Generating Falcon Trapdoors via Gibbs Sampler

Chao Sun[1], Thomas Espitau[2], Junjie Song[1], Jinguang Han[1], and Mehdi Tibouchi[3]

[1] Southeast University, China
{sunchaomt,junjiesong,jghan}@seu.edu.cn
[2] PQShield, France
t.espitau@gmail.com
[3] NTT Social Informatics Laboratories, Japan
mehdi.tibouchi@ntt.com

Abstract. Falcon is a lattice-based signature scheme that has been selected as a standard in NIST post-quantum cryptography standardization project. The trapdoor generation process of Falcon amounts to generating two polynomials, f and g, that satisfy certain conditions to achieve a quality parameter α as small as possible, because smaller α usually leads to higher security levels and shorter signatures. The original approach to generate NTRU trapdoors, proposed by Ducas, Lyubashevsky, and Prest (ASIACRYPT 2014), is based on trial-and-repeat, which generates f and g with small Gaussian coefficients and tests whether they satisfy the condition or not. If not, the process is repeated. In practice, α is chosen as 1.17 because it is the smallest value that keeps the number of repetitions relatively small.

A recent work by Espitau et al. (ASIACRYPT 2023) proposed a new approach to generate NTRU trapdoors: instead of using trial-and-repeat, sample f and g in the Fourier domain that satisfies the targeted quality and map them back to ring elements. In principle, the idea of Fourier sampling applies to Falcon itself as well, but the sampling region in the Fourier domain for Falcon has a distinct, less elegant geometric shape, which makes sampling more challenging.

In this paper, we adopt Markov Chain Monte Carlo (MCMC) methods for sampling. The core idea is to start from an arbitrary point within the target region and perform random walks until the point approximates a random sample from the desired distribution. Specifically, we use Gibbs sampler with Fourier sampling to generate Falcon trapdoors.

Our approach allows us to achieve α values arbitrarily close to 1 efficiently, whereas the original trial-and-repeat method would require impractically many repetitions (far exceeding trillions) to reach even $\alpha = 1.04$. In particular, Falcon-512 currently falls short of the NIST level one requirement of 128 bits, but our method effectively mitigates this gap.

Furthermore, our approach eliminates the need for discrete Gaussian sampling, which is challenging to implement and secure. Instead, our method relies solely on uniform sampling over an interval, simplifying the implementation and improving efficiency.

Keywords: Post-quantum cryptography · Hash-and-sign lattice-based signatures · NTRU trapdoors · Gibbs sampling

1 Introduction

1.1 From GGH to Falcon

Falcon [25] is one of the three signature schemes that are selected by NIST for standardization in the post-quantum competition. It presents the state of the art for lattice-based hash-and-sign signatures, whose basic idea dates back to GGH [16] and NTRUSign [19] in 1990s. In these signature schemes, the signing key is a "good" (short and close to orthogonal) basis of the lattice, and the verification key is a "bad" basis of the same lattice. To sign a message , the signing algorithm first hashes the message to a point in the Euclidean space $\mathbb{R}^n$. With the good basis, the algorithm can find a lattice point that is close to the hashed point, and the difference between these two points is the signature. For verification, the algorithm checks that signature is short and the difference between the signature and hashed point lies in the lattice using the bad basis. However, GGH, NTRUSign and several successive variants are finally broken by statistical attacks [14,22,5,31,20]: it turns out that each signature reveals partial information of the signing key and the adversary can recover the signing key progressively after getting enough number of signatures. This problem was finally solved in 2008, when Gentry, Peikert and Vaikuntanathan (GPV framework) [13] showed how to use Gaussian sampling (a randomized variant of Babai's nearest plane algorithm [2]) to guarantee that each signature does not reveal any information of the signing key. However, there are two issues in terms of compactness and efficiency. The original instantiation of GPV signatures had large parameters and was not actually practical (see [3] for a comparison of parameters). Besides, the Gaussian sampler (Klein-GPV) has quadratic time complexity in the lattice dimension, thus being inefficient for typical parameters.

In order to deal with the first issue, in 2014, Ducas, Lyubashevsky and Prest [4] (DLP) instantiated the GPV framework over NTRU lattices. They carefully analyze the Euclidean length of the Gram-Schmidt orthogonalization of NTRU lattices and are able to generate the trapdoor basis from a distribution where the Gram-Schmidt norm satisfies a certain quality condition. As a result, the DLP scheme has rather compact public key size and signature size (less than 1kB) thanks to the algebraic properties of NTRU lattices.

To address the second issue, Ducas and Prest proposed the Fast Fourier Orthogonalization (FFO) sampler [6] that has quasi-linear time complexity. In fact, the Falcon signature scheme is essentially a combination of DLP scheme and FFO sampler.

Besides the original Falcon signature, there have been substantial efforts to optimize and analyze NTRU-based signatures. In particular, Mitaka [7] was proposed to simplify the signing procedure of Falcon based on a hybrid sampler, making it easier to implement. Furthermore, Antrag [8] introduced a novel

trapdoor generation algorithm based on uniform annulus sampling, and makes MITAKA as secure as FALCON. More recently, Zhang et al. [32] considered GPV preimage sampling under a weak smoothness condition and presented FALCON$^{\text{ws}}$ and ANTRAG$^{\text{ws}}$, achieving smaller signature and public key sizes than FALCON and ANTRAG. Concurrent with our work, [32] focuses on relaxing the smoothing parameter, while our approach explores a different optimization direction by reducing the quality parameter α.

1.2 Trapdoor Generation of FALCON

The security of FALCON is closely related to the Gaussian width $\sigma = \alpha \cdot \eta_\epsilon(\mathbb{Z}^n)$ used in Klein-GPV sampling. Here, α represents the quality parameter of the FALCON trapdoor, and $\eta_\epsilon(\mathbb{Z}^n)$ denotes the smoothing parameter of $\mathbb{Z}^n$. A smaller σ makes the corresponding Approximate-CVP problem more difficult. Consequently, a smaller value of α ($\alpha \geqslant 1$, with 1 being optimal) indicates a higher level of security for FALCON.

To achieve the desired quality parameter α, the secret polynomials f and g must satisfy the following two conditions:

$$|(f, g)|^2 \leq \alpha^2 q \tag{1}$$

$$|(\frac{q\overline{f}}{f * \overline{f} + g * \overline{g}}, \frac{q\overline{g}}{f * \overline{f} + g * \overline{g}})|^2 \leq \alpha^2 q \tag{2}$$

where α, q, $\overline{f}, \overline{g}$ denote the quality parameter, the modulus, and the Hermitian adjoints of f and g, respectively. The FALCON trapdoor generation process relies on a trial-and-repeat approach. This involves sampling a candidate pair (f, g) that satisfies condition (1) and then testing whether it also meets condition (2). If the second condition is not satisfied, the sampling process is repeated. As previously noted, a smaller α corresponds to a higher security level. However, reducing α increases the number of repetitions required, significantly slowing down the trapdoor generation process. In FALCON, $\alpha = 1.17$ is chosen because it is the smallest value that maintains a reasonable number of repetitions while balancing security and efficiency.

The number of repetitions for different values of α is presented in Table (1). As an example, according to our experiments, achieving a quality parameter $\alpha = 1.08$ would require an average number of four million repetitions, which is highly inefficient in practice.

Consequently, a natural question is: *Can we generate FALCON trapdoors with smaller α in an efficient way?*

1.3 Our Contributions

To achieve a significantly smaller value of α, it is necessary to eliminate the large number of repetitions required in the sampling process. Instead of relying on the trial-and-repeat approach, we aim to sample directly from the region defined

Table 1: Number of repetitions for different values of α

α	Average number of repetitions for FALCON-512
1.17	10
1.14	94
1.11	44197
1.08	4226157

by the two required conditions (1) and (2). However, a major challenge arises: this region possesses a highly irregular and complex geometric structure, making direct sampling a difficult task.

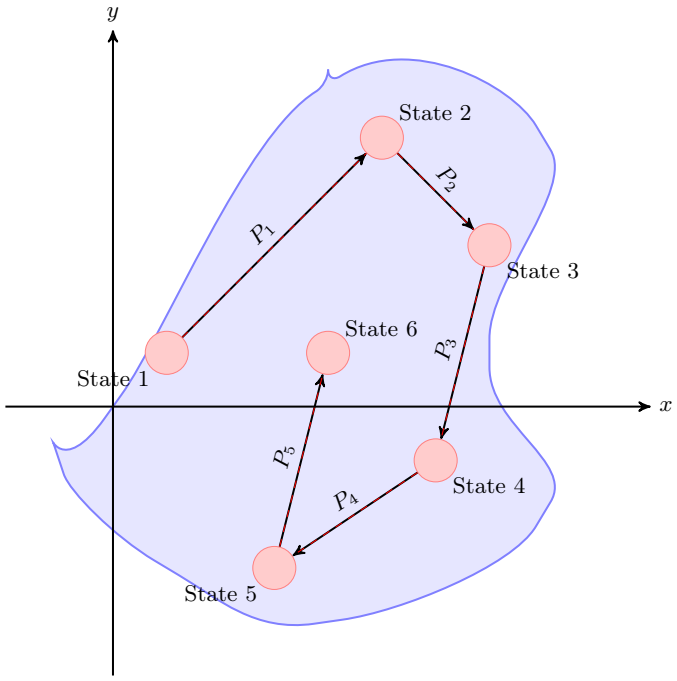


Fig. 1: Markov Chain Monte Carlo inside a peculiar region in 2D Plane

As a result, we utilize Markov Chain Monte Carlo (MCMC) methods to simulate the distribution within the peculiar region. The core idea behind MCMC is to begin with an arbitrary point within the target region and perform random walks within the region. After a sufficiently large number of steps, the point's position converges to a random sample from the desired distribution in the region. An example of this process is shown in Figure (1).

Among the various MCMC algorithms available, we select the Gibbs sampler [12,30,29] for our implementation. Typically, a sample from the target distribution is represented as a d-dimensional vector $\mathbf{x} = (x_1, \ldots, x_d)$. The Gibbs sampler is particularly useful when the conditional distribution of a single coordinate, given all other coordinates, is known. Specifically, the Gibbs sampler works as follows:

- One iteration of the Gibbs sampler consists of the following steps:
  - for each $i \in 1, \cdots, d$:
    1. Sample x_i conditioned on all the other coefficients: x_j $(j \neq i,\ j \in \{1, \cdots, d\})$.
    2. Update x_i according to the sampled value and keep all the other coefficients unchanged.
- Repeat the process for a specified number of iterations.

Perhaps interestingly, a key mathematical property of Markov Chain theory ensures that after a sufficient number of iterations, the output distribution of the Gibbs sampler closely approximates the original joint probability distribution.

In order to generate trapdoors for FALCON, we need to sample polynomials f and g such that they satisfy the two conditions (1) and (2). While these conditions can be expressed in terms of the coefficients of f and g, condition (2) does not have a straightforward closed-form representation due to the convolution operation involved in polynomial multiplication. However, if we represent f and g in their FFT form (i.e., canonical embeddings), both conditions become much simpler and more elegant, shown in condition (3) and (4):

$$|\varphi_1(f)|^2 + |\varphi_1(g)|^2 + \cdots + |\varphi_d(f)|^2 + |\varphi_d(g)|^2 \leq d\alpha^2 q \tag{3}$$

$$\frac{q^2}{|\varphi_1(f)|^2 + |\varphi_1(g)|^2} + \cdots + \frac{q^2}{|\varphi_d(f)|^2 + |\varphi_d(g)|^2} \leq d\alpha^2 q \tag{4}$$

where $(\varphi_1(f), \cdots, \varphi_d(f), \varphi_1(g), \cdots, \varphi_d(g))$ denote the canonical embeddings of f and g. Since complex embeddings always appear as conjugate pairs[4], up to a permutation, we have:

$$|\varphi_i(f)|^2 = |\varphi_{i+\frac{d}{2}}(f)|^2 \quad \text{and} \quad |\varphi_i(g)|^2 = |\varphi_{i+\frac{d}{2}}(g)|^2 \qquad (i = 1, \cdots, d/2)$$

Thus we only need to consider the first $d/2$ coefficients of the embeddings.

Let $x_i = \frac{|\varphi_i(f)|^2 + |\varphi_i(g)|^2}{q}$ $(i = 1, \cdots, d/2)$, then conditions (3) and (4) become

$$x_1 + \cdots + x_{d/2} \leq \frac{d}{2}\alpha^2 \tag{5}$$

$$\frac{1}{x_1} + \cdots + \frac{1}{x_{d/2}} \leq \frac{d}{2}\alpha^2 \tag{6}$$

[4] For power-of-2 cyclotomics, no real embeddings exist.

One notable feature of these two formulas is that computing the conditional distribution of one coefficient, conditioned on all other coefficients, is quite straightforward. For example, if $x_2 = x_3 = \cdots = x_{d/2} = 1$, then the conditional distribution of x_1 is simply a uniform interval:

$$\frac{1}{\frac{d}{2}\alpha^2 - \frac{d}{2} + 1} \leq x_1 \leq \frac{d}{2}\alpha^2 - \frac{d}{2} + 1$$

Upon closer inspection, this scenario is particularly well-suited for the Gibbs sampler because the conditional distribution is a uniform interval, which is both simple to compute and sample from. Consequently, we use the Gibbs sampler to explore the region defined by (5) and (6). After a sufficient number of iterations, we obtain (f, g) in FFT form, which we then transform back into ring elements f and g using the inverse FFT.

However, the resulting f and g may have real-valued coefficients, so we round each coefficient to the nearest integer [8]. While rounding introduces a small probability that f and g fail to achieve the desired quality parameter α due to rounding noise, there is a straightforward solution: during Gibbs sampling, we use a slightly smaller quality parameter $\alpha - \epsilon$, where ϵ is a small constant (e.g., 0.005). This ensures that the final rounded f and g meet the target quality α with high probability.

A natural question arises regarding the number of iterations required for the Gibbs sampler to produce a sufficiently uniform sample. While a formal theoretical proof is challenging, we address this empirically by conducting statistical tests on the sampler's output distribution. Our experiments show that setting the number of iterations to 2500 is sufficient for the Gibbs sampler to pass the statistical tests reliably.

As a result, our approach offers two advantages. First, it improves the security of Falcon by achieving α values arbitrarily close to 1. This improvement allows us to strengthen Falcon's security level. For example, Falcon-512 currently provides only 120 bits of classical security, which falls short of the NIST level one requirement of 128 bits. Our method effectively bridges this gap.

Second, it eliminates the need for a Gaussian sampler. Traditional discrete Gaussian samplers are not only difficult to implement but also challenging to secure against side-channel attacks and less efficient compared to uniform samplers. In contrast, our approach relies solely on uniform sampling over an interval, which simplifies the implementation. As indicated in recent studies, the complex Gaussian samplers used in Falcon are often the primary targets of timing attacks [10] and power analysis [17,33]. By removing this vulnerable component, we eliminate a major source of secret leakage. Although it is still a universal challenge to completely eliminate the sources of side-channel attacks in NTRU-based schemes, our approach has been less difficult to protect against side-channels compared with Falcon.

Moreover, the Gibbs sampling technique proposed in this work is not specific to Falcon. Rather, it constitutes a generic approach that may be applicable to

other NTRU-based schemes sharing similar structural properties, such as BAT KEM scheme [9].

In addition, it also needs to be emphasized that *the practical NTRU-based cryptographic schemes have no security proofs relating them to standard lattice problems*, such as FALCON and MITAKA. Our scheme has no exception, there is no security proof. Similar to FALCON, we assess the concrete security of the resulting signature scheme. A detailed discussion of security will be provided in Section 4.

2 Preliminaries

2.1 Cyclotomic Fields

Let m be a positive integer, and let $d = \phi(m)$ be the degree of the m-th cyclotomic polynomial $\Phi(m)$, where ϕ denotes Euler's totient function. Let ζ be a primitive m-th root of unity. We define $\mathcal{K}$ as the cyclotomic field associated with $\Phi(m)$, and its ring of integers $\mathcal{R}$ as $\mathbb{Z}[\zeta]$. It follows that $\mathcal{K} \cong \mathbb{Q}[x]/(\Phi_m(x))$ and $\mathcal{R} \cong \mathbb{Z}[x]/(\Phi_m(x))$. Both $\mathcal{K}$ and $\mathcal{R}$ are contained in $\mathcal{K}_{\mathbb{R}} := \mathcal{K} \otimes \mathbb{R} = \mathbb{R}[x]/(\Phi_m(x))$.

Each element $f = \sum_{i=1}^{d} f_i \zeta^i \in \mathcal{K}_{\mathbb{R}}$ can be represented by its coefficient vector $(f_1, \cdots, f_d)$. The field isomorphism $\zeta \to \zeta^{-1} = \overline{\zeta}$ corresponds to complex conjugation, and $\overline{f}$ is the image of f under this automorphism.

The cyclotomic field $\mathcal{K}$ has d complex embeddings, typically referred to as canonical embeddings, denoted by $\varphi_i : \mathcal{K} \to \mathbb{C}$. As polynomials, these d embeddings correspond to evaluating f at all the primitive m-th roots of the m-th cyclotomic polynomial.

2.2 NTRU Lattices

Given $f, g \in \mathcal{R}$ such that f is invertible modulo some prime $q \in \mathbb{Z}$ (in FALCON, $q = 12289$ to make it NTT-friendly[5]), we let $h = f^{-1}g \bmod q$. The NTRU module determined by h is $\{(u, v) \in \mathcal{R}^2 : uh - v = 0 \bmod q\}$. Two bases of this free module is of particular interest:

$$B_h = \begin{bmatrix} 1 & h \\ 0 & q \end{bmatrix} \qquad B_{f,g} = \begin{bmatrix} f & g \\ F & G \end{bmatrix}$$

where $F, G \in \mathcal{R}$ such that $fG - gF = q$. Usually, in the form of coefficient embedding, this module is seen as a 2d-dimensional lattice, where each polynomial is interpretted as an anticirculant matrix as following:

$$A_f = \begin{bmatrix} f_1 & f_2 & f_3 & \cdots & f_d \\ -f_d & f_1 & f_2 & \cdots & f_{d-1} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ -f_2 & -f_3 & \cdots & \cdots & f_1 \end{bmatrix}$$

[5] In FALCON, q is a prime of the form $k \cdot 2n + 1$ in order to maximize the efficiency of the NTT. The smallest prime of this form is $q = 12 \cdot 1024 + 1 = 12289$.

2.3 Gaussian and Chi-Squared Distributions

For $\mu \in \mathbb{R}$ and $\sigma > 0$ we let $\mathcal{N}(\mu, \sigma^2)$ be the normal distribution of mean μ and standard deviation σ, that is, the continuous distribution over $\mathbb{R}$ with density proportional to $e^{-(x-\mu)^2/(2\sigma^2)}$. In higher dimensions, for Σ a positive definite matrix and a vector $\mu \in \mathbb{R}^k$, we let $\mathcal{N}(\mu, \Sigma)$ be the normal distribution of density proportional to $e^{-\frac{1}{2}(x-\mu)^t \Sigma^{-1}(x-\mu)}$.

Let $T \sim \mathcal{N}(\mu, \sigma^2 \mathbf{I}_k)$ be a k-dimensional spherical normal random vector. The random variable $\|T\|^2$ follows *a non central chi-squared distribution of degree k, non-centrality $c := \|\mu\|^2$ and scaling σ^2*, denoted by $\chi^2(k, \sigma^2; c)$. Its expectation, variance and cumulative distribution function are described by the following classical result.

Lemma 1. *Let U be a random variable distributed as $\chi^2(k, \sigma^2; c)$. We have $\mathbb{E}[U] = \sigma^2 k + c$ and $\mathrm{Var}[U] = 2\sigma^2(\sigma^2 k + 2c)$. For $0 \leq a < b$, we have $\mathbb{P}[a \leq U \leq b] = Q_{k/2}(\sqrt{c}/\sigma, \sqrt{a}/\sigma) - Q_{k/2}(\sqrt{c}/\sigma, \sqrt{b}/\sigma)$, where $Q_{k/2}$ is the Marcum Q-function[6] of order $k/2$.*

We note that the independent sum of a $\chi^2(k, \sigma^2; c)$ variable and a $\chi^2(k', \sigma^2; c')$ variable, for the same scaling σ^2, follows a $\chi^2(k + k', \sigma^2; c + c')$ distribution.

2.4 Gibbs Sampler

The Gibbs sampler is a type of Markov Chain Monte Carlo (MCMC) algorithm that can be used to approximate a multivariate joint probability distribution in cases where direct sampling is difficult, but the conditional distribution of one variable conditioned on the other variables is known or easy to compute. To elaborate, suppose we want to obtain a sample $X = (x_1, \cdots, x_d)$ from a joint probability distribution. The algorithm operates as follows:

- Begin from some initial value $X^{(0)}$, which is an arbitrarily chosen random point in the required sampling region.
- One iteration of Gibbs sampler is: for $j \in \{1, \cdots, d\}$,
  - Sample x_j conditioned on all the other coefficients of X.
  - Update x_j and keep the other coefficients unchanged.
- Repeat for a certain number of iterations.

The properties of the Markov chain guarantee that the output distribution of the Gibbs sampler will closely approximate the original joint probability distribution after a sufficient number of iterations.[7]

[6] For a detailed definition of Marcum Q-function, please refer to Appendix A.
[7] These are standard results in Markov chain theory (see e.g., [11] Sec. 5.2).

Algorithm 1: FALCON trapdoor generation (simplified version)

Input: The degree d, a target quality parameter α, and modulus q
Result: $f, g \in \mathscr{R}^2$ that reaches a quality parameter α .

1 $\sigma \leftarrow \alpha\sqrt{\frac{q}{2d}}$
2 Sample the coefficients of f, g from discrete Gaussian distribution with standard deviation σ
3 Norm$\leftarrow max\big(|(f,g)|, |(\frac{q\overline{f}}{f*\overline{f}+g*\overline{g}}, \frac{q\overline{g}}{f*\overline{f}+g*\overline{g}})|\ \big)$
4 if Norm $> \alpha\sqrt{q}$, go to step 2.
5 Use NTRUsolver to find F, G s.t. $fG - gF = q$.
6 **return** (f, g, F, G)

3 Generating FALCON Trapdoors with Gibbs Sampler

3.1 Trapdoors Generation in FALCON

Algorithm 1 presents a simplified version of FALCON's trapdoor generation process, with the portion responsible for generating F and G using NTRU solver simplified due to its irrelevance to our context. In this algorithm, both f and g are generated coefficient-wise from a discrete Gaussian distribution with standard deviation $\sigma = \alpha\sqrt{\frac{q}{2d}}$. This yields a sample (f, g) whose Euclidean length follows the Chi-distribution[8] with degree of freedom d. By applying standard concentration inequalities to the Chi-distribution, we can deduce that the Euclidean length $|(f, g)|$ remains close to $\alpha\sqrt{q}$, which implies that $|(f, g)| \leq \alpha\sqrt{q}$ with high probability. However, we must also verify that $|(\frac{q\overline{f}}{f*\overline{f}+g*\overline{g}}, \frac{q\overline{g}}{f*\overline{f}+g*\overline{g}})| \leq \alpha\sqrt{q}$. If this condition is not met, then the entire sampling procedure must be repeated.

As mentioned earlier, a smaller quality parameter α leads to both shorter signature sizes and higher security levels. Ideally, the smallest possible α would be chosen to maximize security. However, as α decreases, the number of required repetitions grows rapidly. For example, achieving $\alpha = 1.08$ demands millions of repetitions, making it impractical. Therefore, a reasonable compromise is to set α to 1.17, which strikes a balance by requiring a manageable number of repetitions while maintaining efficiency.

3.2 Representing Sampling Regions in Fourier Domain

Recall that in order to achieve a quality parameter α, two conditions (1) and (2) need to be satisfied as follows:

$$|(f, g)|^2 \leq \alpha^2 q$$

[8] Although the sample (f, g) actually follows a discrete Gaussian distribution rather than a continuous one, it can be argued that the properties of the discrete case approximates that of the continuous case.

$$|(\frac{q\overline{f}}{f*\overline{f}+g*\overline{g}}, \frac{q\overline{g}}{f*\overline{f}+g*\overline{g}})|^2 \leq \alpha^2 q$$

Instead of using trial-and-repeat approach, a more natural approach [8] is to sample directly from the region defined by these conditions to avoid the need for a large number of repetitions.

If f and g are represented in coefficient form, then the condition (1) can be expressed as:

$$f_1^2 + g_1^2 + f_2^2 + g_2^2 + \cdots + f_d^2 + g_d^2 \leq \alpha^2 q$$

However, there is no simple formula for condition (2) since it involves polynomial multiplication by convolution. Therefore, it seems quite challenging to sample directly from the region defined by (1) and (2). Nevertheless, if f and g are represented in Fourier domain (in other words, caonical embeddings), thanks to the isotropic properties (up to a scaling factor) of cyclotomic fields, both conditions can be represented in a neat and elegant fashion, i.e., conditions (3) and (4):

$$|\varphi_1(f)|^2 + |\varphi_1(g)|^2 + \cdots + |\varphi_d(f)|^2 + |\varphi_d(g)|^2 \leq d\alpha^2 q$$

$$\frac{q^2}{|\varphi_1(f)|^2 + |\varphi_1(g)|^2} + \cdots + \frac{q^2}{|\varphi_d(f)|^2 + |\varphi_d(g)|^2} \leq d\alpha^2 q$$

The detailed derivation of (3) and (4) can be obtained from Appendix B. Since complex embeddings always appear as conjugate pairs, up to a permutation of index, we have:

$$|\varphi_i(f)|^2 = |\varphi_{i+\frac{d}{2}}(f)|^2 \quad \text{and} \quad |\varphi_i(g)|^2 = |\varphi_{i+\frac{d}{2}}(g)|^2 \qquad (i = 1, \cdots, d/2)$$

So only the first $d/2$ coefficients of the embeddings need to be considered.

Let $x_i = \frac{|\varphi_i(f)|^2 + |\varphi_i(g)|^2}{q}$ $(i = 1, \cdots, d/2)$, then conditions (3) and (4) become conditions (5) and (6), respectively:

$$x_1 + \cdots + x_{d/2} \leq \frac{d}{2}\alpha^2$$

$$\frac{1}{x_1} + \cdots + \frac{1}{x_{d/2}} \leq \frac{d}{2}\alpha^2$$

However, despite the fact that the sampling region defined by (5) and (6) has a neat form, the geometric shape of the region is peculiar, making direct sampling difficult.

3.3 Sampling Falcon Trapdoors via Gibbs Sampler

Conditional distribution of x_i. Now we define $X = (x_1, \cdots, x_{d/2})$ as a $d/2$ dimensional variable, where $x_i = \frac{|\varphi_i(f)|^2 + |\varphi_i(g)|^2}{q}$ $(i = 1, \cdots, d/2)$ as previously mentioned. It is easy to know from (5) and (6) that for any index i, the

conditional distribution of x_i (conditioned on the other $(d/2-1)$ coefficients), represented as the following interval:

$$\frac{1}{\frac{d}{2}\alpha^2 - \sum_{j\neq i}\frac{1}{x_j}} \leq x_i \leq \frac{d}{2}\alpha^2 - \sum_{j\neq i} x_j \tag{7}$$

This conditional distribution is fairly easy to compute and sample from, thus making it a good fit for the Gibbs sampler.

Running Gibbs sampler on $\boldsymbol{X}$. As described in Algorithm 2, we initialize X to $(1,\cdots,1)$ [9]. The Gibbs sampler is then run over the random variable X for a certain number of iterations N, after which we obtain the value $X = (x_1,\cdots,x_{d/2})$.

Computing the embeddings. Once the values of x_i $(i = 1,\ldots,d/2)$ are determined, the next step is to compute the embeddings $(\varphi_i(f),\varphi_i(g))$ for $i = 1,\ldots,d/2$. Since $\varphi_i(f)$ and $\varphi_i(g)$ are empirically observed to be independent (this will be elaborated in detail in section 3.4), we sample a uniformly random angle $\theta \in (0,\pi/2)$ and compute the magnitudes of the embeddings as follows:

$$|\varphi_i(f)| = \sqrt{qx_i}\cdot\cos\theta \quad \text{and} \quad |\varphi_i(g)| = \sqrt{qx_i}\cdot\sin\theta \qquad (i = 1,\ldots,d/2).$$

Next, to determine the embeddings, we sample a uniform random angle γ on the complex plane and compute:

$$\varphi_i(f) = |\varphi_i(f)|\cdot e^{i\gamma_f} \quad \text{and} \quad \varphi_i(g) = |\varphi_i(g)|\cdot e^{i\gamma_g} \qquad (i = 1,\ldots,d/2),$$

where $e^{i\gamma_f}$ represents the complex number $\cos\gamma_f + i\sin\gamma_f$.

Inverse FFT and rounding. The final step is to map the Fourier domain values $\big(\varphi_1(f),\ldots,\varphi_{d/2}(f),\varphi_1(g),\ldots,\varphi_{d/2}(g)\big)$ back to the coefficient form $(\tilde{f},\tilde{g})$. However, it is important to note that the coefficients of $(\tilde{f},\tilde{g})$ may not necessarily be integers. A simple solution is to round each coefficient to the nearest integer and then verify whether the resulting f,g satisfy the desired requirements. If not, the sampling procedure is repeated.

A practical approach to address this issue is as follows: to achieve a quality parameter α, we sample from a region targeting a slightly smaller quality $\beta = \alpha - \epsilon$, where ϵ is a very small constant. After rounding each coefficient of $(\tilde{f},\tilde{g})$ to the nearest integer, the resulting (f,g) will achieve a quality α with high probability.

[9] The starting point can be any arbitrary point within the required space, as the choice does not significantly impact the process. For simplicity, we select $(1,\ldots,1)$.

Algorithm 2: Sampling FALCON Trapdoors via Gibbs Sampler

Input: The degree d, a target quality parameter α, a small constant ϵ, number of iterations N, and modulus q

Result: $f, g \in \mathscr{R}^2$ that reaches a quality parameter α.

1 $X = (x_1, \cdots, x_{d/2}) \leftarrow (1, \cdots, 1)$.
2 $\beta = \alpha - \epsilon$
3 **for** $1 \leqslant k \leqslant N$ **do**
4 **for** $1 \leqslant i \leqslant d/2$ **do**
5 Compute the conditional distribution of x_i conditioned on all the other coefficients, which is a uniform interval: $\frac{1}{\frac{d}{2}\beta^2 - \sum_{j \neq i} \frac{1}{x_j}} \leq x_i \leq \frac{d}{2}\beta^2 - \sum_{j \neq i} x_j$
6 Sample x_i uniformly from that interval. Update x_i and keep all the other coefficients x_j $(j \neq i)$ unchanged.
7 **end for**
8 **end for**
9 **for** $1 \leqslant j \leqslant d/2$ **do**
10 sample θ uniformly from $(0, \pi/2)$
11 sample γ_f uniformly from $(0, 2\pi)$
12 sample γ_g uniformly from $(0, 2\pi)$
13 $\varphi_j(f) = \sqrt{qx_j} \cdot \cos\theta \cdot e^{i\gamma_f}$
14 $\varphi_j(g) = \sqrt{qx_j} \cdot \sin\theta \cdot e^{i\gamma_g}$
15 **end for**
16 $\tilde{f} \leftarrow \varphi^{-1}(\varphi_1(f), \ldots, \varphi_{d/2}(f)) \in \mathscr{K}_{\mathbb{R}}$
17 $\tilde{g} \leftarrow \varphi^{-1}(\varphi_1(g), \ldots, \varphi_{d/2}(g)) \in \mathscr{K}_{\mathbb{R}}$
18 $f \leftarrow \lfloor \tilde{f} \rceil$
19 $g \leftarrow \lfloor \tilde{g} \rceil$
20 if (f, g) does not reach a quality α, go to step 1.
21 **return** (f, g)

3.4 Error Analysis

We have mentioned above that taking the magnitudes of the embeddings of $\tilde{f}$ and $\tilde{g}$ with targeted quality parameter α does not always result in f and g of the same quality α after rounding, but that the probability increased greatly when choosing $\tilde{f}$ and $\tilde{g}$ with embedding magnitudes in a narrower sampling region with quality parameter $\beta = \alpha - \epsilon$. In this section, we would like to quantify this claim, based both on a heuristic analysis of the success probability, and on simulation data. Concretely, write $e = (e_f, e_g) = \left(f - \tilde{f}, g - \tilde{g}\right) \in \mathscr{K}_{\mathbb{R}}^2$ for the error term introduced by rounding.

In the polynomial basis, we write:

$$e_f = \sum_{j=0}^{d-1} e_f^{(j)} x^j$$

and similarly for e_g. Heuristically, we expect the coefficients $e_f^{(j)}$ and $e_g^{(j)}$ to behave essentially like independent uniform random variables in $[-1/2, 1/2)$.[10] This is well-supported by experiments (see Fig. 5(a) in Appendix C).

Now consider a single embedding φ_i that is defined by the evaluation at some primitive m-th root of unity $\zeta = e^{i\theta}$, and we have:

$$\varphi_i(e_f) = x_i + iy_i \quad \text{with} \quad x_i = \sum_{j=0}^{d-1} e_f^{(j)} \cos(j\theta) \quad \text{and} \quad y_i = \sum_{j=0}^{d-1} e_f^{(j)} \sin(j\theta).$$

This expresses the real and imaginary parts x_i, y_i of $\varphi_i(e_f)$ as the sum of d independent random variables, with d relatively large, so by the central limit theorem, $\varphi_i(e_f)$ should essentially behave like a normal random variable in $\mathbb{C}$, essentially determined by its expectation and covariance.

Now since $e_f^{(j)}$ has mean 0 and variance 1/12 for all j, we obtain that $\mathbb{E}[x_i] = \mathbb{E}[y_i] = 0$. Therefore, the pair (x_i, y_i) has mean 0, and its covariance matrix is easily expressed as follows:

$$\begin{aligned}\Sigma &= \sum_{j=0}^{d-1} \mathrm{Var}[e_f^{(j)}] \cdot \begin{bmatrix} \cos^2(j\theta) & \cos(j\theta)\sin(j\theta) \\ \cos(j\theta)\sin(j\theta) & \sin^2(j\theta) \end{bmatrix} \\ &= \frac{1}{12} \sum_{j=0}^{d-1} \frac{1}{2} \begin{bmatrix} 1+\cos(2j\theta) & \sin(2j\theta) \\ \sin(2j\theta) & 1-\cos(2j\theta) \end{bmatrix} = \frac{d}{24}\mathbf{I}_2 + E(\theta),\end{aligned}$$

where

$$E(\theta) = \frac{1}{12} \begin{bmatrix} \mathrm{Re}\, S(\theta) & \mathrm{Im}\, S(\theta) \\ \mathrm{Im}\, S(\theta) & -\mathrm{Re}\, S(\theta) \end{bmatrix}.$$

Thus, we expect that $\varphi_i(e_f)$ $(i = 1, \cdots, d/2)$ follows the normal distribution $\mathcal{N}(0, \Sigma)$, and the same argument applies to $\varphi_i(e_g)$ as well. Moreover, heuristically, those two normal distributions should be independent (this is again well-verified in practice: see Fig. 5(b)).

And this leads us to model the distribution of the embeddings of secret keys as follows.

Heuristic 1. Let $(f, g) \in \mathcal{K}^2$ a pair output by Algorithm 2, corresponding to $(\tilde{f}, \tilde{g}) \in \mathcal{K}_{\mathbb{R}}^2$ obtained from the executions of Algorithm 2. For the embedding φ_θ corresponding to the primitive root of unity $e^{i\theta}$, $(\varphi_\theta(f), \varphi_\theta(g))$ is distributed as

$$\big(\varphi_\theta(f),\ \varphi_\theta(g)\big) \sim \mathcal{N}\Big(\big(\varphi_\theta(\tilde{f}),\ \varphi_\theta(\tilde{g})\big)\,,\ \mathbf{I}_2 \otimes \Sigma \Big)$$

Moreover, the pairs $(\varphi_\theta(f), \varphi_\theta(g))$ as φ_θ ranges through all the embeddings of $\mathcal{K}$ are independently distributed.

[10] This is equivalent to saying that the distribution of $\tilde{f}$ and $\tilde{g}$ is uniform modulo $\mathscr{R}$ in $\mathscr{K}_{\mathbb{R}}$, which should indeed happen as soon as we have sufficient width (i.e., if we exceed a regularity metric analogous to the smoothing parameters for Gaussians).

At this point, we would therefore like to estimate the probability that the rounded pair (f, g) satisfies the quality condition, i.e., conditions (5) and (6):

$$x_1 + \cdots + x_{d/2} \leq \frac{d}{2}\alpha^2$$

$$\frac{1}{x_1} + \cdots + \frac{1}{x_{d/2}} \leq \frac{d}{2}\alpha^2$$

The region defined by these two inequalities has complicated geometry, so it is not easy to give a fully precise analysis, especially for inequality (6), since it involves reciprocal. Therefore, we adopt to approximate (6) using Taylor expansion. Expanding $\frac{1}{x_i}$ near the center s and we have:

$$\frac{1}{x_i} = \sum_{k=0}^{\infty} \frac{(-1)^k}{s^{k+1}}(x_i - s)^k$$

And we get an approximation of $\frac{1}{x_i}$ by keeping the first two terms.

$$\frac{1}{x_i} \approx \frac{1}{s} - \frac{1}{s^2}(x_i - s) = \frac{2}{s} - \frac{x_i}{s^2}$$

Consequently, applying this approximation to inequality (6), we get

$$\frac{d}{s} - \frac{(x_1 + \cdots + x_{d/2})}{s^2} \leq \frac{d}{2}\alpha^2$$

Combining inequality (5), we get

$$sd - \frac{s^2 d}{2}\alpha^2 \leq x_1 + \cdots + x_{d/2} \leq \frac{d}{2}\alpha^2$$

As mentioned in Heuristic 1, since $\varphi_i(f)$ and $\varphi_i(g)$ follows the normal distribution, we have:

$$\big(\varphi_i(f),\ \varphi_i(g)\big) \sim \mathcal{N}\Big(\ \big(\varphi_i(\tilde{f}),\ \varphi_i(\tilde{g})\big)\ ,\ \mathbf{I}_2 \otimes \Sigma\Big)$$

Since Falcon works with elements in number fields of the form $Q[x]/(\phi)$, with $\phi = x^n + 1$ for $n = 2^\kappa$ a power-of-two. We concentrate on the case of a power-of-two cyclotomic base ring, rather than the more general cyclotomic rings. In the case of power-of-two cyclotomic fields, the situation is made comparatively simple by the fact that $E(\theta) = 0$ for all embeddings as guaranteed by Lemma 1 in [8], and hence the covariance matrix Σ is just $\frac{d}{24}\mathbf{I}_4$. Then

$$\big(\varphi_i(f),\ \varphi_i(g)\big) \sim \mathcal{N}\Big(\ \big(\varphi_i(\tilde{f}),\ \varphi_i(\tilde{g})\big)\ ,\ \frac{d}{24}\mathbf{I}_4\Big),$$

with $x_i = \frac{|\varphi_i(f)|^2+|\varphi_i(g)|^2}{q}$ $(i = 1, ..., d/2)$, we want to estimate the success probability of Algorithm 2, i.e., the probability that:

$$(sd - \frac{s^2d}{2}\alpha^2)q \leq |\varphi_1(f)|^2 + |\varphi_1(g)|^2 + \cdots + |\varphi_{d/2}(f)|^2 + |\varphi_{d/2}(g)|^2 \leq \frac{d}{2}\alpha^2 q$$

The scaled squared norm $\frac{\|(\varphi_i(f),\varphi_i(g))\|^2}{d/24} = \frac{|\varphi_i(f)|^2+|\varphi_i(g)|^2}{d/24}$ follows a non central chi-squared distribution $\chi^2(4, 1; c)$ of degree 4, scaling $\sigma^2 = 1$ and non-centrality $c = \frac{|\varphi_i(\tilde{f})|^2+|\varphi_i(\tilde{g})|^2}{d/24}$, so $\sum_{i=1}^{d/2} \frac{\|(\varphi_i(f),\varphi_i(g))\|^2}{d/24}$ follows a new non central chi-squared distribution $\chi^2(2d, 1; c')$ of degree $2d$ and non-centrality $c' = \sum_{i=1}^{d/2} \frac{(|\varphi_i(\tilde{f})|^2+|\varphi_i(\tilde{g})|^2)}{d/24}$.

The cumulative distribution function (CDF) of noncentral chi-squared distribution is represented as

$$\mathbb{P}(Z \leqslant z) = 1 - Q_{\frac{k}{2}}(\sqrt{\lambda}, \sqrt{z})$$

where Q is the Marcum Q-function, k is the degree of freedom, and λ is the center of the corresponding noncentral chi-squared distribution. In our context, $k = 2d$ and $\lambda = \frac{\sum_{i=1}^{d/2}(|\varphi_i(\tilde{f})|^2+|\varphi_i(\tilde{g})|^2)}{d/24}$.

Let $X = \sum_{i=1}^{d/2} \|(\varphi_i(f), \varphi_i(g))\|^2$ and $Y = \frac{X}{d/24}$. Now recall that we want to estimate the probability of success $p_{\text{succ}} := \mathbb{P}\big[(sd - \frac{s^2d}{2}\alpha^2)q \leqslant X \leqslant \frac{d}{2}\alpha^2 q\big]$. We have

$$\begin{aligned}\mathbb{P}\big[(sd - \frac{s^2d}{2}\alpha^2)q \leqslant X \leqslant \frac{d}{2}\alpha^2 q\big] &= \mathbb{P}(\frac{(sd - \frac{s^2d}{2}\alpha^2)q}{d/24} \leqslant Y \leqslant \frac{\frac{d}{2}\alpha^2 q}{d/24}) \\ &= -Q_d(\sqrt{\lambda}, \frac{\sqrt{\frac{d}{2}\alpha^2 q}}{\sqrt{d/24}}) + Q_d(\sqrt{\lambda}, \frac{\sqrt{(sd - \frac{s^2d}{2}\alpha^2)q}}{\sqrt{d/24}})\end{aligned}$$

As guaranteed by our sampling algorithm 2, λ lies uniformly in the interval $[\frac{r}{d/24}, \frac{R}{d/24}]$ with

$$r = \big(sd - \frac{s^2d}{2}\beta^2\big)q \quad \text{and} \quad R = \frac{d}{2}\beta^2 q$$

mentioned before, where $\beta = \alpha - \epsilon$. We would like to know the expectation of $\mathbb{P}\big[(sd - \frac{s^2d}{2}\alpha^2)q \leqslant X \leqslant \frac{d}{2}\alpha^2 q\big]$ for λ uniformly distributed in $[\frac{r}{d/24}, \frac{R}{d/24}]$, and hence we obtain

$$\begin{aligned}&\mathbb{E}_\lambda\big(\mathbb{P}\big[(sd - \frac{s^2d}{2}\alpha^2)q \leqslant X \leqslant \frac{d}{2}\alpha^2 q\big]\big) \\ &= \frac{1}{\frac{R}{d/24} - \frac{r}{d/24}} \int_{\frac{r}{d/24}}^{\frac{R}{d/24}} \mathbb{P}\big[(sd - \frac{s^2d}{2}\alpha^2)q \leqslant X \leqslant \frac{d}{2}\alpha^2 q\big] d\lambda \\ &= \frac{1}{\frac{R}{d/24} - \frac{r}{d/24}} \int_{\frac{r}{d/24}}^{\frac{R}{d/24}} \Big(- Q_d(\sqrt{\lambda}, \frac{\sqrt{\frac{d}{2}\alpha^2 q}}{\sqrt{d/24}}) + Q_d(\sqrt{\lambda}, \frac{\sqrt{(sd - \frac{s^2d}{2}\alpha^2)q}}{\sqrt{d/24}})\Big) d\lambda\end{aligned}$$

This formula gives us an expression of the approximate average success probability p_{succ}. Although the formula is a bit complicated, there exists tools that can evaluate it numerically (e.g., Python package `scipy` or Matlab). This, fortunately, turns out to be a good model of what happens in practice: the theoretical values obtained from the integrals above matches the experimental success rate of our algorithm in practice.

As a result, with the model of independent embeddings of the error vector, and with parameters $s = 1$, $q = 12289$, $d = 512$ and $\alpha = 1.04$, we would get a good success probability of around 99.4% for the whole vector when taking a margin factor of $\epsilon = 0.005$, which is the value we pick in our parameter selection.

4 Security Analysis

Provably secure instantiations of NTRU-based GPV-like signatures are possible, as shown in [28]. However, these constructions lead to highly impractical parameters. For instance, the modulus q is $n^{7+\mathcal{O}(1)}$, and the standard deviation of f and g is similarly quite large. As a result, the NTRU-based signatures that aim for practicality typically resort to parameter choices without security reductions relating them to standard lattice problems. Instead, these schemes rely instead on the so-called "NTRU assumption", which claims that for their specific key generation algorithm, the public key $h = f^{-1}g \mod q$ is computationally indistinguishable from random. The GPV framework is then applied to argue security of these signatures based on the NTRU assumption, combined with Module-SIS. This approach is also observed in schemes like Falcon and Mitaka, where no security proof has been provided for their key generation algorithm.

This assumption is believed to hold even for much more extreme choices of distribution than those used in signature schemes. Consequently, although Falcon and Mitaka generate f and g as discrete Gaussians and they also introduce additional constraints that skew their distributions, this approach has not led to any known attacks to date.

Our algorithm faces a similar situation. The security analysis in this section aims to estimate the complexity of the best attacks against *key recovery attacks* and *signature forgery*, rather than security proofs. While the analysis does not provide a proof of the absence of attacks against the key generation algorithm, it is important to note that no such proof exists for other schemes as well, such as Falcon, Mitaka, or other efficient NTRU-based constructions either.

Since our approach only reduces the value of α in Falcon, the security estimate remains essentially the same as that of the original Falcon. Nevertheless, we include the details for completeness.

4.1 Key Recovery Attack

Key recovery involves extracting the private key (i.e., f and g) from the public parameters q and h. The most efficient attacks rely on lattice reduction. This method constructs an algebraic lattice over $\mathcal{R}$, spanned by the vectors $(q, 0)$ and

$(h, 1)$ (i.e., the public NTRU key basis), and searches for the lattice vector (f, g) with norm bounded by $\sqrt{2d} \cdot \sigma_{\{f,g\}}$.

Let ℓ represent the $(2d-B)$-th Gram-Schmidt norm, which approximates the shortest vector's norm in the lattice generated by the last B vectors projected orthogonally to the first $2d - B - 1$ vectors. A sieve algorithm performed on this projected lattice will recover all vectors with norms smaller than $\sqrt{\frac{4}{3}} \cdot \ell$. If the projection of the secret key lies among these vectors, the following condition must hold:

$$\sqrt{B} \cdot \sigma_{\{f,g\}} \leqslant \sqrt{\frac{4}{3}} \cdot \ell,$$

where $\sigma_{\{f,g\}} = \alpha\sqrt{\frac{q}{2d}}$. Using Babai's Nearest Plane algorithm, we can recover the secret key from its projection with high probability, as the remaining Gram-Schmidt norms are much larger than ℓ, making the secret key stand out.

From the lattice reduction algorithm DBKZ [21], the Gram-Schmidt norm is approximated as:

$$\ell = \left(\frac{B}{2\pi e}\right)^{1-\frac{d}{B}} \cdot \sqrt{q}.$$

Substituting this into the condition, we get:

$$\sqrt{B} \cdot \sigma_{\{f,g\}} \leqslant \sqrt{\frac{4}{3}} \cdot \left(\frac{B}{2\pi e}\right)^{1-\frac{d}{B}} \cdot \sqrt{q}.$$

Once the minimum block size B for the attack is determined, it is translated into concrete bit security using the core-SVP methodology, as employed in New Hope [1].

4.2 Signature Forgery Attack

Signature forgery amounts to finding a vector with a Euclidean norm smaller than a bound $\beta = \tau \cdot \sigma\sqrt{2d}$, where τ is the tailcut rate, and $\sigma\sqrt{2d}$ represents the expected length of a valid signature. Let B denote the minimum BKZ block size required for a successful forgery. According to the DBKZ algorithm [21], the success condition is:

$$\left(\frac{B}{2\pi e}\right)^{d/B} \cdot \sqrt{q} \leqslant \beta = \tau \cdot \sigma\sqrt{2d}.$$

To ensure that the signature distribution remains close to an ideal Gaussian (in terms of Rényi divergence), the standard deviation σ must satisfy the following condition [24]:

$$\sigma \geqslant \eta_\epsilon(\mathbb{Z}^{2d}) \cdot \mathrm{GS}_{\max},$$

where:

$$\eta_\epsilon(\mathbb{Z}^{2d}) \geqslant \frac{1}{\pi} \cdot \sqrt{\frac{\log(4d(1+1/\epsilon))}{2}} \cdot \alpha \cdot \sqrt{q}.$$

Here, $\epsilon = 1/\sqrt{Q_s \cdot \lambda}$, where Q_s is the maximum number of queries (typically 2^{64}) and λ is the security parameter. After determining the minimum block size B for the attack, we convert it into concrete bit security using the core-SVP methodology from New Hope [1].

As shown in Table 2, by achieving smaller α values with the Gibbs sampler and selecting a slightly smaller τ, we can improve the security of Falcon-512 by a few bits.[11]

Table 2: Security estimate of Falcon-512 using Gibbs sampler

	α		Block size	Security bits (c/q)
Key recovery	1.17		458	133/121
	1.04		441	128/116
	α	τ	Block size	Security bits (c/q)
Signature forgery	1.17	1.1	411	120/109
	1.04	1.04	439	128/116

5 On the Output Distribution of Our New Trapdoor Sampler

Several questions naturally arise regarding the output distribution of our new trapdoor sampler. First, does reducing α still provide sufficient entropy for the key space? Second, how does it compare to the original trapdoor sampler used in Falcon? Third, how many iterations are required to ensure that the output distribution is close to uniform? In this section, we address these questions in detail.

5.1 Entropy of Falcon Trapdoor with Smaller α

One potential concern is whether decreasing the value of α compromises the entropy of the key space. However, this does not pose a significant problem.

To illustrate, consider an extreme scenario where we focus only on the boundary of the sampling region:

$$x_1 + \cdots + x_{d/2} = \frac{d}{2} \times 1.04^2$$

$$\frac{1}{x_1} + \cdots + \frac{1}{x_{d/2}} = \frac{d}{2} \times 1.04^2$$

[11] For Falcon-1024, we could not increase the security bits, because the estimated security bits for key recovery is lower than those of signature forgery.

Even in this case, the number of solutions to these equations is so large that entropy remains sufficient.

Now, consider an even more extreme case where there is only one solution, $(f, g) = (f_1, \cdots, f_d, g_1, \cdots, g_d)$. Even in this scenario, permuting the coefficients generates an exponential number of valid candidates for key generation. Consequently, choosing a smaller α, such as 1.04, still provides ample entropy for key generation.

5.2 Comparison of Trapdoor Distributions: FALCON vs. Gibbs Sampler

Let us first examine the trapdoor generation process in FALCON. The coefficients of f and g are sampled from a discrete Gaussian distribution with a small standard deviation. This results in a trapdoor distribution that is not uniform. The intuition behind this is that the entropy is sufficient, so a uniform distribution is unnecessary.

However, from a theoretical perspective, the "ideal" distribution would sample uniformly from the set of (f, g) pairs that satisfy the constraints. In this context, if the Gibbs sampler produces a distribution that is closer to uniform, our new trapdoor sampler can be considered "better" in this specific sense.

Of course, this raises the question: how close is the distribution of the Gibbs sampler to uniform? We explore this question in subsequent discussions.

5.3 Number of Iterations for Convergence

A critical parameter for the Gibbs sampler is the number of iterations, N. According to Markov chain theory, as $N \to \infty$, the chain converges to the true posterior distribution, ensuring that the samples accurately represent the desired distribution.

However, for any practical value of N, it is generally challenging to determine whether convergence has been achieved or how close the output distribution is to the true posterior. To address this, we perform statistical tests to evaluate the output distribution and ensure it approximates the desired behavior.

There are several commonly used convergence diagnostic methods for the Gibbs sampler, including Heidelberger and Welch's [18] and Geweke's [15]. We will discuss the methods in a bit more detail.

Heidelberger and Welch's Method. The Heidelberg and Welch convergence diagnostic consists of two parts: the stationarity test and the halfwidth test.

Initially, this diagnostic tests whether stationarity of the Markov chain is attained using the values from an MCMC output. The number of iterations of this Markov chain is N, and a significance level α is defined, typically set to 0.05. The Cramer-von-Mises test statistic is then computed for the entire chain, followed by the calculation of the corresponding p-value. The null hypothesis of stationarity is tested by comparing the p-value to the predefined significance

level α. If the p-value is less than α, the null hypothesis is rejected, then the first 10% of the samples are discarded and the test reapplied to the resulting chain. This processes is repeated until the null hypothesis is accepted or 50% of the chain is discarded, at which point the chain is declared to be non-stationary.

Subsequently, once the stationarity test has passed, the portion of the chain that was not discarded is used for the halfwidth test. The test is performed by computing the mean of the resulting chain, along with the halfwidth of the $(1-\alpha)100\%$ confidence interval (typically with $\alpha = 0.05$ corresponding to a 95% interval).

The chain is considered to pass the halfwidth test if the halfwidth is smaller than the product of the mean and a user-specified level of accuracy ϵ (often set to 0.1). Otherwise, the chain is considered to have failed the test and larger iterations are needed to achieve sufficient accuracy. If all the variables have passed the tests, convergence is considered to have been reached.

Geweke's Method. Geweke proposed a convergence diagnostic based on standard time-series methods. It is based on a single chain and is appropriate when convergence of the mean (of some function) of the sampled variable is of interest. The chain is divided into 2 windows containing the first 10% and the last 50% of the iterates. If the whole chain is stationary, the means of the values early and late in the sequence should be similar. The convergence diagnostic Z is the difference between the 2 means divided by the asymptotic standard error of their difference. As $N \to \infty$, the sampling distribution of Z goes to $\mathcal{N}(0,1)$ if the chain has converged. Hence values of Z which fall in the extreme tails of $\mathcal{N}(0,1)$ indicate that the chain has not yet converged. To be more specific, according to the "three-sigma rule", parameters with $|Z| > 3$ indicate differences in the means between the first and the last set of iterations and hence non-convergence. Nevertheless, in multi-variable models, we allow a 5% of the calculated Z values to lie outside this range.

5.4 Experiment Results

The convergence assessment is performed by using BOA package (see [27]), with incremental iterations ranging from 500 to 4000.

As mentioned above, Heidelberger and Welch's and Geweke's convergence diagnostic methods were considered. After performing Gibbs sampling with $\alpha = 1.04$ and $N = 1024$ according to Steps 1 to 8 of Algorithm 2, we conducted Heidelberger and Welch's convergence diagnostics on the sampling results. As shown in Figure 2, the output indicates that after 2500 iterations, all tests passed, and convergence was achieved. Meanwhile, we also conducted Geweke's method on the sampling results at 2500 iterations, only less than 5% of the Z values fall within the extreme tails of distribution. So in the implementation, we set the number of iterations to 2500.

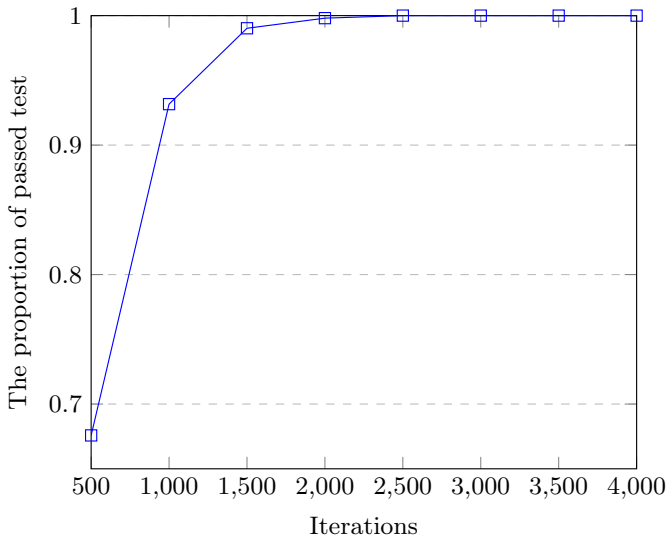


Fig. 2: Heidelberger and Welch's convergence assessment

6 Implementation and Comparison

We have implemented our trapdoor generation algorithm in portable C based on the source codes of Falcon and Mitaka [7] signature scheme. The source code for our implementations is made publicly available at `https://github.com/jjyydzay/Gibbs_Falcon`.

Since we only modify the process to generate the first basis vector (f, g) of Falcon key generation algorithm and the signature scheme is essentially identical to Falcon for signing and verification, we do not compare these two processes. Key generation consists of the new algorithm presented in this paper to generate (f, g), along with code to solve the NTRU equation in order to deduce (F, G), which follows the technique described in [23]. For the Fast Fourier transform and the resulting code for arithmetic in rings, we basically reuse the implementations of Falcon and Mitaka.

A performance comparison with Falcon-512 is provided in this section. Compilation is carried out with `gcc` 11.4.0 with `-O3 -march=native` optimizations enabled. Timings are collected on a single core of an Intel Core i5-1340P @ 1.90 GHz desktop machine. Cycle counts are not provided for Falcon, since the Falcon benchmarking tool only measures clock time.

The comparison results are shown in Table 3. The running time of our key generation is slightly slower than Falcon. However, we stress that we are comparing our trapdoor generation algorithm that reaches a much smaller α (e.g., 1.04) with the original Falcon trapdoor sampler that reaches a quality $\alpha = 1.17$. As previously mentioned, the number of iterations of Gibbs sampler is 2500 and $\epsilon = 0.005$, then we only focus on the comparison of time to generate f and g.

Table 3: Performance comparison with FALCON-512

	Quality α	Tailcut rate τ	Classical Security	`keygen_fg` speed (Mcycles)	`keygen_fg` speed (ms)
FALCON-512	1.17	1.1	120	—	0.80
This paper	1.04	1.04	128	369.7	116.0

6.1 Comparisons of `keygen_fg` number of repetitions and running time between FALCON and this paper

In this section, we set $N = 512$, and for α ranging from 1.17 to 1.08, the number of repetitions and running time of `keygen_fg` are based on experimental observations. However, as α decreases, both the number of repetitions and running time increase rapidly. Consequently, the data for α between 1.08 and 1.04 are estimated values.

The detailed comparisons of `keygen_fg` number of repetitions and running time between FALCON and this paper are given in Figure 3 and Figure 4 respectively.

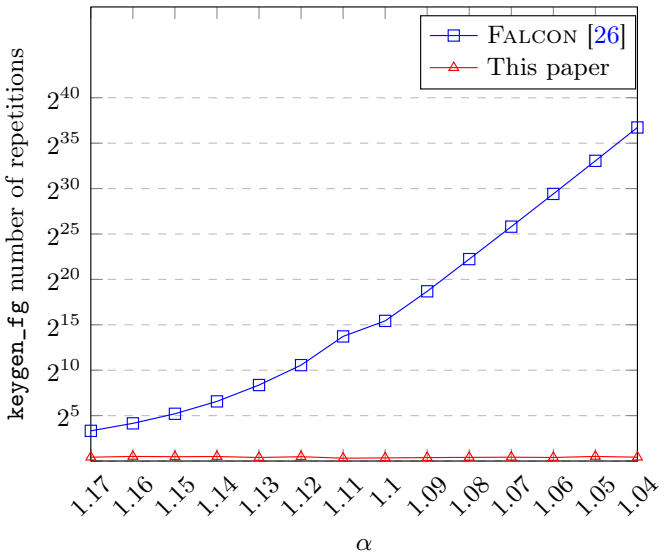


Fig. 3: Comparison of `keygen_fg` number of repetitions between FALCON and This paper.

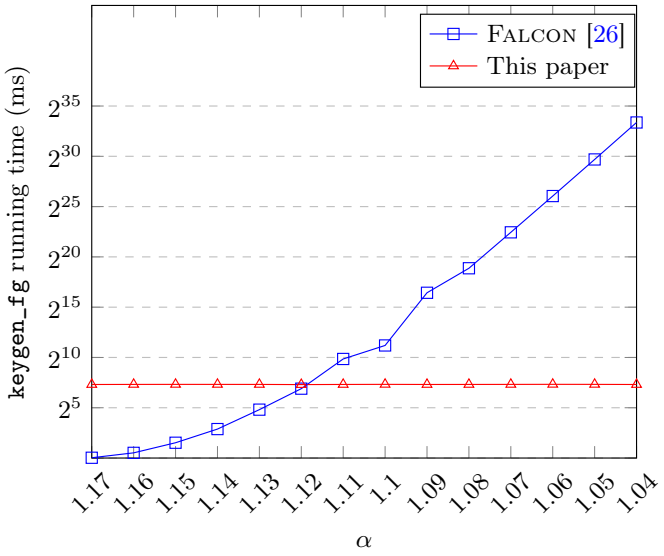


Fig. 4: Comparison of `keygen_fg` running time between Falcon and This paper.

Acknowledgements

We thank the anonymous reviewers of PQCrypto 2026 for their careful reviews and constructive feedback. We appreciate Léo Ducas, Yang Yu and Yu Yu for valuable comments and suggestions, Masayuki Abe for reviewing an early version of this paper, and Ruosi Wan for insightful discussions on MCMC algorithms.

References

1. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange - A new hope. In: Holz, T., Savage, S. (eds.) USENIX Security 2016. pp. 327–343. USENIX Association (Aug 2016)
2. Babai, L.: On Lovasz' lattice reduction and the nearest lattice point problem. Combinatorica **6**(1), 1–13 (1986)
3. Chen, Y., Genise, N., Mukherjee, P.: Approximate trapdoors for lattices and smaller hash-and-sign signatures. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019, Part III. LNCS, vol. 11923, pp. 3–32. Springer, Cham (Dec 2019).
4. Ducas, L., Lyubashevsky, V., Prest, T.: Efficient identity-based encryption over NTRU lattices. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014, Part II. LNCS, vol. 8874, pp. 22–41. Springer, Berlin, Heidelberg (Dec 2014).
5. Ducas, L., Nguyen, P.Q.: Learning a zonotope and more: Cryptanalysis of NTRUSign countermeasures. In: Wang, X., Sako, K. (eds.) ASIACRYPT 2012. LNCS, vol. 7658, pp. 433–450. Springer, Berlin, Heidelberg (Dec 2012).
6. Ducas, L., Prest, T.: Fast Fourier Orthogonalization. In: ISSAC 2016. pp. 191–198 (2016)

7. Espitau, T., Fouque, P.A., Gérard, F., Rossi, M., Takahashi, A., Tibouchi, M., Wallet, A., Yu, Y.: Mitaka: A simpler, parallelizable, maskable variant of falcon. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part III. LNCS, vol. 13277, pp. 222–253. Springer, Cham (May / Jun 2022).
8. Espitau, T., Nguyen, T.T.Q., Sun, C., Tibouchi, M., Wallet, A.: Antrag: Annular NTRU trapdoor generation - making mitaka as secure as falcon. In: Guo, J., Steinfeld, R. (eds.) ASIACRYPT 2023, Part VII. LNCS, vol. 14444, pp. 3–36. Springer, Singapore (Dec 2023).
9. Fouque, P.A., Kirchner, P., Pornin, T., Yu, Y.: BAT: Small and fast KEM over NTRU lattices. IACR TCHES **2022**(2), 240–265 (2022).
10. Fouque, P.A., Kirchner, P., Tibouchi, M., Wallet, A., Yu, Y.: Key recovery from Gram-Schmidt norm leakage in hash-and-sign signatures over NTRU lattices. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part III. LNCS, vol. 12107, pp. 34–63. Springer, Cham (May 2020).
11. Gamerman, D., Lopes, H.F.: Markov Chain Monte Carlo: stochastic simulation for Bayesian inference. CRC press (2006)
12. Geman, S., Geman, D.: Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. IEEE Transactions on pattern analysis and machine intelligence (6), 721–741 (1984)
13. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008).
14. Gentry, C., Szydlo, M.: Cryptanalysis of the revised NTRU signature scheme. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 299–320. Springer, Berlin, Heidelberg (Apr / May 2002).
15. Geweke, J.: Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments. In: Bayesian Statistics 4: Proceedings of the Fourth Valencia International Meeting, Dedicated to the memory of Morris H. DeGroot, 1931–1989. Oxford University Press (08 1992), https://doi.org/10.1093/oso/9780198522669.003.0010
16. Goldreich, O., Goldwasser, S., Halevi, S.: Public-key cryptosystems from lattice reduction problems. In: Kaliski, Jr., B.S. (ed.) CRYPTO'97. LNCS, vol. 1294, pp. 112–131. Springer, Berlin, Heidelberg (Aug 1997).
17. Guerreau, M., Martinelli, A., Ricosset, T., Rossi, M.: The hidden parallelepiped is back again: Power analysis attacks on falcon. IACR TCHES **2022**(3), 141–164 (2022).
18. Heidelberger, P., Welch, P.D.: Simulation run length control in the presence of an initial transient. Oper. Res. **31**(6), 1109–1144 (Dec 1983), https://doi.org/10.1287/opre.31.6.1109
19. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Silverman, J.H., Whyte, W.: NTRUSIGN: Digital signatures using the NTRU lattice. In: Joye, M. (ed.) CT-RSA 2003. LNCS, vol. 2612, pp. 122–140. Springer, Berlin, Heidelberg (Apr 2003).
20. Lin, X., Suzuki, M., Zhang, S., Espitau, T., Yu, Y., Tibouchi, M., Abe, M.: Cryptanalysis of the Peregrine lattice-based signature scheme. In: Tang, Q., Teague, V. (eds.) PKC 2024, Part I. LNCS, vol. 14601, pp. 387–412. Springer, Cham (Apr 2024).
21. Micciancio, D., Walter, M.: Practical, predictable lattice basis reduction. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part I. LNCS, vol. 9665, pp. 820–849. Springer, Berlin, Heidelberg (May 2016).

22. Nguyen, P.Q., Regev, O.: Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 271–288. Springer, Berlin, Heidelberg (May / Jun 2006).
23. Pornin, T., Prest, T.: More efficient algorithms for the NTRU key generation using the field norm. In: Lin, D., Sako, K. (eds.) PKC 2019, Part II. LNCS, vol. 11443, pp. 504–533. Springer, Cham (Apr 2019).
24. Prest, T.: Sharper bounds in lattice-based cryptography using the Rényi divergence. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part I. LNCS, vol. 10624, pp. 347–374. Springer, Cham (Dec 2017).
25. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: Falcon: Submission to the NIST's post-quantum cryptography standardization process, https://csrc.nist.gov/Projects/post-quantum-cryptography/round-3-submissions
26. Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: FALCON. Tech. rep., National Institute of Standards and Technology (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022
27. Smith, B.J.: BOA: An R package for MCMC output convergence assessment and posterior inference. Journal of Statistical Software **21**(11), 1–37 (2007). , https://www.jstatsoft.org/index.php/jss/article/view/v021i11
28. Stehlé, D., Steinfeld, R.: Making NTRU as secure as worst-case problems over ideal lattices. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 27–47. Springer, Berlin, Heidelberg (May 2011).
29. Wang, Z.: Markov chain monte carlo methods for lattice gaussian sampling: Convergence analysis and enhancement. IEEE Trans. Commun. **67**(10), 6711–6724 (2019)
30. Wang, Z., Ling, C.: Lattice gaussian sampling by markov chain monte carlo: Bounded distance decoding and trapdoor sampling. IEEE Trans. Inf. Theory **65**(6), 3630–3645 (2019)
31. Yu, Y., Ducas, L.: Learning strikes again: The case of the DRS signature scheme. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part II. LNCS, vol. 11273, pp. 525–543. Springer, Cham (Dec 2018).
32. Zhang, S., Jia, H., Ran, D., Yu, Y., Yu, Y., Wang, X.: GPV preimage sampling with weak smoothness and its applications to lattice signatures. In: Hanaoka, G., Yang, B. (eds.) ASIACRYPT 2025, Part III. LNCS, vol. 16247, pp. 233–264. Springer, Cham (Dec 2025)
33. Zhang, S., Lin, X., Yu, Y., Wang, W.: Improved power analysis attacks on falcon. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part IV. LNCS, vol. 14007, pp. 565–595. Springer, Cham (Apr 2023).

A Marcum Q-function

In statistics, the generalized Marcum Q-function of order ν is defined as:

$$Q_\nu(a,b) = \frac{1}{a^{\nu-1}} \int_b^\infty x^\nu \exp\left(-\frac{x^2+a^2}{2}\right) I_{\nu-1}(ax)\,\mathrm{d}x$$

where $b \geqslant 0$ and $a, \nu > 0$ and $I_{\nu-1}$ is the modified Bessel function of first kind of order $\nu - 1$. If $b > 0$, the integral converges for any ν. Using the fact that $Q_\nu(a,0) = 1$, the generalized Marcum Q-function can alternatively be defined as a finite integral as:

$$Q_\nu(a,b) = 1 - \frac{1}{a^{\nu-1}} \int_0^b x^\nu \exp\left(-\frac{x^2+a^2}{2}\right) I_{\nu-1}(ax)\,\mathrm{d}x$$

However, it is preferable to have an integral representation of the Marcum Q-function such that (i) the limits of the integral are independent of the arguments of the function, (ii) and that the limits are finite, (iii) and that the integrand is a Gaussian function of these arguments. For positive integer values of $\nu = n$, such a representation is given by the trigonometric integral:

$$Q_n(a,b) = \begin{cases} H_n(a,b) & a < b, \\ \frac{1}{2} + H_n(a,a) & a = b, \\ 1 + H_n(a,b) & a > b, \end{cases}$$

where

$$H_n(a,b) = \frac{\zeta^{1-n}}{2\pi} \exp\left(-\frac{a^2+b^2}{2}\right) \int_0^{2\pi} \frac{\cos(n-1)\theta - \zeta\cos n\theta}{1 - 2\zeta\cos\theta + \zeta^2} \exp(ab\cos\theta)\,\mathrm{d}\theta,$$

and the ratio $\zeta = a/b$ is a constant.

B The derivation of conditions (3) and (4)

If we represent f and g in their FFT form, i.e., $\big(\varphi_1(f), ..., \varphi_d(f)\big)$ and $\big(\varphi_1(g), ..., \varphi_d(g)\big)$, these denote the canonical embeddings of f and g, respectively. Starting from condition (1), i.e.,

$$|(f,g)|^2 \leq \alpha^2 q,$$

we substitute f and g to their canonical embeddings, then get:

$$|(\varphi_1(f), ..., \varphi_d(f), \varphi_1(g), ..., \varphi_d(g))|^2 \leq \alpha^2 q.$$

Then, we expand the left-hand side of the inequality step by step,

$$\begin{aligned} &|(\varphi_1(f), ..., \varphi_d(f), \varphi_1(g), ..., \varphi_d(g))|^2 \\ =&\frac{1}{\deg(\phi)} \cdot \big(\varphi_1(f)\varphi_1(\overline{f}) + ... + \varphi_d(f)\varphi_d(\overline{f}) + \varphi_1(g)\varphi_1(\overline{g}) + ... + \varphi_d(g)\varphi_d(\overline{g})\big) \\ =&\frac{1}{d} \cdot \big(|\varphi_1(f)|^2 + ... + |\varphi_d(f)|^2 + |\varphi_1(g)|^2 + ... + |\varphi_d(g)|^2\big)\,. \end{aligned}$$

Upon expanding, we derive the following inequality:

$$\frac{1}{d} \cdot \left(|\varphi_1(f)|^2 + ... + |\varphi_d(f)|^2 + |\varphi_1(g)|^2 + ... + |\varphi_d(g)|^2\right) \leq \alpha^2 q.$$

After a simple rearrangement, we obtain condition (3).

Starting from condition (2), i.e.,

$$\left|\left(\frac{q\overline{f}}{f*\overline{f}+g*\overline{g}}, \frac{q\overline{g}}{f*\overline{f}+g*\overline{g}}\right)\right|^2 \leq \alpha^2 q,$$

we substitute f and g to their canonical embeddings, then get:

$$\left|\left(\frac{q\cdot\varphi_1(\overline{f})}{\varphi_1(f*\overline{f}+g*\overline{g})}, ..., \frac{q\cdot\varphi_d(\overline{f})}{\varphi_d(f*\overline{f}+g*\overline{g})}, \right.$$
$$\left. \frac{q\cdot\varphi_1(\overline{g})}{\varphi_1(f*\overline{f}+g*\overline{g})}, ..., \frac{q\cdot\varphi_d(\overline{g})}{\varphi_d(f*\overline{f}+g*\overline{g})}\right)\right|^2 \leq \alpha^2 q.$$

Then, we expand the left-hand side of the inequality step by step,

$$\left|\left(\frac{q\cdot\varphi_1(\overline{f})}{\varphi_1(f*\overline{f}+g*\overline{g})}, ..., \frac{q\cdot\varphi_d(\overline{f})}{\varphi_d(f*\overline{f}+g*\overline{g})}, \right.\right.$$
$$\left.\left. \frac{q\cdot\varphi_1(\overline{g})}{\varphi_1(f*\overline{f}+g*\overline{g})}, ..., \frac{q\cdot\varphi_d(\overline{g})}{\varphi_d(f*\overline{f}+g*\overline{g})}\right)\right|^2$$

$$= \left|\left(\frac{q\cdot\varphi_1(\overline{f})}{|\varphi_1(f)|^2+|\varphi_1(g)|^2}, ..., \frac{q\cdot\varphi_d(\overline{f})}{|\varphi_d(f)|^2+|\varphi_d(g)|^2}, \right.\right.$$
$$\left.\left. \frac{q\cdot\varphi_1(\overline{g})}{|\varphi_1(f)|^2+|\varphi_1(g)|^2}, ..., \frac{q\cdot\varphi_d(\overline{g})}{|\varphi_d(f)|^2+|\varphi_d(g)|^2}\right)\right|^2$$

$$= \frac{1}{\deg(\phi)} \cdot \left(\frac{q^2\cdot|\varphi_1(f)|^2}{\left(|\varphi_1(f)|^2+|\varphi_1(g)|^2\right)^2} + ... + \frac{q^2\cdot|\varphi_d(f)|^2}{\left(|\varphi_d(f)|^2+|\varphi_d(g)|^2\right)^2} \right.$$
$$\left. + \frac{q^2\cdot|\varphi_1(g)|^2}{\left(|\varphi_1(f)|^2+|\varphi_1(g)|^2\right)^2} + ... + \frac{q^2\cdot|\varphi_d(g)|^2}{\left(|\varphi_d(f)|^2+|\varphi_d(g)|^2\right)^2}\right)$$

$$= \frac{1}{d} \cdot \left(\frac{q^2\cdot\left(|\varphi_1(f)|^2+|\varphi_1(g)|^2\right)}{\left(|\varphi_1(f)|^2+|\varphi_1(g)|^2\right)^2} + ... + \frac{q^2\cdot\left(|\varphi_d(f)|^2+|\varphi_d(g)|^2\right)}{\left(|\varphi_d(f)|^2+|\varphi_d(g)|^2\right)^2}\right)$$

$$= \frac{1}{d} \cdot \left(\frac{q^2}{|\varphi_1(f)|^2+|\varphi_1(g)|^2} + ... + \frac{q^2}{|\varphi_d(f)|^2+|\varphi_d(g)|^2}\right)$$

Upon expanding, we derive the following inequality:

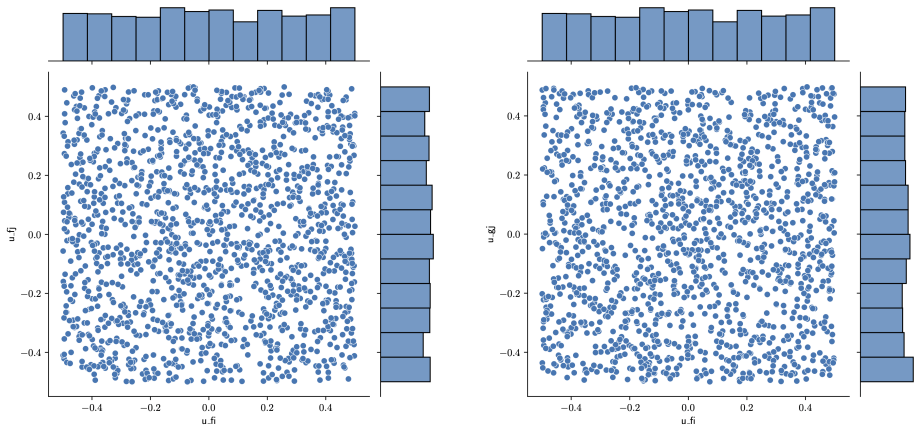
$$\frac{1}{d} \cdot \left(\frac{q^2}{|\varphi_1(f)|^2+|\varphi_1(g)|^2} + ... + \frac{q^2}{|\varphi_d(f)|^2+|\varphi_d(g)|^2}\right) \leq \alpha^2 q,$$

After a simple rearrangement, we obtain condition (4).

C Experimental data

This supplementary material collects data aimed at justifying the heuristics of Section 3.4, in the form of the following figures.

Figure 5 shows that the embeddings of the error e_f and e_g do behave as independent and uniform in $[-1/2, 1/2)$. We additionally confirm this graphical observation of the uniformity and independence using statistical χ^2 tests. Multiple experiments of goodness of fit (with bin size 20 and 1500 samples each) and independence (with bin size 400 and 1500 samples each) yield expectedly random-looking p-values, consistently above 0.05.



(a) $(e_f^{(21)}, e_f^{(50)})$ (b) $(e_f^{(101)}, e_g^{(3)})$

Fig. 5: Empirical joint distributions of two randomly coefficients of e_f (resp. a randomly chosen coefficient of e_f and another of e_g). The data is collected from 1500 samples (f, g) of degree $d = 512$.