Lattice Trapdoors and Gaussian Sampling

We will work with the ℓ_∞ norm throughout these lecture notes; tighter bounds are sometimes possible with the Euclidean norm but we would like to avoid the complication of computing the exact factors in favor of simplicity and conceptual clarity.

1 Lattice Trapdoors

Recall that

$$\Lambda^\perp(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m \,:\, \mathbf{Az} = \mathbf{0} \pmod{q}\}$$

is a rank-m lattice. We will define a lattice trapdoor for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ to be a short basis for $\Lambda^\perp(\mathbf{A})$. More generally, a set of short linearly independent vectors in $\Lambda^\perp(\mathbf{A})$ suffices. More explicitly:

Definition 1. *A matrix $\mathbf{T} \in \mathbb{Z}^{m\times m}$ is a β-good lattice trapdoor for a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ if*

1. *Each column vector of $\mathbf{T}$ is in the (right) mod-q kernel of $\mathbf{A}$, namely, $\mathbf{AT} = \mathbf{0} \pmod{q}$;*
2. *Each column vector of $\mathbf{T}$ is short, namely $\|\mathbf{T}\|_\infty \le \beta$; and*
3. *$\mathbf{T}$ has rank m over $\mathbb{R}$.*

Note that the rank of $\mathbf{T}$ over $\mathbb{Z}_q$ can be no more than $m-n$; so, at first sight, the first and the third conditions may appear to be contradictory. However, the fact that we require *the real rank* over $\mathbf{T}$ to be large is the crucial thing here. This is related to why $\Lambda^\perp(\mathbf{A})$ as a lattice has rank m, even though as a linear subspace of $\mathbb{Z}_q^m$ has rank only $m-n$. Another way to look at $\mathbf{T}$ is that each of its columns is a homogenous SIS solution with respect to $\mathbf{A}$.

What good is such a trapdoor? We will demonstrate (in Section 3) its usefulness by showing that it can be used to solve both LWE and (inhomogenous) SIS with respect to $\mathbf{A}$.

2 Trapdoor Sampling

2.1 Leftover Hash Lemma

We will use the following form of the leftover hash lemma.

Lemma 2. *Let P be a probability distribution over $\mathbb{Z}^m$. The following two distributions have statistical distance at most ε as long as $H_\infty(P) \ge n\log q + 2\log(1/\varepsilon)$:*

$$(\mathbf{A}, \mathbf{Ae} \pmod{q}) \approx (\mathbf{A}, \mathbf{u})$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ is uniformly random, $\mathbf{e} \leftarrow X$ is drawn from the probability distribution P and $\mathbf{u} \leftarrow \mathbb{Z}_q^n$ is uniformly random. Here, $H_\infty(P)$ refers to the min-entropy of P.

For a proof, we refer the reader to these lecture notes.

2.2 Sampling a Random A with a Single Trapdoor Vector

Ajtai in 1996 gave us a procedure to sample a (statistically close to) uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ together with a single short vector $\mathbf{t} \in \mathbb{Z}^m$ such that $\mathbf{At} = \mathbf{0} \pmod q$. We begin our journey into trapdoors by describing this simple procedure.

1. Pick a uniformly random matrix $\mathbf{A}' \in \mathbb{Z}_q^{n\times(m-1)}$.
2. Pick a uniformly random vector $\mathbf{t}' \in \{0,1\}^{m-1}$.
3. Define
$$\mathbf{A} = [\mathbf{A}' \| -\mathbf{A}'\mathbf{t}' \pmod q)] \quad \text{and} \quad \mathbf{t} = \begin{bmatrix} \mathbf{t}' \\ 1 \end{bmatrix}$$
as the matrix and trapdoor vector, respectively.

It is clear that $\mathbf{t}$ is a short vector in the right kernel of $\mathbf{A}$ mod q. It remains to show that $\mathbf{A}$ is close to uniformly random, which reduces to showing that $\mathbf{A}'\mathbf{t}$ is close to uniform given $\mathbf{A}'$. This follows directly from the leftover hash lemma assuming that $m \geq n \log q + \lambda$.

More generally, if we let $||\mathbf{t}||_\infty \leq B$, then we need $m \geq n \log q / \log B + \lambda$.

2.3 Ajtai-MP Trapdoor Sampling

Now, one can try to extend the above procedure to sample $\mathbf{A}$ together with more and more short vectors until you reach m (hopefully) linearly independent vectors and then we have a trapdoor! However, this naïve idea fails to work. Indeed, letting $m^* := n \log q + \lambda$, we can generate a close to uniform matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times(m^*+\ell)}$ together with ℓ trapdoor vectors (We leave it as an exercise to the reader to figure out how.) However, this will never "catch up" as the number of trapdoor vectors (ℓ) always remains short of the rank ($m^* + \ell$).

We start with the observation that an "inhomogenous trapdoor" (a notion that we will define in a minute) will let us achieve our goals of solving LWE and SIS just as well. An inhomogenous trapdoor $\mathbf{T} \in \mathbb{Z}^{m\times n \log q}$ is a matrix with short columns such that $\mathbf{AT} = \mathbf{G} \pmod q$ where $\mathbf{G}$ is a *gadget matrix* defined as below:

$$\mathbf{g} := \begin{bmatrix} 1 & 2 & 4 & \dots & 2^{\lceil \log q \rceil - 1} \end{bmatrix} \in \mathbb{Z}_q^{1\times\lceil \log q\rceil} \text{ and } \mathbf{G} := \mathbf{I} \otimes \mathbf{g}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix. In other words, $\mathbf{G}$ is the block diagonal $n \times (n\lceil \log q \rceil)$ matrix with $\mathbf{g}$ in each of its diagonal blocks.

Why does this suffice to solve LWE and SIS? Let's do LWE first. Given $\mathbf{b}^T = \mathbf{s}^T\mathbf{A} + \mathbf{e}^T \pmod q$, we do

$$\mathbf{b}^T\mathbf{T} = (\mathbf{s}^T\mathbf{A} + \mathbf{e}^T)\mathbf{T} = \mathbf{s}^T\mathbf{G} + \mathbf{e}^T\mathbf{T} \pmod q$$

In other words, we just transformed an LWE sample relative to $\mathbf{A}$ into an LWE sample relative to $\mathbf{G}$, with a slight increase in error. Now, if we have a trapdoor (in the sense of Definition 1) for $\mathbf{G}$ (and we will show in a few minutes that we do indeed have such a trapdoor), we can solve LWE!

To solve SIS w.r.t. $\mathbf{G}$, we simply note that given a vector $\mathbf{v} \in \mathbb{Z}_q^n$, it is easy to compute a bit vector $\mathbf{e}' \in \{0,1\}^{m^*}$ such that $\mathbf{Ge}' = \mathbf{v} \pmod q$. Indeed $\mathbf{e}'$ is simply the $n\lceil \log q \rceil$-dimensional bit-vector that consists of the bit representations of each element in $\mathbf{v}$.

Trapdoor for G: The case of $q = 2^k$. We invite the reader to think about this a bit before reading on. Let us first construct a trapdoor $\mathbf{T_g} \in \mathbb{Z}^{\lceil \log q \rceil \times \lceil \log q \rceil}$. We will then see that $\mathbf{T_G} = \mathbf{I} \otimes \mathbf{T_g}$. Indeed,

$$\mathbf{G} \cdot \mathbf{T_G} = (\mathbf{I} \otimes \mathbf{g}) \cdot (\mathbf{I} \otimes \mathbf{T_g}) = \mathbf{I} \otimes (\mathbf{g}\mathbf{T_g}) = \mathbf{0} \pmod{q}$$

Here is the trapdoor for $\mathbf{g}$:

$$\mathbf{T_g} = \begin{bmatrix} 2 & & & & \\ -1 & 2 & & & \\ & -1 & \cdots & & \\ & & \ddots & & \\ & & & 2 & \\ & & & -1 & 2 \end{bmatrix}$$

Let us check.

- $\mathbf{T_g}$ has short columns. Indeed $||\mathbf{T_g}||_\infty = 2$.
- $\mathbf{g}\mathbf{T_g} = 0 \pmod{q}$.
- The determinant of $\mathbf{T_g}$ is $q = 2^k$. Therefore, it has full rank over $\mathbb{R}$. It decidedly *does not* have full rank over $\mathbb{Z}_q$ since its determinant is 0 mod q. (And this had better be the case!)

Trapdoor for G: The general case. As before, let us construct a trapdoor $\mathbf{T_g} \in \mathbb{Z}^{\lceil \log q \rceil \times \lceil \log q \rceil}$. We will then see that $\mathbf{T_G} = \mathbf{I} \otimes \mathbf{T_g}$. Here is the trapdoor for $\mathbf{g}$:

$$\mathbf{T_g} = \begin{bmatrix} 2 & & & & \vert \\ -1 & 2 & & & \vert \\ & -1 & \cdots & & \vert \\ & & \ddots & & \mathsf{bits}(q) \\ & & & 2 & \vert \\ & & & -1 & \vert \end{bmatrix}$$

The only difference is in the last column which is now the bit representation of the modulus q. Checking that this is indeed a trapdoor for $\mathbf{g}$ is left as an exercise. (Hint: for the full rank property, prove that the determinant of this matrix is q.)

Sampling A together with an Inhomogenous Trapdoor. Sample a uniformly random $\mathbf{B} \in \mathbb{Z}_q^{n \times m^*}$ where $m^* = n \log q + \lambda$ (as before). Set

$$\mathbf{A} = [\mathbf{B} || \mathbf{BR} + \mathbf{G}] \quad (\text{over } \mathbb{Z}_q)$$

where $\mathbf{R} \in \mathbb{Z}_q^{m^* \times m}$ is a uniformly random 0-1 matrix. Notice that

$$\mathbf{A} \cdot \begin{bmatrix} -\mathbf{R} \\ \mathbf{I} \end{bmatrix} = \mathbf{G} \pmod{q}$$

and since $||\mathbf{R}||_\infty \leq 1$, we have an inhomogenous trapdoor! Furthermore, $\mathbf{A}$ is close to random by leftover hash lemma (as before).

Now, one could stop here and directly use the inhomogenous trapdoor to solve LWE and SIS but we will go one step further and show how to get a trapdoor for $\mathbf{A}$.

Sampling A with a Trapdoor, Finally. First of all, we have

$$[\mathbf{B}||\mathbf{B}\mathbf{R}+\mathbf{G}]\cdot\begin{bmatrix}-\mathbf{R}\\ \mathbf{I}\end{bmatrix}=\mathbf{G}$$

Thus,

$$[\mathbf{B}||\mathbf{B}\mathbf{R}+\mathbf{G}]\cdot\begin{bmatrix}\mathbf{I} & -\mathbf{R}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}=[\mathbf{B}||\mathbf{G}]$$

Finally, multiplying this on the right by $\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ -\mathbf{G}^{-}(\mathbf{B}) & \mathbf{T}_{\mathbf{G}}\end{bmatrix}$, we get

$$[\mathbf{B}||\mathbf{B}\mathbf{R}+\mathbf{G}]\cdot\underbrace{\begin{bmatrix}\mathbf{I} & -\mathbf{R}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\cdot\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ -\mathbf{G}^{-}(\mathbf{B}) & \mathbf{T}_{\mathbf{G}}\end{bmatrix}}_{=\mathbf{T}_{\mathbf{A}}}=[\mathbf{B}||\mathbf{G}]\cdot\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ -\mathbf{G}^{-}(\mathbf{B}) & \mathbf{T}_{\mathbf{G}}\end{bmatrix}=\mathbf{0}\pmod q$$

Thus, the lattice trapdoor

$$\mathbf{T}_{\mathbf{A}}=\begin{bmatrix}\mathbf{I}+\mathbf{R}\mathbf{G}^{-}(\mathbf{B}) & -\mathbf{R}\mathbf{T}_{\mathbf{G}}\\ -\mathbf{G}^{-}(\mathbf{B}) & \mathbf{T}_{\mathbf{G}}\end{bmatrix}$$

We already saw that $\mathbf{A}\mathbf{T}_{\mathbf{A}}=\mathbf{0}\pmod q$. The ℓ_∞ norm of $\mathbf{T}_{\mathbf{A}}$ is $O(m)$. Finally, since $\mathbf{T}_{\mathbf{A}}$ is a product of two full-rank matrices, it is full-rank as well. (It has determinant q^n.)

3 Trapdoor Functions

Definition 3. *A family of functions*[1] $\mathcal{F}_n=\{f_i:\{0,1\}^n\to\{0,1\}^m\}$ *for some* $m=m(n)$ *is called a trapdoor function family if it comes with the following three associated polynomial-time algorithms.*

- *A probabilistic function generation algorithm that, on input* 1^n*, outputs an index* i *of a function* f_i *in the family as well as a trapdoor* t_i*.*
- *A deterministic evaluation algorithm that, on input* i *and* $x\in\{0,1\}^n$*, outputs* y*. We need that* $y=f_i(x)$*.*
- *A deterministic inversion algorithm that, on input* i*,* t_i *and* $y\in\{0,1\}^m$*, outputs* $x\in\{0,1\}^n$ *or a special symbol* $\perp$*. We require that if* $y\in\mathsf{Image}(f_i)$*, then* x *is* an *inverse, namely* $f_i(x)=y$*.*

3.1 Injective Trapdoor Function

The function

$$f_{\mathbf{A}}(\mathbf{s},\mathbf{e})=\mathbf{s}^T\mathbf{A}+\mathbf{e}^T\pmod q$$

where $\mathbf{A}\in\mathbb{Z}_q^n$, $\mathbf{s}\in\mathbb{Z}_q^n$ and $\mathbf{e}\leftarrow\chi^m$ is a one-way family of functions, under LWE. Given the trapdoor $\mathbf{T}$, one inverts this as follows.

$$(\mathbf{s}^T\mathbf{A}+\mathbf{e}^T)\mathbf{T}=\mathbf{e}^T\mathbf{T}\pmod q$$

Now, since the latter quantity has absolute value at most $q/4$, it is $\mathbf{e}^T\mathbf{T}$ (over the integers). The mod-q has no effect, and this is the key observation. Now, multiplying the latter by $\mathbf{T}^{-1}$ (the inverse of $\mathbf{T}$ over the reals) recovers $\mathbf{e}$. Here, it is *very* important that $\mathbf{T}$ had full rank over the reals; otherwise, $\mathbf{T}^{-1}$ would not exist.

[1] To be precise, we should be talking about ensemble of such families one for every input length n. However, we will refrain from unnecessary notational gymnastics and will take that as understood.

3.2 Surjective Trapdoor Function

The function

$$g_{\mathbf{A}}(\mathbf{e}) = \mathbf{A}\mathbf{e} \pmod{q}$$

where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ where $m > n \log q$ and $\mathbf{e} \in [-\beta, \beta]^m$ is a one-way family of functions as well, under SIS, where $\beta = \mathsf{poly}(m)$.

With an inhomogeneous trapdoor, we claim that we can easily invert the function. Indeed, given a vector $\mathbf{v} \in \mathbb{Z}_q^n$, and an inhomogeneous trapdoor $\mathbf{R}$, we first compute a bit vector $\mathbf{e}'$ such that $\mathbf{G}\mathbf{e}' = \mathbf{v} \bmod q$ as discussed above. Now, we claim that

$$\mathbf{e} := \mathbf{R} \cdot \mathbf{e}'$$

is a required inverse. Indeed,

$$\mathbf{A} \cdot \mathbf{R} \cdot \mathbf{e}' = \mathbf{G} \cdot \mathbf{e}' = \mathbf{v} \pmod{q}$$

Also $||\mathbf{e}||_\infty \leq m$ since each entry of $\mathbf{R}$ and $\mathbf{e}'$ are binary.

4 Digital Signatures

Here is a simple digital signature scheme. (For a definition of digital signatures and what we mean by a secure digital signature, see Rafael Pass and abhi shelat's book.)

- The key generation algorithm samples a function together with a trapdoor. This would be $\mathbf{A}$ and $\mathbf{T}$. The public key is $\mathbf{A}$ and the secret key is $\mathbf{T}$.

- To sign a message m, first map it into the range of the function, e.g., by hashing it. That is, compute $\mathbf{v} = H(m)$. The signature is an inverse of $\mathbf{v}$ under the function $g_{\mathbf{A}}$. That is, a short vector $\mathbf{e}$ such that $\mathbf{A}\mathbf{e} = \mathbf{v} \pmod{q}$. This is guaranteed by the surjectivity of the function $g_{\mathbf{A}}$.

- Verification, given a message m, public key $\mathbf{A}$ and signature $\mathbf{e}$, consists of checking that $\mathbf{A}\mathbf{e} = H(m) \pmod{q}$ and that $||\mathbf{e}||_\infty \leq m^2$.

Unforgeability (given no signature queries) reduces to SIS in the random oracle model, i.e., assuming that H is a random oracle.

However, given signatures on adversarially chosen messages (in fact, even random messages), this scheme is broken. The key issue is that there are many inverses of $H(m)$, and the particular inverse computed using a trapdoor $\mathbf{T}$ leaks information about $\mathbf{T}$. Collecting this leakage over sufficiently many (polynomially many) signature queries enables an adversary to find $\mathbf{T}$, allowing her to forge signatures at will going forward.

This is most easily seen when the inversion procedure for $g_{\mathbf{A}}$ uses the inhomogenous trapdoor. Note that given $\mathbf{v}$, an adversary can compute $\mathbf{G}^{-}(\mathbf{v}) = \mathbf{e}'$ herself. She now gets a signature

$$\sigma = \mathbf{R} \cdot \mathbf{e}'$$

which gives her one equation on the secret $\mathbf{R}$. Given about m equations, she can solve linear equations and learn $\mathbf{R}$.

The situation remains essentially as dire even if you use the trapdoor (as opposed to the inhomogenous trapdoor). Using rounding vs the nearest plane algorithm does not help either; see the paper of Nguyen and Regev for robust attacks against this signature scheme. The fundamental difficulty seems to stem from the fact that the inversion procedure is deterministic!

To mitigate the difficulty, we need a special kind of inverter for $g_{\mathbf{A}}$. The inverter is a "pre-image sampler"; that is, it is given the trapdoor $\mathbf{T}$ and produces a "random" pre-image. More precisely, we need the following distributions to be statistically close (computational indistinguishabilty is fine, but we will achieve statistical closeness):

$$\left(\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{e} \leftarrow D_{\mathbb{Z}^m,s}, \mathbf{v} := \mathbf{A}\mathbf{e} \pmod q\right)$$
$$\approx_s (\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \mathbf{e} \leftarrow \mathsf{PreSamp}(\mathbf{A}, \mathbf{T}, \mathbf{v}), \mathbf{v} \leftarrow \mathbb{Z}_q^n\Big)$$

That is, the following processes produce statistically close outputs: (a) first sample $\mathbf{e}$ from a discrete Gaussian, and deterministically set $\mathbf{v}$ to be $\mathbf{A}\mathbf{e} \pmod q$; and (b) sample $\mathbf{v}$ uniformly and use the pre-image sampler to produce an inverse of $\mathbf{v}$ under $g_{\mathbf{A}}$ that is distributed according to the right conditional distribution. This distribution happens to be the discrete Gaussian over a coset of the lattice, that is,

$$\Lambda_{\mathbf{v}}^{\perp}(\mathbf{A}) := \{\mathbf{e} \in \mathbb{Z}^m : \mathbf{A}\mathbf{e} = \mathbf{v} \pmod q\}$$

In fact, this not quite enough; we need a multi-sample version of this. That is,

$$\left(\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \{\mathbf{e}_i \leftarrow D_{\mathbb{Z}^m,s}, \mathbf{v} := \mathbf{A}\mathbf{e} \pmod q\}_{i=1}^{\mathsf{poly}(\lambda)}\right)$$
$$\approx_s (\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}, \{\mathbf{e} \leftarrow \mathsf{PreSamp}(\mathbf{A}, \mathbf{T}, \mathbf{v}), \mathbf{v} \leftarrow \mathbb{Z}_q^n\}_{i=1}^{\mathsf{poly}(\lambda)}\Big)$$

This is quite cumbersome to work with, so we propose an alternate stronger definition. That is, we require that for most $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ and any trapdoor $\mathbf{T}$ of length bounded by ℓ and $s \gg \ell$:

$$\left(\mathbf{e} \leftarrow D_{\mathbb{Z}^m,s}, \mathbf{v} := \mathbf{A}\mathbf{e} \pmod q\right)$$
$$\approx_s (\mathbf{e} \leftarrow \mathsf{PreSamp}(\mathbf{A}, \mathbf{T}, \mathbf{v}), \mathbf{v} \leftarrow \mathbb{Z}_q^n\Big)$$

Proof of Security. With the one change that the inverter is replaced by a pre-image sampler, our signature scheme becomes secure in the random oracle model. We showed the proof in the class.

5 Discrete Gaussian Sampling

Throughout, we will deal with sampling from a zero-centered discrete Gaussian.

5.1 Naïve Sampling

Let us first consider sampling from a discrete Gaussian over the simplest possible lattice, namely the one-dimensional lattice of integers $\mathbb{Z}$. The first idea to sample from the discrete Gaussian $D_{\mathbb{Z},s}$ is to sample from a continuous Gaussian N_s with parameter s and round to the nearest integer. Unfortunately, this is not a discrete Gaussian, not even statistically close to it. This is true even if s is much larger than the smoothing parameter. You will show this in the problem set.

For n-dimensional lattices, this statistical distance degrades with n as well making the situation much worse.

The reader may recall that the first step of Regev's worst-case to average-case reduction was sampling from a discrete Gaussian over a lattice for which Regev used the above procedure. However, he could afford to use an exponential s which makes the statistical distance small.

5.2 Sampling Discrete Gaussians over $\mathbb{Z}$

So, how do we sample from $D_{\mathbb{Z},s}$ for polynomial s? We will show that the general method of rejection sampling works. Let $Z = [-t \cdot s, t \cdot s]$ be a sufficiently large interval, where $t = \omega(\sqrt{\log \lambda})$. We do the following:

1. Sample a random integer $z \leftarrow Z$.
2. Output z with probability $\rho_s(z) := e^{-\pi z^2/s^2}$; else go to step 1 and repeat.

First of all, we will show that the probability that $D_{\mathbb{Z},s}$ assigns to numbers outside of the interval Z is negligible.

Lemma 4. *Let $s \geq \eta_\varepsilon(\mathbb{Z})$ for some $\varepsilon = \text{negl}(\lambda)$, and $t > 0$. We have*

$$\Pr_{x \leftarrow D_{\mathbb{Z},s}}[|x| > t \cdot s] \leq c \cdot e^{-\pi t^2}$$

for some absolute constant $c > 0$.

Consider the probability distribution $D'_{\mathbb{Z},s}$ which assigns probability $\rho_s(x)$ for all $x \in Z \cap \mathbb{Z}$ and 0 otherwise. The lemma above shows that $D'_{\mathbb{Z},s}$ is close to $D_{\mathbb{Z},s}$ if s is larger than the negl(λ)-smoothing parameter of $\mathbb{Z}$, namely $\omega(\sqrt{\log n})$, and $t = \omega(\sqrt{\log n})$.

It is not hard to see that the procedure above samples from the distribution $D'_{\mathbb{Z},s}$ exactly. It remains to see that it terminates in polynomial time. We show two things which we leave as an exercise: (a) the probability that z sampled in step 1 lies in $[-s, s]$ is $\Omega(1/t)$ and (b) if such a z is sampled, it is output with probability $\Omega(1)$. Put together, the expected time for termination is $O(t) = \mathsf{poly}(\lambda)$.

5.3 Klein-GPV algorithm

We demonstrate the sampler in two dimensions. The generalization to n dimensions follows quite naturally. [**Vinod:** TODO: Peikert's parallel sampler? or MP sampler?]