Learning a Parallelepiped: Cryptanalysis of GGH and NTRU Signatures

Phong Q. Nguyen*[1] and Oded Regev**[2]

[1] CNRS & École normale supérieure, DI, 45 rue d'Ulm, 75005 Paris, France.
http://www.di.ens.fr/~pnguyen/
[2] Department of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel.
http://www.cs.tau.ac.il/~odedr/

Abstract. Lattice-based signature schemes following the Goldreich-Goldwasser-Halevi (GGH) design have the unusual property that each signature leaks information on the signer's secret key, but this does not necessarily imply that such schemes are insecure. At Eurocrypt '03, Szydlo proposed a potential attack by showing that the leakage reduces the key-recovery problem to that of distinguishing integral quadratic forms. He proposed a heuristic method to solve the latter problem, but it was unclear whether his method could attack real-life parameters of GGH and NTRUSign. Here, we propose an alternative method to attack signature schemes à la GGH, by studying the following learning problem: given many random points uniformly distributed over an unknown n-dimensional parallelepiped, recover the parallelepiped or an approximation thereof. We transform this problem into a multivariate optimization problem that can be solved by a gradient descent. Our approach is very effective in practice: we present the first succesful key-recovery experiments on NTRUSign-251 without perturbation, as proposed in half of the parameter choices in NTRU standards under consideration by IEEE P1363.1. Experimentally, 90,000 signatures are sufficient to recover the NTRUSign-251 secret key. We are also able to recover the secret key in the signature analogue of all the GGH encryption challenges, using a number of signatures which is roughly quadratic in the lattice dimension.

1 Introduction

Inspired by the seminal work of Ajtai [1], Goldreich, Goldwasser and Halevi (GGH) proposed at Crypto '97 [9] a lattice analogue of the coding-theory-based public-key cryptosystem of McEliece [19]. The security of GGH is related to the hardness of approximating the closest vector problem (CVP) in a lattice. The GGH article [9] focused on encryption, and five encryption challenges were

* Part of this work is supported by the Commission of the European Communities through the IST program under contract IST-2002-507932 ECRYPT, and by the French government through the X-Crypt RNRT project.
** Supported by an Alon Fellowship, by the Binational Science Foundation, by the Israel Science Foundation, and by the EU Integrated Project QAP.

issued on the Internet [8]. Two years later, Nguyen [22] found a flaw in the original GGH encryption scheme, which allowed to solve four out of the five GGH challenges, and obtain partial information on the last one. Although GGH might still be secure with an appropriate choice of the parameters, its efficiency compared to traditional public-key cryptosystems is perhaps debatable: it seems that a very high lattice dimension is required, while the keysize grows roughly quadratically in the dimension (even when using the improvement suggested by Micciancio [20]). The only lattice-based scheme known that can cope with very high dimension is NTRU [15] (see the survey [23]), which can be viewed as a very special instantiation of GGH with a "compact" lattice and different encryption/decryption procedures (see [20, 21]).

In [9], Goldreich *et al.* described how the underlying principle of their encryption scheme could also provide a signature scheme. The resulting GGH signature scheme did not attract much interest in the research literature until the company NTRU Cryptosystems proposed a relatively efficient signature scheme called NTRUSign [11], based exactly on the GGH design but using the compact NTRU lattices. NTRUSign had a predecessor NSS [14] less connected to the GGH design, and which was broken in [6, 7]. Gentry and Szydlo [7] observed that the GGH signature scheme has an unusual property (compared to traditional signature schemes): each signature released leaks information on the secret key, and once sufficiently many signatures have been obtained, a certain Gram matrix related to the secret key can be approximated. The fact that GGH signatures are not zero-knowledge can be explained intuitively as follows: for a given message, many valid signatures are possible, and the one selected by the secret key says something about the secret key itself.

This information leakage does not necessarily prove that such schemes are insecure. Szydlo [25] proposed a potential attack on GGH based on this leakage (provided that the exact Gram matrix could be obtained), by reducing the key-recovery problem to that of distinguishing integral quadratic forms. It is however unknown if the latter problem is easy or not, although Szydlo proposed a heuristic method based on existing lattice reduction algorithms applied to quadratic forms. As a result, it was unclear if Szydlo's approach could actually work on real-life instantiations of GGH and NTRUSign. The paper [12] claims that, for NTRUSign without perturbation, significant information about the secret key is leaked after 10,000 signatures. However, it does not identify any attack that would require less than 100 million signatures (see [11, Sect. 4.5] and [12, Sect. 7.2 and App. C]).

Our Results. In this article, we present a new key-recovery attack on lattice-based signature schemes following the GGH design, including NTRUSign. The basic observation is that a list of known pairs $(message, signature)$ gives rise to the following learning problem, which we call the hidden parallelepiped problem (HPP): given many random points uniformly distributed over an unknown n-dimensional parallelepiped, recover the parallelepiped or an approximation thereof. We transform the HPP into a multivariate optimization problem based on the fourth moment (also known as *kurtosis*) of one-dimensional projections.

This problem can be solved by a gradient descent. Our approach is very effective in practice: we present the first succesful key-recovery experiments on NTRUSign-251 without perturbation, as proposed in half of the parameter choices in the NTRU standards [4] being considered by IEEE P1363.1 [18]; the number of required signatures can be as low as 90,000, but the true figure might even be lower. We have also been able to recover the secret key in the signature analogue of all five GGH encryption challenges, using a number of signatures which is roughly quadratic in the lattice dimension. When the number of signatures is sufficiently high, the running time of the attack is only a fraction of the time required to generate all the signatures.

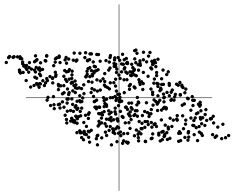

Fig. 1. The Hidden Parallelepiped Problem in dimension two.

Related Work. Interestingly, it turns out that the HPP (as well as related problems) have already been looked at by people dealing with what is known as *Independent Component Analysis* (ICA) (see, e.g., the book by Hyvärinen *et al.* [16]). ICA is a statistical method whose goal is to find directions of independent components, which in our case translates to the n vectors that define the parallelepiped. It has many applications in statistics, signal processing, and neural network research. To the best of our knowledge, this is the first time ICA is used in cryptanalysis.

There are several known algorithms for ICA, and most are based on a gradient method such as the one we use in our algorithm. Our algorithm is closest in nature to the FastICA algorithm proposed in [17], who also considered the fourth moment as a goal function. We are not aware of any rigorous analysis of these algorithms; the proofs we have seen often ignore the effect of errors in approximations. Finally, we remark that the ICA literature offers other, more general, goal functions that are supposed to offer better robustness against noise etc. We have not tried to experiment with these other functions, since the fourth moment seems sufficient for our purposes.

Another closely related result is that by Frieze *et al.* [5], who proposed a polynomial-time algorithm to solve the HPP (and generalizations thereof). Technically, their algorithm is slightly different from those present in the ICA literature as it involves the Hessian, in addition to the usual gradient method. They also claim to have a fully rigorous analysis of their algorithm, taking into account the effect of errors in approximations. Unfortunately, most of the analysis is missing from the preliminary version, and to the best of our knowledge, a full version of the paper has never appeared.

Open Problems. It would be interesting to study natural countermeasures against our attack, such as:

- Perturbation techniques (as suggested by [12, 4, 13]), where the hidden parallelepiped is replaced by a more complicated set. For instance, the second half of parameter choices in NTRU standards [4] involves exactly a single perturbation. In this case, the attacker now has to solve a hidden parallelepiped problem for which the parallelepiped is replaced by the Minkowski sum of two hidden parallelepipeds: the lattice spanned by one of the parallepipeds is public, but not the other one.
- Using secret bases with much larger entries.

However, such countermeasures have an impact on the effiency of the signature scheme.

Road map. The paper is organized as follows. In Section 2, we provide notation and necessary background on lattices, GGH and NTRUSign. In Section 3, we introduce the hidden parallelepiped problem, and explain its relationship to GGH-type signature schemes. In Section 4, we present a method to solve the hidden parallelepiped problem. In Section 5, we present experimental results obtained with the attack on real-life instantiations of GGH and NTRUSign. In Section 6, we provide a theoretical analysis of the main parts of our attack.

2 Background and Notation

Vectors of $\mathbb{R}^n$ will be row vectors denoted by bold lowercase letters such as $\mathbf{b}$, and we will use row representation for matrices. For any ring $\mathcal{R}$, $\mathcal{M}_n(\mathcal{R})$ will denote the ring of $n \times n$ matrices with entries in $\mathcal{R}$. The group of $n \times n$ invertible matrices with real coefficients will be denoted by $GL_n(\mathbb{R})$ and $O_n(\mathbb{R})$ will denote the subgroup of orthogonal matrices. The transpose of a matrix M will be denoted by M^t, so M^{-t} will mean the inverse of the transpose. The notation $\lceil x \rfloor$ denotes a closest integer to x. Naturally, $\lceil \mathbf{b} \rfloor$ will denote the operation applied to all the coordinates of $\mathbf{b}$. If X is a random variable, we will denote by $\mathrm{Exp}[X]$ its expectation. The gradient of a function f from $\mathbb{R}^n$ to $\mathbb{R}$ will be denoted by $\nabla f = (\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n})$.

2.1 Lattices

Let $\| \cdot \|$ and $\langle \cdot, \cdot \rangle$ be the Euclidean norm and inner product of $\mathbb{R}^n$. We refer to the survey [23] for a bibliography on lattices. In this paper, by the term lattice, we mean a full-rank discrete subgroup of $\mathbb{R}^n$. The simplest lattice is $\mathbb{Z}^n$. It turns out that in any lattice L, not just $\mathbb{Z}^n$, there must exist linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in L$ such that:

$$L = \left\{ \sum_{i=1}^{n} n_i \mathbf{b}_i \mid n_i \in \mathbb{Z} \right\}.$$

Any such n-tuple of vectors $[\mathbf{b}_1, \ldots, \mathbf{b}_n]$ is called a basis of L: an n-dimensional lattice can be represented by a basis, that is, a matrix of $GL_n(\mathbb{R})$. Reciprocally, any matrix $B \in GL_n(\mathbb{R})$ spans a lattice: the set of all integer linear combinations of its rows, that is, $\mathbf{m}B$ where $\mathbf{m} \in \mathbb{Z}^n$. The *closest vector problem* (CVP) is the following: given a basis of $L \subseteq \mathbb{Z}^n$ and a target $\mathbf{t} \in \mathbb{Q}^n$, find a lattice vector $\mathbf{v} \in L$ minimizing the distance $\|\mathbf{v} - \mathbf{t}\|$. If we denote by d that minimal distance, then approximating CVP to a factor k means finding $\mathbf{v} \in L$ such that $\|\mathbf{v} - \mathbf{t}\| \leq kd$. A measurable part D of $\mathbb{R}^n$ is said to be a *fundamental domain* of a lattice $L \subseteq \mathbb{R}^n$ if the sets $\mathbf{b}+D$, where $\mathbf{b}$ runs over L, cover $\mathbb{R}^n$ and have pairwise disjoint interiors. If B is a basis of L, then the parallelepiped $\mathcal{P}_{1/2}(B) = \{\mathbf{x}B, \mathbf{x} \in [-1/2, 1/2]^n\}$ is a fundamental domain of L. All fundamental domains of L have the same Lebesgue measure: the volume $\mathrm{vol}(L)$ of the lattice L.

2.2 The GGH Signature Scheme

The GGH scheme [9] works with a lattice L in $\mathbb{Z}^n$. The secret key is a non-singular matrix $R \in \mathcal{M}_n(\mathbb{Z})$, with very short row vectors (their entries are polynomial in n). Two distributions for the generation of R were suggested in [9]:

- The square distribution (called "random lattice" in [9]), where R is uniformly distributed over $\{-\ell, \ldots, \ell\}^{n \times n}$ for some integer bound ℓ. Goldreich *et al.* [9] suggested $\ell = 4$ because the value of ℓ had almost no effect on the quality of the bases in their experiments.
- The hypercubic distribution (called "almost rectangular lattice" in [9]), where R is the sum of kI_n and a noise with square distribution for some ℓ. The GGH challenges [8] used such a distribution, with parameters $k = 4\lceil\sqrt{n} + 1\rfloor + 1$ and a noise with square distribution $\{-4, \ldots, +3\}^{n \times n}$. Micciancio [20] noticed that this distribution has the weakness that it discloses the rough directions of the secret vectors.

The lattice L is the lattice in $\mathbb{Z}^n$ spanned by the rows of R: the knowledge of R enables the signer to approximate CVP rather well in L. The basis R is then transformed to a non-reduced basis B, which will be public. In the original scheme [9], B is the multiplication of R by sufficiently many small unimodular matrices. Micciancio [20] suggested to use the Hermite normal form (HNF) of L instead. As shown in [20], the HNF gives an attacker the least advantage (in a certain precise sense) and it is therefore a good choice for the public basis. The messages are hashed onto a "large enough" subset of $\mathbb{Z}^n$, for instance a large hypercube. Let $\mathbf{m} \in \mathbb{Z}^n$ be the hash of the message to be signed. The signer applies Babai's round-off CVP approximation algorithm [3] to get a lattice vector close to $\mathbf{m}$:

$$\mathbf{s} = \lfloor \mathbf{m}R^{-1} \rceil R,$$

so that $\mathbf{s} - \mathbf{m} \in \mathcal{P}_{1/2}(R) = \{\mathbf{x}R, \mathbf{x} \in [-1/2, 1/2]^n\}$. Of course, any other CVP approximation algorithm could alternatively be applied, for instance Babai's nearest plane CVP approximation algorithm [3]. To verify the signature $\mathbf{s}$ of $\mathbf{m}$, one would first check that $\mathbf{s} \in L$ using the public basis B, and compute the distance $\|\mathbf{s} - \mathbf{m}\|$ to check that it is sufficiently small.

2.3 NTRUSign

NTRUSign [11] is a special instantiation of GGH with the compact lattices from the NTRU encryption scheme [15], which we briefly recall: we refer to [11, 4] for more details. In the NTRU standards [4] being considered by IEEE P1363.1 [18], one selects $n = 251$ and $q = 128$. Let $\mathcal{R}$ be the ring $\mathbb{Z}[X]/(X^n - 1)$ whose multiplication is denoted by $*$. Using resultants, one computes a quadruplet $(f, g, F, G) \in \mathcal{R}^4$ such that $f * G - g * F = q$ in $\mathcal{R}$ and f is invertible mod q, where f and g have 0–1 coefficients (with a prescribed number of 1), while F and G have slightly larger coefficients, yet much smaller than q. This quadruplet is the NTRU secret key. Then the secret basis is the following $(2n) \times (2n)$ matrix:

$$R = \begin{bmatrix} f_0 & f_1 & \cdots & f_{n-1} & g_0 & g_1 & \cdots & g_{n-1} \\ f_{n-1} & f_0 & \cdots & f_{n-2} & g_{n-1} & g_0 & \cdots & g_{n-2} \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ f_1 & \cdots & f_{n-1} & f_0 & g_1 & \cdots & g_{n-1} & g_0 \\ F_0 & F_1 & \cdots & F_{n-1} & G_0 & G_1 & \cdots & G_{n-1} \\ F_{n-1} & F_0 & \cdots & F_{n-2} & G_{n-1} & G_0 & \cdots & G_{n-2} \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ F_1 & \cdots & F_{n-1} & F_0 & G_1 & \cdots & G_{n-1} & G_0 \end{bmatrix},$$

where f_i denotes the coefficient of X^i of the polynomial f. Due to the special structure of R, it turns out that a single row of R is sufficient to recover the whole secret key. Because f is chosen invertible mod q, the polynomial $h = g/f \pmod q$ is well-defined in $\mathcal{R}$: this is the NTRU public key. Its fundamental property is that $f * h \equiv g \pmod q$ in $\mathcal{R}$. The polynomial h defines a natural public basis of L, which we omit (see [11]).

The messages are assumed to be hashed in $\{0, \ldots, q-1\}^{2n}$. Let $\mathbf{m} \in \{0, \ldots, q-1\}^{2n}$ be such a hash. We write $\mathbf{m} = (\mathbf{m}_1, \mathbf{m}_2)$ with $\mathbf{m}_i \in \{0, \ldots, q-1\}^n$. It is shown in [11] that the vector $(\mathbf{s}, \mathbf{t}) \in \mathbb{Z}^{2n}$ which we would obtain by applying Babai's round-off CVP approximation algorithm to $\mathbf{m}$ using the secret basis R can be alternatively computed using convolution products involving $\mathbf{m}_1$, $\mathbf{m}_2$ and the NTRU secret key (f, g, F, G). In practice, the signature is simply $\mathbf{s}$ and not $(\mathbf{s}, \mathbf{t})$, as $\mathbf{t}$ can be recovered from $\mathbf{s}$ thanks to $\mathbf{h}$. Besides, $\mathbf{s}$ might be further reduced mod q, but its initial value can still be recovered because it is such that $\mathbf{s} - \mathbf{m}_1$ ranges over a small interval (this is the same trick used in NTRU decryption). This gives rise for standard parameter choices to a signature length of $251 \times 7 = 1757$ bits. While this signature length is much smaller than other lattice-based signature schemes such as GGH, it is still significantly larger than more traditional signature schemes such as DSA.

This is the basic NTRUSign scheme [11]. In order to strengthen the security of NTRUSign, perturbation techniques have been proposed in [12, 4, 13]. Roughly speaking, such techniques perturb the hashed message $\mathbf{m}$ before signing with the NTRU secret basis. However, it is worth noting that there is no perturbation in half of the parameter choices recommended in NTRU standards [4] under consideration by IEEE P1363.1. Namely, this is the case for the parameter choices `ees251sp2`, `ees251sp3`, `ees251sp4` and `ees251sp5` in [4]. For the other

half, only a single perturbation is recommended. But NTRU has stated that the parameter sets presented in [13] are intended to supersede these parameter sets.

3 The Hidden Parallelepiped Problem

Consider the signature generation in the GGH scheme described in Section 2. Let $R \in \mathcal{M}_n(\mathbb{Z})$ be the secret basis used to approximate CVP in the lattice L. Let $\mathbf{m} \in \mathbb{Z}^n$ be the message digest. Babai's round-off CVP approximation algorithm [3] computes the signature $\mathbf{s} = \lfloor \mathbf{m}R^{-1} \rceil R$, so that $\mathbf{s} - \mathbf{m}$ belongs to the parallelepiped $\mathcal{P}_{1/2}(R) = \{\mathbf{x}R, \mathbf{x} \in [-1/2, 1/2]^n\}$, which is a fundamental domain of L. In other words, the signature generation is simply a reduction of the message $\mathbf{m}$ modulo the parallelepiped spanned by the secret basis R. If we were using Babai's nearest plane CVP approximation algorithm [3], we would have another fundamental parallelepiped (spanned by the Gram-Schmidt vectors of the secret basis) instead: we will not further discuss this case in this paper, since it does not create any significant difference and since this is not the procedure chosen in NTRUSign.

GGH [9] suggested to hash messages into a set much bigger than the fundamental domain of L. This is for instance the case in NTRUSign where the cardinality of $\{0, \ldots, q-1\}^{2n}$ is much bigger than the lattice volume q^n. Whatever the distribution of the message digest $\mathbf{m}$ might be, it would be reasonable to assume that the distribution $\mathbf{s} - \mathbf{m}$ is uniform (or very close to uniform) in the secret parallelepiped $\mathcal{P}_{1/2}(R)$. This is because the parallelepiped is a fundamental domain, and we would expect the output distribution of the hash function to be random in a natural sense. In other words, it seems reasonable to make the following assumption:

Assumption 1 (The Uniformity Assumption) *Let R be the secret basis of the lattice $L \subseteq \mathbb{Z}^n$. When the GGH scheme signs polynomially many "randomly chosen" message digests $\mathbf{m}_1, \ldots, \mathbf{m}_k \in \mathbb{Z}^n$ using Babai's round-off algorithm, the signatures $\mathbf{s}_1, \ldots, \mathbf{s}_k$ are such that the vectors $\mathbf{s}_i - \mathbf{m}_i$ are independent and uniformly distributed over $\mathcal{P}_{1/2}(R) = \{\mathbf{x}R, \mathbf{x} \in [-1/2, 1/2]^n\}$.*

Note that this is only an idealized assumption: in practice, the signatures and the message digests are integer vectors, so the distribution of $\mathbf{s}_i - \mathbf{m}_i$ is discrete rather than continuous, but this should not be a problem if the lattice volume is sufficiently large, as is the case in NTRUSign. Similar assumptions have been used in previous attacks [7, 25] on lattice-based signature schemes. We emphasize that all our experiments on NTRUSign do not use this assumption and work with real-life signatures.

We thus arrive at the following geometric learning problem (see Fig.1):

Problem 2 (The Hidden Parallelepiped Problem or HPP) *Let $V = [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in GL_n(\mathbb{R})$. Define the parallelepiped spanned by V as $\mathcal{P}(V) = \{\sum_{i=1}^n x_i \mathbf{v}_i, x_i \in [-1, 1]\}$. Denote by $U(\mathcal{P})$ the uniform distribution on a parallelepiped $\mathcal{P}$. Given $\mathrm{poly}(n)$ samples from $U(\mathcal{P}(V))$, find a good approximation of the rows of $\pm V$.*

Algorithm 1 Solving the Hidden Parallelepiped Problem

Input: A polynomial number of samples uniformly distributed over a parallelepiped $\mathcal{P}(V)$.

Output: Approximations of rows of $\pm V$.

1: Compute an approximation G of the Gram matrix V^tV of V^t (see Section 4.1).
2: Compute the Cholesky factor L of G^{-1}, so that $G^{-1} = LL^t$.
3: Multiply the samples of $\mathcal{P}(V)$ by L to the right to obtain samples of $\mathcal{P}(C)$ where $C = VL$.
4: Compute approximations of rows of $\pm C$ by Algorithm 2 from Section 4.3.
5: Multiply each approximation by L^{-1} to the right to derive an approximation of a row of $\pm V$.

In the definition of the HPP, we chose $[-1, 1]$ rather than $[-1/2, 1/2]$ like in Assumption 1 to simplify subsequent calculations. Clearly, if one could solve the HPP, then one would be able to approximate the secret basis in GGH by collecting random pairs $(message, signature)$. If the approximation was sufficiently good, one would recover the secret vectors simply by rounding the coordinates to their closest integer. If simple rounding failed, one could apply approximate CVP algorithms to try to recover the secret lattice vectors, as one knows a lattice basis from the GGH public key. The experiments of [22] on the GGH-challenges [8] show that in practice, one does not need to be extremely close to the lattice to recover the closest lattice vector, even in high dimension. But only experiments can tell if the approximation will be sufficiently good for existing lattice reduction algorithms, if simple rounding failed.

4 Learning a Parallelepiped

In this section, we propose a method to solve the Hidden Parallelepiped Problem (HPP), based on the following steps. First, we approximate the covariance matrix of the given distribution. This covariance matrix is essentially V^tV (where V defines the given parallelepiped). We then exploit this approximation in order to transform our hidden parallelepiped $\mathcal{P}(V)$ into a unit hypercube: in other words, we reduce the HPP to the case where the hidden parallelepiped is a hypercube. Finally, we show how hypercubic instances of the HPP are related to a multivariate optimization problem based on the fourth moment, which we solve by a gradient descent.

We remark that the idea of approximating the covariance matrix was already present in the work of Gentry and Szydlo [25, 7]; however, after this basic step, our strategy differs completely from theirs. We now describe our algorithm in more detail.

4.1 The Gram Matrix/Covariance Leakage

It was first observed by Gentry and Szydlo [7, 25] that GGH signatures leak an approximation of the Gram matrix of the transpose of the secret basis. Here, we simply translate this observation to the HPP setting:

Lemma 1 (Gram Leakage). *Let $V \in GL_n(\mathbb{R})$. Let $\mathbf{v}$ be chosen uniformly at random over the parallelepiped $\mathcal{P}(V)$. Then:*

$$\mathrm{Exp}[\mathbf{v}^t\mathbf{v}] = V^tV/3.$$

Proof. We can write $\mathbf{v} = \mathbf{x}V$ where $\mathbf{x}$ has uniform distribution over $[-1, 1]^n$. Hence,

$$\mathbf{v}^t\mathbf{v} = V^t\mathbf{x}^t\mathbf{x}V.$$

An elementary computation shows that $\mathrm{Exp}[\mathbf{x}^t\mathbf{x}] = I_n/3$ where I_n is the $n \times n$ identity matrix, and the lemma follows. □

Hence, by multiplying by 3 the average of $\mathbf{v}^t\mathbf{v}$ over all the samples $\mathbf{v}$ of the hidden parallelepiped $\mathcal{P}(V)$, we obtain an approximation of V^tV, which is the Gram matrix of V^t. Note that $V^tV/3$ is simply the covariance matrix of $U(\mathcal{P}(V))$.

4.2 Morphing a Parallelepiped into a Hypercube

The second stage is explained by the following result:

Lemma 2 (Hypercube Transformation). *Let $V \in GL_n(\mathbb{R})$. Denote by $G \in GL_n(\mathbb{R})$ the symmetric positive definite matrix V^tV. Denote by $L \in GL_n(\mathbb{R})$ the Cholesky factor[3] of G^{-1}, that is, L is the unique lower-triangular matrix such that $G^{-1} = LL^t$. Then the matrix $C = VL \in GL_n(\mathbb{R})$ satisfies the following:*

1. *The rows of C are unit vectors which are pairwise orthogonal. In other words, C is an orthogonal matrix in $O_n(\mathbb{R})$ and $\mathcal{P}(C)$ is a unit hypercube.*
2. *If $\mathbf{v}$ is chosen uniformly at random over the parallelepiped $\mathcal{P}(V)$, then $\mathbf{c} = \mathbf{v}L$ is uniformly distributed over the hypercube $\mathcal{P}(C)$.*

Proof. The Gram matrix $G = V^tV$ is clearly symmetric positive definite. Then $G^{-1} = V^{-1}V^{-t}$ is also symmetric positive definite: it has a Cholesky factorization $G^{-1} = LL^t$ where L is lower-triangular matrix. Hence, $V^{-1}V^{-t} = LL^t$. Let $C = VL \in GL_n(\mathbb{R})$. Then:

$$CC^t = VLL^tV^t = VV^{-1}V^{-t}V^t = I.$$

For the second claim, let $\mathbf{v}$ have the uniform distribution $U(\mathcal{P}(V))$, then $\mathbf{v} = \mathbf{x}V$ where $\mathbf{x}$ has uniform distribution over $[-1, 1]^n$. It follows that $\mathbf{v}L = \mathbf{x}VL = \mathbf{x}C$ has uniform distribution over $\mathcal{P}(C)$. □

[3] Instead of the Cholesky factor, one can take any matrix L such that $G^{-1} = LL^t$. We work with Cholesky factorization as this turns out to be more convenient in our experiments.

Lemma 2 says that by applying the transform L, we can map our parallelepiped samples uniformly distributed over $\mathcal{P}(V)$ into hypercube samples uniformly distributed over $\mathcal{P}(C)$. If we could approximate the rows of $\pm C$, we could also approximate the rows of $\pm V$ thanks to L^{-1}. In other words, we have reduced the Hidden Parallelepiped Problem into what one might call the Hidden Hypercube Problem (see Fig. 2). From an implementation point of view, we

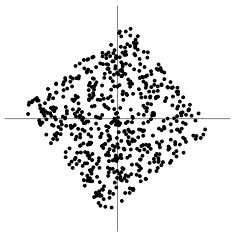

Fig. 2. The Hidden Hypercube Problem in dimension two.

note that the Cholesky factorization (required for obtaining L) can easily be computed by a process close to the Gram-Schmidt orthogonalization process (see [10]). Lemma 2 assumes that we know the Gram matrix $G = V^t V$ exactly. If we only have an approximation of the Gram matrix G, then C will only be close to some orthogonal matrix in $O_n(\mathbb{R})$: the Gram matrix CC^t of C will be close to the identity matrix, and the images of our parallelepiped samples will have a distribution close to the uniform distribution of some unit hypercube.

4.3 Learning a Hypercube

For any $V \in GL_n(\mathbb{R})$ and any integer $k \geq 1$, we define the k-th moment over a vector $\mathbf{w} \in \mathbb{R}^n$ as:

$$\mathrm{mom}_{V,k}(\mathbf{w}) = \mathrm{Exp}[\langle \mathbf{u}, \mathbf{w} \rangle^k],$$

where $\mathbf{u}$ is uniformly distributed over the parallelepiped $\mathcal{P}(V)$. Clearly, $\mathrm{mom}_{V,k}(\mathbf{w})$ can be approximated thanks to the samples of $\mathcal{P}(V)$. We stress that our moments are different from the moments previously considered in [11, 13]: our moments are functions, rather than fixed values. We are interested in the second and fourth moments. A straightforward calculation shows that for any $\mathbf{w} \in \mathbb{R}^n$, they are given by

$$\mathrm{mom}_{V,2}(\mathbf{w}) = \frac{1}{3} \sum_{i=1}^{n} \langle \mathbf{v}_i, \mathbf{w} \rangle^2$$

$$\mathrm{mom}_{V,4}(\mathbf{w}) = \frac{1}{5} \sum_{i=1}^{n} \langle \mathbf{v}_i, \mathbf{w} \rangle^4 + \frac{1}{3} \sum_{i \neq j} \langle \mathbf{v}_i, \mathbf{w} \rangle^2 \langle \mathbf{v}_j, \mathbf{w} \rangle^2$$

Note that the second moment is related to the Gram matrix/covariance mentioned in Section 4.1. When $V \in O_n(\mathbb{R})$, the second moment becomes $\|\mathbf{w}\|^2/3$

while the fourth moment becomes

$$\mathrm{mom}_{V,4}(\mathbf{w}) = \frac{1}{3}\|\mathbf{w}\|^4 - \frac{2}{15}\sum_{i=1}^{n}\langle \mathbf{v}_i, \mathbf{w}\rangle^4.$$

The gradient of the latter is therefore

$$\nabla \mathrm{mom}_{V,4}(\mathbf{w}) = \sum_{i=1}^{n}\left(\frac{4}{3}\left(\sum_{j=1}^{n}\langle \mathbf{v}_j, \mathbf{w}\rangle^2\right)\langle \mathbf{v}_i, \mathbf{w}\rangle - \frac{8}{15}\langle \mathbf{v}_i, \mathbf{w}\rangle^3\right)\mathbf{v}_i.$$

For $\mathbf{w}$ on the unit sphere the second moment is constantly $1/3$, and

$$\mathrm{mom}_{V,4}(\mathbf{w}) = \frac{1}{3} - \frac{2}{15}\sum_{i=1}^{n}\langle \mathbf{v}_i, \mathbf{w}\rangle^4$$

$$\nabla \mathrm{mom}_{V,4}(\mathbf{w}) = \frac{4}{3}\mathbf{w} - \frac{8}{15}\sum_{i=1}^{n}\langle \mathbf{v}_i, \mathbf{w}\rangle^3\mathbf{v}_i. \tag{1}$$

Lemma 3. *Let $V = [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in O_n(\mathbb{R})$. Then the global minimum of $\mathrm{mom}_{V,4}(\mathbf{w})$ over the unit sphere of $\mathbb{R}^n$ is $1/5$ and this minimum is obtained at $\pm\mathbf{v}_1, \ldots, \pm\mathbf{v}_n$. There are no other local minima.*

Proof. The method of Lagrange multipliers shows that for $\mathbf{w}$ to be an extremum point of $\mathrm{mom}_{V,4}$ on the unit sphere, it must be proportional to $\nabla \mathrm{mom}_{V,4}(\mathbf{w})$. By writing $\mathbf{w} = \sum_{i=1}^{n}\langle \mathbf{v}_i, \mathbf{w}\rangle \mathbf{v}_i$ and using Eq. (1), we see that there must exist some α such that $\langle \mathbf{v}_i, \mathbf{w}\rangle^3 = \alpha\langle \mathbf{v}_i, \mathbf{w}\rangle$ for $i = 1, \ldots, n$. In other words, each $\langle \mathbf{v}_i, \mathbf{w}\rangle$ is either zero or $\pm\sqrt{\alpha}$. It is easy to check that among all such points, only $\pm\mathbf{v}_1, \ldots, \pm\mathbf{v}_n$ form local minima. □

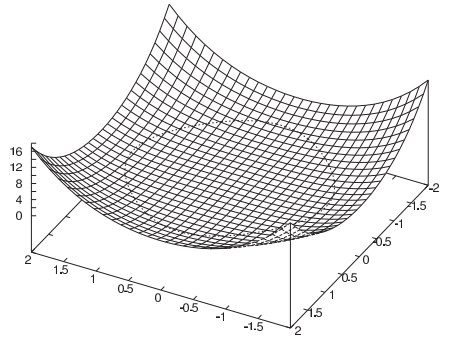

Fig. 3. The fourth moment for $n = 2$. The dotted line shows the restriction to the unit circle.

In other words, the hidden hypercube problem can be reduced to a minimization problem of the fourth moment over the unit sphere. A classical technique to solve such minimization problems is the gradient descent described in Algorithm 2. The gradient descent typically depends on a parameter δ, which has

Algorithm 2 Solving the Hidden Hypercube Problem by Gradient Descent

Parameters: A descent parameter δ.

Input: A polynomial number of samples uniformly distributed over a unit hypercube $\mathcal{P}(V)$.

Output: An approximation of some row of $\pm V$.

Let $\mathbf{w}$ be chosen uniformly at random from the unit sphere of $\mathbb{R}^n$.
Compute an approximation $\mathbf{g}$ of the gradient $\nabla \mathrm{mom}_4(\mathbf{w})$ (see Section 4.3).
Let $\mathbf{w}_{new} = \mathbf{w} - \delta \mathbf{g}$.
Divide $\mathbf{w}_{new}$ by its Euclidean norm $\|\mathbf{w}_{new}\|$.
if $\mathrm{mom}_{V,4}(\mathbf{w}_{new}) \geq \mathrm{mom}_{V,4}(\mathbf{w})$ where the moments are approximated by sampling **then**
  return the vector $\mathbf{w}$.
else
  Replace $\mathbf{w}$ by $\mathbf{w}_{new}$ and go back to Step 2.
end if

to be carefully chosen. Since we want to minimize the function here, we go in the opposite direction of the gradient. To approximate the gradient in Step 2 of Algorithm 2, we notice that

$$\nabla \mathrm{mom}_{V,4}(\mathbf{w}) = \mathrm{Exp}[\nabla(\langle \mathbf{u}, \mathbf{w} \rangle^4)] = 4\mathrm{Exp}[\langle \mathbf{u}, \mathbf{w} \rangle^3 \mathbf{u}].$$

This allows to approximate the gradient $\nabla \mathrm{mom}_{V,4}(\mathbf{w})$ using averages over samples, like for the fourth moment itself.

5 Experimental Results

As usual in cryptanalysis, perhaps the most important question is whether or not the attack works in practice. We therefore implemented the attack in C++ and ran it on a 2GHz PC/Opteron. The critical parts of the code were written in plain C++ using double arithmetic, while the rest used Shoup's NTL library version 5.4 [24]. Based on early experiments, we chose $\delta = 0.7$ in the gradient descent (Algorithm 2), for all the experiments mentioned here. The choice of δ has a big impact on the behaviour of the gradient descent. We stress that our choices of the parameters may not be optimal, so the experimental results should be taken with caution. When doing several descents in a row, it is useful to relax the halting condition 5 in Algorithm 2 to abort descents which seem to make very little progress.

5.1 NTRUSign

We applied Algorithm 1 to real-life parameters of NTRUSign. More precisely, we ran the attack on NTRUSign-251 without perturbation, corresponding to the parameter choices `ees251sp2`, `ees251sp3`, `ees251sp4` and `ees251sp5` in the NTRU standards [4] under consideration by IEEE P1363.1 [18]. This corresponds

to a lattice dimension of 502. We did not rely on the uniformity assumption: we generated genuine NTRUSign signatures of messages generated uniformly at random over $\{0, \ldots, q-1\}^n$. The results of the experiments are summarized in Figure 4. For each given number of signatures, we generated a new set of

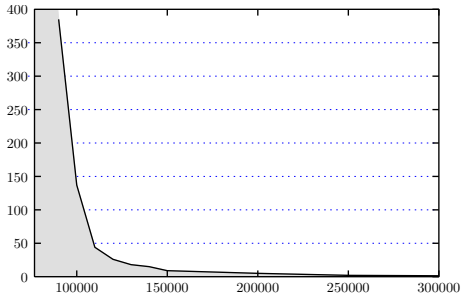

Fig. 4. Experiments on NTRUSign-251 without perturbation. The curve shows the average number of random descents required to recover the secret key, depending on the number of signatures, which is in the range 80,000–300,000.

signatures, and applied Algorithm 1: from the set of samples, we derived an approximation of the Gram matrix, and used it to transform the parallelepiped into a hypercube, and finally, we ran a series of random descents, starting with random points. The curve shows the average number of random descents needed to recover the secret key, based on the number of signatures: the average was computed using roughly a thousand descents. A success is counted when the simple rounding of the approximation obtained discloses *exactly* one of the vectors of the secret basis (which is sufficient to recover the whole secret basis in the case of NTRUSign): no additional approximate CVP stage is required here. Typically, a single random descent does not take much time: for instance, a usual descent for 150,000 signatures takes roughly ten minutes. When successful, a gradient descent may take as little as a few seconds. The minimal number of signatures to make the attack successful in our experiments was 90,000, in which case the required number of random descents to run was about 400. With 80,000 signatures, we tried 5,000 descents without any success, but maybe a substantially larger (yet realistic) number of descents would disclose the secret key: our experiments should not be considered as optimal. The curve given in Fig. 4 may vary a little bit, depending on the secret basis: for instance, for the basis used in the experiments of Fig. 4, the average number of random descents was 15 with 140,000 signatures, but it was 23 for another basis generated with the same NTRU parameters. It seems that the exact geometry of the secret basis has an influence, as will be seen in the analysis of Section 6. It is now clear that perturbation techniques are really mandatory for the security of NTRUSign, though it is currently unknown if such techniques are sufficient to prevent this kind of attacks.

5.2 The GGH Challenges

We also did a few experiments on the GGH-challenges [8], which range from dimension 200 to 400. Because there is actually no GGH signature challenge, we simply generated secret bases like in the GGH encryption challenges. This time, we relied on the uniformity assumption: we created samples uniformly distributed over the secret parallelepiped, and tried to recover the secret basis. Because the secret vectors are significantly longer than in the NTRUSign case, we never recovered directly the secret vector by simple rounding when the descent was successful, but the approximation obtained was sufficiently good to disclose the secret vector after applying Babai's CVP nearest plane algorithm [3], provided that the number of samples was large enough. When starting the descent, rather than starting with a random point on the unit sphere, we took advantage of the fact that we knew the rough directions of the secret vectors, due to the hypercubic distribution. As a result, the gradient descent takes very few iterations compared to the general case. For instance, in dimension 400, with 360,000 signatures, the gradient descent required only 6 iterations. The difference between the rounded vector found and the closest lattice vector was a 400-dimensional integer vector with only four ± 1 coefficients, the rest being zero. By applying Babai's CVP nearest plane algorithm [3] on an LLL-reduced basis (obtained by LLL reduction of the public HNF basis), we obtained immediately an exact vector of the secret basis: no strong reduction algorithm was needed (note that the error vector was much smaller than in the experiments of [22]). From our limited experiments ranging from dimension 200 to 400, it seemed that the number of required signatures was roughly quadratic in the dimension, if one wished for a very good success rate. To thwart such attacks, it seems that the GGH scheme would need to use secret bases with much bigger entries, unlike in the GGH challenges. This would certainly impact the efficiency of the signature scheme, notably the size of the signature, which is already not negligible even for NTRUSign.

6 Theoretical Analysis

Our goal in this section is to give a rigorous theoretical justification to the success of the attack. We will not try to give a tight estimate on the performance of the attack. Instead, we will show that given a large enough polynomial number of samples, Algorithm 1 succeeds in finding a good approximation to a row of V with some constant probability. Let us remark that it is possible that a rigorous analysis already exists in the ICA literature, although we were unable to find any (an analysis under some simplifying assumptions can be found in [17]). Also, Frieze et al. [5] sketch a rigorous analysis of a similar algorithm.

In order to approximate the covariance matrix, the fourth moment, and its gradient, our attack computes averages over samples. Mainly because the samples are independent (and identically distributed), we can use known bounds on large deviations such as the Chernoff bound (see, e.g., [2]) to obtain that with extremely high probability the approximations are very close to the true values.

In our analysis below we omit the explicit calculations, as these are relatively standard.

6.1 Analysis of Algorithm 2

We start by analyzing Algorithm 2. For simplicity, we consider only the case in which the descent parameter δ equals $3/4$. A similar analysis holds for $0 < \delta < 3/4$. Another simplifying assumption we make is that instead of the stopping rule in Step 5 we simply repeat the descent step some small number r of times (which will be specified later).

For now, let us assume that the matrix V is an orthogonal matrix, so our samples are drawn from a unit hypercube $\mathcal{P}(V)$. We will later show that the actual matrix V, as obtained from Algorithm 1, is very close to orthogonal, and that this approximation does not affect the success of Algorithm 2.

Let us first analyze the behavior of Algorithm 2 under the assumption that all gradients are computed exactly without any error. Let $\mathbf{v}_1, \ldots, \mathbf{v}_n$ be n orthonormal vectors that define the parallelepiped $\mathcal{P}(V)$, and write any $\mathbf{w} \in \mathbb{R}^n$ as $\mathbf{w} = \sum_{i=1}^{n} w_i \mathbf{v_i}$. Then, using Eq. (1), we see that for $\mathbf{w}$ on the unit sphere,

$$\nabla \mathrm{mom}_{V,4}(\mathbf{w}) = \frac{4}{3}\mathbf{w} - \frac{8}{15}\sum_{i=1}^{n} w_i^3 \mathbf{v}_i.$$

Since we took $\delta = 3/4$, Step 3 in Algorithm 2 performs

$$\mathbf{w}_{new} = \frac{2}{5}\sum_{i=1}^{n} w_i^3 \mathbf{v}_i.$$

The vector is then normalized in Step 4. So we see that each step in the gradient descent takes a vector $(w_1, \ldots, w_n)$ to the vector $\alpha \cdot (w_1^3, \ldots, w_n^3)$ for some normalization factor α (where both vectors are written in the $\mathbf{v}_i$ basis). Hence, after r iterations, a vector $(w_1, \ldots, w_n)$ is transformed to the vector

$$\alpha \cdot (w_1^{3^r}, \ldots, w_n^{3^r})$$

for some normalization factor α.

Recall now that the original vector $(w_1, \ldots, w_n)$ is chosen uniformly from the unit sphere. It can be shown that with some constant probability, one of its coordinates is greater in absolute value than all other coordinates by a factor of at least $1 + \Omega(1/\log n)$ (first prove this for a vector distributed according to the standard multivariate Gaussian distribution, and then note that by normalizing we obtain a uniform vector from the unit sphere). For such a vector, after only $r = O(\log \log n)$ iterations, this gap is amplified to more than, say, $n^{\log n}$, which means that we have one coordinate very close to ± 1 and all others are at most $n^{-\log n}$ in absolute value. This establishes that if all gradients are known precisely, Algorithm 2 succeeds with some constant probability.

To complete the analysis of Algorithm 2, we now argue that it succeeds with good probability even in the presence of noise in the approximation of

the gradients. First, it can be shown that for any $c > 0$, given a large enough polynomial number of samples, with very high probability all our gradient approximations are accurate to within an additive error of n^{-c} in the ℓ_2 norm (we have r such approximations during the course of the algorithm). This follows by a standard application of the Chernoff bound followed by a union bound. Now let $\mathbf{w} = (w_1, \ldots, w_n)$ be a unit vector in which one coordinate, say the jth, is greater in absolute value than all other coordinates by some factor $\eta \geq 1 + \Omega(1/\log n)$. Since $\mathbf{w}$ is a unit vector, this in particular means that $w_j > 1/\sqrt{n}$. As we saw before, in the vector $\tilde{\mathbf{w}}_{new} := \mathbf{w} - \delta \nabla \mathrm{mom}_4(\mathbf{w})$, this factor increases to η^3. We also have that $\tilde{\mathbf{w}}_{new,j} = \frac{2}{5} w_j^3 > \frac{2}{5} n^{-1.5} > n^{-2}$. By our assumption on the approximation $\mathbf{g}$, we have that for each i, $|\tilde{\mathbf{w}}_{new,i} - \mathbf{w}_{new,i}| \leq n^{-c}$. So for any $k \neq j$,

$$\frac{|\mathbf{w}_{new,k}|}{|\mathbf{w}_{new,j}|} \leq \frac{|\tilde{\mathbf{w}}_{new,k}| + n^{-c}}{|\tilde{\mathbf{w}}_{new,j}| - n^{-c}} \leq \frac{|\tilde{\mathbf{w}}_{new,k}| + n^{-c}}{|\tilde{\mathbf{w}}_{new,j}|(1 - n^{-c+2})} \leq (1 - n^{-c+2})^{-1}\eta^{-3} + 2n^{-c+2}.$$

So we see that a gap of η turns into a gap of $((1 - n^{-c+2})^{-1}\eta^{-3} + 2n^{-c+2})^{-1}$. A straightforward calculation shows that after $O(\log \log n)$ steps, this gap becomes $\Omega(n^{c-2})$. Hence, by choosing a large enough c, we can make the output vector very close to one of the $\pm \mathbf{v}_i$s. This completes our analysis of Algorithm 2.

6.2 Analysis of Algorithm 1

We now complete the analysis of the attack by analyzing Algorithm 1. Recall that a sample $\mathbf{v}$ from $\mathcal{P}(V)$ can be written as $\mathbf{x}V$ where $\mathbf{x}$ is chosen uniformly from $[-1, 1]^n$. So let $\mathbf{v}_i = \mathbf{x}_i V$ for $i = 1, \ldots, N$ be the input samples. Then our approximation G to the Gram matrix $V^t V$ is given by $G = V^t \tilde{I} V$ where $\tilde{I} = \frac{3}{N} \sum \mathbf{x}_i^t \mathbf{x}_i$. We claim that with high probability, $\tilde{I}$ is very close to the identity matrix. Indeed, for $\mathbf{x}$ chosen randomly from $[-1, 1]^n$, each diagonal entry of $\mathbf{x}^t \mathbf{x}$ has expectation $1/3$ and each off-diagonal entry has expectation 0. Moreover, these entries take values in $[-1, 1]$. By the Chernoff bound we obtain that for any approximation parameter $c > 0$, if we choose, say, $N = n^{2c+1}$ then with very high probability each entry in $\tilde{I} - I$ is at most n^{-c} in absolute value. In particular, this implies that all eigenvalues of the symmetric matrix $\tilde{I}$ are in the range $1 \pm n^{-c+1}$.

Recall that we define L to be the Cholesky factor of $G^{-1} = V^{-1}\tilde{I}^{-1}V^{-t}$ and that $C = VL$. Now $CC^t = VLL^tV^t = \tilde{I}^{-1}$, which implies that C is close to an orthogonal matrix. Let us make this precise. Consider the singular value decomposition of C, given by $C = U_1 D U_2$ where U_1, U_2 are orthogonal matrices and D is diagonal. Then $CC^t = U_1 D^2 U_1^t$ and hence $D^2 = U_1^t \tilde{I}^{-1} U_1$. From this it follows that the diagonal of D consists of the square roots of the reciprocals of the eigenvalues of $\tilde{I}$, which in particular means that all values on the diagonal of D are also in the range $1 \pm n^{-c+1}$.

Consider the orthogonal matrix $\tilde{C} = U_1 U_2$. We will show below that with some constant probability, Step 4 yields a good approximation of a row of $\pm\tilde{C}$, call it $\tilde{\mathbf{c}}$. The output of Algorithm 1 is therefore

$$\tilde{\mathbf{c}} L^{-1} = \tilde{\mathbf{c}} C^{-1} V = (\tilde{\mathbf{c}}\tilde{C}^{-1})(\tilde{C}C^{-1})V = (\tilde{\mathbf{c}}\tilde{C}^{-1})(U_1 D^{-1} U_1^t)V.$$

As we have seen before, all eigenvalues of $U_1 D^{-1} U_1^t$ are close to 1, and therefore the above is a good approximation to a row of $\pm V$, given that $\tilde{\mathbf{c}}$ is a good approximation to a row of $\pm\tilde{C}$.

To complete the analysis, we will show that for large enough c, samples from $\mathcal{P}(C)$ 'look like' samples from $\mathcal{P}(\tilde{C})$. More precisely, assume that c is chosen such that the number of samples required by Algorithm 2 is less than, say, n^{c-4}. Then, it follows from Lemma 4 below that the statistical distance between a set of n^{c-4} samples from $\mathcal{P}(C)$ and a set of n^{c-4} samples from $\mathcal{P}(\tilde{C})$ is at most $O(n^{-1})$. By our analysis of Algorithm 2, we know that when given samples from $\mathcal{P}(\tilde{C})$, it outputs an approximation of a row of $\pm\tilde{C}$ with some constant probability. Hence, when given samples from $\mathcal{P}(C)$, it must still output an approximation of a row of $\pm\tilde{C}$ with a probability that is smaller by at most $O(n^{-1})$ and in particular, constant.

Lemma 4. *The statistical distance between the uniform distribution on $\mathcal{P}(C)$ and that on $\mathcal{P}(\tilde{C})$ is at most $O(n^{-c+3})$.*

Proof. We first show that the parallelepiped $\mathcal{P}(C)$ is almost contained and almost contains the cube $\mathcal{P}(\tilde{C})$:

$$(1 - n^{-c+2})\mathcal{P}(\tilde{C}) \subseteq \mathcal{P}(C) \subseteq (1 + n^{-c+2})\mathcal{P}(\tilde{C}).$$

To show this, take any vector $\mathbf{y} \in [-1, 1]^n$. The second containment is equivalent to showing that all the coordinates of $\mathbf{y}U_1 DU_1^t$ are at most $1 + n^{-c+2}$ in absolute value. Indeed, by the triangle inequality,

$$\begin{aligned}\|\mathbf{y}U_1 DU_1^t\|_\infty \leq \|\mathbf{y}\|_\infty + \|\mathbf{y}U_1(D - I)U_1^t\|_\infty &\leq 1 + \|\mathbf{y}U_1(D - I)U_1^t\|_2 \\ &\leq 1 + n^{-c+1}\sqrt{n} < 1 + n^{-c+2}.\end{aligned}$$

The first containment is proved similarly. On the other hand, the ratio of volumes between the two cubes is $((1 + n^{-c+2})/(1 - n^{-c+2}))^n = 1 + O(n^{-c+3})$. From this it follows that the statistical distance between the uniform distribution on $\mathcal{P}(C)$ and that on $\mathcal{P}(\tilde{C})$ is at most $O(n^{-c+3})$. □

References

1. M. Ajtai. Generating hard instances of lattice problems. In *Proc. of 28th STOC*, pages 99–108. ACM, 1996.
2. N. Alon and J. H. Spencer. *The probabilistic method*. Wiley-Interscience [John Wiley & Sons], New York, second edition, 2000.
3. L. Babai. On Lovász lattice reduction and the nearest lattice point problem. *Combinatorica*, 6:1–13, 1986.
4. Consortium for Efficient Embedded Security. Efficient embedded security standards #1: Implementation aspects of NTRUEncrypt and NTRUSign. Version 2.0 available available at [18], June 2003.
5. A. Frieze, M. Jerrum, and R. Kannan. Learning linear transformations. In *37th Annual Symposium on Foundations of Computer Science (Burlington, VT, 1996)*, pages 359–368. IEEE Comput. Soc. Press, Los Alamitos, CA, 1996.

6. C. Gentry, J. Jonsson, J. Stern, and M. Szydlo. Cryptanalysis of the NTRU signature scheme (NSS) from Eurocrypt 2001. In *Proc. of Asiacrypt '01*, volume 2248 of *LNCS*. Springer-Verlag, 2001.
7. C. Gentry and M. Szydlo. Cryptanalysis of the revised NTRU signature scheme. In *Proc. of Eurocrypt '02*, volume 2332 of *LNCS*. Springer-Verlag, 2002.
8. O. Goldreich, S. Goldwasser, and S. Halevi. Challenges for the GGH cryptosystem. Available at `http://theory.lcs.mit.edu/~shaih/challenge.html`.
9. O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *Proc. of Crypto '97*, volume 1294 of *LNCS*, pages 112–131. IACR, Springer-Verlag, 1997. Full version vailable at ECCC as TR96-056.
10. G. Golub and C. van Loan. *Matrix Computations*. Johns Hopkins Univ. Press, 1996.
11. J. Hoffstein, N. A. Howgrave Graham, J. Pipher, J. H. Silverman, and W. Whyte. NTRUSIGN: Digital signatures using the NTRU lattice. Full version of [12]. Draft of April 2, 2002, available on NTRU's website.
12. J. Hoffstein, N. A. Howgrave Graham, J. Pipher, J. H. Silverman, and W. Whyte. NTRUSIGN: Digital signatures using the NTRU lattice. In *Proc. of CT-RSA*, volume 2612 of *LNCS*, pages 122–140. Springer-Verlag, 2003.
13. J. Hoffstein, N. A. Howgrave Graham, J. Pipher, J. H. Silverman, and W. Whyte. Performances improvements and a baseline parameter generation algorithm for NTRUsign. In *Proc. of Workshop on Mathematical Problems and Techniques in Cryptology*, pages 99–126. CRM, 2005.
14. J. Hoffstein, J. Pipher, and J. H. Silverman. NSS: An NTRU lattice-based signature scheme. In *Proc. of Eurocrypt '01*, volume 2045 of *LNCS*. Springer-Verlag, 2001.
15. J. Hoffstein, J. Pipher, and J.H. Silverman. NTRU: a ring based public key cryptosystem. In *Proc. of ANTS III*, volume 1423 of *LNCS*, pages 267–288. Springer-Verlag, 1998. First presented at the rump session of Crypto '96.
16. A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley & Sons, 2001.
17. A. Hyvärinen and E. Oja. A fast fixed-point algorithm for independent component analysis. *Neural Computation*, 9(7):1483–1492, 1997.
18. IEEE P1363.1. Public-key cryptographic techniques based on hard problems over lattices. `http://grouper.ieee.org/groups/1363/lattPK/`, June 2003.
19. R.J. McEliece. A public-key cryptosystem based on algebraic number theory. Technical report, Jet Propulsion Laboratory, 1978. DSN Progress Report 42-44.
20. D. Micciancio. Improving lattice-based cryptosystems using the Hermite normal form. In *Proc. of CALC '01*, volume 2146 of *LNCS*. Springer-Verlag, 2001.
21. D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: A Cryptographic Perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, 2002.
22. P. Q. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97. In *Proc. of Crypto '99*, volume 1666 of *LNCS*, pages 288–304. IACR, Springer-Verlag, 1999.
23. P. Q. Nguyen and J. Stern. The two faces of lattices in cryptology. In *Proc. of CALC '01*, volume 2146 of *LNCS*. Springer-Verlag, 2001.
24. V. Shoup. NTL: A library for doing number theory. Available at `http://www.shoup.net/ntl/`.
25. M. Szydlo. Hypercubic lattice reduction and analysis of GGH and NTRU signatures. In *Proc. of Eurocrypt '03*, volume 2656 of *LNCS*. Springer-Verlag, 2003.