J. Cryptol. (2009) 22: 139–160
DOI: 10.1007/s00145-008-9031-0

Journal of
CRYPTOLOGY

Learning a Parallelepiped: Cryptanalysis of GGH and NTRU Signatures

Phong Q. Nguyen and Oded Regev
INRIA & École Normale Supérieure, DI, 45 rue d'Ulm, 75005, Paris, France
url: http://www.di.ens.fr/~pnguyen/
and
School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel
url: http://www.cs.tau.ac.il/~odedr/

Online publication 11 November 2008

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 provably be solved by a gradient descent. Our approach is very effective in practice: we present the first successful 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, 400 signatures are sufficient to recover the NTRUSIGN-251 secret key, thanks to symmetries in NTRU lattices. We are also able to recover the secret key in the signature analogue of all the GGH encryption challenges.

Key words. GGH, NTRUSIGN, Lattices, Moment, Gradient descent, Public-key cryptanalysis.

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

P.Q. Nguyen 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.

O. Regev supported by the Binational Science Foundation, by the Israel Science Foundation, by the European Commission under the Integrated Project QAP funded by the IST directorate as Contract Number 015848, and by a European Research Council (ERC) Starting Grant.

cryptosystem of McEliece [22]. 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 issued on the Internet [10]. Two years later, Nguyen [29] found a flaw in the original GGH encryption scheme, which allowed one 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 [23]). The only lattice-based scheme known that can cope with very high dimension is NTRU [12] (see the survey [31]), which can be viewed as a very special instantiation of GGH with a "compact" lattice and different encryption/decryption procedures (see [23,25]).

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 [15], based exactly on the GGH design but using the compact NTRU lattices. NTRUSIGN had a predecessor NSS [13] less connected to the GGH design, and which was broken in [6,8]. Gentry and Szydlo [6] 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 [35] 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 [14] 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 [15, Sect. 4.5] and [14, Sect. 7.2 and Appendix C]).

1.1. *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 (see Fig. 1). We transform the HPP into a multivariate

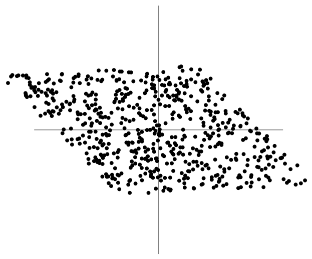

Fig. 1. The Hidden Parallelepiped Problem in dimension two.

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 successful 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 [19]; experimentally, 400 signatures are enough to disclose the NTRUSIGN-251 secret key. We have also been able to recover the secret key in the signature analogue of all five GGH encryption challenges; the GGH case requires significantly more signatures because NTRU lattices have special properties which can be exploited by the attack. 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.

From the theoretical side, we are able to show that under a natural assumption on the distribution of signatures, an attacker can recover a good approximation of the secret key of NTRUSIGN and the GGH challenges in polynomial time, given a polynomial number of signatures of random messages. Since the secret key in both NTRUSIGN and the GGH challenges has very small entries, this approximation leads to the exact secret key by simple rounding.

1.2. *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. [18]). 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.

1.3. *Open Problem*

Our attack does not work against the perturbation techniques proposed in [4,14,16] as efficient countermeasures: these modify the signature generation process in such a way that 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 has to solve an extension of the hidden parallelepiped problem in which the parallelepiped is replaced by the Minkowski sum of two hidden parallelepipeds: the lattice spanned by one of the parallelepipeds is public, but not the other one. The existence of efficient attacks against perturbation techniques is an open problem. The drawbacks of perturbations is that they slow down signature generation, increase both the size of the secret key, and the distance between the signature and the message.

1.4. *Other Schemes*

We now mention some other lattice-based signature schemes, all of which come with an associated security proof, showing that any (asymptotic) attack on the scheme must necessarily lead to an efficient algorithm for a certain lattice problem that is believed to be hard. Moreover, their security is established based on *worst-case hardness*, i.e., any asymptotic attack (even with a small probability of success) implies an efficient solution to *any* instance of the underlying lattice problem. For more details on provably secure lattice-based cryptography and on the signature schemes mentioned below, see, e.g., [24,26,32].

From a theoretical point of view, signature schemes can be constructed from one-way functions in a black-box way without any further assumptions [28]. Therefore, one can obtain signature schemes that are provably secure based on the worst-case hardness of lattice problems by using known constructions of lattice-based one-way functions, such as those in Ajtai's seminal work [1] and followup work. These black-box constructions, however, incur a large overhead and are impractical.

The first construction of *efficient* lattice-based signature schemes with a supporting proof of security (in the random oracle model) was suggested by Micciancio and Vadhan [27]. More efficient schemes were recently proposed by Gentry, Peikert, and Vaikuntanathan [7] and by Lyubashevsky and Micciancio [21].

The former scheme can be seen as a theoretically justified variant of the GGH and NTRUSIGN signature schemes, with worst-case security guarantees based on general lattices in the random oracle model. Compared to the GGH scheme, their construction differs in two main aspects. First, it is based on lattices chosen from a distribution that enjoys a worst-case connection (the lattices in GGH and NTRU are believed to be hard but not known to have a worst-case connection). The second and crucial difference is that their signing algorithm is designed so that it does not reveal any information about the secret basis. This is achieved by replacing Babai's round-off procedure with a "Gaussian sampling procedure," originally due to Klein [20], whose distinctive feature is that its output distribution, for the range of parameters considered in [7], is essentially independent of the secret basis used. The effect of this on our attack is that, instead of observing points chosen uniformly from the parallelepiped generated by the secret basis, the attack observes points chosen from a spherically symmetric Gaussian distribution and therefore learns nothing about the secret basis.

The scheme of Lyubashevsky and Micciancio [21] has worst-case security guarantees based on a type of lattices known as ideal lattices, and it is the most (asymptotically) efficient construction known to date, yielding signature generation and verification algorithms that run in almost linear time. Moreover, the security of [21] does not rely on the random oracle model.

Despite these significant advances, no concrete choice of parameters has been proposed yet, and it is probably fair to say that provably-secure lattice-based signature schemes are not yet at the level of efficiency and maturity that would allow them to be used extensively in real-life applications.

1.5. *Road Map*

The paper is organized as follows. In Sect. 2, we provide notation and necessary background on lattices, GGH and NTRUSign. In Sect. 3, we introduce the hidden parallelepiped problem and explain its relationship to GGH-type signature schemes. In Sect. 4, we present a method to solve the hidden parallelepiped problem. In Sect. 5, we present experimental results obtained with the attack on real-life instantiations of GGH and NTRUSign. In Sect. 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 $\mathbf{b}$. The matrix whose rows are $\mathbf{b}_1, \ldots, \mathbf{b}_n$ is denoted by $[\mathbf{b}_1, \ldots, \mathbf{b}_n]$. We denote the ring of $n \times n$ integer matrices by $\mathcal{M}_n(\mathbb{Z})$. 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, and 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 [31] 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

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

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 measure: the volume $\operatorname{vol}(L)$ 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 nonsingular matrix $R \in \mathcal{M}_n(\mathbb{Z})$, with very short row vectors (their entries are polynomial in n). In the GGH challenges [10], R was chosen as a perturbation of a multiple of the identity matrix, so that its vectors were almost orthogonal: more precisely, $R = kI_n + E$, where $k = 4\lceil \sqrt{n} + 1 \rfloor + 1$, and each entry of the $n \times n$ matrix E is chosen uniformly at random in $\{-4, \ldots, +3\}$. Micciancio [23] 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 nonreduced basis B, which will be public. In the original scheme [9], B is the multiplication of R by sufficiently many small unimodular matrices. Micciancio [23] suggested to use the Hermite normal form (HNF) of L instead. As shown in [23], 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 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 [15] is a special instantiation of GGH with the compact lattices from the NTRU encryption scheme [12], which we briefly recall: we refer to [4,15] for more details. In the NTRU standards [4] being considered by IEEE P1363.1 [19], one selects $N = 251$, $q = 128$. 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. Thus, the lattice dimension is $n = 2N$. 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 the following (natural) public basis of the lattice:

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N-1} \\ 0 & 1 & \cdots & 0 & h_{N-1} & h_0 & \cdots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & \cdots & h_{N-1} & h_0 \\ 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & q \end{bmatrix},$$

which implies that the lattice volume is q^N.

The messages are assumed to be hashed in $\{0, \dots, q-1\}^{2N}$. Let $\mathbf{m} \in \{0, \dots, q-1\}^{2N}$ be such a hash. We write $\mathbf{m} = (\mathbf{m}_1, \mathbf{m}_2)$ with $\mathbf{m}_i \in \{0, \dots, q-1\}^N$. It is shown in [15] 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 [15]. In order to strengthen the security of NTRUSIGN, perturbation techniques have been proposed in [4,14,16]. 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 [16] are intended to supersede these parameter sets.

3. The Hidden Parallelepiped Problem

Consider the signature generation in the GGH scheme described in Sect. 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 $\{0, \dots, q-1\}^{2N}$ is much greater 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 $\mathcal{P}_{1/2}(R)$. More precisely, 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 $\mathbf{m}_1, \dots, \mathbf{m}_k \in \mathbb{Z}^n$ using Babai's round-off algorithm, the signatures $\mathbf{s}_1, \dots, \mathbf{s}_k$ 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 [6,35] 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 $V = [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in GL_n(\mathbb{R})$, and let $\mathcal{P}(V) = \{\sum_{i=1}^{n} x_i \mathbf{v}_i : x_i \in [-1, 1]\}$ be the parallelepiped spanned by V. The input to the HPP is a sequence of poly(n) independent samples from $U(\mathcal{P}(V))$, the uniform distribution over $\mathcal{P}(V)$. The goal is to find a good approximation of the rows 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*). To complete the attack, we need to show how to obtain the actual secret basis given a good enough approximation of it. One simple way to achieve this is by rounding the coordinates of the approximation to the closest integer. This approach will work if and only if the error $\mathbf{e}$ in the approximation has max-norm less than 1/2: we will see that this property can be guaranteed if the entries of the secret basis are small in absolute value, provided that one is given sufficiently many random pairs (*message*, *signature*).

Alternatively, one can use approximate-CVP algorithms to try to recover the secret basis from the approximation, since one knows a lattice basis from the GGH public key. One popular method is to reduce the public basis as much as possible (say using BKZ reduction [33]) and then apply Babai's nearest plane algorithm [3] to the approximation, using the reduced basis. This approach succeeds if and only if the error vector $\mathbf{e}$ lies in the parallelepiped $\mathcal{P}_{1/2}(R^*)$, where R^* is formed by the Gram–Schmidt vectors of the reduced basis. Depending on the geometry of the reduced basis and the error $\mathbf{e}$, this might work for certain cases where the simple rounding approach fails. Previous experiments of [29] on the GGH challenges [10] seem to suggest that in practice, even a moderately good approximation of a lattice vector is sufficient to recover the closest lattice vector, even in high dimension.

The Special Case of NTRUSIGN

Following the announcement of our result in [30], Whyte observed [36] that symmetries in the NTRU lattices might lead to attacks that require far less signatures. Namely, Whyte noticed that in the particular case of NTRUSIGN, the hidden parallelepiped $\mathcal{P}(R)$ has the following property: for each $\mathbf{x} \in \mathcal{P}(R)$, the block-rotation $\sigma(\mathbf{x})$ also belongs to $\mathcal{P}(R)$, where σ is the function that maps any $(x_1, \ldots, x_N, y_1, \ldots, y_N) \in \mathbb{R}^{2N}$ to $(x_N, x_1, \ldots, x_{N-1}, y_N, y_1, \ldots, y_{N-1})$. This is because σ is a linear operation that permutes the rows of R and hence leaves $\mathcal{P}(R)$ invariant. As a result, by using the N possible rotations, each signature actually gives rise to N samples in the parallelepiped $\mathcal{P}(R)$ (as opposed to just one in the general case of GGH). For instance, 400 NTRUSIGN-251 signatures give rise to 100,400 samples in the NTRU parallelepiped. Notice that these samples are no longer independent, and hence Assumption 1 does not hold. Nevertheless, as we will describe later, this technique leads in practice to attacks using a significantly smaller number of signatures.

4. Learning a Parallelepiped

In this section, we describe our solution to 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 $V^t V$ (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. The algorithm is summarized in Algorithms 1 and 2 and is described in more detail in the following.

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 $V^t V$ of $V^t$ (see Sect. 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 Sect. 4.3.
5: Multiply each approximation by $L^{-1}$ to the right to derive an approximation of a row of $\pm V$.
```

4.1. *The Covariance Matrix Leakage*

The first step in our algorithm is based on the idea of approximating the covariance matrix, which was already present in the work of Gentry and Szydlo [6,35] (after this basic step, our strategy differs completely from theirs). Namely, Gentry and Szydlo observed that from GGH signatures one can easily obtain an approximation of $V^t V$, the Gram matrix of the transpose of the secret basis. Here, we simply translate this observation to the HPP setting.

Lemma 1 (Covariance Matrix Leakage). *Let $V \in GL_n(\mathbb{R})$. Let $\mathbf{v}$ be chosen from the uniform distribution over the parallelepiped $\mathcal{P}(V)$. Then*

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

In other words, the covariance matrix of the distribution $U(\mathcal{P}(V))$ is $V^t V/3$.

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 taking the average of $\mathbf{v}^t\mathbf{v}$ over all our samples $\mathbf{v}$ from $U(\mathcal{P}(V))$ and multiplying the result by 3, we can obtain an approximation of V^tV.

4.2. *Morphing a Parallelepiped into a Hypercube*

The second stage is explained by the following lemma.

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*[1] *of G^{-1}, that is, L is the unique lower-triangular matrix with positive diagonal entries 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 pairwise orthogonal unit vectors. 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 uniformly distributed 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. Hence $G^{-1} = V^{-1}V^{-t}$ is also symmetric positive definite and has a Cholesky factorization $G^{-1} = LL^t$, where L is a lower-triangular matrix. Therefore, $V^{-1}V^{-t} = LL^t$. 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 $\mathbf{v}$ be uniformly distributed over $\mathcal{P}(V)$. Then we can write $\mathbf{v} = \mathbf{x}V$, where $\mathbf{x}$ is uniformly distributed over $[-1, 1]^n$. It follows that $\mathbf{v}L = \mathbf{x}VL = \mathbf{x}C$ has the uniform distribution over $\mathcal{P}(C)$. □

Lemma 2 says that by applying the transformation L, we can map our samples from the parallelepiped $\mathcal{P}(V)$ into samples from the hypercube $\mathcal{P}(C)$. Then, if we could approximate the rows of $\pm C$, we would also obtain an approximation of the rows of $\pm V$ by applying 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 note that the Cholesky factorization (required for obtaining L) can be easily computed by a process close to the Gram–Schmidt orthogonalization process (see [11]). Lemma 2 assumes that we know $G = V^tV$ exactly. If we only have an approximation of 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 image under L of our parallelepiped samples will be uniformly distributed over a body that is close to being a unit hypercube.

[1] 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.

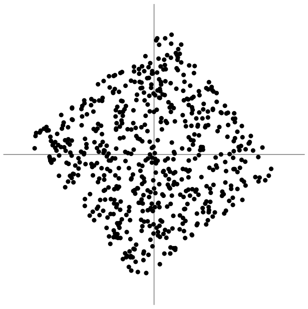

Fig. 2. The Hidden Hypercube Problem in dimension two.

4.3. *Learning a Hypercube*

For any $V = [\mathbf{v}_1, \dots, \mathbf{v}_n] \in GL_n(\mathbb{R})$ and any integer $k \geq 1$, we define the kth moment of $\mathcal{P}(V)$ over a vector $\mathbf{w} \in \mathbb{R}^n$ as

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

where $\mathbf{u}$ is uniformly distributed over the parallelepiped $\mathcal{P}(V)$.[2] Clearly, $\mathrm{mom}_{V,k}(\mathbf{w})$ can be approximated by using the given samples from $U(\mathcal{P}(V))$. Since all the odd moments are zero, we are interested in the first even moments, namely the second and fourth moments. A straightforward calculation shows that for any $\mathbf{w} \in \mathbb{R}^n$, they are given by

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

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

Note that the second moment is given by the covariance matrix mentioned in Sect. 4.1. When $V \in O_n(\mathbb{R})$ (i.e., the vectors $\mathbf{v}_i$ are orthonormal), the second moment becomes $\|\mathbf{w}\|^2/3$, while the fourth moment becomes

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

By expressing $\mathbf{w}$ in the $\mathbf{v}_i$ basis and taking derivatives in each of the directions $\mathbf{v}_i$, it is not hard to verify that the gradient of the latter is

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

[2] This should not be confused with an unrelated notion of moment considered in [6,15,16].

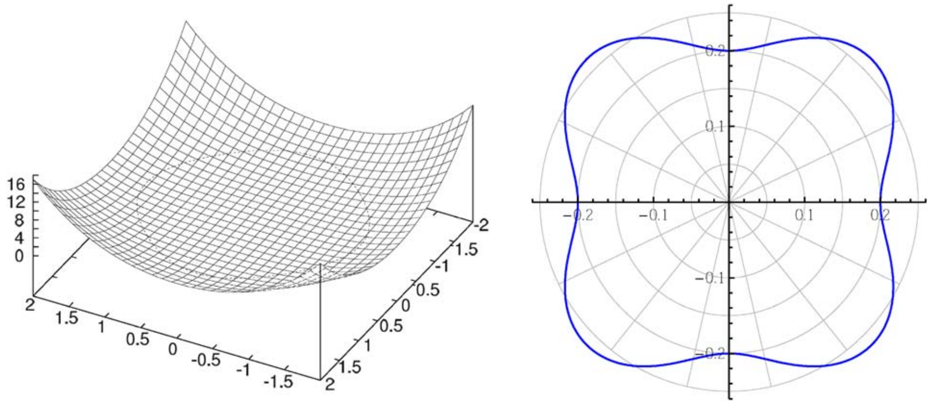

Fig. 3. The fourth moment for $n = 2$. On the *left*: the *dotted line* shows the restriction to the *unit circle*. On the *right*: a *polar plot* restricted to the *unit circle*.

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

$$\begin{aligned} \mathrm{mom}_{V,4}(\mathbf{w}) &= \frac{1}{3} - \frac{2}{15}\sum_{i=1}^{n}\langle \mathbf{v}_i, \mathbf{w}\rangle^4, \\ \nabla\mathrm{mom}_{V,4}(\mathbf{w}) &= \frac{4}{3}\mathbf{w} - \frac{8}{15}\sum_{i=1}^{n}\langle \mathbf{v}_i, \mathbf{w}\rangle^3 \mathbf{v}_i. \end{aligned} \tag{1}$$

See Fig. 3.

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 (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. □

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 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

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

Algorithm 2 Solving the Hidden Hypercube Problem by Gradient Descent

```
Parameters: A descent parameter $\delta$.
Input: A polynomial number of samples uniformly distributed over a unit hypercube
    $\mathcal{P}(V)$.
Output: An approximation of some row of $\pm V$.

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

This allows us 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 2 GHz 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 [34]. Based on trial-and-error, 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 behavior of the gradient descent: the choice $\delta = 0.7$ works well, but we do not claim that it is optimal. When doing several descents in a row, it is useful to relax the halting condition 2 in Algorithm 2 to abort descents which seem to make very little progress.

5.1. NTRUSign

We performed two kinds of experiments against NTRUSign, depending on whether the symmetries of NTRU lattices explained in Sect. 3 were used or not. All the experiments make it 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.

5.1.1. *Without Exploiting the Symmetries of NTRU Lattices*

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

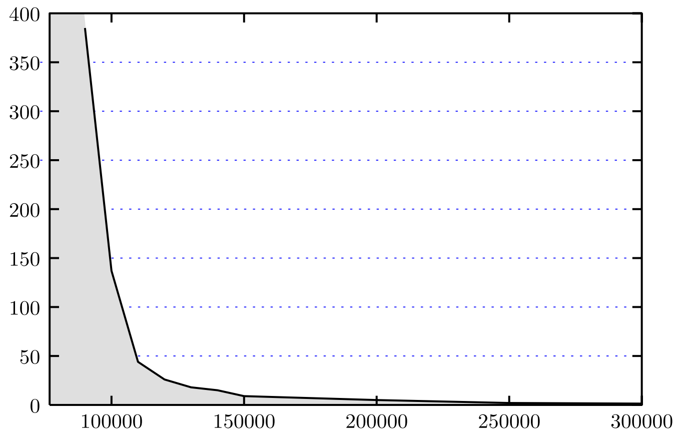

Fig. 4. Experiments on NTRUSIGN-251 without perturbation and without using NTRU symmetries. The curve shows the average number of random descents required to recover the secret key, depending on the number of signatures, which is in the range 80,000–300,000.

NTRU standards [4] under consideration by IEEE P1363.1 [19]. 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 Fig. 4. For each given number of signatures in the range 80,000–300,000, we generated a set of signatures and applied Algorithm 1 to it: from the set of samples we derived an approximation of the Gram matrix, used it to transform the parallelepiped into a hypercube, and finally, we ran a series of about a thousand descents, starting with random points. We regard a descent as successful if, when rounded to the nearest integer vector, the output of the descent gives *exactly* one of the vectors of the secret basis (which is sufficient to recover the whole secret basis in the case of NTRUSIGN). We did not notice any improvement using Babai's nearest plane algorithm [3] (with a BKZ-20 reduced basis [33] computed from the public basis) as a CVP approximation. The curve shows the average number of random descents needed for a successful descent as a function of the number of signatures.

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 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 was about 400. With 80,000 signatures, we tried 5,000 descents without any success. 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 Sect. 6.

Table 1. Experiments on NTRUSIGN-251 without perturbation, using NTRU symmetries.

Number of signatures	Expected number of descents to recover the secret key
1,000	2
500	40
400	100

5.1.2. *Exploiting the Symmetries of NTRU Lattices*

Based on Whyte's observation described in Sect. 3, one might hope that the number of signatures required by the attack can be shrunk by a factor of roughly N. Luckily, this is indeed the case in practice (see Table 1): as few as 400 signatures are enough in practice to recover the secret key, though the corresponding 100,400 parallelepiped samples are not independent. This means that the previous number of 90,000 signatures required by the attack can be roughly divided by $N = 251$. Hence, NTRUSIGN without perturbation should be considered totally insecure.

5.2. *The GGH Challenges*

We also did experiments on the GGH challenges [10], 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. To decrease the cost of sample generation, and because there was no concrete proposal of a GGH signature scheme, we relied on the uniformity assumption: we created samples uniformly distributed over the secret parallelepiped and tried to recover the secret basis.

When the number of samples becomes sufficiently high, the approximation obtained by a random descent is sufficiently good to disclose one of the vectors of the secret basis by simple rounding, just as in the NTRUSIGN case: however, the number of required samples is significantly higher than for NTRUSIGN; for instance, with 200,000 samples in dimension 200, three descents are enough to disclose a secret vector by rounding; whereas three descents are also enough with 250,000 samples for NTRUSIGN, but this corresponds to a dimension of 502, which is much greater than 200. This is perhaps because the secret vectors of the GGH challenges are significantly longer than those of NTRUSIGN.

However, one can significantly improve the result by using a different rounding procedure, as pointed out in Sect. 3. Namely, instead of rounding the approximation to an integer vector, one can apply a CVP approximation algorithm such as Babai's nearest plane algorithm [3]: such algorithms will succeed if the approximation is sufficiently close to the lattice, and one can improve the chances of the algorithm by computing a lattice basis as reduced as possible (using, for instance, BKZ reduction [33]). For instance, with only 20,000 samples in dimension 200, it was impossible to recover a secret vector by rounding, but it became easy with Babai's nearest plane algorithm on a BKZ-20 reduced basis (obtained by BKZ reduction of the public HNF basis): more precisely, three random descents sufficed on the average. More generally, Fig. 5 shows the average number of samples required so that ten random descents disclose a secret

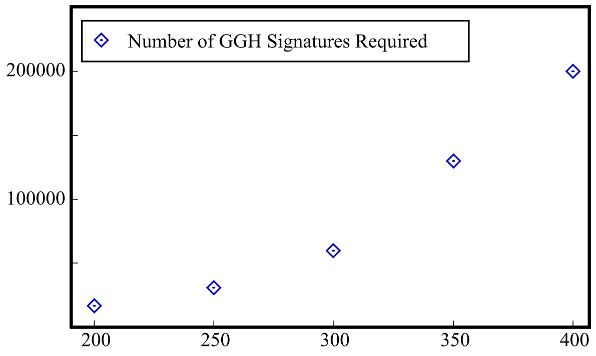

Fig. 5. Average number of GGH signatures required so that ten random descents coupled with Babai's nearest plane algorithm disclose with high probability a secret vector, depending on the dimension of the GGH challenge.

vector with high probability, depending on the dimension of the GGH challenge, using Babai's nearest plane algorithm on a BKZ-20 reduced basis. Figure 5 should not be interpreted as the minimal number of signatures required for the success of the attack: it only gives an upper bound for that number. Indeed, there are several ways to decrease the number of signatures:

- One can run much more than ten random descents.
- One can take advantage of the structure of the GGH challenges. When starting a descent, rather than starting with a random point on the unit sphere, we may exploit the fact that we know the rough directions of the secret vectors.
- One can use better CVP approximation algorithms or use better reduction algorithms in conjunction with Babai's nearest plane algorithm.

6. Theoretical Analysis

Our goal in this section is to give a rigorous theoretical justification to the success of the attack. Namely, 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. For sake of clarity and simplicity, we will not make any attempt to optimize this polynomial bound on the number of samples. We will also assume that we can perform operations on real numbers; modifying the analysis to work with finite precision numbers should be straightforward. 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. 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.

Theorem 3. *For any $c_0 > 0$, there exists $c_1 > 0$ such that given n^{c_1} samples uniformly distributed over some unit hypercube $\mathcal{P}(V)$, $V = [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in O_n(\mathbb{R})$, Algorithm 2 with $\delta = 3/4$ and $r = O(\log\log n)$ descent steps outputs with constant probability a vector that is within ℓ_2 distance n^{-c_0} of $\pm\mathbf{v}_i$ for some i.*

Proof. We first analyze the behavior of Algorithm 2 under the assumption that all gradients are computed exactly without any error. We write any vector $\mathbf{w} \in \mathbb{R}^n$ as $\mathbf{w} = \sum_{i=1}^n w_i \mathbf{v_i}$. Then, using (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

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

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

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

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 exactly, 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 at least a factor of $1 + \Omega(1/\log n)$. Since $\mathbf{w}$ is a unit vector, this in particular means that $w_j > 1/\sqrt{n}$. Let $\tilde{\mathbf{w}}_{\text{new}} = \mathbf{w} - \delta\nabla\text{mom}_4(\mathbf{w})$. Recall that for each i, $\tilde{\mathbf{w}}_{\text{new},i} = \frac{2}{5}w_i^3$, which in particular implies that $\tilde{\mathbf{w}}_{\text{new},j} > \frac{2}{5}n^{-1.5} > n^{-2}$. By our assumption on the approximation $\mathbf{g}$, we have that for each i, $|\tilde{\mathbf{w}}_{\text{new},i} - \mathbf{w}_{\text{new},i}| \leq n^{-c}$. So for any $k \neq j$,

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

If $|\tilde{\mathbf{w}}_{\text{new},k}| > n^{-(c-1)}$, then the above is at least $(1 - O(1/n))(w_j/w_k)^3$. Otherwise, the above is at least $\Omega(n^{c-3})$. Hence, after $O(\log\log n)$ steps, the gap w_j/w_k becomes $\Omega(n^{c-3})$. Therefore, for any $c_0 > 0$, we can make the distance between the output vector and one of the $\pm\mathbf{v}_i$'s less than n^{-c_0} by choosing a large enough c. □

6.2. *Analysis of Algorithm 1*

The following theorem completes the analysis of the attack. In particular, it implies that if V is an integer matrix all of whose entries are bounded in absolute value by some polynomial, then running Algorithm 1 with a large enough polynomial number of samples from the uniform distribution on $\mathcal{P}(V)$ gives (with constant probability) an approximation to a row of $\pm V$ whose error is less than 1/2 in each coordinate and therefore leads to an *exact* row of $\pm V$ simply by rounding each coordinate to the nearest integer. Hence we have a rigorous proof that our attack can efficiently recover the secret key in both NTRUSign and the GGH challenges.

Theorem 4. *For any $c_0 > 0$, there exists $c_1 > 0$ such that given n^{c_1} samples uniformly distributed over some parallelepiped $\mathcal{P}(V)$, $V = [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in GL_n(\mathbb{R})$, Algorithm 1 outputs with constant probability a vector $\tilde{\mathbf{e}}V$, where $\tilde{\mathbf{e}}$ is within ℓ_2 distance n^{-c_0} of some standard basis vector $\mathbf{e}_i$.*

Proof. Recall that a sample $\mathbf{v}$ from $U(\mathcal{P}(V))$ 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^tV 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. This implies that all eigenvalues of the symmetric matrix $\tilde{I}$ are in the range $1 \pm n^{-c+1}$ (and in particular we obtain that $\tilde{I}$ and hence also G are invertible).

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 $C = U_1DU_2$, where U_1, U_2 are orthogonal matrices, and D is diagonal. Then $CC^t = U_1D^2U_1^t$ 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 $\tilde{C} = U_1U_2$. We claim 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 so 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[3] 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 Theorem 3, we know that when given samples from $\mathcal{P}(\tilde{C})$, Algorithm 2 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 equally good approximation of a row of $\pm\tilde{C}$ with a probability that is smaller by at most $O(n^{-1})$ and in particular, constant.

To complete the proof, let $\tilde{\mathbf{c}}$ be the vector obtained in Step 4. The output of Algorithm 1 is then

$$\tilde{\mathbf{c}}L^{-1} = \tilde{\mathbf{c}}C^{-1}V = \left(\tilde{\mathbf{c}}\tilde{C}^{-1}\right)\left(\tilde{C}C^{-1}\right)V = \left(\tilde{\mathbf{c}}\tilde{C}^{-1}\right)\left(U_1D^{-1}U_1^t\right)V.$$

As we have seen before, all eigenvalues of $U_1D^{-1}U_1^t$ are close to 1. It therefore follows that the above is a good approximation to a row of $\pm V$, and it is not hard to verify that the quality of this approximation satisfies the requirements stated in the theorem. □

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 $\mathcal{P}(\tilde{C})$:

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

To show this, take any vector $\mathbf{y} \in [-1, 1]^n$. The second containment is equivalent to showing that all the coordinates of $\mathbf{y}U_1DU_1^t$ are at most $1 + n^{-c+2}$ in absolute value. Indeed, by the triangle inequality,

$$\begin{aligned}\left\|\mathbf{y}U_1DU_1^t\right\|_\infty &\le \|\mathbf{y}\|_\infty + \left\|\mathbf{y}U_1(D-I)U_1^t\right\|_\infty \le 1 + \left\|\mathbf{y}U_1(D-I)U_1^t\right\|_2 \\ &\le 1 + n^{-c+1}\sqrt{n} < 1 + n^{-c+2}.\end{aligned}$$

[3] The *statistical distance* (or *total variation distance*) between two distributions is the maximum probability with which one can distinguish between an input sampled from the first distribution and an input sampled from the second distribution.

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 $O(n^{-c+3})$. □

Acknowledgements

We thank William Whyte for helpful discussions and the anonymous referees for useful comments.

References

[1] M. Ajtai, Generating hard instances of lattice problems, in *Complexity of Computations and Proofs*. Quad. Mat., vol. 13 (Dept. Math., Seconda Univ. Napoli, Caserta, 2004), pp. 1–32

[2] N. Alon, J.H. Spencer, *The Probabilistic Method*. Wiley-Interscience Series in Discrete Mathematics and Optimization, 2nd edn. (Wiley, New York, 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 at http://grouper.ieee.org/groups/1363/lattPK/index.html, June (2003)

[5] A. Frieze, M. Jerrum, R. Kannan, Learning linear transformations, in *37th Annual Symposium on Foundations of Computer Science*, Burlington, VT, 1996 (IEEE Comput. Soc. Press, Los Alamitos, 1996), pp. 359–368

[6] C. Gentry, M. Szydlo, Cryptanalysis of the revised NTRU signature scheme, in *Proc. of Eurocrypt '02*. LNCS, vol. 2332 (Springer, Berlin, 2002)

[7] C. Gentry, C. Peikert, V. Vaikuntanathan, Trapdoors for hard lattices and new cryptographic constructions, in *Proc. 40th ACM Symp. on Theory of Computing (STOC)*, pp. 197–206 (2008)

[8] C. Gentry, J. Jonsson, J. Stern, M. Szydlo, Cryptanalysis of the NTRU signature scheme (NSS) from Eurocrypt 2001, in *Proc. of Asiacrypt '01*. LNCS, vol. 2248 (Springer, Berlin, 2001)

[9] O. Goldreich, S. Goldwasser, S. Halevi, Public-key cryptosystems from lattice reduction problems, in *Proc. of Crypto '97*. LNCS, vol. 1294 (Springer, Berlin, 1997), pp. 112–131. Full version available at ECCC as TR96-056

[10] O. Goldreich, S. Goldwasser, S. Halevi, Challenges for the GGH cryptosystem. Available at http://theory.lcs.mit.edu/~shaih/challenge.html

[11] G. Golub, C. Loan, *Matrix Computations* (Johns Hopkins Univ. Press, Baltimore, 1996)

[12] J. Hoffstein, J. Pipher, J. Silverman, NTRU: a ring based public key cryptosystem, in *Proc. of ANTS III*. LNCS, vol. 1423 (Springer, Berlin, 1998), pp. 267–288. First presented at the rump session of Crypto '96

[13] J. Hoffstein, J. Pipher, J.H. Silverman, NSS: An NTRU lattice-based signature scheme, in *Proc. of Eurocrypt '01*. LNCS, vol. 2045 (Springer, Berlin, 2001)

[14] J. Hoffstein, N.A.H. Graham, J. Pipher, J.H. Silverman, W. Whyte, NTRUsign: Digital signatures using the NTRU lattice. Full version of *Proc. of CT-RSA*. LNCS, vol. 2612. Draft of April 2, 2002, available on NTRU's website

[15] J. Hoffstein, N.A.H. Graham, J. Pipher, J.H. Silverman, W. Whyte, NTRUsign: Digital signatures using the NTRU lattice, in *Proc. of CT-RSA*. LNCS, vol. 2612 (Springer, Berlin, 2003)

[16] J. Hoffstein, N.A.H. Graham, J. Pipher, J.H. Silverman, W. Whyte, Performances improvements and a baseline parameter generation algorithm for NTRUsign, in *Proc. of Workshop on Mathematical Problems and Techniques in Cryptology* (CRM, 2005), pp. 99–126

[17] A. Hyvärinen, E. Oja, A fast fixed-point algorithm for independent component analysis. *Neural Comput.* **9**(7), 1483–1492 (1997)

[18] A. Hyvärinen, J. Karhunen, E. Oja, *Independent Component Analysis* (Wiley, New York, 2001)

[19] IEEE P1363.1. Public-key cryptographic techniques based on hard problems over lattices. See http://grouper.ieee.org/groups/1363/lattPK/index.html, June 2003
[20] P. Klein, Finding the closest lattice vector when it's unusually close, in *Proc. of SODA '00* (ACM–SIAM, 2000)
[21] V. Lyubashevsky, D. Micciancio, Asymptotically efficient lattice-based digital signatures, in *Fifth Theory of Cryptography Conference (TCC)*. Lecture Notes in Computer Science, vol. 4948 (Springer, Berlin, 2008)
[22] R. McEliece, A public-key cryptosystem based on algebraic number theory. Technical report, Jet Propulsion Laboratory, 1978. DSN Progress Report 42-44
[23] D. Micciancio, Improving lattice-based cryptosystems using the Hermite normal form, in *Proc. of CALC '01*. LNCS, vol. 2146 (Springer, Berlin, 2001)
[24] D. Micciancio, Cryptographic functions from worst-case complexity assumptions. Survey paper prepared for the LLL+25 conference. To appear
[25] D. Micciancio, S. Goldwasser, *Complexity of Lattice Problems: A Cryptographic Perspective*. The Kluwer International Series in Engineering and Computer Science, vol. 671 (Kluwer Academic, Boston, 2002)
[26] D. Micciancio, O. Regev, Lattice-based cryptography, in *Post-Quantum Cryprography*, ed. by D.J. Bernstein, J. Buchmann (Springer, Berlin, 2008)
[27] D. Micciancio, S. Vadhan, Statistical zero-knowledge proofs with efficient provers: lattice problems and more, in *Advances in Cryptology—Proc. CRYPTO '03*. Lecture Notes in Computer Science, vol. 2729 (Springer, Berlin, 2003), pp. 282–298
[28] M. Naor, M. Yung, Universal one-way hash functions and their cryptographic applications, in *Proc. 21st ACM Symp. on Theory of Computing (STOC)*, pp. 33–43 (1989)
[29] P.Q. Nguyen, Cryptanalysis of the Goldreich–Goldwasser–Halevi cryptosystem from Crypto '97, in *Proc. of Crypto '99*. LNCS, vol. 1666 (Springer, Berlin, 1999), pp. 288–304
[30] P.Q. Nguyen, O. Regev, Learning a Parallelepiped: Cryptanalysis of GGH and NTRU Signatures, in *Advances in Cryptology—Proceedings of EUROCRYPT '06*. LNCS, vol. 4004 (Springer, Berlin, 2006), pp. 215–233
[31] P.Q. Nguyen, J. Stern, The two faces of lattices in cryptology, in *Proc. of CALC '01*. LNCS, vol. 2146 (Springer, Berlin, 2001)
[32] O. Regev, Lattice-based cryptography, in *Advances in Cryptology—Proc. of CRYPTO '06*. LNCS, vol. 4117 (Springer, Berlin, 2006), pp. 131–141
[33] C.P. Schnorr, M. Euchner, Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Math. Program.* **66**, 181–199 (1994)
[34] V. Shoup, NTL: A library for doing number theory. Available at http://www.shoup.net/ntl/
[35] M. Szydlo, Hypercubic lattice reduction and analysis of GGH and NTRU signatures, in *Proc. of Eurocrypt '03*. LNCS, vol. 2656 (Springer, Berlin, 2003)
[36] W. Whyte, Improved NTRUSign transcript analysis. Presentation at the rump session of Eurocrypt '06, on May 30 (2006)