so, they use a gaussian sampler which leads to various difficulties and complexity. A more satisfactory solution to this problem was given by Micciancio and Peikert [21]. Their work brought several improvements both for security and efficiency in GPV scheme line of work.

1.2 Related work

In this work, we study constructions based on Ajtai's one-way function and trapdoor [3]. In lattice-based hash-and-sign GPV signature [13], a signer is assigned a uniformly random public matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ along with a trapdoor $\boldsymbol{S} \in \mathbb{Z}_q^{m \times m}$ which verifies $\boldsymbol{AS} = \boldsymbol{0} \pmod q$. The trapdoor $\boldsymbol{S}$ is usually a basis of short lattice vectors solution to the SIS problem with regards to $\boldsymbol{A}$. Thus, using $\boldsymbol{S}$, one can find short preimages for the Ajtai's function defined by $\boldsymbol{A}$ and sign a message. The resulting signature's norm depends on the norms of the

columns in $\boldsymbol{S}$. In order to further optimize and improve this kind of digital signature, it is highly relevant to improve the algorithms for trapdoor and key generation.

At first, improvements of Ajtai's trapdoor generation algorithm [5, 23] were rather complex and inefficient. Only in 2012, the introduction by Micciancio and Peikert of their elegant $\boldsymbol{G}$-trapdoor construction [21] enabled faster and shorter signatures. However, even using the $\boldsymbol{G}$-trapdoor construction, hash-and-sign signature based on Ajtai's function is still impractical due to large keys and signatures sizes. For instance, when compared to lattice-based signatures candidates of NIST PQC standardization process [4, 10, 11], the hash-and-sign signature instantiated with $\boldsymbol{G}$-trapdoors has about six times larger public keys and signatures sizes for a same level of security.

In order to reduce this difference, Chen, Genise and Mukherjee constructed a $\boldsymbol{F}$-trapdoor [9] from the $\boldsymbol{G}$-trapdoor [21]. The innovation in their work is the definition of the approximate ISIS problem which reduces to the ISIS problem. It allows a certain error when sampling a preimage for Ajtai's function. By allowing a little error, the $\boldsymbol{G}$-trapdoor is reduced to an approximate version called $\boldsymbol{F}$-trapdoor. The hash-and-sign signature instantiated with $\boldsymbol{F}$-trapdoors enjoys sEU-CMA security and much smaller public keys and signatures sizes than the one with $\boldsymbol{G}$-trapdoors. However, these sizes are still too large when compared with state-of-the-art digital signatures based on NTRU lattices [11] or on the rejection sampling approach [4, 10].

1.3 Contributions

Our main contribution is the definition of the higher-bit approximate ISIS problem along with its reduction to the ISIS problem. This newly defined problem permits improvements in constructions based on the ISIS problem. It is based on discarding low-weighted bits of coefficients in the matrix $\boldsymbol{A}$ which defines Ajtai's function. As an application of the higher-bit approximate ISIS problem, we adapt the hash-and-sign signature by Chen, Genise and Mukherjee [9] : we construct a sEU-CMA secure hash-and-sign digital signature along with adapted trapdoor generation and preimage sampling algorithms.

In our application, the public key $\boldsymbol{A}$ is constructed from the high-weighted bits of the public key in [9]. This idea fits in the approximate setting. Furthermore, with the right parameters setting, discarding low-weighted bits in the public key allows for a possible similar optimization of the signature. Our construction seems like a natural following of the $\boldsymbol{F}$-trapdoor signature scheme [9]. Indeed, the gadget matrix $\boldsymbol{F}$ is basically defined as the gadget matrix $\boldsymbol{G}$ [21] but without low-weighted bits entries.

With our modification, the public key belongs to $\mathbb{Z}_{\frac{q}{b^d}}^{n\times m}$ rather than $\mathbb{Z}_q^{n\times m}$ (where $q = b^k, d < k$). This is a direct consequence of using the higher-bit approximate ISIS problem as the underlying hardness problem. Moreover, the signature is in $\mathbb{Z}_{\frac{q}{b^d}}^{m}$ rather than $\mathbb{Z}_q^m$. Applying our technique to the $\boldsymbol{F}$-trapdoor signature scheme allows to save $n \times m \times d\lceil \log_2 b \rceil$ bits in the public key and

$m \times d\lceil \log_2 b \rceil$ bits in the signature.

However, this setting implies a trade-off between security and memory space. This trade-off is due to the reduction loss when using the higher-bit approximate ISIS rather than the approximate ISIS. In order to assess this trade-off, we give some concrete parameters and results. We expect our construction to reduce the public key size by about half and significantly reduce the signature size at the expense of a reasonable drop in the security level. Moreover, providing a higher security parameter, we estimate 155-bit security level rather than 88-bit security as given in [9] for about the same or smaller key sizes. We may note that optimization based on discarding low-weighted bits can be seen in the lattice-based signature CRYSTALS-Dilithium [10].

We note that our hash-and-sign signature construction in the random oracle model can translate to the Ring setting under Ring-LWE and Ring-SIS assumptions [18].

1.4 Organization

In Section 3, we define and study the higher-bit approximate ISIS problem and its reduction to the ISIS problem. In section 4, we introduce our main idea for a new construction based on the higher-bit approximate ISIS. In Section 5, we construct new trapdoor generation and preimage sampling algorithms and study the resulting distributions. Finally, in Section 6, we instantiate a sEU-CMA secure hash-and-sign signature using our algorithms.

2 Preliminaries

2.1 Notations and Linear Algebra

We denote the set of real numbers by $\mathbb{R}$, the set of integers by $\mathbb{Z}$ and the set of positive integers by $\mathbb{N}$. Denote $\mathbb{Z}/\mathrm{q}\mathbb{Z}$ by $\mathbb{Z}_q$. We use the notation $x \leftarrow U(S)$ when a variable x is drawn uniformly at random from the set S. Moreover, we use $\approx_s$ as the abbreviation for statistically close. A vector $\boldsymbol{v}$ is always in column form and represented in lower-case bold letters. A matrix $\boldsymbol{A}$ is always represented in upper-case bold letters. For a vector $\boldsymbol{v}$, we denote the i^{th} component of $\boldsymbol{v}$ as v_i. We do the same for a matrix $\boldsymbol{A}$ and denote the i^{th} component of the j^{th} column of $\boldsymbol{A}$ as $a_{i,j}$. We denote the l_p-norm of a vector $\boldsymbol{v}$ as $\|\boldsymbol{v}\|_p := (\sum v_i^p)^{\frac{1}{p}}$. The norm of a matrix is the norm of its longest column : $\|\boldsymbol{A}\|_p := max_i \|\boldsymbol{a}_i\|_p$. By default we use l_2-norm. A short vector is a vector whose norm is small but not necessarily its dimension.

If a symmetric matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$ verifies that for all $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{x}^t \boldsymbol{\Sigma} \boldsymbol{x} > 0$ ($\geq$0) then $\boldsymbol{\Sigma}$ is positive (semi)-definite. For two positive (semi)-definite matrices $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$, we note $\boldsymbol{\Sigma}_1 > \boldsymbol{\Sigma}_2$ ($\geq$) if $\boldsymbol{\Sigma}_1 - \boldsymbol{\Sigma}_2$ is positive (semi)-definite. $\sqrt{\boldsymbol{\Sigma}}$ designates any full rank matrix $\boldsymbol{T}$ such that $\boldsymbol{\Sigma} = \boldsymbol{T}\boldsymbol{T}^t$.

2.2 Lattices Background

A m-dimensional lattice Λ of rank $k \leq m$ is a discrete additive subgroup of $\mathbb{R}^m$. It is generated by all linear combinations with integers coefficients of k linearly independant basis vectors $\boldsymbol{B} = \{\boldsymbol{b}_1, ..., \boldsymbol{b}_k\}$.

In many cryptographic work, we use q-ary integer lattices. For some positive integers $m, n \in \mathbb{N}$, $q \geq 2$, $\boldsymbol{u} \in \mathbb{Z}_q^n$ and $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ we can define the following m-dimensional full rank q-ary lattices :

$$\Lambda^{\perp}(\boldsymbol{A}) = \Lambda_q^{\perp}(\boldsymbol{A}) := \{\boldsymbol{x} \in \mathbb{Z}^m : \boldsymbol{A}.\boldsymbol{x} = \boldsymbol{0} \pmod q\};$$
$$\Lambda_u^{\perp}(\boldsymbol{A}) := \{\boldsymbol{x} \in \mathbb{Z}^m : \boldsymbol{A}.\boldsymbol{x} = \boldsymbol{u} \pmod q\}.$$

In this work, we study vectors distributions obtained when sampling in q-ary lattices. To do so, we first need to define what is a discrete Gaussian distribution over a lattice Λ.

Definition 1 (Gaussian function on $\mathbb{R}^n$ with parameter s : ρ_s [9])**.** For any $s > 0$,

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \quad \rho_s(\boldsymbol{x}) = e^{-\pi ||\boldsymbol{x}||^2 / s^2}$$

Definition 2 (Discrete Gaussian distribution $D_{\Lambda+c,s}$ [9])**.** For any $\boldsymbol{c} \in \mathbb{R}^n$, real $s > 0$, and n-dimensional lattice Λ,

$$\forall \boldsymbol{x} \in \Lambda + \boldsymbol{c}, \quad D_{\Lambda+c,s}(\boldsymbol{x}) = \frac{\rho_s(\boldsymbol{x})}{\sum_{\boldsymbol{a} \in \Lambda + \boldsymbol{c}} \rho_s(\boldsymbol{a})}$$

When omitted, s and $\boldsymbol{c}$ are taken to be 1 and 0 respectively.

This definition of discrete Gaussian distribution can be extended to non-spherical Gaussians [9]. However we do not make use of this definition in our work, thus we omit it here.

Moreover, in this work, some conditions on the parameters are set in regards with the smoothing parameter. We recall its definition.

Definition 3 (Smoothing parameter [19])**.** For any lattice Λ and positive real $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(\Lambda)$ is the smallest real $s > 0$ such that $\rho_{1/s}(\Lambda^* \{\boldsymbol{0}\}) \leq \epsilon$.

Definition 4 ([9])**.** For a positive semi-definite $\boldsymbol{\Sigma} = \boldsymbol{T}\boldsymbol{T}^t$, $\epsilon > 0$, and lattice Λ with $span(\Lambda) \subseteq span(\boldsymbol{\Sigma})$, we say $\eta_\epsilon(\Lambda) \leq \sqrt{\boldsymbol{\Sigma}}$ if $\eta_\epsilon(\boldsymbol{T}^+ \Lambda) \leq 1$.

2.3 LWE, SIS, ISIS and Approximate ISIS

First we recall the definition of the learning with errors problem.

Definition 5 (Decisional learning with errors [24])**.** For $n, m \in \mathbb{N}$ and modulus $q \geq 2$, distributions $\theta, \pi, \chi \subseteq \mathbb{Z}_q$. An LWE sample is obtained from sampling secret vector $\boldsymbol{s} \leftarrow \theta^n$, public matrix $\boldsymbol{A} \leftarrow \pi^{n \times m}$, and error vector $\boldsymbol{e} \leftarrow \chi^m$, and outputting $(\boldsymbol{A}, \boldsymbol{y}^t := \boldsymbol{s}^t \boldsymbol{A} + \boldsymbol{e}^t \pmod q)$.

We say that an algorithm solves $\mathrm{LWE}_{n,m,q,\theta,\pi,\chi}$ if it distinguishes the LWE sample from a random sample distributed as $\pi^{n \times m} \times U(\mathbb{Z}_q^m)$ with probability greater than $1/2$ plus non-negligible.

Lemma 1 ([6])**.** *For n, m, q, s chosen as LWE hardness is based on GapSVP and SIVP,*

$LWE_{n,m',q,D_{\mathbb{Z},s},U(\mathbb{Z}_q),D_{\mathbb{Z},s}}$ is as hard as $LWE_{n,m,q,U(\mathbb{Z}_q),U(\mathbb{Z}_q),D_{\mathbb{Z},s}}$ for $m' \leq m - (16n + 4 \log \log q)$.

Now we recall the SIS and ISIS problems.

Definition 6 (SIS [2])**.** For any $n, m \in \mathbb{N}$, $q \in \mathbb{Z}$ and $\beta \in \mathbb{R}$, define the short integer solution problem $\mathrm{SIS}_{n,m,q,\beta}$ as follows: Given $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, find a non-zero vector $\boldsymbol{x} \in \mathbb{Z}^m$ such that $\|\boldsymbol{x}\| \leq \beta$, and

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0} \pmod q$$

Definition 7 (ISIS)**.** For any $n, m \in \mathbb{N}$, $q \in \mathbb{Z}$ and $\beta \in \mathbb{R}$, define the inhomogeneous short integer solution problem $\mathrm{ISIS}_{n,m,q,\beta}$ as follows: Given $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, $\boldsymbol{y} \in \mathbb{Z}_q^n$, find a vector $\boldsymbol{x} \in \mathbb{Z}^m$ such that$\|\boldsymbol{x}\| \leq \beta$, and

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y} \pmod q$$

In their work, Chen, Genise and Mukherjee introduce a relaxed notion of the ISIS problem. We will be using their approximate setting in our work.

Definition 8 (Approx.ISIS [9])**.** For any $n, m \in \mathbb{N}$, $q \in \mathbb{Z}$ and $\alpha, \beta \in \mathbb{R}$, define the approximate inhomogeneous short integer solution problem $\mathrm{Approx.ISIS}_{n,m,q,\alpha,\beta}$ as follows: Given $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, $\boldsymbol{y} \in \mathbb{Z}_q^n$, find a vector $\boldsymbol{x} \in \mathbb{Z}^m$ such that $\|\boldsymbol{x}\| \leq \beta$, and there is a vector $\boldsymbol{z} \in \mathbb{Z}^n$ satisfying

$$\|\boldsymbol{z}\| \leq \alpha \quad \textit{and} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{z} \pmod q$$

With the right parameters, we have the following reductions [9] :

- $LWE_{n,m,q,\theta,\mathbb{U}(\mathbb{Z}_q),\chi} \leq_p Approx.ISIS_{n,m,q,\alpha,\beta}$
- $ISIS_{n,n+m,q,\beta} \geq_p Approx.ISIS_{n,m,q,\alpha+\beta,\beta}$
- $ISIS_{n,n+m,q,\alpha+\beta} \leq_p Approx.ISIS_{n,m,q,\alpha,\beta}$

An *approximate trapdoor* for a public matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$ is a string that allows one to solve efficiently the Approx.ISIS and LWE problems w.r.t $\boldsymbol{A}$.

2.4 Recall : F-trapdoors [9]

The work of Chen, Genise and Mukherjee is itself based on the gadget-based trapdoor generation and preimage sampling algorithms of Micciancio and Peikert [21]. In their work on approximate trapdoors, Chen, Genise and Mukherjee create a new gadget matrix $\boldsymbol{F}$ which is an adaptation of the $\boldsymbol{G}$-gadget matrix from [21] where the l lower-orders entries are dropped.

The integer $b \geq 2$ defines the base for the $\boldsymbol{F}$-lattice and q the modulus ($k = \lceil \log_b q \rceil$).

The gadget matrix $\boldsymbol{F}$ is chosen such that it is easy to sample a short approximate preimage from $\Lambda_u^{\perp}(\boldsymbol{F})$. To do so, the approximate gadget-vector is set as $\boldsymbol{f}^t := (b^l, b^{l+1}, ..., b^{k-1})^t \in \mathbb{Z}_q^{(k-l)}$. Let $w = n(k-l)$ be the number of columns of the approximate gadget matrix $\boldsymbol{F} := \boldsymbol{I}_n \otimes \boldsymbol{f}^t \in \mathbb{Z}_q^{(n \times w)}$. The numbers of columns of $\boldsymbol{A}$ as defined below is $m := 2n+w$. (To sample approximately from $\Lambda_u^{\perp}(\boldsymbol{F})$, we first sample from $\Lambda_u^{\perp}(\boldsymbol{G})$ as described in [21].)

Recall that the public matrix $\boldsymbol{A}$ is defined as :

$$\boldsymbol{A} = [\bar{\boldsymbol{A}} | \boldsymbol{F} - \bar{\boldsymbol{A}}\boldsymbol{R}] \in \mathbb{Z}_q^{n \times m} \quad \text{with} \quad \bar{\boldsymbol{A}} = [\boldsymbol{I}_n, \hat{\boldsymbol{A}}] \in \mathbb{Z}_q^{n \times 2n}$$

where $\boldsymbol{R}$ is a secret, trapdoor matrix with small random entries. $\boldsymbol{R}$ is sampled from the distribution $\chi^{2n \times w}$ where $\chi \subseteq \mathbb{Z}$ is chosen to be a distribution such that $LWE_{n,n,q,\chi,U(\mathbb{Z}_q),\chi}$ is hard. $\hat{\boldsymbol{A}}$ is sampled from $U(\mathbb{Z}_q^{n \times n})$. Doing so, $\boldsymbol{A}$ is pseudorandom.

In order to sample a short approximate preimage of $\boldsymbol{u}$, we use the trapdoor $\boldsymbol{R}$ to map short approximate coset representatives of $\Lambda^{\perp}(\boldsymbol{F})$ to short approximate coset representatives of $\Lambda^{\perp}(\boldsymbol{A})$ by the relation

$$\boldsymbol{A} \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{I} \end{bmatrix} = \boldsymbol{F}$$

However, using this relation alone would leak information about the secret trapdoor $\boldsymbol{R}$. To avoid this, the perturbation-based Gaussian sampler technique of [21] is used. The covariance of the perturbation $\boldsymbol{p}$ is defined as the positive definite matrix $\boldsymbol{\Sigma}_p := s^2 \boldsymbol{I}_m - \sigma^2 \begin{bmatrix} \boldsymbol{R}\boldsymbol{R}^t & \boldsymbol{R} \\ \boldsymbol{R}^t & \boldsymbol{I} \end{bmatrix}$ where σ is at least $\eta_\epsilon(\Lambda^{\perp}(\boldsymbol{G}))$ and s is a parameter. This perturbation can be computed offline as $\boldsymbol{p} \leftarrow D_{\mathbb{Z}^m, \sqrt{\boldsymbol{\Sigma}_p}}$.

To approximately sample from $\Lambda_u^{\perp}(\boldsymbol{A})$, first define $\boldsymbol{v} = \boldsymbol{u} - \boldsymbol{A}\boldsymbol{p}$ and sample a vector $\boldsymbol{z}$ following the distribution $D_{\Lambda_v^{\perp}(F),\sigma}$ as described in [21] . Finally, the approximate preimage is set to be :

$$\boldsymbol{y} := \boldsymbol{p} + \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{I} \end{bmatrix} \boldsymbol{z}.$$

3 Hardness of higher-bit version problems

In this work, we aim at optimizing the memory space used to store elements in cryptographic constructions based on Ajtai's function upon some slight approximation. Our main idea is to use the base decomposition of elements in $\mathbb{Z}_q$. Using this decomposition, we discard low-weighted bits and only keep high-weighted ones.

To create such bitwise setting, we define the higher-bit approximate inhomogeneous short integer problem as well as the higher-bit near collision resistance of Ajtai's function. These instances are defined in regard to a higher-bit version of Ajtai's function.

3.1 Notations - High/low order bits functions

Let $b \geq 2$ be the base used in decomposition and $q \in \mathbb{Z}$ ($k = \lceil \log_b q \rceil$). Let d be an integer s.t $0 \leq d < k$. d is chosen as the turning point exponent between high order and low order bits.

Definition 9 (Decomposition in base b)**.** For $z \in \mathbb{Z}_q$, define the *decomposition in base b of z* as the elements $\{\alpha_{z,r}\}_{r=0}^{k-1}$ in $[|0, b-1|]$ s.t :

$$z = \sum_{r=0}^{k-1} \alpha_{z,r} b^r$$

Definition 10 (HighBits and LowBits functions)**.** For $z \in \mathbb{Z}_q$,

$$HighBits_d(z) = \sum_{r=d}^{k-1} \alpha_{z,r} b^r$$
$$LowBits_d(z) = \sum_{r=0}^{d-1} \alpha_{z,r} b^r$$

In introducing these definitions, our goal is to apply them to matrices in $\mathbb{Z}_q^{n \times m}$ and vectors in $\mathbb{Z}_q^m$ ($n, m \in \mathbb{N}$). Thus, we extend these definitions as in the following.

Definition 11. For $\boldsymbol{y} \in \mathbb{Z}_q^n$,

$$\boldsymbol{y}^H = (HighBits_d(y_i))_{0 \leq i < n} \quad \text{and} \quad \boldsymbol{y}^L = (LowBits_d(y_i))_{0 \leq i < n}$$

For $\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$,

$$\boldsymbol{A}^H = (HighBits_d(a_{i,j}))_{0 \leq i < n; 0 \leq j < m} \quad \text{and} \quad \boldsymbol{A}^L = (LowBits_d(a_{i,j}))_{0 \leq i < n; 0 \leq j < m}$$

3.2 Hardness of higher-bit approximate ISIS

Let $b \geq 2$ be the base used in decomposition.

Definition 12. For any $n, m \in \mathbb{N}$, $q \in \mathbb{Z}$, $\alpha, \beta \in \mathbb{R}$ and $d \in \mathbb{N}$ ($d < \lceil \log_b q \rceil$), define the higher-bit approximate inhomogeneous short integer solution problem H.Approx.ISIS$_{n,m,q,d,\alpha,\beta}$ as follows :

Given $\boldsymbol{A} \in \mathbb{Z}_{\frac{q}{b^d}}^{n \times m}$, $\boldsymbol{y} \in \mathbb{Z}_q^n$, find a vector $\boldsymbol{x} \in \mathbb{Z}^m$ such that $\|\boldsymbol{x}\| \leq \beta$ and there is a vector $\boldsymbol{z} \in \mathbb{Z}^n$ satisfying :

$$\|\boldsymbol{z}\| \leq \alpha \quad \text{and} \quad b^d \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{z} \pmod q.$$

We show that the higher-bit approximate ISIS problem is as hard as the standard ISIS. We know that the approximate ISIS is as hard as the standard ISIS under the right parameters setting (see section 2). Thus, we only need to show the reductions between the higher-bit approximate ISIS and the approximate ISIS.

Lemma 2.

$$\begin{aligned} Approx.ISIS_{n,m,q,\alpha,\beta} &\geq_p H.Approx.ISIS_{n,m,q,d,\alpha,\beta}; \\ H.Approx.ISIS_{n,m,q,d,\alpha,\beta} &\geq_p Approx.ISIS_{n,m,q,\alpha+\sqrt{n}b^d\beta,\beta}. \end{aligned}$$

Proof. The first reduction is straightforward.

Suppose there is a polynomial time algorithm $\mathcal{A}$ that solves H.Approx.ISIS$_{n,m,q,d,\alpha,\beta}$, we build a polynomial time algorithm $\mathcal{B}$ that solves Approx.ISIS$_{n,m,q,\alpha+\sqrt{n}b^d\beta,\beta}$. Given an Approx.ISIS instance ($\boldsymbol{A} \in \mathbb{Z}_q^{n \times m}$, $\boldsymbol{y} \in \mathbb{Z}_q^n$), $\mathcal{B}$ passes ($\frac{\boldsymbol{A}^H}{b^d} \in \mathbb{Z}_{\frac{q}{b^d}}^{n \times m}$, $\boldsymbol{y}$) to $\mathcal{A}$ and get $\boldsymbol{x} \in \mathbb{Z}_q^m$ such that :

$$\boldsymbol{A}^H \boldsymbol{x} = \boldsymbol{y} + \boldsymbol{z} \pmod q \quad \text{with} \quad \|\boldsymbol{x}\| \leq \beta, \|\boldsymbol{z}\| \leq \alpha.$$

We do :

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{z} + \boldsymbol{A}^L \boldsymbol{x} \pmod q$$

Moreover,

$$\begin{aligned} \|\boldsymbol{z} + \boldsymbol{A}^L \boldsymbol{x}\| &\leq \|\boldsymbol{z}\| + \|\boldsymbol{A}^L \boldsymbol{x}\| \\ &\leq \alpha + \|\boldsymbol{A}^L\| \|\boldsymbol{x}\| \\ &\leq \alpha + \sqrt{n} b^d \beta \end{aligned}$$

since all coefficients in $\boldsymbol{A}^L$ are less than b^d.

So $\boldsymbol{x}$ is a valid solution to Approx.ISIS$_{n,m,q,\alpha+\sqrt{n}b^d\beta,\beta}$.

□

Theorem 1.

$$ISIS_{n,n+m,q,\beta} \geq_p H.Approx.ISIS_{n,m,q,d,\alpha+\beta,\beta};$$
$$H.Approx.ISIS_{n,m,q,d,\alpha,\beta} \geq_p ISIS_{n,n+m,q,\alpha+(\sqrt{n}b^d+1)\beta}.$$

Proof. We can prove this Theorem by using both Lemma 2 above and reductions from [9] (see section 2).

□

3.3 The near collision resistance of higher-bit Ajtai's function

Let $b \geq 2$ be the base used in decomposition.

Lemma 3 (The near-collision-resistance of Ajtai's function [9])**.** *For any $n, m, q \in \mathbb{N}$ and $\alpha, \beta \in \mathbb{R}$,*

***If** there is an efficient adversary $\mathcal{A}$ that given $\boldsymbol{A} \leftarrow U(\mathbb{Z}_q^{n\times m})$, finds $\boldsymbol{x}_1 \neq \boldsymbol{x}_2 \in \mathbb{Z}^m$ such that :*

$$\|\boldsymbol{x}_1\| \leq \beta \quad and \quad \|\boldsymbol{x}_2\| \leq \beta \quad and \quad \|\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2 \pmod q\| \leq 2\alpha$$

***Then** there is an efficient adversary $\mathcal{B}$ that solves $SIS_{n,n+m,q,2(\alpha+\beta)}$.*

Lemma 4. *For any $n, m, q \in \mathbb{N}$, $\alpha, \beta \in \mathbb{R}$ and $d \in \mathbb{N}$ $(d < \lceil \log_b q \rceil)$,*

***If** there is an efficient adversary $\mathcal{A}$ that given $\boldsymbol{A} \leftarrow U(\mathbb{Z}_{\frac{q}{b^d}}^{n\times m})$, finds $\boldsymbol{x}_1 \neq \boldsymbol{x}_2 \in \mathbb{Z}^m$ such that :*

$$\|\boldsymbol{x}_1\| \leq \beta \quad and \quad \|\boldsymbol{x}_2\| \leq \beta \quad and \quad \|b^d\boldsymbol{A}\boldsymbol{x}_1 - b^d\boldsymbol{A}\boldsymbol{x}_2 \pmod q\| \leq 2\alpha$$

***Then** there is an efficient adversary $\mathcal{B}$ that given $\boldsymbol{A} \leftarrow U(\mathbb{Z}_q^{n\times m})$, finds $\boldsymbol{x}_1 \neq \boldsymbol{x}_2 \in \mathbb{Z}^m$ such that :*

$$\|\boldsymbol{x}_1\| \leq \beta \quad and \quad \|\boldsymbol{x}_2\| \leq \beta \quad and \quad \|\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2 \pmod q\| \leq 2(\alpha + \sqrt{n}b^d\beta)$$

Proof. Suppose $\mathcal{B}$ gets $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$. $\mathcal{B}$ sends $\frac{\boldsymbol{A}^H}{b^d}$ to $\mathcal{A}$ and gets back $\boldsymbol{x}_1 \neq \boldsymbol{x}_2 \in \mathbb{Z}^m$ such that :

$$\|\boldsymbol{x}_1\| \leq \beta \quad \text{and} \quad \|\boldsymbol{x}_2\| \leq \beta \quad \text{and} \quad \|\boldsymbol{A}^H\boldsymbol{x}_1 - \boldsymbol{A}^H\boldsymbol{x}_2 \pmod q\| \leq 2\alpha$$

We define $\boldsymbol{z} = \boldsymbol{A}^H\boldsymbol{x}_1 - \boldsymbol{A}^H\boldsymbol{x}_2 \pmod q$,

$$\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2 = \boldsymbol{z} + \boldsymbol{A}^L\boldsymbol{x}_1 - \boldsymbol{A}^L\boldsymbol{x}_2 \pmod q$$

Thus ,

$$\begin{aligned}\|\boldsymbol{A}\boldsymbol{x}_1 - \boldsymbol{A}\boldsymbol{x}_2 \pmod q\| &\leq \|\boldsymbol{z}\| + \|\boldsymbol{A}^L\boldsymbol{x}_1\| + \|\boldsymbol{A}^L\boldsymbol{x}_2\| \\ &\leq 2(\alpha + \sqrt{n}b^d\beta)\end{aligned}$$

□

Theorem 2 (The near collision resistance of higher-bit Ajtai's function)**.** *For any $n, m, q \in \mathbb{N}$, $\alpha, \beta \in \mathbb{R}$ and $d \in \mathbb{N}$ ($d < \lceil \log_b q \rceil$),*

***If** there is an efficient adversary $\mathcal{A}$ that given $\boldsymbol{A} \leftarrow U(\mathbb{Z}_{\frac{q}{b^d}}^{n\times m})$, finds $\boldsymbol{x}_1 \neq \boldsymbol{x}_2 \in \mathbb{Z}^m$ such that :*

$$\|\boldsymbol{x}_1\| \leq \beta \quad \textit{and} \quad \|\boldsymbol{x}_2\| \leq \beta \quad \textit{and} \quad \|b^d\boldsymbol{A}\boldsymbol{x}_1 - b^d\boldsymbol{A}\boldsymbol{x}_2 \pmod q\| \leq 2\alpha$$

***Then** there is an efficient adversary $\mathcal{B}$ that solves $SIS_{n,n+m,q,2[\alpha+(\sqrt{n}b^d+1)\beta]}$*

Proof. We can prove this Theorem using both lemma 3 [9] and lemma 4 above. □

4 New construction - Main idea

We construct an application of the higher-bit approximate ISIS problem. Our goal is to reduce the sizes of both the matrix $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$ generated with an approximate trapdoor as in the algorithms of [9], and of the sampled approximate preimage $\boldsymbol{y} \in \mathbb{Z}_q^m$ by Ajtai's Function defined by $\boldsymbol{A}$.

Let b be the base for the matrix $\boldsymbol{F}$ of [9] with parameter l. As mentioned above, we will be using the decomposition in base b ($k = \lceil \log_b q \rceil$). Let d be an integer s.t $0 \leq d \leq l$.

4.1 Modification in the public matrix $\boldsymbol{A}$

Modification in the construction.

We recall that in [9], the public matrix $\boldsymbol{A}$ is defined as :

$$\boldsymbol{A} := [\bar{\boldsymbol{A}}|\boldsymbol{F} - \bar{\boldsymbol{A}}\boldsymbol{R}] \in \mathbb{Z}_q^{n\times m}$$

where $\boldsymbol{F}$ is the public approximate gadget matrix and $\boldsymbol{R}$ is the approximate trapdoor associated with the Ajtai's Function defined by $\boldsymbol{A}$ (see section 2.4).

The selected modification on $\boldsymbol{A}$ is straightforward. We construct $\boldsymbol{A}^{new} \in \mathbb{Z}_{\frac{q}{b^d}}^{n\times m}$ by doing the same as above and applying the HighBits function on $\boldsymbol{A}$.

$$\boldsymbol{A}^{new} = \frac{\boldsymbol{A}^H}{b^d} \quad \text{where} \quad \boldsymbol{A}^H = [\bar{\boldsymbol{A}}^H|(\boldsymbol{F} - \bar{\boldsymbol{A}}\boldsymbol{R})^H]$$

In the following work, we need to isolate $\boldsymbol{F}$. We observe that $\boldsymbol{F}$ is already in a higher-bit form since $\boldsymbol{F} = \boldsymbol{I}_n \otimes (b^l, b^{l+1}, ..., b^{k-1})^t$ and $d \leq l$.

We use this property to express $\boldsymbol{A}^H$ while keeping $\boldsymbol{F}$ untouched :

$$\boldsymbol{A}^H = [\bar{\boldsymbol{A}}^H | \boldsymbol{F} + (-\bar{\boldsymbol{A}}\boldsymbol{R})^H]$$

It is easy to see that $\boldsymbol{A}^{new} \in \mathbb{Z}_{\frac{q}{b^d}}^{n \times m}$.

Optimization in the public matrix size. Using this modification, we save $n \times m \times d\lceil \log_2 b \rceil$ bits in the public matrix $\boldsymbol{A}^{new}$ memory space.

4.2 Repercussion on the security and underlying problem

This change in the public matrix $\boldsymbol{A}$ implies a modification in the hardness of the underlying problem. In this construction, security relies on the higher-bit approximate $\text{ISIS}_{n,m,q,d,\alpha,\beta}$ problem. As seen in Theorem 1, there is a reduction from this problem to the $\text{SIS}_{n,m,q,\alpha+\sqrt{n}b^d\beta}$ problem. For same α and β as in the original construction from [9], we need to deal with an additional $\sqrt{n}b^d$ factor in the SIS problem solution length.

5 New construction - Algorithms

Let n, m, q, k and d be defined as in section 4.

In the following section, we present our compact approximate trapdoor generation algorithm and approximate preimage sampling algorithm. Our algorithms use those from [9]. Our method generates a pseudorandom $\boldsymbol{A} \in \mathbb{Z}_{\frac{q}{b^d}}^{n \times m}$ along with an approximate trapdoor $\boldsymbol{R}$ which allows to sample an approximate preimage $\boldsymbol{y} \in \mathbb{Z}_{\frac{q}{b^d}}^{m}$ for higher-bit Ajtai's function defined by $\boldsymbol{A}$.

5.1 The higher-bit version algorithms

We consider that HIGHBITS and LOWBITS are two functions implemented as described in section 3.1.

Algorithm 1: HIGHBITS.APPROX.TRAPGEN$_\chi$

Input: Security parameter λ.

Output: Matrix-approximate trapdoor pair $(\boldsymbol{A}, \boldsymbol{R}) \in \mathbb{Z}_{\frac{q}{b^d}}^{n \times m} \times \mathbb{Z}^{2n \times w}$; matrix $\boldsymbol{A}_0^L \in \mathbb{Z}_{b^d}^{n \times m}$

1 $(\boldsymbol{A}_0, \boldsymbol{R}) =$ APPROX.TRAPGEN$_\chi(\lambda)$ // Algorithm from [9]

2 $\boldsymbol{A}_0^H, \boldsymbol{A}_0^L = HIGHBITS(\boldsymbol{A}_0, d), LOWBITS(\boldsymbol{A}_0, d)$

3 $\boldsymbol{A} = \frac{\boldsymbol{A}_0{}^H}{b^d}$

4 **return** $((\boldsymbol{A}, \boldsymbol{R}), \boldsymbol{A}_0^L)$

Algorithm 2: HIGHBITS.APPROX.SAMPLEPRE

Input: $(\boldsymbol{A}, \boldsymbol{A}_0^L, \boldsymbol{R}, \boldsymbol{u}, \text{s})$.

Output: An approximate preimage of $\boldsymbol{u} \in \mathbb{Z}_q^n$ for $b^d\boldsymbol{A} : \boldsymbol{y} \in \mathbb{Z}_{\frac{q}{b^d}}^m$

1 $\boldsymbol{y}_0 =$ APPROX.SAMPLEPRE$(b^d\boldsymbol{A} + \boldsymbol{A}_0^L, \boldsymbol{R}, \boldsymbol{u}, \text{s})$ // $\boldsymbol{A}_0 = b^d\boldsymbol{A} + \boldsymbol{A}_0^L$; // Algorithm from [9]

2 $\boldsymbol{y} = LOWBITS(\boldsymbol{y}_0, k - d)$

3 **return** $\boldsymbol{y}$

Figure 1: Pseudocode for the higher-bit version approximate trapdoor generation and approximate preimage sampling algorithms. The distribution χ is chosen so that $\text{LWE}_{n,n,q,\chi,U(\mathbb{Z}_q),\chi}$ is hard. For the sake of optimization in Algorithm 2, we need to set $q = b^k$.

Algorithm 1 . This algorithm is instantiated such as described in section 4. The overall goal is to use only the high-weighted bits of the previous public matrix $\boldsymbol{A}_0$ as our new public key. Doing so, we induce a b^d-approximation on every coefficient of the resulting public key $\boldsymbol{A}$ when compared to $\boldsymbol{A}_0$.

One should note that this algorithm does not only generate a matrix-approximate trapdoor pair. It also returns the low-weighted bits of the original matrix $\boldsymbol{A}_0$. This information is given to the approximate preimage sampling algorithm. We should notice that information on $\boldsymbol{A}_0^L$ leaks through the error distribution. However this is not a problem because it does not leak information on the secret trapdoor $\boldsymbol{R}$ since $\boldsymbol{A}_0^L$ is pseudorandom as we will see in subsection 5.2.

Algorithm 2. This algorithm samples an approximate preimage $\boldsymbol{y} \in \mathbb{Z}_{\frac{q}{b^d}}^m$ of $\boldsymbol{u} \in \mathbb{Z}_q^n$ by the higher-bit Ajtai's function $\boldsymbol{A} \in \mathbb{Z}_{\frac{q}{b^d}}^{n \times m}$.

First, we sample an approximate preimage $\boldsymbol{y}_0 \in \mathbb{Z}_q^m$ of the Ajtai's function defined by $\boldsymbol{A}_0$ using the algorithm from [9].

Secondly, in order to reduce the signature size, we use a little trick. It relies on the following lemma :

Lemma 5. *For $z \in \mathbb{Z}_q$, $q = b^k$, and integers d, j such that $j \geq d$,*

$$b^j z = b^j LowBits_{k-d}(z) \pmod q$$

Proof.

$$b^j z = b^{j-d} \sum_{r=d}^{k-1} \alpha_{z,r-d} b^r \pmod q = b^j \sum_{r=0}^{k-1-d} \alpha_{z,r} b^r \pmod q$$

□

Using lemma 5 and the fact that $b^d \boldsymbol{A}$ is in a higher-bit form, we see that the d highest bits of $\boldsymbol{y}_0$ have no impact on the product $b^d \boldsymbol{A}\boldsymbol{y}_0$.

Theorem 3. *For* $\boldsymbol{A} \in \mathbb{Z}_{\frac{q}{b^d}}^{n \times m}$ *and* $\boldsymbol{y}_0 \in \mathbb{Z}^m$, $q = b^k$,

$$b^d \boldsymbol{A}\boldsymbol{y} = b^d \boldsymbol{A}\boldsymbol{y}_0 \pmod q \quad \textit{where} \quad \boldsymbol{y} = LowBits_{k-d}(\boldsymbol{y}_0)$$

Therefore, our modification in the public key $\boldsymbol{A}$ allows for an optimization in the approximate preimage.

Remark. The norm of the approximate preimage is decreased by this modification. Thus, if $\boldsymbol{y}_0$ is short then $\boldsymbol{y}$ is too.

Remark. This optimization needs the additional condition $q = b^k$. If this condition is not met, we should use the approximate preimage $\boldsymbol{y}_0$ from [9].

Optimization in the preimage size. Using this modification, we save $m \times d\lceil \log_2 b \rceil$ bits in the approximate preimage memory space.

Remark. An idea to optimize the preimage size even more would be to apply the HighBits function on $\boldsymbol{y}$ in the same way as for $\boldsymbol{A}$. However, doing so would increase a lot more the error term and thus impact security. We decide not to add such modification as a trade-off between size and security.

Error term. We define the error $\boldsymbol{e} \in \mathbb{Z}_q^n$ as $\boldsymbol{e} = \boldsymbol{u} - b^d \boldsymbol{A}\boldsymbol{y} \pmod q$.
$\boldsymbol{e}_0$ defines the error term induced by $\boldsymbol{y}_0$ i.e $\boldsymbol{e}_0 = \boldsymbol{u} - \boldsymbol{A}_0 \boldsymbol{y}_0 \pmod q$.

The error term $\boldsymbol{e}$ can be expressed as :

$$\boxed{\boldsymbol{e} = \boldsymbol{e}_0 + \boldsymbol{e}_{new} \pmod q} \quad \text{where} \quad \boldsymbol{e}_{new} = \boldsymbol{A}_0^L \boldsymbol{y}_0 \pmod q$$

Proof.

$$\begin{aligned}
\boldsymbol{e} &= \boldsymbol{u} - b^d \boldsymbol{A}\boldsymbol{y} \pmod q \\
&= \boldsymbol{u} - \boldsymbol{A}_0^H \boldsymbol{y}_0 \pmod q \quad \textit{Theorem 3} \\
&= \boldsymbol{u} - (\boldsymbol{A}_0 - \boldsymbol{A}_0^L)\boldsymbol{y}_0 \pmod q \\
&= \underbrace{\boldsymbol{u} - \boldsymbol{A}_0 \boldsymbol{y}_0}_{\boldsymbol{e}_0} + \underbrace{\boldsymbol{A}_0^L \boldsymbol{y}_0}_{\boldsymbol{e}_{new}} \pmod q
\end{aligned}$$

□

Remark. If we had chosen to calculate $\boldsymbol{y}_0$ with regard to $b^d\boldsymbol{A}$ rather than $\boldsymbol{A}_0$, the error term $\boldsymbol{e}$ would be $\boldsymbol{e} = \boldsymbol{e}_0 + \boldsymbol{A}_0^L \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{I} \end{bmatrix} \boldsymbol{z}$ ($\boldsymbol{z}$ is an approximate preimage for $\boldsymbol{F}$). We observe that $\boldsymbol{A}_0^L \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{I} \end{bmatrix} = \bar{\boldsymbol{A}}^L\boldsymbol{R} + (-\bar{\boldsymbol{A}}\boldsymbol{R})^L + \boldsymbol{F}$. Thus, information on the secret $\boldsymbol{R}$ would leak from the distribution of $\boldsymbol{e}$. Even though the norm of $\boldsymbol{e}$ is decreased by this method, the security is compromised.

5.2 Study of the resulting distributions

The results of this subsection are summarized in the following Theorem.

Theorem 4. *There exist probabilistic, polynomial time algorithms HIGHBITS.APPROX.TRAPGEN$_\chi$ and HIGHBITS.APPROX.SAMPLEPRE satisfying the following :*

1. *HIGHBITS.APPROX.TRAPGEN$_\chi(\lambda)$ returns a matrix-approximate trapdoor pair $(\boldsymbol{A},\boldsymbol{R}) \in \mathbb{Z}_{\frac{q}{b^d}}^{n\times m} \times \mathbb{Z}^{2n\times n(k-l)}$ along with a matrix $\boldsymbol{A}_0^L \in \mathbb{Z}_{b^d}^{n\times m}$.*

   The matrices $\boldsymbol{A}$ and $\boldsymbol{A}_0^L$ are pseudorandom assuming the hardness of $LWE_{n,n,q,\chi,U(\mathbb{Z}_q),\chi}$.

2. *Let $((\boldsymbol{A},\boldsymbol{R}),\boldsymbol{A}_0^L)$ be generated by HIGHBITS.APPROX.TRAPGEN$_\chi(\lambda)$. The following two distributions are statistically indistinguishable :*

$$\begin{aligned}\{(\boldsymbol{A},\boldsymbol{y},\boldsymbol{u},\boldsymbol{e}) :& \boldsymbol{u} \leftarrow U(\mathbb{Z}_q^n),\\ & \boldsymbol{y} \leftarrow HIGHBITS.APPROX.SAMPLEPRE(\boldsymbol{A},\boldsymbol{A}_0^L,\boldsymbol{R},\boldsymbol{u},s),\\ & \boldsymbol{e} = \boldsymbol{u} - b^d\boldsymbol{A}\boldsymbol{y} \pmod q\}\end{aligned}$$

and

$$\begin{aligned}\{(\boldsymbol{A},\boldsymbol{y},\boldsymbol{u},\boldsymbol{e}) :& \boldsymbol{y}_0 \leftarrow D_{\mathbb{Z}^m,s}, \boldsymbol{e}_0 \leftarrow D_{\mathbb{Z}^n,\sigma\sqrt{(b^{2l}-1)/(b^2-1)}} \pmod q,\\ & \boldsymbol{y} = LowBits_{k-d}(\boldsymbol{y}_0), \boldsymbol{e} = \boldsymbol{e}_0 + \boldsymbol{A}_0^L\boldsymbol{y}_0 \pmod q,\\ & \boldsymbol{u} = b^d\boldsymbol{A}\boldsymbol{y} + \boldsymbol{e} \pmod q\}\end{aligned}$$

for any $\sigma \geq \sqrt{b^2+1}.w(\sqrt{\log n})$ and $s \gtrsim \sqrt{b^2+1}\frac{s_1^2(\boldsymbol{R})}{s_{2n}(\boldsymbol{R})}\eta_\epsilon(\mathbb{Z}^{nk})$.

Proof. The proof is described in the end of this section. We use the distributions study results in Theorem 4 from [9].

□

5.2.1 Distributions of $\boldsymbol{A}$ and $\boldsymbol{A}_0^L$.

Lemma 6. *For any matrix $\boldsymbol{M}$ with distribution U($\mathbb{Z}_q^{n\times m}$),*

$\frac{\boldsymbol{M}^H}{b^d}$ follows the distribution U($\mathbb{Z}_{\frac{q}{b^d}}^{n\times m}$) and $\boldsymbol{M}^L$ follows the distribution U($\mathbb{Z}_{b^d}^{n\times m}$). The distributions of $\frac{\boldsymbol{M}^H}{b^d}$ and $\boldsymbol{M}^L$ are independent.

Proof. Let i,j be two integers s.t $0 \leq i \leq n$ and $0 \leq j \leq m$. Let x be an integer in $[|0, \frac{q}{b^d} - 1|]$.

$$\begin{aligned}\mathbb{P}\left(\frac{m_{i,j}^H}{b^d} = x \pmod{\frac{q}{b^d}}\right) &= \mathbb{P}(HighBits_d(m_{i,j}) = b^d x \pmod q) \\ &= \sum_{l_0 \in [|0, b^d-1|]} \mathbb{P}(m_{i,j} = b^d x + l_0 \pmod q) \\ &= \sum_{l_0 \in [|0, b^d-1|]} \frac{1}{q} = \frac{b^d}{q} = \frac{1}{\frac{q}{b^d}}\end{aligned}$$

Using the same kind of reasoning, we can find that for any x $\in [|0, b^d - 1|]$, $\mathbb{P}(m_{i,j}^L = x \pmod{b^d}) = \frac{1}{b^d}$

$\boldsymbol{M}^H$ and $\boldsymbol{M}^L$ do not share any random sources thus their distributions are independent.

□

We know that $\boldsymbol{A}_0$ is computationally indistinguishable from random assuming $\text{LWE}_{n,n,q,\chi,U(\mathbb{Z}_q),\chi}$ [9].

Thus, using lemma 6, we deduce that $\boldsymbol{A} \approx_s U(\mathbb{Z}_{\frac{q}{b^d}}^{n\times m})$ and $\boldsymbol{A}_0^L \approx_s U(\mathbb{Z}_{b^d}^{n\times m})$.

5.2.2 Distribution of $\boldsymbol{y}$.

We know that the distribution of
$\boldsymbol{y}_0 \leftarrow APPROX.SAMPLEPRE(\boldsymbol{A_0}, \boldsymbol{R}, \boldsymbol{u}, s)$ is statistically indistinguishable from $\boldsymbol{y}_0 \leftarrow D_{\mathbb{Z}^m, s}$ for a random target. Since $\boldsymbol{y} = LowBits_{k-d}(\boldsymbol{y}_0)$, we can say that the distribution of $\boldsymbol{y}$ is statistically indistinguishable from $\{\boldsymbol{y}_0 \leftarrow D_{\mathbb{Z}^m, s}, \boldsymbol{y} = LowBits_{k-d}(\boldsymbol{y}_0)\}$ for a random target.

Thus, the distribution of $\boldsymbol{y}$ is simulatable without knowing the secret $\boldsymbol{R}$ nor the public key $\boldsymbol{A}$.

5.2.3 Distribution of $\boldsymbol{e}$.

We know that the distribution of
$\{\boldsymbol{y}_0 \leftarrow APPROX.SAMPLEPRE(\boldsymbol{A_0}, \boldsymbol{R}, \boldsymbol{u}, s), \boldsymbol{e}_0 = \boldsymbol{u} - \boldsymbol{A}_0\boldsymbol{y}_0\}$ is statistically indistinguishable from $\{\boldsymbol{y}_0 \leftarrow D_{\mathbb{Z}^m, s}, \boldsymbol{e}_0 \leftarrow D_{\mathbb{Z}^n, \sigma\sqrt{(b^{2l}-1)/(b^2-1)}} \pmod q\}$ for a random target $\boldsymbol{u}$. Since $\boldsymbol{e} = \boldsymbol{e}_0 + \boldsymbol{A}_0^L\boldsymbol{y}_0 \pmod q$, we can say that the

distribution of $\boldsymbol{e}$ is statistically indistinguishable from $\{\boldsymbol{y}_0 \leftarrow D_{\mathbb{Z}^m,s}, \boldsymbol{e}_0 \leftarrow D_{\mathbb{Z}^n,\sigma\sqrt{(b^{2l}-1)/(b^2-1)}} \pmod q, \boldsymbol{e} = \boldsymbol{e}_0 + \boldsymbol{A}_0^L\boldsymbol{y}_0 \pmod q\}$ for a random target.

Thus, the distribution of $\boldsymbol{e}$ is simulatable without knowing the secret $\boldsymbol{R}$. Compared to [9], we need to know $\boldsymbol{A}_0^L$ to simulate $\boldsymbol{e}$. However, as seen in 5.2.1, $\boldsymbol{A}_0^L$ is computationally indistinguishable from random and thus do not leak information about $\boldsymbol{R}$.

6 Hash-and-Sign Signature

This section is dedicated to the construction of a sEU-CMA secure [9] hash-and-sign signature scheme instantiated with the algorithms and parameters from figure 1. Let $\sigma, s \in \mathbb{R}^+$ be the discrete Gaussian widths of the distributions over the cosets of $\Lambda_q^\perp(\boldsymbol{G})$ [21] and approximate $\Lambda_q^\perp(\boldsymbol{A}_0)$ [9] respectively. We choose a distribution χ to sample $\boldsymbol{R}$ so that $\mathrm{LWE}_{n,n,q,\chi,U(\mathbb{Z}_q),\chi}$ is hard.

6.1 Construction of a hash-and-sign signature

The following construction is written in the same way as the one in section 5 from [9]. This shows how it is adjusted to fit the "higher-bit setting".

Construction 1. *Given the algorithms from Theorem 4, a hash function $H = \{H_\lambda : \{0,1\}^* \to \mathbb{Z}_q^n\}$ modeled as a random oracle, we build a signature scheme as follows.*

- *Gen(1^λ) : The key-generation algorithm samples $\boldsymbol{A} \in \mathbb{Z}_{\frac{q}{b^d}}^{n\times m}$ together with its (α,β)-approximate trapdoor $\boldsymbol{R}$ and the matrix $\boldsymbol{A}_0^L \in \mathbb{Z}_{b^d}^{n\times m}$ from HIGHBITS.APPROX.TRAPGEN$_\chi(\lambda)$. It outputs $\boldsymbol{A}$ as the verification key, keeps $\boldsymbol{R}$ as the secret signing key and gives $\boldsymbol{A}_0^L$ to the signing algorithm.*
- *Sig($\boldsymbol{R}, \boldsymbol{m}$) : The signing algorithm checks if the message-signature pair ($\boldsymbol{m}$,$\boldsymbol{x}_m$) has been produced before. If so, it outputs $\boldsymbol{x}_m$ as the signature of $\boldsymbol{m}$; if not, it computes $\boldsymbol{u} = H(\boldsymbol{m})$, and samples an approximate preimage $\boldsymbol{x}_m \leftarrow$ HIGHBITS.APPROX.SAMPLEPRE$(\boldsymbol{A}, \boldsymbol{A}_0^L, \boldsymbol{R}, \boldsymbol{u}, s)$. It outputs $\boldsymbol{x}_m$ as the signature and stores ($\boldsymbol{m}$,$\boldsymbol{x}_m$) in the list.*
- *Ver($\boldsymbol{A}, \boldsymbol{m}, \boldsymbol{x}$) : The verification algorithm checks if $\|\boldsymbol{x}\| \le \beta$ and $\|b^d\boldsymbol{A}\boldsymbol{x} - H(\boldsymbol{m})\| \le \alpha$. If so, it outputs accept; otherwise, it outputs reject.*

6.2 Correctness

It is straightforward to verify that construction 1 is correct with overwhelming probability by the settings of the parameters and definitions of our algorithms.

6.3 Proof of security

For a random target, the preimage and error term are simulatable from distributions without knowing the secret key $\boldsymbol{R}$ (see Theorem 4). We denote these distributions by D_{pre} and D_{err}. To prove that our construction satisfies sEU-CMA security, we rely on Theorem 2 about "higher-bit near-collision-resistance" property for Ajtai's function. We use the same definition for sEU-CMA security as defined in [9].

Theorem 5. *Construction 1 is strongly existentially unforgeable under a chosen-message attack in the random oracle model assuming the hardness of* $SIS_{n,n+m,q,2[\alpha+(\sqrt{n}b^d+1)\beta]}$ *and* $LWE_{n,n,q,\chi,U(\mathbb{Z}_q),\chi}$.

Proof. Assume that there is an adversary $\mathcal{A}$ which breaks the sEU-CMA security of construction 1 in polynomial time. We describe a polynomial time adversary $\mathcal{B}$ invoking $\mathcal{A}$ that breaks the higher-bit near-collision-resistance of Ajtai's function, which is as hard as $\mathrm{SIS}_{n,n+m,q,2[\alpha+(\sqrt{n}b^d+1)\beta]}$ (Theorem 2).

$\mathcal{B}$ receives the matrix $\boldsymbol{A}$ as a challenge for "the higher-bit near-collision-resistance of Ajtai's function". $\mathcal{B}$ runs $\mathcal{A}$ on input pk $\boldsymbol{A}$. $\mathcal{B}$ answers hash queries to random oracle H and signing queries as follows. We note that its answers are indistinguishable from the real ones due to the properties of D_{pre} and D_{err}, and that a real public key is indistinguishable from random under $\mathrm{LWE}_{n,n,q,\chi,U(\mathbb{Z}_q),\chi}$.

Simulation of hash queries. We assume that $\mathcal{B}$ has chosen a random $\boldsymbol{A}_0^L$ to calculate D_{err}. $\mathcal{A}$'s hash query $H(\boldsymbol{m})$ on a message $\boldsymbol{m}$ is answered by $\mathcal{B}$ as follows : $\mathcal{B}$ samples $\boldsymbol{x} \leftarrow D_{pre}$, gives $\boldsymbol{u} := b^d\boldsymbol{A}\boldsymbol{x} + D_{err} \pmod q$ to $\mathcal{A}$ as $H(\boldsymbol{m})$.

$\mathcal{B}$ stores $(\boldsymbol{m},\boldsymbol{u})$ in the random oracle storage, $(\boldsymbol{m},\boldsymbol{x})$ in the message-signature pair storage.

Simulation of signing queries. Assume that on $\mathcal{A}$'s signature query $\boldsymbol{m}$, $\boldsymbol{m}$ has been queried to the random oracle before. $\mathcal{B}$ generates the signature $\boldsymbol{x}$ by finding $(\boldsymbol{m},\boldsymbol{x})$ in the message-signature pair storage.

Forgery. Generality is equivalent to assumption that before $\mathcal{A}$'s attempt to forge a signature on $\boldsymbol{m}^*$, $\mathcal{A}$ has queried H on $\boldsymbol{m}^*$. We denote $(\boldsymbol{m}^*,\boldsymbol{u}^*)$ and $(\boldsymbol{m}^*,\boldsymbol{x}^*)$ as the pairs prepared by $\mathcal{B}$ in the random oracle storage and message-signature pair storage respectively. $\mathcal{A}$ forges a signature $\boldsymbol{x}$ on $\boldsymbol{m}^*$. By the definition of a correct signature, we have $\|b^d\boldsymbol{A}(\boldsymbol{x}-\boldsymbol{x}^*) \pmod q\| \leq 2\alpha$.

In the case where $\boldsymbol{m}^*$ has been queried to the signing oracle, $\boldsymbol{x} \neq \boldsymbol{x}^*$ by the definition of a successful forgery. Otherwise, we know that $D_{\mathbb{Z}^m,s}$ is set with high min-entropy. Thus, D_{pre} is also with high min-entropy since D_{pre} means compressing b^d points to one point when using $D_{\mathbb{Z}^m,s}$. So, $\boldsymbol{x} \neq \boldsymbol{x}^*$ with overwhelming probability.

□

6.4 Implementation and analysis

The results in Theorem 4 induce the following length bounds on the signature $\boldsymbol{x}$ and error term $\boldsymbol{e}$: $||\boldsymbol{x}|| \leq s$ and $||\boldsymbol{e}|| \leq b^l\sigma + \sqrt{n}b^d s$.

We need to respect these bounds to set the parameters α and β of the underlying security problem. Thus, combining with the results in Theorem 5, we observe a trade-off between security and memory space. This trade-off is due to the increase in the norm of a solution to the SIS problem. It is summarized in figure 2 for the matrix setting.

	$\boldsymbol{F}$-trapdoor [9]	This work
Norm of a short solution in SIS problem in the underlying SIS problem	$2(s + b^l\sigma)$	$2(s + b^l\sigma) + 4\sqrt{n}b^d s$
Signature size (in bits)	$m \times k \times log_2(b)$	$m \times (k - d) \times log_2(b)$
Public key size (in bits)	$m \times n \times k \times log_2(b)$	$m \times n \times (k - d) \times log_2(b)$

Figure 2: The parameters are for a fixed lattice dimension n, vector dimension m, a base b, a modulus q where $k = \lceil \log_b q \rceil$. Parameters l, s and σ are the same as in [9].

Proof-of-concept implementation. Due to this trade-off, we need to analyse the benefits of our construction for different parameters sets. To do so, we implement our construction for different concrete parameters. The code used in this implementation is provided by Dr. Chen [8]. We get our security assuming the hardness of Ring-LWE and Ring-SIS. Our goal in doing this implementation is to compare our construction for different parameters choices with the two best reference implementations from [9]. We realize an exhaustive search on all parameters combinations from the following sets :

- $n \in \{512; 1024; 2048\}$
- $b \in \{2; 4; 8\}$
- $k \in \{16; 20; 22; 24\}/log_2 b$
- $l \in \{1; 2; ...; k - 1\}$
- $d \in \{1; 2; ...; l\}$

In the end, we obtained 1245 experiment results each with a different parameters set. We conducted a comparison between all of these results in terms of security and storage to find the best choices.

6.4.1 Analysis

In Figure 3 , we list experiment results for three selected groups of parameters.

	$\boldsymbol{F}$-trapdoor [9]	$\boldsymbol{F}$-trapdoor [9]	This work	This work	This work
n	512	1024	512	1024	1024
$k = \lceil \log_b q \rceil$	8	9	16	16	8
b	4	4	2	2	4
l	4	5	11	11	3
d	-	-	11	11	3
τ	2.6	2.8	2.6	2.8	2.8
s	2505.6	3733.1	1453.0	2163.9	3989.3
m	3072	6144	3584	7168	7168
$\|\|y\|\|_2$	138244.3	296473.0	1072.2	1535.5	50533.7
$\|\|e\|\|_2$	20627.9	1502259.7	428806.9	607601.6	596704.5
PK (kB)	5.12	11.52	1.92	3.84	7.68
Sig (kB)	4.5	9.4	2.25	4.5	8.98
LWE	104.7	192.7	104.7	192.7	192.7
AISIS	87.8	183.7	75.0	155.4	153.9

Figure 3: Some concrete parameters. LWE and AISIS refers to the security levels of breaking the associated problems. $||y||$ and $||e||$ are the norms of the preimage and error term. τ is the Gaussian width of R.

Figure 3 shows that, *for a same security parameter n,* we can expect our construction to reduce the public key and signature sizes at least by half at the expense of a reasonable drop in the security level. Our implementation shows that an estimation of 75-bit security could be achieved for a public key size of 1.92kB and a signature size of 2.25kB. Our construction brings a quite important gain in storage.

For a same security parameter n, it is difficult to obtain different levels of security as it is quite fixed by it. Furthermore, we do not have a lot of choices for n. Indeed, in order to increase this parameter (which leads to better security), there is no choice than doubling it. This problem is the same in the construction by Chen, Denise and Mukherjee. Thus, the higher-bit approximate setting can also be considered as a solution for obtaining other security levels than [9] along with optimized object sizes.

Last but not least, if we increase the security parameter n compared to [9], we obtain a better security along with better public key and signature sizes. This can be seen as a *win-win* scenario for security and storage. For example, we obtain an estimation of 155-bit security for smaller public key and signature than those in 88-bit security reference implementation of [9]. The only downside is that doubling the security parameter n can lead to bigger algorithms running times. The impact on running times and how to decrease it remains as an open problem for future works.

When realizing this implementation, a bigger security drop was expected. However, as it turns out we are able to achieve an interesting scenario. A possible explanation for this result relies on the fact that using the higher-bit approximate setting allows for some bigger parameters. Namely, our construction allows us to use a bigger l than compared to [9]. In F-trapdoors construction,

such a big parameter l would lead to a very big drop in the security as the approximation error grows really big. However, in our construction, this bigger error in the trapdoor is counterbalanced by having a much smaller approximate preimage size which leads to better security. That is why recommended and optimized parameters sets in the higher-bit approximate setting are different than the ones in the approximate setting [9]. For information, experience results for the same parameters as F-trapdoors construction are given in Figure 4 below. It might helps one to understand better the impact of this work.

Remark. In our construction of the higher-bit approximate setting, the parameter d is different than l which defines the gadget-matrix $\boldsymbol{F}$. It is set in order to allow more parameter choices. Indeed, the induced error grows as d gets bigger. Thus, a bigger d implies a bigger reduction loss and reduced security. However, it appears that as long as it is decided to use the higher-bit approximate setting, one should always set d as big as possible (i.e $d = l$). Indeed, the biggest security loss is already caused just by using the higher-bit algorithms (due to the $\sqrt{n}$ factor). Thus, security drop implied by setting a bigger d is negligible and should be ignored in parameters choices.

	$\boldsymbol{F}$-trapdoor [9]	$\boldsymbol{F}$-trapdoor [9]	This work	This work
n	512	1024	512	1024
$k = \lceil \log_b q \rceil$	8	9	8	9
b	4	4	4	4
l	4	5	4	5
d	-	-	4	5
τ	2.6	2.8	2.6	2.8
s	2505.6	3733.1	2494.5	3741.7
m	3072	6144	3072	6144
$\|\|y\|\|_2$	138326.9	296473.0	8273.1	11534.9
$\|\|e\|\|_2$	19793.8	1502259.7	433381.2	2422789.0
PK (kB)	5.12	11.52	2.56	5.12
Sig (kB)	4.5	9.4	3.09	6.14
LWE	104.7	192.7	104.7	192.7
AISIS	87.8	183.7	75.0	140.5

Figure 4: Some concrete parameters. The size of PK is measured in kB. LWE and AISIS refers to the security levels of breaking the associated problems. $||y||$ and $||e||$ are the norms of the preimage and error term. τ is the Gaussian width of R.

Acknowledgments. We would like to thank Yilei Chen, Nicholas Genise and Pratyay Mukherjee for kindly sharing with us their implementation of Hash-and-Sign signature based on F-trapdoors. We are especially grateful to Yilei Chen for his invaluable advice to our work.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (h)ibe in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. vol. LNCS 6110, pp. 553–572. Springer (2010). https://doi.org/10.1007/978-3-642-13190-5_28

2. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). STOC '96, ACM, New York, NY, USA (1996). https://doi.org/10.1145/237814.237838

3. Ajtai, M.: Generating hard instances of the short basis problem. In: ICALP. vol. LNCS 1644. Springer (1999). https://doi.org/10.1007/3-540-48523-6_1

4. Alkim, E., et al.: The lattice-based digital signature scheme qtesla. In: ACNS 2020. vol. LNCS 12146, pp. 441–460. Springer (2020). https://doi.org/10.1007/978-3-030-57808-4_22

5. Alwen, J., Peikert, C.: Generating shorter bases for hard random lattices. Theor. Comp. Sys. **48**(3), 535–553 (2010). https://doi.org/10.1007/s00224-010-9278-3

6. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. STOC '13, ACM, New York, NY, USA (2013). https://doi.org/10.1145/2488608.2488680

7. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. J. Cryptol. **25**(4), 601–639 (2011). https://doi.org/10.1007/s00145-011-9105-2

8. Chen, Y.: Private communication

9. Chen, Y., Genise, N., Mukherjee, P.: Approximate trapdoors for lattices and smaller hash-and-sign signatures. In: ASIACRYPT 2019. vol. LNCS 11923, pp. 3–32. Springer (2019). https://doi.org/10.1007/978-3-030-34618-8_1

10. Ducas, L., et al.: Crystals-dilithium: A lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems (1), 238–268 (2018). https://doi.org/10.13154/tches.v2018.i1.238-268

11. Fouque, P.-A., e.a.: Falcon: fast-fourier lattice-based compact signatures over ntru (2018), `https://falcon-sign.info/`

12. Gentry, C.: Fully homomorphic encryption using ideal lattices. STOC '09, ACM (2009). https://doi.org/10.1145/1536414.1536440

13. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. STOC '08, ACM (2008). https://doi.org/10.1145/1374376.1374407

14. Hoffstein, J., Pipher, J., Silverman, J.H.: Ntru: A ring-based public key cryptosystem. In: Algorithmic Number Theory. vol. LNCS 1423, pp. 267–288. Springer (1998). https://doi.org/10.1007/BFb0054868

15. Lindner, R., Peikert, C.: Better key sizes (and attacks) for lwe-based encryption. In: CT-RSA 2011. vol. LNCS 6558, pp. 319–339. Springer (2011). https://doi.org/10.1007/978-3-642-19074-2_21

16. Lyubashevsky, V.: Lattice signatures without trapdoors. In: EUROCRYPT 2012. vol. LNCS 7237, pp. 738–755. Springer (2012). https://doi.org/10.1007/978-3-642-29011-4_43

17. Lyubashevsky, V., Micciancio, D.: Asymptotically efficient lattice-based digital signatures. In: Theory of Cryptography. vol. LNCS 4948, pp. 37–54. Springer (2008). https://doi.org/10.1007/978-3-540-78524-8_3

18. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: EUROCRYPT 2010. vol. LNCS 6110, pp. 1–23. Springer (2010). https://doi.org/10.1007/978-3-642-13190-5_1

19. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. In: 45th IEEE Symposium on FOCS. pp. 372–381 (2004). https://doi.org/10.1109/FOCS.2004.72

20. Micciancio, D.: Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. FOCS '02, IEEE (2002). https://doi.org/10.1109/SFCS.2002.1181960

21. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: EUROCRYPT 2012. vol. LNCS 7237, pp. 700–718. Springer (2012). https://doi.org/10.1007/978-3-642-29011-4_41

22. Moody, D., et al.: Status report on the second round of the nist post-quantum cryptography standardization process (2020). https://doi.org/10.6028/NIST.IR.8309

23. Peikert, C.: An efficient and parallel gaussian sampler for lattices. In: CRYPTO 2010. vol. LNCS 6223, pp. 80–97. Springer (2010). https://doi.org/10.1007/978-3-642-14623-7_5

24. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. ACM **56**(6) (2009). https://doi.org/10.1145/1568318.1568324

25. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. SFCS '94, IEEE Computer Society (1994). https://doi.org/10.1109/SFCS.1994.365700