An Introduction to the Theory of Lattices and Applications to Cryptography

Joseph H. Silverman

Brown University and
NTRU Cryptosystems, Inc.

Summer School on
Computational Number Theory and Applications to Cryptography
University of Wyoming

June 19 – July 7, 2006

Outline

- Introduction
- Lattices and Lattice Problems
- Fundamental Lattice Theorems
- Lattice Reduction and the LLL Algorithm
- Knapsack Cryptosystems and Lattice Cryptanalysis
- Lattice-Based Cryptography
- The NTRU Public Key Cryptosystem
- Convolution Modular Lattices and NTRU Lattices
- Further Reading

Public Key Cryptography and Hard Mathematical Problems

- Underlying every public key cryptosystem is a hard mathematical problem.
- Unfortunately, in very few instances is there a proof that breaking the cryptosystem is **equivalent** to solving the hard mathematical problem. But we won't worry about that for now!
- The best known examples are:

RSA	Integer Factorization Problem
Diffie-Hellman	Discrete Logarithm Problem in $\mathbb{F}_q^*$
ECC	Discrete Logarithm Problem on an Elliptic Curve

A Different Hard Problem for Cryptography

- There are many other hard mathematical problems that one might use for cryptography.
- An appealing class of problems involves finding closest and shortest vectors in lattices.
- The general **Closest Vector Problem** (**CVP**) is known to be NP-hard and the **Shortest Vector Problem** (**SVP**) is NP-hard under a randomized reduction hypothesis.
- In this lecture I will discuss the mathematics of lattices, alogrithms to solve SVP and CVP, and give some applications to breaking cryptosystems. In the next lecture I will describe some cryptosystems that are based on the difficulty of solving SVP and CVP.

Lattices
and
Lattice Problems

Lattices — Definition and Notation

Definition. A lattice L of dimension n is a maximal discrete subgroup of $\mathbb{R}^n$.

Equivalently, a lattice is the $\mathbb{Z}$-linear span of a set of n linearly independent vectors:

$$L = \{a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \cdots + a_n\mathbf{v}_n : a_1, a_2, \ldots, a_n \in \mathbb{Z}\}.$$

The vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ are a **Basis for** L. Lattices have many bases. Some bases are "better" than others.

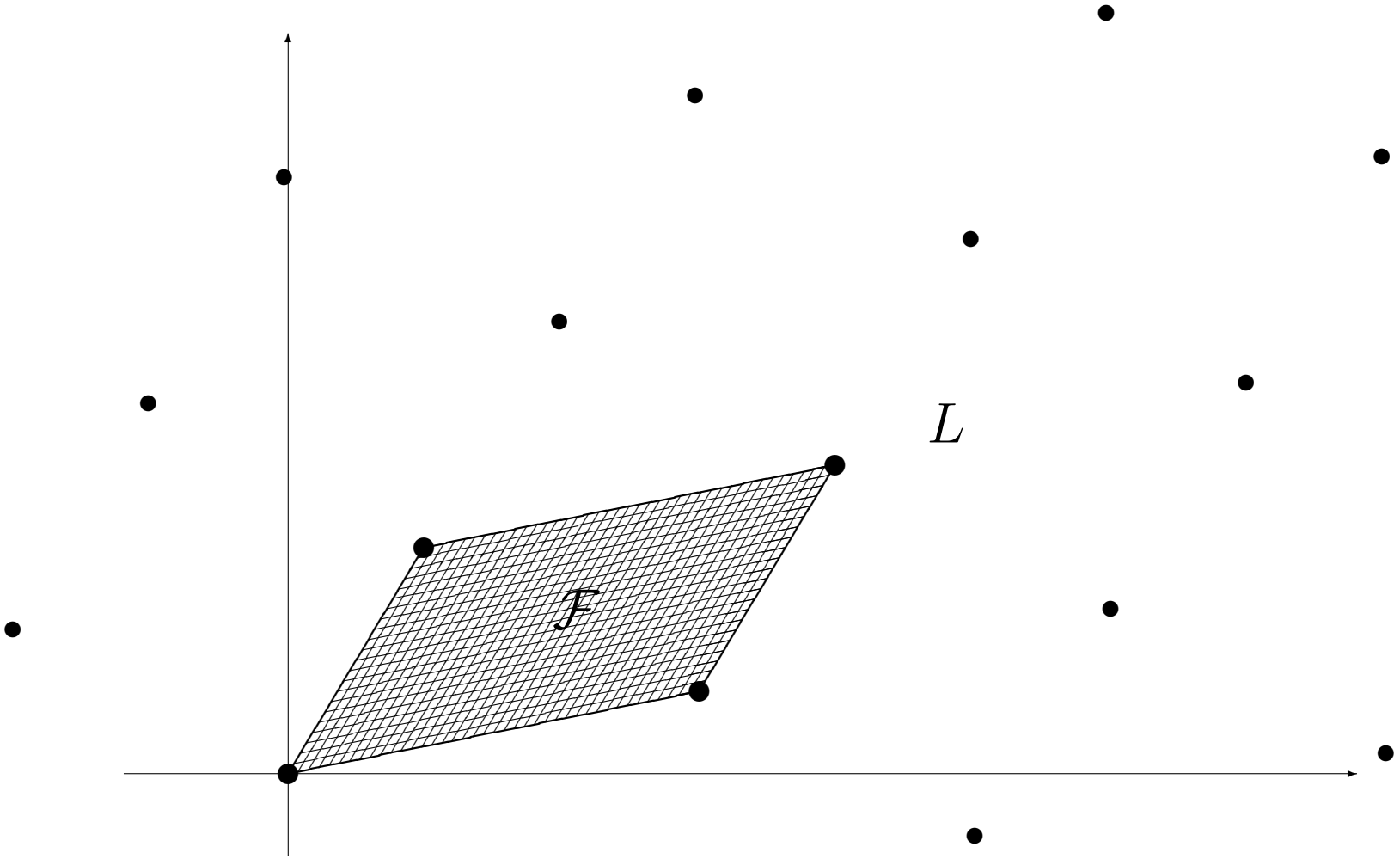
A **fundamental domain** for the quotient $\mathbb{R}^n/L$ is the set

$$\mathcal{F}(L) = \{t_1\mathbf{v}_1 + t_2\mathbf{v}_2 + \cdots + t_n\mathbf{v}_n : 0 \le t_i < 1\}.$$

The **Discriminant** (or "volume") of L is

$$\mathrm{Disc}(L) = \mathrm{Volume}(\mathcal{F}(L)) = \det\big(\mathbf{v}_1|\mathbf{v}_2|\cdots|\mathbf{v}_n\big).$$

A Two Dimensional Example



A 2-dimensional lattice L with fundamental domain $\mathcal{F}$

The Two Fundamental Hard Lattice Problems

Let L be a lattice of dimension n. The two most important computational problems are:

Shortest Vector Problem (SVP)

Find a shortest nonzero vector in L.

Closest Vector Problem (CVP)

Given a vector $\mathbf{t} \in \mathbb{R}^n$ not in L, find a vector in L that is closest to $\mathbf{t}$.

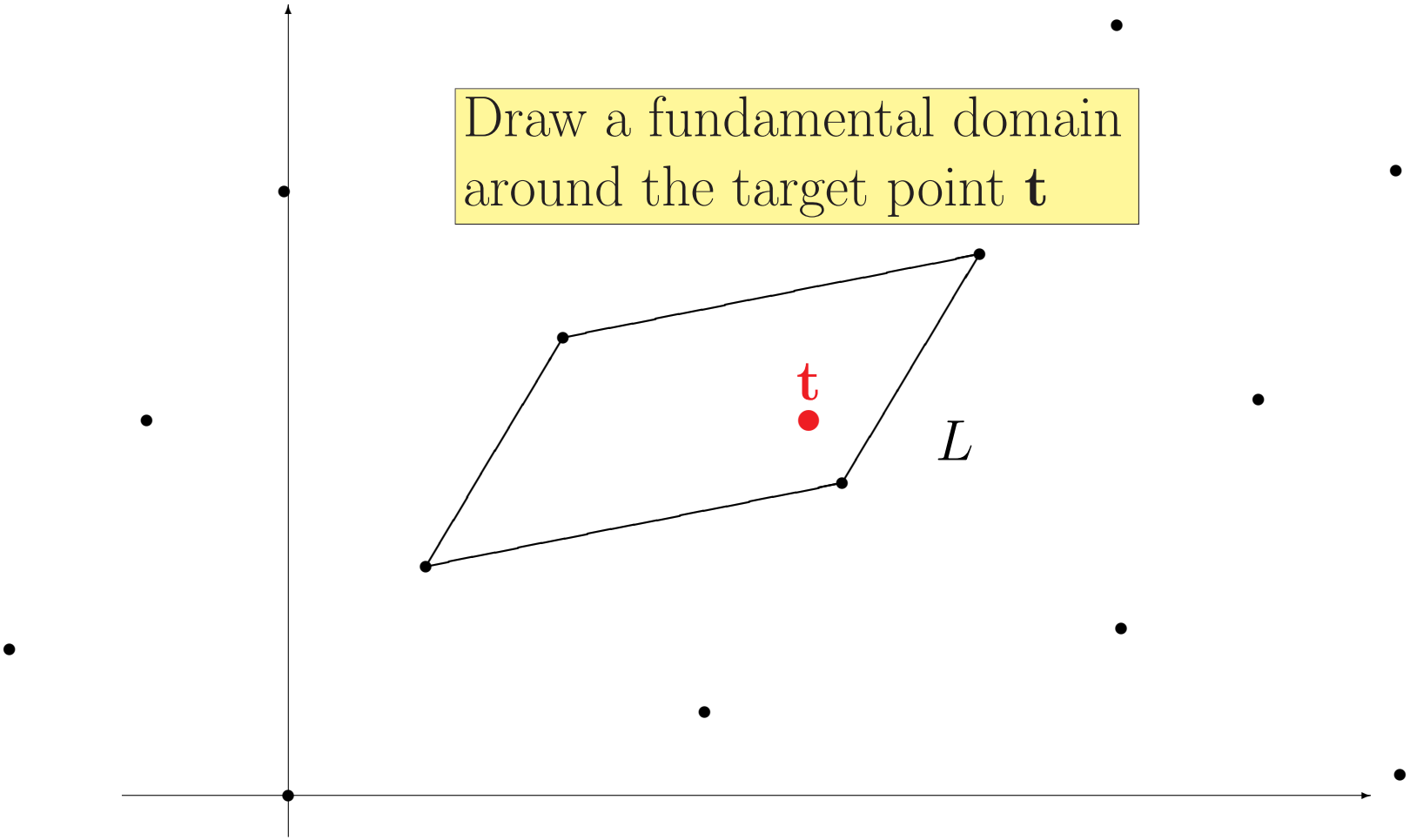
The Approximate Closest Vector Problem (apprCVP)

is to find a vector $\mathbf{v} \in L$ so that $\|\mathbf{v} - \mathbf{t}\|$ is small. For example,

$$\|\mathbf{v} - \mathbf{t}\| \leq \kappa \min_{\mathbf{w} \in L} \|\mathbf{w} - \mathbf{t}\|$$

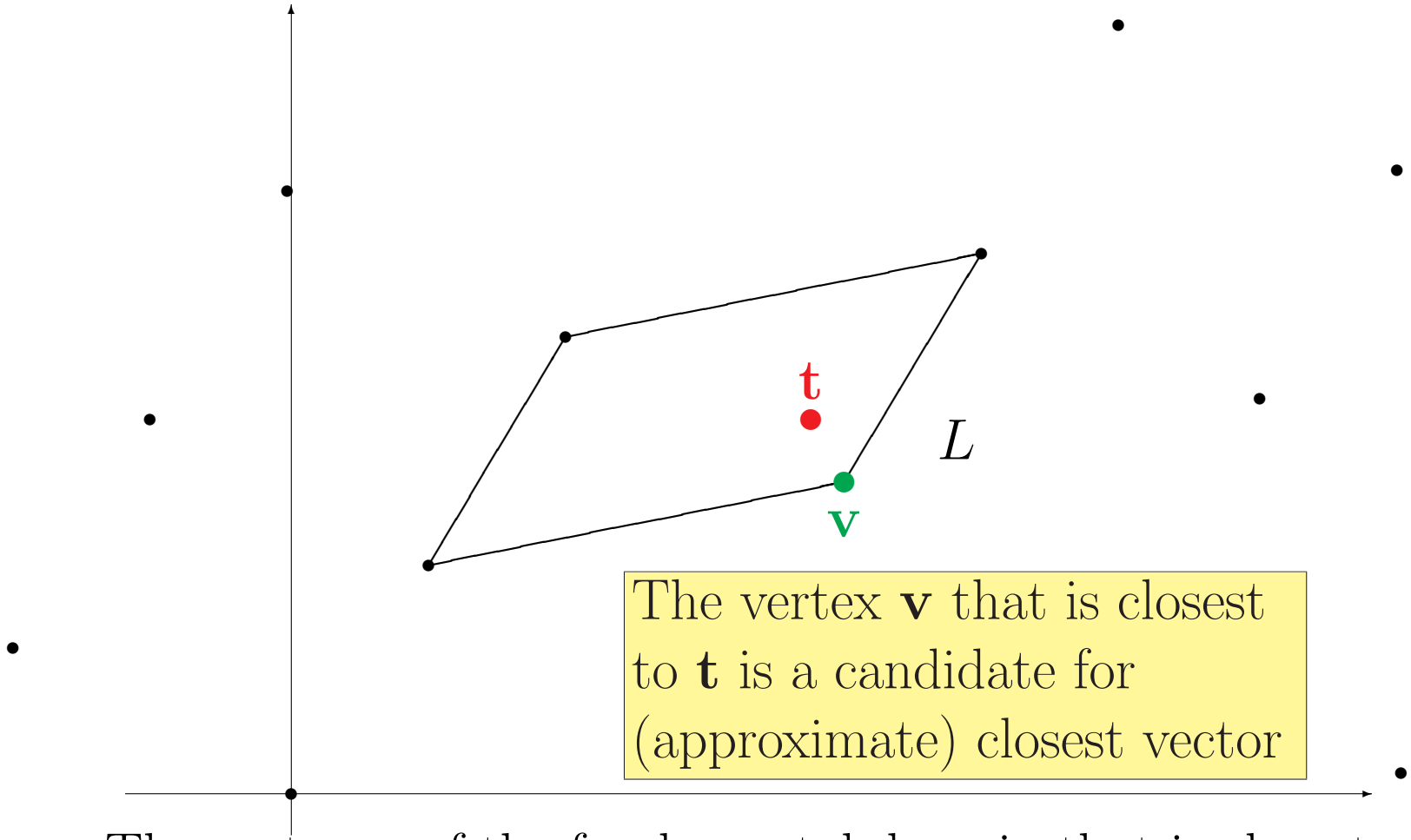
for a small constant κ.

Using a Basis to Try to Solve the Closest Vector Problem



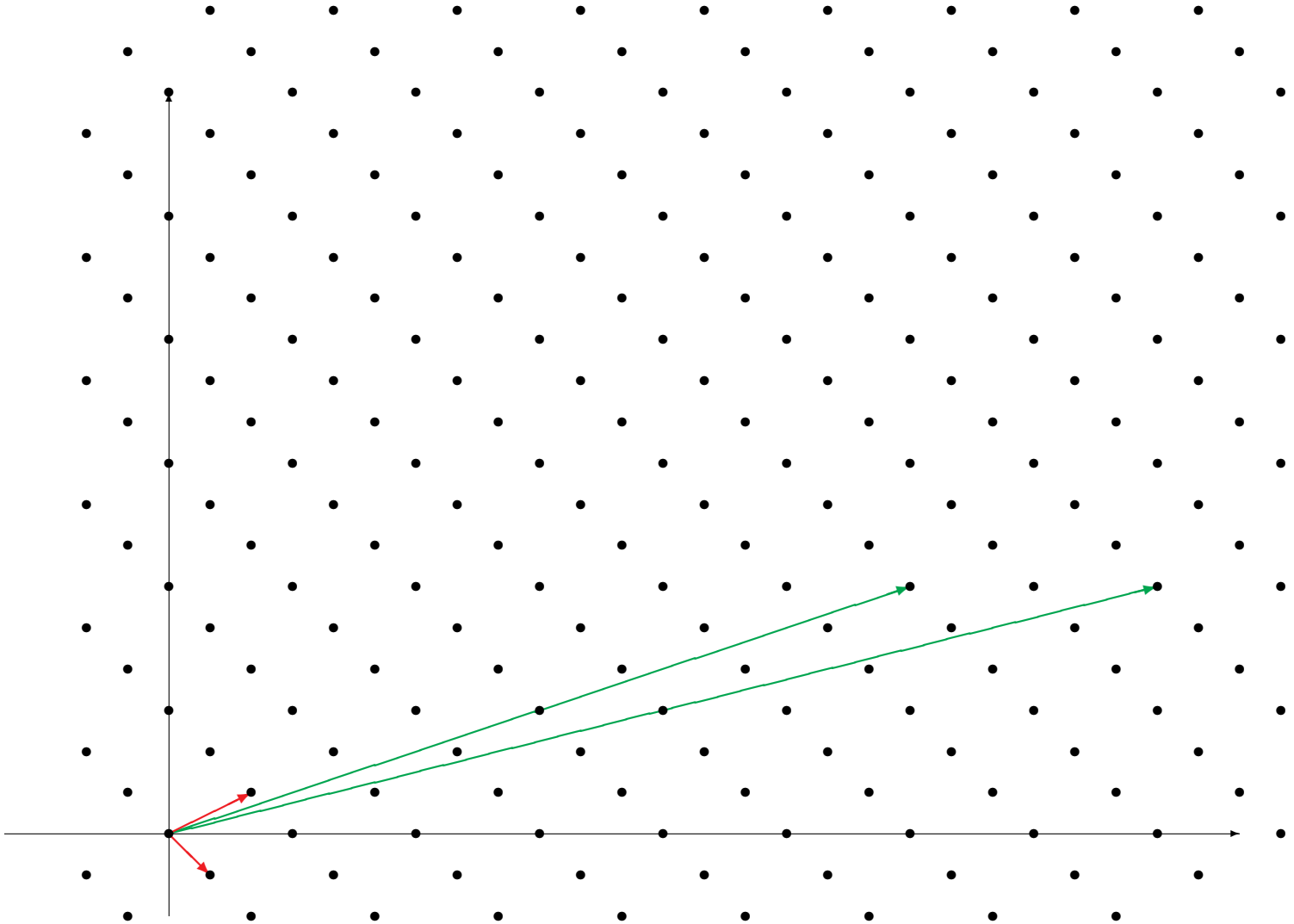
Use a basis for the lattice to draw a parallelogram around the target point.

Using a Basis to Try to Solve the Closest Vector Problem



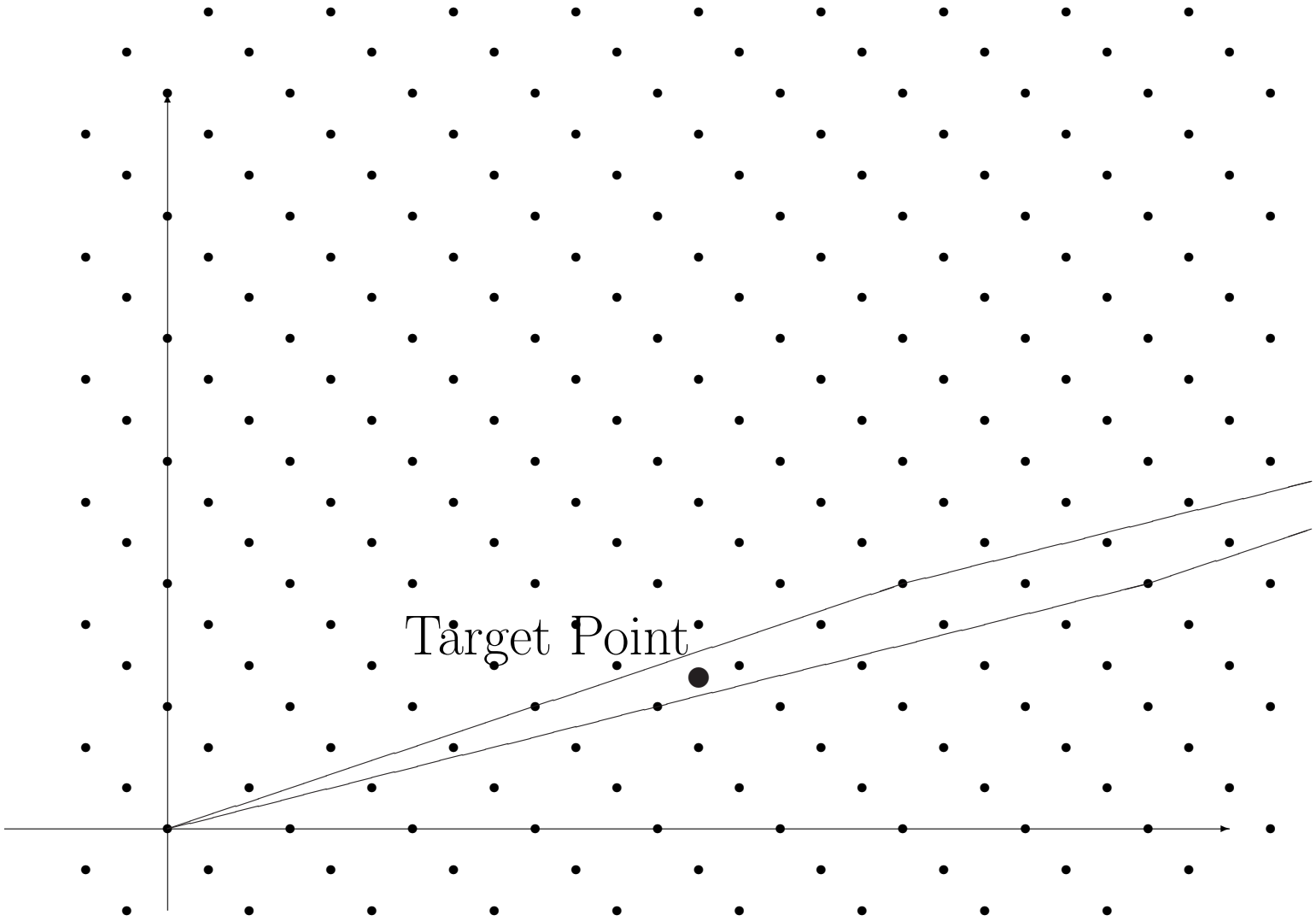
The vertex $\mathbf{v}$ of the fundamental domain that is closest to $\mathbf{t}$ will be a close lattice point if the basis is "good", meaning if the basis consists of short vectors that are reasonably orthogonal to one another.

Good and Bad Bases



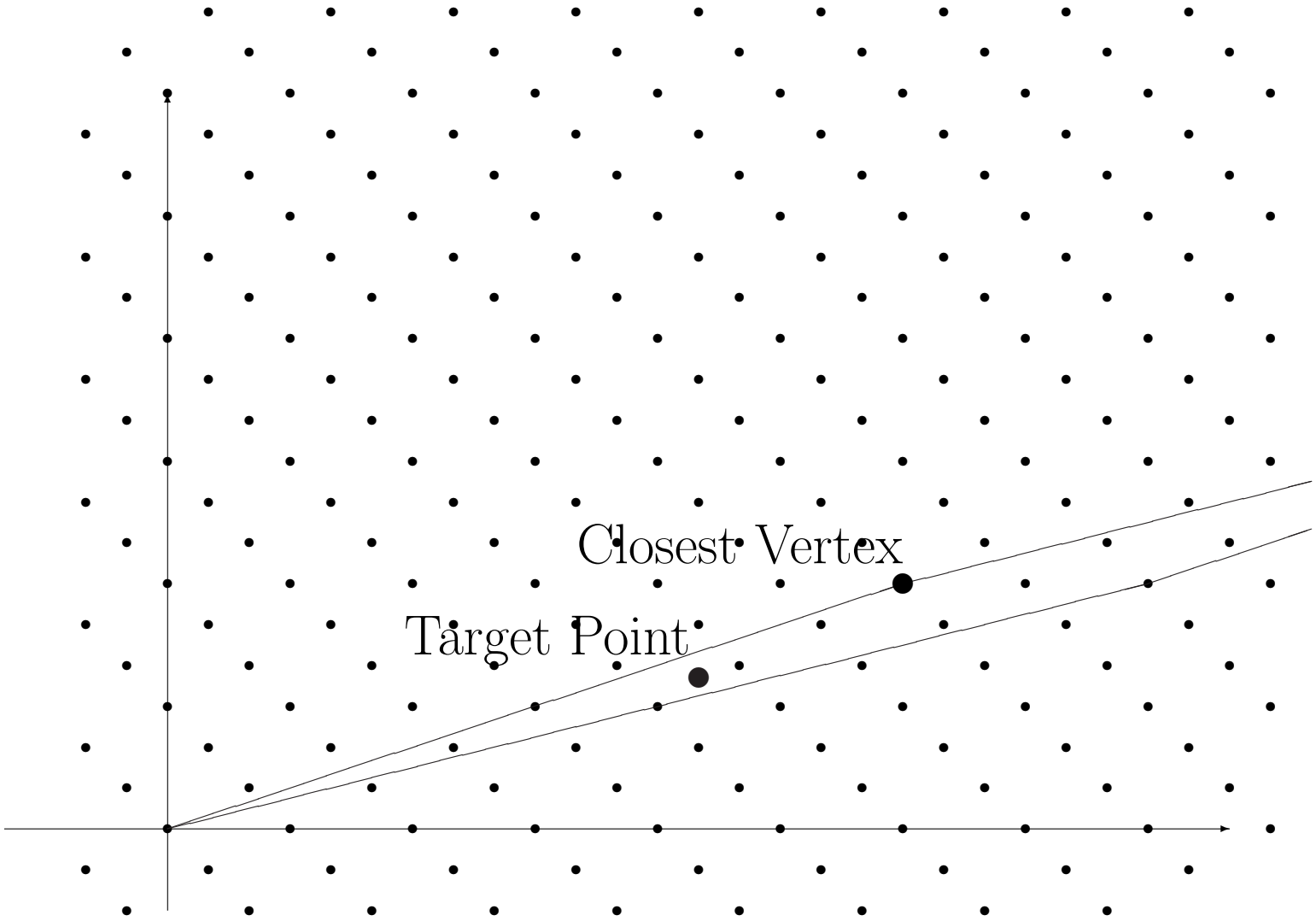
A "good" basis and a "bad" basis

The Closest Vertex Method Using a Bad Basis



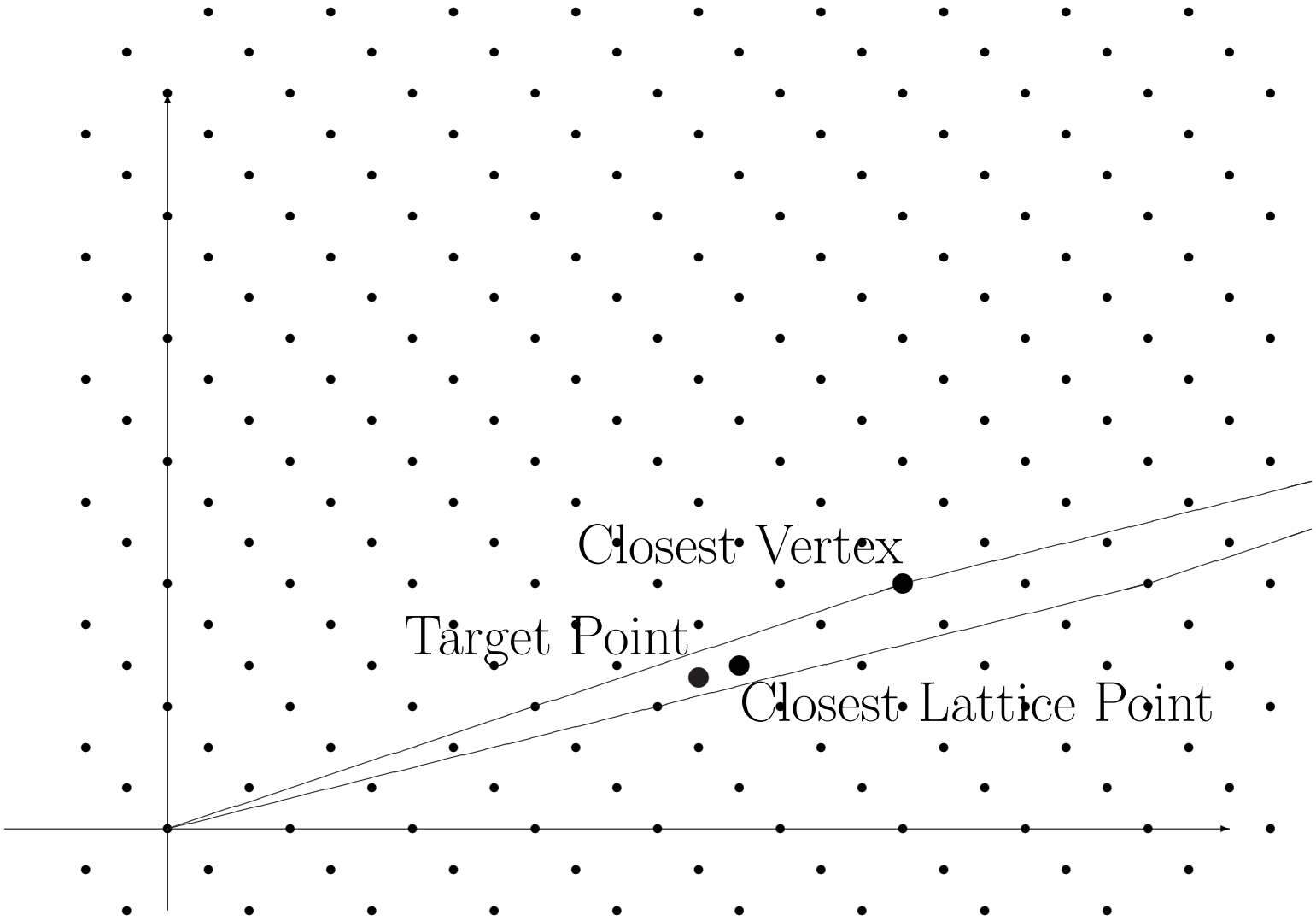
Here is the parallelogram spanned by a "bad" basis and a CVP target point.

The Closest Vertex Method Using a Bad Basis



It is easy to find the vertex of the parallelogram that is closest to the target point.

The Closest Vertex Method Using a Bad Basis



However, the lattice point that actually solves CVP is much closer to the target than the closest vertex.

Theory and Practice

Lattices, SVP and CVP, have been intensively studied for more than 100 years, both as intrinsic mathematical problems and for applications in pure and applied mathematics, physics and cryptography.

The theoretical study of lattices is often called the

Geometry of Numbers,

a name bestowed on it by Minkowski in his 1910 book *Geometrie der Zahlen*.

The practical process of finding short(est) or close(st) vectors in lattices is called **Lattice Reduction**.

Lattice reduction methods have been extensively developed for applications to number theory, computer algebra, discrete mathematics, applied mathematics, combinatorics, cryptography,...

Fundamental Lattice Theorems

How Orthogonal is a Basis of a Lattice?

Hademard's Inequality. Let $\mathbf{v}_1, \ldots, \mathbf{v}_n$ be any basis for L. Then

$$\text{Disc}(L) \leq \|\mathbf{v}_1\| \cdot \|\mathbf{v}_2\| \cdots \|\mathbf{v}_n\|.$$

Hadamard's inequality is true because the volume of a parallelopiped is never greater than the product of the lengths of its sides.

Hadamard's inequality is an equality if and only if the basis vectors are orthogonal (perpendicular) to one another. The extent to which it is an inequality measures the extent to which the basis is nonorthogonal.

A famous theorem of Hermite says that every lattice has a basis that is reasonably orthogonal, where the amount of nonorthogonality is bounded solely in terms of the dimension.

A Fundamental Lattice Theorem from the 19th Century

Theorem. (Hermite): There is a constant γ_n so that for all lattices L of dimension n:

(a) There is a nonzero vector $\mathbf{v} \in L$ satisfying

$$\|\mathbf{v}\| \leq \gamma_n \operatorname{Disc}(L)^{1/n}.$$

(b) There is a basis $\mathbf{v}_1, \ldots, \mathbf{v}_n$ for L satisfying

$$\|\mathbf{v}_1\| \cdot \|\mathbf{v}_2\| \cdots \|\mathbf{v}_n\| \leq \gamma_n^{n/2} \operatorname{Disc}(L).$$

The constant γ_n is called **Hermite's constant**. It is known that for large n,

$$\sqrt{\frac{n}{2\pi e}} \lesssim \gamma_n \lesssim \sqrt{\frac{n}{\pi e}},$$

but the exact value of γ_n is known only for $n \leq 8$.

Finding Points in Lattices — A Theoretical Result

I will start by sketching the proof of the following important result. Then Hermite's Theorem will be an immediate consequence.

Theorem. (Minkowski): Let L be a lattice of dimension n. Then every compact convex symmetric region $\mathcal{R}$ of volume at least $2^n \operatorname{Disc}(L)$ contains a nonzero lattice point.

The region $\mathcal{R}$ in Minkowski's Theorem is assumed to have the following three properties:

Compact: closed and bounded

Convex: $\mathbf{v}, \mathbf{w} \in \mathcal{R} \implies$ line segment $\overline{\mathbf{vw}} \subset \mathcal{R}$

Symmetric: $\mathbf{v} \in \mathcal{R} \implies -\mathbf{v} \in \mathcal{R}$

Proof of Minkowski's Theorem

Let $\mathcal{R} \subset \mathbb{R}^n$ be a compact convex symmetric region with

$$\mathrm{Vol}(\mathcal{R}) > 2^n \,\mathrm{Disc}(L).$$

Goal: Prove that $\mathcal{R}$ contains a nonzero lattice point.

Let $\mathbf{v}_1, \ldots, \mathbf{v}_n$ be a basis for L and let

$$\mathcal{F} = \left\{ t_1\mathbf{v}_1 + \cdots + t_n\mathbf{v}_n : 0 \le t_i < 1 \right\}$$

be the usual fundamental domain for L.

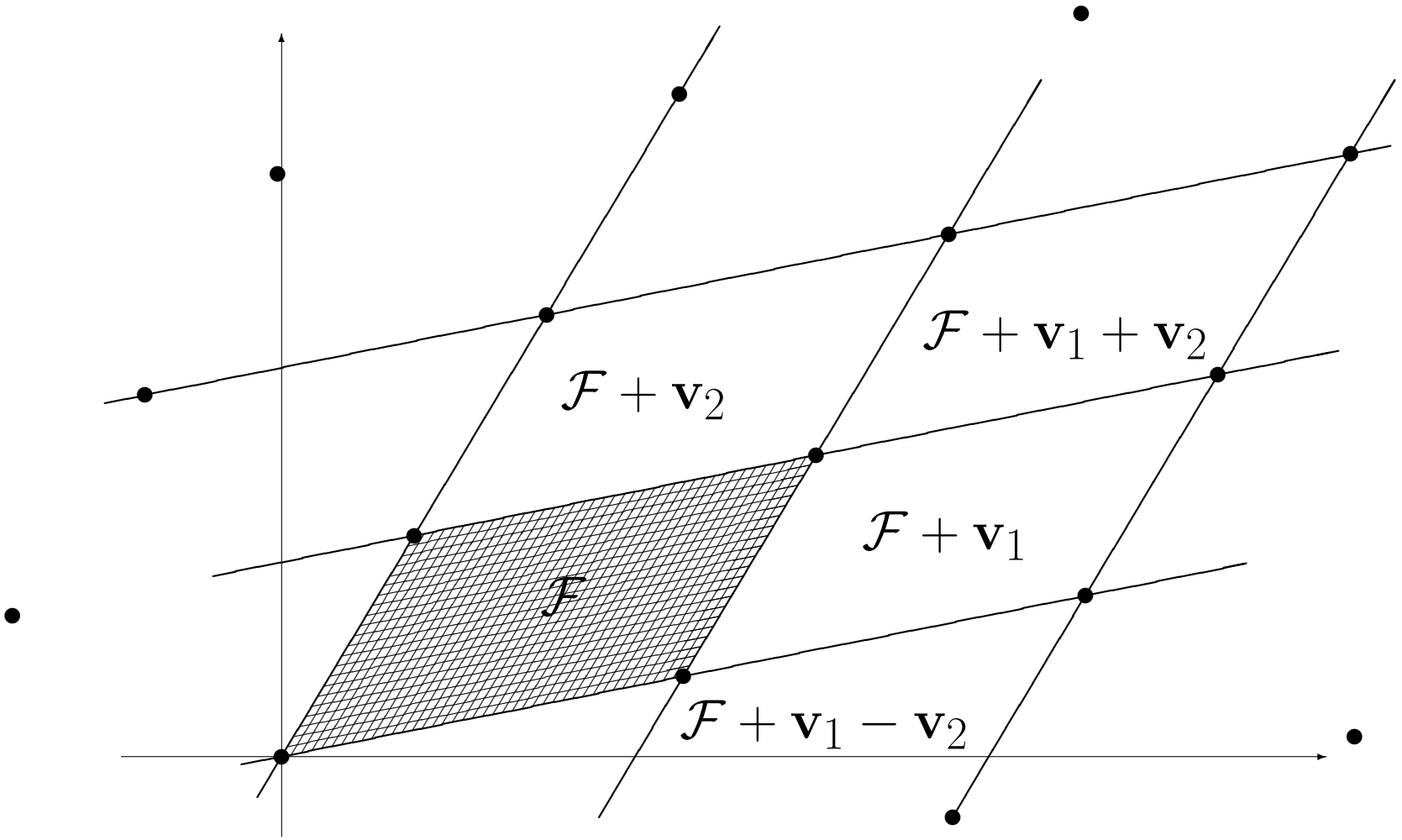
For each $\mathbf{v} \in L$ we look at the translation of $\mathcal{F}$,

$$\mathcal{F} + \mathbf{v} = \{\mathbf{w} + \mathbf{v} : \mathbf{w} \in \mathcal{F}\}.$$

As $\mathbf{v}$ varies over L, the translates $\mathcal{F} + \mathbf{v}$ cover all of $\mathbb{R}^n$,

$$\bigcup_{\mathbf{v}\in L} (\mathcal{F} + \mathbf{v}) = \mathbb{R}^n.$$

Translations of $\mathcal{F}$ By Vectors in L



Translating the fundamental domain $\mathcal{F}$ using the vectors in the lattice L covers all of $\mathbb{R}^n$.

Proof of Minkowski's Theorem (continued)

In particular, each $\mathbf{r} \in \mathcal{R}$ can be written uniquely in the form $\mathbf{r} = \mathbf{v_r} + \mathbf{w_r}$ with $\mathbf{v_r} \in L$ and $\mathbf{w_r} \in \mathcal{F}$.

In other words, take $\mathbf{r}$ and translate it by an element of L so that it lies in $\mathcal{F}$.

We dilate (shrink) $\mathcal{R}$ by a factor of 2,

$$\tfrac{1}{2}\mathcal{R} = \{\tfrac{1}{2}\mathbf{r} : \mathbf{r} \in \mathcal{R}\},$$

and consider the map

$$\tfrac{1}{2}\mathcal{R} \longrightarrow \mathcal{F}, \qquad \tfrac{1}{2}\mathbf{r} \longmapsto \mathbf{w}_{\frac{1}{2}\mathbf{r}}.$$

Shrinking by a factor of 2 changes volume by a factor of 2^n, so $\mathrm{Vol}(\tfrac{1}{2}\mathcal{R}) = \frac{1}{2^n}\mathrm{Vol}(\mathcal{R}) > \mathrm{Vol}(\mathcal{F})$.

So there must be two *different* points $\frac{1}{2}\mathbf{r}_1$ and $\frac{1}{2}\mathbf{r}_2$ in $\frac{1}{2}\mathcal{R}$ with the same image in $\mathcal{F}$.

Proof of Minkowski's Theorem (continued)

We have found two points in $\frac{1}{2}\mathcal{R}$ satisfying

$$\tfrac{1}{2}\mathbf{r}_1 = \mathbf{v}_1 + \mathbf{w} \quad \text{and} \quad \tfrac{1}{2}\mathbf{r}_2 = \mathbf{v}_2 + \mathbf{w}$$
$$\text{with} \quad \mathbf{v}_1, \mathbf{v}_2 \in L \quad \text{and} \quad \mathbf{w} \in \mathcal{F}.$$

Subtracting them yields a nonzero vector

$$\tfrac{1}{2}\mathbf{r}_1 - \tfrac{1}{2}\mathbf{r}_2 = \mathbf{v}_1 - \mathbf{v}_2 \in L.$$

We now observe that

$$\underbrace{\tfrac{1}{2}\mathbf{r}_1 + \overbrace{\left(-\tfrac{1}{2}\mathbf{r}_2\right)}^{\substack{\mathcal{R} \text{ is symmetric} \\ \text{so } -\mathbf{r}_2 \text{ is in } \mathcal{R}}}}_{\substack{\text{this is the midpoint of the line} \\ \text{segment from } \mathbf{r}_1 \text{ to } -\mathbf{r}_2, \\ \text{so it is in } \mathcal{R} \text{ by convexity}}}$$

Hence

$$\mathbf{0} \neq \mathbf{v}_1 - \mathbf{v}_2 \in \mathcal{R} \cap L.$$

Proof of Minkowski's Theorem (finalé)

This completes the proof of Minkowski's Theorem assuming $\mathrm{Vol}(\mathcal{R}) > 2^n \,\mathrm{Disc}(L)$.

To deal with regions satisfying

$$\mathrm{Vol}(\mathcal{R}) = 2^n \,\mathrm{Disc}(L)$$

we apply our result to find nonzero points

$$\mathbf{0} \neq \mathbf{v}_k \in \left(1 + \tfrac{1}{k}\right)\mathcal{R} \cap L \quad \text{for each } k = 1, 2, 3, \ldots.$$

The lattice points $\mathbf{v}_1, \mathbf{v}_2, \ldots$ are all in $2\mathcal{R}$, so there are only finitely many possibilities for them. Hence there is a nonzero lattice point $\mathbf{v} \in L$ in the intersection

$$\bigcap_{k=1}^{\infty} \left(1 + \tfrac{1}{k}\right) \mathcal{R} = \mathcal{R}.$$

Note that they are equal because $\mathcal{R}$ is compact. QED

Corollary. (Hermite's Theorem Part (a)) A lattice L of dimension n always has a nonzero point $\mathbf{v} \in L$ of length at most

$$\|\mathbf{v}\| \lesssim \sqrt{\frac{2n}{\pi e}}\,\mathrm{Disc}(L)^{1/n}$$

Proof. Let $\mathcal{B}_R \subset \mathbb{R}^n$ be a ball of radius R,

$$\big(\{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x}\| \le R\}\big).$$

If n is reasonably large, then $\mathcal{B}_R$ has volume

$$\mathrm{Vol}(\mathcal{B}_R) \approx \left(\frac{2\pi e}{n}\right)^{n/2} R^n.$$

Hence if we take $R \approx \sqrt{2n/\pi e}\,\mathrm{Disc}(L)^{1/n}$, then we get

$$\mathrm{Vol}(\mathcal{B}_R) \gtrsim 2^n\,\mathrm{Disc}(L).$$

Minkowski's Theorem tells us that $\mathcal{B}_R$ contains a nonzero lattice point. QED

The Successive Minima of a Lattice

Suppose that we select vectors in L as:

$$\begin{aligned}
\mathbf{v}_1 &= \text{shortest nonzero vector in } L,\\
\mathbf{v}_2 &= \text{shortest vector in } L \text{ linearly independent of } \mathbf{v}_1,\\
\mathbf{v}_3 &= \text{shortest vector in } L \text{ linearly independent of } \mathbf{v}_1, \mathbf{v}_2,\\
&\vdots \qquad\qquad\qquad\qquad \vdots\\
\mathbf{v}_n &= \text{shortest vector in } L \text{ linearly independent}\\
&\qquad\qquad\qquad\qquad\qquad \text{of } \mathbf{v}_1, \mathbf{v}_2 \ldots \mathbf{v}_{n-1}.
\end{aligned}$$

The lengths

$$\lambda_1 = \|\mathbf{v}_1\|, \;\; \lambda_2 = \|\mathbf{v}_2\|, \;\; \ldots, \;\; \lambda_n = \|\mathbf{v}_n\|$$

are called the **successive minima** of the lattice L. In particular, $\lambda_1 = \lambda_1(L)$ is the length of a shortest nonzero vector. We proved that

$$\lambda_1(L) \le \sqrt{\frac{2n}{\pi e}}\,\mathrm{Disc}(L)^{1/n}.$$

Lattice Reduction and the LLL Algorithm

Solving SVP and CVP in Practice

- The shortest vector problem (SVP) and the closest vector problems (CVP) are clearly closely related. In practice, CVP seems slightly harder than SVP.
- If the dimension of the lattice L is large, both SVP and CVP are very difficult to solve.
- In full generality, CVP is known to be NP-hard and SVP is NP-hard under a randomized reduction hypothesis.
- **Lattice Reduction** is the name given to the practical problem of solving SVP and CVP, or more generally of finding reasonably short vectors and reasonably good bases.

Algorithms to (Approximately) Solve SVP

- The best lattice reduction methods currently known are based on the **LLL Algorithm** of Lenstra, Lenstra, and Lovász, orginally described in *Mathematische Annalen* **261** (1982), 515-534

- LLL finds moderately short lattice vectors in polynomial time. This suffices for many applications.

- However, finding very short (or very close) vectors is currently still exponentially hard.

- It is worth noting that current lattice reduction algorithms such as LLL are highly sequential. Thus they are not distributable (although somewhat parallelizable). Further, there are no quantum algorithms known to solve SVP or CVP.

The Gram-Schmidt Orthogonalization Process

It is quite easy to turn a given basis $\mathbf{v}_1, \ldots, \mathbf{v}_n$ of $\mathbb{R}^n$ into a basis whose vectors are pairwise orthogonal. This process, which you learned when you took linear algebra, is called the

Gram-Schmidt Orthogonalization Algorithm

$$
\begin{aligned}
\mathbf{v}_1^* &= \mathbf{v}_1 \\
\mathbf{v}_2^* &= \mathbf{v}_2 - \frac{\mathbf{v}_2 \cdot \mathbf{v}_1^*}{\|\mathbf{v}_1^*\|^2}\mathbf{v}_1^* \\
\mathbf{v}_3^* &= \mathbf{v}_3 - \frac{\mathbf{v}_3 \cdot \mathbf{v}_2^*}{\|\mathbf{v}_2^*\|^2}\mathbf{v}_2^* - \frac{\mathbf{v}_3 \cdot \mathbf{v}_1^*}{\|\mathbf{v}_1^*\|^2}\mathbf{v}_1^* \\
&\vdots \qquad\qquad\qquad\qquad \ddots \\
\mathbf{v}_n^* &= \mathbf{v}_n - \frac{\mathbf{v}_n \cdot \mathbf{v}_{n-1}^*}{\|\mathbf{v}_{n-1}^*\|^2}\mathbf{v}_{n-1}^* - \frac{\mathbf{v}_n \cdot \mathbf{v}_{n-2}^*}{\|\mathbf{v}_{n-2}^*\|^2}\mathbf{v}_{n-2}^* \cdots - \frac{\mathbf{v}_n \cdot \mathbf{v}_1^*}{\|\mathbf{v}_1^*\|^2}\mathbf{v}_1^*
\end{aligned}
$$

Intuition:

$\mathbf{v}_i^* =$ Projection of $\mathbf{v}_i$ onto $\mathrm{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-1})^{\perp}$.

The Size and Quasiorthogonality Conditions

If some coefficient in the Gram-Schmidt process satisfies

$$\frac{|\mathbf{v}_i \cdot \mathbf{v}_j^*|}{\|\mathbf{v}_j^*\|^2} > \frac{1}{2},$$

then replacing $\mathbf{v}_i$ by

$$\mathbf{v}_i - a\mathbf{v}_j \quad \text{for an appropriate } a \in \mathbb{Z}$$

makes the coefficient smaller. We say that a basis satisfies the Size Condition if

$$\text{Size Condition:} \qquad \frac{|\mathbf{v}_i \cdot \mathbf{v}_j^*|}{\|\mathbf{v}_j^*\|^2} \leq \frac{1}{2} \quad \text{for all } j < i.$$

To balance this, we want the basis vectors to be somewhat orthogonal to one another, so we impose the

$$\text{QuasiOrthogonality Condition:} \quad \|\mathbf{v}_{i+1}^*\| \geq \frac{\sqrt{3}}{2}\|\mathbf{v}_i^*\|.$$

The Lovász Condition

Theorem. (Hermite) Every lattice has a basis satisfying both the Size Condition and the QuasiOrthogonality Condition.

Unfortunately, the best known algorithms to find such a basis are exponential in the dimension.

So we relax the QuasiOrthogonality Condition to

Lovász Condition: $\|\mathbf{v}_{i+1}^*\| \geq \sqrt{\frac{3}{4} - \frac{|\mathbf{v}_{i+1} \cdot \mathbf{v}_i^*|^2}{\|\mathbf{v}_i^*\|^2}}\, \|\mathbf{v}_i^*\|.$

What a mess, right! But geometrically the Lovász Condition says that

$$\begin{aligned}\text{Projection of } \mathbf{v}_{i+1} \text{ onto } &\text{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-1})^\perp \\ &\geq \frac{3}{4} \cdot \text{Projection of } \mathbf{v}_i \text{ onto } \text{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_{i-1})^\perp.\end{aligned}$$

The LLL Algorithm

Theorem. (Lenstra,Lenstra,Lovász) There is a polynomial time algorithm that finds a basis for L satisfying both the Size Condition and the Lovász Condition. Such bases are called **LLL Reduced Bases**.

[1] $k = 2$
[2] LOOP WHILE $k < n$
[3] Replace $\mathbf{v}_1, \ldots, \mathbf{v}_k$ with linear combinations so the Size Condition is true
[4] If the Lovász Condition is false
[5] Swap $\mathbf{v}_k \leftrightarrow \mathbf{v}_{k-1}$ and set $k = k - 1$
[6] Else
[7] Set $k = k + 1$
[8] If $k = n$, then basis is LLL reduced
[9] END LOOP

The Basic LLL Algorithm

Operating Characteristics of LLL

- It is clear that if $k = n$ in Step 8, then the basis is LLL reduced.
- Step 7 helps us by incrementing k. But potentially there is problem because the Swapping Step (Step 5) decrements k.
- It is not hard to prove that Step 5 is executed only finitely many times and the number of executions is bounded by a polynomial in n. Thus LLL is a polynomial-time algorithm.
- The LLL algorithm is guaranteed to find a $\mathbf{v} \in L$ satisfying

$$0 < \|\mathbf{v}\| \leq 2^{(n-2)/2}\lambda_1(L).$$

- In practice, LLL generally does better than this. But also in practice, if n is large, then LLL will not find a vector just a few times longer than $\lambda_1(L)$.

Variants and Improvements to LLL

Many methods of improving LLL have been proposed over the years. Often they sacrifice provable polynomial time performance for improved operation on most lattices. One of the most important replaces the Swapping Step with a more complicated procedure.

Definition A **KZ Reduced Basis** is a basis that satisfies both the Size Condition and the following:

For all i, $\mathbf{v}_i^*$ is the shortest vector in the projection of L onto $\text{Span}(\mathbf{v}_1, \ldots, \mathbf{v}_i)$.

Block Reduction Algorithm (BKZ-LLL). (Schnorr) Instead of swapping $\mathbf{v}_k$ and $\mathbf{v}_{k-1}$ in Step 5 of LLL, instead take the lattice spanned by a block of vectors $\mathbf{v}_i, \mathbf{v}_{i+1}, \ldots, \mathbf{v}_{i+\beta-1}$ and replace them with a KZ Reduced Basis.

Operating Characteristics of BKZ-LLL

An advantage of BKZ-LLL is that the output improves as one increases the block size β. Indeed, taking $\beta = n$ gives a full KZ reduced basis for L, so it solves SVP.
Of course, the improved output comes at a cost of increased running time.
For a moderately large block size β, one can prove that BKZ-LLL finds a nonzero vector $\mathbf{v} \in L$ satisfying

$$\|\mathbf{v}\| \leq \left(\frac{\beta}{\pi e}\right)^{\frac{n-1}{\beta-1}} \lambda_1(L).$$

Unfortunately, the running time of standard LLL is increased by a factor of (at least) C^β for some constant C.

Experimentally one finds this borne out: For a fixed (small) constant c, the time for LLL-BKZ to find a $\mathbf{v} \in L$ satisfying

$$\|\mathbf{v}\| \leq n^c \lambda_1(L) \quad \text{is exponential in } n.$$

Knapsack Cryptosystems and Lattice Cryptanalysis

The Knapsack (Subset Sum) Problem

Let

$$\mathbf{a} = (a_1, a_2, \ldots, a_n)$$

be a list of positive integers.

Knapsack (Subset Sum) Problem

Given a target integer t, determine if there are values $x_1, x_2, \ldots, x_n \in \{0, 1\}$ satisfying

$$x_1 a_1 + x_2 a_2 + \cdots + x_n a_n = t.$$

If this decision problem can be solved efficiently, then we can actually find $x_1, \ldots, x_n$. For example, to find a value for x_1, it suffices to determine if either

$$x_2 a_2 + \cdots + x_n a_n = t \qquad \text{or}$$
$$x_2 a_2 + \cdots + x_n a_n = t - a_1$$

has a solution.

How Hard is the General Knapsack Problem?

The general Knapsack Problem is an NP-complete problem, so it is (presumably) very hard.

The trivial solution method is to try all 2^n possible values for $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$.

A better method is to sort the following two sets and look for a collision:

$$\left\{ \sum_{j \le n/2} x_j a_j : x_j = 0 \text{ or } 1 \right\}.$$

$$\left\{ t - \sum_{j > n/2} x_j a_j : x_j = 0 \text{ or } 1 \right\}.$$

This takes $O(n2^{n/2})$ operations.

There is still no algorithm known that solves all Knapsack Problems in fewer than $O(2^{n/2-\epsilon})$ operations!

Building a Cryptosystem from a Knapsack Problem

There is a natural way to try to build a cryptosystem based on a hard knapsack problem.

Bob's Public Key	$\mathbf{a} = (a_1, a_2, \ldots, a_n)$
Alice's Plaintext	$\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$
Alice's Ciphertext	$t = x_1 a_1 + \cdots + x_n a_n$

The problem with this approach is that in order to decipher the message, Bob needs to solve the knapsack problem!

So Bob needs some sort of trapdoor.

Building a Cryptosystem from a Knapsack Problem

Some knapsack problems are very easy to solve.

Suppose the weights $a_1, \ldots, a_n$ are **superincreasing**,

$$a_j > a_1 + a_2 + \cdots + a_{j-1} \quad \text{for each } 1 < j \leq n.$$

Then we can easily find x_n, since

$$x_n = 1 \quad \text{if and only if} \quad t > a_1 + a_2 + \cdots + a_{n-1}.$$

Having determined x_n, we are reduced to the lower dimensional knapsack problem

$$x_1 a_1 + \cdots + x_{n-1} a_{n-1} = t - x_n a_n,$$

so we can recover $x_{n-1}, \ldots, x_1$ recursively.

Unfortunately, since $a_1, \ldots, a_n$ are public knowledge, an attacker can deciper the message as easily as Bob.

Building a Cryptosystem from a Knapsack Problem

The solution proposed by Merkle and Hellman in 1978 was to conceal Bob's private superincreasing set

$$\mathbf{a} = (a_1, a_2, \ldots, a_n)$$

by some sort of invertible transformation.

To illustrate the general method, I will describe Merkle and Hellman's original single-transformation system and show how it can be viewed as a lattice problem and (often) solved using lattice reduction.

Merkle and Hellman and others subsequently proposed more complicated knapsack-based cryptosystems, but as far as I am aware, all practical systems have been broken using lattice reduction methods.

The Merkle-Hellman Knapsack Cryptosystem

Bob's Private Key: Superincreasing $b_1, \ldots, b_n$ with $b_1 \approx 2^n$ and $b_n \approx 2^{2n}$, and $M, W \in \mathbb{Z}$ with $M > b_1 + \cdots + b_n$ and $\gcd(M, W) = 1$, and a permutation π of the integers $\{1, \ldots, n\}$.

Bob's Public Key: Bob's public key is $\{a_1, \ldots, a_n\}$ with $a_j \equiv W b_{\pi(j)} \pmod{M}$.

Alice's Plaintext: $\mathbf{x} = (x_1, \ldots, x_n) \in \{0, 1\}^n$.

Alice's Ciphertext: $t = x_1 a_1 + \cdots + x_n a_n$.

Decryption: Bob computes

$$c \equiv W^{-1} t \equiv \sum_{j=1}^{n} x_{\pi^{-1}(j)} b_j \pmod{M}.$$

The modulus M is large, so c exactly equals the sum. Also $b_1, \ldots, b_n$ is superincreasing, so Bob can easily solve this knapsack problem and recover the plaintext $\mathbf{x}$.

Converting a Knapsack Problem to a Lattice Problem

Consider a knapsack problem to be solved:

$$t = x_1 a_1 + x_2 a_2 + \cdots + x_n a_n \qquad (*)$$

Define a lattice $L_{\mathbf{a}}$ using the rows of the matrix

$$L_{\mathbf{a}} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & a_1 \\ 0 & 1 & 0 & \cdots & 0 & a_2 \\ 0 & 0 & 1 & \cdots & 0 & a_3 \\ \vdots & & & \ddots & \vdots & \\ 0 & 0 & 0 & \cdots & 1 & a_n \\ 0 & 0 & 0 & \cdots & 0 & -t \end{pmatrix}$$

If $\mathbf{x} = (x_1, \ldots, x_n) \in \{0,1\}^n$ solves $(*)$, then

$$\mathbf{v} = (x_1, \ldots, x_n, 0) \in L_{\mathbf{a}}.$$

Note that $\mathbf{v}$ is a short vector. If it is the shortest vector in $L_{\mathbf{a}}$, then LLL or one of its variants may be able to find $\mathbf{v}$.

Other Applications of Lattices to Cryptanalysis

There are many other applications of lattice reduction to cryptanalysis. For example, suppose that p and q are unknown large primes with $p \approx q$ and that $n = pq$ is given. Suppose further than somehow the top-order bits of p have been leaked. Then the attacker knows numbers p_0 and q_0 so that

$$x = p - p_0 \qquad \text{and} \qquad y = q - q_0 \qquad \text{are "small".}$$

If $x < n^{1/4}$ and $y < n^{1/4}$, then Don Coppersmith showed how to set up a lattice problem whose solution would reveal x and y.

Another example is the use of lattice reduction to break RSA when the decryption exponent is small, or when the encryption exponent is small and similar messages are transmitted. (But no general method is known for small encryption exponenets.)

Lattice-Based Cryptography

Why Attempt To Use Lattices To Build Cryptosystems?

The reason that the Merkle-Hellman and other knapsack cryptosystems attracted attention is because they are much faster than RSA, often by a factor of 10 to 100. For example, if N and d are n bit numbers, it takes approximately

$$n^3 \text{ steps} \quad \text{to compute} \quad a^d \bmod N.$$

But knapsack encrypt/decrypt take only about n^2 steps.

On the other hand, it is sadly also true that slow secure cryptosystems do have some "small" advantages over fast insecure cryptosystems!

However, the speed advantages available from lattice operations combined with the fact that SVP and CVP are well-studied hard problems make it worth looking for other constructions whose security depends more directly on SVP and CVP.

The Ajtai-Dwork Lattice Cryptosystem

- Ajtai and Dwork (1995) described a lattice-based public key cryptosystem whose security relies on the difficulty of solving CVP in certain class of lattices $\mathcal{L}_{\mathrm{AD}}$.
- They proved that breaking their system in the average case (i.e., for a randomly chosen lattice of dimension m in $\mathcal{L}_{\mathrm{AD}}$) is as difficult as solving SVP for all lattices of dimension n (for a certain n that depends on m).
- This average case-worst case equivalence is a theoretical cryptographic milestone, but unfortunately the Ajtai-Dwork cryptosystem is impractical.
- Inspired by the work of Ajtai and Dwork, a more practical lattice-based cryptosystem was proposed in 1996 by Goldreich, Goldwasser, and Halevi.

The GGH Public Key Cryptosystem

Key Creation: Choose a lattice L and

Private Key $= \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ a good (short) basis,

Public Key $= \{\mathbf{w}_1, \ldots, \mathbf{w}_n\}$ a bad (long) basis.

Encryption: The plaintext $\mathbf{m}$ is a binary vector. Also choose a small random "perturbation" vector $\mathbf{r}$. The ciphertext is $\mathbf{e} = m_1\mathbf{w}_1 + m_2\mathbf{w}_2 + \cdots + m_n\mathbf{w}_n + \mathbf{r}$.

Note that the ciphertext vector $\mathbf{e}$ is not in the lattice L.

Decryption: Find a vector $\mathbf{u}$ in L that is closest to $\mathbf{e}$. If $\mathbf{r}$ is small enough, then $\mathbf{u} = m_1\mathbf{w}_1 + \cdots + m_n\mathbf{w}_n$, so solving CVP for $\mathbf{e}$ in L will recover $\mathbf{m}$. The private good basis can be used to find $\mathbf{u}$. First write

$\mathbf{e} = \mu_1\mathbf{v}_1 + \cdots + \mu_n\mathbf{v}_n$ using **real** $\mu_1, \ldots, \mu_n \in \mathbb{R}$.

Then round $\mu_1, \ldots, \mu_n$ to the nearest integer:

$\lfloor \mu_1 \rceil \mathbf{v}_1 + \cdots + \lfloor \mu_n \rceil \mathbf{v}_n$ will equal $\mathbf{u}$.

GGH versus LLL: A Lesson in Practicality

The security of GGH rests on the difficulty of solving CVP using a highly nonorthogonal basis.

The LLL lattice reduction algorithm finds a **moderately** orthogonal basis in polynomial time.

In practice, if $n = \dim(L) < 100$, then LLL easily finds a good enough basis to break GGH. Even for $n < 200$, variants of LLL give a practical way to break GGH.

The public key for GGH is a basis for L, so

Size of GGH Public Key $= O(n^2)$ bits.

GGH is currently secure for (say) $n = 500$, but 2 megabit keys are impractical!

The NTRU Public Key Cryptosystem solves this problem by using a type of lattice whose bases can be described using only $\frac{1}{2}n\log_2(n)$ bits.

NTRUEncrypt: The NTRU Public Key Cryptosystem

The Ring of Convolution Polynomials

Leaving lattices for the moment, we start with the ring of polynomials $R = \mathbb{Z}[X]/(X^N - 1)$.

These are polynomials with integer coefficients

$$a(X) = a_0 + a_1 X + a_2 X^2 + \cdots + a_{N-1} X^{N-1}$$

that are multiplied using the **convolution multiplication** rule $X^N = 1$. Thus the k^{th} coefficient of $c(X) = a(X)b(X)$ is

$$c_k = a_0 b_k + a_1 b_{k-1} + \cdots + a_{N-1} b_{k+1}.$$

Example with $N = 4$ (so the extra rule is $X^4 = 1$)

$$\begin{aligned}(X^3 + 2X - 1) * (3X^3 - X^2 + X + 2) \\ &= 3X^6 - X^5 + 7X^4 - 3X^3 + 3X^2 + 3X - 2 \\ &= 3X^2 - X + 7 - 3X^3 + 3X^2 + 3X - 2 \\ &= -3X^3 + 6X^2 + 2X + 5\end{aligned}$$

Modular Reduction of Polynomials

The coefficients of polynomials may be reduced modulo various integers (such as p or q) into various ranges.

Example: Reduce mod 16 so that $-3 \leq a_i < 13$:

$$19X^4 - 6X^3 + 7X^2 - 17 \equiv 3X^4 + 10X^3 + 7X^2 - 1 \pmod{16}$$

The **inverse of** $a(X)$ **modulo** q is a polynomial $a(X)^{-1} \in R$ satisfying

$$a(X)a(X)^{-1} \equiv 1 \pmod{q}.$$

The inverse (if it exists) is easily computed using the Euclidean algorithm and Hensel's lemma.

Example: $N = 5$ and $q = 16$. Working in the ring $\mathbb{Z}[X]/(X^5 - 1)$ mod 16, we find

$$\begin{aligned}(3X^4 + 10X^3 + 7X^2 - 1)^{-1} \\ \equiv 5X^4 + 3X^3 + 13X^2 + 8X + 14 \pmod{16}.\end{aligned}$$

How NTRUEncrypt Works

Key Creation: Fix N, p, q with N prime and with $\gcd(p, q) = 1$. Choose random polynomials $f, g \in R$ with small coefficients. Compute inverses

$$F_q \equiv f^{-1} \pmod{q} \qquad \text{and} \qquad F_p \equiv f^{-1} \pmod{p}$$

and set $h = g \cdot F_q \pmod{q}$.

Public Key $= h$ and Private Key $= f$ (and F_p)

Encryption: The plaintext m is a polynomial with mod p coefficients. Choose a random small polynomial r. The ciphertext is $e \equiv p \cdot r \cdot h + m \pmod{q}$.

Decryption: Compute

$$a \equiv e \cdot f \pmod{q},$$

choosing the coefficients of a to satisfy $A \le a_i < A + q$. Then $F_p \cdot a \bmod p$ is equal to the plaintext m.

Why NTRUEncrypt Works

The first decryption step gives the polynomial

Computation (mod q)	Reason
$a \equiv e \cdot f$	
$\equiv (p \cdot r \cdot h + m) \cdot f$	$e \equiv p \cdot r \cdot h + m$
$\equiv p \cdot r \cdot g + m \cdot f$	$h \cdot f \equiv g \cdot F_q \cdot f = g$

The coefficients of r, g, m, f are **small**, so the coefficients of

$$p \cdot r \cdot g + m \cdot f$$

will lie in an interval of length less than q. Choosing the appropriate interval, the polynomial

$$a \quad \textbf{equals} \quad p \cdot r \cdot g + m \cdot f \quad \textbf{exactly},$$

and not merely modulo q. Now multiply by F_p.

$$\begin{aligned} F_p \cdot a &= F_p \cdot (p \cdot r \cdot g + m \cdot f) \\ &\equiv F_p \cdot m \cdot f \pmod{p} \\ &\equiv m \pmod{p} \qquad \text{since } F_p \cdot f \equiv 1 \pmod{p}. \end{aligned}$$

Comparison of Operating Characteristics

Two reasons to consider lattice-based cryptosystems:

1. Potential speed and size advantages.
2. Backup in case other systems are broken.

The table compares operating characteristics of naive implementations of RSA, ECC, and NTRUEncrypt.

	RSA	ECC	NTRU
Encrypt/Decrypt	$O(n^3)$	$O(n^3)$	$O(n^2)$
Key size (bits)	n	n	$\approx \frac{1}{2} n \log_2 n$
Key Create	—	$O(n^3)$	$O(n^2)$
Typical n	1024	168	502

Among the many implementation tricks are:

1. Small RSA encryption exponent makes encrypt $O(n^2)$.
2. ECC precomputation/windowing speed encrypt/decrypt.
3. Karatsuba mult makes NTRU encrypt/decrypt $O(n \log n)$.

History of NTRUEncrypt

- NTRUEncrypt is in fact a lattice-based public key cryptosystem, because underlying the convolution polynomial ring

$$\mathbb{Z}[X]/(X^N - 1) \quad \text{modulo } q.$$

are

Convolution Modular Lattices.

The security of NTRU rests on the difficulty of solving CVP in these lattices.

- The original idea for NTRUEncrypt is due to Jeffrey Hoffstein in 1994.
- The system was developed by Jeffrey Hoffstein, Jill Pipher, and Joseph Silverman during 1994-96.
- NTRUEncrypt was first publicly presented at a Crypto rump sesssion in 1996.

Convolution Modular Lattices and NTRU Lattices

Polynomials and Vectors

It is often convenient to identify a polynomial $a(X) = a_0 + a_1 X + \cdots + a_{N-1} X^{N-1}$ with its vector of coefficients

$$\mathbf{a} = [a_0, \ldots, a_{N-1}].$$

Then $c(X) = a(X) \cdot b(X)$ with the rule $X^N = 1$ is

Vector Convolution Product $\mathbf{c} = \mathbf{a} * \mathbf{b}$.

The **norm** of a vector is $|\mathbf{a}| = \sqrt{a_0^2 + \cdots + a_{N-1}^2}$.

When one knows the average $\mu = (a_0 + \cdots + a_{N-1})/N$, the **Centered Norm** is often more useful:

$$\|\mathbf{a}\| = \sqrt{(a_0 - \mu)^2 + \cdots + (a_{N-1} - \mu)^2}.$$

Minimizing $\|\mathbf{a}\|$ is the same as solving CVP for $[\mu, \ldots, \mu]$.

Exercise: For "most" $\mathbf{a}$ and $\mathbf{b}$, $\|\mathbf{a} * \mathbf{b}\| \approx \|\mathbf{a}\| \cdot \|\mathbf{b}\|$.

Convolution Modular Lattices

The **Convolution Modular Lattice** $L_{\mathbf{h}}$ associated to the vector $\mathbf{h}$ and modulus q is the $2N$ dimensional lattice with basis given by the rows of the matrix:

$$L_{\mathbf{h}} = \text{RowSpan}\left(\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N-1} \\ 0 & 1 & \cdots & 0 & h_{N-1} & h_0 & \cdots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & h_2 & \cdots & h_0 \\ \hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array}\right)$$

Another way to describe $L_{\mathbf{h}}$ is the set of vectors

$$L_{\mathbf{h}} = \left\{(\mathbf{a}, \mathbf{b}) \in \mathbb{Z}^{2N} : \mathbf{a} * \mathbf{h} \equiv \mathbf{b} \pmod{q}\right\}.$$

Small Vectors in NTRU Convolution Modular Lattices

In an **NTRU Convolution Modular Lattice**,

$$f(X) \cdot h(X) \equiv g(X) \pmod{q} \quad \text{with "small" } f \text{ and } g.$$

This convolution relation implies that the NTRU lattice $L_{\mathbf{h}}$ contains the short vector

$$[\mathbf{f}, \mathbf{g}] = [f_0, f_1, \ldots, f_{N-1}, g_0, g_1, \ldots, g_{N-1}].$$

To see that $[\mathbf{f}, \mathbf{g}]$ is in $L_{\mathbf{h}}$, let

$$u(X) = \frac{-f(X) \cdot h(X) + g(X)}{q} \in \mathbb{Z}[X].$$

Then

$$[f_0, \ldots, f_{N-1}, u_0, \ldots, u_{N-1}] \begin{pmatrix} 1 & \cdots & 0 & h_0 & \cdots & h_{N-1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & h_1 & \cdots & h_0 \\ 0 & \cdots & 0 & q & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & q \end{pmatrix} = [f_0, \ldots, f_{N-1}, g_0, \ldots, g_{N-1}].$$

Convolution Modular Lattices as R-Modules

It is enlightening to describe $L_{\mathbf{h}}$ as a 2-dimensional module over the convolution polynomial ring

$$R = \mathbb{Z}[X]/(X^N - 1).$$

Then $L_{\mathbf{h}}$ can be described as the set

$$L_{\mathbf{h}} = \{[u, v] \in R^2 : u \cdot h \equiv v \pmod{q}\}.$$

The lattice $L_{\mathbf{h}}$ contains the short vector $[f, g]$ and the long vectors $[1, h]$ and $[0, q]$.

$$L_{\mathbf{h}} = \text{RowSpan} \begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix} = \text{RowSpan} \begin{pmatrix} f & g \\ * & * \end{pmatrix}$$

Long (Bad) Public Basis

Short (Good) Private Basis

The CVP Problem Underlying NTRU Keys

The vector $[f, g]$ is almost certainly the shortest vector in $L_{\mathbf{h}}$, so it can be found by solving SVP in $L_{\mathbf{h}}$.

If (say) f and g are binary with d ones and $N - d$ zeros, then

$$\big|[f, g]\big| = \sqrt{2d}.$$

However, the centered norm $\big\|[f, g]\big\|$, which is the distance from $[f, g]$ to $\left[\frac{d}{N}, \frac{d}{N}, \ldots, \frac{d}{N}\right]$, is smaller:

$$\big\|[f, g]\big\| = \sqrt{2d}\sqrt{1 - \frac{d}{N}} \qquad (*)$$

Thus it is easier to find $[f, g]$ by solving CVP in $L_{\mathbf{h}}$.

When N is large and the target distance $(*)$ not too small, the (extrapolated) running time for LLL to find the private key vector $[f, g]$ is very large.

NTRU Decryption as a CVP Problem

Recall that the ciphertext $e(X)$ has the form

$$e(X) = p \cdot r(X) \cdot h(X) + m(X) \pmod{q}.$$

We can rewrite this relation in vector form as

$$\begin{aligned}[0, e] &= [0, p \cdot r \cdot h + m \pmod{q}] \\ &\equiv [r, r \cdot (p \cdot h) \pmod{q}] + [-r, m].\end{aligned}$$

The vector $[r, r \cdot (p \cdot h) \pmod{q}]$ is in the convolution modular lattice $L_{p\mathbf{h}}$ obtained by using $p \cdot h(X)$ in place of $h(X)$. Further, the vector $[-r, m]$ is quite short.

Conclusion. For appropriate parameters, recovery of the plaintext m from the ciphertext e is equivalent to finding the vector in $L_{\mathbf{h}}$ that is closest to the vector $[0, e]$.

The difficulty of solving this CVP can be estimated experimentally.

The NTRU Lattice and Lattice Reduction

The most effective method known for finding short or close vectors in an NTRU lattice $L_{\mathbf{h}}$ is LLL and its variants.

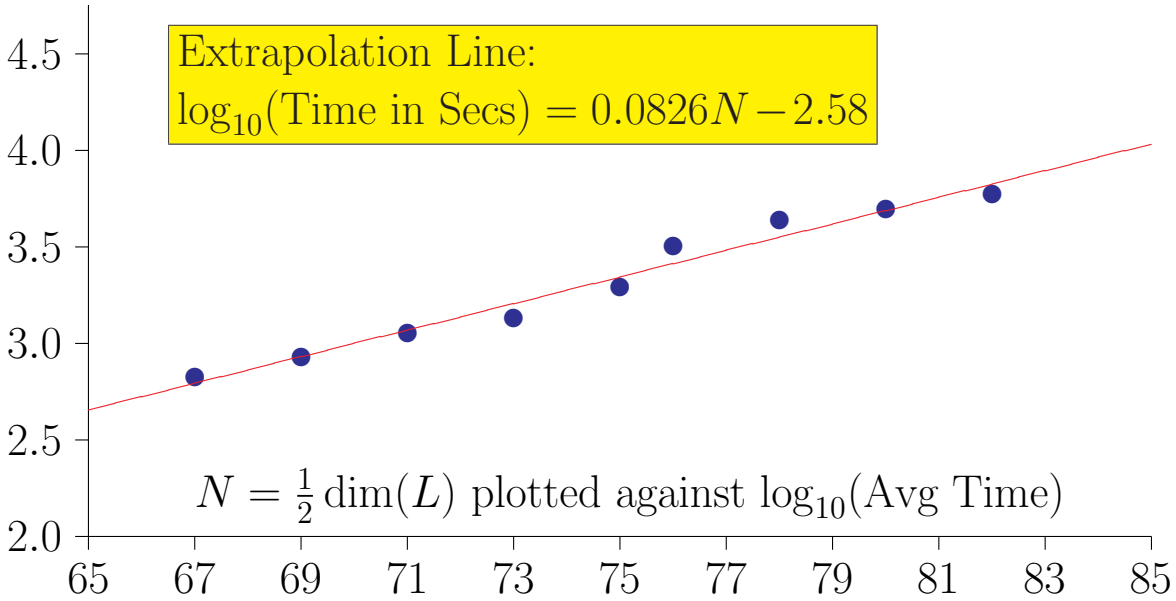
In practice, LLL tends to perform better than its provable upper bounds, so in order to assess the security of NTRUEncrypt, one performs experiments on lower dimensional lattices and does an extrapolation.

Here are some sample parameter sizes with their experimentally derived equivalent RSA security level.

	Public Key	Private Key	Security Level
NTRU 251	1757 bits	384 bits	RSA 1024 bit
NTRU 503	4024 bits	1000 bits	RSA 4096 bit

The next slide illustrates the results of one such experiment.

Running BKZ-LLL on NTRU Lattices



q	N	d	Avg(T)
34	67	20	975.5
35	69	20	1305.7
36	71	20	1846.9
37	73	21	2278.6
38	75	22	3532.8
39	76	23	6352.3
40	78	24	9251.1
41	80	24	10924.9
42	82	24	13407.1

LLL Running Time for NTRU Lattices:

- Time in seconds on a 400 MHz Pentium
- 10 trials for each value of N

Extrapolated Running Time: $N = 251$
Time $\approx 10^{18.15}$ Secs $\approx 10^{10.65}$ Years

Random Lattices and the Gaussian Heuristic

The Gaussian Heuristic

If $L \subset \mathbb{R}^n$ is a "random" lattice, how long would we expect its shortest vector to be?

And if $\mathbf{t} \in \mathbb{R}^n$ is a "random" target point, how far would we expect the closest lattice point to be to $\mathbf{t}$?

The **Gaussian Heuristic** answers these questions, but first...

we start with a different question.

If R is large, then how many copies of a fundamental domain $\mathcal{F}$ of L would we expect to fit inside an n-dimensional ball $\mathcal{B}_R$ of radius R?

Answer : $$\left(\begin{array}{c}\text{Number of copies}\\ \text{of } \mathcal{F} \text{ in } \mathcal{B}_R\end{array}\right) \approx \frac{\text{Vol}(\mathcal{B}_R)}{\text{Disc}(L)}.$$

Conclusion: If we choose R so that $\text{Vol}(\mathcal{B}_R) \approx \text{Disc}(L)$, then a ball of radius R centered at $\mathbf{t}$ is likely to contain a point of L (other than $\mathbf{t}$ itself).

The Gaussian Heuristic (continued)

Recall that if n is reasonably large, then the volume of an n-dimensional ball $\mathcal{B}_R$ of radius R is

$$\mathrm{Vol}(\mathcal{B}_R) \approx \left(\frac{2\pi e}{n}\right)^{n/2} R^n.$$

Solving $\mathrm{Vol}(\mathcal{B}_R) \approx \mathrm{Disc}(L)$ for R yields:

The Gaussian Heuristic. The shortest nonzero vector in a "random" lattice $L \subset \mathbb{R}^n$ has length approximately

$$\lambda_1(L) = \min_{\mathbf{v}\in L,\, \mathbf{v}\neq \mathbf{0}} \|\mathbf{v}\| \approx \sqrt{\frac{n}{2\pi e}}\, \mathrm{Disc}(L)^{1/n}.$$

Similarly, a "random" target vector $\mathbf{t} \in \mathbb{R}^n$ satisfies

$$\min_{\mathbf{v}\in L} \|\mathbf{v} - \mathbf{t}\| \approx \sqrt{\frac{n}{2\pi e}}\, \mathrm{Disc}(L)^{1/n}.$$

The Gaussian Heuristic and NTRU Lattices

The NTRU lattice $L_{\mathbf{h}}$ has dimension $n = 2N$ and its basis is an upper diagonal matrix whose diagonal is half 1's and half q's. Hence $\mathrm{Disc}(L_{\mathbf{h}}) = q^N$, so the Gaussian heuristic suggests that

$$\lambda_1(L_{\mathbf{h}}) \approx \sqrt{\frac{2N}{2\pi e}}\,(q^N)^{1/2N} = \sqrt{\frac{qN}{\pi e}}.$$

However, by construction the NTRU lattice contains a short vector $[\mathbf{f}, \mathbf{g}]$ of length $\sqrt{2d}$. Typically $d \approx \frac{1}{3}N$ and $q \approx \frac{1}{2}N$, so in a typical NTRU lattice,

$$\frac{\text{Gaussian Heuristic}}{\text{Actual Shortest Vector}} \approx \frac{\sqrt{qN/\pi e}}{\sqrt{2d}} \approx \frac{1}{5}\sqrt{\dim(L_{\mathbf{h}})}.$$

Conclusion. The private key vectors in an NTRU lattice are $O\big(\sqrt{\dim}\big)$ shorter than the other vectors. In particular, solving SVP (or CVP) breaks NTRU.

The Gaussian Heuristic and Knapsack Lattices

The lattice L used to analyze knapsack cryptosystems has dimension $n+1$ and its basis is an upper triangular matrix with 1's on the diagonal except for one entry

$$t = x_1 a_1 + \cdots + x_n a_n.$$

The $x_i \in \{0, 1\}$ are small, but the a_i satisfy $a_i \approx 2^{2n}$. Thus $\mathrm{Disc}(L) = t \approx \frac{1}{2} n 2^{2n}$. But L contains the vector $\mathbf{v} = (x_1, \ldots, x_n, 0)$ of length $\|\mathbf{v}\| \approx \sqrt{n/2}$.

Hence for large n,

$$\frac{\text{Gaussian Heuristic}}{\text{Actual Shortest Vector}} \approx \frac{4}{\pi e} \approx 1.37.$$

Thus the shortest vector in L is very likely to be the plaintext vector $(x_1, \ldots, x_n, 0)$, so solving SVP breaks the knapsack cryptosystem.

Some Further Remarks

Balancing the NTRU Lattice

Recall that an NTRUEncrypt private key consists of two small polynomials f and g, and that the small target vector in the lattice $L_{\mathbf{h}}$ is the vector $[f, g]$.

If f and g are of different lengths, then Coppersmith and Shamir pointed out that the lattice problem becomes easier if one balances the lattice by taking

$$L_{\mathbf{h}}^{\text{bal}} = \text{RowSpan}\left(\begin{array}{cccc|cccc} \lambda & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N-1} \\ 0 & \lambda & \cdots & 0 & h_{N-1} & h_0 & \cdots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda & h_1 & h_2 & \cdots & h_0 \\ \hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array}\right)$$

so that the new target vector $(\lambda f, g)$ has $\|\lambda f\| = \|g\|$.

Similarly, if r and m have different lengths, one can balance the associated CVP problem to find $[r, m]$.

Further Reading

Further Reading

- Ajtai, M., Dwork, C., A public-key cryptosystem with worst-case/average-case equivalence. STOC '97 (El Paso, TX), 284-293 ACM, New York, 1999. [A fundamental theoretical advance in lattice-based cryptography.]
- Buchmann, J., Ludwig, C. , Practical Lattice Basis Sampling Reduction, Cryptology ePrint Archive, Report 2005/072, `http://eprint.iacr.org/`. [An improved lattice reduction method using random sampling.]
- Cryptography and Lattices Conference (CaLC), Providence, RI, Lecture Notes in Comput. Sci. 2146, Springer, 2001. [A conference devoted to the uses of lattices in cryptography, with may interesting articles.]
- Cassels, J. W. S. An introduction to the geometry of numbers. Classics in Mathematics. Springer-Verlag, Berlin, 1997. [An excellent introduction.]
- Coppersmith, D., Shamir, A., Lattice attacks on NTRU. Advances in cryptology-EUROCRYPT '97, 52-61, Lect. Notes in Comput. Sci., 1233, Springer, Berlin, 1997. [The first paper containing an analysis of NTRU.]
- Goldreich, O., Goldwasser, S., Halevi, S., Public-key cryptosystems from lattice reduction problems. Advances in cryptology-CRYPTO '97, 112-131, Lecture Notes in Comput. Sci., 1294, Springer, Berlin, 1997. [Lattice-based public key cryptosystem and digital signature scheme.]
- Gruber, P. M.; Lekkerkerker, C. G. Geometry of numbers. North-Holland Mathematical Library, 37. North-Holland Publishing Co., Amsterdam, 1987. [The "bible" of the subject, comprehensive and dense.]

Further Reading

- Hoffstein, J, Pipher, J, Silverman, J.H., NTRU: A ring-based public key cryptosystem. Algorithmic number theory (Portland, OR, 1998), 267-288, Lecture Notes in Comput. Sci., 1423, Springer, Berlin, 1998. [The original article describing the NTRU lattice-based cryptosystem.]
- Lenstra, A., Lenstra, H., Lovasz, L., Factoring polynomials with rational coefficients, Mathematische Ann. 261 (1982), 513-534. [The famous LLL algorithm.]
- Nguyen, P., Stern, J., The two faces of lattices in cryptography. Cryptography and lattices-CaLC 2001, 146-180, Lecture Notes in Comput. Sci., 2146, Springer, Berlin, 2001. [Survey of how lattices are used both to create and to break cryptosystems.]
- NTRU tutorials and technical notes `<www.ntru.com>`. [NTRU Cryptosystems markets NTRU lattice-based cryptographic products.]
- Odlyzko, A. The rise and fall of knapsack cryptosystems. Cryptology and computational number theory (Boulder, CO, 1989), 75-88, Proc. Sympos. Appl. Math., 42, Amer. Math. Soc., 1990. [The title says it all!]
- Schnorr, C., A hierarchy of polynomial time lattice basis reduction algorithms. Theoretical Computer Science 53 (1987), 201-224. [One of many articles by Schnorr and colleagues giving improvements to the LLL algroithm.]
- Siegel, C.L., Lectures on the geometry of numbers. Springer-Verlag, Berlin, 1989. [Another excellent introduction to the subject.]