CHAPTER 5

Lattices

Chapter Goals

- To describe lattice basis reduction algorithms and give some examples of how they are used to break cryptographic systems.
- To introduce the hard problems of SVP, CVP, BDD and their various variations.
- To introduce q-ary lattices.
- To explain the technique of Coppersmith for finding small roots of modular polynomial equations.

5.1. Lattices and Lattice Reduction

In this chapter we present the concept of a lattice. Traditionally in cryptography lattices have been used in cryptanalysis to break systems, and we shall see applications of this in Chapter 15. However, recently they have also been used to construct cryptographic systems with special properties, as we shall see in Chapter 17.

5.1.1. Vector Spaces: Before presenting lattices, and the technique of lattice basis reduction, we first need to recap some basic linear algebra. Suppose $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is an n-dimensional real vector, i.e. for all i we have $x_i \in \mathbb{R}$. The set of all such vectors is denoted $\mathbb{R}^n$. On two such vectors we can define an *inner product*

$$\langle \mathbf{x}, \mathbf{y} \rangle = x_1 \cdot y_1 + x_2 \cdot y_2 + \cdots + x_n \cdot y_n,$$

which is a function from pairs of n-dimensional vectors to the real numbers. You probably learnt at school that two vectors $\mathbf{x}$ and $\mathbf{y}$ are orthogonal, or meet at right angles, if and only if we have

$$\langle \mathbf{x}, \mathbf{y} \rangle = 0.$$

Given the inner product we can then define the size, or length, of a vector by

$$\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle} = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}.$$

This length corresponds to the intuitive notion of length of vectors; in particular the length satisfies a number of properties.

- $\|\mathbf{x}\| \geq 0$, with equality if and only if $\mathbf{x}$ is the zero vector.
- Triangle inequality: For two n-dimensional vectors $\mathbf{x}$ and $\mathbf{y}$

$$\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|.$$

- Scaling: For a vector x and a real number a

$$\|a \cdot \mathbf{x}\| = |a| \cdot \|\mathbf{x}\|.$$

N.P. Smart, *Cryptography Made Simple*, Information Security and Cryptography, DOI 10.1007/978-3-319-21936-3_5

A set of vectors $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$ in $\mathbb{R}^n$ is called linearly independent if the equation

$$a_1 \cdot \mathbf{b}_1 + \cdots + a_m \cdot \mathbf{b}_m = 0,$$

for real numbers a_i, implies that all a_i are equal to zero. If the set is linearly independent then we must have $m \leq n$. Suppose we have a set of m linearly independent vectors, $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$. We can look at the set of all real linear combinations of these vectors,

$$V = \left\{ \sum_{i=1}^{m} a_i \cdot \mathbf{b}_i : a_i \in \mathbb{R} \right\}.$$

This is a vector subspace of $\mathbb{R}^n$ of dimension m and the set $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$ is called a basis of this subspace. If we form the matrix B with ith column of B being equal to $\mathbf{b}_i$ for all i then we have

$$V = \{B \cdot \mathbf{a} : \mathbf{a} \in \mathbb{R}^m\}.$$

The matrix B is called the basis matrix.

5.1.2. The Gram–Schmidt Process: Every subspace V has a large number of possible basis matrices. Given one such basis it is often required to produce a basis with certain prescribed nice properties. Often in applications throughout science and engineering one requires a basis which is pairwise orthogonal, i.e.

$$\langle \mathbf{b}_i, \mathbf{b}_j \rangle = 0$$

for all $i \neq j$. Luckily there is a well-known method which takes one basis, $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$ and produces a basis $\{\mathbf{b}_1^*, \ldots, \mathbf{b}_m^*\}$ which is pairwise orthogonal. This method is called the Gram–Schmidt process and the basis $\{\mathbf{b}_1^*, \ldots, \mathbf{b}_m^*\}$ produced from $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$ via this process is called the Gram–Schmidt basis corresponding to $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$. One computes the $\mathbf{b}_i^*$ from the $\mathbf{b}_i$ via the recursive equations

$$\mu_{i,j} \leftarrow \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle}, \text{ for } 1 \leq j < i \leq n,$$
$$\mathbf{b}_i^* \leftarrow \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \cdot \mathbf{b}_j^*.$$

For example if we have

$$\mathbf{b}_1 = \begin{pmatrix} 2 \\ 0 \end{pmatrix} \text{ and } \mathbf{b}_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

then we compute

$$\mathbf{b}_1^* \leftarrow \mathbf{b}_1 = \begin{pmatrix} 2 \\ 0 \end{pmatrix},$$
$$\mathbf{b}_2^* \leftarrow \mathbf{b}_2 - \mu_{2,1} \cdot \mathbf{b}_1^* = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \frac{1}{2} \cdot \begin{pmatrix} 2 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix},$$

since

$$\mu_{2,1} = \frac{\langle \mathbf{b}_2, \mathbf{b}_1^* \rangle}{\langle \mathbf{b}_1^*, \mathbf{b}_1^* \rangle} = \frac{2}{4} = \frac{1}{2}.$$

Notice how we have $\langle b_1^*, b_2^* \rangle = 0$, so the new Gram–Schmidt basis is orthogonal.

5.1.3. Lattices: A *lattice* L is like the vector subspace V above, but given a set of basis vectors $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$ instead of taking the *real* linear combinations of the $\mathbf{b}_i$ we are only allowed to take the *integer* linear combinations of the $\mathbf{b}_i$,

$$L = \left\{ \sum_{i=1}^{m} a_i \cdot \mathbf{b}_i : a_i \in \mathbb{Z} \right\} = \{B \cdot \mathbf{a} : \mathbf{a} \in \mathbb{Z}^m\}.$$

The set $\{\mathbf{b}_1, \ldots, \mathbf{b}_m\}$ is still called the set of basis vectors and the matrix B is still called a basis matrix. To see why lattices are called lattices, consider the lattice L generated by the two vectors

$$\mathbf{b}_1 = \begin{pmatrix} 2 \\ 0 \end{pmatrix} \text{ and } \mathbf{b}_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

This is the set of all vectors of the form

$$\begin{pmatrix} 2 \cdot x + y \\ y \end{pmatrix},$$

where $x, y \in \mathbb{Z}$. If one plots these points in the plane then one sees that these points form a two-dimensional lattice. See for example Figure 5.1.

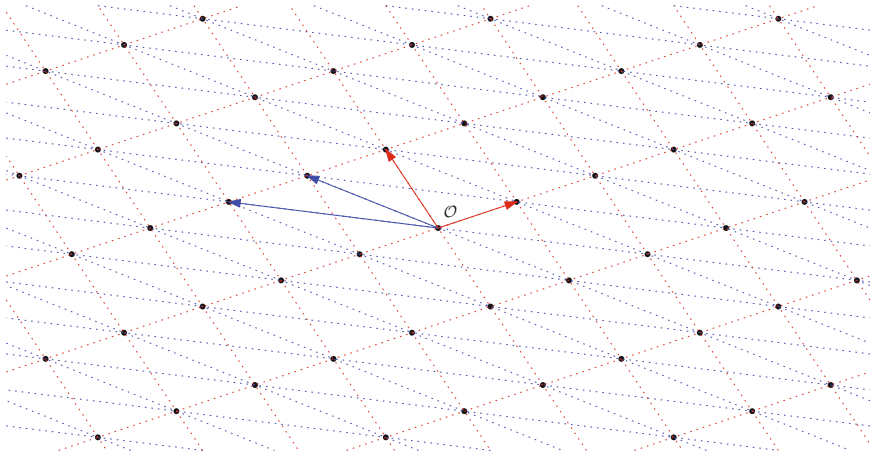

FIGURE 5.1. A lattice with two bases marked. A "nice" one in red, and a "bad" one in blue

A lattice is a discrete version of a vector subspace. Since it is discrete there is a well-defined smallest element, bar the trivially small element of the zero vector of course. This allows us to define the non-zero minimum of any lattice L, which we denote by

$$\lambda_1(L) := \min\{\|\mathbf{x}\| : \mathbf{x} \in L, \mathbf{x} \neq \mathbf{0}\}.$$

We can also define the successive minima $\lambda_i(L)$, which are defined as the smallest radius r such that the n-dimensional ball of radius r centred on the origin contains i linearly independent lattice points. Many tasks in computing, and especially cryptography, can be reduced to trying to determine the smallest non-zero vector in a lattice. We shall see some of these applications later, but before continuing with our discussion of lattices in general we pause to note that it is generally considered to be a hard problem to determine the smallest non-zero vector in an arbitrary lattice. Later we shall see that, whilst this problem is hard in general, it is in fact easy in low dimension, a situation which we shall use to our advantage later on.

Just as with vector subspaces, given a lattice basis one could ask, is there a nicer basis? For example in Figure 5.1 the red basis is certainly "more orthogonal" than the blue basis. Suppose B is a basis matrix for a lattice L. To obtain another basis matrix B' the only operation we are allowed to use is post-multiplication of B by a uni-modular integer matrix. This means we must have

$$B' = B \cdot U$$

for some integer matrix U with $\det(U) = \pm 1$. This means that the absolute value of the determinant of a basis matrix of a lattice is an invariant of the lattice, i.e. it does not depend on the choice of basis. Given a basis matrix B for a lattice L, we call

$$\Delta(L) = |\det(B^t \cdot B)|^{1/2}$$

the *discriminant* of the lattice. If L is a lattice of full rank, i.e. B is a square matrix, then we have

$$\Delta(L) = |\det(B)|.$$

The value $\Delta(L)$ represents the volume of the fundamental parallelepiped of the lattice bases; see for example Figure 5.2, which presents a lattice with two parallelepipeds marked. Each parallelepiped has the same volume, as both correspond to a basis of the lattice. If the lattice is obvious from the context we just write Δ. From now on we shall only consider full-rank lattices, and hence $n = m$, and our basis matrices will be square.

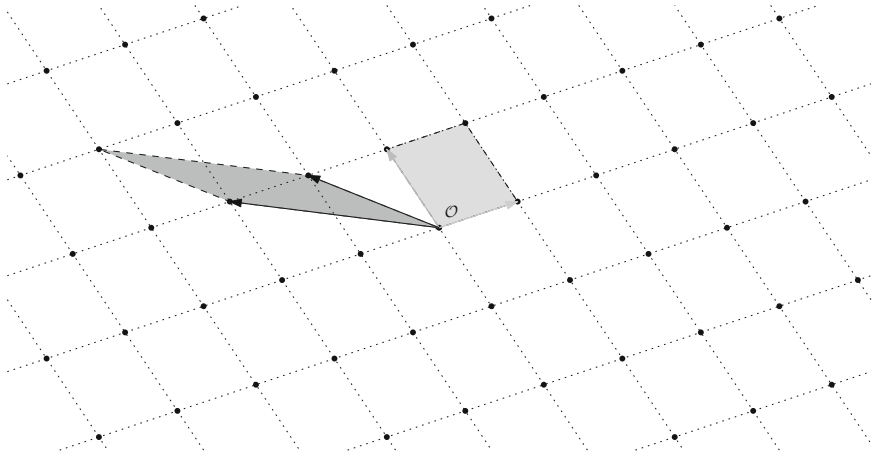

FIGURE 5.2. A lattice with two fundamental parallelepipeds marked

Hermite showed that there is an absolute constant γ_n, depending only on n, such that

$$\lambda_1(L) \leq \sqrt{\gamma_n} \cdot \Delta(L)^{1/n}.$$

Although the value of γ_n is only known for $1 \leq n \leq 8$, for "random lattices" the first minimum, and hence Hemite's constant γ_n, can approximated by appealing to the *Gaussian Heuristic*, which states that for a "random lattice" we have

$$\lambda_1(L) \approx \sqrt{\frac{n}{2 \cdot \pi \cdot e}} \cdot \Delta(L)^{1/n}.$$

The classic result in lattice theory (a.k.a. geometry of numbers), is that of Minkowski, which relates the minimal distance to the volume of the fundamental parallelepiped.

Theorem 5.1 (Minkowski)**.** *Let L be a rank-n lattice and $\mathbb{C} \subset L$ be a convex symmetric body about the origin with volume $\mathsf{Vol}(\mathcal{C}) > 2^n \cdot \Delta(L)$. Then $\mathcal{C}$ contains a non-zero vector $\mathbf{x} \in L$.*

The following immediate corollary is also often referred to as "Minkowski's Theorem".

Corollary 5.2 (Minkowski's Theorem)**.** *For any n-dimensional lattice L we have*

$$\lambda_1(L) \leq \sqrt{n} \cdot \Delta(L)^{1/n}.$$

The *dual* L^* of a lattice L is the set of all vectors $\mathbf{y} \in \mathbb{R}^n$ such that $\mathbf{y} \cdot \mathbf{x}^\mathsf{T} \in \mathbb{Z}$ for all $\mathbf{x} \in L$. Given a basis matrix B of L we can compute the basis matrix B^* of L^* via $B^* = (B^{-1})^\mathsf{T}$. Hence we have $\Delta(L^*) = 1/\Delta(L)$. The first minimum of L and the nth minimum of L^* are linked by the *transference theorem* of Banaszczyk which states that for all n-dimensional lattices we have

$$1 \leq \lambda_1(L) \cdot \lambda_n(L^*) \leq n.$$

Thus a lower bound on $\lambda_1(L)$ can be translated into an upper bound on $\lambda_n(L^*)$ and vice versa, a fact which is used often in the analysis of lattice algorithms.

5.1.4. LLL Algorithm: One could ask, given a lattice L does there exist an orthogonal basis? In general the answer to this last question is no. If one looks at the Gram–Schmidt process in more detail one sees that, even if one starts out with integer vectors, the coefficients $\mu_{i,j}$ almost always end up not being integers. Hence, whilst the Gram–Schmidt basis vectors span the same *vector subspace* as the original basis they do not span the same *lattice* as the original basis. This is because we are not allowed to make a change of basis which consists of non-integer coefficients. However, we could try to make a change of basis so that the new basis is "close" to being orthogonal in that

$$|\mu_{i,j}| \leq \frac{1}{2} \text{ for } 1 \leq j < i \leq n.$$

These considerations led Lenstra, Lenstra and Lovász to define the following notion of reduced basis, called an LLL reduced basis after its inventors.

Definition 5.3. *A basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is called LLL reduced if the associated Gram–Schmidt basis $\{\mathbf{b}_1^*, \ldots, \mathbf{b}_n^*\}$ satisfies*

$$|\mu_{i,j}| \leq \frac{1}{2} \text{ for } 1 \leq j < i \leq n, \tag{7}$$

$$\|\mathbf{b}_i^*\|^2 \geq \left(\frac{3}{4} - \mu_{i,i-1}^2\right) \cdot \|\mathbf{b}_{i-1}^*\|^2 \text{ for } 1 < i \leq n. \tag{8}$$

What is truly amazing about an LLL reduced basis is

- An LLL reduced basis can be computed in polynomial time; see below for the method.
- The first vector in the reduced basis is very short, in fact it is close to the shortest non-zero vector in that for all non-zero $\mathbf{x} \in L$ we have
$$\|\mathbf{b}_1\| \leq 2^{(n-1)/2} \cdot \|\mathbf{x}\|.$$
- $\|\mathbf{b}_1\| \leq 2^{n/4} \cdot \Delta^{1/n}$.

The constant $2^{(n-1)/2}$ in the second bullet point above is a worst-case constant, and is called the *approximation factor*. In practice for many lattices of small dimension after one applies the LLL algorithm to obtain an LLL reduced basis, the first vector in the LLL reduced basis is in fact equal to the smallest vector in the lattice. Hence, in such cases the approximation factor is one.

The LLL algorithm works as follows: We keep track of a copy of the current lattice basis B and the associated Gram–Schmidt basis B^*. At any point in time we are examining a fixed column k, where we start with $k = 2$.

- If condition (7) does not hold for $\mu_{k,j}$ with $1 \leq j < k$ then we alter the basis B so that it does.
- If condition (8) does not hold for column k and column $k-1$ we swap columns k and $k-1$ around and decrease the value of k by one (unless k is already equal to two). If condition (8) holds then we increase k by one.

At some point (in fact in polynomial time) we will obtain $k = n$ and the algorithm will terminate. To prove this, one shows that the number of iterations where one decreases k is bounded, and so it is guaranteed that the algorithm will terminate. We will not prove this, but note that clearly if the algorithm terminates it will produce an LLL reduced basis.

For a Gram–Schmidt basis B^* of a basis B we define the Gram–Schmidt Log, $\mathsf{GSL}(B)$, as the vector

$$\mathsf{GSL}(B) = \left(\log\left(\|\mathbf{b}_i^*\|/\Delta(L)^{1/n}\right)\right)_{i=1,\ldots,n}.$$

It is "folklore" that the output of the LLL algorithm produces a basis B whose $\mathsf{GSL}(B)$ when plotted looks like a straight line. The (average) slope η_B of this line can be then computed from $\mathsf{GSL}(B)$ via

$$\eta_B := \frac{12}{(n+1)\cdot n\cdot(n-1)} \cdot \left(\sum_{i=1}^{n} i \cdot \mathsf{GSL}(B)_i\right).$$

The Geometric Series Assumption (GSA) is that the output of the LLL algorithm does indeed behave in this way for a given input basis.

Example: As an example we take the above basis of a two-dimensional lattice in $\mathbb{R}^2$

$$\mathbf{b}_1 = \begin{pmatrix} 2 \\ 0 \end{pmatrix},\ \mathbf{b}_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

The associated Gram–Schmidt basis is given by

$$\mathbf{b}_1^* = \begin{pmatrix} 2 \\ 0 \end{pmatrix},\ \mathbf{b}_2^* = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

But this is not a basis of the associated lattice since one cannot pass from $\{\mathbf{b}_1, \mathbf{b}_2\}$ to $\{\mathbf{b}_1^*, \mathbf{b}_2^*\}$ via a unimodular integer transformation.

We now apply the LLL algorithm with $k = 2$ and find that the first condition (7) is satisfied since $\mu_{2,1} = \frac{1}{2}$. However, the second condition (8) is not satisfied because

$$1 = \|\mathbf{b}_2^*\|^2 < \left(\frac{3}{4} - \mu_{2,1}^2\right) \cdot \|\mathbf{b}_1^*\|^2 = \frac{1}{2} \cdot 4 = 2.$$

Hence, we need to swap the two basis vectors around, to obtain the new lattice basis vectors

$$\mathbf{b}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix},\ \mathbf{b}_2 = \begin{pmatrix} 2 \\ 0 \end{pmatrix},$$

with the associated Gram–Schmidt basis vectors

$$\mathbf{b}_1^* = \begin{pmatrix} 1 \\ 1 \end{pmatrix},\ \mathbf{b}_2^* = \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

We now go back to the first condition again. This time we see that we have $\mu_{2,1} = 1$ which violates the first condition. To correct this we subtract $\mathbf{b}_1$ from the vector $\mathbf{b}_2$ so as to obtain $\mu_{2,1} = 0$. We now find the lattice basis is given by

$$\mathbf{b}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix},\ \mathbf{b}_2 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

and the Gram–Schmidt basis is then identical to this lattice basis, in that

$$\mathbf{b}_1^* = \begin{pmatrix} 1 \\ 1 \end{pmatrix},\ \mathbf{b}_2^* = \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

Now we are left to check the second condition again, we find

$$2 = \|\mathbf{b}_2^*\|^2 \geq \left(\frac{3}{4} - \mu_{2,1}^2\right) \cdot \|\mathbf{b}_1^*\|^2 = \frac{3}{4} \cdot 2 = \frac{3}{2}.$$

Hence, both conditions are satisfied and we conclude that

$$\mathbf{b}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \ \mathbf{b}_2 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

is an LLL reduced basis for the lattice L.

5.1.5. Continued Fractions: We end this section by noticing that there is a link between continued fractions and short vectors in lattices. Let $\alpha \in \mathbb{R}$, and define the following sequences, starting with $\alpha_0 = \alpha$, $p_0 = a_0$ and $q_0 = 1$, $p_1 = a_0 \cdot a_1 + 1$ and $q_1 = a_1$,

$$\begin{aligned} a_i &= \lfloor \alpha_i \rfloor, \\ \alpha_{i+1} &= \frac{1}{\alpha_i - a_i}, \\ p_i &= a_i \cdot p_{i-1} + p_{i-2} \text{ for } i \geq 2, \\ q_i &= a_i \cdot q_{i-1} + q_{i-2} \text{ for } i \geq 2. \end{aligned}$$

The integers $a_0, a_1, a_2, \ldots$ are called the continued fraction expansion of α and the fractions

$$\frac{p_i}{q_i}$$

are called the convergents. The denominators of these convergents grow at an exponential rate and the convergent above is a fraction in its lowest terms since one can show $\gcd(p_i, q_i) = 1$ for all values of i. The important result is that if p and q are two integers with

$$\left|\alpha - \frac{p}{q}\right| \leq \frac{1}{2 \cdot q^2}$$

then $\frac{p}{q}$ is a convergent in the continued fraction expansion of α.

A similar effect can be achieved using lattices by applying the LLL algorithm to the lattice generated by the columns of the matrix

$$\begin{pmatrix} 1 & 0 \\ C \cdot \alpha & -C \end{pmatrix},$$

for some constant C. This is because the lattice L contains the "short" vector

$$\begin{pmatrix} q \\ C \cdot (q \cdot \alpha - p) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ C \cdot \alpha & -C \end{pmatrix} \cdot \begin{pmatrix} q \\ p \end{pmatrix}.$$

Therefore, in some sense we can consider the LLL algorithm to be a multi-dimensional generalization of the continued fraction algorithm.

5.2. "Hard" Lattice Problems

There are two basic hard problems associated with a lattice. The first is the *shortest vector problem* and the second is the *closest vector problem*. However, these problems are only hard for large dimensions, since for large dimension the approximation factor of the LLL algorithm, $2^{(n-1)/2}$, becomes too large and the LLL algorithm is no longer able to find a good basis.

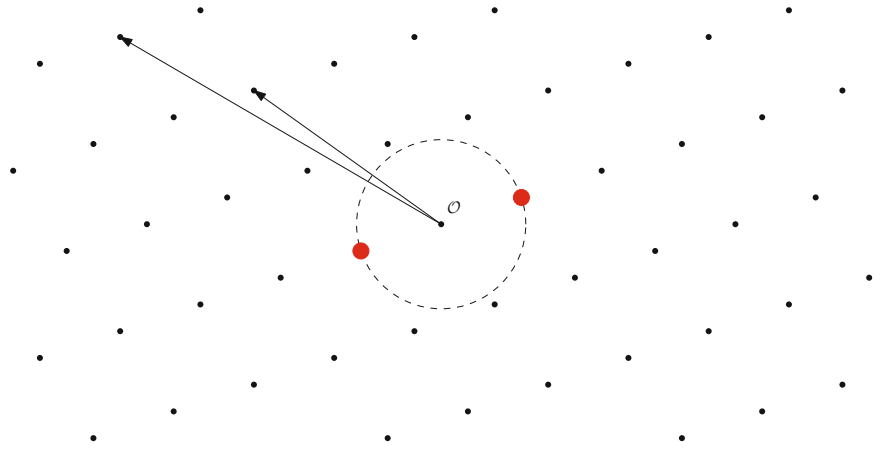

FIGURE 5.3. An SVP solution

5.2.1. Shortest Vector Problem: The simplest, and most famous, hard problem in a lattice is to determine the shortest vector within the lattice. This problem comes in a number of variants:

Definition 5.4 (Shortest Vector Problem)**.** *Given a lattice basis B there are three variants of this problem:*

- *The shortest vector problem* SVP *is to find a non-zero vector* $\mathbf{x}$ *in the lattice L generated by B for which*
$$\|\mathbf{x}\| \leq \|\mathbf{y}\|$$
for all non-zero $\mathbf{y} \in L$, i.e. $\|\mathbf{x}\| = \lambda_1(L)$.
- *The approximate-SVP problem* SVP_γ *is to find a* $\mathbf{x}$ *such that*
$$\|\mathbf{x}\| \leq \gamma \cdot \lambda_1(L),$$
for some "small" constant γ.
- *The γ-unique SVP problem* uSVP_γ *is given a lattice and a constant $\gamma > 1$ such that $\lambda_2(L) > \gamma \cdot \lambda_1(L)$, find a non-zero $\mathbf{x} \in L$ of length $\lambda_1(L)$.*

See Figure 5.3 for an example two-dimensional lattice, the input basis, and the two shortest lattice vectors which an SVP solver should find. Note that a short lattice vector is not unique, since if $\mathbf{x} \in L$ then we also have $-\mathbf{x} \in L$. The LLL algorithm will heuristically solve the SVP, and for large dimension will solve the approximate-SVP problem with a value of γ of $2^{(n-1)/2}$ in the worst case. The γ-unique SVP problem is potentially easier than the others, since we are given more information about the underlying lattice.

5.2.2. Closest Vector Problem: We now present the second most important problem in lattices, namely the closest vector problem. See Figure 5.4 for an example two-dimensional lattice, the input basis, the target vector $\mathbf{x}$ in blue, and the closest lattice vector $\mathbf{y}$ in red.

Definition 5.5 (Closest Vector Problem)**.** *Given a lattice basis B generating a lattice L in n-dimensional real space, and a vector $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{x} \notin L$, there are two variants of this problem:*

- *The closest vector problem* CVP *is to find a lattice vector* $\mathbf{y} \in L$ *such that*
$$\|\mathbf{x} - \mathbf{y}\| \leq \|\mathbf{x} - \mathbf{z}\|$$
for all $\mathbf{z} \in L$.

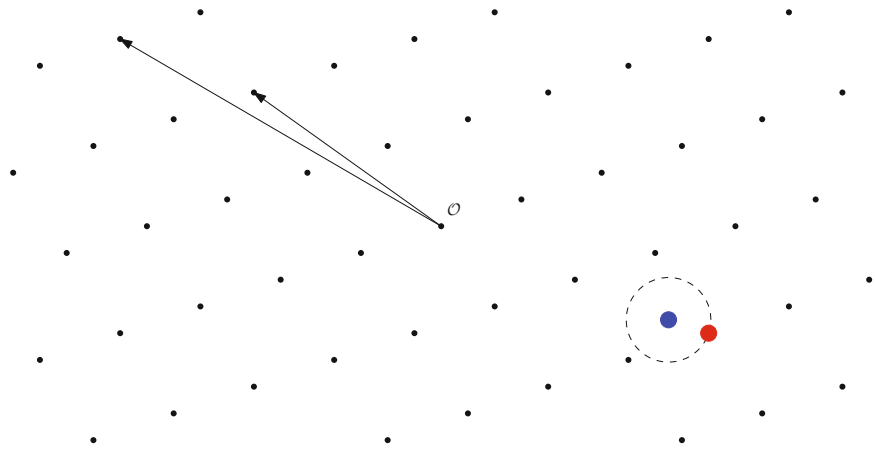

FIGURE 5.4. A CVP solution

- *The approximate-CVP problem* CVP_γ *is to find a* $\mathbf{y}$ *such that*

$$\|\mathbf{x}-\mathbf{y}\| \leq \gamma \cdot \|\mathbf{x}-\mathbf{z}\|$$

for all $\mathbf{z} \in L$*, for some "small" constant* γ*.*

The SVP and CVP problems are related in that, in practice, we can turn an algorithm which finds approximate-SVP solutions into one which finds approximate-CVP solutions as follows. Let $(B, \mathbf{x})$ denote the input to the CVP problem, where without loss of generality we assume that B is an $n \times n$ matrix, i.e. the lattice is full rank and in $\mathbb{R}^n$. We now form the lattice in $\mathbb{R}^{n+1}$ generated by the columns of the matrix

$$B' = \left(\begin{array}{c|c} B & \mathbf{x} \\ \hline \mathbf{0} & C \end{array}\right),$$

where C is some "large" constant. We let $\mathbf{b}'$ denote a short vector in the lattice generated by B', so we have that

$$\mathbf{b}' = \begin{pmatrix} B & \mathbf{x} \\ \mathbf{0} & C \end{pmatrix} \cdot \begin{pmatrix} \mathbf{a} \\ t \end{pmatrix} = \begin{pmatrix} \mathbf{y} + t \cdot \mathbf{x} \\ t \cdot C \end{pmatrix},$$

where $\mathbf{y}$ is in the lattice generated by L, i.e. $\mathbf{y} = B \cdot \mathbf{a}$ for $\mathbf{a} \in \mathbb{Z}^n$. We then have that

$$\|\mathbf{b}'\|^2 = \|\mathbf{y} + t \cdot \mathbf{x}\|^2 + t^2 \cdot C^2$$

Now since $\|\mathbf{b}'\|$ is "small" and C is "large" we expect that $t = \pm 1$. Without loss of generality we can assume that $t = -1$. But this then implies that $\|\mathbf{y} - \mathbf{x}\|^2$ is very small, and so $\mathbf{y}$ is highly likely to be an approximate solution to the original closest vector problem.

5.2.3. Bounded-Distance Decoding Problem: There is a "simpler" problem (meaning we give the solver of the problem more information) associated with the closest vector problem called the Bounded-Distance Decoding problem (or BDD problem). Here we not only give the adversary the lattice and a non-lattice vector, but we also give the adversary a bound on the distance between the lattice and the non-lattice vector. Thus we are giving the adversary more information than in the above two CVP problems; thus the BDD problem is akin to the γ-unique SVP problem.

Definition 5.6 (Bounded-Distance Decoding)**.** *Given a lattice basis* B *generating a lattice* L *in* n*-dimensional real space, a vector* $\mathbf{x} \in \mathbb{R}^n$ *with* $\mathbf{x} \notin L$*, and a real number* λ*. The Bounded-Distance Decoding problem* BDD_λ *is to find a lattice vector* $\mathbf{y} \in L$ *such that*

$$\|\mathbf{x}-\mathbf{y}\| \leq \lambda \cdot \lambda_1(L).$$

If the shortest non-zero vector in the lattice has size $\lambda_1(L)$, then if we have $\lambda < 1/2$ the solution to the BDD problem is guaranteed to be unique. An interesting aspect of the BDD problem is that if the vector $\mathbf{x}$ is known to be really close to the lattice, i.e. λ is much smaller than one would expect for a random value of $\mathbf{x}$, then solving the BDD problem becomes easier.

The BDD problem will be very important later in Chapter 17 so we now discuss how hard BDD is in relation to other more standard lattice problems. The traditional "practical" method of solving BDD_α is to embed the problem into a uSVP_γ problem of a lattice of dimension one larger than the original problem. This is exactly our strategy above for solving CVP. For BDD we can formalize this and show that the two problems are (essentially) equivalent. We present the reduction from BDD_α to uSVP_γ, since we can use lattice reduction algorithms as a uSVP_γ oracle, and hence the reduction in this direction is more practically relevant. We simplify things by assuming that the distance in the BDD_α problem is *exactly* known. Note that a more complex algorithm to that given in the proof can deal with the case where this distance is not known, but is just known to be less than $\alpha \cdot \lambda_1(L)$. In any case, in many examples the distance will be known with a high degree of certainty.

Theorem 5.7. *Given a lattice L via a basis B and a vector $\mathbf{x} \in \mathbb{R}^n$, with $\mathbf{x} \notin L$, such that the minimum distance from $\mathbf{x}$ to a lattice point $\mathsf{dist}(\mathbf{x}, L) = \mu \leq \alpha \cdot \lambda_1(L)$, and an oracle for uSVP_γ for $\gamma = 1/(2 \cdot \alpha)$ then, assuming μ is known, there is an algorithm which will output $\mathbf{y}$ such that $\mathbf{y} \in L$ and $\|\mathbf{y} - \mathbf{x}\| = \mathsf{dist}(\mathbf{x}, L)$.*

Proof. First define the matrix

$$B' = \begin{pmatrix} B & \mathbf{x} \\ \mathbf{0} & \mu \end{pmatrix},$$

and consider the lattice L' generated by the matrix B'. For the target vector $\mathbf{y}$ define $\mathbf{z}$ by $\mathbf{y} = B \cdot \mathbf{z}$, where $\mathbf{z} \in \mathbb{Z}^n$. Consider the vector $\mathbf{y}' \in L'$ defined by

$$\mathbf{y}' = B' \cdot \begin{pmatrix} \mathbf{z} \\ -1 \end{pmatrix} = \begin{pmatrix} B \cdot \mathbf{z} - \mathbf{x} \\ -\mu \end{pmatrix} = \begin{pmatrix} \mathbf{y} - \mathbf{x} \\ -\mu \end{pmatrix}.$$

We have $\|\mathbf{y}'\| = \sqrt{\mu^2 + \mu^2} = \sqrt{2} \cdot \mu$ by assumption that $\mathsf{dist}(\mathbf{x}, L) = \mu$. If we pass the basis B' to our uSVP_γ oracle then this will output a vector $\mathbf{v}'$. We would like $\mathbf{v}' = \mathbf{y}'$ i.e. $\|\mathbf{y}'\| = \lambda_1(L')$ and all other vectors in L' are either a multiple of $\mathbf{y}'$ or have length greater than $\gamma \cdot \|\mathbf{y}'\| = \sqrt{2} \cdot \mu \cdot \gamma$. If this is true we can solve our BDD_α problem by taking the first n coordinates of $\mathbf{y}'$ and adding the vector $\mathbf{x}$ to the result so as to obtain the solution $\mathbf{y}$.

So assume, for sake of contradiction, that $\mathbf{w}'$ is a vector in L' of length less than $\sqrt{2} \cdot \mu \cdot \gamma$ and that $\mathbf{w}'$ is not a multiple of $\mathbf{y}'$. We can write, for $\mathbf{z}_1 \in \mathbb{Z}^n$ and $\beta \in \mathbb{Z}$,

$$\mathbf{w}' = B' \cdot (\mathbf{z}_1, -\beta)^\mathsf{T} = (B \cdot \mathbf{z}_1 - \beta \cdot \mathbf{x}, -\beta \cdot \mu)^\mathsf{T} = (\mathbf{w} - \beta \cdot \mathbf{x}, -\beta \cdot \mu)^\mathsf{T}$$

where $\mathbf{w} = B \cdot \mathbf{z}_1 \in L$. So we have $\sqrt{\|\mathbf{w} - \beta \cdot \mathbf{x}\|^2 + (\beta \cdot \mu)^2} = \|\mathbf{w}'\| < \sqrt{2} \cdot \mu \cdot \gamma$, which implies that $\|\mathbf{w} - \beta \cdot \mathbf{x}\| < \sqrt{2 \cdot \mu^2 \cdot \gamma^2 - \beta^2 \cdot \mu^2}$.

Now consider the vector $\mathbf{w} - \beta \cdot \mathbf{y} \in L$. Since $\mathbf{w}'$ is not a multiple of $\mathbf{y}'$, neither is $\mathbf{w}$ a multiple of $\mathbf{y}$, and so $\mathbf{w} - \beta \cdot \mathbf{y} \neq \mathbf{0}$. We wish to upper bound the length of $\mathbf{w} - \beta \cdot \mathbf{y}$:

$$\begin{aligned} \|\mathbf{w} - \beta \cdot \mathbf{y}\| &= \|(\mathbf{w} - \beta \cdot \mathbf{x}) - \beta \cdot (\mathbf{y} - \mathbf{x})\| \\ &\leq \|\mathbf{w} - \beta \cdot \mathbf{x}\| + \beta \cdot \|\mathbf{y} - \mathbf{x}\| \\ &< \sqrt{2 \cdot \mu^2 \cdot \gamma^2 - \beta^2 \cdot \mu^2} + \beta \cdot \mu. \end{aligned}$$

Now maximizing the right hand side by varying β, we find the maximum is $2 \cdot \gamma \cdot \mu$, which is achieved when $\beta = \gamma$ for real β, and hence we will have $\|\mathbf{w} - \beta \cdot \mathbf{y}\| < 2 \cdot \gamma \cdot \mu$ for integer values of β as well.

We now use the equality $\gamma = 1/(2 \cdot \alpha)$ and the inequality $\mu \leq \alpha \cdot \lambda_1(L)$ to obtain that

$$\|\mathbf{w} - \beta \cdot \mathbf{y}\| < 2 \cdot \left(\frac{1}{2 \cdot \alpha}\right) \cdot (\alpha \cdot \lambda_1(L)) = \lambda_1(L).$$

Thus for all integer values of β we conclude that $\mathbf{w} - \beta \cdot \mathbf{y} \in L$, that $\mathbf{w} - \beta \cdot \mathbf{y} \neq \mathbf{0}$ (since $\mathbf{w}$ is not a multiple of $\mathbf{y}$) and that $\|\mathbf{w} - \beta \cdot \mathbf{y}\| < \lambda_1(L)$ which is a contradiction, since $\lambda_1(L)$ is the length of the smallest non-zero vector in the lattice. □

So if we want to solve the BDD_α problem we convert it into a $\mathsf{uSVP}_{1/2\cdot\alpha}$ problem. We then apply lattice basis reduction to the resulting $\mathsf{uSVP}_{1/2\cdot\alpha}$ problem, and hope the resulting first basis element allows us to solve the original BDD_α problem. Since lattice reduction gets worse as the dimension increases this means as n increases we can only solve BDD problems in which the distance between the target vector and the lattice is very small. Alternatively, if we want BDD to be hard when the distance between the target and the lattice is very small then we need the dimension to be large.

The other approach in the literature for solving BDD_α in practice is to apply Babai's algorithm for solving closest vector problems. Babai's algorithm takes as input a lattice basis B and a non-lattice vector $\mathbf{x}$ and then outputs a lattice vector $\mathbf{w}$ such that the "error" $\mathbf{e} = \mathbf{w} - \mathbf{x}$ lies in the fundamental parallelepiped of the matrix B^*, where B^* is the Gram–Schmidt basis associated with B. In particular we have

$$\|\mathbf{e}\|^2 \leq \frac{1}{4} \sum_{i=1}^{n} \|\mathbf{b}_i^*\|^2.$$

If the input basis is reduced we expect this latter quantity to be small, and hence the error vector $\mathbf{e}$ is itself small and represents the distance to the closest lattice vector to $\mathbf{x}$.

5.3. q-ary Lattices

Of importance in one of the systems described later are so-called q-ary lattices. A q-ary lattice L is one such that $q\mathbb{Z}^n \subset L \subset \mathbb{Z}^n$ for some integer q. Note, that all integer lattices are q-ary lattices for a value of q which is an integer multiple of $\Delta(L)$. Our interest will be in special forms of q-ary lattice which are q-ary for a q-value much less than the determinant.

Suppose we are given a matrix $A \in \mathbb{Z}_q^{n \times m}$, with $m \geq n$; we then define the following two m-dimensional q-ary lattices.

$$\Lambda_q(A) = \{\mathbf{y} \in \mathbb{Z}^m : \mathbf{y} = A^\mathsf{T} \cdot \mathbf{z} \pmod q \text{ for some } \mathbf{z} \in \mathbb{Z}^n\},$$
$$\Lambda_q^\perp(A) = \{\mathbf{y} \in \mathbb{Z}^m : A \cdot \mathbf{y} = 0 \pmod q\}.$$

Suppose we have $\mathbf{y} \in \Lambda_q(A)$ and $\mathbf{y}' \in \Lambda_q^\perp(A)$, then we have $\mathbf{y} = A^\mathsf{T} \cdot \mathbf{z} \pmod q$ and $A \cdot \mathbf{y}' = 0 \pmod q$. This implies that

$$\mathbf{y}^\mathsf{T} \cdot \mathbf{y}' = (\mathbf{z}^\mathsf{T} \cdot A) \cdot \mathbf{y}' = \mathbf{z}^\mathsf{T} \cdot (A \cdot \mathbf{y}') \in q \cdot \mathbb{Z}.$$

Hence, the two lattices are, up to normalization, duals of each other. We have $\Lambda_q(A) = q \cdot \Lambda_q^\perp(A)^*$ and $\Lambda_q^\perp(A) = q \cdot \Lambda_q(A)^*$.

Example: To fix ideas, consider the following example; Let $n = 2$, $m = 3$, $q = 1009$ and set

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 5 & 6 \end{pmatrix}.$$

To define a basis B of $\Lambda_q(A)$ we can take the column-Hermite Normal Form (HNF) of the 3×5 matrix $\left(A^{\mathsf{T}} \mid q \cdot I_3\right)$ to obtain

$$B = \begin{pmatrix} 1009 & 1 & 336 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The basis of $\Lambda_q^{\perp}(A)$ is given by

$$B^* = q \cdot ((B^{\mathsf{T}})^{-1}) = \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1009 & 0 \\ -336 & 0 & 1009 \end{pmatrix}.$$

The properties of the above example hold in general; namely if q is prime and m is a bit larger than n then we have $\Delta(\Lambda_q(A)) = q^{m-n}$ and $\Delta(\Lambda_q^{\perp}(A)) = q^n$. From this, using the Gaussian Heuristic, we find for $A \in \mathbb{Z}_q^{n \times m}$ that we expect

$$\lambda_1(\Lambda_q(A)) \approx \sqrt{\frac{m}{2 \cdot \pi \cdot e}} \cdot q^{(m-n)/m},$$
$$\lambda_1(\Lambda_q^{\perp}(A)) \approx \sqrt{\frac{m}{2 \cdot \pi \cdot e}} \cdot q^{n/m}.$$

Another lattice-based problem which is of interest in cryptography, and is related to these q-ary lattices, is the Short Integer Solution problem (or SIS problem).

Definition 5.8 (Short Integer Solution)**.** *Given an integer q and vectors $\mathbf{a}_1, \ldots, \mathbf{a}_m \in (\mathbb{Z}/q\mathbb{Z})^n$ the SIS problem is to find a* short *$\mathbf{z} \in \mathbb{Z}^m$ such that*

$$z_1 \cdot \mathbf{a}_1 + \cdots + z_m \cdot \mathbf{a}_m = \mathbf{0} \pmod{q}.$$

Here "short" often means $z_i \in \{-1, 0, 1\}$.

The SIS problem is related to q-ary lattices in the following way. If we set $A = (\mathbf{a}_1, \ldots, \mathbf{a}_m) \in (\mathbb{Z}/q\mathbb{Z})^{n \times m}$ and set

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

then the SIS problem becomes the shortest vector problem for the lattice $\Lambda^{\perp}(A)$.

5.4. Coppersmith's Theorem

In this section we examine a standard tool which is used when one applies lattices to attack certain systems. Much of the work in this area is derived from initial work of Coppersmith, which was later simplified by Howgrave-Graham. Coppersmith's contribution was to provide a method to solve the following problem: Given a polynomial of degree d

$$f(x) = f_0 + f_1 \cdot x + \cdots + f_{d-1} \cdot x^{d-1} + x^d$$

over the integers and the side information that there exists a root x_0 modulo N which is small, say $|x_0| < N^{1/d}$, can one efficiently find the small root x_0? The answer is surprisingly yes, and this leads to a number of interesting cryptographic consequences.

The basic idea is to find a polynomial $h(x) \in \mathbb{Z}[x]$ which has the same root modulo N as the target polynomial $f(x)$. This new polynomial $h(x)$ should be small in the sense that the norm of its coefficients,

$$\|h\|^2 = \sum_{i=0}^{\deg(h)} h_i^2$$

should be small. If such an $h(x)$ can be found then we can appeal to the following lemma.

Lemma 5.9. *Let $h(x) \in \mathbb{Z}[x]$ denote a polynomial of degree at most n and let X and N be positive integers. Suppose*

$$\|h(X \cdot x)\| < N/\sqrt{n}$$

then if $|x_0| < X$ satisfies

$$h(x_0) = 0 \pmod{N}$$

then $h(x_0) = 0$ over the integers and not just modulo N.

Thus to solve our original problem we need to find the roots of $h(X)$ over the integers, which can be done in polynomial time via a variety of methods. Now we return to our original polynomial $f(x)$ of degree d and notice that if

$$f(x_0) = 0 \pmod{N}$$

then we also have

$$f(x_0)^k = 0 \pmod{N^k}.$$

Moreover, if we set, for some given value of m,

$$g_{u,v}(x) \leftarrow N^{m-v} \cdot x^u \cdot f(x)^v$$

then

$$g_{u,v}(x_0) = 0 \pmod{N^m}$$

for all $0 \leq u < d$ and $0 \leq v \leq m$. We then fix m and try to find $a_{u,v} \in \mathbb{Z}$ so that

$$h(x) = \sum_{u \geq 0} \left(\sum_{v=0}^{m} a_{u,v} \cdot g_{u,v}(x) \right)$$

satisfies the conditions of the above lemma. In other words we wish to find integer values of $a_{u,v}$, so that the resulting polynomial h satisfies

$$\|h(X \cdot x)\| \leq N^m/\sqrt{d \cdot (m+1)},$$

with

$$h(X \cdot x) = \sum_{u \geq 0} \left(\sum_{v=0}^{m} a_{u,v} \cdot g_{u,v}(X \cdot x) \right).$$

This is a minimization problem which can be solved using lattice basis reduction, as we shall now show in a simple example.

Example: Suppose our polynomial $f(x)$ is given by

$$f(x) = x^2 + a \cdot x + b$$

and we wish to find an x_0 such that

$$f(x_0) = 0 \pmod{N}.$$

We set $m = 2$ in the above construction and compute

$$g_{0,0}(X \cdot x) \leftarrow N^2,$$
$$g_{1,0}(X \cdot x) \leftarrow X \cdot N^2 \cdot x,$$
$$g_{0,1}(X \cdot x) \leftarrow b \cdot N + a \cdot X \cdot N \cdot x + N \cdot X^2 \cdot x^2,$$
$$g_{1,1}(X \cdot x) \leftarrow b \cdot N \cdot X \cdot x + a \cdot N \cdot X^2 \cdot x^2 + N \cdot X^3 \cdot x^3,$$
$$g_{0,2}(X \cdot x) \leftarrow b^2 + 2 \cdot b \cdot a \cdot X \cdot x + (a^2 + 2 \cdot b) \cdot X^2 \cdot x^2 + 2 \cdot a \cdot X^3 \cdot x^3 + X^4 \cdot x^4,$$
$$g_{1,2}(X \cdot x) \leftarrow b^2 \cdot X \cdot x + 2 \cdot b \cdot a \cdot X^2 \cdot x^2 + (a^2 + 2 \cdot b) \cdot X^3 \cdot x^3 + 2 \cdot a \cdot X^4 \cdot x^4 + X^5 \cdot x^5.$$

We are looking for a linear combination of the above six polynomials such that the resulting polynomial has small coefficients. Hence we are led to look for small vectors in the lattice generated

by the columns of the following matrix, where each column represents one of the polynomials above and each row represents a power of x,

$$A = \begin{pmatrix} N^2 & 0 & b \cdot N & 0 & b^2 & 0 \\ 0 & X \cdot N^2 & a \cdot X \cdot N & b \cdot N \cdot X & 2 \cdot a \cdot b \cdot X & X \cdot b^2 \\ 0 & 0 & N \cdot X^2 & a \cdot N \cdot X^2 & (a^2 + 2 \cdot b) \cdot X^2 & 2 \cdot a \cdot b \cdot X^2 \\ 0 & 0 & 0 & N \cdot X^3 & 2 \cdot a \cdot X^3 & (a^2 + 2 \cdot b) \cdot X^3 \\ 0 & 0 & 0 & 0 & X^4 & 2 \cdot a \cdot X^4 \\ 0 & 0 & 0 & 0 & 0 & X^5 \end{pmatrix}.$$

This matrix has determinant equal to

$$\det(A) = N^6 \cdot X^{15},$$

and so applying the LLL algorithm to this matrix we obtain a new lattice basis B. The first vector $\mathbf{b}_1$ in B will satisfy

$$\|\mathbf{b}_1\| \leq 2^{6/4} \cdot \det(A)^{1/6} = 2^{3/2} \cdot N \cdot X^{5/2}.$$

So if we set $\mathbf{b}_1 = A \cdot \mathbf{u}$, with $\mathbf{u} = (u_1, u_2, \ldots, u_6)^\mathsf{T}$, then we form the polynomial

$$h(x) = u_1 \cdot g_{0,0}(x) + u_2 \cdot g_{1,0}(x) + \cdots + u_6 \cdot g_{1,2}(x)$$

then we will have

$$\|h(X \cdot x)\| \leq 2^{3/2} \cdot N \cdot X^{5/2}.$$

To apply Lemma 5.9 we will require that

$$2^{3/2} \cdot N \cdot X^{5/2} < N^2/\sqrt{6}.$$

Hence by determining an integer root of $h(x)$ we will determine the small root x_0 of $f(x)$ modulo N, assuming that

$$|x_0| \leq X = \frac{N^{2/5}}{48^{1/5}}.$$

In particular this will work when $|x_0| \leq N^{0.39}$.

A similar technique can be applied to any polynomial of degree d so as to obtain the following.

Theorem 5.10 (Coppersmith)**.** *Let $f \in \mathbb{Z}[x]$ be a monic polynomial of degree d and N an integer. If there is some root x_0 of f modulo N such that $|x_0| \leq X = N^{1/d-\epsilon}$ then one can find x_0 in time polynomial in $\log N$ and $1/\epsilon$, for fixed values of d.*

Similar considerations apply to polynomials in two variables the analogue of Lemma 5.9 is as follows:

Lemma 5.11. *Let $h(x, y) \in \mathbb{Z}[x, y]$ denote a sum of at most w monomials and suppose*

$$h(x_0, y_0) = 0 \pmod{N^e}$$

for some positive integers N and e where the integers x_0 and y_0 satisfy

$$|x_0| < X \quad \text{and} \quad |y_0| < Y$$

and

$$\|h(X \cdot x, Y \cdot y)\| < N^e/\sqrt{w}.$$

Then $h(x_0, y_0) = 0$ holds over the integers.

However, the analogue of Theorem 5.10 then becomes only a heuristic result.

Chapter Summary

- Lattices are discrete analogues of vector spaces; as such they have a shortest non-zero vector.
- Lattice basis reduction often allows us to find the shortest non-zero vector in a given lattice, thus lattice reduction allows us to solve the shortest vector problem.
- Other lattice problems, such as the CVP and BDD problems, can also be solved if we can find good bases of lattices.
- In small dimensions the LLL algorithm works very well. In larger dimensions, whilst it is fast, it does not produce such a good output lattice.
- The SIS problem is related to the SVP problem in q-ary lattices.
- Coppersmith's Theorem allows us to solve a modular polynomial equation when the solution is known to be small. The method works by building a lattice depending on the polynomial and then applying lattice basis reduction to obtain short vectors within this lattice.

Further Reading

A complete survey of lattice-based methods in cryptography is given in the survey article by Nguyen and Stern. The main paper on Coppersmith's approach is by Coppersmith himself, however the approach was simplified somewhat in the paper of Howgrave-Graham.

D. Coppersmith. *Small solutions to polynomial equations, and low exponent RSA vulnerabilities.* J. Cryptology, **10**, 233–260, 1997.

P. Nguyen and J. Stern. *The two faces of lattices in cryptology.* In CALC '01, LNCS 2146, 146–180, Springer, 2001.

N. Howgrave-Graham. *Finding small roots of univariate modular equations revisited.* In Cryptography and Coding, LNCS 1355, 131–142, Springer, 1997.