Lattices and Homomorphic Encryption, Spring 2013 — Instructors: Shai Halevi, Tal Malkin

A Special "Easy" Lattice

March 12, 2013

In this note we cover a few aspects of the special easy lattice of Micciancio-Peikert [1], which is used in their trapdoor construction. Below let n be the security parameter and let q be another parameter, polynomial in n. Denote $k = |q| = O(\log n)$ and let the binary representation of q be $q_{k-1} \dots q_1 q_0$, namely the q_i's are bits such that $q = \sum_{i=0}^{k-1} q_i 2^i$.

1 A Small Basis

A. Consider the vector $\vec{g} = \langle 1, 2, 4, \dots, 2^{k-1} \rangle \in \mathbb{Z}^k$, and the lattice $\mathcal{L}^\perp(\vec{g}) = \{\vec{x} \in \mathbb{Z}^k : \langle \vec{g}, \vec{x} \rangle = 0 \pmod q\}$. Prove that the columns of the following matrix S_k form a basis for $\mathcal{L}^\perp(\vec{g})$:

$$S_k \stackrel{\text{def}}{=} \begin{pmatrix} 2 & & & & & q_0 \\ -1 & 2 & & & & q_1 \\ & -1 & 2 & & & q_2 \\ & & \ddots & \ddots & & \\ & & & -1 & 2 & q_{k-2} \\ & & & & -1 & q_{k-1} \end{pmatrix} \tag{1}$$

Proof. It it easy to see that $\vec{g} \cdot S_k = 0 \pmod q$. Hence all the columns of S_k are in $\mathcal{L}^\perp(\vec{g})$, which means that $\mathcal{L}(S_k) \subseteq \mathcal{L}^\perp(\vec{g})$.

One way to see that the lattice $\mathcal{L}(S_k)$ in fact equals $\mathcal{L}^\perp(\vec{g})$ is to compute the determinants. On one hand the determinant of $\mathcal{L}^\perp(\vec{g})$ is exactly q, using the argument from class about a "full rank $1 \times k$ system of equations" modulo q. On the other hand, by iteratively adding twice row $i+1$ to row i (starting from the bottom row and going up), we can transform S_k into a matrix of the form

$$\begin{pmatrix} 0 & & & & & q \\ -1 & 0 & & & & * \\ & -1 & 0 & & & * \\ & & \ddots & \ddots & & \\ & & & -1 & 0 & * \\ & & & & -1 & * \end{pmatrix},$$

so clearly it has determinant $\pm q$. Hence $\mathcal{L}(S_k)$ cannot be a proper sublattice of $\mathcal{L}^\perp(\vec{g})$, so they must be equal. □

B. Consider the $n \times nk$ matrix

$$G \stackrel{\text{def}}{=} \begin{pmatrix} -\vec{g}- & & & \\ & -\vec{g}- & & \\ & & \ddots & \\ & & & -\vec{g}- \end{pmatrix} \tag{2}$$

Describe a basis for the lattice $\mathcal{L}^\perp(G) \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{Z}^{nk} : G\vec{x} = 0 \pmod q\}$. What is the determinant of this lattice?

Answer. A basis for $\mathcal{L}^\perp(G)$ with determinant q^n is $\begin{pmatrix} S_k & & \\ & \ddots & \\ & & S_k \end{pmatrix}$.

2 Small Integer Solutions

A. For any $u \in \mathbb{Z}_q$, denote the u-coset of $\mathcal{L}^\perp(\vec{g})$ by $\mathcal{L}_u^\perp(\vec{g}) \stackrel{\text{def}}{=} \{\vec{x} \in \mathbb{Z}^k : \langle \vec{g}, \vec{x} \rangle = u \pmod q\}$. Describe a poly$(n)$-time algorithm that given $u \in \mathbb{Z}_q$ outputs a vector $\vec{x} \in \mathcal{L}_u^\perp(\vec{g})$ of length at most $\sqrt{k}$.

Answer. Denote the binary representation of u by $u_{k-1} \ldots u_1 u_0$, then the vector $\vec{x} = \langle u_0, u_1, \ldots, u_{k-1} \rangle$ (that has Euclidean length $\leq \sqrt{k}$) satisfies $\langle \vec{g}, \vec{x} \rangle = \sum_i u_i 2^i = u$, hence $\vec{x} \in \mathcal{L}_u^\perp(\vec{g})$.

B. Recall that the discrete Gaussian distibution with parameter s over a lattice (or coset) $L \subset \mathbb{R}^d$, outputs each point $\vec{x} \in L$ with probability proportional to the Gaussian measure $\rho_s(\vec{x})$. Namely,

$$D_{L,s}(\vec{x}) \stackrel{\text{def}}{=} \frac{\rho_s(\vec{x})}{\rho_s(L)}, \quad \text{where } \rho_s(\vec{x}) \stackrel{\text{def}}{=} \exp\big(-\pi\|\vec{x}\|^2/s^2\big) \text{ and } \rho_s(L) = \sum_{\vec{u} \in L} \rho_s(\vec{u})$$

Describe a poly(n)-time algorithm that given $u \in \mathbb{Z}_q$ samples from the distribution $D_{\mathcal{L}_u^\perp(\vec{g}),s}$, for a small parameter s. How small can you make s while still keeping the algorithm poly(n)-time?

Answer. We will show a very simple algorithm that works for the parameter $s = 1$ (say). The algorithm just keeps choosing at random vectors from the Gaussian distribution over the integers $\vec{x} \leftarrow D_{\mathbb{Z}^k,s}$ until it gets a vector satisfying $\langle \vec{g}, \vec{x} \rangle = u \pmod q$. Clearly this yields the right distribution, it is only left to prove that it runs in expected poly(n) time.

For the proof, let us partition the integer lattice $\mathbb{Z}^k$ into little cubes of size 2^k, namely for every even vector $\vec{z} = \langle z_1, \ldots, z_k \rangle \in 2\mathbb{Z}^k$ we consider the cube $C_{\vec{z}} = \{\vec{x} : x_i \in \{z_i, z_i + 1\} \forall i\}$. We view the process of choosing $\vec{x} \leftarrow D_{\mathbb{Z}^k,s}$ as first choosing a cube (by the induced distribution) and then choosing $\vec{x}$ from the conditioned distribution on this cube.

Note that if the cube is not too far from the origin, then the conditional distibution over the cube is not too far from uniform. Assume (for simplicity of notations) that all the entries in $\vec{z}$ are non-negative, then the ratio between probability mass of the most likely point in $C_{\vec{z}}$ (which is the point $\vec{z}$ itself) and that of the least likely point (namely $\vec{z} + \vec{1}$) is

$$\frac{\exp(-\pi\|\vec{z}/s\|^2)}{\exp(-\pi\|(\vec{z}+\vec{1})/s\|^2)} = \exp\left(\frac{\pi}{s^2} \cdot \sum_i ((z_i+1)^2 - z_i^2)\right) = \exp\left(\frac{\pi}{s^2} \cdot \sum_i (2z_i + 1)\right)$$

Since the z_i's are integers then $\sum z_i \leq \sum z_i^2$, and therefore this ratio is bounded by

$$\exp\left(\frac{\pi}{s^2} \cdot \sum_i (2z_i + 1)\right) \leq \exp\left(\frac{\pi}{s^2} \cdot (2\|\vec{z}\|^2 + k)\right)$$

(By symmetry, the same bound holds also when $\vec{z}$ has negative entries, in which case $\vec{z}$ in the above expression is replaced by the point in $C_{\vec{z}}$ closest to the origin.)

Next, observe that when choosing a random $\vec{x} \leftarrow D_{\mathbb{Z}^k,s}$, the expected suqared length of $\|x\|$ is at most $s^2k/2\pi$. Hence there is a constant probability to get (say) $\|\vec{x}\|^2 \leq s^2k/\pi$. Thus, when choosing a cube according to the induced distribution there is a constant probability to choose one where the ratio between the probability mass of the most likely and least likely points is bounded by

$$\exp\left(\frac{\pi}{s^2} \cdot (2\frac{s^2k}{\pi} + k)\right) \quad = \quad \exp\left(k(2 + \frac{\pi}{s^2})\right) \quad \stackrel{(a)}{<} \quad \exp(6k) \quad \stackrel{(b)}{<} \quad (2q)^9,$$

where Inequality (a) follows from $s = 1$ and Inequality (b) follows from $\exp(6k) = (2^k)^{6/\ln 2} < (2q)^9$. Below we call cubes that satisfy this bound "*good cubes.*"

Last, observe that as $\vec{x}$ ranges over all the 2^k points inside one small cube, the value of the inner product $\langle \vec{g}, \vec{x} \rangle \bmod q$ ranges over all of $\mathbb{Z}_q$, with each value in $\mathbb{Z}_q$ obtained either once or twice. (This follows from the structure of $\vec{g}$, from the fact that we only vary the LSB's of the entries in $\vec{x}$, and from the fact that $2^k < 2q$.) Hence when we choose $\vec{x}$ from the conditional distribution over a good cube, the ratio between the probability masses of the most likely and the least likely values of $\langle \vec{g}, \vec{x} \rangle \bmod q$ is bounded by $2 \cdot (2q)^9$. Since the most likely value of $\langle \vec{g}, \vec{x} \rangle \bmod q \in \mathbb{Z}_q$ has probability at least $1/q$, then the least-likely value is obtained with probability at least $(2q)^{-10}$ in such good cubes.

We are now ready to complete the proof. We have shown that there is an event with constant probability (i.e., chosing a good cube), such that conditioned on this event every value of $\langle \vec{g}, \vec{x} \rangle$ mod q is obtained with probability at least $(2q)^{-10}$. Hence for every input value $u \in \mathbb{Z}_q$, the overall probability of getting $\langle \vec{g}, \vec{x} \rangle = u \pmod q$ is at least $\Omega(q^{-10})$, so the expected number of samples is bounded by $O(q^{10})$, which is polynomial in n. $\square$

C. Describe a poly(n)-time algorithm that given $\vec{u} \in \mathbb{Z}_q^n$ outputs a vector in $\mathcal{L}_{\vec{u}}^{\perp}(G) \overset{\text{def}}{=} \{\vec{x} \in \mathbb{Z}^{nk} : G\vec{x} = \vec{u} \pmod q\}$ of size at most $\sqrt{nk}$ (for the matrix G from Equation 2). Also describe a poly(n)-time algorithm that given $\vec{u} \in \mathbb{Z}_q^n$ samples from $D_{\mathcal{L}_{\vec{u}}^{\perp}(G),s}$, for a small parameter s.

Answer. For the first algorithm, given a vector $\vec{u}$ we go over all the entries in $\vec{u}$, and for each entry u_i use the algorithm from Part A to find a vector $\vec{x_i}$ such that $\langle \vec{g}, x \vec{+} i \rangle = u_i \pmod q$. Then the final vector is just the concatenation of all the $\vec{x_i}$'s.

Similarly for the second algorithm, for each entry u_i we choose at random $\vec{x_1} \leftarrow D_{\mathcal{L}_{u}^{\perp}(\vec{g}),s}$ using the algorithm from Part B, and then concatenate all the $\vec{x_i}$'s.

3 Learning with Errors

A. Describe a poly(n)-time algorithm that solves the *learning with errors* problem with respect to $\vec{g}$. Namely, for a secret scalar s, the algorithm is given as input a vector $\vec{u} = s\vec{g} + \vec{e} \bmod q$ where $\vec{e}$ is a "small error vector" with entries smaller than $q/8$ in absolute value. Your algorithm needs to recover the secret s.

Answer. Since q is polynomial in n, we can do exhaustive search,[1] going over all possible values of $s \in \mathbb{Z}_q$ and outputting the first one which is consistent with the received vector $\vec{u}$. It is only left to show that this value matches the secret s that was used to generate $\vec{u}$.

To see this, let $s' \in \mathbb{Z}_q$ be some different value, $s \neq s'$, and we show that s' cannot be consistent with $\vec{u}$. Denote $\delta = s - s' \bmod q$, where we think of the mod opeation as mapping integers into the interval $[-q/2, q/2)$. Also for $i = 0, 1, \ldots, k-1$ denote $\delta_i = 2^i \cdot \delta \bmod q$. Then since $k \geq \log_2 q$ there must be some index i such that $|\delta_i| \geq q/4$. For this index i, we know that $u_i = 2^i s + e_i$ where $|e_i| < q/8$. Hence $u_i - 2^i s' = 2^i(s - s') + e_i = \delta_i + e_i$. But $|\delta_i| \geq q/4$ and $|e_i| < q/8$, so the distance between u_i and $2^i \cdot s'$ is larger than $q/8$, hence s' cannot be consistent with $\vec{u}$.

B. Describe a poly(n)-time algorithm that inverts the function $\text{LWE}_G(\vec{s}, \vec{e}) = \vec{s}G + \vec{e} \bmod q$, where $\vec{s} \in \mathbb{Z}_q^n$ and $\vec{e} \in \left[-\left\lfloor \frac{q-1}{8} \right\rfloor, \left\lfloor \frac{q-1}{8} \right\rfloor \right]^{nk}$.

Answer. We break $\vec{u}$ into n subvectors of size k each $\vec{u_1} \ldots \vec{u_n}$. For each subvector $\vec{u_i}$ we use the algorithm from Part A to find a scalar s such that $\vec{u_i} = s\vec{g} + \vec{e_i}$ for a small e_i, then we have the vector $\vec{s} = \langle s_1, s_2, \ldots, s_n \rangle$, and we recover $\vec{e}$ as $\vec{e} = \vec{u} - \vec{s}G \bmod q$.

[1] Alternatively, there is a simple binary-search strategy that would work also for super-polynomial values of q.

References

[1] Daniele Micciancio and Chris Peikert. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller, In *EUROCRYPT 2012*, LNCS vol. 7237, pages 700-718, Springer, 2012.