MITAKA: A Simpler, Parallelizable, Maskable Variant of FALCON

Thomas Espitau[1], Pierre-Alain Fouque[3], François Gérard[6], Mélissa Rossi[5], Akira Takahashi[2], Mehdi Tibouchi[1], Alexandre Wallet[3], Yang Yu[4]

[1] NTT Corporation, Japan
{thomas.espitau.ax,mehdi.tibouchi.br}@hco.ntt.co.jp
[2] Aarhus University, Denmark
takahashi@cs.au.dk
[3] IRISA, Univ Rennes 1, Inria, Rennes Bretagne-Atlantique Center, France
pa.fouque@gmail.com, alexandre.wallet@inria.fr
[4] Tsinghua University, China
yang.yu0986@gmail.com
[5] ANSSI, France
melissa.rossi@ssi.gouv.fr
[6] University of Luxembourg
francois.gerard@uni.lu

Abstract. This work describes the MITAKA signature scheme: a new hash-and-sign signature scheme over NTRU lattices which can be seen as a variant of NIST finalist FALCON. It achieves comparable efficiency but is considerably simpler, online/offline, and easier to parallelize and protect against side-channels, thus offering significant advantages from an implementation standpoint. It is also much more versatile in terms of parameter selection.

We obtain this signature scheme by replacing the FFO lattice Gaussian sampler in FALCON by the "hybrid" sampler of Ducas and Prest, for which we carry out a detailed and corrected security analysis. In principle, such a change can result in a substantial security loss, but we show that this loss can be largely mitigated using new techniques in key generation that allow us to construct much higher quality lattice trapdoors for the hybrid sampler relatively cheaply. This new approach can also be instantiated on a wide variety of base fields, in contrast with FALCON's restriction to power-of-two cyclotomics.

We also introduce a new lattice Gaussian sampler with the same quality and efficiency, but which is moreover compatible with the integral matrix Gram root technique of Ducas et al., allowing us to avoid floating point arithmetic. This makes it possible to realize the *same* signature scheme as MITAKA efficiently on platforms with poor support for floating point numbers.

Finally, we describe a provably secure masking of MITAKA. More precisely, we introduce novel gadgets that allow provable masking at any order at much lower cost than previous masking techniques for Gaussian sampling-based signature schemes, for cheap and dependable side-channel protection.

1 Introduction

The third round finalists for signatures in the NIST postquantum standardization process consist of just three candidates: Rainbow [10], a multivariate scheme, Dilithium [13,33], a lattice-based scheme in the Fiat–Shamir with aborts framework, and FALCON [44], a hash-and-sign signature over NTRU lattices. They occupy fairly different positions within the design space of post-quantum signature schemes, and it is therefore important to understand, for each of them, to what extent they could possibly be improved by exploring similar designs that overcome some of their limitations. This paper aims at doing so for the FALCON signature scheme.

Hash-and-sign lattice-based signatures. FALCON fits within the long and hectic history of hash-and-sign signatures based on lattices. In those schemes, the signing key is a "good" representation of a lattice, the *trapdoor*, which makes it possible, given an arbitrary point in the ambient space, to find lattice points that are relatively close to it (i.e. solve the *approximate closest vector* problem, ApproxCVP[7]); the verification key, on the other hand, is a "bad" representation: it allows anyone to check if a point is in the lattice, but not to solve ApproxCVP. In order to sign a message, it is then hashed to a random point in the ambient space, and the signature is a lattice point close to it, obtained using the trapdoor. To verify, one checks that the signature is in the lattice and sufficiently close to the hash digest.

Early constructions along those lines, such as the GGH signature scheme [21] and multiple iterations of NTRUSign [23,22], were later shown to be insecure due to a common critical vulnerability: the lattice points obtained as signatures would leak information about the trapdoor used to compute them, which could then be recovered using more or less advanced statistical techniques [38,15]. One of the first round NIST candidates was in fact broken using the same idea [46].

It is thus crucial for security to prove that signatures are sampled according to a distribution that is *statistically independent* of the trapdoor. The first approach to do so, which remains the state of the art,[8] is due to Gentry, Peikert and Vaikuntanathan (GPV) [19]: sample the ApproxCVP solution according to a discrete Gaussian distribution centered at the target point and supported over the lattice, with covariance independent from the trapdoor (usually spherical). This type of lattice discrete Gaussian sampling can be carried out by randomizing known deterministic algorithms for ApproxCVP, like Babai rounding and Babai's nearest plane algorithm.

Within the overall GPV framework, specific signature schemes vary according to the lattices over which they are instantiated, the construction of the corresponding trapdoors, and the lattice Gaussian sampling algorithms they rely on based on those trapdoors. The security level achieved by such a scheme is then essentially determined by the *quality* of the trapdoor and of the Gaussian sampling algorithm, defined as the minimal standard deviation achievable in Gaussian sampling, while still preserving the statistical independence of the output.

A complete overview of existing proposals for each of those elements is beyond the scope of the current work. We focus instead on the particular case of NTRU lattices with the usual NTRU trapdoors first considered in NTRUSign, as those lattices appear to offer the most efficient implementations by a significant margin, thanks to their compact trapdoors.

Hash-and-sign signatures over NTRU lattices. NTRU lattices are, in essence, free rank 2 module lattices over cyclotomic rings, and the NTRU designers showed how to construct good trapdoors for them, even though the original signature schemes based on them proved insecure.

They were brought within the GPV framework (and thus gained provable security) thanks to the work of Ducas, Lyubashevsky and Prest (DLP) [14], who combined them with the Gaussian sampling algorithm obtained by randomizing Babai's nearest plane algorithm (this randomization is sometimes called the *Klein sampler* for lattice Gaussians). They analyzed the security of the construction and provided what became the first reasonably efficient implementation of a signature scheme in the GPV framework.

[7] Sometimes, this is also seen as a *bounded distance decoding* problem, BDD, but with large enough decoding bound that there are exponentially many solutions, instead of a unique one as is typically the case in the traditional formulation of BDD.

[8] Other techniques have been proposed that avoid Gaussian distributions, as in [34], but they tend not to be competitive.

This DLP signature scheme offers relatively compact keys and signatures, but suffers from a relatively long signing time, quadratic in the $\mathbb{Z}$-rank of the underlying lattice. This is because the nearest plane computation is carried out after descending to $\mathbb{Z}$, essentially ignoring the module structure of the lattice.

FALCON is a direct improvement of this scheme, obtained by replacing this quadratic Gaussian sampling by a quasilinear one, derived from the quasilinear nearest plane algorithm described in the Fast Fourier Orthogonalization paper of Ducas and Prest [16] (and refining the parameter selection using a tighter statistical analysis based on the Rényi divergence). The computation still ultimately descends to $\mathbb{Z}$, but takes advantage of the tower field structure of the underlying number field (assumed to be a power-of-two cyclotomic) to achieve a better complexity.

These two approaches are equivalent in terms of the quality of the resulting Gaussian sampler, which is essentially the best possible for the given NTRU lattice. However, DLP does so at the cost of a slow signing algorithm, whereas FALCON, while fast, suffers from a very complex signing algorithm that is hard to implement, poorly suited for parallelization and difficult to protect against side-channel attacks. On the last point, both schemes have been shown to suffer from potential vulnerabilities with respect to side-channel leakage [18,27], and even though the most recent implementation of FALCON appears to be protected against timing attacks [40,24], countermeasures against stronger side-channel attacks like DPA seem difficult to achieve. FALCON is also limited to NTRU lattices over power-of-two cyclotomic fields, which limits its flexibility in terms of parameter selection. That latter limitation can be overcome to some extent by extending the construction to higher rank modules, as done in MODFALCON [8], but the other drawbacks remain.

Another possibility is to instantiate the randomized ApproxCVP algorithm directly over the underlying ring, instead of doing so over $\mathbb{Z}$. For the randomized version of Babai rounding, this gives rise to (the ring version of) Peikert's sampler, as introduced in [39]. This can also be done for Babai's nearest plane algorithm, leading to what Ducas and Prest call the *hybrid sampler*. The resulting algorithms consist of a constant number of ring multiplications, so that quasilinear complexity is obtained "for free" as long as the underlying ring has a fast multiplication algorithm (as certainly holds for arbitrary cyclotomics). This makes them highly versatile in terms of parameter selection. They are also much simpler than FALCON, easy to parallelize, and support fairly inexpensive masking for side-channel protection.

Their downside, however, is the lower quality of the corresponding samplers compared to FALCON and DLP. Indeed, by not descending to $\mathbb{Z}$ but only to the ring itself, the ApproxCVP algorithm achieves less tight of a bound compared to the Klein sampler, and hence the Gaussian sampling has a larger standard deviation. This is analyzed in details in Prest's Ph.D. thesis [42] (although certain heuristic assumptions are incorrect), and results in a substantially lower security level than FALCON and DLP.

Our contributions: the MITAKA *signature scheme.* In this work, we revisit in particular the hybrid sampler mentioned above, and show that the security loss compared to FALCON can be largely mitigated using a novel technique to generate higher quality trapdoors. The resulting scheme, MITAKA,[9] offers an attractive alternative to FALCON in many settings since:

[9] Trivia: Mitaka is a neighborhood in Tokyo, Japan whose name means "the three falcons". It sounded fitting considering the maskable, parallelizable nature of our scheme and its strong points compared to FALCON.

- it is considerably simpler from an algorithmic and an implementation standpoint, while keeping the same complexity (in fact, it is likely faster at the same dimension due to better cache locality);
- signature generation is parallel(izable);
- like Peikert's sampler, it has an online/offline structure, with the online part requiring only one-dimensional discrete Gaussian sampling with very small, constant standard deviation and simple linear operations;
- it can be instantiated over arbitrary cyclotomic fields[10], which makes it quite versatile in terms of parameter selection;
- it is easier to protect against side-channels and can be cheaply masked even at high order.

The main idea that allows us to achieve higher security than previously expected is as follows. It is well-known that, given NTRU generators (f, g), it is easy to compute the quality of the corresponding NTRU trapdoor for the hybrid sampler (in particular, it can be done without computing the whole trapdoor). It is thus very easy to check whether a given (f, g) reaches a certain threshold in terms of bit security, and as a result, the costly part of key generation is the sampling of the random ring elements f and g themselves (with discrete Gaussian coefficients). One can therefore achieve a greatly improved security level at the same cost in terms of randomness and not much more computation time if one can "recycle" already sampled ring elements f and g.

We propose several ways of doing so. The simplest one is to generate lists $\{f_i\}$, $\{g_j\}$ of candidate elements for f and g, and test the pairs (f_i, g_j): this increases the space of candidates quadratically, instead of linearly, in the amount of generated randomness. One can also generate the f_i's, g_j's themselves as sums of Gaussians with smaller standard deviation (as long as it remains above the smoothing parameters), and consider the Galois conjugates of a given g_j. By combining these techniques appropriately, we achieve a substantial security increase, of around 15 bits for typical parameter sizes. Concretely, we achieve the same security level as Dilithium–II [13] (which was argued to reach NIST Level-I) in dimension $d = 512$, and attain roughly NIST Level–V in dimension $d = 1024$, with intermediate parameter settings possible.

We also provide a detailed security analysis of our construction, and while most of the presentation focuses on power-of-two cyclotomics for simplicity's sake and easier comparison with previous work, we also show that intermediate NIST security levels can be conveniently achieved using other base fields, e.g. of dimension 648 (same security as FALCON–512), 768 (NIST Level–II), 864 (NIST Level–III) and 972 (NIST Level–IV).

As an additional contribution, we also introduce a novel, alternate lattice Gaussian sampler for MITAKA that achieves the same complexity and the same quality as the hybrid sampler, but has a different structure, closer to Peikert's sampler. The advantage of that alternate sampler is that it is compatible with the integral lattice Gram root technique of Ducas et al. [12], making it possible to instantiate it *without floating point arithmetic*. We call the resulting construction MITAKA$_{\mathbb{Z}}$. We stress that MITAKA and MITAKA$_{\mathbb{Z}}$ are two different approaches to implement the *same* signature scheme (in the sense that the generated signatures have statistically close distributions), and one can choose one or the other as preferred depending on whether the target platform has fast floating point arithmetic or not.

Finally, we introduce a new, concrete approach to mask those signature generation algorithms efficiently. In previous work, efficiently masking signature schemes using Gaussian sampling has proved quite challenging: even for the case of 1-dimensional *centered* discrete Gaussians, as in the

[10] In principle, even more general number fields are possible as well, provided a good basis is known for their canonical embedding. The corresponding security analysis is cumbersome, however.

BLISS signature scheme [11], this is far from straightforward [4]. Since MITAKA and MITAKA$_{\mathbb{Z}}$, like FALCON and DLP, require discrete Gaussian sampling with *variable* centers, a naive approach to masking is unlikely to yield fast results. Instead, we introduce and prove a novel gadget for sampling Gaussian distribution with an arithmetically masked center and a fixed standard deviation. This allows us to completely avoid masking Gaussian sampling operations in the online phase:[11] this works for a masked center, because picking a uniform center in $[0, M)$ with fixed denominator and sampling a discrete Gaussian around that center results in a close to uniform distribution modulo M. Carrying out this share-by-share sampling directly causes a slight decrease in the quality of the resulting sampler (depending on the number of shares), but this can be overcome completely with careful use of rejection sampling. Combining these statistical techniques with usual provable masking methodology, we achieve very efficient side-channel protection for both MITAKA and MITAKA$_{\mathbb{Z}}$.

2 Preliminaries

For any $a \in \mathbb{R}$ and $q > 0$, let $[a]_q = \lfloor aq \rceil / q \in (1/q)\mathbb{Z}$.

2.1 Linear algebra and lattices

Write $\mathbf{A}^t$ for the transpose of any matrix $\mathbf{A}$. Let $s_1(\mathbf{A}) = \max_{\mathbf{x}\neq\mathbf{0}} \frac{\|\mathbf{Ax}\|}{\|\mathbf{x}\|}$ the largest singular value of $\mathbf{A}$. Let $\Sigma \in \mathbb{R}^{n\times n}$ be a symmetric matrix. We write $\Sigma \succ 0$ when Σ is *positive definite*, i.e. $\mathbf{x}^t\Sigma\mathbf{x} > 0$ for all non-zero $\mathbf{x} \in \mathbb{R}^n$. We also write $\Sigma_1 \succ \Sigma_2$ when $\Sigma_1 - \Sigma_2 \succ 0$. It holds that $\Sigma \succ 0$ if and only if $\Sigma^{-1} \succ 0$ and that $\Sigma_1 \succ \Sigma_2 \succ 0$ if and only if $\Sigma_2^{-1} \succ \Sigma_1^{-1} \succ 0$. A lattice $\mathscr{L}$ is a discrete additive subgroup of a Euclidean space. When the space is $\mathbb{R}^m$, and if it is generated by (the columns of) $\mathbf{B} \in \mathbb{R}^{m\times d}$, we also write $\mathscr{L}(\mathbf{B}) = \{\mathbf{Bx} \mid \mathbf{x} \in \mathbb{Z}^d\}$. If $\mathbf{B}$ has full column rank, then we call $\mathbf{B}$ a basis and d the rank of $\mathscr{L}$. The volume of $\mathscr{L}$ is $\mathrm{Vol}(\mathscr{L}) = \det(\mathbf{B}^t\mathbf{B})^{\frac{1}{2}}$ for any basis $\mathbf{B}$.

2.2 Power-of-two cyclotomic fields

For the sake of simplicity and readability, we focus in the rest of this article on the case where the number field is a cyclotomic field of conductor a power of 2. In any case, the content of this section generalizes straightforwardly to other cyclotomic number fields, as well as most of our results. Besides, the use of cyclotomic fields is nowadays pervasive in lattice-based cryptography. In this section we therefore keep only the minimum amount of notation and definitions to follow the article. More details can be found in Appendix A.

Let $d = 2^\ell$ for some integer $\ell \geqslant 1$ and ζ_d to be a $2d$-th primitive root of 1. Then for a fixed d, $\mathscr{K} := \mathbb{Q}(\zeta_d)$ is the d-th power-of-two cyclotomic field, and its ring of algebraic integers is $\mathscr{R} := \mathbb{Z}[\zeta_d]$. The field automorphism $\zeta_d \mapsto \zeta_d^{-1} = \overline{\zeta_d}$ corresponds to the complex conjugation, and we write the image f^* of f under this automorphism. We have $\mathscr{K} \simeq \mathbb{Q}[x]/(x^d + 1)$ and $\mathscr{R} \simeq \mathbb{Z}[x]/(x^d + 1)$, and both are contained in $\mathscr{K}_\mathbb{R} := \mathscr{K} \otimes \mathbb{R} \simeq \mathbb{R}[x]/(x^d + 1)$. Each $f = \sum_{i=0}^{d-1} f_i\zeta_d^i \in \mathscr{K}_\mathbb{R}$ can be identified[12] with its coefficient vector $(f_0, \ldots, f_{d-1}) \in \mathbb{R}^d$. The adjoint operation extends naturally to $\mathscr{K}_\mathbb{R}$, and $\mathscr{K}_\mathbb{R}^+$ is the subspace of elements satisfying $f^* = f$.

[11] The same idea can be adapted to the offline phase by masking the zero center. This is a bit less compelling, however, as it requires more shares, and replaces centered Gaussian sampling by variable center sampling.

[12] This is the so-called coefficient embedding.

The cyclotomic field $\mathscr{K}$ comes with d complex field embeddings $\varphi_i : \mathscr{K} \to \mathbb{C}$ which map f seen as a polynomial to its evaluations at the odd powers of ζ_d. This defines the so-called *canonical embedding* $\varphi(f) := (\varphi_1(f), \ldots, \varphi_d(f))$. It extends straightforwardly to $\mathscr{K}_\mathbb{R}$ and identifies it to the space $\mathcal{H} = \{\mathbf{v} \in \mathbb{C}^d \ : \ v_i = \overline{v_{d/2+i}}, 1 \leqslant i \leqslant d/2\}$. Note that $\varphi(fg) = (\varphi_i(f)\varphi_i(g))_{i \leqslant d}$. When needed, this embedding extends entry-wise to vectors or matrices over $\mathscr{K}_\mathbb{R}$. We let $\mathscr{K}_\mathbb{R}^{++}$ be the subset of $\mathscr{K}_\mathbb{R}^{+}$ which have all positive coordinates in the canonical embedding.

2.3 Matrices of algebraic numbers and NTRU modules

2.3.1 2×2 $\mathscr{K}$-valued matrices. This work deals with free $\mathscr{R}$-modules of rank 2 in $\mathscr{K}^2$, or in other words, groups of the form $\mathscr{R}\mathbf{x} + \mathscr{R}\mathbf{y}$ where $\mathbf{x} = (x_1, x_2), \mathbf{y} = (y_1, y_2)$ span $\mathscr{K}^2$. There is a natural $\mathscr{K}$-bilinear form over $\mathscr{K}^2$ defined by $\langle \mathbf{x}, \mathbf{y} \rangle_\mathscr{K} := x_1^* y_1 + x_2^* y_2 \in \mathscr{K}$. It can be checked that for all $\mathbf{x} \in \mathscr{K}^2$, $\langle \mathbf{x}, \mathbf{x} \rangle_\mathscr{K} \in \mathscr{K}_\mathbb{R}^{++}$. This form comes with a corresponding notion of orthogonality. In particular, the well-known Gram-Schmidt orthogonalization procedure for a pair of linearly independent vectors $\mathbf{b}_1, \mathbf{b}_2 \in \mathscr{K}^2$ is defined as

$$\widetilde{\mathbf{b}}_1 := \mathbf{b}_1, \ \widetilde{\mathbf{b}}_2 := \mathbf{b}_2 - \frac{\langle \mathbf{b}_1, \mathbf{b}_2 \rangle_\mathscr{K}}{\langle \mathbf{b}_1, \mathbf{b}_1 \rangle_\mathscr{K}} \cdot \widetilde{\mathbf{b}}_1.$$

One readily checks that $\langle \widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_2 \rangle_\mathscr{K} = 0$. The Gram-Schmidt matrix with columns $\widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_2$ is denoted by $\widetilde{\mathbf{B}}$ and we have $\det \widetilde{\mathbf{B}} = \det \mathbf{B}$.

For $\Sigma \in \mathscr{K}_\mathbb{R}^{2\times 2}$, we write Σ^* its conjugate-transpose, where $*$ is the conjugation in $\mathscr{K}_\mathbb{R}$. We extend the notion of positive definiteness for matrices with entries in $\mathscr{K}_\mathbb{R}$: $\Sigma \in \mathscr{K}_\mathbb{R}^{2\times 2}$ is positive definite when $\Sigma = \Sigma^*$ and all the d matrices induced by the field embeddings are positive definite. We then write $\Sigma \succ 0$. For example, $\mathbf{B}^*\mathbf{B}$ is a positive definite matrix for all $\mathbf{B} \in \mathscr{K}_\mathbb{R}^{2\times 2}$. Positive definite matrices admit "square roots", that is, matrices $\sqrt{\Sigma}$ such that $\sqrt{\Sigma}\sqrt{\Sigma}^* = \Sigma$.

This work uses fundamental quantities for matrices over $\mathscr{K}$. The first is defined as $|\mathbf{B}|_\mathscr{K} := \max_{1 \leqslant i \leqslant 2} \|\varphi(\langle \widetilde{\mathbf{b}}_i, \widetilde{\mathbf{b}}_i \rangle_\mathscr{K})\|_\infty^{1/2}$. Since the eigenvalues λ_1, λ_2 of $\mathbf{B}^*\mathbf{B}$ are all in $\mathscr{K}^{++}$, coordinate-wise square roots are well-defined. The largest singular value of (the embeddings of) $\mathbf{B}$ is recovered as $s_1(\mathbf{B}) := \max_{1 \leqslant i \leqslant 2} \|\varphi(\sqrt{\lambda_i})\|_\infty$.

NTRU Modules. Given $f, g \in \mathscr{R}$ such that f is invertible modulo some prime $q \in \mathbb{Z}$, we let $h = f^{-1}g \bmod q$. The NTRU module determined by h is $\mathscr{L}_{\text{NTRU}} = \{(u, v) \in \mathscr{R}^2 \ : \ uh - v = 0 \bmod q\}$. Two bases of this free module are of particular interest for cryptography:

$$\mathbf{B}_h = \begin{bmatrix} 1 & 0 \\ h & q \end{bmatrix} \text{ and } \mathbf{B}_{f,g} = \begin{bmatrix} f & F \\ g & G \end{bmatrix},$$

where $F, G \in \mathscr{R}$ are such that $fG - gF = q$. This module is usually seen as a lattice of volume q^d in $\mathbb{R}^{2d}$ thanks to the coefficient embedding. From the explicit description of $\mathbf{B}_{f,g}$, one can derive formulas for the associated quality parameters. These are proven in Appendix A.

Lemma 1 ([42], adapted). *Let $\mathbf{B}_{f,g}$ be a basis of an NTRU module. We have $\sqrt{q} \leqslant |\mathbf{B}_{f,g}|_\mathscr{K} \leqslant s_1(\mathbf{B}_{f,g})$ and :*

$$|\mathbf{B}_{f,g}|_\mathscr{K}^2 = \max\left(\|\varphi(ff^* + gg^*)\|_\infty, \left\|\frac{q^2}{\varphi(ff^* + gg^*)}\right\|_\infty\right),$$
$$s_1(\mathbf{B}_{f,g})^2 = \frac{1}{2}\|\varphi\left(T + \sqrt{T^2 - 4q^2}\right)\|_\infty,$$

where $T := ff^* + gg^* + FF^* + GG^*$. *We have* $|\mathbf{B}_{f,g}|_{\mathscr{K}} = s_1(\widetilde{\mathbf{B}})$, *where* $\widetilde{\mathbf{B}}$ *is the Gram-Schmidt orthogonalization (over* $\mathscr{K}$*) of* $\mathbf{B}_{f,g}$.

2.4 Gaussians over rings

The Gaussian function on $\mathbb{R}^d$ centered at $\mathbf{c}$ and with covariance matrix $\Sigma \succ 0$ is defined as $\rho_{\mathbf{c},\Sigma}(\mathbf{x}) = \exp(-\frac{1}{2}(\mathbf{x}-\mathbf{c})^t\Sigma^{-1}(\mathbf{x}-\mathbf{c}))$. If $\Sigma = s^2\mathbf{I}_d$, we write also $\rho_{\mathbf{c},s} = \exp(-\|\mathbf{x}-\mathbf{c}\|^2/(2s^2))$ and call the associated Gaussian *spherical*. We omit $\mathbf{c}$ if it is $\mathbf{0}$. The normal distribution $\mathcal{N}_\Sigma$ of covariance Σ then has density probability function $((2\pi)^d \cdot \det\Sigma)^{-1/2}\rho_\Sigma$. When we write $\mathcal{N}_{\mathscr{K}_\mathbb{R},s}$, we mean that $(z_1,\ldots,z_d) \leftarrow (\mathcal{N}_{s/\sqrt{d}})^d$ is sampled and $(z_1 + iz_2, \ldots, z_{d-1} + iz_d)$ is outputted.

The discrete Gaussian distribution over a full rank lattice $\mathscr{L}$, centered at $\mathbf{c}$ and with covariance matrix $\Sigma \succ 0$ has density function given by

$$\forall\, \mathbf{x} \in \mathscr{L}, D_{\mathscr{L},\mathbf{c},\Sigma}(\mathbf{x}) = \frac{\rho_{\mathbf{c},\Sigma}(\mathbf{x})}{\rho_{\mathbf{c},\Sigma}(\mathscr{L})}.$$

For $c \in \mathscr{K}_\mathbb{R}$ and $s > 0$, we also use the notation $\lfloor c \rceil_s$ to denote the distribution $D_{\varphi(\mathscr{R}),\varphi(c),s}$. It extends coordinate-wise to vectors in $\mathscr{K}_\mathbb{R}^2$. For $\varepsilon > 0$, the smoothing parameter of a lattice $\mathscr{L}$ is $\eta_\varepsilon(\mathscr{L}) = \min\{s > 0 : \rho_{1/s}(\mathscr{L}^\vee) \leqslant 1+\varepsilon\}$, where $\mathscr{L}^\vee$ is the dual lattice. The exact definition of the lattice dual is not needed in this work, and when $\mathscr{L} = \mathscr{L}(\mathbf{B}) \subset \mathbb{R}^d$, it is enough to know the matrix $\mathbf{B}^{-t}$ encodes it. We say that $\sqrt{\Sigma} \geqslant \eta_\varepsilon(\mathscr{L})$ when $\rho_1(\sqrt{\Sigma}^*\mathscr{L}^\vee) = \rho_{\Sigma^{-1}}(\mathscr{L}^\vee) \leqslant 1+\varepsilon$. In particular, one checks that $r\mathbf{B} \succ \eta_\varepsilon(\varphi(\mathbf{B}\mathscr{R}^2))$ when $r \geqslant \eta_\varepsilon(\mathscr{R}^2)$. We use the following bound.

Lemma 2 (Adapted from [19]). *Let* $\mathbf{B}\mathscr{R}^2$ *be free* $\mathscr{R}$*-module, and let* $\mathscr{L} = \mathrm{M}(\mathbf{B})\mathbb{Z}^{2d}$ *be the associated rank* d *lattice in* $\mathbb{R}^{2d}$. *For all* $\varepsilon > 0$,

$$\eta_\varepsilon(\mathscr{L}) \leqslant |\mathbf{B}|_{\mathscr{K}} \cdot \frac{1}{\pi}\sqrt{\frac{\log(4d(1+1/\varepsilon))}{2}}.$$

3 Sampling discrete Gaussians in $\mathscr{R}$-modules

We present three approaches to sample discrete Gaussian over rings. The first two are respectively Peikert's perturbative approach adapted from [39], and the hybrid sampler of Ducas and Prest [42], which is core to MITAKA and uses the first as a subroutine. Then we describe a new sampler based on [12] which can involve integer arithmetic only and combines the ideas of the other two others. In Appendix H, we also explicitly specify a set of operations that can be precomputed during the offline phase, as well as where FFT or NTT should be performed in practice.

3.1 Peikert's sampler

In [39], Peikert presented an efficient algorithm to sample discrete Gaussians in a target lattice, using small continuous Gaussian perturbation. On a high level, it can be thought of as a randomized version of Babai's round-off algorithm, using random (normal) perturbations to hide the lattice structure, and can be formulated directly over the algebra $\mathscr{K}_\mathbb{R}$. The pseudo-code in Algorithm 1 outputs discrete Gaussians in a free rank 2 $\mathscr{R}$-module $\mathscr{L}$ described by a basis $\mathbf{B} \in \mathscr{K}^{2\times 2}$, with an arbitrary center in $\mathscr{K}_\mathbb{R}^2$. When $\Sigma \succ r^2\mathbf{B}\mathbf{B}^*$, the existence of Σ_0 below is guaranteed.

Algorithm 1: RingPeikert sampler

Input: A matrix $\mathbf{B} \in \mathscr{K}^{2\times 2}$ such that $\mathscr{L} = \varphi(\mathbf{B}\mathscr{R}^2)$ and a target center $\mathbf{c} \in \mathscr{K}_{\mathbb{R}}^2$.
Result: $\mathbf{z} \in \mathscr{L}$ with distribution negligibly far from $D_{\mathscr{L},\mathbf{c},\Sigma}$.

1 *Precomputed: a parameter* $r \geqslant \eta_\varepsilon(\mathscr{R}^2)$*, and* $\Sigma_0 \in \mathscr{K}_{\mathbb{R}}^{2\times 2}$ *such that* $\Sigma_0\Sigma_0^* = \Sigma - r^2\mathbf{B}\mathbf{B}^*$
2 $\mathbf{x} \leftarrow \Sigma_0 \cdot (\mathcal{N}_{\mathscr{K}_{\mathbb{R}},1})^2$
3 $\mathbf{z} \leftarrow \lceil \mathbf{B}^{-1}(\mathbf{c} - \mathbf{x}) \rfloor_r$
4 **return** $\mathbf{B}\mathbf{z}$

Algorithm 2: RingPeikert, one-dimensional version

Input: A target center $c \in \mathscr{K}_{\mathbb{R}}$.
Result: $z \in \mathscr{R}$ with distribution negligibly far from $D_{\mathscr{R},c,\Sigma}$.

1 *Precomputed: a parameter* $r \geqslant \eta_\varepsilon(\mathscr{R})$*, and* $\sigma_0 \in \mathscr{K}_{\mathbb{R}}$ *such that* $\sigma_0^*\sigma_0 = \Sigma - r^2$
2 $x \leftarrow \sigma_0 \cdot \mathcal{N}_{\mathscr{K}_{\mathbb{R}},1}$
3 **return** $\lceil c - x \rfloor_r$

Theorem 1 ([39], adapted). *Let $\mathscr{D}$ be the output distribution of Algorithm 1. If $\varepsilon \leqslant 1/2$ and $\sqrt{\Sigma} \geqslant s_1(\mathbf{B}) \cdot \eta_\varepsilon(\mathscr{R}^2)$, then the statistical distance between $\mathscr{D}$ and $D_{\mathscr{L},\mathbf{c},\Sigma}$ is bounded by 2ε. Moreover, we have*

$$\sup_{\mathbf{x}\in\mathbf{B}\mathscr{R}^2} \left| \frac{\mathscr{D}(\mathbf{x})}{D_{\mathscr{L}(\mathbf{B}),\mathbf{c},\Sigma}(\mathbf{x})} - 1 \right| \leqslant 4\varepsilon.$$

Theorem 1 is reproved in Appendix E for the sake of completeness, where fundamental parameters for its implementation are also analyzed. From Lemma 1 and Lemma 2, note that the condition in the statement ensures that we are above the smoothing parameter of the target lattice. In practice, the covariance parameter is a scalar multiple of the identity matrix, or a positive real "constant" if seen in $\mathscr{K}_{\mathbb{R}}^{++}$. We highlight in Algorithm 2 the one-dimensional version of Peikert's sampler, that is, outputting discrete Gaussians in $\mathscr{R}$, because it appears as a subroutine of the hybrid sampler in the next section.

3.2 Ducas & Prest's hybrid sampler

In [42], a so-called hybrid sampler is presented that outputs discrete Gaussians in free $\mathscr{R}$ modules of finite rank. On a high level, this hybrid sampler follows Klein's approach, which is a randomized version of the Nearest Plane algorithm. In the ring context, the randomization subroutine happens "at the ring level" thanks to a ring Gaussian sampler, instead of "at the integer level". To again hide the lattice structure, perturbations are also involved but their distribution now depends on the target center. The hybrid sampler is described in Algorithm 3, which makes use of floating-point arithmetic, and is core to the MITAKA scheme.

It relies on a ring sampler which can be instantiated by Algorithm 2. For the sake of clarity, these "Peikert sampling" steps are made explicit in lines 4–6 and 9–11. We restrict to "totally spherical" standard deviation parameters (that is, scalar matrices) as they are the main use-case of this work.

The next result is proved in Appendix F, where we also analyze the necessary floating-point precision to preserve enough bits of security.

Algorithm 3: Hybrid Gaussian sampler

Input: A target center $\mathbf{c} \in \mathscr{K}_{\mathbb{R}}^2$, a matrix $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2]$ such that $\mathscr{L} = \varphi(\mathbf{B}\mathscr{R}^2)$ and its GSO $[\widetilde{\mathbf{b}_1}, \widetilde{\mathbf{b}_2}]$ over $\mathscr{K}$, a parameter $\sigma > 0$ (corresponding to $(\sigma, \ldots, \sigma) \in \mathscr{K}_{\mathbb{R}}$).
Result: $\mathbf{z}$ with distribution negligibly far from $D_{\mathscr{L},\mathbf{c},\sigma^2\mathbf{I}_{2d}}$.

Precomputed: $\sigma_i := \sqrt{\frac{\sigma^2}{\langle \widetilde{\mathbf{b}_i}, \widetilde{\mathbf{b}_i} \rangle} - r^2} \in \mathscr{K}_{\mathbb{R}}^{++}$.
$\mathbf{c}_2 \leftarrow \mathbf{c}, \mathbf{v}_2 \leftarrow 0$
$d_2 \leftarrow \frac{\langle \widetilde{\mathbf{b}_2}, \mathbf{c}_2 \rangle_{\mathscr{K}}}{\langle \widetilde{\mathbf{b}_2}, \widetilde{\mathbf{b}_2} \rangle_{\mathscr{K}}}$
$u_2 \leftarrow \mathcal{N}_{\mathscr{K}_{\mathbb{R}},1}$
$y_2 \leftarrow \sigma_2 \cdot u_2$
$x_2 \leftarrow \lfloor d_2 - y_2 \rceil_r$
$\mathbf{c}_1 \leftarrow \mathbf{c}_2 - x_2\mathbf{b}_2, \mathbf{v}_1 \leftarrow x_2\mathbf{b}_2$
$d_1 \leftarrow \frac{\langle \widetilde{\mathbf{b}_1}, \mathbf{c}_1 \rangle_{\mathscr{K}}}{\langle \widetilde{\mathbf{b}_1}, \widetilde{\mathbf{b}_1} \rangle_{\mathscr{K}}}$
$u_1 \leftarrow \mathcal{N}_{\mathscr{K}_{\mathbb{R}},1}$
$y_1 \leftarrow \sigma_1 \cdot u_1$
$x_1 \leftarrow \lfloor d_1 - y_1 \rceil_r$
$\mathbf{v}_0 \leftarrow \mathbf{v}_1 + x_1\mathbf{b}_1$
return $\mathbf{v}_0$

Algorithm 4: Hybrid Gaussian sampler, $\mathbf{U}$ version

Input: A target center $\mathbf{c} = (c_1, c_2) \in \mathscr{K}^2$, an upper triangular matrix $\mathbf{U} = [(1, 0), (u, 1)]$ with $u \in \mathscr{K}$, a parameter $r > 0$ (corresponding to $(r, \ldots, r) \in \mathscr{K}_{\mathbb{R}}$).
Result: $\mathbf{z}$ with distribution negligibly far from $D_{\mathscr{L}(\mathbf{U}),\mathbf{c},r}$.

$z_2 \leftarrow \mathsf{RingSampler}_{\mathbb{Z}}(c_2, r)$
$c_1' \leftarrow c_1 - z_2 u$
$z_1 \leftarrow \mathsf{RingSampler}_{\mathbb{Z}}(c_1', r)$
return $\mathbf{z} = \mathbf{U}(z_1, z_2)$.

Theorem 2 ([42], Theorem 5.10, adapted). *Let $\mathscr{D}$ be the output distribution of Algorithm 3. If $\varepsilon \leqslant 2^{-5}$ and $\sqrt{\Sigma} \geqslant |\mathbf{B}|_{\mathscr{K}} \cdot \eta_\varepsilon(\mathscr{R}^2)$, then the statistical distance between $\mathscr{D}$ and $D_{\mathscr{L},\mathbf{c},\Sigma}$ is bounded by 7ε. Moreover, we have*

$$\sup_{\mathbf{x} \in \mathbf{B}\mathscr{R}^2} \left| \frac{\mathscr{D}(\mathbf{x})}{D_{\mathscr{L},\mathbf{c},\Sigma}(\mathbf{x})} - 1 \right| \leqslant 14\varepsilon.$$

In our integer arithmetic friendly sampler presented in the next section, we rely on a specific variant where the target lattice is described by an upper triangular matrix $\mathbf{U}$, or equivalently, when the Gram-Schmidt orthogonalization is the identity matrix. It is presented in Algorithm 4, and is core to MITAKA$_{\mathbb{Z}}$. In particular, in MITAKA$_{\mathbb{Z}}$ the ring sampler becomes a *discrete* Gaussian sampler that can be emulated in integer arithmetic. This is emphasized below by $\mathsf{RingSampler}_{\mathbb{Z}}$.

As a simpler version of Algorithm 3, Algorithm 4 deviates less from the discrete Gaussian it emulates; the proof can be found in Appendix F.

3.3 An integer arithmetic friendly sampler

To clarify the presentation, in this section we identify matrices over $\mathscr{K}$ to their structured version over $\mathbb{Q}$. Fundamentally, our new sampler for $D_{\mathscr{L}(\mathbf{B}),\mathbf{c},s}$ combines Peikert's approach of Algorithm 1 and hybrid sampling in the case where the Gram-Schmidt is the identity. What allows us

Algorithm 5: Integer arithmetic ring Gaussian sampler

Input: a matrix $\widehat{\mathbf{B}} \in \mathscr{R}^{2\times 2}$ such that $\widehat{\mathbf{B}}\mathbf{U}_{\hat{u}} = \mathbf{B} = \widetilde{\mathbf{B}}\mathbf{U}_u$, where $\hat{u} = [u]_p \in \frac{1}{p}\mathscr{R}$, a center $\mathbf{c} \in \mathscr{R}^2$, and parameters $r, s > 0$.
Result: $\mathbf{z}$ with distribution negligibly far from $D_{\mathscr{L}(\mathbf{B}),\mathbf{c},rs}$.

Precomputed: $\Sigma_p = s^2\mathbf{I} - \widehat{\mathbf{B}}\widehat{\mathbf{B}}^t$ and $\mathbf{A} \leftarrow$ IntGram$(p^2(\Sigma_p - \mathbf{I}))$ /* $\mathbf{A}\mathbf{A}^t = p^2(\Sigma_p - \mathbf{I})$ */
$\mathbf{p} \leftarrow$ Algorithm 6$(p, \mathbf{A})$ /* $\mathbf{p} \sim D_{\mathscr{R}^2, r^2\Sigma_p}$ */
$\hat{\mathbf{c}} \leftarrow \widehat{\mathbf{B}}^{-1}(\mathbf{c} - \mathbf{p})$
$\mathbf{z}' \leftarrow$ Algorithm 4$(\hat{u}, \hat{\mathbf{c}}, s)$ /* $\mathbf{z}' \sim D_{\mathscr{L}(\mathbf{U}_{\hat{u}}),\hat{\mathbf{c}},s}$ */
return $\mathbf{z} = \widehat{\mathbf{B}}\mathbf{z}'$

Algorithm 6: Offline sampler

Input: An integer $p > 0$, a matrix $\mathbf{A} \in \mathscr{R}^{2\times m}$.
Result: $\mathbf{p} \in \mathscr{R}^2$ with distribution negligibly far from $D_{\mathscr{R}^2, r^2\Sigma}$, where $\Sigma = \frac{1}{p^2}\mathbf{A}\mathbf{A}^t + \mathbf{I}$.

Precomputed: integers $r > \eta_\varepsilon(\mathscr{R}^2)$ and L such that $Lr \geqslant \eta_\varepsilon(\Lambda(A)^\perp)$.
$\mathbf{x} \leftarrow (\lfloor 0 \rceil_{Lr})^m$
$\mathbf{p}' \leftarrow \frac{1}{pL}\mathbf{A}\mathbf{x}$
$\mathbf{p} \leftarrow \lfloor \mathbf{p}' \rceil_r$
return $\mathbf{p}$.

to restrict to integer arithmetic is to rely on the work of [12]. There, the authors showed how to generate small *integral* perturbation vectors, relying on a generalization of the Cholesky decomposition that can be also computed purely in integer arithmetic.

On the "hybrid side", as observed in the previous section, it is enough for us to have access to a discrete Gaussian sampler in integer arithmetic. Multiplying the output of Algorithm 4 by the Gram-Schmidt orthogonalization $\widetilde{\mathbf{B}} = \mathbf{B}\mathbf{U}^{-1}$ of the target lattice basis, one would obtain vector with the correct support. The Gram-Schmidt basis may however contain entries in $\mathscr{K}$ that may have very large denominators. We avoid this thanks to an approximation $\widehat{\mathbf{B}} \in (1/(pq))\mathscr{R}^{2\times 2}$ of $\widetilde{\mathbf{B}}$, obtained by p-rounding of the upper right coefficient of $\mathbf{U}$. The quality of this approach is essentially driven by $|\mathbf{B}_{f,g}|_{\mathscr{K}} = s_1(\widetilde{\mathbf{B}})$.

Algorithm 5 describes this approach. The notation $\mathbf{U}_{\hat{u}}$ denotes that the upper-right coefficient of the matrix is $\hat{u}$. The procedure IntGram is fully described in [12], and impacts the choice of parameters for the algorithm to be actually correct. The determination of its parameters is discussed in Appendix F.3. On a high level, given in input a positive definite matrix $\Sigma \in \mathscr{R}^{2\times 2}$, it outputs a matrix $\mathbf{A} \in \mathscr{R}^{2\times m}$ such that $\mathbf{A}\mathbf{A}^t = \Sigma$, and where $m \geqslant 2$. In our context, the input is a small perturbation covariance matrix $\Sigma_p = s^2\mathbf{I} - \widehat{\mathbf{B}}\widehat{\mathbf{B}}^t$, where s is a large enough integer.

The offline sampler in Algorithm 6 is adapted from [12] and outputs from the expected distribution as long as $\mathbf{A}$ has been suitably computed. In terms of notation, recall that $\Lambda(\mathbf{A})^\perp \subset \mathscr{R}^m$ is the lattice of integer solutions of $\mathbf{A}\mathbf{x} = \mathbf{0}$. Its analysis is given in Appendix F.3.

We now state the correctness of Algorithm 5, stressing that the statement is correct as long as the integral root decomposition could be carried out. In particular, the proof assumes at least that $p \geqslant d$, and can be found in Appendix F.3.

Theorem 3. *Keep the notation of Algorithm 5, assuming also that* IntGram *correctly computes* $\mathbf{A}$. *For* $\varepsilon \in (0,1)$, *let* $s > |\mathbf{B}_{f,g}|_{\mathscr{K}}(1 + \sqrt{2d/p}) + 1$ *be an integer and* $r \geqslant \eta_\varepsilon(\mathbb{Z}^{2d})$. *Then the*

distribution $\mathscr{D}$ of the output of Algorithm 5 is at statistical distance at most 15ε from $D_{\mathscr{L}(\mathbf{B}),\mathbf{c},sr}$. Moreover, we have

$$\sup_{\mathbf{z}\in\mathscr{L}(\mathbf{B})}\left|\frac{\mathscr{D}(\mathbf{z})}{D_{\mathscr{L}(\mathbf{B}),\mathbf{c},sr}(\mathbf{z})}-1\right|\leqslant 30\varepsilon.$$

3.4 Asymptotic security of the samplers

Hash-and-sign signatures over lattices are constructed, following the GPV framework [19], by hashing a message to the ambient space of the lattice, and returning as a signature a lattice point close to that hash digest. This is done using a "good" representation of the lattice, called the *trapdoor*, that enables the signer to solve the ApproxCVP problem with a relatively small approximation factor. Moreover, to prevent signatures from leaking information about the secret trapdoor, the close lattice points need to be sampled according to a distribution that is statistically independent of the trapdoor: usually a spherical discrete Gaussian distribution supported over the lattice and centered at the hash digest. This is where the algorithms from the previous sections come into play.

The security of the resulting signature scheme depends on the standard deviation of the discrete Gaussian distribution output by the sampler: the smaller the standard deviation, the closer the distance to the hash digest, the harder the corresponding ApproxCVP problem, and hence the higher the security level. As we have seen, however, there is a lower bound (depending on the trapdoor) to how small of a standard deviation the sampler can achieve while remaining statistically close to the desired spherical Gaussian: lower than that, and the distribution may start to deviate from that Gaussians in ways that could expose information about the secret trapdoor, and thus compromise the security of the signing key.

In the case of NTRU lattices, the trapdoor is the secret basis:

$$\mathbf{B}_{f,g}=\begin{bmatrix}f & F\\ g & G\end{bmatrix},$$

and the standard deviation of the discrete Gaussian obtained from this trapdoor varies depending on the sampling algorithm, as discussed in particular in [42, §6]. It can be written as:

$$\sigma=\alpha\cdot\eta_\varepsilon(\mathscr{R}^2)\cdot\sqrt{q} \tag{1}$$

where the factor $\alpha\geqslant 1$, which we call the *quality*, depends on the sampler for a given trapdoor.

For the so-called Klein sampler used in DLP and FALCON, $\alpha\sqrt{q}$ is the Gram–Schmidt norm $\|\mathbf{B}_{f,g}\|_{\mathrm{GS}}:=\max_{1\leqslant i\leqslant 2d}\|\widetilde{\mathbf{b}}_i^{\mathbb{Z}}\|_2$ of $\mathbf{B}_{f,g}$ *over the integers*. For the Peikert sampler over $\mathscr{K}$, Theorem 1 shows that $\alpha\sqrt{q}=s_1(\mathbf{B}_{f,g})$. Finally, for the hybrid sampler, Theorem 2 shows that $\alpha\sqrt{q}=|\mathbf{B}_{f,g}|_{\mathscr{K}}$.

For a given sampler, the generators f,g should be sampled appropriately to minimize the corresponding α. In his thesis [42], Prest analyzed the optimal choices both theoretically (under suitable heuristics) and experimentally. The resulting optimal choices for α are as follows (after correcting the flawed heuristic analysis of Prest in the case of the hybrid sampler: see our discussion in Appendix C):

- heuristically, the quality of the Peikert sampler satisfies $\alpha=O(d^{1/4}\sqrt{\log d})$ [42, §6.5.2];
- for the hybrid sampler, following Appendix C, we show $\alpha=O(d^{1/8}\log^{1/4}d)$ (and not $O(\sqrt{\log d})$ contrary to what was claimed in [42, §6.5.2] based on flawed heuristics);

Table 1. Comparison of the best achievable trapdoor quality α for the various Gaussian samplers over NTRU lattices.

Sampler	$\alpha\sqrt{q}$	Best achievable α
Peikert	$s_1(\mathbf{B}_{f,g})$	$O(d^{1/4}\sqrt{\log d})$ [42, §6.5.2]
Hybrid (MITAKA)	$\lvert\mathbf{B}_{f,g}\rvert_{\mathscr{K}}$	$O(d^{1/8}\log^{1/4} d)$ [Appendix C]
Klein (FALCON)	$\lVert\mathbf{B}_{f,g}\rVert_{\text{GS}}$	$O(1)$ [42, §6.5.1]

– for the Klein sampler (used in DLP, and in modified form, FALCON), the heuristic analysis in [42, §6.5.1] show that it can be taken as low as $\sqrt{e/2} \approx 1.17$ independently of the dimension, and in particular $\alpha = O(1)$.

These properties are summarized in Table 1.

3.5 The MITAKA signature scheme

The previous samplers can be plugged directly into the GPV framework [19] to construct secure hash-and-sign signature schemes in the random oracle model. The idea is to sign a message by first hashing it as a point in the ambient space of the lattice, and then using the sampler to construct a lattice point close to that target. The signature is then the difference between the target and the lattice point (a small vector in the lattice coset defined by the target). This is described more precisely in Algorithm 7. Both MITAKA and MITAKA$_{\mathbb{Z}}$ are specific instantiations of this paradigm, using the samplers of Algorithms 3 and 5 respectively.

In Algorithm 7, the acceptance bound γ for signatures is chosen slightly larger than $\sigma\sqrt{d}$, for σ the standard deviation of the sampler given by Eq. (1) above, in order to ensure a low repetition rate for signature generation. (In the concrete security evaluation of Section 5, γ is selected so as to ensure $< 10\%$ rejection; this gives e.g. $\gamma = 1.042\sigma\sqrt{2d}$ for $d = 512$). Signature verification simply recovers the second component $s_2 = s_1 \cdot h + c \bmod q$ and checks that the vector $\mathbf{s} = (s_1, s_2)$ is of length at most γ.

The security argument of Gentry, Peikert, and Vaikuntanathan reduces the security of the signature scheme to the hardness of SIS in the underlying lattice up to bound 2γ. It is therefore invalidated if an attacker can obtain two distinct outputs of the sampler with the same center (since their difference would be a solution to this SIS problem) [19, Section 6.1]. This is avoided in the signature scheme by randomizing the hash value associated with the message using a sufficiently long random salt $\mathsf{r} \in \{0,1\}^k$. To avoid collisions, it suffices to pick $k \geqslant \lambda + \log_2 q_s$ for λ bits of security and q_s signature queries. The choice of $k = 320$ as in [44,8] suffices for up to 256 bits of security.

4 Improved Trapdoor Generation

The Peikert, hybrid and FALCON samplers for an NTRU basis $\mathbf{B}_{f,g}$ all have essentially the same complexity, and the first two are significantly simpler, easier to implement, slightly faster in the same dimension, and offer better avenues for parallelization and side-channel resistance (see Section 7). It would therefore be desirable to adopt one of the first two for practical implementations.

However, as seen in Section 3.4, the FALCON sampler has a substantial advantage in terms of security, since its Gaussian standard deviation is proportional to $\lVert\mathbf{B}_{f,g}\rVert_{\text{GS}}$, whereas the Peikert and hybrid samplers are proportional to $s_1(\mathbf{B}_{f,g})$ and $\lvert\mathbf{B}_{f,g}\rvert_{\mathscr{K}}$ respectively, which are both larger.

Algorithm 7: Signature scheme

Input: A message m, a secret key sk, a bound γ.
Result: A signature sig of m.

do
  $\mathsf{r} \xleftarrow{\$} \{0,1\}^k$
  $\mathbf{c} \leftarrow H(\mathsf{r} \| \mathsf{m})$
  $\mathbf{z} \leftarrow \mathsf{Sampler}(\mathsf{sk}, (0, c))$ /* Algorithm 3 or 5 */
  $\mathbf{s} \leftarrow (s_1, s_2) = (0, c) - \mathbf{z}$ /* $s_1 \cdot h - s_2 \equiv -c \bmod q$ */
while $\|\mathbf{s}\|^2 > \gamma^2$
return $\mathsf{sig} = (\mathsf{r}, s_1)$.

This results in a significant difference in asymptotic terms, as shown in Table 1, and also in bit security terms as will become apparent in the next section.

To increase the security level achievable using the first two samplers, and in particular the hybrid sampler, we propose a new technique to significantly improve the quality of NTRU trapdoors. We note in passing that it also applies to FALCON: while it cannot yield significant improvements in terms of security, since the standard deviation it achieves is already a very small factor away from the theoretical optimum, it can be used to speed up key generation substantially. The idea is as follows.

Recall that NTRU trapdoor generation for FALCON, say, works by sampling f, g with discrete Gaussian coefficient, computing the $\|\mathbf{B}_{f,g}\|_{\mathrm{GS}}$ of the resulting NTRU basis, and checking if this value is below the desired quality threshold. If not, we try again, and if so, the NTRU basis is completed and kept as the secret key. Trapdoor sampling for the hybrid sampler is similar. (On the other hand, for Peikert, completion has to be recomputed at each step to evaluate $s_1(\mathbf{B}_{f,g})$).

In this process, the costly operations are, on the one hand, the generation of the discrete Gaussian randomness, which has to be repeated several dozen times over in order to reach the desired threshold (this is not explicitly quantified by the authors of FALCON, but experiments suggest that, in many instances, upwards of 50 iterations are necessary), and, on the other hand, the completion of the basis (still costly despite recent optimizations [41]), which is only carried out once at the end and not for each iteration[13].

To optimize the process, our idea is to amortize the cost of discrete Gaussian sampling, by constructing several candidate trapdoors from the same randomness. We propose three main ideas to do so.

Lists of candidates for f and g. The usual key generation algorithm for FALCON, as already mentioned, normally ends up generating many pairs (f_i, g_i), and tests each of them as a candidate first vector for the NTRU lattice.

Since we are generating Gaussian vectors f_i and g_i anyway, we can easily recycle this generated randomness by testing all the mixed pairs (f_i, g_j) instead: this results in a set of possible candidates which increases quadratically with the number of random vectors we generate, instead of just linearly.

[13] This is the case at least for FALCON and for the hybrid sampler, as for both of them, one can compute the quality of the trapdoor given only (f, g). This is especially fast for the hybrid sampler. For the Peikert sampler, however, doing so without also obtaining (F, G) seems difficult, and is left as an open problem.

Generating the Gaussian vectors as linear combinations. Independently, one can generate each candidate vector f as a linear combination $\sum_{k=1}^{\ell} f^{(k)}$ where each $f^{(k)}$ is sampled from a discrete Gaussian of standard deviation $\sigma_0/\sqrt{\ell}$, for σ_0 the desired standard deviation of f. It is well-known that this results in the correct distribution provided that $\sigma_0/\sqrt{\ell}$ remains above the smoothing parameter of $\mathbb{Z}$ [37]. In fact, the FALCON implementation already does so for $d = 512$, where the candidate vectors are sums of two Gaussians vectors of standard deviation $\sqrt{2}$ times lower.

Now, when generating several f_i's, one obtains ℓ lists $L_k = \{f_i^{(k)}\}_i$ of Gaussian vectors. It is again possible to recycle this generated randomness by mixing and matching among those lists, and constructing candidates f of the form $\sum f_{i_k}^{(k)}$ for varying indices i_k, so that the total set of candidates is in bijection with $\prod_k L_k$. Its size increases like the ℓ-th power of the size of the lists.

Using the Galois action. Finally, one can expand the set of candidates for g, say, by applying the action of the Galois group. In principle, other unitary transformations of g, even less structured ones like randomly permuting the coefficients in the power basis, could also be considered, but the Galois action in particular is convenient as it is expressed as a circular permutation on the embeddings $\varphi_i(g)$ of g (i.e., the Fourier coefficients), and for the hybrid sampler, the computation of the quality is entirely carried out in the Fourier domain.

Concretely, recall from Lemma 1 that the quality parameter α of the hybrid sampler associated with $\mathbf{B}_{f,g}$ satisfies:

$$\alpha^2 = \frac{|\mathbf{B}_{f,g}|_{\mathscr{K}}^2}{q} = \max\left(\frac{\max_i z_i}{q}, \frac{q}{\min_i z_i}\right)$$
$$\text{where } z_i = \varphi_i(ff^* + gg^*) = |\varphi_i(f)|^2 + |\varphi_i(g)|^2 \in \mathbb{R}^+.$$

It is easy to compute the embeddings z_i^τ associated to $\mathbf{B}_{f,\tau(g)}$ for some Galois automorphism τ of $\mathscr{K}$ simply by applying the corresponding permutation on the components of $\varphi(g)$. Moreover, we see from this representation that the conjugation $\tau_*\colon g \mapsto g^*$ leaves this quality invariant, so the relevant Galois elements to consider are a set of representatives of $\mathrm{Gal}(\mathscr{K}/\mathbb{Q})/\langle\tau_*\rangle$. For power-of-two cyclotomics, one can for example use τ_5^k for $k = 0, \ldots, d/2-1$, where $\tau_5(\zeta_d) = \zeta_d^5$.

Security considerations. The techniques above can potentially skew the distribution of f and g somewhat compared to the case when each tested (f, g) that fails to pass the security threshold is thrown away. However, this is not really cause for concern: the precise distribution of f and g is not known to affect the security of the signature scheme other than in two ways:

- the extent to which it affects the geometry of the trapdoor, as encoded in the quality parameter α already; and
- the length of (f, g) itself as it affects key recovery security, but this length is always at least as large in our setting as in FALCON.

This indicates that our optimized secret keys do not weaken the scheme.

Concrete example. In Algorithm 8, we describe an example key generation procedure that combines all three techniques presented above: we construct lists of candidates for f and g and test all possible pairs. Moreover, each f and g itself is sampled as a sum of $\ell = 2$ narrower Gaussians, and the list of g's is expanded using the Galois action. Of course, different combinations of the techniques are also possible, but this particular one offers a good balance between efficiency and achievable security.

Algorithm 8: MITAKA optimized key generation

Input: Desired standard deviation σ_0 of f and g, target quality α of the Gaussian, number of samples m to generate, set G of Galois automorphisms to apply. The total search space is of size $\#G \cdot m^4$ for $4m$ generated discrete Gaussian vectors.
Result: NTRU first trapdoor vector (f, g) with quality better than α.

1 **for** $i \in [1, m]$ **do**
2 $\quad f'_i \leftarrow D_{\mathscr{R},\sigma_0/\sqrt{2}}$, $f''_i \leftarrow D_{\mathscr{R},\sigma_0/\sqrt{2}}$
3 $\quad g'_i \leftarrow D_{\mathscr{R},\sigma_0/\sqrt{2}}$, $g''_i \leftarrow D_{\mathscr{R},\sigma_0/\sqrt{2}}$
4 **end for**
5 $L_f \leftarrow \{f'_i + f''_j \mid i, j \in [1, m]\}$
6 $L_g \leftarrow \{\tau(g'_k + g''_\ell) \mid k, \ell \in [1, m], \tau \in G\}$
7 $L_u \leftarrow \{(f, \varphi(ff^*)) \mid f \in L_f\}$
8 $L_v \leftarrow \{(g, \varphi(gg^*)) \mid g \in L_g\}$
9 **for** $(f, u) \in L_u, (g, v) \in L_v$ **do**
10 $\quad z \leftarrow u + v$
11 $\quad$ **if** $q/\alpha^2 \leqslant z_i \leqslant \alpha^2 q$ *for all* i **then return** (f, g)
12 **end for**
13 **restart**

Using this approach, as shown in Fig. 1, we are able to efficiently generate trapdoors with $\alpha \leqslant 2.04$ for $d = 512$, and $\alpha \leqslant 2.33$ for $d = 1024$ by $m \approx 16$ (corresponding to generating 64 narrow Gaussian vectors to select one candidate (f, g), largely in line with FALCON).

Improved search via early aborts and filtering. Key generation using the technique above involves an exhaustive search in a relatively large set of candidates $L_u \times L_v$, and testing each candidate involves $O(d)$ comparisons:

$$q/\alpha^2 \leqslant u_i + v_i \leqslant \alpha^2 q \quad \text{for } 1 \leqslant i \leqslant d/2,$$

as done in Step 11 of Algorithm 8. One can of course reject a candidate immediately as soon as one of the comparison fails, but this can happen arbitrarily late in the loop through the indices.

However, it results from the analysis of Appendix C that the lower bound condition is much more likely to fail than the upper bound for a given candidate. Moreover, if we fix u, then it is more likely to fail on any given v for the indices i such that u_i is small. One can therefore

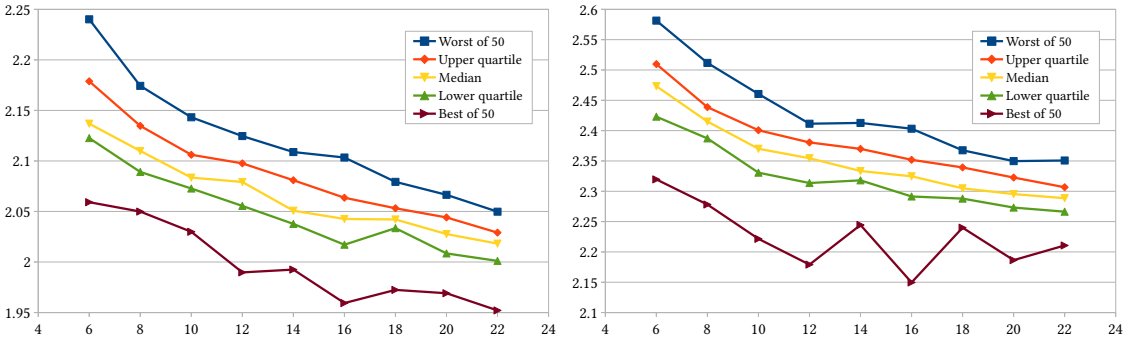


Fig. 1. Quality α reached by the optimized sampler of Algorithm 8 for various choices of m (50 trials each, $\sigma_0 = 1.17\sqrt{q/2d}$, G coset representatives of $\mathrm{Gal}(\mathscr{K}/\mathbb{Q})/\langle\tau_*\rangle$). Reachable α in dimension 512 (left) and 1024 (right).

improve the algorithm by a wide margin by carrying out a simple precomputation on u: extract the list of indices $S_u(w)$ of the w smallest elements of u for some $w \ll d/2$ (this can be done without sorting, in time $O(d)$). Then, for each corresponding candidate v, first check in time $O(w)$ whether the lower bound condition holds on the indices in $S_u(w)$: if so, the comparison is carried out normally, and otherwise v is rejected early.

Picking for example $w = 25$, we find that around 99.8% of candidates are rejected early in that way for our parameters, greatly reducing the number of full-fledged comparisons. All in all, this lets us achieve a speed-up of more than 5- to 10-fold as d ranges from 512 to 1024.

An additional, very simple optimization is to filter out values u, v such that $\|u\|_\infty > \alpha^2 q$ (and similarly for v) from the lists L_u and L_v, since such candidates clearly cannot satisfy the comparison.

5 Security analysis of MITAKA

Concrete security. In order to assess the concrete security of our signature scheme, we proceed using the usual cryptanalytic methodology of estimating the complexity of the best attacks against *key recovery attacks* on the one hand, and *signature forgery* on the other. The detailed analysis is carried out in Appendix G. For the parameter choices relevant to our scheme (in which the vectors of the trapdoor basis are not unusually small), key recovery is always harder than signature forgery, and therefore the cost of signature forgery is what determines the security level. Using the condition of Eq. (12), we see that the security of the forgery is a function of the standard deviation of the lattice Gaussian sampler used in the signature function, which itself depends on the quality α of the trapdoor, as discussed in Section 3.4. This analysis translates into concrete bit-security estimates following the methodology of NEWHOPE [1], sometimes called "core-SVP methodology". In this model [5,30], the bit complexity of lattice sieving (which is asymptotically the best SVP oracle) is taken as $\lfloor 0.292\beta \rfloor$ in the classical setting and $\lfloor 0.265\beta \rfloor$ in the quantum setting in dimension β. The resulting security in terms of α is given in Fig. 2 in dimensions 512 and 1024. This allows us to compare MITAKA with FALCON as well as with a "naive" version of the hybrid sampler that would omit the optimizations of Section 4; the results are presented in Table 2.

In addition, as mentioned earlier, our construction can be instantiated over more general base fields than power-of-two cyclotomics, which enables us to choose security level in a much more flexible way than FALCON. This is analyzed in Appendix B. Example security levels which can be reached in this way are presented in Table 3. For such fields, we can choose the modulus q to be the first prime which is congurant to 1 modulo the conductor.

Asymptotic security. As for all signature schemes in the GPV framework, the EUF–CMA security of our scheme in an asymptotic sense reduces, both in the classical [19] and quantum random oracle models [6], to the SIS problem in the underlying lattice (in this case, an instance of Module–SIS [31]). However, as is the case for FALCON (and as holds, similarly, for Dilithium), the SIS bound in Euclidean norm for the standard parameter choice ($q = 12289$) makes the underlying assumption vacuous. This is not known to lead to any attack, and can be addressed by increasing q if so desired, or reducing to the SIS problem in infinity norm instead.

Table 2. Concrete values for sampler quality and associated bit security level.

	$d = 512$				$d = 1024$			
	Quality α	Classical	Quantum	NIST Level	Quality α	Classical	Quantum	NIST Level
FALCON	1.17	124	112	I	1.17	285	258	V
Naive Hybrid[a]	3.03	90	82	below I	3.58	207	188	IV
MITAKA	2.04	**102**	**92**	I[b]	2.33	**233**	**211**	V

[a] Key generation with the same median amount of randomness as MITAKA Algorithm 8 with $m = 16$, but without the optimizations of Section 4.
[b] Taking into account the heavy memory cost of sieving. This is the same level as Dilithium–II; see [32, §5.3].

Table 3. Intermediate parameters and security levels for MITAKA.

	$d = 512$	$d = 648$	$d = 768$	$d = 864$	$d = 972$	$d = 1024$
Conductor	2^{10}	$2^3 \cdot 3^5$	$2^8 \cdot 3^2$	$2^5 \cdot 3^4$	$2^2 \cdot 3^6$	2^{11}
Security (C/Q)	102/92	136/123	167/151	192/174	220/199	233/211
NIST level	I[a]	I[b]	II	III	IV	V
Modulus q	12289	3889	18433	10369	17497	12289
Quality α	2.04	2.13	2.20	2.25	2.30	2.33
Sig. size (bytes)	713	827	1080	1176	1359	1405

[a] Slightly above Dilithium–II. [b] Above FALCON–512; arguably reaches level II.

6 Implementation Results

In order to assess the practicality of MITAKA, we carried out a preliminary, pure C implementation[14] of the scheme (using the sampler described in Algorithm 3). For easier and fairer comparison, we reused the polynomial arithmetic and FFT of the reference implementation of FALCON, as well as its pseudorandom generator (an implementation of ChaCha20).

An important caveat is that the current version of our code includes direct calls to floating point transcendental functions, and therefore cannot be guaranteed to run in constant time as is. It is well-known that this can be addressed using the polynomial approximation techniques used e.g. in [47,4,24], but full precision estimates for the required functions are left as future work.

[14] It will soon be available on a public repository.

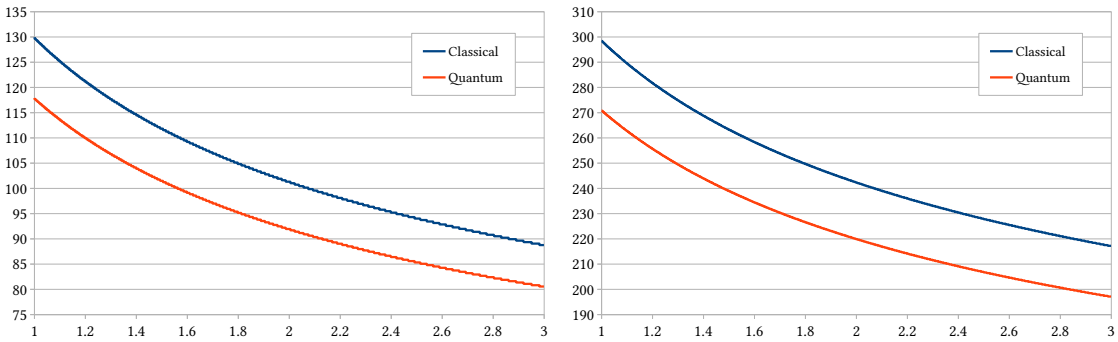


Fig. 2. Security (classical and quantum) against forgery as a function of the quality $1 \leqslant \alpha \leqslant 3$ of the lattice sampler (left: dimension 512 and right: dimension 1024).

Nevertheless, in our tests run on a single core of an Intel Core i7–1065G7 @ 1.30GHz laptop, this implementation is more than twice as fast as the reference implementation of FALCON: about 6300 signatures per second for our code using $d = 512$ (resp. 3100 signatures per second for $d = 1024$), compared to 2800 for FALCON–512 (resp. 1400 for FALCON–1024). We believe that these preliminary results are quite promising for MITAKA.

Furthermore, performance is mainly driven by the cost of the continuous and discrete 1D Gaussian samplers. Since signature generation can be split in an offline part and an online part (see Algorithm 13), MITAKA can offer even better speed results if some computations can be performed between signatures. While these results are favorable to MITAKA, optimized implementations on specific architectures would be needed for a definitive comparison to FALCON.

7 Side-channel Countermeasure

First, our signature scheme can be easily made isochronous. According to [24], isochrony ensures independence between the running time of the algorithm and the secret values. For our signature, the absence of conditional branches implies that one can implement our signature isochronously using standard techniques.

In a second step, we turn our signature scheme into an equivalent one which is protected against more powerful side-channel attacks that exploit the leakage of several executions. More precisely, following the seminal work due to Ishai, Sahai, and Wagner [26], we aim to protect our samplers for MITAKA and MITAKA$_{\mathbb{Z}}$ alternative described in Section 3 from the so-called *t-probing side-channel adversary*, who is able to peek at up to t intermediate variables per each invocation of an algorithm. The *masking* countermeasure is a technique to mitigate such attacks, by additively *secret-sharing* every sensitive variables into $t + 1$ values in $\mathcal{R}$. The integer t is often referred to as *masking order*. Essentially, we will provide two functionally equivalent alternative algorithms for MITAKA and MITAKA$_{\mathbb{Z}}$ where any set of at most t intermediate variables is independent from the secret. In this paper, we consider the masking order as a—potentially large—arbitrary variable t. Clearly, high masking order allows a side-channel protected implementation to tolerate stronger probing attacks with larger number of probes. For a ring element $a \in \mathcal{R}$, we say that a vector $(a_i)_{0 \leqslant i \leqslant t} \in \mathcal{R}^{t+1}$ is an arithmetic masking of a if $a = \sum_{i \in [0,t]} a_i$. For readability, we often write $[\![a]\!] := (a_i)_{0 \leqslant i \leqslant t}$.

The masking of our signature presents three unprecedented difficulties in masked lattice-based schemes.

1. Compared to Fiat-Shamir with aborts, masking the Gaussian sampling is unavoidable. We here present a novel technique to efficiently mask Gaussian sampling in Section 7.2.1. Notably, our approach only requires arithmetic masking, allowing us to avoid any conversion between arithmetic and Boolean shares during the online phase.
2. The computations are performed in $\mathbb{Z}$ instead of a modular ring. This feature does not appear in any other lattice-based scheme. Thus, we need to fix a bound on the size of the masks and make sure that the computations will never pass this bound. Let $\mathsf{Q}^{\mathrm{mask}}$ be the bound on the largest manipulated integer, the shares of $[\![a]\!]$ are implicitly reduced modulo $\mathsf{Q}^{\mathrm{mask}}$. Sometimes we refine the notation $[\![\cdot]\!]$ into $[\![\cdot]\!]_M$ to explicitly specify a modulus $M < \mathsf{Q}^{\mathrm{mask}}$ for secret-sharing.
3. Some polynomial multiplications need both inputs to be masked. This unusual operation does not appear in LWE-based schemes where the multiplications are performed between a public matrix of polynomial and a masked vector. We handle this problem with a function in Section 7.2.2.

7.1 Preliminaries on masking countermeasure

The most basic security notion for a masking countermeasure is the t-*privacy* of a gadget G [26]. This notion guarantees that any set of at most t intermediate variables observed during the computation is independent of the secret input. While the idea behind the notion is relatively simple, t-private gadgets are unfortunately not composable, meaning that a gadget consisting of multiple t-private sub-gadgets may not be necessarily secure. Hence in this work we rely on the following more handy security notions introduced by Barthe et al. [2].

Definition 1 (**t-NI, t-SNI**). *Let G be a gadget with inputs $(x_i)_{0\leqslant i\leqslant t} \in \mathcal{R}^{t+1}$ and outputs $(y_i)_{0\leqslant i\leqslant t} \in \mathcal{R}^{t+1}$. Suppose that for any set of t_1 intermediate variables and any subset of $O \subseteq [1,t]$ of output indices with $t_1 + |O| \leqslant t$, there exists a subset of indices $I \subseteq [1,t]$ such that the output distribution of the t_1 intermediate variables and the output variables $(y_i)_{i\in O}$ can be simulated from $(x_i)_{i\in I}$. Then*

1. *if $|I| \leqslant t_1 + |O|$ we say G is* t-non-interfering (t-NI)*, and*
2. *if $|I| \leqslant t_1$ we say G is* t-strong-non-interfering (t-SNI).

It is easy to check that t-SNI implies t-NI which implies t-probing security. The above notion can be naturally extended for a gadget with multiple input and output sharings. Note that *linear* operations performed share-wise (such as addition of two sharings, or multiplication by a constant) are trivially t-NI, as each computation on share i can be simulated from the input share x_i. Building blocks satisfying either NI or SNI can be easily composed with each other, by inserting the Refresh gadgets at suitable locations to re-randomize shares [2, Proposition 4]. It is also internally used in the Unmask gadget before taking the sum of shares, so that a probe on any partial sum doesn't leak more information than one input share [3].

Typically, the non-interference notions only deal with gadgets where all of the inputs and outputs are sensitive. To also handle public, non-sensitive values, a weaker notion called *NI with public output* (t-NIo) was proposed in [3]. As stated in [3, Lemma 1], if a gadget G is t-NI secure it is also t-NIo secure for any public outputs.

In the sequel, we use the SecMult gadget that computes the multiplication of two masked inputs. It is one of the key building blocks of masking theory and has been introduced in [26,45] and proved t-SNI in [2].

We also use the MaskedCDT gadget that generates a masked sample that follows a tabulated Gaussian distribution of a fixed center c and a fixed standard deviation r. The table values are not sensitive so they are the same as for the unmasked implementation. This masked CDT algorithm was introduced in [4,20] and proved t-NI.

7.2 Two new gadgets

In Table 4, we introduce the known and new gadgets necessary for our sampler along with their properties. These properties will be proved in the following subsections.

7.2.1 Share-by-share Gaussian sampling In this section, we present a novel technique for generating a masked Gaussian sampling with an arbitrary masked center $[\![c]\!]$ of $c \in 1/C \cdot \mathbb{Z}$ for some fixed integer C. Note that $1/C \cdot \mathbb{Z}$ is not a ring, and thus the multiplication is not well-defined for shares in $1/C \cdot \mathbb{Z}$. This is not an issue in our application, since we never carry out multiplication of two sharings in this form.

Table 4. Masking properties of known and new gadgets

Gadget name	Security property	Reference
SecMult	t-SNI	[26,45,2]
Refresh	t-SNI	[9,2]
Unmask	t-NIo	[3]
MaskedCDT	t-NI	[4,20]
SecNTTMult	t-SNI	This work, Lemma 4
GaussShareByShare	t-NIo	This work, Lemma 3

We aim at considering a share by share generation. A direct and fast approach is to generate $z_i \leftarrow D_{\mathbb{Z},c_i,r/\sqrt{t+1}}$ for each share of c and to output $(z_0, \cdots, z_t)$ as $[\![z]\!]$. To ensure $z \sim D_{\mathbb{Z},c,r}$, it requires $r \geq \sqrt{2(t+1)}\eta_\epsilon(\mathbb{Z})$ according to [35], which yields a considerable security loss. To overcome this issue, we propose a different approach sampling shares over $1/B \cdot \mathbb{Z}$ with $B := \lceil\sqrt{2(t+1)}\rceil$ and utilizing rejection sampling to keep the masked output over $\mathbb{Z}$. Our masked Gaussian sampling algorithm is presented in Algorithm 9.

Correctness We now show that Algorithm 9 is correct for $r \geq \eta_\epsilon(\mathbb{Z})$. Since $r \geq \eta_\epsilon(\mathbb{Z}) \geq \frac{\sqrt{2(t+1)}}{B}\eta_\epsilon(\mathbb{Z})$, by [35, Theorem 3], in step 4, $z = \sum_{i=0}^{t} z_i$ follows $D_{1/B\cdot\mathbb{Z},c,r}$. Thanks to the rejection sampling, the support of the final output z is $\mathbb{Z}$ and noticing that the probability of each output z is proportional to $\rho_{r,c}(z)$, it follows that the distribution of z is $D_{\mathbb{Z},c,r}$. The rejection rate is $\frac{\rho_{r,c}(\mathbb{Z})}{\rho_{r,c}(1/B\cdot\mathbb{Z})} \approx 1/B$ as $r \geq \eta_\epsilon(\mathbb{Z}) \geq \eta_\epsilon(1/B \cdot \mathbb{Z})$. All in all, we have shown that Algorithm 9 provides $[\![z]\!] \sim D_{\mathbb{Z},c,r}$ at the cost of about $\sqrt{2(t+1)}$ average rejections.

Masking security Let $\bar{z} = \sum_i \bar{z}_i \bmod 1$. As the Unmask gadget is only NIo secure with public output $\bar{z}$, we need to show that $\bar{z}$ does not leak sensitive information, i.e. the output z and the center c. Indeed, the output only occurs when $\bar{z} = 0$, hence $\bar{z}$ is independent of the output. The support of $\bar{z}$ is $\frac{1}{B}\{0, 1, \cdots, B-1\}$ and $\Pr[\bar{z} = \frac{i}{B}] \propto \rho_{c,r}(\mathbb{Z} + \frac{i}{B}) = \rho_{c-\frac{i}{B},r}(\mathbb{Z}) \in [\frac{1-\epsilon}{1+\epsilon}, 1]\rho_r(\mathbb{Z})$ due to the smoothness condition $r \geq \eta_\epsilon(\mathbb{Z})$. Therefore the distribution of $\bar{z}$ is negligibly close to uniform independent of c. Consequently, $\bar{z}$ can be securely unmasked. As all the operations are performed share by share and assuming uniformly distributed shares of the input center c, we can deduce the following lemma.

Lemma 3. *The gadget* GaussShareByShare *is* t-NIo *secure with public output* $\bar{z}$.

In the implementation, one needs to instantiate an unmasked Gaussian sampling with arbitrary center and fixed standard deviation (line 2 of Algorithm 9). We chose a table based approach and follow the technique of [37] to use a reduced number of tables.

7.2.2 Polynomial multiplication In some lattice-based schemes such as Kyber or Dilithium, polynomial multiplication is always performed between a sensitive and a public polynomial. This means that, using polynomials protected with arithmetic masking, one can multiply each share independently by the public unmasked polynomial and obtain an arithmetic sharing of the result of the multiplication. In this work, we have polynomials multiplications with both operand in arithmetic masked form. Given $[\![a]\!]$ and $[\![b]\!] \in \mathscr{R}^{t+1}$, we want to compute $[\![c]\!] \in \mathscr{R}^{t+1}$ such

Algorithm 9: GaussShareByShare

Input: An unmasked standard deviation r. An arithmetic masking $[\![c]\!]$ of the center $c \in 1/C \cdot \mathbb{Z}$. Let $B := \lceil \sqrt{2(t+1)} \rceil$.
Result: An arithmetic masking $[\![z]\!]$ with z's distribution negligibly far from $D_{\mathbb{Z},c,r}$.

1 **for** $i \in [0, t]$ **do**
2 $\quad z_i \leftarrow D_{1/B \cdot \mathbb{Z}, c_i, r/\sqrt{t+1}}$
3 **end for**
/* Extracting the fractional part of z */
4 $[\![\bar{z}]\!]_1 \leftarrow (z_0 \bmod 1, \ldots, z_t \bmod 1)$ /* secret-sharing in $(\frac{1}{B} \cdot \mathbb{Z})/\mathbb{Z}$ */
5 **if** Unmask$([\![\bar{z}]\!]_1) \neq 0$ **then**
6 $\quad$ **restart** to step 1
7 **end if**
8 **return** $(z_0, \ldots, z_t)$

that $\sum_{i=0}^{t} c_i = \left(\sum_{i=0}^{t} a_i\right) \cdot \left(\sum_{i=0}^{t} b_i\right)$. To perform this masked polynomial multiplication, we propose to rely on an NTT-based multiplication. Using NTT, the product of two polynomials $a, b \in \mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}[x]/(x^d + 1)$ is given by

$$\mathsf{NTT}^{-1}(\mathsf{NTT}(a) \circ \mathsf{NTT}(b))$$

with $\circ$ the *coefficient-wise* product between two vectors in $\mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}$. Since the NTT is linear, it can be applied on each share independently and we only have to mask the coefficient-wise multiplication between elements of $\mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}$ using the technique of [26]. While a naive multiplication algorithm would require d^2 ISW multiplications, we only need d of them. Since we want to multiply the polynomials in $\mathbb{Z}$ and not in $\mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}$, we need to work with a modulus large enough to avoid any reduction in the result. Recall that it is also possible to use several $\mathsf{Q}^{\mathrm{mask}}$ with CRT techniques to reduce the size.

Let us define SecNTTMult, the masked product of two polynomials $[\![a]\!], [\![b]\!]$ arithmetically masked in $\mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}[x]/(x^d + 1)$ by

$$\mathsf{NTT}^{-1}((\mathsf{SecMult}(\mathsf{NTT}([\![a]\!])_j, \mathsf{NTT}([\![b]\!])_j)_{0 \leq j \leq d-1}).$$

This product is detailed in algorithm 16.

Lemma 4. SecNTTMult *(Alg. 16) is* t-SNI *secure.*

Note that here the shares are entire polynomials containing d coefficients. So, t probes actually provide $t \times d$ coefficients to the attacker.

Proof. Let $\delta \leq t$ be the number of observations made by the attacker. Assume the following distribution of the attacker δ observations of intermediate shared polynomials: δ_1 observations on the first NTT computation on $\hat{a}$, δ_2 observations on the first NTT computation on $\hat{b}$, δ_3 observations on the SecMult part (which provides the knowledge of the $d \times \delta_3$ *coefficients* of the probed polynomials), δ_4 observations on the last NTT^{-1} computation, and δ_5 observations of the returned values. Finally, we have $\sum_{i=1}^{5} \delta_i \leq \delta$. The algorithm NTT^{-1} is linear, thus it is t-NI and all the observations on steps 6 and 7 can be perfectly simulated with at most $\delta_4 + \delta_5$ shares of $\hat{c}$. The algorithm SecMult is applied coefficient-wise, thus each i-th execution has δ_3 observations of intermediate values (here coefficients) and $\delta_4 + \delta_5$ observations on the outputs (here coefficients too). By applying d times the t-SNI property for each SecMult operation, we can conclude that every

Algorithm 10: MaskedMITAKA$_{\mathbb{Z}}$Sampler

Input: A masked secret key in the following form: $([\![\widetilde{\mathbf{B}^*}]\!], [\![\widetilde{\mathbf{B}^*}^{-1}]\!], [\![\widetilde{v}]\!], [\![\mathbf{A}]\!])$ and a masked vector $[\![\mathbf{c}]\!]$ for a center $\mathbf{c} \in \mathscr{R}^2$, both arithmetically masked mod $\mathsf{Q}^{\mathrm{mask}}$.
Result: An unmasked sample $\mathbf{z} \sim D_{\mathscr{L}(\mathbf{B}),s,\mathbf{c}}$.

1 **Offline**
2 $[\![\mathbf{p}]\!] \leftarrow$ MaskedOfflineSampling$([\![\mathbf{A}]\!])$
3 **Online**
4 $[\![\mathbf{c}^{\mathrm{pert}}]\!] \leftarrow [\![\mathbf{c}]\!] - [\![\mathbf{p}]\!]$
5 $[\![\mathbf{c}^{\mathrm{pert}}]\!] \leftarrow$ SecNTTMult$([\![\widetilde{\mathbf{B}^*}^{-1}]\!], [\![\mathbf{c}^{\mathrm{pert}}]\!])$
6 $[\![\mathbf{v}]\!] \leftarrow$ MaskedOnlineSampling$([\![\widetilde{v}]\!], [\![\mathbf{c}^{\mathrm{pert}}]\!])$
7 $[\![\mathbf{z}]\!] \leftarrow$ SecNTTMult$([\![\widetilde{\mathbf{B}^*}]\!], [\![\mathbf{v}]\!])$
8 **return** $\sum_{i=0}^{t} \mathbf{z}_i \bmod \mathsf{Q}^{\mathrm{mask}}$

observation from steps 3 to 7 can be perfectly simulated with at most δ_3 shared (polynomials) of $\hat{a}$ and $\hat{b}$. The linearity of the NTT with arithmetic masking allows to finish proving that every set of size at most t observations containing $\sum_{i=1}^{4} \delta_i$ (resp. δ_5) intermediate (resp. returned) *polynomial shares* can be perfectly simulated with at most $\sum_{i=1}^{4} \delta_i$ polynomial shares of each input.

In the following, we extrapolate this polynomial multiplication technique to matrices of polynomials and keep the same notation SecNTTMult. We also remark that although SecNTTMult are sometimes called back-to-back in the masked samplers, this can be further optimized in practice: to minimize the number of NTT/NTT^{-1} invocations in an implementation, one could keep the NTT representation as much as possible, and then bring it back to the coefficient domain whenever it encounters GaussShareByShare, as explicitly described in H.

7.3 Masking the MITAKA$_{\mathbb{Z}}$ sampler

The detailed overall structure of the sampler is presented in Algorithm 10; the algorithms for the online and offline samplings are detailed in Appendix I.1. We remark that Algorithm 10 consists in a linear succession of gadgets with no dependency cycle, i.e. each line depends on freshly computed masked inputs. Thus, one can show that this algorithm is t-NI, as proved in Theorem 4 below. The proof is detailed in Section I.2.

Theorem 4. *The masked* MITAKA$_{\mathbb{Z}}$ *sampler (Alg. 10) is* t-NIo *with public output* $\mathbf{z}$.

7.4 Masking the MITAKA samplers

Algorithm 17 (resp. Algorithm 18) corresponds to our masked version of the RingPeikert sampler of Algorithm 1 (resp. the Hybrid sampler of Algorithm 3). Although masked MITAKA is instantiated with the MaskedHybrid sampler, we also include MaskedRingPeikert for completeness because the former can be essentially obtained by extending the basic masking paradigm outlined in the latter.

Contrary to MITAKA$_{\mathbb{Z}}$, one can remark that we here need to mask floating-point arithmetic. However, we can avoid it by representing each sensitive variable from $\mathscr{K}_{\mathbb{R}}$ as a fixed-point number. Concretely, an element $x \in \mathscr{K}_{\mathbb{R}}$ is approximated by $\tilde{x} \in \mathscr{K}_{\mathbb{R}}$ such that every coefficient of $q^k\tilde{x}$ is an integer, where k is a parameter determining the precision. Then we can secret-share $q^k\tilde{x}$ in

$\mathbb{Z}^d_{\mathsf{Q}^{\mathrm{mask}}}$ for $\mathsf{Q}^{\mathrm{mask}} \gg q^k$. Since we do not perform many multiplication operations, an accumulated scaling factor does not break the correctness of sampler if we choose sufficiently large $\mathsf{Q}^{\mathrm{mask}}$.

We also remark that a secret-shared center in fixed-point representation must be divided by a scaling factor q^k for the following 1-dimensional discrete Gaussian sampling to work share-by-share (i.e. division of $[\![q^k v_{i,j}]\!]$ at line 9-12 of Alg. 17, and of $[\![q^k z_{i,j}]\!]$ at lines 12-14 and 19-21 of Alg. 18, respectively). This division can be performed in floating-point arithmetic in practice. For the sum of shares to represent the correct center in the MaskedRingPeikert sampler, we further set $\mathsf{Q}^{\mathrm{mask}} = q^{k+\ell}$ for some $\ell > 0$. The resulting shares after division form a sharing of $\mathbf{v} = [(v_{1,j})_{j\in[0,d-1]}, (v_{2,j})_{j\in[0,d-1]}]$ over $(\mathbb{Q}/q^\ell\mathbb{Z})^{2d}$. Thanks to our GaussShareByShare introduced earlier, we are able to perform the discrete Gaussian sampling independently w.r.t each share of the center, while avoiding a factor of $\sqrt{t+1}$ overhead incurred in the standard deviation. As a result we obtain shares of discrete Gaussian samples $[\![z_{i,j}]\!]_{q^\ell}$ such that the distribution of $z_{i,j}$ is statistically close to $D_{\mathbb{Z},v_{i,j},r} \bmod q^\ell$ for every $i = 1, 2$ and $j \in [0, d-1]$. Since the output values of the signature are defined $\bmod q$ we can further map the shares to $[\![\cdot]\!]_q$ and the remaining computations can be performed $\bmod q$.

Since we invoke the above routine twice in the MaskedHybrid sampler, the initial masking modulus needs to be increased so that no wrap-around occurs during the masked computation of the second nearest plane. Concretely, the first nearest plane operations are computed with modulus $\mathsf{Q}^{\mathrm{mask}} = q^{2k+\ell}$; the second nearest plane operations are performed on $[\![\cdot]\!]_{q^{k+\ell}}$, with corresponding arithmetic shares of sensitive inputs; the output values can be represented in $[\![\cdot]\!]_q$ as in the MaskedRingPeikert.

Although a naive implementation of the MITAKA sampler should rely on floating-point arithmetic and thus naturally carry out FFT-based polynomial multiplications (as presented in Alg. 12), we instead make use of NTT in our masked algorithms. Notice that the masked instances only deal with a multiplication between polynomials mapped to $\mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}[x]/(x^d+1)$ (or $\mathbb{Z}_{q^{\ell+k}}[x]/(x^d+1)$ during the second nearest plane of the Hybrid sampler) thanks to the fixed-point representation. This allows us to exploit SecNTTMult as in MaskedMITAKA$_{\mathbb{Z}}$Sampler. One caveat is that in the current setting $\mathsf{Q}^{\mathrm{mask}}$ is restricted to a *power* of q, but we are able to show such a choice is indeed NTT-friendly. Recall that the prime q is usually chosen such that $q = 1 \bmod 2d$, so that $x^d + 1$ has exactly d roots $(\zeta, \zeta^3, \ldots, \zeta^{2d-1})$ over $\mathbb{Z}_q$. Now thanks to the Hensel lifting, one can construct another set of d roots $(\omega, \omega^3, \ldots, \omega^{2d-1})$ over $\mathbb{Z}_{q^2}$, such that $\omega = \zeta \bmod q$. By iterating this procedure until the roots for a sufficiently large modulus $\mathsf{Q}^{\mathrm{mask}}$ are obtained, we can indeed utilize the NTT for evaluating $f(x) \in \mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}[x]/(x^d+1)$ on the primitive $2d$-th roots of unity.

We are able to prove that both masked samplers meet the standard security notion (t-NIO) for masked signature schemes. The proof is detailed in Section I.3 and I.4.

Theorem 5. *The masked* MITAKA *sampler (Alg. 18) is t-NIO secure with public output* $\mathbf{v}_0$.

References

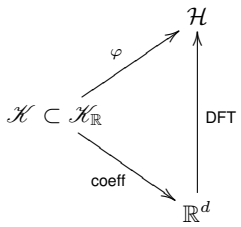
1. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange - A new hope. In: Holz, T., Savage, S. (eds.) USENIX Security 2016. pp. 327–343. USENIX Association (Aug 2016)
2. Barthe, G., Belaïd, S., Dupressoir, F., Fouque, P.A., Grégoire, B., Strub, P.Y., Zucchini, R.: Strong non-interference and type-directed higher-order masking. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) ACM CCS 2016. pp. 116–129. ACM Press (Oct 2016).
3. Barthe, G., Belaïd, S., Espitau, T., Fouque, P.A., Grégoire, B., Rossi, M., Tibouchi, M.: Masking the GLP lattice-based signature scheme at any order. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 354–384. Springer, Heidelberg (Apr / May 2018).

4. Barthe, G., Belaïd, S., Espitau, T., Fouque, P.A., Rossi, M., Tibouchi, M.: GALACTICS: Gaussian sampling for lattice-based constant- time implementation of cryptographic signatures, revisited. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019. pp. 2147–2164. ACM Press (Nov 2019).
5. Becker, A., Ducas, L., Gama, N., Laarhoven, T.: New directions in nearest neighbor searching with applications to lattice sieving. In: Krauthgamer, R. (ed.) 27th SODA. pp. 10–24. ACM-SIAM (Jan 2016).
6. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 41–69. Springer, Heidelberg (Dec 2011).
7. Box, G.E.P., Muller, M.E.: A note on the generation of random normal deviates. The Annals of Mathematical Statistics **29**(2), 610–611 (1958)
8. Chuengsatiansup, C., Prest, T., Stehlé, D., Wallet, A., Xagawa, K.: ModFalcon: Compact signatures based on module-NTRU lattices. In: Sun, H.M., Shieh, S.P., Gu, G., Ateniese, G. (eds.) ASIACCS 20. pp. 853–866. ACM Press (Oct 2020).
9. Coron, J.S.: Higher order masking of look-up tables. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 441–458. Springer, Heidelberg (May 2014).
10. Ding, J., Chen, M.S., Petzoldt, A., Schmidt, D., Yang, B.Y., Kannwischer, M., Patarin, J.: Rainbow. Tech. rep., National Institute of Standards and Technology (2020), available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions
11. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal Gaussians. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 40–56. Springer, Heidelberg (Aug 2013).
12. Ducas, L., Galbraith, S., Prest, T., Yu, Y.: Integral matrix gram root and lattice gaussian sampling without floats. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part II. LNCS, vol. 12106, pp. 608–637. Springer, Heidelberg (May 2020).
13. Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-Dilithium: A lattice-based digital signature scheme. IACR TCHES **2018**(1), 238–268 (2018). , https://tches.iacr.org/index.php/TCHES/article/view/839
14. 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, Heidelberg (Dec 2014).
15. 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, Heidelberg (Dec 2012).
16. Ducas, L., Prest, T.: Fast Fourier orthogonalization. Cryptology ePrint Archive, Report 2015/1014 (2015), https://eprint.iacr.org/2015/1014
17. Espitau, T., Kirchner, P.: The nearest-colattice algorithm: Time-approximation tradeoff for approx-cvp. ANTS XIV p. 251
18. 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, Heidelberg (May 2020).
19. 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).
20. Gérard, F., Rossi, M.: An efficient and provable masked implementation of qtesla. In: Belaïd, S., Güneysu, T. (eds.) CARDIS 2019. Lecture Notes in Computer Science, vol. 11833, pp. 74–91. Springer (2019)
21. 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, Heidelberg (Aug 1997).
22. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Silverman, J.H., Whyte, W.: Performance improvements and a baseline parameter generation algorithm for NTRUSign. Cryptology ePrint Archive, Report 2005/274 (2005), https://eprint.iacr.org/2005/274
23. 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, Heidelberg (Apr 2003).
24. Howe, J., Prest, T., Ricosset, T., Rossi, M.: Isochronous gaussian sampling: From inception to implementation. In: Ding, J., Tillich, J.P. (eds.) Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020. pp. 53–71. Springer, Heidelberg (2020).
25. Howgrave-Graham, N.: A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 150–169. Springer, Heidelberg (Aug 2007).
26. Ishai, Y., Sahai, A., Wagner, D.: Private circuits: Securing hardware against probing attacks. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 463–481. Springer, Heidelberg (Aug 2003).
27. Karabulut, E., Aysu, A.: Falcon down: Breaking Falcon post-quantum signature scheme through side-channel attacks (2021)

28. Kirchner, P., Espitau, T., Fouque, P.A.: Fast reduction of algebraic lattices over cyclotomic fields. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part II. LNCS, vol. 12171, pp. 155–185. Springer, Heidelberg (Aug 2020).
29. Kirchner, P., Fouque, P.A.: Revisiting lattice attacks on overstretched NTRU parameters. In: Coron, J.S., Nielsen, J.B. (eds.) EUROCRYPT 2017, Part I. LNCS, vol. 10210, pp. 3–26. Springer, Heidelberg (Apr / May 2017).
30. Laarhoven, T.: Search problems in cryptography. Ph.D. thesis, Eindhoven University of Technology (2015)
31. Langlois, A., Stehlé, D.: Worst-case to average-case reductions for module lattices. Des. Codes Cryptogr. **75**(3), 565–599 (2015)
32. Lyubashevsky, V., Ducas, L., Kiltz, E., Lepoint, T., Schwabe, P., Seiler, G., Stehlé, D.: CRYSTALS-DILITHIUM. Tech. rep., National Institute of Standards and Technology (2019), available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions
33. Lyubashevsky, V., Ducas, L., Kiltz, E., Lepoint, T., Schwabe, P., Seiler, G., Stehlé, D., Bai, S.: CRYSTALS-DILITHIUM. Tech. rep., National Institute of Standards and Technology (2020), available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions
34. Lyubashevsky, V., Wichs, D.: Simple lattice trapdoor sampling from a broad class of distributions. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 716–730. Springer, Heidelberg (Mar / Apr 2015).
35. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 21–39. Springer, Heidelberg (Aug 2013).
36. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM J. Comput. **37**(1), 267–302 (2007)
37. Micciancio, D., Walter, M.: Gaussian sampling over the integers: Efficient, generic, constant-time. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 455–485. Springer, Heidelberg (Aug 2017).
38. Nguyen, P.Q., Regev, O.: Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. Journal of Cryptology **22**(2), 139–160 (Apr 2009).
39. Peikert, C.: An efficient and parallel Gaussian sampler for lattices. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 80–97. Springer, Heidelberg (Aug 2010).
40. Pornin, T.: New efficient, constant-time implementations of Falcon. Cryptology ePrint Archive, Report 2019/893 (2019), https://eprint.iacr.org/2019/893
41. 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, Heidelberg (Apr 2019).
42. Prest, T.: Gaussian Sampling in Lattice-Based Cryptography. Ph.D. thesis, École Normale Supérieure, Paris, France (2015)
43. 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, Heidelberg (Dec 2017).
44. 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 (2020), available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions
45. Rivain, M., Prouff, E.: Provably secure higher-order masking of AES. In: Mangard, S., Standaert, F.X. (eds.) CHES 2010. LNCS, vol. 6225, pp. 413–427. Springer, Heidelberg (Aug 2010).
46. 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, Heidelberg (Dec 2018).
47. Zhao, R.K., Steinfeld, R., Sakzad, A.: FACCT: Fast, compact, and constant-time discrete gaussian sampler over integers. IEEE Transactions on Computers **69**(1), 126–137 (2020)

A Additional background

In the coefficient representation, the adjoint is given by $f^* = (f_0, -f_{d-1}, \ldots, -f_1)$. We have $\sqrt{d}\|f\| = \|\varphi(f)\|$, where both norms are ℓ_2 norm over the coefficients of the elements. All $f \in \mathcal{K}_\mathbb{R}$ can also be represented by the nega-circulant matrix $\mathrm{M}(f) \in \mathbb{R}^{d\times d}$ of multiplication by f in the power basis $1, x, \ldots, x^{d-1}$ of $\mathbb{R}[x]/(x^d+1)$. This extends block-wise to matrices over $\mathcal{K}_\mathbb{R}$.



For this work, there are two useful representations of $\mathcal{K}_\mathbb{R}$, and embeddings of $\mathcal{K}$ extend straightforwardly to this algebra. The coefficient embedding directly identifies a real polynomial in $\mathbb{R}[x]/(x^d+1)$ with its coefficient vector in $\mathbb{R}^d$ endowed with the standard dot product, while the embedding φ identifies it to the space $\mathcal{H} = \{\mathbf{v} \in \mathbb{C}^d : v_i = \overline{v_{d/2+i}}, 1 \leqslant i \leqslant d/2\}$. The standard Hermitian product on $\mathbb{C}^d$ induces a real inner product on $\mathcal{H}$. The Discrete Fourier Transform (DFT) gives a linear isomorphism between the coefficient embedding space and $\mathcal{H}$, which is depicted by the commutative diagram on the left. In practice, it can be computed in quasilinear time using the Fast Fourier Transform (FFT) algorithm.

In this work we also consider implicitly a version of NTRU lattices under the embedding φ, which can be described thanks to the matrices

$$\varphi_i(\mathbf{B}_{f,g}) := \begin{bmatrix} \varphi_i(f) & \varphi_i(F) \\ \varphi_i(g) & \varphi_i(G) \end{bmatrix}, \text{ and}$$

$$\varphi(\mathbf{B}_{f,g}) := \begin{bmatrix} \varphi_1(\mathbf{B}_{f,g}) & & \\ & \ddots & \\ & & \varphi_d(\mathbf{B}_{f,g}) \end{bmatrix} \in \mathbb{C}^{2d\times 2d}.$$

Noting that $\det \varphi_i(\mathbf{B}_{f,g}) = \varphi_i(\det \mathbf{B}_{f,g}) = q$, the lattice $\varphi(\mathbf{B}_{f,g}\mathscr{R}^2)$ has volume $q^d\Delta_\mathscr{K}$ in $\mathcal{H}$.

Let $\mathbf{B}_{f,g} = [\mathbf{b}_1|\mathbf{b}_2]$ and $\widetilde{\mathbf{b}}_2 = (\widetilde{F}, \widetilde{G})$ the second Gram-Schmidt vector of $\mathbf{B}_{f,g}$. For all $i \leqslant d/2$, note that $\varphi_i(\langle \mathbf{b}_1, \mathbf{b}_1\rangle_\mathscr{K}) = |\varphi_i(f)|^2 + |\varphi_i(g)|^2$ and $\varphi_i(\langle \widetilde{\mathbf{b}}_2, \widetilde{\mathbf{b}}_2\rangle_\mathscr{K}) = |\varphi_i(\widetilde{F})|^2 + |\varphi_i(\widetilde{G})|^2$. In particular, the Euclidean norm of the largest Gram-Schmidt vector of $\varphi(\mathbf{B}_{f,g})$ is $|\mathbf{B}_{f,g}|_\mathscr{K}$.

Proof (Proof of Lemma 1). With a basis $\mathbf{B}_{f,g} = [\mathbf{b}_1|\mathbf{b}_2]$, we have $\widetilde{\mathbf{b}}_1 = (f, g)$ of squared norm $ff^* + gg^* = \langle \widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_1\rangle_\mathscr{K}$, and $\det \widetilde{\mathbf{B}}^*\widetilde{\mathbf{B}} = q^2 = \langle \widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_1\rangle_\mathscr{K}\langle \widetilde{\mathbf{b}}_2, \widetilde{\mathbf{b}}_2\rangle_\mathscr{K}$. Both the left side of the claimed inequality and the expression of $|\mathbf{B}_{f,g}|_\mathscr{K}$ follow. The $\mathcal{K}_\mathbb{R}$-spectrum of $\widetilde{\mathbf{B}}^*\widetilde{\mathbf{B}}$ is $\{\langle \mathbf{b}_1, \mathbf{b}_1\rangle_\mathscr{K}, \langle \widetilde{\mathbf{b}}_2, \widetilde{\mathbf{b}}_2\rangle_\mathscr{K}\}$. In particular, we get

$$s_1(\widetilde{\mathbf{B}})^2 = \max(\|\varphi(\langle \mathbf{b}_1, \mathbf{b}_1\rangle)\|_\infty, \|\varphi(\langle \widetilde{\mathbf{b}}_2, \widetilde{\mathbf{b}}_2\rangle)\|_\infty),$$

which gives the last claimed equality.

Let $\mathbf{e}_1, \ldots, \mathbf{e}_{2d}$ be the canonical basis of $\mathbb{C}^{2d}$. Consequently, we have $\|\varphi(\mathbf{B}_{f,g})\mathbf{e}_{2i-1}\|^2 = |\varphi_i(f)|^2 + |\varphi_i(g)|^2$ and $\|\varphi(\mathbf{B}_{f,g})\mathbf{e}_{2i}\|^2 = |\varphi_i(F)|^2 + |\varphi_i(G)|^2$. By definition of the operator norm, this gives $\max(\|\varphi(\langle \mathbf{b}_1, \mathbf{b}_1\rangle_\mathscr{K})\|_\infty, \|\varphi(\langle \mathbf{b}_2, \mathbf{b}_2\rangle_\mathscr{K})\|_\infty) \leqslant s_1(\mathbf{B})^2$. We check that $\varphi_i(\widetilde{\mathbf{b}}_2)$ is the Gram-Schmidt orthogonalization of $\varphi_i(\mathbf{b}_2)$ with respect to $\varphi_i(\mathbf{b}_1)$, hence we have

$$\|\varphi_i(\widetilde{\mathbf{b}}_2)\|^2 = \varphi_i(\langle \widetilde{\mathbf{b}}_2, \widetilde{\mathbf{b}}_2\rangle_\mathscr{K}) \leqslant \|\varphi_i(\mathbf{b}_2)\|^2 = \varphi_i(\langle \mathbf{b}_2, \mathbf{b}_2\rangle_\mathscr{K}),$$

for all i. This implies the right side of the inequality. Next, the trace of $\mathbf{B}_{f,g}^*\mathbf{B}_{f,g}$ is $T \in \mathscr{K}^{++}$, so that its characteristic polynomial is $\chi = X^2 - TX + q^2$ with totally real discriminant $\Delta =$

$T^2 - 4q^2$. If there was an embedding $\mathscr{K} \to \mathbb{C}$ where Δ is negative, this would contradict that the corresponding embedding of $\mathbf{B}_{f,g}^* \mathbf{B}_{f,g}$ is positive definite in the usual sense. Hence $\Delta \in \mathscr{K}^{++}$, and this also means that (each embedding of) $T + \sqrt{\Delta}$ is larger than (the corresponding embedding of) $T - \sqrt{\Delta}$. The last claim follows.

In matrix form, one can write the Gram-Schmidt orthogonalization as $\mathbf{B} = \widetilde{\mathbf{B}}\mathbf{U}_u$ with $u = \frac{\langle \mathbf{b}_1, \mathbf{b}_2 \rangle_{\mathscr{K}}}{\langle \mathbf{b}_1, \mathbf{b}_1 \rangle_{\mathscr{K}}}$ and

$$\mathbf{U}_u := \begin{pmatrix} 1 & u \\ & 1 \end{pmatrix} \in \mathscr{K}^{2\times 2}.$$

For all $u, u' \in \mathscr{K}$, one has $\mathbf{U}_u^{-1} = \mathbf{U}_{-u}$ and $\mathbf{U}_u \mathbf{U}_{u'} = \mathbf{U}_{u+u'}$.

B Finer-grained selection parameter using cyclotomic fields of composite conductors

As mentioned earlier, a strength of the MITAKA scheme lies in the possibility to instantiate it over any number fields $\mathscr{K}$. To remain practically competitive, one needs to be sable to sample efficiently discrete Gaussian over its ring of integers $\mathcal{R} = \mathcal{O}_{\mathscr{K}}$. We saw that on cyclotomic of conductor 2^n, this is trivially the case as the power basis is orthogonal; and as a result sampling over $\mathcal{R}$ boils down to perform a coefficient wise sampling. In this section, we show that we can efficiently sample in cyclotomic fields of *smooth* enough conductor.

B.1 Geometry of the power basis

Let $\mathbb{Q}(\zeta_m)$ be the cyclotomic field of conductor $m = p^a 2^b$ for an odd prime number p and non-negative integers a, b. Set $\mathbf{B} = (1, \zeta_m, \ldots, \zeta_m^{\varphi(m)-1})$. As $\mathbb{Q}(\zeta_m)$ is the compositum of the prime-power cyclotomic fields $\mathbb{Q}(\zeta_{2^a}$ and $\mathbb{Q}(\zeta_{2^b}$, its ring of integers is the tensor product of the ring of integers of these two fields, so that a routine computation ensures that the Gram-matrix of a well chosen reordering of $\mathbf{B}$ in the canonical embedding is:

$$\frac{\varphi(m)}{p-1} G(p,b) \otimes \mathrm{Id}_{\frac{\varphi(m)}{p-1}},$$

where

$$G(p,b) = \begin{cases} \mathrm{Circ}_{p-1}(p-1, -1, \ldots, -1) & \text{if } b = 0 \\ \mathrm{Circ}_{p-1}(p-1, 1, -1, \ldots, -1, 1) & \text{if } b > 0 \end{cases},$$

for $\mathrm{Circ}_{p-1}(X)$ designating the circulant matrix of size $p-1 \times p-1$ and coefficients following the $(p-1)$-uple X.

B.2 Sampling

As a consequence, we can use the hybrid sampler to reduce the sampling over $\mathcal{R}$ to a module sampling over a module of rank $p-1$, which Gram-matrix is $G(p,B)$, defined over a subring isometric to $\frac{\varphi(m)}{p-1}\mathbb{Z}^{\frac{\varphi(m)}{p-1}}$. Let us study this matrix in more details. Its principal minor G_i of order i is the circulant matrix $\mathrm{Circ}_i(p-1, -1, \ldots, -1)$. Elementary theory of Toeplitz-like matrices ensures that this minor have for spectrum:

$$\mathrm{Sp}(G_i) = \{p - i, \underbrace{p, \ldots, p}_{i-1-\text{times}}\},$$

so that its determinant is $\det(G_i) = (p-i)p^{i-1}$. Henceforth a direct induction ensures that the (square of the) diagonal of the Cholesky decomposition of $G(p,b)$ is

$$\left[p-1, \frac{(p-2)p}{p-1}, \frac{(p-3)p}{(p-2)}, \ldots, \frac{p}{2}\right],$$

which is a decreasing sequence. Hence the maximum value of the diagonal elements of the Cholesky decomposition of $G(p,b)$ is $\sqrt{p-1}$. As it is also the norm of the corresponding Gram-Schmidt vectors. As such, the hybrid sampler induces an additional $\sqrt{p-1}$ compared to the sampling over the power-of-two cyclotomic fields.

B.3 Practical impact on the parameter selction

Using this analysis, we get a wider range of parameters for instantiating the MITAKA scheme. In particular, the 3-smooth conductors are of particular interest as they only induce a loss of a factor $\sqrt{2}$ in the sampler quality and allows sampling which is asymptotically as fast as the sampling in power-of-two conductors, using the hybrid sampler. In Fig. 3, we show the impact of the base field on the bit-security with regards to the parameter α. Moreover, achievable quality parameters α for the MITAKA hybrid sampler when using the key generation algorithm of Section 4 are displayed in Fig. 4 (for each individual degree) and summarized in Fig. 5 based on the median output of Algorithm 8.

C Concentration bounds for Gamma distributions

Let $X \sim \Gamma(\alpha, \beta)$ be a Gamma-distributed random variable. Its logarithmic moment generating function is as follows for all $\lambda < \beta$:

$$\begin{aligned}
\Phi_X(\lambda) &= \log \mathbb{E}\left[e^{\lambda X}\right] \\
&= \log \int_0^{+\infty} e^{\lambda x} \cdot \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} dx \\
&= \log \left(\frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^{+\infty} x^{\alpha-1} e^{(\lambda-\beta)x} dx \right) \\
&= \log \left(\frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^{+\infty} \frac{z^{\alpha-1}}{(\beta-\lambda)^{\alpha-1}} e^{-z} \frac{dz}{\beta-\lambda} \right) \\
&= \log \left(\frac{\beta^\alpha}{\Gamma(\alpha)} \cdot \frac{\Gamma(\alpha)}{(\beta-\lambda)^\alpha} \right) \\
&= -\alpha \log \left(1 - \frac{\lambda}{\beta}\right).
\end{aligned}$$

In particular, $\Phi'_X(\lambda) = \frac{\alpha}{\beta-\lambda}$, and thus the map $\lambda \mapsto t\lambda - \Phi_X(\lambda)$ is maximal at λ_0 such that:

$$t = \frac{\alpha}{\beta - \lambda_0}, \quad \text{namely} \quad \lambda_0 = \beta - \frac{\alpha}{t}$$

(assuming $t > 0$; otherwise the function is unbounded on $(-\infty, \beta)$). Therefore, the Cramér transform of X is given by:

$$\Phi^*(t) = \sup_{\lambda < 1} \left(t\lambda - \Phi_X(\lambda)\right) = t\lambda_0 - \Phi_X(\lambda_0) = \beta t - \alpha - \alpha \log \frac{\beta t}{\alpha}$$

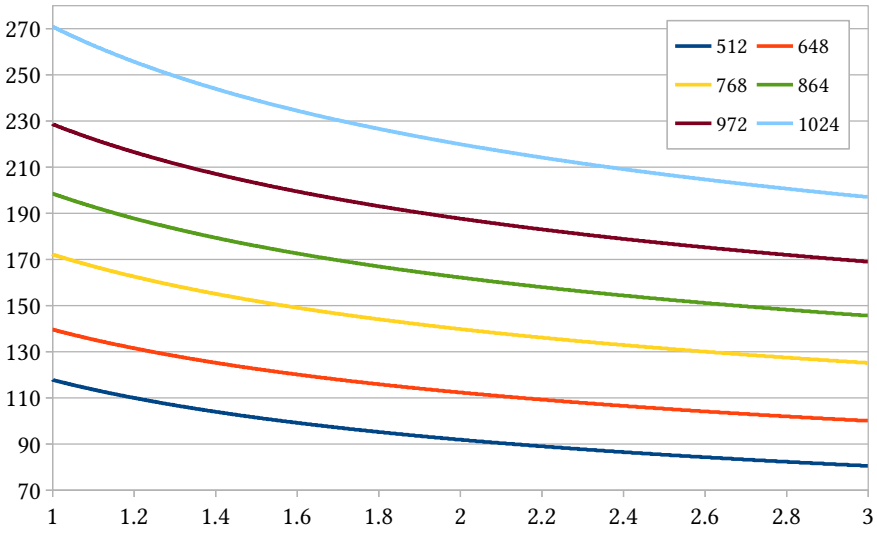


Fig. 3. Security of the MITAKA scheme for different choices of cyclotomic fields with 3-powersmooth conductors.

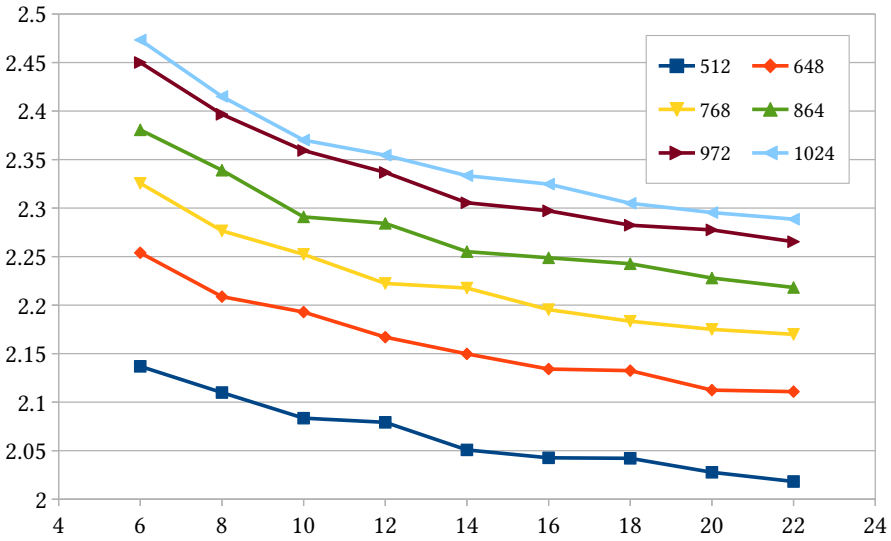
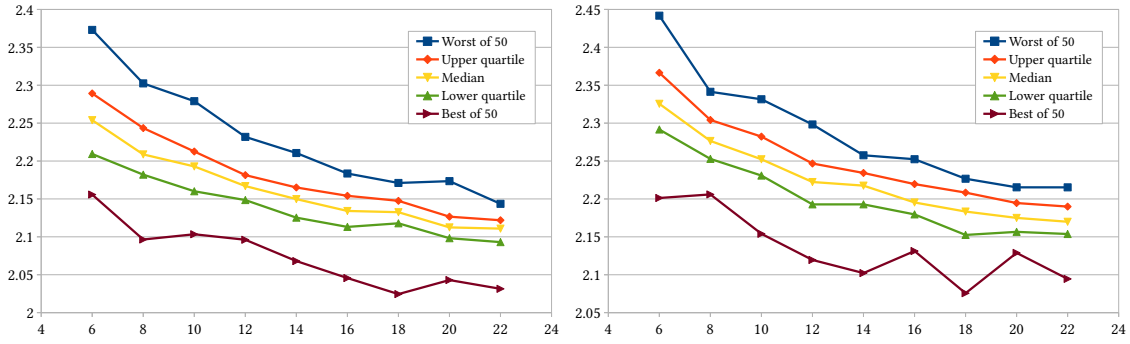


Fig. 4. Median quality α reached in the same cyclotomic fields by the optimized sampler of Algorithm 8 for various choices of m (50 trials each, $\sigma_0 = 1.17\sqrt{q/2d}$, G coset representatives of $\mathrm{Gal}(\mathscr{K}/\mathbb{Q})/\langle\tau_*\rangle$).



(a) Degree 648 (b) Degree 768

(c) Degree 864 (d) Degree 972

Fig. 5. Quality α reached in 3-powersmooth degrees by the optimized sampler of Algorithm 8 for various choices of m (50 trials each, $\sigma_0 = 1.17\sqrt{q/2d}$, G coset representatives of $\mathrm{Gal}(\mathscr{K}/\mathbb{Q})/\langle\tau_*\rangle$).

for $t > 0$. Noting that $\mathbb{E}[X] = \Phi'_X(0) = \alpha/\beta$, the Cramér–Chernoff inequality then yields:

$$\Pr[X > t] \leqslant e^{\alpha-\beta t}\Big(\frac{\beta t}{\alpha}\Big)^{\alpha} \leqslant \Big(\frac{e\beta t}{\alpha}\Big)^{\alpha} \qquad \text{for } t > \frac{\alpha}{\beta}, \tag{2}$$

$$\Pr[X < t] \leqslant e^{\alpha-\beta t}\Big(\frac{\beta t}{\alpha}\Big)^{\alpha} \leqslant (2e)^{\alpha} e^{-\beta t/2} \qquad \text{for } 0 < t < \frac{\alpha}{\beta}, \tag{3}$$

where the second inequalities in the two cases follow from $e^{-\beta t} \leqslant 1$ and $\beta t/\alpha \leqslant 2e^{\beta t/2\alpha}$ respectively.

Now we consider $X_1, \ldots, X_k \sim \Gamma(\alpha, \beta)$ independent gamma-distributed random variables, and want to estimate the tail bound on $\max(X_1, \ldots, X_k, \gamma/X_1, \ldots, \gamma/X_k)$ for some constant $\gamma > 0$.

On the one hand, by the union bound and Eq. (3), we have, for $0 < \gamma/t < \alpha/\beta$:

$$\Pr\Big[\max\big(\frac{\gamma}{X_1}, \ldots, \frac{\gamma}{X_k}\big) > t\Big] = \Pr\Big[\min(X_1, \ldots, X_k) < \frac{\gamma}{t}\Big] \leqslant k\Big(\frac{e\beta\gamma}{\alpha t}\Big)^{\alpha}.$$

In particular, we have $\Pr[\max(\gamma/X_1, \ldots, \gamma/X_k) > t] \leqslant \delta_{\min}$ as soon as:

$$\begin{aligned} & k\Big(\frac{e\beta\gamma}{\alpha t}\Big)^{\alpha} \leqslant \delta_{\min} \\ \iff \quad & \frac{e\beta\gamma}{\alpha t} \leqslant \Big(\frac{\delta_{\min}}{k}\Big)^{1/\alpha} \\ \iff \quad & t \geqslant \frac{e\beta\gamma}{\alpha}\Big(\frac{k}{\delta_{\min}}\Big)^{1/\alpha}. \end{aligned}$$

On the other hand, by the union bound and Eq. (2), we also have, for $t > \alpha/\beta$:

$$\Pr[\max(X_1, \ldots, X_k) > t] \leqslant k(2e)^{\alpha} e^{-\beta t/2}.$$

In particular, we have $\Pr[\max(X_1, \ldots, X_k) > t] \leqslant \delta_{\max}$ as soon as:

$$\begin{aligned} & k(2e)^{\alpha} e^{-\beta t/2} \leqslant \delta_{\max} \\ \iff \quad & \alpha\log(2e) - \frac{\beta}{2}t \leqslant -\log\frac{k}{\delta_{\max}} \\ \iff \quad & t \geqslant \frac{2}{\beta}\Big(\log\frac{k}{\delta_{\max}} + \alpha\log(2e)\Big). \end{aligned}$$

This results in the following theorem.

Theorem 6. *Let $X_1, \ldots, X_k \sim \Gamma(\alpha, \beta)$ independent gamma-distributed random variables, and fix $\gamma > 0$, $\delta_{\min} \in (0, 1)$ and $\delta_{\max} \in (0, 1)$. We have:*

$$\Pr\Big[\max\big(X_1, \ldots, X_k, \frac{\gamma}{X_1}, \ldots, \frac{\gamma}{X_k}\big) < \max(t_{\min}, t_{\max})\Big] \leqslant \delta_{\min} + \delta_{\max}$$

where:

$$t_{\min} = \frac{e\beta\gamma}{\alpha}\Big(\frac{k}{\delta_{\min}}\Big)^{1/\alpha} \quad \textit{and} \quad t_{\max} = \frac{2}{\beta}\Big(\log\frac{k}{\delta_{\max}} + \alpha\log(2e)\Big).$$

This simply follows from the union bound applied to the previous inequalities, together with the observation that $t_{\min}$ and $t_{\max}$ always satisfy the required bounds $0 < \gamma/t_{\min} < \alpha/\beta$ and $t_{\max} > \alpha/\beta$.

We can then apply this result to estimate the quality of the hybrid sampler obtained from an NTRU basis $\mathbf{B}_{f,g}$, which is given by the norm $|\mathbf{B}_{f,g}|_{\mathscr{K}}$, where (as recalled in Lemma 1):

$$|\mathbf{B}_{f,g}|_{\mathscr{K}}^2 = \max\left(\|\varphi(ff^* + gg^*)\|_\infty, \left\|\frac{q^2}{\varphi(ff^* + gg^*)}\right\|_\infty\right). \tag{4}$$

The ring elements f, g have all their coefficients sampled independently according to the centered discrete Gaussian of standard deviation $\sigma_0 = \sqrt{\frac{\nu q}{4d}}$ for some $\nu > 0$. Then, all the embeddings $\varphi_i(f), \varphi_i(g) \in \mathbb{C}$ are (statistically close to) 2-dimensional discrete Gaussian of standard deviation $\sqrt{\nu q/4}$, which we heuristically assume behave like normal vectors of the same standard deviation.[15] Then, the $\varphi_i(ff^*), \varphi_i(gg^*)$ $(1 \leqslant i \leqslant d/2)$ are independent scaled $\chi^2(2)$ distributed random variables, or equivalently $\Gamma\left(1, \frac{2}{\nu q}\right)$-distributed random variables. As a result, the $X_i = \varphi_i(ff^* + gg^*)$ $(1 \leqslant i \leqslant d/2)$ are independent $\Gamma\left(2, \frac{2}{\nu q}\right)$-distributed random variables. Now, by Eq. (4), we have:

$$|\mathbf{B}_{f,g}|_{\mathscr{K}}^2 = \max\left(X_1, \ldots, X_{d/2}, \frac{q^2}{X_1}, \ldots, \frac{q^2}{X_{d/2}}\right),$$

which is of the form considered in Theorem 6 with $\alpha = 2$, $\beta = \frac{2}{\nu q}$, $\gamma = q^2$ and $k = d/2$. Therefore, it follows that for any choice of $\delta_{\min}, \delta_{\max} \in (0, 1)$, $\Pr\left[|\mathbf{B}_{f,g}|_{\mathscr{K}}^2 < \max(t_{\min}, t_{\max})\right] \leqslant \delta_{\min} + \delta_{\max}$, where:

$$t_{\min} = \frac{eq}{\nu}\sqrt{\frac{d}{2\delta_{\min}}} \quad \text{and} \quad t_{\max} = \nu q \log\frac{2e^2 d}{\delta_{\max}}.$$

We can moreover choose ν in key generation in such a way that $t_{\min} = t_{\max}$, by setting:

$$\nu = \sqrt{\frac{e}{\log(2e^2 d/\delta_{\max})}} \cdot \sqrt[4]{\frac{d}{2\delta_{\min}}}.$$

Picking e.g. $\delta_{\max} = \delta_{\min} = 1/4$, we get $\Pr[|\mathbf{B}_{f,g}|_{\mathscr{K}}^2 < t] \leqslant 1/2$, where:

$$t = q\sqrt{e\log(8e^2 d)}\sqrt[4]{2d} = O(q \cdot d^{1/4}\log^{1/2} d).$$

Thus, the quality of the hybrid sampler scales as $O(d^{1/8}\log^{1/4} d)$, and not $\sqrt{\log d}$ as incorrectly claimed in [42]. The error in that work was due to the overly optimistic assumption that the $q^2/\varphi_i(ff^* + gg^*)$ components had the same tail behavior as the $\varphi_i(ff^* + gg^*)$, which is not the case, since they are Inverse-Gamma distributed, and thus not sub-Gamma; in fact, they do not even have finite variance.

D Security arguments from Renyi divergence

Let $\mathscr{D}, \mathscr{D}'$ be two distributions sharing the same support. Their relative error is defined as $\Delta_{RE}(\mathscr{D}, \mathscr{D}') = \sup_{\mathbf{x}\in\mathrm{Supp}(\mathscr{D})}\left|\frac{\mathscr{D}(\mathbf{x})}{\mathscr{D}'(\mathbf{x})} - 1\right|$.

[15] We can do away with this heuristic assumption in the analysis by generalizing Theorem 6 to sub-Gamma random variables.

Lemma 5 ([43], adapted). *Assume that for two distributions $\mathscr{D}, \mathscr{D}'$ with the same support, we have $\Delta_{RE}(\mathscr{D}, \mathscr{D}') \leqslant \delta$ for some $\delta > 0$. Then we have*

$$R_{2\lambda}(\mathscr{D}\|\mathscr{D}') \leqslant \left(1 + \frac{\lambda(2\lambda-1)\delta^2}{(1-\delta)^{2\lambda+1}}\right)^{\frac{1}{2\lambda-1}}.$$

Additionally, if $\lambda \in \{128, 256\}$ and $\delta < 2^{-10}$, then:

$$R_{2\lambda}(\mathscr{D}\|\mathscr{D}') \leqslant 1 + 2\lambda\delta^2.$$

In practice, the parameter δ is quite smaller than the bound in the statement (which ensures the correctness of the second inequality). As argued in [43], when trying to solve a search problem, one cannot distinguish if one queried $\mathscr{D}$ or $\mathscr{D}'$ up to Q times as long as $2\lambda\delta^2 \leqslant (4Q)^{-1}$. In practice for signature algorithms, it is often the case that $Q = 2^{64}$ and λ is a target security level, e.g. 128 or 256. The value for Q can be larger if rejection sampling happens, as the target sampler will definitely be queried more time. Contrary to, say, BLISS, this does not happen for the lattice samplers considered in this work.

E Analysis of Peikert's sampler.

The following lemma can be identified as an intermediate step when calculating that the convolution of two Gaussians distribution is again a Gaussian distribution. It is very useful in analyzing Peikert's sampler and more generally, perturbative arguments for sampling discrete Gaussian distributions.

Lemma 6. *[e.g. [39, Fact 2.1]] Let Σ_1, Σ_2 be positive definite $d \times d$ matrices and $\mathbf{x}, \mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}^d$. We have*

$$\rho_{\Sigma_1}(\mathbf{c}_1 - \mathbf{x})\rho_{\Sigma_2}(\mathbf{x} - \mathbf{c}_2) = \rho_{\Sigma_1+\Sigma_2}(\mathbf{c}_1 - \mathbf{c}_2)\rho_{\Sigma'}(\mathbf{x} - \mathbf{c}'),$$

where $\Sigma' = (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1}$ and $\mathbf{c}' = \Sigma'(\Sigma_1^{-1}\mathbf{c}_1 + \Sigma_2^{-1}\mathbf{c}_2)$.

In our work, the second factor on the right-hand side will lead to the Gaussian mass of a lattice coset "above smoothing", which means that the mass is essentially that of the lattice itself. In particular, we rarely need explicit expressions for either $\mathbf{c}'$ or Σ'.

E.1 Correctness of Algorithm 1, smoothing parameter choice

Proof (of Theorem 1, adapted from [39]). Let $\mathbf{Y}$ be a random variable of distribution $\mathcal{N}_{\Sigma_0\Sigma_0^*}$. If $\mathbf{x} \leftarrow D_{\mathscr{R}^2,\mathbf{B}^{-1}(\mathbf{c}-\mathbf{y}),r^2}$, then by composition $\mathbf{z} := \mathbf{B}\mathbf{x} \leftarrow D_{\mathbf{B}\mathscr{R}^2,\mathbf{c}-\mathbf{y},r^2\mathbf{B}\mathbf{B}^*}$. Denoting by $\texttt{Out}$ the output of Algorithm 1, we then have:

$$\mathbb{P}[\texttt{Out} = \mathbf{z} \wedge \mathbf{Y} = \mathbf{y}] = \frac{\rho_{\Sigma_0\Sigma_0^*}(\mathbf{y})}{\det \Sigma_0} \cdot \frac{\rho_{r^2\mathbf{B}\mathbf{B}^*}(\mathbf{z} - (\mathbf{c} - \mathbf{y}))}{\rho_{r^2\mathbf{B}\mathbf{B}^*}(\mathbf{B}\mathscr{R}^2 - (\mathbf{c} - \mathbf{y}))}.$$

Next recall that $\Sigma = \Sigma_0\Sigma_0^* + r^2\mathbf{B}\mathbf{B}^*$, and use Lemma 6 to obtain the identity

$$\rho_{\Sigma_0\Sigma_0^*}(\mathbf{y})\rho_{r^2\mathbf{B}\mathbf{B}^*}(\mathbf{z} - (\mathbf{c} - \mathbf{y})) = \rho_{\Sigma}(\mathbf{z} - \mathbf{c})\rho_{\Sigma'}(\mathbf{y} - \mathbf{c}'),$$

where the exact expressions for $\mathbf{c}'$ and Σ' are actually not needed. Let $\mathscr{D}(\mathbf{z})$ the probability of Algorithm 1 to output $\mathbf{z}$, and we obtain:

$$\mathscr{D}(\mathbf{z}) = \frac{\rho_{\Sigma}(\mathbf{z}-\mathbf{c})}{\det \Sigma_0} \int_{\mathbb{R}^n} \frac{\rho_{\Sigma'}(\mathbf{y}-\mathbf{c}')}{\rho_{r^2\mathbf{B}\mathbf{B}^*}(\mathbf{B}\mathscr{R}^2 - (\mathbf{c}-\mathbf{y}))}\mathrm{d}\mathbf{y}.$$

Because all matrices here are positive definite over $\mathscr{K}_{\mathbb{R}}$ we also have $\det(\Sigma_0^*) = \det(\Sigma_0)$ (it could have been equal to $(\det \Sigma_0)^*$). Note that this gives $\det(\Sigma') = \det(r\mathbf{B})^2 \det(\Sigma)^{-1} \det(\Sigma_0)^2$. As we are above $\eta_{\varepsilon}(\mathbf{B}\mathscr{R}^2)$ in the denominator of the integral, we get

$$\mathscr{D}(\mathbf{z}) \in \left[1, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \frac{\det(r\mathbf{B})}{(\det \Sigma)^{1/2}} \cdot \frac{\rho_{\Sigma}(\mathbf{z}-\mathbf{c})}{\rho_{r^2\mathbf{B}\mathbf{B}^*}(\mathbf{B}\mathscr{R}^2)},$$

By definition of discrete Gaussians, it is equivalent to

$$\mathscr{D}(\mathbf{z}) \in \left[1, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \alpha \cdot D_{\mathbf{B}\mathscr{R}^2,\mathbf{c},\Sigma}(\mathbf{z}),$$

where we let $\alpha = \frac{\det(r\mathbf{B})}{(\det \Sigma)^{1/2}} \cdot \frac{\rho_{\Sigma}(\mathbf{B}\mathscr{R}^2-\mathbf{c})}{\rho_{r^2\mathbf{B}\mathbf{B}^*}(\mathbf{B}\mathscr{R}^2)}$. Summing both the left-hand side and right-hand side over all possible $\mathbf{z}$'s we see that $\alpha \leqslant 1 \leqslant \frac{1+\varepsilon}{1-\varepsilon}\alpha$, or equivalently, that $\alpha \in [\frac{1-\varepsilon}{1+\varepsilon}, 1]$. We thus obtain

$$\mathscr{D}(\mathbf{z}) \in \left[\frac{1-\varepsilon}{1+\varepsilon}, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot D_{\mathbf{B}\mathscr{R}^2,\mathbf{c},\Sigma}(\mathbf{z}). \tag{5}$$

Using that $\varepsilon \leqslant 1/2$, we see that the statistical distance between $\mathscr{D}$ and $D_{\mathbf{B}\mathscr{R}^2,\mathbf{c},\Sigma}$ is bounded by 2ε, as well as

$$\left|\frac{\mathscr{D}(\mathbf{z})}{D_{\mathbf{B}\mathscr{R}^2,\mathbf{c},\Sigma}(\mathbf{z})} - 1\right| \leqslant 4\varepsilon.$$

Following the discussion in Appendix D, we see that the Rényi divergence of order 2λ is bounded by $32\lambda\varepsilon^2$. To preserve up to $\lambda = 128$, resp. 256 bits of security for $Q = 2^{64}$ queries, it is then enough to set $\varepsilon \leqslant 2^{-39}$, resp. 2^{-40}. This allows us to set accordingly an upper bound on the needed smoothing parameter.

Floating point precision analysis In practice, one can want to use floating point arithmetic to instantiate the Peikert sampler. This means that a sufficiently high precision of computation must be selected to avoid an adversary to distinguish between an "ideal" (infinite precision) sampler and the one that is implemented.

We now turn to the precision analysis of Algorithm 1. To make the analysis simpler, we in fact consider its variant in Algorithm 11, where one checks that $\Sigma_1 = \mathbf{B}^{-1}\Sigma_0$ is a valid choice.

Observe that in practice, $\mathbf{c}$ is usually an integer vector (outputted by some hash function with range in $\mathscr{R}^2$) and since $\mathbf{B} = \mathbf{B}_{f,g}$, $q\mathbf{B}^{-1}$ is in $\mathscr{R}^{2\times 2}$. Hence, we may assume that $\mathbf{B}^{-1}\mathbf{c}$ is known exactly. However Σ_0 and vectors sampled from $\mathcal{N}_{1,\mathscr{K}_{\mathbb{R}}^2}$ have real entries and therefore only approximations of their values can be known. In the statement below, one may think of the technical assumptions as analyzing alternative versions of Algorithm 11 which aborts if both the continuous and discrete Gaussian sampler output a large element. Assuming both these samplers are close to perfect (which can be done in practice), Gaussian tail bounds tell us that the probability that both output large elements can be made smaller than $2^{-\lambda}$, where λ is a target security level. Hence, for suitable parameters and by a hybrid argument, it makes no difference to consider such versions.

Algorithm 11: RingPeikert sampler, variant

Input: A matrix $\mathbf{B} \in \mathcal{K}^{2\times 2}$ such that $\mathcal{L} = \varphi(\mathbf{B}\mathcal{R}^2)$ and a target center $\mathbf{c} \in \mathcal{K}_{\mathbb{R}}^2$.
Result: $\mathbf{z} \in \mathcal{L}$ with distribution negligibly far from $D_{\mathcal{L},\mathbf{c},\Sigma}$.

1 *Precomputed: a parameter* $r \geqslant \eta_\varepsilon(\mathcal{R}^2)$*, and* $\Sigma_1 \in \mathcal{K}_{\mathbb{R}}^{2\times 2}$ *such that* $\Sigma_1\Sigma_1^* = \mathbf{B}^{-1}\Sigma\mathbf{B}^{-*} - r^2$.
2 $\mathbf{y} \leftarrow \Sigma_1 \cdot (\mathcal{N}_{\mathcal{K}_{\mathbb{R}},1})^2$
3 $\mathbf{x} \leftarrow \lceil \mathbf{B}^{-1}\mathbf{c} - \mathbf{y} \rfloor_r$
4 **return** $\mathbf{z} \leftarrow \mathbf{B}\mathbf{x}$

Proposition 1. *Let* $r, \delta, \varepsilon > 0$*, and let* Σ_1 *as in Algorithm 11. For* $\mathbf{u} \in \mathcal{K}_{\mathbb{R}}^2$ *with* $\|\mathbf{u}\| \leqslant 2\sqrt{d}$*, let* $\mathbf{y} = \Sigma_1\mathbf{u}$*. Assume that we are given* $\hat{\mathbf{u}}, \hat{\Sigma}_1$ *and* $\hat{\mathbf{y}} = \hat{\Sigma}_1\hat{\mathbf{u}}$ *satisfying*

- $\|\mathbf{u} - \hat{\mathbf{u}}\| \leqslant \delta \cdot \|\mathbf{u}\|$*;*
- $s_1(\Sigma_1 - \hat{\Sigma}_1) \leqslant \delta \cdot s_1(\Sigma_1)$.
- $\max(\|\mathbf{x} - \mathbf{t}\|, \|\mathbf{x} - \hat{\mathbf{t}}\|) \leqslant 2r\sqrt{\pi d}$*,*

where we let $\mathbf{t} = \mathbf{B}^{-1}\mathbf{c} - \mathbf{y}, \hat{\mathbf{t}} = \mathbf{B}^{-1}\mathbf{c} - \hat{\mathbf{y}}$*. Let* $\Delta := \frac{15d\cdot\delta\cdot s_1(\Sigma_1)}{r} \cdot \left(1 + \frac{\varepsilon}{2(1-\varepsilon)\sqrt{\pi d}}\right)$*, then we have*

$$\exp(-\Delta) \leqslant \frac{D_{\mathcal{R}^2,\mathbf{B}^{-1}\mathbf{c}-\mathbf{y},r}(\mathbf{x})}{D_{\mathcal{R}^2,\mathbf{B}^{-1}\mathbf{c}-\hat{\mathbf{y}},r}(\mathbf{x})} \leqslant \exp(\Delta).$$

We note that requiring a relative error at most δ on each complex embedding of u, that is, $|\varphi_i(u_j) - \varphi_i(\hat{u_j})| \leqslant \delta \cdot |\varphi_i(u_j)|$ for $i \leqslant 2d$ and $j \leqslant 2$, implies the first relative error bound.

Proof. Note that by assumptions we also have $s_1(\hat{\Sigma}) \leqslant (1+\delta)s_1(\Sigma_1)$. Moreover, we have $\mathbf{y} - \hat{\mathbf{y}} = \Sigma_1\mathbf{u} - \hat{\Sigma}_1(\hat{\mathbf{u}} - \mathbf{u}) - \hat{\Sigma}_1\mathbf{u}$, which gives us

$$\begin{aligned}\|\hat{\mathbf{y}} - \mathbf{y}\| &\leqslant s_1(\hat{\Sigma}_1 - \Sigma_1)\|\mathbf{u}\| + s_1(\hat{\Sigma}_1)\|\hat{\mathbf{u}} - \mathbf{u}\| \\ &\leqslant (2+\delta)\delta \cdot s_1(\Sigma_1) \cdot \|\mathbf{u}\|. \end{aligned} \tag{6}$$

Lemma 9 states that

$$\exp\left(\psi(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\leftarrow D_{\mathcal{R}^2,\mathbf{t},r}}[\psi(\mathbf{x})]\right) \leqslant \frac{D_{\mathcal{R}^2,\mathbf{t},r}(\mathbf{x})}{D_{\mathcal{R}^2,\hat{\mathbf{t}},r}(\mathbf{x})} \leqslant \exp\left(\psi(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\leftarrow D_{\mathcal{R}^2,\hat{\mathbf{t}},r}}[\psi(\mathbf{x})]\right).$$

Let us call $A(\mathbf{x})$ the quantity in the left-hand exponential in the inequality above. Thanks to Lemma 9, Lemma 8 and Inequality (6), we have

$$\begin{aligned}|A(\mathbf{x})| &\leqslant \frac{(2+\delta)\delta \cdot s_1(\Sigma_1) \cdot \|\mathbf{u}\|}{r^2} \cdot \left(\|\mathbf{x} - \hat{\mathbf{t}}\| + \frac{r\cdot\varepsilon}{1-\varepsilon}\right) \\ &\leqslant \frac{15 \cdot d \cdot \delta \cdot s_1(\Sigma_1)}{r} \cdot \left(1 + \frac{\varepsilon}{2(1-\varepsilon)\sqrt{\pi d}}\right),\end{aligned}$$

where our assumptions gives that $4\sqrt{\pi}(2+\delta) \leqslant 15$, which is used for the second line. The right-hand side is identical. This gives our claim.

We will now deduce the minimal precision δ needed for our finite precision samplers to be indistinguishable from the ideal one. The assumptions below reflect the practical situation for the NTRU lattices we consider. In the statement below, discrete Gaussian tail bounds tell us that only

an exponentially small portion of $\mathbf{x}$'s are outside Ω. For example, the first sampled vector $\mathbf{u}$ is normal, its norm follows a chi-squared law of dimension $2d$. Hence the probability that $\|\mathbf{u}\| > 2\sqrt{d}$ is less[16] than $\exp(-d/5)$. The constant α is defined in Section 3.4 as the quality of the basis for sampling.

Corollary 1. *Let ε, δ such that $0 \leqslant \max(\varepsilon, \delta) \leqslant 2^{-40}$. Keep the notation of Theorem 1 with $\Sigma_1 = \mathbf{B}^{-1}\Sigma_0$. Let $\Omega = \{\mathbf{x} \in \operatorname{Supp} D_{\mathscr{R}^2,\mathbf{t},r} : \|\mathbf{x}-\hat{\mathbf{t}}\| \leqslant 2r\sqrt{\pi d}\}$, where $r = (1/\pi)\sqrt{(1/2)\log(4d(1+1/\varepsilon))}$. Assume $\varphi(\mathbf{B}\mathscr{R}^2)$ is a lattice of rank $2d \geqslant 512$ with $s_1(\mathbf{B}) < c\sqrt{q}$ for some constants $1 \leqslant \alpha < 16$ and $q > 2^{10}$, and let $\Sigma = (\alpha r)^2 q\mathbf{I}_2 \in \mathscr{K}_{\mathbb{R}}^{2\times 2}$. Let $\mathrm{IP}(\mathbf{x})$ resp. $\mathrm{FP}(\mathbf{x})$ be the probability that the infinite, resp. the finite precision version of Algorithm 11 outputs $\mathbf{Bx}$. Then we have*

$$\sup_{\mathbf{x}\in\Omega}\left|\frac{\mathrm{IP}(\mathbf{x})}{\mathrm{FP}(\mathbf{x})} - 1\right| \leqslant 17d \cdot \alpha^2 \cdot \delta.$$

Proof. We want an upper bound on Δ defined in the previous proposition. Let $s_+(\mathbf{B}^{-1}), s_-(\mathbf{B}^{-1}) \in \mathscr{K}_{\mathbb{R}}$ be the singular values of $\mathbf{B}^{-1}$. They satisfy $s_+(\mathbf{B}^{-1}) \cdot s_-(\mathbf{B}^{-1}) = 1/q$ so for each complex embedding we have $|\varphi_i(s_+(\mathbf{B}^{-1}))| \cdot |\varphi_i(s_-(\mathbf{B}^{-1}))| = 1/q$ as well. By construction, this means that there are d pairs of the singular values of $\varphi(\mathbf{B}^{-1})$ with a product equal to $1/q$. In particular, we deduce that

$$s_1(\mathbf{B}^{-1}) \cdot s_{2d}(\mathbf{B}^{-1}) = s_1(\mathbf{B}^{-1})s_1(\mathbf{B})^{-1} \leqslant \frac{1}{q}.$$

Using properties of the spectral norm of matrices, we then obtain

$$s_1(\Sigma_1) \leqslant s_1(\mathbf{B}^{-1})s_1(\Sigma_0) \leqslant \frac{\alpha \cdot s_1(\Sigma_0)}{\sqrt{q}}.$$

By assumptions, the spectrum of $\varphi(\mathbf{B}\mathbf{B}^*)$ is contained in the interval $(\frac{1}{\alpha^2}, \alpha^2)q$, $\Sigma - r^2\mathbf{B}\mathbf{B}^* = \Sigma_0\Sigma_0^*$ and Σ and $\mathbf{B}\mathbf{B}^*$ commute, so we have $s_1(\Sigma_0) \leqslant \alpha r\sqrt{q}$. With the definition of Δ and our assumptions again, we can now write

$$\Delta \leqslant 16d \cdot \alpha^2 \cdot \delta.$$

The result follows using that $\exp(16d \cdot \alpha^2\delta) \leqslant 1 + 17d \cdot \alpha^2\delta$ for our choice of parameters, Proposition 1 and the definition of the set Ω.

The last corollary follows with Lemma 5.

Corollary 2. *For $\varepsilon \leqslant 2^{-40}$ and $\delta \leqslant 2^{-51-2\log_2(\alpha)}$ (resp. $\delta \leqslant 2^{-52-2\log_2(\alpha)}$), Algorithm 11 in finite precision preserves up to 128 (resp. 256) bits of security if queried less than 2^{64} times over an NTRU lattice $\mathscr{L}(\mathbf{B})$ of rank 1024 (resp. 2048) and with $s_1(\mathbf{B}) = \alpha\sqrt{q}$.*

F Analysis of the hybrid samplers

F.1 Useful results on Gaussian functions

The next lemmata are well-known.

Lemma 7 ([36], implicit in Lemma 4.4). *Let $\mathscr{L}$ be a rank d lattice, and $\Sigma \succ 0$ such that $\sqrt{\Sigma} \geqslant \eta_\varepsilon(\mathscr{L})$. Then we have $\rho_{\mathbf{c},\Sigma}(\mathscr{L}) \in \left[\frac{1-\varepsilon}{1+\varepsilon}; 1\right] \cdot \rho_\Sigma(\mathscr{L})$.*

[16] In practice, it is also less than $2^{-\lambda}$ as $d \geqslant 4\lambda$.

Lemma 8 ([36]). *For any d dimensional lattice $\mathscr{L}$, any $\mathbf{c}$ and unit vector $\mathbf{u}$ in $\mathbb{R}^d$, and for all $0 < \varepsilon < 1$ and $r \geqslant 2\eta_\varepsilon(\mathscr{L})$, we have*

$$\left|\mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{L},\mathbf{c},r}}[\langle \mathbf{x}-\mathbf{c}, \mathbf{u}\rangle]\right| \leqslant \frac{\varepsilon r}{1-\varepsilon}.$$

Gaussian ratios The following results and inequalities can be found in [42], but we rework them for the sake of diffusion.

Lemma 9 ([42], adapted from Lemma 3.10). *Let $k \in \mathbf{N}^*$ and $r > 0$. For fixed $\mathbf{t}, \hat{\mathbf{t}} \in \mathscr{K}_{\mathbb{R}}^k$, let $\psi(\mathbf{x}) := \frac{1}{2r^2}(\|\mathbf{x}-\hat{\mathbf{t}}\|^2 - \|\mathbf{x}-\mathbf{t}\|^2)$. For all $\mathbf{x} \in \mathscr{K}_{\mathbb{R}}^k$, we have $\frac{\rho_r(\mathbf{x}-\mathbf{t})}{\rho_r(\mathbf{x}-\hat{\mathbf{t}})} = \exp(\psi(\mathbf{x}))$, and also:*

$$\exp\big(-\mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\mathbf{t},r}}[\psi(\mathbf{x})]\big) \leqslant \frac{\rho_r(\mathscr{R}^k-\hat{\mathbf{t}})}{\rho_r(\mathscr{R}^k-\mathbf{t})} \leqslant \exp\big(-\mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^2,\hat{\mathbf{t}},r}}[\psi(\mathbf{x})]\big), \tag{7}$$

$$\exp\big(\psi(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\mathbf{t},r}}[\psi(\mathbf{x})]\big) \leqslant \frac{D_{\mathscr{R}^k,\mathbf{t},r}(\mathbf{x})}{D_{\mathscr{R}^k,\hat{\mathbf{t}},r}(\mathbf{x})} \leqslant \exp\big(\psi(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\hat{\mathbf{t}},r}}[\psi(\mathbf{x})]\big). \tag{8}$$

With $\mathbf{v} = \frac{\mathbf{t}-\hat{\mathbf{t}}}{\|\mathbf{t}-\hat{\mathbf{t}}\|}$, we also have

$$\left|\psi(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\mathbf{t},r}}[\psi(\mathbf{x})]\right| \leqslant \frac{\|\hat{\mathbf{t}}-\mathbf{t}\|}{r^2}\cdot\left(\|\mathbf{x}-\mathbf{t}\| + \left|\mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\mathbf{t},r}}[\langle \mathbf{x}-\mathbf{t}, \mathbf{v}\rangle]\right|\right),$$

$$\left|\psi(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\hat{\mathbf{t}},r}}[\psi(\mathbf{x})]\right| \leqslant \frac{\|\hat{\mathbf{t}}-\mathbf{t}\|}{r^2}\cdot\left(\|\mathbf{x}-\hat{\mathbf{t}}\| + \left|\mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\hat{\mathbf{t}},r}}[\langle \mathbf{x}-\hat{\mathbf{t}}, \mathbf{v}\rangle]\right|\right).$$

Proof. The claimed equality amounts to unrolling the definitions. Since $\mathbf{x}, \mathbf{t}, \hat{\mathbf{t}}$ have all their coordinates in $\mathscr{K}_{\mathbb{R}}$, their Hermitian inner products are all real valued, so that $2r^2\psi(\mathbf{x}) = \|\mathbf{t}\|^2 + \|\hat{\mathbf{t}}\|^2 + 2\langle \mathbf{x}, \mathbf{t}-\hat{\mathbf{t}}\rangle$. In particular, ψ is an affine function of $\mathbf{x}$, and as such is convex, as well as the composition $\exp\circ\psi$. The left-hand side of the first inequality comes from

$$\frac{\rho_r(\mathbf{x}-\hat{\mathbf{t}})}{\rho_r(\mathscr{R}^k-\mathbf{t})} = \exp(-\psi(x))D_{\mathscr{R}^k,\mathbf{t},r}(\mathbf{x}),$$

summing over all $\mathbf{x}$'s and using Jensen's inequality. The right-hand side is obtained mutatis mutandis. The discrete Gaussian version of the inequality then amounts to unrolling the definition of the density function conjointly with Inequality (7) to bound the ratio of the total masses over $\mathscr{R}^k$. By linearity of the expectation, we then have

$$\begin{aligned} &r^2\big(\psi(\mathbf{x}) - \mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\mathbf{t},r}}[\psi(\mathbf{x})]\big) \\ &= \langle \mathbf{x}-\mathbf{t}, \mathbf{t}-\hat{\mathbf{t}}\rangle - \|\hat{\mathbf{t}}-\mathbf{t}\|\cdot\mathbb{E}_{\mathbf{x}\leftarrow D_{\mathscr{R}^k,\mathbf{t},r}}\left[\langle \mathbf{x}-\mathbf{t}, \mathbf{v}\rangle\right]. \end{aligned}$$

Using Cauchy-Schwarz inequality, we thus obtain the next claim, and the other one follows by observing that the above can also be unrolled for $\mathbf{x} \leftarrow D_{\mathscr{R}^k,\hat{\mathbf{t}},r}$.

F.2 Correctness of Algorithm 3, smoothing parameter choice

For the sake of readability of the following proof, let us recall some notations of Algorithm 3. The target standard deviation of the algorithm is some $\sigma > 0$, seen (by abuse of notation) as "the constant" $\sigma \in \mathscr{K}_{\mathbb{R}}^{++}$. Steps $4-6$ and $9-11$ are actually explicit Peikert's sampling with target covariance $\Sigma_i = \frac{\sigma^2}{\langle \tilde{\mathbf{b}}_i, \tilde{\mathbf{b}}_i\rangle} \in \mathscr{K}_{\mathbb{R}}^{++}$. When $\sigma \geqslant |\mathbf{B}|_{\mathscr{K}}\cdot\eta_\varepsilon(\mathscr{R}^2)$ as in Theorem 2's assumptions, we let $\sigma_i = \sqrt{\Sigma_i - r^2} \in \mathscr{K}_{\mathbb{R}}^{++}$.

Proof (of Theorem 2, adapted from [42]). Let $\mathscr{D}(\mathbf{v})$ the probability that we obtain $\mathbf{v} = x_1\mathbf{b}_1 + x_2\mathbf{b}_2$ at Step 9, and $P(x_i)$ be the probability that the "Peikert steps" outputs $x_i \in \mathscr{R}$ at Steps $4-6$ and $9-11$ of Algorithm 3. By construction and Identity (5), we see that

$$\mathscr{D}(\mathbf{v}) = P(x_1)P(x_2) \in \left[\left(\frac{1-\varepsilon}{1+\varepsilon}\right)^2, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\right] \cdot D_{\mathscr{R}^2,d_1,\Sigma_1}(x_1)D_{\mathscr{R}^2,d_2,\Sigma_2}(x_2).$$

Our choices for the Σ_i's and the Gram-Schmidt decomposition $\mathbf{B} = \widetilde{\mathbf{B}}\mathbf{U}$ gives us that $\rho_\sigma(\mathbf{v}-\mathbf{c}) = \rho_{\Sigma_1}(x_1-d_1)\rho_{\Sigma_2}(x_2-d_2)$. Note that the sampling covariances Σ_1 and Σ_2 "are above" $\eta_\varepsilon(\mathscr{R})$, so that using the definitions of discrete Gaussians and Lemma 7, we obtain

$$\mathscr{D}(\mathbf{v}) \in \left[\left(\frac{1-\varepsilon}{1+\varepsilon}\right)^2, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^4\right] \cdot \alpha \cdot D_{\mathbf{B}\mathscr{R}^2,\mathbf{c},\Sigma}(\mathbf{v}),$$

where we let $\alpha = \frac{\rho_\Sigma(\mathbf{B}\mathscr{R}^2-\mathbf{c})}{\rho_{\Sigma_1}(\mathscr{R})\rho_{\Sigma_2}(\mathscr{R})}$. Similarly as for Identity (5), we see that $\alpha \in [(\frac{1-\varepsilon}{1+\varepsilon})^4, (\frac{1+\varepsilon}{1-\varepsilon})^2]$, which leads us to

$$\mathscr{D}(\mathbf{v}) \in \left[\left(\frac{1-\varepsilon}{1+\varepsilon}\right)^6, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^6\right] \cdot D_{\mathbf{B}\mathscr{R}^2,\mathbf{c},\Sigma}(\mathbf{v}). \tag{9}$$

We now differ mildly from [42] for the sake of parameter tuning. With our choice for ε, we see that $6(\log(1+\varepsilon)-\log(1-\varepsilon)) \leqslant 13\varepsilon$, and also that $(13\varepsilon)^2 \leqslant \varepsilon$ and $13\varepsilon \leqslant 1/2$, so that $\exp(13\varepsilon)-1 \leqslant 14\varepsilon$. This means that

$$|\mathscr{D}(\mathbf{v}) - D_{\mathbf{B}\mathscr{R}^2,\mathbf{c},\Sigma}(\mathbf{v})| \leqslant 14\varepsilon \cdot D_{\mathbf{B}\mathscr{R}^2,\mathbf{c},\Sigma}(\mathbf{v}),$$

from which we get our claims.

Using the same method for the Peikert sampler, setting $\varepsilon \leqslant 2^{-40}$, resp. 2^{-41} preserves up to $\lambda = 128$, resp. 256 bits of security for $Q = 2^{64}$ queries.

Correctness of Algorithm 4 For the sake of completeness, we analyze Algorithm 4. We assume that we have a perfect routine to sample discrete Gaussians in $\mathscr{R}$ inside Algorithm 4.

Theorem 7. *Let $\mathscr{D}$ be the output distribution of Algorithm 4. If $\varepsilon \leqslant 2^{-5}$ and $r \geqslant \eta_\varepsilon(\mathscr{R})$, then the statistical distance between $\mathscr{D}$ and $D_{\mathscr{L}(\mathbf{U}),\mathbf{c},r}$ is bounded by 3ε. Moreover, we have*

$$\sup_{\mathbf{x}\in\mathbf{U}\mathscr{R}^2} \left|\frac{\mathscr{D}(\mathbf{z})}{D_{\mathscr{L}(\mathbf{U}),\mathbf{c},r}(\mathbf{z})} - 1\right| \leqslant 6\varepsilon.$$

Proof. Since $r \geqslant \eta_\varepsilon(\mathscr{R})$, we have by definition of the discrete Gaussian distribution.

$$\mathscr{D}(\mathbf{a}) = D_{\mathscr{R},c_2,r}(u_2)D_{\mathscr{R},c_1',r}(u_1) \in \left[1, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\right] \frac{\rho_{r,c_2}(u_2)\rho_{r,c_1'}(u_1)}{\rho_r(\mathscr{R}^2)}$$

Calculations give $\rho_{r,c_2}(u_2)\rho_{r,c_1'}(u_1)/\rho_r(\mathscr{R}^2) = D_{\mathbf{U}\mathscr{R}^2,\mathbf{c},r}(\mathbf{a})$. Using that $(1+\varepsilon)^2/(1-\varepsilon)^2 \leqslant 1+6\varepsilon$, we obtain the claim on the relative error. A routine computation gives the claim on the statistical distance.

Floating point precision for Algorithm 3 Since the hybrid sampler relies on the ring version of the Peikert sampler, it is no surprise that the precision analysis is quite similar. As we are interested in the particular case of NTRU lattices, the situation is a bit simpler than in [42]. Indeed, here the different sampling centers are known exactly because in practice the starting center $\mathbf{c}$ and the $\langle \widetilde{\mathbf{b}}_i, \widetilde{\mathbf{b}}_i \rangle$'s are in $\mathscr{K}$. Also, it is enough to target a scalar covariance matrix. Steps 4 to 6 and 9 to 11 are the explicit steps done in the Peikert sampler; in particular, x_2 and x_1 are essentially discrete Gaussians in $\mathscr{R}$ with standard deviation parameter σ.

Again, we implicitly consider versions of the algorithm that abort whenever the continuous Gaussian sampler or the discrete Gaussian sampler within the Peikert sampler output too large elements.

Proposition 2. *Let $r, \delta, \varepsilon > 0$, and keep the notation of Algorithm 3. For $u_1, u_2 \in \mathscr{K}_{\mathbb{R}}$ with $\|u_i\| \leqslant \sqrt{2d}$, let $y_i = \sigma_i u_i$. Assume that we are given $\hat{u}_i, \hat{\sigma}_i$'s and $\hat{y}_i = \hat{\sigma}_i \hat{u}_i$ such that for $i = 1, 2$:*

- $\|u_i - \hat{u}_i\| \leqslant \delta \cdot \|u_i\|$;
- $\|\sigma_i - \hat{\sigma}_i\|_\infty \leqslant \delta \cdot \|\sigma_i\|_\infty$.

Further, let $t_i = d_i - y_i$ and $\hat{t}_i = d_i - \hat{y}_i$. Assume that x_1, x_2 are such that $\max(\|x_i - t_i\|, \|x_i - \hat{t}_i\|) \leqslant r\sqrt{2\pi d}$ for $i = 1, 2$, and let $\Delta := \frac{15d \cdot \delta \cdot \max(\|\sigma_1\|_\infty, \|\sigma_2\|_\infty)}{r} \cdot \left(1 + \frac{\varepsilon}{(1-\varepsilon)\sqrt{2\pi d}}\right)$. Then we have

$$\exp(-\Delta) \leqslant \frac{\mathscr{D}(\mathbf{x})}{\hat{\mathscr{D}}(\mathbf{x})} \leqslant \exp(\Delta),$$

where $\mathscr{D}(\mathbf{x}) = D_{\mathscr{R}, t_2, r}(x_2) D_{\mathscr{R}, t_1, r}(x_1)$, $\hat{\mathscr{D}}(\mathbf{x}) = D_{\mathscr{R}, \hat{t}_2, r}(x_2) D_{\mathscr{R}, \hat{t}_1, r}(x_1)$.

Proof. With our assumptions, we have

$$\begin{aligned} \|y_i - \hat{y}_i\| &\leqslant \|\hat{\sigma}_i - \sigma_i\|_\infty \|u_i\| + \|\sigma_i\|_\infty \|u_i - \hat{u}_i\| \\ &\leqslant \delta(2 + \delta) \|\sigma_i\|_\infty \cdot \|u_i\|. \end{aligned}$$

Then the proof amounts to using twice Lemma 9 with $k = 1$ and our assumptions, in an identical way as in the proof of Proposition 1.

Corollary 3. *Let ε, δ such that $0 \leqslant \max(\varepsilon, \delta) \leqslant 2^{-40}$. Keep the notation of Proposition 2. For $r = (1/\pi)\sqrt{(1/2)\log(4d(1 + 1/\varepsilon))}$, let $\Omega_i = \{x_i \in \operatorname{Supp} D_{\mathscr{R}, t_i, r} : \|x_i - \hat{t}_i\| \leqslant r\sqrt{2\pi d}\}$. Assume $\varphi(\mathbf{B}\mathscr{R}^2)$ is a lattice of rank $2d \geqslant 512$ with $|\mathbf{B}|_{\mathscr{K}} \leqslant \alpha\sqrt{q}$ for some constants $1 \leqslant \alpha < 16$ and $q > 2^{10}$, and let $\sigma = \alpha r \sqrt{q} \in \mathbb{R}$. Let $\mathrm{IH}(x_1, x_2)$ resp. $\mathrm{FH}(x_1, x_2)$ be the probability that the infinite, resp. the finite precision version of Algorithm 3 outputs $\mathbf{v} = \mathbf{B}(x_1, x_2)$. Then we have*

$$\sup_{(x_1, x_2) \in \Omega_1 \times \Omega_2} \left| \frac{\mathrm{IH}(x_1, x_2)}{\mathrm{FH}(x_1, x_2)} - 1 \right| \leqslant 17d \cdot \alpha^2 \cdot \delta.$$

Proof. By construction and because $q^2 = \langle \widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_1 \rangle \langle \widetilde{\mathbf{b}}_2, \widetilde{\mathbf{b}} \rangle_2$, we have $\|\sigma_i\|_\infty^2 \leqslant \frac{\sigma^2}{q^2} |\mathbf{B}|_{\mathscr{K}}^2 - r^2 \leqslant \alpha^4 r^2$. With the definition of Δ and our assumptions again, we can now write

$$\Delta \leqslant 16d \cdot \alpha^2 \cdot \delta,$$

and we conclude as in Corollary 1.

Corollary 4. *For $\varepsilon \leqslant 2^{-40}$ (resp. $\varepsilon \leqslant 2^{-41}$) and $\delta \leqslant 2^{-50 - 2\log_2(\alpha)}$ (resp. $\delta \leqslant 2^{-52 - 2\log_2(\alpha)}$), Algorithm 11 in finite precision preserves up to 128 (resp. 256) bits of security if queried less than 2^{64} times over an NTRU lattice $\mathscr{L}(\mathbf{B})$ of rank 1024 (resp. 2048) and with $|\mathbf{B}|_{\mathscr{K}} = \alpha\sqrt{q}$.*

F.3 Analysis of the integer friendly sampler

Correctness of Algorithm 6

Lemma 10 ([12], adapted). *Assume that $\mathbf{A} \in \mathscr{R}^{2\times m}$ is such that $\mathbf{A}\mathbf{A}^t = p^2(\Sigma - \mathbf{I})$, and let $\varepsilon \leqslant 2^{-5}$ Let $\mathscr{D}$ be the probability distribution of outputs from Algorithm 6. If $r\sqrt{\Sigma} \geqslant \eta_\varepsilon(\mathscr{R}^2)$ and $r\sqrt{\mathbf{I} - \Sigma^{-1}} \geqslant \eta_\varepsilon(\mathscr{R}^2)$, then the statistical distance between $\mathscr{D}$ and $D_{\mathscr{R}^2,0,r^2\Sigma}$ is bounded by 5ε. Moreover, we have*

$$\sup_{\mathbf{p}\in\mathscr{R}^2} \left| \frac{\mathscr{D}(\mathbf{p})}{D_{\mathscr{R}^2,r^2\Sigma}(\mathbf{p})} - 1 \right| \leqslant 10\varepsilon.$$

Proof. Inspecting the proof of [12], we see that the probability to obtain $\mathbf{p}'$ at Step 3 is in $[\frac{1-\varepsilon}{1+\varepsilon}, \frac{1+\varepsilon}{1-\varepsilon}] \cdot D_{(1/pL)\mathscr{R}^2,0,r^2(\Sigma-\mathbf{I})}(\mathbf{p}')$. This means that the probability $\mathscr{D}(\mathbf{p})$ to output $\mathbf{p}$ satisfies

$$\mathscr{D}(\mathbf{p}) \in \left[\frac{1-\varepsilon}{1+\varepsilon}, \frac{1+\varepsilon}{1-\varepsilon}\right] \cdot \sum_{\mathbf{p}'\in\mathscr{R}^2} D_{(1/pL)\mathscr{R}^2,0,r^2(\Sigma-\mathbf{I})}(\mathbf{p}') D_{\mathscr{R}^2,\mathbf{p}',r^2}(\mathbf{p}).$$

By Lemma 6 one gets that $\mathscr{D}(\mathbf{p})$ is proportional to a quantity in the interval

$$[\frac{1-\varepsilon}{1+\varepsilon}, \frac{1+\varepsilon}{1-\varepsilon}] \cdot \sum_{\mathbf{p}'\in\mathscr{R}^2} \rho_{r^2\Sigma}(\mathbf{p}) \rho_{\Sigma'}(\mathbf{p}' - \Sigma'\mathbf{p} - \mathbf{p}),$$

where an exact expression of Σ' is not needed for the rest of the proof. Using that both $r\sqrt{\Sigma}$ and $\sqrt{\Sigma'}$ are greater than $\eta_\varepsilon(\mathscr{R}^2)$, and handling proportionality constants as in the previous sections, we may rewrite with Lemma 7 that

$$\mathscr{D}(\mathbf{p}) \in \left[\left(\frac{1-\varepsilon}{1+\varepsilon}\right)^4, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^4\right] \cdot D_{\mathscr{R}^2,r^2\Sigma}(\mathbf{p}). \tag{10}$$

Our claims follow.

Correctness of Algorithm 5

Proof. We first show that our assumption on s implies that $\Sigma_p - \mathbf{I} = (s^2-1)\mathbf{I} - \widehat{\mathbf{B}}\widehat{\mathbf{B}}^t$ is positive-definite. Thanks to Lemma 1, note that $|\mathbf{B}_{f,g}|_{\mathscr{K}} = \|\widetilde{\mathbf{B}}\|_2$ so that we have $s_1(\widehat{\mathbf{B}}) = \|\widehat{\mathbf{B}}\|_2 \leqslant \|\widetilde{\mathbf{B}}\|_2\|\mathbf{V}\|_2 = |\mathbf{B}_{f,g}|_{\mathscr{K}} s_1(\mathbf{V})$, where $\mathbf{V} = \mathbf{U}_{\hat{u}-u}$. Let $v = \hat{u} - u$, so that the positive definite matrix $\mathbf{V}^*\mathbf{V} \in \mathscr{K}^{2\times 2}$ has for characteristic polynomial $\chi = X^2 - (2 + vv^*)X + 1$, with totally positive discriminant $\Delta = vv^*(vv^* + 4)$. Let $\lambda_+ \succ \lambda_- \in \mathscr{K}_{\mathbb{R}}^{++}$ its two eigenvalues, and we find that $\lambda_+ = 1 + (vv^* + \sqrt{\Delta})/2$. The coefficient of v (as a polynomial) are between $-1/(2p)$ and $1/(2p)$, so that $\|\varphi(v)\|_\infty \leqslant d/(2p)$. Using $\|\varphi(v)\|_\infty^2 = \|\varphi(vv^*)\|_\infty$, one finds the upper bounds $\|\sqrt{\Delta}\|_\infty \leqslant \|\varphi(v)\|_\infty\sqrt{4 + \|\varphi(v)\|_\infty^2}$ and $\|\varphi(vv^*)\|_\infty \leqslant d^2/(4p^2)$. Using $\sqrt{1+a^2} \leqslant 1 + a$ and $p > d$, this gives us

$$\begin{aligned} s_1(\mathbf{V}) = \sqrt{\|\lambda_+\|_\infty} &\leqslant 1 + \frac{d}{2p}\sqrt{1 + \sqrt{1 + \frac{16p^2}{d^2}}} \\ &\leqslant 1 + \frac{d}{2p}\sqrt{2 + \frac{4p}{d}} \\ &\leqslant 1 + \sqrt{\frac{2d}{p}}, \end{aligned}$$

Table 5. Parameters for Algorithm 5, for the power-of-two cyclotomic case.

d	ε	r	$\alpha\sqrt{q}$	s'	s	p	b	L
512	2^{-43}	1.39	227.3	233	256	2^{21}	8192	2^{35}
1024	2^{-43}	1.39	260.5	265	292	2^{23}	16384	2^{35}

so that $\Sigma_p - \mathbf{I}$ is indeed positive definite.

Let $\mathscr{D}(\mathbf{z})$ the probability that Algorithm 5 outputs a vector $\mathbf{z}$, and $\textsc{Off}(\mathbf{p})$ that the offline sampling outputs $\mathbf{p}$. We will again make use of Lemma 6. For the sake of clarity, we now explicit the parameters that will be used. Let $\Sigma_1 = r^2\widehat{\mathbf{B}}\widehat{\mathbf{B}}^t$, $\Sigma_2 = r^2\Sigma_p$ and $\Sigma' = \Sigma_2(\Sigma_1 + \Sigma_2)^{-1}\Sigma_1$. One checks that $\Sigma'^{-1} = \Sigma_1^{-1} + \Sigma_2^{-1} = (1/r^2)((\widehat{\mathbf{B}}\widehat{\mathbf{B}}^t)^{-1} + (s^2\mathbf{I} - \widehat{\mathbf{B}}\widehat{\mathbf{B}}^t)^{-1})$. Combining our analysis of Algorithm 4 with Equation (10), we have

$$\begin{aligned}\mathscr{D}(\mathbf{z}) &\in \left[\left(\frac{1-\varepsilon}{1+\varepsilon}\right)^2, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^2\right] \sum_{\mathbf{p}} \textsc{Off}(\mathbf{p}) \cdot D_{\mathbf{U}_{-\hat{u}}\mathscr{R}^2, \hat{\mathbf{c}}, r}(\mathbf{z}') \\ &\in \left[\left(\frac{1-\varepsilon}{1+\varepsilon}\right)^6, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^6\right] \sum_{\mathbf{p}} D_{\mathscr{R}^2, \Sigma_2}(\mathbf{p}) \cdot D_{\mathbf{B}\mathscr{R}^2, \mathbf{c}-\mathbf{p}, \Sigma_1}(\mathbf{z}).\end{aligned}$$

As $\det(\widehat{\mathbf{B}}\widehat{\mathbf{B}}^t)^{-1} = q^{-2d}$, we see that $(\widehat{\mathbf{B}}\widehat{\mathbf{B}}^t)^{-1} \prec (1/q)\mathbf{I}$. With our choice of s, we also have $\Sigma_p^{-1} \prec (1/4)\mathbf{I}$. We obtain that $\sqrt{\Sigma'} \geqslant \eta_\varepsilon(\mathscr{R}^2)$, and $r\widehat{\mathbf{B}} \geqslant \eta_\varepsilon(\mathscr{L}(\mathbf{B}))$ as well thanks to $\widehat{\mathbf{B}}^{-1}\mathbf{B}_{f,g}$ being an upper triangular matrix. Hence, using Lemma 6 and Lemma 7, we find that $\mathscr{D}(\mathbf{z})$ is proportional to a quantity in $[(\frac{1-\varepsilon}{1+\varepsilon})^7, (\frac{1+\varepsilon}{1-\varepsilon})^7] \cdot \rho_{(rs)^2}(\mathbf{c} - \mathbf{z})$. Dealing with the constant, one finds for all $\mathbf{z} \in \mathscr{R}^2$ that

$$\mathscr{D}(\mathbf{z}) \in \left[\left(\frac{1-\varepsilon}{1+\varepsilon}\right)^{14}, \left(\frac{1+\varepsilon}{1-\varepsilon}\right)^{14}\right] \cdot D_{\mathscr{L}(\mathbf{B}), \mathbf{c}, rs}(\mathbf{z}).$$

Our claims follow from the same routine computations as usual.

With the discussion of Appendix D, we preserve λ bits of security if $2\lambda(30\varepsilon)^2 \leqslant 2^{-66}$. In particular, setting $\varepsilon = 2^{-43}$ preserves both 128 and 256 bits of security in Algorithm 5

Parameter analysis for Gram root decomposition in the NTRU case It is worth recalling some notation. Recall that $\widehat{\mathbf{B}} = \mathbf{B}_{f,g}\mathbf{U}_{\hat{u}}$, and fix some integer $p \geqslant 6d$. Let $s' \geqslant |\mathbf{B}_{f,g}|_{\mathscr{K}}(1+\sqrt{2d/p}) \geqslant s_1(\widehat{\mathbf{B}})$ and let $B = ps'$. Note that $|\mathbf{B}_{f,g}|_{\mathscr{K}} = \alpha\sqrt{q}$, where α is deduced from Table 1. To set parameters, we allow ourselves some slack $s = 1.1s'$ for the target standard deviation. From the requirements of [12, Algorithm 6] using their eigenvalue reduction technique, the integral Gram root is correctly computed for integers b (the gadget decomposition base), p such that

$$\begin{aligned}b^3 &\geqslant \lceil B\sqrt{d(d+1)} + \frac{d(d+1)}{8}\rceil + 3(2d-1)b^2 \\ p^2(s^2-1) &\geqslant (b^4 + b^2 + 1)\log_2(d) + b^3 + B^2 + 1.\end{aligned}$$

This leads to the parameters in Table 5.

G On the security of MITAKA

In all of the following, we follow the so-called *Geometric series assumption* (GSA), asserting that a reduced basis sees its Gram-Schmidt vectors' norm decrease with geometric decay. More formally, it can be instantiated as follows for self-dual BKZ (DBKZ) reduction algorithm of Micciancio and Walter [37]: an output basis $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ yielded by DBKZ algorithm with block size β on a lattice $\mathscr{L}$ of rank n satisfies

$$\|\widetilde{\mathbf{b}}_i\| = \delta_\beta^{n-2(i-1)} \operatorname{covol}(\mathscr{L})^{\frac{1}{n}}, \quad \text{where} \quad \delta_\beta = \left(\frac{(\pi\beta)^{\frac{1}{\beta}} \cdot \beta}{2\pi e}\right)^{\frac{1}{2(\beta-1)}}.$$

G.1 Key recovery attack

The key recovery consists in finding the private secret key (i.e. $f, g \in \mathcal{R}^2$) from the sole data of the public elements q and h. The most powerful attacks are up-to-our-knowledge realized through lattice reduction. It consists in constructing the algebraic lattice over $\mathcal{R}$ spanned by the vectors $(q, 0)$ and $(h, 1)$ (i.e. the public basis of the NTRU key) and retrieve the lattice vector $\mathbf{s} = (\mathbf{g}, \mathbf{f})$ among all possible lattice vectors of norm bounded by $\|\mathbf{s}\| = \sqrt{2n}\sigma$. We make use of the so-called *projection trick* to avoid enumerating over all this sphere. More precisely we proceed as follows. Set β to be the block size parameter of the DBKZ algorithm and start by reducing the public basis with this latter algorithm. Call $[\mathbf{b}_1, \ldots, \mathbf{b}_{2n}]$ the resulting vectors. Then if we can recover the *projection* of the secret key onto $\mathcal{P}$, the orthogonal space to $\mathrm{Span}(\mathbf{b}_1, \ldots, \mathbf{b}_{2n-\beta-1})$, then we can retrieve in polynomial time the full key by *Babai nearest plane* algorithm to lift it to a lattice vector of the desired norm. Hence it suffices to be able find the projection of the secret key among the shortest vector of the lattice generated by the last β vectors projected onto $\mathcal{P}$. Classically, sieving on this projected lattice will recover all vectors of norm smaller than $\sqrt{\frac{4}{3}}\ell$, where ℓ is the norm of the $2n - \beta$-th Gram-Schmidt vector $\widetilde{b}_{2m-\beta}$ of the reduced basis. Under the GSA assumption we have:

$$\ell = \sqrt{q}\delta_\beta^{-2n+2\beta+2} \approx \left(\frac{\beta}{2\pi e}\right)^{1-\frac{n}{\beta}}.$$

Moreover, considering that $\mathbf{s}$ behaves as a random vector of norm $\sqrt{2n}\sigma$, and using the GSA to bound the norm of the Gram-Schmidt vectors $[\widetilde{\mathbf{b}}_1, \ldots, \widetilde{\mathbf{b}}_{2n-\beta}]$, that the norm of its projection over $\mathcal{P}$ is roughly

$$\sqrt{\frac{\beta}{2n}}\|\mathbf{s}\| = \beta^{\frac{1}{2}}\sigma.$$

Hence, we will retrieve the projection among the sieved vectors if $\beta^{\frac{1}{2}}\sigma \leqslant \sqrt{\frac{4}{3}}\ell$, that is if the following condition is fulfilled:

$$\sigma^2 \leqslant \frac{4q}{3\beta}\delta_\beta^{4(\beta+1-n)} \tag{11}$$

G.2 Signature forgery by ApproxCVP reduction.

As a Hash-and-Sign paradigm signature, forging a signature stems to feeding a lattice point $\mathbf{v}$ at a bounded distance from a random space point $\mathbf{x}$. This ApproxCVP problem can be solved using the so-called *Nearest-Cospace* framework developed in [17]. Under the Geometric Series assumption,

Theorem 3.3 of [17] states that under the condition: $\|\mathbf{x}-\mathbf{v}\| \leqslant \left(\delta_\beta^{2n} q^{\frac{1}{2}}\right)$, the decoding can be done in time $\mathrm{Poly}(n)$ calls to a CVP oracle in dimension β.

As mentioned in [8] a standard optimization of this attack consists only considering the lattice spanned by a subset of the vectors of the public basis and perform the decoding within this sublattice. The only interesting subset seems to consists in forgetting the $k \leqslant n$ first vectors. The dimension is of course reduced by k, at the cost of working with a lattice with covolume $q^{\frac{k}{2(2n-k)}}$ bigger. Henceforth the global condition of decoding becomes the (slightly more general) inequality $\|\mathbf{x}-\mathbf{v}\| \leqslant \min_{k \leqslant n} \left(\delta_\beta^{2n-k} q^{\frac{n}{2n-k}}\right)$ As such, we need to enforce the condtion:

$$\gamma \geqslant \min_{k \leqslant n} \left(\delta_\beta^{2n-k} q^{\frac{n}{2n-k}}\right) \tag{12}$$

G.3 On the other attacks on MITAKA

In this section, we list the other possible type of attacks on the signature, which are nonetheless irrelevant for the set of parameters we are using.

G.3.1 Algebraic attacks As remarked in the design of NTRU-based schemes (such as for instance FALCON or MODFALCON signatures), there exists a rich algebraic structure in the modules over the convolution ring $\mathcal{R}$ used in MITAKA. However, there is no known way to improve all the algorithms previously mentioned with respect to their general lattice equivalent by more than polynomial factors (see for instance the speedup on lattice reduction of [28]).

G.3.2 Overstretched NTRU-type As observed in [29], when the modulus q is significantly larger than the magnitudes of the NTRU secret key coefficients, the attack on the key based on lattice reduction recovers the secret key better than the results presented above. This so-called "overstretched NTRU" parameters occurs when $q > n^{2.83}$ for binary secrets, implying that, as it is the case for Falcon and other NTRU based NIST candidates, that even *very* significant improvements of this attack would still be irrelevant for the security of the scheme.

G.3.3 Hybrid attacks Odlyzko's meet on the middle attack, or more recently the hybrid attack of Howgrave-Graham [25] which combines a meet-in-the-middle algorithm with a key recovery by lattice reduction were used effectively against NTRU, mainly due to its design using sparse polynomials. As it is not the case (secrets are dense elements in the ring $\mathcal{R}$), their impact is not sufficient to be a problem on the parameter selection of MITAKA.

H More explicit steps to compute samplers

See Algorithm 12 and 13 for more explicit procedures to compute the RingPeikert and Hybrid samplers, respectively. Note that sampling from a *continuous* Gaussian distribution can be preprocessed since these operations are independent of the input center $\mathbf{c}$. We also specify where the FFT or NTT-based polynomial multiplications happen.

Algorithm 12: RingPeikert with explicit FFT/NTT

Input: A target center $\mathbf{c} \in \mathscr{K}_{\mathbb{R}}^2$; a matrix $\mathbf{B} \in \mathscr{K}^{2\times 2}$ such that $\mathscr{L} = \varphi(\mathbf{B}\mathscr{R}^2)$; a parameter $r \geqslant \eta_\varepsilon(\mathscr{R}^2)$; canonical embeddings $\widehat{\mathbf{B}} = \mathrm{FFT}(\mathbf{B})$, $\widehat{\mathbf{B}}^{-1} = \mathrm{FFT}(\mathbf{B}^{-1})$ and $\widehat{\Sigma}_0 = \mathrm{FFT}(\Sigma_0)$, where $\Sigma_0 \in \mathscr{K}_{\mathbb{R}}^{2\times 2}$ is precomputed such that $\Sigma_0\Sigma_0^* = \Sigma - r^2\mathbf{B}\mathbf{B}^*$; NTT representation $\mathbf{B}' = \mathsf{NTT}(\mathbf{B})$.
Result: $\mathbf{z} \in \mathscr{L}$ with distribution negligibly far from $D_{\mathscr{L},\mathbf{c},\Sigma}$.

Offline
1 **for** $i \in [0, d-1]$ **do**
2 $\quad y_{1,i} \leftarrow \mathcal{N}_{1/\sqrt{d}}$
3 $\quad y_{2,i} \leftarrow \mathcal{N}_{1/\sqrt{d}}$
4 **end for**
5 $\mathbf{y} := (y_{1,0}, \ldots, y_{1,d-1}, y_{2,0}, \ldots, y_{2,d-1})$
6 $\hat{\mathbf{y}} \leftarrow \mathrm{FFT}(\mathbf{y})$
7 $\hat{\mathbf{x}} \leftarrow \widehat{\Sigma}_0 \odot \hat{\mathbf{y}}$
Online
8 $\hat{\mathbf{c}} \leftarrow \mathrm{FFT}(\mathbf{c})$
9 $\mathbf{z} \leftarrow \lfloor \mathrm{iFFT}(\widehat{\mathbf{B}}^{-1} \odot (\hat{\mathbf{c}} - \hat{\mathbf{x}})) \rceil_r$
10 $\mathbf{z}' \leftarrow \mathsf{NTT}(\mathbf{z})$
11 **return** $\mathsf{iNTT}(\mathbf{B}' \odot \mathbf{z}' \mod q)$

I Masking

I.1 Extra masked algorithms

We present the masked online and offline parts of Algorithm 10 in Algorithms 14 and 15. The masked polynomial multiplication routine SecNTTMult is also described in Algorithm 16.

I.2 Proof of Theorem 4

Proof. Only a proof for each line of Algorithm 10 is necessary. Line 2 is a linear operation thus it is t-NI. Lines 3 and 5 are proved t-NI in 7.2. Line 6 does not manipulate any sensitive value. So it remains to prove that the algorithms MaskedOnlineSampling (Alg. 14) and MaskedOfflineSampling (Alg. 15) are t-NI. First, Alg. 15 is directly t-NI because it is a linear succession of t-NI gadgets. Secondly, let us prove the t-NI security of Alg. 14. We consider that the attacker made $\delta \leq t$ observations during the execution of MaskedOnlineSampling. In the following, we prove that all these δ observations can be perfectly simulated with at most δ shares of $[\![\widetilde{v}]\!]$, $[\![c_1]\!]$ and $[\![c_2]\!]$.
We consider the following distribution of the attacker's δ observations: δ_1 made during the first call to GaussShareByShare, δ_2 made during the call to SecNTTMult, δ_3 made during the subtraction, δ_4 made during the second call to GaussShareByShare, and δ_5 made during the final addition. We have

$$\sum_{i=1}^{5} \delta_i \leq \delta \leq t.$$

We build the proof classically from right to left. Since the addition is linear with respect to the arithmetic masking type, the final step is t-NI. It is also an affine gadget. In other words, each observation can be simulated with exactly either one share of $[\![x]\!]$ or one share of $[\![u_1]\!]$. Thus, all the observations from its call can be simulated with at most δ_5 shares among all the shares of $[\![x]\!]$ and $[\![u_1]\!]$. More precisely, all the observations from its call can be simulated with δ_5^1 shares of

Algorithm 13: Hybrid Gaussian sampler with explicit FFT/NTT

Input: A target center $\mathbf{c} \in \mathscr{K}_{\mathbb{R}}^2$; a matrix $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2]$ such that $\mathscr{L} = \varphi(\mathbf{B}\mathscr{R}^2)$ and its GSO $[\widetilde{\mathbf{b}_1}, \widetilde{\mathbf{b}_2}]$ over $\mathscr{K}$; a parameter $r \geqslant \eta_\varepsilon(\mathscr{R})$; a parameter $\sigma > 0$ (corresponding to $(\sigma, \ldots, \sigma) \in \mathscr{K}_{\mathbb{R}}$); $\sigma_i := \sqrt{\frac{\sigma^2}{\langle \widetilde{\mathbf{b}_i}, \widetilde{\mathbf{b}_i} \rangle} - r^2} \in \mathscr{K}_{\mathbb{R}}^{++}$; precomputed canonical embeddings $\hat{\sigma}_i = \mathrm{FFT}(\sigma_i)$, $\hat{\mathbf{b}}_i = \mathrm{FFT}(\mathbf{b}_i)$, and $\hat{\boldsymbol{\beta}}_i = \mathrm{FFT}(\frac{\widetilde{\mathbf{b}}_i^*}{\langle \widetilde{\mathbf{b}_i}, \widetilde{\mathbf{b}_i} \rangle_{\mathscr{K}}})$ for $i = 1, 2$.

Result: $\mathbf{z}$ with distribution negligibly far from $D_{\mathscr{L}, \mathbf{c}, \sigma^2 \mathbf{I}_{2d}}$.

Offline

for $i = 1, 2$ **do**
  for $j \in [0, d-1]$ **do**
    $u_{i,j} \leftarrow \mathcal{N}_1$
  end for
  $u_i := (u_{i,0}, \ldots, u_{i,d-1})$
  $\hat{u}_i \leftarrow \mathrm{FFT}(u_i)$ `/* Can be omitted by scaling the` $\mathcal{N}$`. */`
  $\hat{y}_i \leftarrow \hat{\sigma}_i \odot \hat{u}_i$
end for

Online

`/* first nearest plane */`
$\hat{\mathbf{c}}_2 \leftarrow \mathrm{FFT}(\mathbf{c}), \hat{\mathbf{v}}_2 \leftarrow \mathbf{0}$
$\hat{d}_2 \leftarrow \hat{\beta}_{2,1} \odot \hat{c}_{2,1} + \hat{\beta}_{2,2} \odot \hat{c}_{2,2}$
$x_2 \leftarrow \lfloor \mathrm{iFFT}(\hat{d}_2 - \hat{y}_2) \rceil_r$
`/* second nearest plane */`
$\hat{x}_2 \leftarrow \mathrm{FFT}(x_2)$
$\hat{\mathbf{v}}_1 \leftarrow \hat{x}_2 \odot \hat{\mathbf{b}}_2$
$\hat{\mathbf{c}}_1 \leftarrow \hat{\mathbf{c}}_2 - \hat{\mathbf{v}}_1$
$\hat{d}_1 \leftarrow \hat{\beta}_{1,1} \odot \hat{c}_{1,1} + \hat{\beta}_{1,2} \odot \hat{c}_{1,2}$
$x_1 \leftarrow \lfloor \mathrm{iFFT}(\hat{d}_1 - \hat{y}_1) \rceil_r$
$\hat{x}_1 \leftarrow \mathrm{FFT}(x_1)$
$\hat{\mathbf{v}}_0 \leftarrow \hat{\mathbf{v}}_1 + \hat{x}_1 \odot \hat{\mathbf{b}}_1$
return $\mathrm{iFFT}(\mathbf{v}_0) \mod q$

$[\![x]\!]$ and δ_5^2 shares of $[\![u_1]\!]$ such that $\delta_5^1 + \delta_5^2 = \delta_5$. By Table 4 properties, GaussShareByShare is t-NI. Hence, all the observations from its call can be simulated with $\delta_5^1 + \delta_4$ shares of $[\![c_1']\!]$ and δ_5^2 shared of $[\![x]\!]$. Next, the subtraction is also a linear operation (we can ignore its affine property here). Thus, all the observations from its call can be simulated with $\delta_5^1 + \delta_4 + \delta_3$ shares of $[\![c_1]\!]$ and $\delta_5^1 + \delta_5^2 + \delta_4 + \delta_3 = \delta_5 + \delta_4 + \delta_3$ shares of $[\![x]\!]$. Still by Table 4 properties, SecNTTMult is t-NI. Hence, the observations from its call can be simulated with $\delta_5 + \delta_4 + \delta_3 + \delta_2$ shares of $[\![u_2']\!]$ and $[\![\widetilde{v}]\!]$ (and still $\delta_5^1 + \delta_4 + \delta_3$ shares of $[\![c_1]\!]$). Finally, by the t-NI property of GaussShareByShare, the observations from the whole algorithm can be simulated with

- $\delta_5 + \delta_4 + \delta_3 + \delta_2 + \delta_1 \leq t$ shares of $[\![c_2]\!]$;
- $\delta_5 + \delta_4 + \delta_3 + \delta_2 \leq t$ shares of $[\![\widetilde{v}]\!]$;
- $\delta_5^1 + \delta_4 + \delta_3 \leq \delta_5 + \delta_4 + \delta_3 \leq t$ shares of $[\![c_1]\!]$;

which concludes the proof.

I.3 Masking Peikert's sampler

See Alg. 17 for our masked version of the RingPeikert sampler (see Alg. 1 and 12). Although it relies on MaskedCDT, one could instantiate the offline sampling in a different manner, depending

Algorithm 14: MaskedOnlineSampling

Input: Two arithmetically masked mod $\mathsf{Q}^{\mathrm{mask}}$ values $[\![\widetilde{v}]\!]$ and $[\![\mathbf{c}^{\mathrm{pert}}]\!] = \begin{pmatrix}[\![c_1]\!]\\ [\![c_2]\!]\end{pmatrix}$.

Result: An arithmetically masked $[\![\mathbf{u}]\!]$.

1 $[\![u_2']\!] \leftarrow \mathsf{GaussShareByShare}([\![c_2]\!])$
2 $[\![x]\!] \leftarrow \mathsf{SecNTTMult}([\![u_2']\!], [\![\widetilde{v}]\!])$
3 $[\![c_1']\!] \leftarrow [\![c_1]\!] - [\![x]\!]$
4 $[\![u_1']\!] \leftarrow \mathsf{GaussShareByShare}([\![c_1']\!])$
5 $[\![\mathbf{u}]\!] \leftarrow \begin{pmatrix}[\![u_1']\!] + [\![x]\!]\\ [\![u_2']\!]\end{pmatrix}$
6 **return** $[\![\mathbf{u}]\!]$

Algorithm 15: MaskedOfflineSampling

Input: An arithmetically masked mod $\mathsf{Q}^{\mathrm{mask}}$ matrix $[\![A]\!]$.
Result: An arithmetically masked $[\![\mathbf{p}]\!]$ for $\mathbf{p} \in \mathscr{R}^2$.

1 $[\![\mathbf{p}']\!] \leftarrow \mathsf{MaskedCDT}_{Lr,0}$
2 $[\![\mathbf{p}']\!] \leftarrow \frac{1}{pL}\mathsf{SecNTTMult}([\![\vec{A}]\!], [\![\mathbf{p}']\!])$
3 $[\![\mathbf{p}]\!] \leftarrow \mathsf{GaussShareByShare}(r, [\![\mathbf{p}']\!])$
4 **return** $[\![\mathbf{p}]\!]$

on the required precision of samples. One plausible option would be to employ a secure gadget computing the Box–Muller transform [7]. Since masking the Box–Muller transform amounts to evaluating a few non-linear functions (i.e. $\cos, \sin, \log, \sqrt{\cdot}$) on random Boolean shares representing values in $\mathbb{R} \cap [0, 1]$, one could easily achieve such a gadget using the existing polynomial approximation techniques [4]. The same remark applies to the MaskedHybrid sampler.

I.3.1 Security Proof

Theorem 8. *Assuming t-NI security of* GaussShareByShare, SecNTTMult *and* MaskedCDT*, and t-NIo security of* Unmask*, the* MaskedRingPeikert *sampler (Alg. 17) is t-NIo secure with public output* $\mathbf{z}$.

Proof. Let us assume that an attacker has access to $\delta \leqslant t$ observations on the whole sampler. Our goal is to prove that all these δ observations can be perfectly simulated with at most δ shares of each secret among $[\![\mathbf{B}]\!]_q$, $[\![q^k\mathbf{B}^{-1}]\!]$, and $[\![q^{k_2}\Sigma_1]\!]$. We consider an attacker who peeks at internal computations as follows: δ_1 observations during line 15; δ_2 observations during line 14; δ_3 observations during line 9-12; δ_4 observations during line 8; δ_5 observations during line 7; δ_6 observations during line 6; δ_7 observations during line 1-4. Suppose $\sum_i \delta_i \leqslant \delta$.

- Since Unmask is t-NIo secure with public output $\mathbf{z}$, all the observations at line 15 can be simulated with at most δ_1 shares of $[\![\mathbf{z}]\!]_q$ and $\mathbf{z}$.
- Since SecNTTMult is t-NI secure, all the observations at line 14 can be simulated with at most $\delta_1 + \delta_2$ shares of $[\![\mathbf{z}]\!]_q$ and $[\![\mathbf{B}]\!]_q$.
- Since line 9-12 consists of independent local operations on each share of $[\![q^k\mathbf{v}]\!]$ and GaussShareByShare is t-NI secure, all the observations during line 9-12 can be simulated with at most $\delta_1 + \delta_2 + \delta_3$ shares of $[\![q^k\mathbf{v}]\!]$.

Algorithm 16: SecNTTMult

Input: Arithmetic maskings $[\![a]\!]$ and $[\![b]\!]$ of $a, b \in \mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}[x]/(x^d+1)$.
Result: An arithmetic masking $[\![c]\!]$ of $c \in \mathbb{Z}_{\mathsf{Q}^{\mathrm{mask}}}[x]/(x^d+1)$ such that $c = a \cdot b$.

1 $[\![\hat{a}]\!] \leftarrow \mathsf{NTT}([\![a]\!])$ /* Apply NTT on each share independently */
2 $[\![\hat{b}]\!] \leftarrow \mathsf{NTT}([\![b]\!])$
3 **for** $j := 1$ *to* d **do**
4 $\quad [\![\hat{c}]\!] \leftarrow \mathsf{SecMult}([\![\hat{a}[j]]\!], [\![\hat{b}[j]]\!])$
5 **end for**
6 $[\![c]\!] \leftarrow \mathsf{NTT}^{-1}([\![\hat{c}]\!])$
7 **return** $[\![c]\!]$

- Since line 8 consists of linear operations on input shares, all the observations during line 8 can be simulated with at most $\delta_1 + \delta_2 + \delta_3 + \delta_4$ shares of $[\![q^k \mathbf{B}^{-1}\mathbf{c}]\!]$ and $[\![q^k \mathbf{x}]\!]$.
- Since SecNTTMult is t-NI secure, all the observations at line 7 can be simulated with at most $\delta_1 + \delta_2 + \delta_3 + \delta_4 + \delta_5$ shares of $[\![q^k \mathbf{B}^{-1}]\!]$.
- Since SecNTTMult is t-NI secure, all the observations at line 6 can be simulated with at most $\delta_1 + \delta_2 + \delta_3 + \delta_4 + \delta_6$ shares of $[\![q^{k_2} \Sigma_1]\!]$ and $[\![q^{k_1} \mathbf{y}]\!]$.
- Since MaskedCDT is t-NI secure, all the observations during line 1-4 can be simulated with at most $\delta_1 + \delta_2 + \delta_3 + \delta_4 + \delta_6 + \delta_7$ shares of randomness as input to MaskedCDT.

Clearly, the number of total observations for each sensitive input does not exceed δ.

I.4 Masking Ducas & Prest's hybrid sampler

See Alg. 18 for our masked variant of the hybrid sampler (see Alg. 3 and 13), which can be seen as a straightforward extension of the masked RingPeikert sampler. Notice that the only variable used more than once is $\mathbf{v}_1$, so by applying the SNI-secure Refresh gadget [2] to its shares we can prove NIo-security of the algorithm.

I.4.1 Proof of Theorem 5 Following Table 4, the proof below relies on t-NI security of GaussShareByShare, SecNTTMult and MaskedCDT, t-SNI security of Refresh, and t-NIo security of Unmask.

Proof. Below we omit the subscripts of masked variables denoting modulus for the sake of readability. Let us assume that an attacker has access to $\delta \leqslant t$ observations on the whole sampler. Our goal is to prove that all these δ observations can be perfectly simulated with at most δ shares of each secret among $[\![\mathbf{b}_1]\!]$, $[\![\mathbf{b}_2]\!]$, $[\![q^{k_2}\sigma_1]\!]$, $[\![q^{k_2}\sigma_2]\!]$, $[\![q^k \boldsymbol{\beta}_1]\!]$ and $[\![q^k \boldsymbol{\beta}_2]\!]$. We consider an attacker who peeks at internal computations as follows: δ_1 at line 25; δ_2 during addition at line 24; δ_3 during SecNTTMult at line 24; δ_4 at line 23; δ_5 during line 19-21; δ_6 at line 18; δ_7 during addition at line 17; δ_8 during first SecNTTMult at line 17; δ_9 during second SecNTTMult at line 17; δ_{10} at line 15; δ_{11} during line 12-14; δ_{12} at line 11; δ_{13} during addition at line 10; δ_{14} during first SecNTTMult at line 10; δ_{15} during second SecNTTMult at line 10; δ_{16} at line 7; δ_{17} at line 8; δ_{18} during line 1-4; Suppose $\sum_i \delta_i \leqslant \delta$.

- Since Unmask is t-NIo secure with public output $\mathbf{v}_0$, the observations during line 25 can be simulated with at most δ_1 shares of $[\![\mathbf{v}_0]\!]$ and $\mathbf{v}_0$.
- Due to the linear operations, the observations during addition at line 24 can be simulated with at most $\delta_1 + \delta_2$ shares of $[\![\mathbf{v}'_1]\!]$ and $[\![x_1 \mathbf{b}_1]\!]$.

Algorithm 17: MaskedRingPeikert sampler

Input: An arithmetic masking modulus $\mathsf{Q}^{\text{mask}} = q^{k+\ell}$ such that $k = k_1 + k_2$ and $\ell > 0$; a masked secret bases $[\![\mathbf{B}]\!]_q, [\![q^k\mathbf{B}^{-1}]\!]$ such that $\mathbf{B} \in \mathscr{K}^{2\times 2}$, $\mathscr{L} = \varphi(\mathbf{B}\mathscr{R}^2)$; a target center $\mathbf{c} \in \mathscr{R}$; a precomputed parameter $r \geqslant \eta_\varepsilon(\mathscr{R}^2)$ and masked matrix $[\![q^{k_2}\Sigma_1]\!]$ such that $\Sigma_1 = \mathbf{B}^{-1}\Sigma_0$, where $\Sigma_0 \in \mathscr{K}_{\mathbb{R}}^{2\times 2}$ is such that $\Sigma_0\Sigma_0^* = \Sigma - r^2\mathbf{B}\mathbf{B}^*$.

Result: $\mathbf{z} \in \mathscr{L}$ with distribution negligibly far from $D_{\mathscr{L}(\mathbf{B}),\mathbf{c},\Sigma}$.

Offline

1 **for** $j \in [0, d-1]$ **do**
2 $\quad [\![q^{k_1}y_{1,j}]\!] \leftarrow \mathsf{MaskedCDT}(t, \mathsf{Q}^{\text{mask}})$
3 $\quad [\![q^{k_1}y_{2,j}]\!] \leftarrow \mathsf{MaskedCDT}(t, \mathsf{Q}^{\text{mask}})$
4 **end for**
5 $[\![q^{k_1}\mathbf{y}]\!] := (([\![q^{k_1}y_{1,j}]\!])_{j\in[0,d-1]}, ([\![q^{k_1}y_{2,j}]\!])_{j\in[0,d-1]})$
6 $[\![q^k\mathbf{x}]\!] \leftarrow \mathsf{SecNTTMult}([\![q^{k_2}\Sigma_1]\!], [\![q^{k_1}\mathbf{y}]\!])$

Online

7 $[\![q^k\mathbf{B}^{-1}\mathbf{c}]\!] \leftarrow \mathsf{SecNTTMult}([\![q^k\mathbf{B}^{-1}]\!], \mathbf{c})$[a]
8 $[\![q^k\mathbf{v}]\!] \leftarrow [\![q^k\mathbf{B}^{-1}\mathbf{c}]\!] - [\![q^k\mathbf{x}]\!]$
9 **for** $j \in [1, d]$ **do**
10 $\quad [\![z_{1,j}]\!]_{q^\ell} \leftarrow \mathsf{GaussShareByShare}(\frac{[\![q^k v_{1,j}]\!]}{q^k})$
11 $\quad [\![z_{2,j}]\!]_{q^\ell} \leftarrow \mathsf{GaussShareByShare}(\frac{[\![q^k v_{2,j}]\!]}{q^k})$
12 **end for**
13 $[\![\mathbf{z}]\!]_q := [\![\mathbf{z}]\!]_{q^\ell}$ `/* compute every share mod q */`
14 $[\![\mathbf{z}]\!]_q \leftarrow \mathsf{SecNTTMult}([\![\mathbf{B}]\!]_q, [\![\mathbf{z}]\!]_q)$
15 $\mathbf{z} \leftarrow \mathsf{Unmask}([\![\mathbf{z}]\!]_q)$
16 **return** $\mathbf{z}$

[a] As the input center is not sensitive this is in practice a simpler variant of SecNTTMult where the second input is unmasked.

- Since SecNTTMult is t-NI secure, the observations during SecNTTMult at line 24 can be simulated with at most $\delta_1 + \delta_2 + \delta_3$ shares of $[\![x_1]\!]$ and $[\![\mathbf{b}_1]\!]$.
- Since Refresh is t-SNI secure, the observations during line 23 can be simulated with at most δ_4 shares of $[\![\mathbf{v}_1]\!]$.
- Since line 19-21 consists of independent local operations on each share of $[\![q^k z_1]\!]$ and GaussShareByShare is t-NI secure, all the observations during line 19-21 can be simulated with at most $\delta_1 + \delta_2 + \delta_3 + \delta_5$ shares of $[\![q^k z_1]\!]$.
- Due to the linear operations, the observations during addition at line 18 can be simulated with at most $\delta_1 + \delta_2 + \delta_3 + \delta_5 + \delta_6$ shares of $[\![q^k d_1]\!]$ and $[\![q^k y_1]\!]$.
- Due to the combination of linear operations and t-NI secure SecNTTMult, the observations during line 17 can be simulated with at most $\delta_1 + \delta_2 + \delta_3 + \delta_5 + \delta_6 + \delta_7 + \delta_8$ shares of $[\![q^k \beta_{1,1}]\!]$ and $[\![c_{1,1}]\!]$, and $\delta_1 + \delta_2 + \delta_3 + \delta_5 + \delta_6 + \delta_7 + \delta_9$ shares of $[\![q^k \beta_{1,2}]\!]$ and $[\![c_{1,2}]\!]$.
- Since SecNTTMult is t-NI secure, the observations at line 15 can be simulated with at most $\sum_{i\in[1,10]} \delta_i$ shares of $[\![x_2]\!]$ and $[\![\mathbf{b}_2]\!]$.
- Since line 12-14 consists of independent local operations on each share of $[\![q^k z_2]\!]$ and GaussShareByShare is t-NI secure, all the observations during line 12-14 can be simulated with at most $\sum_{i\in[1,11]} \delta_i$ shares of $[\![q^k z_2]\!]$.
- Due to the linear operations, the observations during addition at line 11 can be simulated with at most $\sum_{i\in[1,12]} \delta_i$ shares of $[\![q^k d_2]\!]$ and $[\![q^k y_2]\!]$.

– Due to the combination of linear operations and t-NI secure SecNTTMult, the observations during line 10 can be simulated with at most $\sum_{i\in[1,14]} \delta_i$ shares of $[\![q^k \beta_{2,1}]\!]$, and $\sum_{i\in[1,13]} \delta_i + \delta_{15}$ shares of $[\![q^k \beta_{2,2}]\!]$.
– Since SecNTTMult is t-NI secure, the observations during SecNTTMult at line 7 (resp. line 8) can be simulated with at most $\delta_1 + \delta_2 + \delta_3 + \delta_5 + \delta_6 + \delta_{16}$ shares of $[\![q^{k_2}\sigma_1]\!]$ and $[\![q^{k_1}u_1]\!]$ (resp. $\sum_{i\in[1,12]} \delta_i + \delta_{17}$ shares of $[\![q^{k_2}\sigma_2]\!]$ and $[\![q^{k_1}u_2]\!]$).
– Since MaskedCDT is t-NI secure, all the observations during line 1-4 can be simulated with at most $\sum_{i\in[1,12]} \delta_i + \delta_{16} + \delta_{17} + \delta_{18}$ shares of randomness as input to MaskedCDT.

Clearly, the number of total observations for each sensitive input does not exceed δ.

Algorithm 18: MaskedHybrid Gaussian sampler

Input: An arithmetic masking modulus $\mathsf{Q}^{\mathrm{mask}} = q^{2k+\ell}$ such that $k = k_1 + k_2$ and $\ell > 0$; a target center $\mathbf{c} \in \mathscr{R}^2$; a masked secret matrix $[[\![\mathbf{b}_1]\!]_q, [\![\mathbf{b}_2]\!]_{q^{\ell+k}}]$ such that $\mathscr{L} = \varphi(\mathbf{B}\mathscr{R}^2)$; a masked covariance $[\![q^{k_2}\sigma_1]\!]_{q^{\ell+k}}$ and $[\![q^{k_2}\sigma_2]\!]$ such that $\sigma_i := \sqrt{\frac{\sigma^2}{\langle \widetilde{\mathbf{b}}_i, \widetilde{\mathbf{b}}_i \rangle} - r^2} \in \mathscr{K}_{\mathbb{R}}^{++}$; a masked precomputed elements $[\![q^k\boldsymbol{\beta}_1]\!]_{q^{\ell+k}}$ and $[\![q^k\boldsymbol{\beta}_2]\!]$ such that $\boldsymbol{\beta}_i = \frac{\widetilde{\mathbf{b}}_i^*}{\langle \widetilde{\mathbf{b}}_i, \widetilde{\mathbf{b}}_i \rangle_{\mathscr{K}}}$, where $[\widetilde{\mathbf{b}}_1, \widetilde{\mathbf{b}}_2]$ is GSO of $\mathbf{B}$ over $\mathscr{K}$.

Result: $\mathbf{z}$ with distribution negligibly far from $D_{\mathscr{L},\mathbf{c},\sigma^2\mathbf{I}_{2d}}$.

Offline

for $j \in [0, d-1]$ **do**
  $[\![q^{k_1}u_{1,j}]\!]_{q^{\ell+k}} \leftarrow \mathsf{MaskedCDT}(t, q^\ell)$
  $[\![q^{k_1}u_{2,j}]\!] \leftarrow \mathsf{MaskedCDT}(t, \mathsf{Q}^{\mathrm{mask}})$
end for
$[\![q^{k_1}u_1]\!]_{q^{\ell+k}} := ([\![q^{k_1}u_{1,j}]\!]_{q^{\ell+k}})_{j\in[0,d-1]}$
$[\![q^{k_1}u_2]\!] := ([\![q^{k_1}u_{2,j}]\!])_{j\in[0,d-1]}$
$[\![q^k y_1]\!]_{q^{\ell+k}} \leftarrow \mathsf{SecNTTMult}([\![q^{k_2}\sigma_1]\!]_{q^{\ell+k}}, [\![q^{k_1}u_1]\!]_{q^{\ell+k}})$
$[\![q^k y_2]\!] \leftarrow \mathsf{SecNTTMult}([\![q^{k_2}\sigma_2]\!], [\![q^{k_1}u_2]\!])$

Online

`/* first nearest plane */`
$\mathbf{c}_2 \leftarrow \mathbf{c}, \mathbf{v}_2 \leftarrow \mathbf{0}$
$[\![q^k d_2]\!] \leftarrow \mathsf{SecNTTMult}([\![q^k\beta_{2,1}]\!], c_{2,1}) + \mathsf{SecNTTMult}([\![q^k\beta_{2,2}]\!], c_{2,2})$
$[\![q^k z_2]\!] \leftarrow [\![q^k d_2]\!] - [\![q^k y_2]\!]$
for $j \in [0, d-1]$ **do**
  $[\![x_{2,j}]\!]_{q^{\ell+k}} \leftarrow \mathsf{GaussShareByShare}(\frac{[\![q^k z_{2,j}]\!]}{q^k})$
end for
`/* second nearest plane */`
$[\![\mathbf{v}_1]\!]_{q^{\ell+k}} \leftarrow \mathsf{SecNTTMult}([\![x_2]\!]_{q^{\ell+k}}, [\![\mathbf{b}_2]\!]_{q^{\ell+k}})$
$[\![\mathbf{c}_1]\!]_{q^{\ell+k}} \leftarrow \mathbf{c}_2 - [\![\mathbf{v}_1]\!]_{q^{\ell+k}}$
$[\![q^k d_1]\!]_{q^{\ell+k}} \leftarrow \mathsf{SecNTTMult}([\![q^k\beta_{1,1}]\!]_{q^{\ell+k}}, [\![c_{1,1}]\!]_{q^{\ell+k}}) + \mathsf{SecNTTMult}([\![q^k\beta_{1,2}]\!]_{q^{\ell+k}}, [\![c_{1,2}]\!]_{q^{\ell+k}})$
$[\![q^k z_1]\!]_{q^{\ell+k}} \leftarrow [\![q^k d_1]\!]_{q^{\ell+k}} - [\![q^k y_1]\!]_{q^{\ell+k}}$
for $j \in [0, d-1]$ **do**
  $[\![x_{1,j}]\!]_{q^\ell} \leftarrow \mathsf{GaussShareByShare}(\frac{[\![q^k z_{1,j}]\!]_{q^{\ell+k}}}{q^k})$
end for
$[\![x_1]\!]_q := [\![x_1]\!]_{q^\ell}, [\![\mathbf{v}_1]\!]_q := [\![\mathbf{v}_1]\!]_{q^{\ell+k}}$ `/* compute every share mod q */`
$[\![\mathbf{v}_1']\!]_q \leftarrow \mathsf{Refresh}([\![\mathbf{v}_1]\!]_q)$
$[\![\mathbf{v}_0]\!]_q \leftarrow [\![\mathbf{v}_1']\!]_q + \mathsf{SecNTTMult}([\![x_1]\!]_q, [\![\mathbf{b}_1]\!]_q)$
$\mathbf{v}_0 \leftarrow \mathsf{Unmask}([\![\mathbf{v}_0]\!]_q)$
return $\mathbf{v}_0$