An erf Analog for Discrete Gaussian Sampling

Nicolas Gama[1,*], Anand Kumar Narayanan[1], Ryder LiuLin[1], Dongze Yue[1]

[1]SandboxAQ, Palo Alto, USA

Received: | *Revised:* | *Accepted:*

Abstract *Most of the current lattice-based cryptosystems rely on finding Gaussian Samples from a lattice that are close to a given target. To that end, two popular distributions have been historically defined and studied: the Rounded Gaussian distribution and the Discrete Gaussian distribution. The first one is nearly trivial to sample: simply round the coordinates of continuous Gaussian samples to their nearest integer. Unfortunately, the security of resulting cryptosystems are not as well understood. In the opposite, the second distribution is only implicitly defined by a restriction of the support of the continuous Gaussian distribution to the discrete lattice points. Thus, algorithms to achieve such distribution are more involved, even in dimension one. The justification for exerting this computational effort is that the resulting lattice-based cryptographic schemes are validated by rigorous security proofs, often by leveraging the fact that the distribution is radial and discrete Gaussians behave well under convolutions, enabling arithmetic between samples, as well as decomposition across dimensions.*
In this work, we unify both worlds. We construct out of infinite series, the cumulative density function of a new continuous distribution that acts as surrogate for the cumulative distribution of the discrete Gaussian. If μ is a center and x a sample of this distribution, then rounding $\mu + x$ yields a faithful Discrete Gaussian sample. This new sampling algorithm naturally splits into a pre-processing/offline phase and a very efficient online phase. The online phase is simple and has a trivial constant time implementation. Modulo the offline phase, our algorithm offers both the efficiency of rounding and the security guarantees associated with discrete Gaussian sampling.

Keywords: Gaussian distribution, Lattice-based cryptography, post-quantum cryptography

2010 Mathematics Subject Classification: 94A60, 11Y40

1 INTRODUCTION

The following sampling problem often arises in lattice-based cryptography: given a short basis of a lattice (embedded in a high-dimensional Euclidean space) and a target point, find a lattice point close to the target. For instance, such a sampling is critical to generating signatures in hash-and-sign [3] schemes culminating in Falcon [2] as well as Fiat-Shamir with abort signature schemes [6]. The short basis is often part of the secret key; the trapdoor enabling efficient computation. Deterministic solutions, such as finding the closest lattice point to the target, leak information about the short basis/secret key. Therefore randomization is necessary and one has to draw from a distribution of lattice points centered around the target. There are restrictions on the shape of the distribution: for example, drawing uniformly from the intersection of the lattice and a parallelepiped centered at the target also leaks the secret [8]. To not leak information about the secret key, it suffices to draw from a spherical discrete Gaussian distribution on the lattice centered at the target [3]. We chose to motivate the sampling problem through signature schemes instead of encryption schemes, as the former seems more demanding.

We call drawing samples from a spherical Gaussian distribution (with a prescribed center and standard deviation) over a lattice, given a short basis generating the lattice as the discrete Gaussian sampling problem. This is a well studied problem with two prominent algorithms respectively due to Klein [5] and Peikert [9]. Both algorithms eventually split the problem into several discrete Gaussian sampling problems over the integers; as long as either the basis is orthogonal or the standard deviation is larger than the smoothing factor [3]. The discrete Gaussian sampling problem over the integers takes the following form. Given a center/mean $\mu \in \mathbb{R}$ and a standard deviation $\sigma \in \mathbb{R}_{>0}$, draw samples (integers) from the distribution with the density function $\mathcal{D}_{\mathbb{Z},\sigma,\mu} : \mathbb{Z} \longrightarrow [0,1]$ sending $x \longmapsto \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) / \sum_{x\in\mathbb{Z}} \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right)$. We refrain from formally defining the rounded Gaussian distribution over lattices (c.f. [4, § 3.1]). Over cubical lattices, drawing from the rounded Gaussian is informally drawing from the continuous Gaussian distribution with the same parameters and rounding each coordinate to the nearest integer. Over the integers (that is, in one dimension), there exists a continuous bijection $E_\sigma : \mathbb{R} \to (0,1)$ such that a rounded Gaussian sample is $\lfloor E_\sigma^{-1}(r) + \mu \rfloor$. Here, (σ, μ) is the prescribed standard deviation-center pair and r is a uniform sample from the unit interval $(0,1)$. Further, E_σ may be set to $\mathrm{erf}(\frac{x-0.5}{\sqrt{2}\sigma})$ where erf is the error function associated with the mean zero, standard deviation one continuous Gaussian. This inversion formula splits

*Corresponding Author: nicolas.gama@sandboxaq.com

the sampling algorithm in two phases. An offline phase that samples from the uniform distribution in the unit interval which it then inverts under E_σ. An online phase that adds the center and rounds by taking the floor. We choose the floor function for rounding throughout as it is easier to implement (by bit truncation) than rounding to the nearest integer.

We propose a novel analogue F_σ of the erf-based E_σ function, to sample discrete Gaussians of any prescribed center and standard deviation over the integers. Key to our algorithm is the proof of existence (and explicit construction) of such function F_σ for discrete Gaussians analogous to the aforementioned E_σ for rounded Gaussians. That is, we construct a family of continuous strictly increasing functions $F_\sigma : \mathbb{R} \to (0, 1)$ parametrized by σ with the property that sampling from the discrete Gaussian $\mathcal{D}_{\mathbb{Z},\sigma,\mu}$ is identical to $\lfloor F_\sigma^{-1}(r) + \mu \rfloor$ for uniform $r \in (0, 1)$. Remarkably, our method and algorithm work for any desired μ, σ, even below the smoothing parameters. Our construction of F_σ also allows to split the discrete Gaussian sampling algorithm as an offline and an online phase. The Offline phase draws a uniform r from the unit interval and computes the inverse $F_\sigma^{-1}(r)$. In the motivating cryptographic schemes such as Falcon, the standard deviation σ of the Gaussian to sample from is determined during key generation. The randomness r is independent of the remainder of the protocol and hence can be generated offline. We prove that this offline phase runs in polynomial time to obtain approximations of the inverses F_σ^{-1} to arbitrary precision, irrespective of whether σ is bigger or smaller than the smoothing parameter. It remains an open problem to make this offline phase really efficient. The Online phase becomes as easy as adding the center and rounding by taking floors. It is multiples order of magnitude faster than any known counterpart. Further, it is trivial to implement the online phase in constant time, for applications (such as the Falcon signature scheme) that demand it as a precaution against side-channel attacks. A constant time discrete Gaussian sampler with an online/offline phase separation was first proposed by [7], using different techniques. Discrete Gaussian samplers with fixed σ and varying centers were also proposed in [1]. We construct the function F_σ as a fraction of two infinite series of Gaussians followed by a non trivial argument that F_σ is continuous and strictly increasing. In particular, this proves that F_σ^{-1} exists, a key fact in our proof that $\mathcal{D}_{\mathbb{Z},\sigma,\mu}$ is identical to $\lfloor F_\sigma^{-1}(r) + \mu \rfloor$ for uniform $r \in (0, 1)$. In algorithms for inverting F_σ, we truncate the infinite series appearing in F_σ, validated by the tail bounds we derive for the series. We describe two algorithms for performing the offline phase. The first is to invert F_σ by divide-and-conquer, guided by evaluations of F_σ using truncated series. The second is through rejection sampling on the derivative F'_σ, which we prove is correct by showing that the F'_σ is sub-Gaussian.

2 A NEW DISCRETE GAUSSIAN SAMPLING ALGORITHM ON INTEGERS

Every lattice algorithm that involves discrete Gaussian Sampling recursively splits into calls of the 1-dimensional sampling $\mathcal{D}_{\mathbb{Z},\sigma,\mu}$ over $\mathbb{Z}$, with a precomputed parameter σ, and varying center μ. Let $\rho(x) := \exp(-x^2/2\sigma^2)/\sqrt{2\pi}\sigma$. For $\sigma \in \mathbb{R}_{>0}$, consider the real function

$$F_\sigma(x) = \frac{\sum_{i=1}^{\infty} \rho(x - i)}{\sum_{i \in \mathbb{Z}} \rho(x - i)} = \frac{\rho(x - \mathbb{N}^*)}{\rho(x - \mathbb{Z})}$$

Our main result is that F_σ acts as a continuous cumulative distribution function for discrete Gaussian Samples in the sense of the following theorem:

Theorem 1. *For all $\sigma \in \mathbb{R}_{>0}$ and center $\mu \in \mathbb{R}$, the distribution of $\lfloor F_\sigma^{-1}(r) + \mu \rfloor$ where r has uniform distribution in the interval $(0, 1)$ is exactly $\mathcal{D}_{\mathbb{Z},\sigma,\mu}$*

In this theorem, it is easy to see that for all $\mu \in \mathbb{R}$ and $n \in \mathbb{Z}$, $F_\sigma(n - c)$ is by definition the probability that a sample from $\mathcal{D}_{\mathbb{Z},\sigma,\mu}$ is $< n$. The highly non-trivial fact is that F_σ^{-1} actually exists, if at first sight, we consider that its denominator is periodic. Despite all odds, our main Lemma below proves that F_σ is continuous and strictly monotonous from $\mathbb{R}$ to $(0, 1)$, and thus, Theorem 1 holds without any condition on the parameter σ.

Lemma 1. *F_σ is defined C^∞ and strictly increasing from $\mathbb{R}$ to $]0, 1[$ (bijective), and centered in $(0.5, 0.5)$ (in the sense $F_\sigma(x) + F_\sigma(1 - x) = 1$).*

Proof.
- F_σ is C^∞: the denominator is 1-periodic with an image > 0, and when written as $\sum c_k e^{-2\pi k x}$, its Fourier coefficients are Gaussian (so $c_k = o(n^{-k})$). In addition, we have normal convergence of the successive derivatives of the numerator, which makes the function C^∞.
- $F_\sigma(x)+F_\sigma(1-x) = 1$: since ρ being even implies $\sum_{i=-\infty}^{0} \rho(x-i) = \sum_{j=1}^{\infty} \rho(x-(1-j)) = \sum_{j=0}^{\infty} \rho((-x+1)-j)$.
- F_σ is strictly increasing from $\mathbb{R}$ to $]0, 1[$: let $(u, v) \in \mathbb{R}$ with $u < v$. The numerator of $F_\sigma(v) - F_\sigma(u)$ is $\sum_{i \geq 1, j \in \mathbb{Z}} \rho(v - i)\rho(u - j) - \sum_{i \in \mathbb{Z}, j \geq 1} \rho(v - i)\rho(u - j)$. All terms for $i, j \geq 1$ in both sums cancel out. Terms that remain are those where the sign is different: $\sum_{i \geq 1, j \leq 0} \rho(v - i)\rho(u - j) - \sum_{i \leq 0, j \geq 1} \rho(v - i)\rho(u - j)$.

Swap i, j in the second sum and regroup to get

$$\sum_{i\geq 1, j\leq 0} \rho(v-i)\rho(u-j) - \rho(v-j)\rho(u-i).$$

Note that $(v-j)^2+(u-i)^2 > (v-i)^2+(u-j)^2$, simply because the difference is $2(v-u)(i-j) > 0$. Since $x \to \exp(-x)$ strictly decreases over positive x, each term $\rho(v-i)\rho(u-j) - \rho(v-j)\rho(u-i)$ is > 0.

□

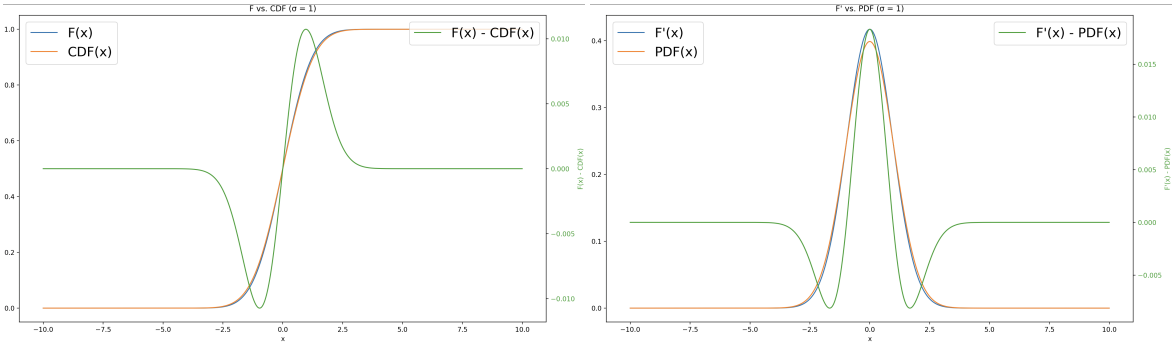

Figure 1: Comparison of $F_\sigma(x+1/2)$ and $F'_\sigma(x+1/2)$ versus the Gaussians CDF ($\texttt{erf}_\sigma$) and PDF (ρ_σ). The difference is plotted in green, and rescaled to make it visible.

2.1 THE FASTEST CONSTANT TIME ONLINE PHASE

In all practical cryptographic scenarios that require discrete Gaussians, σ is known in advance, and the center μ belongs to a discrete domain such as $2^{-p}\mathbb{Z}$ where p is a fixed precision parameter. Therefore, without any loss of precision, it is sufficient for the offline phase to provide one floored sample $x = 2^{-p}\lfloor 2^p.F_\sigma(r)\rfloor \in 2^{-p}\mathbb{Z}$ where $r \in (0,1)$ has uniform distribution, and the online phase returns $\lfloor x+\mu \rfloor \in \mathbb{Z}$ whose distribution is exactly $\mathcal{D}_{\mathbb{Z},\sigma,\mu}$.

To give one concrete use-case, if we represent $2^{52}.x$ and $2^{52}.\mu$ as 64-bit signed integers, this online phase has just one addition and one right shift: it is embarrassingly constant time and parallelizable, and at least 80x faster than the online phase of Falcon's official sampler.

As a bonus, for rare cases where the parameter σ is not known in advance, but just guaranteed to be above the smoothing parameter $\sigma_0 = \eta_\varepsilon(\mathbb{Z})$, the classical Discrete/Continuous convolution theorems still provide an interesting online phase: $z = \lfloor F^{-1}_{\sigma_0}(r) + y \rfloor$ where y is a continuous Gaussian sample of mean μ and parameter $\beta = \sqrt{\sigma^2 - \sigma_0^2}$. Indeed, the distribution of z remains at statistical distance ε from $\mathcal{D}_{\mathbb{Z},\sigma,\mu}$

2.2 POLYNOMIAL TIME ALGORITHMS FOR THE OFFLINE PHASE

We now sketch a few strategies for the offline phase that given a granularity $p \in \mathbb{N}$, a standard deviation $\sigma > 0$, and a precision $\varepsilon > 0$, samples x in time polynomial in p and $\log(\varepsilon)$, whose distribution is at a distance at most ε from $\lfloor F^{-1}(\text{uniform})\rfloor_{2^{-p}}$.

Binary search Since F is increasing, the first strategy that comes to mind is to invert it by binary search. To that end, all we need is to be able to obtain arbitrary precise evaluations of $F(x)$. The denominator is approximated via its Fourier series until the ratio of two consecutive terms is bounded by $1+\varepsilon/2$, and then we do the same for the numerator on the natural series. Both numerator and denominator series have a Gaussian decay, thus the number of terms considered is in $O(\sqrt{\log(\varepsilon^{-1})})$. Such binary search can be bootstrapped by precomputing and storing a table of $2^{p'}$ values of $F(k/2^{p'})$ for $k \in [0, 2^{p'}]$. In this case, the binary search recursion only requires p' lookups and $p-p'$ evaluations of F. As a side note, if the parameter σ is larger than the smoothing parameter $\sqrt{2\pi}\eta_\varepsilon(\mathbb{Z})$, which is the case in all cryptographic applications, the denominator of F is already within $[1-\varepsilon, 1+\varepsilon]$ and does not need to be evaluated at all.

Rejection sampling The binary search approach above is not "constant time", which means that a side-channel analysis of the memory access patterns (especially in the lookup part and recursion parts) leak in general some information about x. Making a binary search algorithm constant time is often detrimental to its performance; if

constant-timeness is an issue for the use-case (e.g. digital signature), we can also obtain the offline phase samples x by rejection sampling, which is enabled by the following lemma that shows that the tails of F' are sub-Gaussian:

Lemma 2. *For all $\sigma > 0$, $C_\sigma = \sup_{t\in\mathbb{R}} F'_\sigma(t+1/2)/\rho_\sigma(t)$ is finite. In addition, $\lim_{\sigma\to\infty} C_\sigma = 1$*

Before we prove this lemma, we point out that as an immediate consequence, the offline phase can operate as follow: Draw a continuous Gaussian sample $x \leftarrow \mathcal{D}_{\mathbb{R},\sigma,0}$ (e.g. via Box-Muller transform), return $\lfloor x \rfloor_{2^{-p}}$ iff. $F'_\sigma(x) \leq C_\sigma.\rho_\sigma(x-1/2)$, otherwise restart. The rejection probability is $1/C_\sigma$.

Proof. Let $t \in \mathbb{R}$, and N,D be the numerator and denominator of F, then $F'_\sigma(t) = N'(t)/D(t) - (D'(t)/D(t)^2)N(t)$. The functions D and D'/D^2 are continuous and periodic, therefore bounded over $\mathbb{R}$. Moreover, if $\sigma \geq \eta_\varepsilon(\mathbb{Z})$, those bounds are respectively $1+\varepsilon$ and ε. It remains to bound $N(t+0.5)/\rho_\sigma(t)$ and $N'(t+0.5)/\rho_\sigma(t)$. By parity, it is enough to consider $t = -u \leq 0$. Let $\mathbb{H} = 1/2 + \mathbb{N}$, we have

$$\frac{N'(t+0.5)}{\rho_\sigma(t)} = \sum_{i\in\mathbb{H}} \frac{\rho'_\sigma(t-i)}{\rho_\sigma(t)} = \sum_{i\in\mathbb{H}} \frac{i+u}{\sigma^2} \exp\left(\frac{-2iu-i^2}{2\sigma^2}\right) \quad \text{and} \quad \frac{N(t+0.5)}{\rho_\sigma(t)} = \sum_{i\in\mathbb{H}} \exp\left(\frac{-2iu-i^2}{2\sigma^2}\right)$$

Now that both i and u are > 0, we therefore have

$$\left|\frac{N'(t+0.5)}{\rho_\sigma(t)}\right| \leq \left(\frac{u}{\sigma^2}\sum_{i\in\mathbb{H}} \exp\left(\frac{-u-i^2}{2\sigma^2}\right)\right) + \left(\sum_{i\in\mathbb{H}} \frac{i}{\sigma^2}\exp\left(\frac{-i^2}{2\sigma^2}\right)\right) \quad \text{and} \quad \left|\frac{N(t+0.5)}{\rho_\sigma(t)}\right| \leq \sum_{i\in\mathbb{H}} \exp\left(\frac{-i^2}{2\sigma^2}\right)$$

The proof ends by noticing that $u\exp(-u)$ is bounded for $u \in \mathbb{R}^+$ and these Gaussian sums $\sum_{i\in\mathbb{H}} i\rho_\sigma(i)$ and $\sum_{i\in\mathbb{H}} \rho_\sigma(i)$ are finite, and the limits of these terms are obtained via Riemann integration. □

2.3 EXPERIMENTAL VALIDATION

As a proof of concept, we implemented numerical evaluations of F and F'. This allows to plot the corresponding cdf and pdf from Figure 1 and compare it to the Gaussian counterpart. For $\sigma = 1$, the maximal ratio between F'_σ and ρ_σ (in log-scale in Figure 2) is already smaller than 1.1, which makes rejection sampling quite effective (less than 10% of rejections).

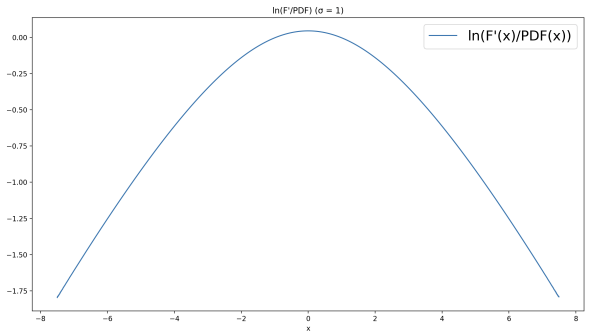

Figure 2: Experimental plot of $\log(F'_\sigma(x+1/2)/\rho_\sigma)$ for $\sigma = 1$. The ratio is maximal at 0, and $\log(C_\sigma)$ is very close to 0, which makes rejection sampling very effective.

We also implemented the binary search for $p = 24$, and measured an Offline phase of 70 microseconds per sample, and an online phase of 0.6 nanoseconds per sample. If the online phase is already optimal, the offline phase on the other hand is practical, but sill a bit slow to to effectively replace the current bi-modal samplers. Therefore, it remains an open problem to improve the offline phase by a few orders of magnitude, and/or to provide hardware support for this important probability distribution.

REFERENCES

[1] Carlos Aguilar Melchor, Martin R. Albrecht **and** Thomas Ricosset. "Sampling from Arbitrary Centered Discrete Gaussians for Lattice-Based Cryptography". **in***International Conference on Applied Cryptography and Network Security (ACNS 2017)*: Kanazawa, Japan, **july** 2017, **pages** 3–19. URL: `https://hal.science/hal-02548105`.

[2] Pierre-Alain Fouque **andothers**. "Falcon: Fast-Fourier lattice-based compact signatures over NTRU". **in***Submission to the NIST's post-quantum cryptography standardization process*: 36.5 (2018).

[3] Craig Gentry, Chris Peikert **and** Vinod Vaikuntanathan. "Trapdoors for hard lattices and new cryptographic constructions". **in***Proceedings of the fortieth annual ACM symposium on Theory of computing*: 2008, **pages** 197–206.

[4] Andreas Hülsing, Tanja Lange **and** Kit Smeets. "Rounded Gaussians: fast and secure constant-time sampling for lattice-based crypto". English. **in***Public-Key Cryptography - PKC 2018 - 21st IACR International Conference on Practice and Theory of Public-Key Cryptography, Proceedings*: **byeditor**Michel Abdalla **and** Ricardo Dahab. Lecture Notes in Computer Science. 21st IACR International Conference on Practice and Theory of Public-Key Cryptography (PKC 2018), PKC2018 ; Conference date: 25-03-2018 Through 29-03-2018. Germany: Springer, 2018, **pages** 728–757. ISBN: 9783319765778. DOI: 10.1007/978-3-319-76581-5_25. URL: https://pkc.iacr.org/2018/.

[5] Philip Klein. "Finding the closest lattice vector when it's unusually close". **in***Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*: 2000, **pages** 937–941.

[6] Vadim Lyubashevsky. "Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures". **in***Advances in Cryptology–ASIACRYPT 2009: 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings 15*: Springer. 2009, **pages** 598–616.

[7] Daniele Micciancio **and** Michael Walter. "Gaussian Sampling over the Integers: Efficient, Generic, Constant-Time". **in***Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*: **byeditor**Jonathan Katz **and** Hovav Shacham. **volume** 10402. Lecture Notes in Computer Science. Springer, 2017, **pages** 455–485. DOI: 10.1007/978-3-319-63715-0_16. URL: https://doi.org/10.1007/978-3-319-63715-0%5C_16.

[8] Phong Q Nguyen **and** Oded Regev. "Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures". **in***Advances in Cryptology-EUROCRYPT 2006: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28-June 1, 2006. Proceedings 25*: Springer. 2006, **pages** 271–288.

[9] Chris Peikert. "An efficient and parallel Gaussian sampler for lattices". **in***Advances in Cryptology–CRYPTO 2010: 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings 30*: Springer. 2010, **pages** 80–97.

A EXPERIMENTAL IMPLEMENTATION OF THE SAMPLER

We present DynXHalfBatchSamplerOnline, an online routine that generates batches of discrete Gaussian samples from $\mathcal{D}_{\mathbb{Z},\sigma,c}$ with fixed σ and varying c. The fast online sampler largely depends on the inversion samples pre-computed offline via DynXHalfBatchSamplerOffline, which draws points from uniform distribution and computes their inversion of F.

Algorithm 1 DynamicXHalfBatchSamplerOnline

Require: N centers of discrete Gaussian distribution $\mathbf{c} = \{c_i\}_{i\in[0,N-1]}$. N samples $\mathbf{t} = \{t_i\}_{i\in[0,N-1]}$ generated by DynamicXHalfBatchSamplerOffline.

Ensure: Output are valid samples of $\mathcal{D}_{\mathbb{Z},\sigma,c_i}$

1: **return** $\{\lfloor t_i + c_i \rfloor\}_{i\in[0,N-1]}$

For the offline routine, we first wrote a naive inverter of F that computes the truth table of the entire function on a desired interval and input value resolution. For a given u, the naive inverter traverses through the truth table and looks for the image closest to u and returns its preimage. Immediately, from the fact that F looks and behaves like a Gaussian CDF, we can optimize a few things: first, we can avoid computing half the values of F, since if $0 \leq u \leq 0.5$, the inverse will be in the lower half, and vice versa. Moreover, we can avoid computing $F(x)$ for x far from the center following the definition of standard deviation for Gaussian distributions. For example, we can be 99.7% sure that a random output's input will exist if we stop computing $F(x)$ when $x = \mu \pm 3\sigma$. This optimization also removes the need for specifying a desired interval of computation, instead changing the interval based on μ and σ.

Lastly, we employ the binary search approach mentioned earlier to perform an approximate binary search to look for the point within a 3σ domain such that its evaluation in F is closest to u with the precision of 2^{-p}. Since the binary search routine only checks evaluations of F at pivot points, we don't need to precompute the full truth table of F and only need to lazily evaluate F at the pivot points and their immediate neighbors. The final major optimization we implemented was for sampling many different t values at once. For a fixed σ, center, and n_{sum},

the evaluation of F stays the same. Therefore, we can reuse the setup for as many randomly sampled outputs u_i as we want, yielding better amortized run time.

Experimentally, the batch sampling, together with the other optimizations to the routine, far outpaces all single-sample algorithms, and since there are no dependencies between each sample, batch sampling can likely be further parallelized. The graphs were generated using somewhat unoptimized Python code for visualization and testing's sake, but we also implemented the reference algorithm in C with no advanced compiler options or special instruction sets and were able to draw 1000 samples with amortized performance of 70us per sample. The online phase takes 0.6ns per sample.

Algorithm 2 DynamicXHalfBatchSamplerOffline

Require: Resolution of linear space x_{res}, standard deviation σ, n_{res} inversion resolution, n_{sum} terms of summations for F, N samples to generate.

Ensure: Output are N random inversions of F.

samples $\leftarrow$ []
$i \leftarrow 0$
while $i < N$ **do**
    $u \leftarrow \text{Uniform}(0, 1)$
    start $\leftarrow 0.5 - K\sigma$ **if** $u < 0.5$ **else** 0.5
    end $\leftarrow 0.5$ **if** $u < 0.5$ **else** $0.5 + K\sigma$
    samples.append(FindClosestInversion-BinarySearch(u, start, end, $n_{\text{res}}, \sigma, n_{\text{sum}}$))
    $i \leftarrow i + 1$
end while
return samples

Algorithm 3 FindClosestInversion-BinarySearch

Require: Search target u. Start and end of the search, inversion resolution n_{res}, standard deviation σ, and number of F approximation terms n_{sum}.

Ensure: Output is in range (start, end) such that its corresponding evaluation in F with standard deviation σ up to n_{sum} terms is closest to u in the search range.

If $u \leq F_{\sigma,n_{\text{sum}}}$(start), return start.
If $u \geq F_{\sigma,n_{\text{sum}}}$(end), return end.
loop
    If end $-$ start $\leq 2^{-p}$, **return** start.
    mid $\leftarrow$ (start + end)/2 and fmid = $F_{\sigma,n_{\text{sum}}}$(mid)
    If u =fmid **return** mid.
    if $u <$ fmid **then**
        end $\leftarrow$ mid
    else
        start $\leftarrow$ mid
    end if
end loop