An Efficient and Parallel Gaussian Sampler for Lattices

Chris Peikert*

February 18, 2010

Abstract

At the heart of many recent lattice-based cryptographic schemes is a polynomial-time algorithm that, given a 'high-quality' basis, generates a lattice point according to a Gaussian-like distribution. Unlike most other operations in lattice-based cryptography, however, the known algorithm for this task (due to Gentry, Peikert, and Vaikuntanathan; STOC 2008) is rather inefficient, and is inherently sequential.

We present a new Gaussian sampling algorithm for lattices that is *efficient* and *highly parallelizable*. At a high level, the algorithm resembles the "perturbation" heuristic proposed as part of NTRUSign (Hoffstein *et al.*, CT-RSA 2003), though the details are quite different. To our knowledge, this is the first algorithm and rigorous analysis demonstrating the security of a perturbation-like technique.

1 Introduction

In recent years, there has been rapid development in the use of *lattices* for constructing rich cryptographic schemes.[1] These include digital signatures (both 'tree-based' [LM08] and 'hash-and-sign' [GPV08, Pei09]), identity-based encryption [GPV08] and hierarchical IBE [Pei09, CHK09], noninteractive zero knowledge [PV08], and even a fully homomorphic cryptosystem [Gen09].

The cornerstone of many of these schemes (particularly, but not exclusive to, those that 'answer queries') is the polynomial-time algorithm of [GPV08] that samples from a so-called *discrete Gaussian* probability distribution over a lattice Λ (or a desired coset thereof). More precisely, for a vector $\mathbf{c} \in \mathbb{R}^n$ and a "width" parameter $s > 0$, the distribution $D_{\Lambda+\mathbf{c},s}$ assigns to each $\mathbf{v} \in \Lambda + \mathbf{c}$ a probability proportional to $\exp(-\pi\|\mathbf{v}\|^2/s^2)$. Given $\mathbf{c}$, a basis $\mathbf{B}$ of Λ, and a sufficiently large s (related to the 'quality' of $\mathbf{B}$), the GPV algorithm outputs a sample from a distribution statistically close to $D_{\Lambda+\mathbf{c},s}$.[2] Informally speaking, this algorithm is 'zero-knowledge' in the sense that it leaks *no information* about its input basis $\mathbf{B}$ (aside from a bound on its quality), because $D_{\Lambda+\mathbf{c},s}$ is defined without reference to any particular basis.

While the sampling algorithm of [GPV08] has numerous applications in cryptography and beyond, for both practical and theoretical purposes it also has some drawbacks:

*School of Computer Science, College of Computing, Georgia Institute of Technology. Email: cpeikert@cc.gatech.edu.
This material is based upon work supported by the National Science Foundation under Grant CNS-0716786. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

[1] A lattice $\Lambda \subset \mathbb{R}^n$ is a periodic 'grid' of points, or more formally, a discrete subgroup of $\mathbb{R}^n$ under addition. It is generated by a (not necessarily unique) *basis* $\mathbf{B} \subset \mathbb{R}^{n \times k}$ of k linearly independent vectors, as $\Lambda = \{\mathbf{Bz} : \mathbf{z} \in \mathbb{Z}^k\}$. In this paper we are concerned only with *full-rank* lattices, i.e., where $k = n$.

[2] Equivalently, by subtracting $\mathbf{c}$ from the output it samples a lattice point from a Gaussian distribution centered at $-\mathbf{c}$.

- First, it is rather *inefficient*: on an n-dimensional lattice, a straightforward implementation using exact arithmetic requires $\Omega(n^4)$ bit operations. (While it might be possible to improve this running time somewhat by using lower precision, great care would be needed to ensure a correct output distribution.)

- Second, it is *inherently sequential*: to generate a sample, the algorithm performs n adaptive iterations, where the choices made in each iteration affect the values used in the next. This stands in stark contrast to other 'embarrassingly parallelizable' operations that are typical of lattice-based cryptography.

1.1 Contributions

We give a new algorithm that samples from a discrete Gaussian distribution $D_{\Lambda+\mathbf{c},s}$ over a lattice, given a 'high-quality' basis for Λ. The algorithm is:

- *Efficient*: it requires only $\tilde{O}(n^2)$ work on a lattice of dimension n;

- *Fully parallelizable*: it can allocate $\tilde{O}(w)$ work to n^2/w processors, for any $w \geq 1$;

- *Offline / online*: assuming the basis is known in advance of the point $\mathbf{c}$ (which is the norm), much of its work can be performed as precomputation.

At a very high level, the algorithm resembles the "perturbation" heuristic proposed for NTRUSign [HHGP$^+$03], but the details differ significantly (see Section 1.2 for a comparison). To our knowledge, this is the first algorithm and analysis to demonstrate the theoretical soundness of a perturbation-like technique. Along the way, we also prove some general facts relating continuous and discrete Gaussians, which we expect will be useful elsewhere.

The improved efficiency of our new algorithm does not come *entirely* for free: the one aspect in which the algorithm can be inferior to that of [GPV08] is in the *width* s of the sampled Gaussian. Given the same n-dimensional basis, the new algorithm may be looser than the old one by a factor of at most n in the worst case (over the choice of basis). Fortunately, bases exhibiting such a large gap are quite rare — for bases of 'hard' cryptographic lattices as generated according to a method described in [AP09], we show that the gap is only $O(\sqrt{n})$ with high probability. Moreover, we show that in other important cases the ratio is nearly constant (see below for more details).

In a cryptographic application, the width s of the sampled Gaussian is the main quantity governing the concrete security and, if applicable, the approximation factor of the underlying worst-case lattice problems (via worst-case/average-case reductions as pioneered by Ajtai [Ajt04]). This is because for security, it needs to be hard for an adversary to find a lattice point within the likely radius $s\sqrt{n}$ of the Gaussian (i.e., after truncating its negligibly likely tail). The wider the distribution, the more leeway the adversary has in an attack, and the larger the scheme's parameters must be to compensate. On the other hand, a faster and parallelizable sampling algorithm allows for the use of larger parameters without sacrificing efficiency. Based on current estimates of the concrete security for random cryptographic lattices [GN08, MR09], we believe that our new algorithm can offer a *significant net gain* in efficiency for comparable levels of security, especially when exploiting parallelism. Of course, testing this conjecture requires careful experiments on concrete implementations and key sizes, which we leave for later work. Even putting aside potential efficiency improvements, our parallel sampling algorithm and its analysis also offer theoretical advantages that we expect to be useful elsewhere.

We now describe the algorithm's properties in more detail, in comparison with that of [GPV08]. Given a lattice basis $\mathbf{B}$, the GPV algorithm can sample from a discrete Gaussian having width as small as $\|\widetilde{\mathbf{B}}\| =$

$\max_i \|\widetilde{\mathbf{b}_i}\|$, where $\widetilde{\mathbf{B}}$ denotes the *Gram-Schmidt orthogonalization* of $\mathbf{B}$.[3] (The quantity $\|\widetilde{\mathbf{B}}\|$ is at most $\max_i \|\mathbf{b}_i\|$, and in some cases can be substantially smaller.) In contrast, our new algorithm works for a width as small as the *largest singular value* $\sigma_1(\mathbf{B})$ of the basis $\mathbf{B}$, or equivalently, the square root of the largest eigenvalue of $\mathbf{B}\mathbf{B}^t$. Any basis $\mathbf{B}$ can always be processed (without increasing $\|\widetilde{\mathbf{B}}\|$) to guarantee that $\sigma_1(\mathbf{B}) \leq n \cdot \|\widetilde{\mathbf{B}}\|$, so our algorithm is at worst an n factor looser than that of [GPV08]. However, this bound is rather loose, and much tighter bounds can fortunately be established in important special cases. For instance, the basis generation technique of [AP09] (following [Ajt99]) for 'worst-case-hard' lattices produces a basis $\mathbf{B}$ for which the ratio $\sigma_1(\mathbf{B})/\|\widetilde{\mathbf{B}}\| = O(\sqrt{n})$ with high probability. Even better, when the vectors of $\mathbf{B}$ are themselves drawn from a discrete Gaussian (e.g., as in the basis-delegating applications of [Pei09, CHK09]), we can show that $\sigma_1(\mathbf{B})$ is only a slightly super-$\sqrt{\log n}$ factor larger than $\|\widetilde{\mathbf{B}}\|$, with high probability. In this case, the performance improvements of our algorithm therefore come at almost no asymptotic cost in security.

1.2 Technical Overview

The GPV sampling algorithm [GPV08] is based closely on Babai's "nearest-plane" decoding algorithm for lattices [Bab86]. Babai's algorithm takes a point $\mathbf{c} \in \mathbb{R}^n$ and a lattice basis $\mathbf{B} = \{\mathbf{b}_1, \ldots, b_n\}$, and for $i = n, \ldots, 1$ computes a coefficient $z_i \in \mathbb{Z}$ for $\mathbf{b}_i$ by iteratively projecting ('rounding') $\mathbf{c}$ orthogonally to the nearest hyperplane of the form $z_i\mathbf{b}_i + \mathrm{span}(\mathbf{b}_1, \ldots, \mathbf{b}_{i-1})$. The output is the lattice vector $\sum_i z_i\mathbf{b}_i$, whose distance from the original $\mathbf{c}$ can be bounded by the quality of $\mathbf{B}$. The GPV algorithm, whose goal is to *randomly sample* from a discrete Gaussian centered at $\mathbf{c}$, uses *randomized rounding* in each iteration to select a 'nearby' plane, under a carefully defined probability distribution. (This technique is also related to another randomized-rounding algorithm of Klein [Kle00] for a different problem.)

In addition to his nearest-plane algorithm, Babai also proposed a simpler (but somewhat looser) lattice decoding algorithm, which we call "simple rounding." In this algorithm, a given point $\mathbf{c} \in \mathbb{R}^n$ is rounded to the lattice point $\mathbf{B}\lfloor\mathbf{B}^{-1}\mathbf{c}\rceil$, where each coordinate of $\mathbf{B}^{-1}\mathbf{c} \in \mathbb{R}^n$ is independently rounded to its nearest integer. With precomputation of $\mathbf{B}^{-1}$, this algorithm can be quite practical, and is easily parallelized among up to n^2 processors. Unfortunately, its *deterministic* form it turns out to be completely insecure for 'answering queries' (e.g., digital signatures), as demonstrated by Nguyen and Regev [NR09].

A natural question, given the approach of [GPV08], is whether a *randomized* variant of the simple-rounding algorithm is secure. Unlike with the randomized nearest-plane algorithm, in this case the resulting probability distribution is *not* spherical, nor is it zero-knowledge. Specifically, the *covariance* matrix of the distribution, which can be measured efficiently given a small number of samples, is approximately $\mathbf{B}\mathbf{B}^t$; this leaks the entire 'geometry' of the secret basis $\mathbf{B}$, up to rigid rotations. One might still wonder whether $\mathbf{B}\mathbf{B}^t$ can be *simulated* efficiently (without any privileged knowledge about the lattice) when $\mathbf{B}$ is itself drawn from a 'nice' distribution, such as a discrete Gaussian. Indeed, when the vectors of $\mathbf{B}$ are drawn independently from a *continuous* Gaussian, the matrix $\mathbf{B}\mathbf{B}^t$ has the so-called *Wishart distribution*, which can be generated 'obliviously' (without knowledge of $\mathbf{B}$ itself) using the *Bartlett decomposition*. (See, e.g., [Ksh59] and references therein). Unfortunately, these facts do not quite seem to carry over to *discrete* Gaussians, though they may be useful in another context.

Instead we take a different route, which is inspired by the following facts. Recall that if X and Y are two independent random variables, the probability distribution of their sum $X + Y$ is the *convolution* of their individual distributions. In addition, for continuous (not necessarily spherical) Gaussians, covariance matrices

[3] Actually, the width also includes a small $\omega(\sqrt{\log n})$ factor, which is also present in our new algorithm, so for simplicity we ignore it in this summary.

are additive under convolution. In particular, if Σ_1 and Σ_2 are covariance matrices such that $\Sigma_1 + \Sigma_2 = s^2\mathbf{I}$, then the convolution of two Gaussians with covariance matrices Σ_1, Σ_2 (respectively) is a *spherical* Gaussian with standard deviation s.

The above facts give the basic idea for our algorithm, which is simply to convolve the (randomized) simple-rounding algorithm with a suitable non-spherical (continuous) Gaussian, yielding a spherically distributed output. However, note that we want the algorithm to generate a *discrete* distribution — i.e., it must output a lattice point — so we should not alter the output of the randomized rounding step. Instead, we *first* perturb the desired center $\mathbf{c}$ by a suitable non-spherical Gaussian, then apply randomized rounding to the resulting perturbed point. Strictly speaking this is not a convolution, because the rounding step now depends on the output of the perturbation step, but we can reduce the analysis to a true convolution using bounds related to the smoothing parameter of the lattice [MR07].

The main remaining question is: for a given covariance matrix $\Sigma_1 = \mathbf{B}\mathbf{B}^t$ (corresponding to the rounding step), for what values of s is there an (efficiently sampleable) Gaussian with covariance matrix $\Sigma_2 = s^2\mathbf{I} - \Sigma_1$? The covariance matrix of any Gaussian is (symmetric) positive definite, i.e., all its eigenvalues are positive.[4] Conversely, every positive definite matrix is the covariance of some Gaussian, which can sampled efficiently using (for example) the Cholesky decomposition. Since any eigenvector of Σ_1 (with eigenvalue σ^2) is also an eigenvector of $s^2\mathbf{I}$ (with eigenvalue s^2), it must be an eigenvector of Σ_2 (with eigenvalue $s^2 - \sigma^2$) as well. Therefore, a necessary and sufficient condition is that all the eigenvalues of Σ_1 be less than s^2. Equivalently, the algorithm works for any s that exceeds the largest singular value of the given basis $\mathbf{B}$. More generally, our algorithm can generate any (possibly non-spherical) discrete Gaussian with covariance parameter $\Sigma > \Sigma_1$ (i.e., $\Sigma - \Sigma_1$ is positive definite).

In retrospect, the high-level structure of our algorithm resembles the *perturbation* heuristic proposed for NTRUSign [HHGP+03], though the details are quite different. First, the perturbation and rounding steps in NTRUSign are both *deterministic* with respect to two or more bases, and there is evidence that this is insecure [MPSW09], at least for a large polynomial number of signatures. Second, the signing and perturbation bases used in NTRUSign are chosen *independently*, whereas our perturbations are carefully chosen to conceal the statistics that would otherwise be leaked by randomized rounding.

1.3 Organization

Section 2 reviews the necessary linear algebra and lattice background. In Section 3 we present the new sampling algorithm, prove its correctness, and analyze its efficiency under suggested optimizations. In Section 4 we give bounds on the largest singular value of a basis relative to other important quantities, both in general and in some important special cases.

2 Preliminaries

2.1 Notation

A nonnegative function $f\colon \mathbb{N} \to \mathbb{R}$ is called *negligible*, written $f(n) = \mathrm{negl}(n)$, if it vanishes faster than any inverse polynomial, i.e., $f(n) = o(n^{-c})$ for every constant $c \geq 0$. A function $g\colon \mathbb{N} \to [0,1]$ is called *overwhelming* if it is $1 - \mathrm{negl}(n)$. We say that two probability distributions (implicitly parameterized by n) are *statistically indistinguishable* if their statistical distance is $\mathrm{negl}(n)$.

[4] For simplicity, in this paper we deal only with non-degenerate (full-rank) Gaussians; degenerate Gaussians can have positive *semidefinite* covariances matrices.

We use bold lower-case letters (e.g., $\mathbf{x}$) to denote vectors in $\mathbb{R}^n$, for some positive integer dimension n that remains the same throughout the paper. We use bold upper-case letters (e.g., $\mathbf{B}$) for ordered sets of vectors, and identify the set with the matrix having the vectors as its columns. We frequently use upper-case Greek letters (e.g., Σ) to denote (symmetric) *positive definite* matrices (defined below). In contexts where a matrix is expected, we sometimes use a scalar $c \in \mathbb{R}$ to denote $c \cdot \mathbf{I}$, where $\mathbf{I}$ is the identity matrix of appropriate dimension. We let $\|\mathbf{B}\| = \max_i \|\mathbf{b}_i\|$, where $\|\cdot\|$ denotes the Euclidean norm.

2.2 Linear Algebra

A symmetric matrix $\Sigma \in \mathbb{R}^{n \times n}$ is *positive definite*, written $\Sigma > \mathbf{0}$, if $\mathbf{x}^t \Sigma \mathbf{x} > 0$ for all nonzero $\mathbf{x} \in \mathbb{R}^n$.[5] Equivalently, its spectral decomposition is

$$\Sigma = \mathbf{Q}\mathbf{D}^2\mathbf{Q}^{-1} = \mathbf{Q}\mathbf{D}^2\mathbf{Q}^t,$$

where $\mathbf{Q} \in \mathbb{R}^{n \times n}$ is an orthogonal matrix (i.e., one for which $\mathbf{Q}^t\mathbf{Q} = \mathbf{Q}\mathbf{Q}^t = \mathbf{I}$) whose columns are eigenvectors of Σ, and $\mathbf{D}$ is the real diagonal matrix of the square roots of the corresponding eigenvalues, all of which are positive. From this, we have $\Sigma > 0$ if and only if $\Sigma^{-1} > 0$. Positive definiteness defines a partial ordering on symmetric matrices: we say that $\mathbf{X} > \mathbf{Y}$ if $(\mathbf{X} - \mathbf{Y}) > \mathbf{0}$.

For any nonsingular matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$, the symmetric matrix $\Sigma = \mathbf{B}\mathbf{B}^t$ is positive definite, because

$$\mathbf{x}^t \Sigma \mathbf{x} = \langle \mathbf{B}^t\mathbf{x}, \mathbf{B}^t\mathbf{x} \rangle = \|\mathbf{B}^t\mathbf{x}\|^2 > 0$$

for nonzero $\mathbf{x} \in \mathbb{R}^n$. We say that $\mathbf{B}$ is a *square root* of $\Sigma > \mathbf{0}$, written $\mathbf{B} = \sqrt{\Sigma}$, if $\mathbf{B}\mathbf{B}^t = \Sigma$. Every $\Sigma > 0$ has a square root $\mathbf{B} = \mathbf{Q}\mathbf{D}$, where $\Sigma = \mathbf{Q}\mathbf{D}^2\mathbf{Q}^t$ is the spectral decomposition of Σ as above. Moreover, the square root is unique up to right-multiplication by an orthogonal matrix, i.e., $\mathbf{B}' = \sqrt{\Sigma}$ if and only if $\mathbf{B}' = \mathbf{B}\mathbf{P}$ for some orthogonal matrix $\mathbf{P}$. A square root of particular interest is given by the *Cholesky decomposition* $\Sigma = \mathbf{L}\mathbf{L}^t$, where $\mathbf{L}$ is a (unique) lower-triangular matrix. It can be computed efficiently in fewer than n^3 multiplication and addition operations (over the reals).

For a nonsingular matrix $\mathbf{B}$, a *singular value decomposition* is $\mathbf{B} = \mathbf{Q}\mathbf{D}\mathbf{P}^t$, where $\mathbf{Q}, \mathbf{P} \in \mathbb{R}^{n \times n}$ are orthogonal matrices, and $\mathbf{D}$ is a diagonal matrix with positive entries $\sigma_i > 0$ (called the singular values) on the diagonal, in non-increasing order. Under this convention, $\mathbf{D}$ is uniquely determined, and $\sigma_1(\mathbf{B}) = \max_{\mathbf{u}} \|\mathbf{B}\mathbf{u}\| = \max_{\mathbf{u}} \|\mathbf{B}^t\mathbf{u}\|$, where the maximum is taken over all *unit* vectors $\mathbf{u} \in \mathbb{R}^n$. Note that

$$\Sigma = \mathbf{B}\mathbf{B}^t = \mathbf{Q}\mathbf{D}\mathbf{P}^t\mathbf{P}\mathbf{D}^t\mathbf{Q}^t = \mathbf{Q}\mathbf{D}^2\mathbf{Q}^t,$$

so the eigenvalues of Σ are the squares of the singular values of $\mathbf{B}$.

2.3 Lattices

A *lattice* Λ is a discrete additive subgroup of $\mathbb{R}^n$. In this work we are only concerned with full-rank lattices, which are generated by some nonsingular *basis* $\mathbf{B} \in \mathbb{R}^{n \times n}$, as the set $\Lambda = \{\mathbf{B}\mathbf{z} : \mathbf{z} \in \mathbb{Z}^n\}$. When $n \geq 2$, every lattice has infinitely many bases, which are related by unimodular transformations: $\mathbf{B}'$ and $\mathbf{B}$ generate the same lattice if and only if $\mathbf{B}' = \mathbf{B}\mathbf{U}$ for some unimodular $\mathbf{U} \in \mathbb{Z}^{n \times n}$.

[5] By relaxing the condition to $\mathbf{x}^t \Sigma \mathbf{x} \geq 0$, we obtain the notion of a positive *semidefinite* matrix $\Sigma \geq \mathbf{0}$. Our results can be extended to work for this notion as well, but at the expense of technical complications that provide little concrete benefit.

2.4 Gaussians

The Gaussian function $\rho : \mathbb{R}^n \to (0, 1]$ is defined as

$$\rho(\mathbf{x}) \stackrel{\Delta}{=} \exp(-\pi \cdot \|\mathbf{x}\|^2) = \exp(-\pi \cdot \langle \mathbf{x}, \mathbf{x} \rangle).$$

Applying a linear transformation given by a nonsingular matrix $\mathbf{B}$ yields the Gaussian function

$$\rho_{\mathbf{B}}(\mathbf{x}) \stackrel{\Delta}{=} \rho(\mathbf{B}^{-1}\mathbf{x}) = \exp\left(-\pi \cdot \left\langle \mathbf{B}^{-1}\mathbf{x}, \mathbf{B}^{-1}\mathbf{x} \right\rangle\right) = \exp\left(-\pi \cdot \mathbf{x}^t \Sigma^{-1} \mathbf{x}\right),$$

where $\Sigma = \mathbf{B}\mathbf{B}^t > 0$. Because $\rho_{\mathbf{B}}$ is distinguished only up to Σ, we usually refer to it as $\rho_{\sqrt{\Sigma}}$.

Normalizing $\rho_{\sqrt{\Sigma}}$ by its total measure $\int_{\mathbb{R}^n} \rho_{\sqrt{\Sigma}}(\mathbf{x})\, d\mathbf{x} = \sqrt{\det \Sigma}$ over $\mathbb{R}^n$, we obtain the probability distribution function of the (continuous) *Gaussian distribution* $D_{\sqrt{\Sigma}}$. It is easy to check that a random variable $\mathbf{x}$ having distribution $D_{\sqrt{\Sigma}}$ can be written as $\sqrt{\Sigma} \cdot \mathbf{z}$, where $\mathbf{z}$ has spherical Gaussian distribution D_1. Therefore, the random variable $\mathbf{x}$ has *covariance*

$$\mathop{\mathrm{E}}_{\mathbf{x} \sim D_{\sqrt{\Sigma}}}\left[\mathbf{x} \cdot \mathbf{x}^t\right] = \sqrt{\Sigma} \cdot \mathop{\mathrm{E}}_{\mathbf{z} \sim D_1}\left[\mathbf{z} \cdot \mathbf{z}^t\right] \cdot \sqrt{\Sigma}^t = \sqrt{\Sigma} \cdot \frac{\mathbf{I}}{2\pi} \cdot \sqrt{\Sigma}^t = \frac{\Sigma}{2\pi},$$

by linearity of expectation. (The $\frac{\mathbf{I}}{2\pi}$ covariance of $\mathbf{z} \sim D_1$ arises from the independence of its entries, which are each distributed as D_1 in one dimension, and have variance $\frac{1}{2\pi}$.) *For convenience, in this paper we implicitly scale all covariance matrices by a 2π factor, and refer to Σ as the covariance matrix of $D_{\sqrt{\Sigma}}$.*

For two functions $f, g : \mathbb{R}^n \to \mathbb{R}$, their *convolution* $(f * g) : \mathbb{R}^n \to \mathbb{R}$ is the function

$$(f * g)(\mathbf{x}) = \int_{\mathbb{R}^n} f(\mathbf{x} - \mathbf{t}) \cdot g(\mathbf{t})\, d\mathbf{t}.$$

It is well-known that the convolution of two Gaussian functions is another Gaussian function, whose covariance matrix is the sum of the original covariance matrices.

Lemma 2.1. *For any positive definite matrices $\Sigma_1, \Sigma_2 \in \mathbb{R}^{n \times n}$, we have $(\rho_{\sqrt{\Sigma_1}} * \rho_{\sqrt{\Sigma_2}}) = \rho_{\sqrt{\Sigma_1 + \Sigma_2}}$.*

An elementary (but somewhat cumbersome) proof of Lemma 2.1 can be obtained directly from the definition of convolution, by a matrix version of 'completing the square' in the exponent. Alternatively, a more concise proof may be obtained by employing the convolution theorem (which says that $\widehat{f * g} = \widehat{f} \cdot \widehat{g}$, where $\widehat{\cdot}$ denotes the Fourier transform) and the fact that $\widehat{\rho_{\sqrt{\Sigma}}} = \rho_{\sqrt{\Sigma^{-1}}}$.

2.5 Gaussians on Lattices

Let Λ be a lattice and $\mathbf{c}$ be a point in $\mathbb{R}^n$, and let $\Sigma > \mathbf{0}$ be a positive definite matrix. The *discrete Gaussian distribution* $D_{\Lambda + \mathbf{c}, \sqrt{\Sigma}}$ is simply the Gaussian distribution restricted so that its support is the coset $\Lambda + \mathbf{c}$. That is, for all $\mathbf{x} \in \Lambda + \mathbf{c}$,

$$D_{\Lambda + \mathbf{c}, \sqrt{\Sigma}}(\mathbf{x}) = \frac{\rho_{\sqrt{\Sigma}}(\mathbf{x})}{\rho_{\sqrt{\Sigma}}(\Lambda + \mathbf{c})} \propto \rho_{\sqrt{\Sigma}}(\mathbf{x}).$$

Definition 2.2. Let $\Sigma > \mathbf{0}$. We say that $\sqrt{\Sigma} \geq \eta_\epsilon(\Lambda)$ if $\rho_{\sqrt{\Sigma^{-1}}}(\Lambda^* \backslash \{\mathbf{0}\}) = \rho(\sqrt{\Sigma} \cdot \Lambda^* \backslash \{\mathbf{0}\}) \leq \epsilon$, i.e., if $\eta_\epsilon(\sqrt{\Sigma^{-1}} \cdot \Lambda) \leq 1$.

The next lemma says that when the standard deviation $\sqrt{\Sigma}$ exceeds the smoothing parameter of Λ, the Gaussian measure is essentially the same on all cosets of Λ. The lemma following that one gives a tight bound on the smoothing parameter of the integer lattice $\mathbb{Z}^n$.

Lemma 2.3 (Extension of [MR07, Lemma 4.4]). *Let Λ be any n-dimensional lattice. For any $\epsilon \in (0,1)$, $\Sigma > \mathbf{0}$ such that $\sqrt{\Sigma} \geq \eta_\epsilon(\Lambda)$, and $\mathbf{c} \in \mathbb{R}^n$,*

$$\rho_{\sqrt{\Sigma}}(\Lambda + \mathbf{c}) \in [\tfrac{1-\epsilon}{1+\epsilon}, 1] \cdot \rho_{\sqrt{\Sigma}}(\Lambda).$$

Proof. Follows directly by applying $\sqrt{\Sigma^{-1}}$ as a linear transform to Λ, and by $\eta_\epsilon(\sqrt{\Sigma^{-1}} \cdot \Lambda) \leq 1$. □

Lemma 2.4 (Special case of [MR07, Lemma 3.3]). *For any $\epsilon > 0$,*

$$\eta_\epsilon(\mathbb{Z}^n) \leq \sqrt{\ln(2n(1+1/\epsilon))/\pi}.$$

In particular, for any $\omega(\sqrt{\log n})$ function, there is a negligible $\epsilon = \epsilon(n)$ such that $\eta_\epsilon(\mathbb{Z}^n) \leq \omega(\sqrt{\log n})$.

3 Discrete Gaussian Sampling Algorithm

In this section we define and analyze the new sampling algorithm. In Section 3.1 we first present an abstract algorithm and prove its correctness. In Section 3.2 we then describe efficient and parallelizable instantiations of each abstract step of the algorithm, for both general lattices and those used in cryptography.

3.1 Abstract Algorithm

Algorithm 1 Abstract algorithm $\mathsf{SampleD}(\mathbf{B}_1, \Sigma, \mathbf{c})$ for sampling from a discrete Gaussian distribution.

Input: Basis $\mathbf{B}_1$ of a lattice $\Lambda = \mathcal{L}(\mathbf{B}_1)$, covariance matrix $\Sigma > \Sigma_1 = r^2 \cdot \mathbf{B}_1\mathbf{B}_1^t$ where $r = \omega(\sqrt{\log n})$, and vector $\mathbf{c} \in \mathbb{R}^n$.

Output: Vector $\mathbf{x} \in \Lambda + \mathbf{c}$ drawn from a distribution statistically close to $D_{\Lambda+\mathbf{c},\sqrt{\Sigma}}$.

1: Let $\Sigma_2 = \Sigma - \Sigma_1 > 0$, and compute some $\mathbf{B}_2 = \sqrt{\Sigma_2}$.
2: Let $\mathbf{t} \leftarrow D_{\sqrt{\Sigma_2}}$.
3: **return** $\mathbf{x} \leftarrow \mathbf{t} + D_{\Lambda+\mathbf{c}-\mathbf{t},\sqrt{\Sigma_1}}$.

Theorem 3.1 (Correctness of abstract algorithm). *Adopt the notation from Algorithm 1. The output distribution of $\mathsf{SampleD}(\mathbf{B}_1, \Sigma, \mathbf{c})$ is within $\mathrm{negl}(n)$ statistical distance of $D_{\Lambda+\mathbf{c},\sqrt{\Sigma}}$. Moreover, the conditional distribution of $\mathbf{t}$ given $\mathbf{x} = \bar{\mathbf{x}}$ is within $\mathrm{negl}(n)$ statistical distance of $\mathbf{z} + D_{\sqrt{\Sigma_3}}$, where $\mathbf{z} = (\mathbf{I} + \Sigma_1\Sigma_2^{-1})^{-1} \cdot \bar{\mathbf{x}}$ and $\Sigma_3^{-1} = \Sigma_1^{-1} + \Sigma_2^{-1}$.*

Proof. By construction, Algorithm 1 outputs an element of $\Lambda + \mathbf{c}$. Now for all $\bar{\mathbf{x}} \in \Lambda + \mathbf{c}$, we have

$$\begin{aligned}
&\Pr[\mathsf{SampleD}(\mathbf{B}_1, \Sigma, \mathbf{c}) \text{ outputs } \bar{\mathbf{x}}] \\
&= \int_{\mathbf{t}\in\mathbb{R}^n} D_{\Lambda+\mathbf{c}-\mathbf{t},\sqrt{\Sigma_1}}(\bar{\mathbf{x}} - \mathbf{t}) \cdot D_{\sqrt{\Sigma_2}}(\mathbf{t}) \cdot d\mathbf{t} && \text{(by construction)} \\
&\propto \int_{\mathbf{t}\in\mathbb{R}^n} \frac{\rho_{\sqrt{\Sigma_1}}(\bar{\mathbf{x}} - \mathbf{t}) \cdot \rho_{\sqrt{\Sigma_2}}(\mathbf{t})}{\rho_{\sqrt{\Sigma_1}}(\Lambda + \mathbf{c} - \mathbf{t})} \cdot d\mathbf{t} && \text{(def. of } D_\Lambda \text{ and } D) \\
&\in [1, \tfrac{1+\epsilon}{1-\epsilon}] \cdot \frac{1}{\rho_{\sqrt{\Sigma_1}}(\Lambda)} \cdot \int_{\mathbf{t}\in\mathbb{R}^n} \rho_{\sqrt{\Sigma_1}}(\bar{\mathbf{x}} - \mathbf{t}) \cdot \rho_{\sqrt{\Sigma_2}}(\mathbf{t}) \cdot d\mathbf{t} && \text{(Lemma 2.3)} \\
&\propto [1, \tfrac{1+\epsilon}{1-\epsilon}] \cdot (\rho_{\sqrt{\Sigma_1}} * \rho_{\sqrt{\Sigma_2}})(\bar{\mathbf{x}}) && \text{(def. of convolution)} \\
&= [1, \tfrac{1+\epsilon}{1-\epsilon}] \cdot \rho_{\sqrt{\Sigma}}(\bar{\mathbf{x}}). && \text{(Lemma 2.1)}
\end{aligned}$$

Note that we can apply Lemma 2.3 above because $r\mathbf{B}_1 = \sqrt{\Sigma_1} \geq \eta_\epsilon(\Lambda)$ for some $\epsilon(n) = \mathrm{negl}(n)$, by Lemma 2.4 and the fact that $\mathbf{B}_1^{-1} \cdot \Lambda = \mathbb{Z}^n$.

For the conditional distribution, let $\bar{\mathbf{x}} \in \Lambda + \mathbf{c}$ be arbitrary. Then for any $\mathbf{t} \in \mathbb{R}^n$, we have

$$\begin{aligned}
&\Pr\left[\mathbf{t} = \bar{\mathbf{t}} \mid \mathbf{x} = \bar{\mathbf{x}}\right] \\
&\propto \Pr\left[\mathbf{t} = \bar{\mathbf{t}} \wedge \mathbf{x} = \bar{\mathbf{x}}\right] && \text{(Bayes rule)} \\
&= \rho_{\sqrt{\Sigma_2}}(\bar{\mathbf{t}}) \cdot \frac{\rho_{\sqrt{\Sigma_1}}(\bar{\mathbf{x}} - \bar{\mathbf{t}})}{\rho_{\sqrt{\Sigma_1}}(\Lambda + \mathbf{c} - \bar{\mathbf{t}})} && \text{(by construction)} \\
&\in [1, \tfrac{1+\epsilon}{1-\epsilon}] \cdot \frac{1}{\rho_{\sqrt{\Sigma_1}}(\Lambda)} \cdot (\rho_{\sqrt{\Sigma_2}} \cdot \rho_{\sqrt{\Sigma_1},\bar{\mathbf{x}}})(\bar{\mathbf{t}}), && \text{(Lemma 2.3)}
\end{aligned}$$

where $\rho_{\sqrt{\Sigma_1},\bar{\mathbf{x}}}$ is the Gaussian function centered at $\bar{\mathbf{x}}$. It is well-known (see, e.g., [Tou09]) that the product of the two Gaussian functions $\rho_{\sqrt{\Sigma_2}}$ and $\rho_{\sqrt{\Sigma_1},\bar{\mathbf{x}}}$ is the (centered) Gaussian function $\rho_{\sqrt{\Sigma_3},\mathbf{z}}$, where $\mathbf{z}$ and Σ_3 are as in the theorem statement. The claim follows. □

3.2 Efficient Instantiations

Here we describe how the abstract Algorithm 1 admits efficient implementations that can make extensive use of preprocessing and parallelism. For the common case where $\Sigma = s^2$ for some $s > \sigma_1(\mathbf{B}_1) \cdot \omega(\sqrt{\log n})$, we combine all the ideas to describe a complete concrete instantiation in Algorithm 2.

3.2.1 Preprocessing Step 1

In a typical application, $\mathsf{SampleD}(\mathbf{B}_1, \Sigma, \mathbf{c})$ is invoked several times on the same basis $\mathbf{B}_1$ and covariance matrix Σ, but different values of $\mathbf{c}$. For example, in digital signature and IBE schemes [GPV08, Pei09, CHK09], the point $\mathbf{c}$ varies with the message (or identity), but the secret basis $\mathbf{B}_1$ and covariance matrix $\Sigma = s^2$ remain the same. Therefore, Step 1 is most naturally implemented in an offline preprocessing stage, and the output $\mathbf{B}_2$ can be retained for all future invocations.

There are a few ways to compute a matrix $\mathbf{B}_2 = \sqrt{\Sigma_2}$. One way is to compute the Cholesky decomposition $\Sigma_2 = \mathbf{L}\mathbf{L}^t$, where $\mathbf{L}$ is lower-triangular, and let $\mathbf{B}_2 = \mathbf{L}$. This method may be preferred if space is constrained, as $\mathbf{L}$ takes only about half as much storage as a dense matrix.

Another method, which is particularly attractive in the typical case where $\Sigma = s^2$ for some real $s > r \cdot \sigma_1(\mathbf{B}_1)$, is to use the singular value decomposition of $\mathbf{B}_1 = \mathbf{Q}\mathbf{D}\mathbf{P}^t$, where $\mathbf{Q}$ and $\mathbf{P}$ are orthogonal matrices and $\mathbf{D}$ is a diagonal matrix having positive diagonal entries $\sigma_i = \sigma_i(\mathbf{B}_1) < s$. Then we may define

$$\mathbf{B}_2 = \mathbf{Q}\sqrt{s^2 - (r\mathbf{D})^2} = \sqrt{\Sigma_2},$$

because

$$r^2 \cdot \mathbf{B}_1\mathbf{B}_1^t + \mathbf{B}_2\mathbf{B}_2^t = \mathbf{Q}(r\mathbf{D})^2\mathbf{Q}^t + \mathbf{Q}(s^2 - (r\mathbf{D})^2)\mathbf{Q}^t = s^2 \cdot \mathbf{Q}\mathbf{Q}^t = \Sigma.$$

In particular, for the matrix square root $\sqrt{s^2 - (r\mathbf{D})^2}$ we can use the diagonal matrix having positive diagonal entries $\sqrt{s^2 - (r\sigma_i)^2}$. This choice has additional benefits; see the next step.

3.2.2 Preprocessing Step 2

Given a matrix $\mathbf{B}_2 = \sqrt{\Sigma_2}$ as computed in Step 1, generating $\mathbf{t} \leftarrow D_{\sqrt{\Sigma_2}}$ can be done simply by choosing $\mathbf{v} \leftarrow D_1$ and letting $\mathbf{t} = \mathbf{B}_2\mathbf{v}$. Each coordinate of $\mathbf{v}$ is independently distributed as D_1 in one dimension, and the matrix-vector product $\mathbf{B}_2\mathbf{v}$ can be computed in parallel in the standard way.

Moreover, when $\mathbf{B}_2 = \mathbf{Q}\sqrt{s^2 - (r\mathbf{D})^2} = \sqrt{\Sigma_2}$ as described above, the random variable $\mathbf{t} \leftarrow D_{\sqrt{\Sigma_2}}$ is of the form $\mathbf{Q}\mathbf{v}'$, where the coordinates v_i' are independently distributed as the one-dimensional Gaussians $D_{\sqrt{s^2-\sigma_i^2}}$, respectively.

Finally, observe that this step is independent of the input $\mathbf{c}$ to SampleD, so it can also be precomputed in an offline phase. Of course, each call to SampleD must use a fresh value of $\mathbf{t}$.

3.2.3 Instantiating Step 3

For this step, we first define a Gaussian 'randomized rounding' operation $\lfloor v \rceil_r$ for arbitrary $v \in \mathbb{R}$ and real $r > 0$. The output of this operation is a random variable over $\mathbb{Z}$, distributed as $v + D_{\mathbb{Z}-v,r}$. We extend this operation to vectors $\mathbf{v} \in \mathbb{R}^n$, where the ith coordinate of $\lfloor \mathbf{v} \rceil_r$ is independently distributed as $\lfloor v_i \rceil_r$. It follows that for any $\bar{\mathbf{z}} \in \mathbb{Z}^n$, the probability that $\lfloor \mathbf{v} \rceil_r = \bar{\mathbf{z}}$ is proportional to

$$\prod_{i\in[n]} \rho_r(\bar{z}_i - v_i) = \exp(-\pi \cdot \sum_{i\in[n]} (\bar{z}_i - v_i)^2/r^2) = \exp(-\pi \cdot \|\bar{\mathbf{z}} - \mathbf{v}\|^2/r^2) = \rho_r(\bar{\mathbf{z}} - \mathbf{v}).$$

That is, $\lfloor \mathbf{v} \rceil_r$ is distributed as $\mathbf{v} + D_{\mathbb{Z}^n-\mathbf{v},r}$, because the standard basis for $\mathbb{Z}^n$ is orthonormal.

In [GPV08] it is shown how to sample from $D_{\mathbb{Z}-v,r}$ for any $v \in \mathbb{R}$ and $r > 0$, by rejection sampling. In our setting, though, we can be more efficient by exploiting the fact that r is fixed, known in advance, and small (slightly super-$\sqrt{\log n}$). For instance, for any $v \in [0, 1)$ we can (pre)compute a table of the cumulative distribution function, i.e., the probabilities that $v + D_{\mathbb{Z}-v,r} \le \bar{z} \in \mathbb{Z}$, for each $\bar{z}$ within an interval $\pm\omega(\log n)$. (Outside of that interval are the negligibly rare tails of the distribution.) Then we can directly sample from $v + D_{\mathbb{Z}-v,r}$ by choosing a uniform value in $[0, 1)$ and performing a binary search through the table.

Finally, to perform Step 3 we can sample from $D_{\Lambda+\mathbf{c}-\mathbf{t},\sqrt{\Sigma_1}}$ simply by invoking the following lemma with $\mathbf{w} = \mathbf{c} - \mathbf{t}$. Note that the randomized rounding can be performed on each coordinate in parallel.

Lemma 3.2. *Let $\mathbf{B}_1$, $\Lambda = \mathcal{L}(\mathbf{B}_1)$, and $\Sigma_1 = r^2 \cdot \mathbf{B}_1\mathbf{B}_1^t$ be as in Algorithm 1 for any real $r > 0$, and let $\mathbf{w} \in \mathbb{R}^n$ be arbitrary. The random variable $\mathbf{x} = \mathbf{w} - \mathbf{B}_1\lfloor \mathbf{B}_1^{-1}\mathbf{w} \rceil_r$ has distribution $D_{\Lambda+\mathbf{w},\sqrt{\Sigma_1}}$.*

Proof. Let $\mathbf{v} = \mathbf{B}_1^{-1}\mathbf{w}$. The support of $\lfloor \mathbf{v} \rceil_r$ is $\mathbb{Z}^n$, so the support of $\mathbf{x}$ is $\mathbf{w} - \mathbf{B}_1 \cdot \mathbb{Z}^n = \Lambda + \mathbf{w}$. Now for any $\bar{\mathbf{x}} = \mathbf{w} - \mathbf{B}_1\bar{\mathbf{z}}$ where $\bar{\mathbf{z}} \in \mathbb{Z}^n$, we have $\mathbf{x} = \bar{\mathbf{x}}$ if and only if $\lfloor \mathbf{v} \rceil_r = \bar{\mathbf{z}}$. As desired, this event occurs with probability proportional to

$$\rho_r(\bar{\mathbf{z}} - \mathbf{v}) = \rho_r(\mathbf{B}_1^{-1}(\mathbf{w} - \bar{\mathbf{x}}) - \mathbf{B}_1^{-1}\mathbf{w}) = \rho_r(-\mathbf{B}_1^{-1}\bar{\mathbf{x}}) = \rho_{r\mathbf{B}_1}(\bar{\mathbf{x}}) = \rho_{\sqrt{\Sigma_1}}(\bar{\mathbf{x}}). \qquad \square$$

Algorithm 2 Concrete algorithm $\mathsf{SampleD}(\mathbf{B}_1, s, \mathbf{c})$ for sampling from a discrete Gaussian distribution.

Input: Basis $\mathbf{B}_1$ of a lattice $\Lambda = \mathcal{L}(\mathbf{B}_1)$, real parameter $s > r \cdot \sigma_1(\mathbf{B}_1)$ where $r = \omega(\sqrt{\log n})$, and $\mathbf{c} \in \mathbb{R}^n$.
Output: Vector $\mathbf{x} \in \Lambda + \mathbf{c}$ drawn from a distribution statistically close to $D_{\Lambda+\mathbf{c},s}$.

Preprocessing phase:

1: Compute a singular value decomposition $\mathbf{B}_1 = \mathbf{QDP}^t$, and let $\mathbf{D}' = \sqrt{(s/\mathbf{D})^2 - r^2}$.
2: Choose a fresh $\mathbf{y} \leftarrow \mathbf{P} \cdot \mathbf{D}' \cdot D_1$ for each desired sample.

Online phase:

3: **return** $\mathbf{c} - \mathbf{B}_1\lfloor \mathbf{B}_1^{-1}\mathbf{c} - \mathbf{y} \rceil_r$, using a fresh $\mathbf{y}$ from the preprocessing phase.

4 Singular Value Bounds

In this section we give bounds on the largest singular value of a basis $\mathbf{B}$ and relate them to other geometric quantities that are relevant to the prior sampling algorithm of [GPV08].

4.1 General Bounds

The *Gram-Schmidt orthogonalization* of a nonsingular matrix $\mathbf{B}$ is $\mathbf{B} = \mathbf{QG}$, where $\mathbf{Q}$ is an orthogonal matrix and $\mathbf{G}$ is right-triangular, with positive diagonal entries $g_{i,i} > 0$ (without loss of generality). The Gram-Schmidt vectors for $\mathbf{B}$ are $\widetilde{\mathbf{b}_i} = g_{i,i} \cdot \mathbf{q}_i$. That is, $\widetilde{\mathbf{b}_1} = \mathbf{b}_1$, and $\widetilde{\mathbf{b}_i}$ is the component of $\mathbf{b}_i$ orthogonal to the linear span of $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$. The Gram-Schmidt orthogonalization can be computed efficiently in a corresponding iterative manner.

Let $\mathbf{B} = \mathbf{QG}$ be the Gram-Schmidt orthogonalization of $\mathbf{B}$. Without loss of generality we can assume that $\mathbf{B}$ is *size-reduced*, i.e., that $|g_{i,j}| \leq g_{i,i}/2$ for every $i < j$. This condition can be achieved efficiently by the following process: for each $j = 1, \ldots, n$, and for each $i = j-1, \ldots, 1$, replace $\mathbf{b}_j$ by $\mathbf{b}_j - \lfloor g_{i,j}/g_{i,i} \rceil \cdot \mathbf{b}_i$. Note that the size reduction process leaves the lattice $\mathcal{L}(\mathbf{B})$ and Gram-Schmidt vectors $\widetilde{\mathbf{b}_i} = g_{i,i} \cdot \mathbf{q}_i$ unchanged. Note also that $\|\mathbf{g}_i\| \leq \sqrt{n} \cdot \max_i g_{i,i}$, by the Pythagorean theorem.

Lemma 4.1. *Let $\mathbf{B} \in \mathbb{R}^{n \times n}$ be a size-reduced nonsingular matrix. We have*

$$\sigma_1(\mathbf{B}) \leq \sqrt{n} \cdot \sqrt{\sum_{i \in [n]} \|\widetilde{\mathbf{b}_i}\|^2} \leq n \cdot \|\widetilde{\mathbf{B}}\|.$$

The lemma is tight up to a constant factor, which may be seen by considering the right-triangular matrix with 1s on the diagonal and $1/2$ in every entry above the diagonal.

Proof. Let $\mathbf{B}$ have Gram-Schmidt orthogonalization $\mathbf{B} = \mathbf{QG}$. We have

$$\sigma_1(\mathbf{B}) = \max_{\mathbf{x}} \|\mathbf{B}^t\mathbf{x}\| = \max_{\mathbf{x}} \|\mathbf{G}^t\mathbf{x}\| \leq \sqrt{\sum_{i \in [n]} (\sqrt{n} \cdot g_{i,i})^2} = \sqrt{n} \cdot \sqrt{\sum_{i \in [n]} g_{i,i}^2},$$

where the maxima are taken over all unit vectors $\mathbf{x} \in \mathbb{R}^n$, the second equality uses the fact that $\mathbf{Q}$ is orthogonal, and the first inequality is by Cauchy-Schwarz. $\square$

4.2 Bases for Cryptographic Lattices

Ajtai [Ajt99] gave a procedure for generating a lattice uniformly from a certain family of cryptographic (or 'worst-case-hard') lattices, together with a relatively short basis. Alwen and Peikert [AP09] recently improved and extended the construction. Here we show that one of the constructions of [AP09] yields (with overwhelming probability) a basis whose largest singular value is only an $O(\sqrt{n})$ factor larger than the optimal $\|\widetilde{\mathbf{B}}\|$ over *any* basis $\mathbf{B}$ of the lattice. (Note that it is standard to use m as the dimension of the cryptographic lattice, and n as the dimension of the underlying worst-case lattices. For consistency with the rest of this paper, here we instead use n as the dimension of the random lattice.)

Our tightest bounds apply to the construction from [AP09, Section 3.2], instantiated with base $r = 2$, and slightly modified to use a uniformly random ± 1 (rather than 0-1) component matrix $\mathbf{R}$ and odd modulus q.

Lemma 4.2. *The construction from [AP09, Section 3.2] (modified as described above) outputs a basis* $\mathbf{S}$ *such that* $\sigma_1(\mathbf{S}) \leq O(n)$ *with overwhelming probability.*

It is known that, with high probability over the random lattice Λ chosen from the hard family, $\|\widetilde{\mathbf{B}}\| \geq \Omega(\sqrt{n})$ for *every* basis $\mathbf{B}$ of Λ. This implies the claimed $O(\sqrt{n})$ gap.

Proof. The construction of [AP09, Section 3.2] output a basis $\mathbf{S}$ with block structure

$$\mathbf{S} = \begin{pmatrix} (\mathbf{G} + \mathbf{R})\mathbf{B} & \mathbf{RP} - \mathbf{I} \\ \mathbf{B} & \mathbf{P} \end{pmatrix} \in \mathbb{Z}^{n \times n}.$$

The exact definitions of the components are not too important for our purposes; here we only need the following facts:

- $\sigma_1(\mathbf{B}) \leq 3$, and $\sigma_1(\mathbf{GB}) \leq O(n)$ because $\mathbf{GB}$ is a 0-1 matrix.
- $\sigma_1(\mathbf{R}) \leq O(\sqrt{n})$ with overwhelming probability, because $\mathbf{R}$ is a uniformly random ± 1 matrix.
- $\sigma_1(\mathbf{P}) = 1$ because $\mathbf{P}$ has a single 1 in each column and 0s elsewhere.

By the triangle inequality and the definition of σ_1, it is straightforward to verify that $\sigma_1(\mathbf{S}) \leq O(n)$. $\square$

4.3 Gaussian-Distributed Bases

Here we show that for a lattice basis generated by choosing its vectors from a discrete Gaussian distribution over the lattice (following by some post-processing), the largest singular value $\sigma_1(\mathbf{B})$ of the resulting basis is essentially the same as the maximal Gram-Schmidt length $\|\widetilde{\mathbf{B}}\|$ (with high probability). Such a bound is useful because applications that use 'basis delegation,' such as the hierarchical ID-based encryption schemes of [Pei09, CHK09], generate random bases in exactly the manner just described. Our bounds imply that the efficient Gaussian sampling algorithm is essentially as tight as the GPV algorithm on such bases.

Algorithm 3 recalls the precise method for generating and post-processing a Gaussian-distributed basis.

Theorem 4.3. *With overwhelming probability, Algorithm 3 outputs a basis* $\mathbf{T}$ *such that* $\|\widetilde{\mathbf{T}}\| \geq s \cdot \Omega(\sqrt{n})$*, and for any* $\omega(\sqrt{\log n})$ *function,* $\sigma_1(\mathbf{T}) \leq s \cdot O(\sqrt{n}) \cdot \omega(\sqrt{\log n})$*. In particular,* $\sigma_1(\mathbf{T})/\|\widetilde{\mathbf{T}}\| = \omega(\sqrt{\log n})$*.*

Algorithm 3 Abstract algorithm for sampling and post-processing a Gaussian-distributed basis.

Input: An arbitrary basis $\mathbf{B}$ of a lattice Λ, and an oracle for $D_{\Lambda,s}$, where $s \geq \eta_\epsilon(\Lambda)$ for some $\epsilon = \mathrm{negl}(n)$.
Output: A basis $\mathbf{T}$ of Λ.

$i \leftarrow 0$
repeat
  Draw a fresh $\mathbf{s} \leftarrow D_{\Lambda,s}$.
  if $\mathbf{s}$ is linearly independent of $\{\mathbf{s}_1, \ldots, \mathbf{s}_i\}$ **then**
    $i \leftarrow i+1$, $\mathbf{s}_i \leftarrow \mathbf{s}$
  end if
until $i = n$
return $\mathsf{ToBasis}(\mathbf{S}, \mathbf{B})$

We use the remainder of this subsection to prove the theorem. First we recall the algorithm ToBasis, used by Algorithm 3 to transform a *full-rank set* $\mathbf{S}$ of lattice vectors into a *basis* $\mathbf{T}$ of the lattice, without increasing the Gram-Schmidt lengths of the vectors; here we show that it also does not increase the largest singular value of the matrix. This means that it is enough to consider the largest singular value of $\mathbf{S}$.

Lemma 4.4. *There is a deterministic polynomial-time algorithm* $\mathsf{ToBasis}(\mathbf{S}, \mathbf{B})$ *that, given a full-rank set of lattice vectors* $\mathbf{S} \subset \Lambda$ *and an arbitrary basis* $\mathbf{B}$ *of* $\Lambda = \mathcal{L}(\mathbf{B})$*, outputs a basis* $\mathbf{T}$ *of* Λ *such that* $\sigma_1(\mathbf{T}) \leq \sigma_1(\mathbf{S})$*, and* $\|\widetilde{\mathbf{t}_i}\| \leq \|\widetilde{\mathbf{s}_i}\|$ *for all* i*.*

Proof. The algorithm works as follows: write $\mathbf{S} = \mathbf{BZ}$ for some nonsingular integer matrix $\mathbf{Z}$. Decompose $\mathbf{Z} = \mathbf{UR}$ for a unimodular matrix $\mathbf{U}$ and (nonsingular) right-triangular integer matrix $\mathbf{R}$. Output $\mathbf{T} = \mathbf{BU}$.

Clearly $\mathbf{T}$ is a basis of Λ, because $\mathbf{U}$ is unimodular. Observe that $\mathbf{T} = \mathbf{SR}^{-1}$. Now because $\mathbf{R}$ is a (nonsingular) triangular integral matrix, all its singular values $\sigma_i(\mathbf{R}) \geq 1$, hence every $\sigma_i(\mathbf{R}^{-1}) \leq 1$. We conclude that $\sigma_1(\mathbf{T}) \leq \sigma_1(\mathbf{S}) \cdot \sigma_1(\mathbf{R}^{-1}) \leq \sigma_1(\mathbf{S})$.

For the Gram-Schmidt lengths, let $\mathbf{S} = \mathbf{QG}$ be the G-S decomposition of $\mathbf{S}$, where $\mathbf{G}$ is right-triangular. Then $\mathbf{T} = \mathbf{Q}(\mathbf{GR}^{-1})$ is the G-S decomposition of $\mathbf{T}$, because $\mathbf{R}^{-1}$ is also right-triangular. The ith diagonal entry of $\mathbf{R}^{-1}$ is $r_{i,i}^{-1}$, hence $\|\widetilde{\mathbf{t}_i}\| \leq \|\widetilde{\mathbf{s}_i}\|/|r_{i,i}| \leq \|\widetilde{\mathbf{s}_i}\|$. □

We next prove the lower bound on $\|\widetilde{\mathbf{T}}\|$. First we claim that in the decomposition $\mathbf{S} = \mathbf{BUR}$ above, $|r_{1,1}| = 1$ with overwhelming probability. This is because $\mathbf{s}_1 \in |r_{1,1}| \cdot \Lambda$, and the probability that $D_{\Lambda,s}$ outputs an element in $r \cdot \Lambda$ for an integer $r > 1$ is negligible: the probability is maximized for $r = 2$, and is $(1 + \mathrm{negl}(n)) \cdot 2^{-n}$ in that case by the fact that there are 2^n cosets of 2Λ, and by Lemma 2.3. Therefore, $\widetilde{\mathbf{t}_1} = \mathbf{t}_1 = \mathbf{s}_1$. By [MR07, Lemma 4.2], we know that $\mathbf{s}_1$ has length $s \cdot \Omega(\sqrt{n})$ with overwhelming probability, and hence $\|\widetilde{\mathbf{T}}\| \geq \|\widetilde{\mathbf{t}_1}\| = s \cdot \Omega(\sqrt{n})$.

We now work to prove the upper bound on $\sigma_1(\mathbf{S})$. The next lemma bounds the singular values of a (possibly non-square) matrix whose columns are drawn from $D_{\Lambda,s}$.

Lemma 4.5. *Let* $\Lambda \subset \mathbb{R}^n$ *be a lattice and let* $s \geq \eta_\epsilon(\Lambda)$ *for some* $\epsilon = \mathrm{negl}(n)$*. Let* $\mathbf{S}' \in \mathbb{R}^{n \times m}$ *be a matrix whose* m *columns* $\mathbf{s}'_i$ *are drawn independently from* $D_{\Lambda,s}$*. Then*

$$\sigma_1(\mathbf{S}') \leq s \cdot O(\sqrt{n} + \sqrt{m})$$

with overwhelming probability (where O *hides a universal constant).*

Proof. We give an outline of the proof, whose details are straightforward but somewhat tedious. Without loss of generality, assume that $s = 1$. The first fact we need is that projecting the distribution D_Λ onto any one dimension (specified by a unit vector $\mathbf{u}$) yields a *subgaussian* distribution, i.e., one whose tails outside the interval $[-t, +t]$ carry at most $C \cdot \exp(-\pi t^2)$ of the probability mass, for some fixed constant C and any $t > 0$. This fact is established in [Pei08, Lemma 5.1], using techniques of [Ban95].

The second fact we need is that an n-by-m matrix with independent subgaussian columns has largest singular value $O(\sqrt{n} + \sqrt{m})$ with overwhelming probability. This can be established using a standard ϵ-net argument; see for example [Ver07, Lecture 6]. □

Now let $\mathbf{S}' \in \mathbb{R}^{n \times m}$ be the matrix consisting of *every* vector $\mathbf{s} \leftarrow D_{\Lambda,s}$ chosen by Algorithm 3, irrespective of whether it is linearly independent of its predecessors. Because $\mathbf{S}$ is made up of a subset of the columns of $\mathbf{S}'$, it follows immediately from the definition that $\sigma_1(\mathbf{S}) \leq \sigma_1(\mathbf{S}')$.

It simply remains to bound the total number m of samples that Algorithm 3 draws from $D_{\Lambda,s}$. By [Reg09, Lemma 3.15], each sample $\mathbf{s}$ is linearly independent of $\mathbf{s}_1, \ldots, \mathbf{s}_i$ with probability at least $1/10$. Therefore, for any $\omega(\log n)$ function, the algorithm draws a total of $n \cdot \omega(\log n)$ samples with all but some negligible probability. By Lemma 4.5, we conclude that $\sigma_1(\mathbf{S}') \leq s \cdot O(\sqrt{n}) \cdot \omega(\sqrt{\log n})$ with overwhelming probability. This completes the proof.

References

[Ajt99] Miklós Ajtai. Generating hard instances of the short basis problem. In *ICALP*, pages 1–9, 1999.

[Ajt04] Miklós Ajtai. Generating hard instances of lattice problems. *Quaderni di Matematica*, 13:1–32, 2004. Preliminary version in STOC 1996.

[AP09] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. In *STACS*, pages 75–86, 2009.

[Bab86] László Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[Ban95] Wojciech Banaszczyk. Inequalites for convex bodies and polar reciprocal lattices in R^n. *Discrete & Computational Geometry*, 13:217–231, 1995.

[CHK09] David Cash, Dennis Hofheinz, and Eike Kiltz. How to delegate a lattice basis. Cryptology ePrint Archive, Report 2009/351, July 2009. `http://eprint.iacr.org/`.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

[GN08] Nicolas Gama and Phong Q. Nguyen. Predicting lattice reduction. In *EUROCRYPT*, pages 31–51, 2008.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

[HHGP+03] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. NTRUSIGN: Digital signatures using the NTRU lattice. In *CT-RSA*, pages 122–140, 2003.

[Kle00] Philip N. Klein. Finding the closest lattice vector when it's unusually close. In *SODA*, pages 937–941, 2000.

[Ksh59] A. M. Kshirsagar. Bartlett decomposition and Wishart distribution. *The Annals of Mathematical Statistics*, 30(1):239–241, March 1959. Available at `http://www.jstor.org/stable/2237140`.

[LM08] Vadim Lyubashevsky and Daniele Micciancio. Asymptotically efficient lattice-based digital signatures. In *TCC*, pages 37–54, 2008.

[MPSW09] Tal Malkin, Chris Peikert, Rocco A. Servedio, and Andrew Wan. Learning an overcomplete basis: Analysis of lattice-based signatures with perturbations. Manuscript, 2009.

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007. Preliminary version in FOCS 2004.

[MR09] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post Quantum Cryptography*, pages 147–191. Springer, February 2009.

[NR09] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. *J. Cryptology*, 22(2):139–160, 2009. Preliminary version in Eurocrypt 2006.

[Pei08] Chris Peikert. Limits on the hardness of lattice problems in ℓ_p norms. *Computational Complexity*, 17(2):300–351, May 2008. Preliminary version in CCC 2007.

[Pei09] Chris Peikert. Bonsai trees (or, arboriculture in lattice-based cryptography). Cryptology ePrint Archive, Report 2009/359, July 2009. `http://eprint.iacr.org/`.

[PV08] Chris Peikert and Vinod Vaikuntanathan. Noninteractive statistical zero-knowledge proofs for lattice problems. In *CRYPTO*, pages 536–553, 2008.

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009. Preliminary version in STOC 2005.

[Tou09] Marc Toussaint. Lecture notes: Gaussian identities, 2009. Available at `user.cs.tu-berlin.de/~mtoussai/notes/gaussians.pdf`, last accessed 17 Feb 2010.

[Ver07] Roman Vershynin. Lecture notes on non-asymptotic theory of random matrices, 2007. Available at `http://www-personal.umich.edu/~romanv/teaching/2006-07/280/`, last accessed 17 Feb 2010.