Fun with Lattices!
(a.k.a Fundamentals of Lattice-Based Cryptography)

Jose Cardona

OQuant

LambdaConf 2019

Why lattices?

Lattices are incredible mathematical objects with applications outside of the ones we'll view today!

- In Number theory, lattices have been used to test various conjectures (See: disproof of Merten's conjecture)
- In Complexity Theory, lattices have a special place in that for certain problems, finding a solution in a "random lattice" is as hard as a solution in worst case complexity! (Ajitai)
- In Error Correcting Codes, Lattices + The Sphere packing problem (Which also uses lattices) are used for ECC construction.

Today, we tackle the application that interests me the most: Lattices used to construct Cryptosystems!

Table Of Contents

Our Goal Today

1. **Vector Spaces**
2. Lattices
3. GGH
4. Convolution Polynomials and NTRU
5. (BONUS) Lattice-Reductions (LLL)
6. (BONUS) Short Integer Solutions
7. (BONUS) LWE

Vector spaces

A vector space V over some field F is an abelian group with a *scalar product* $\forall \alpha \in F, \forall v \in V : \alpha v$ defined, that satisfies the following axioms:

- $\forall \alpha, \beta \in F, \forall v \in V : \alpha(\beta v) = (\alpha\beta)v$
- $\forall \alpha, \beta \in F, \forall v \in V : (\alpha + \beta)v = \alpha v + \beta v$
- $\forall \alpha, \beta \in F, \forall v \in V : \alpha(u + v) = \alpha u + \alpha v$
- $\forall v \in V : 1_F v = v$

Such that

$$\forall \alpha, \beta \in F, \forall u, v \in V : \alpha u + \beta v \in V$$

Vector spaces are more general than how we use them (and can be generalized even more by modules), but in this presentation, the group we will be concerned with will be vectors in $\mathbb{R}^n$ for some $n \in \mathbb{N}$

Vector spaces

A linear combination of some set of vectors $v_1, v_2, .., v_k \in V$ is any vector of the form

$$u = \alpha_1 v_1 + \alpha_2 v_2 + .. + \alpha_k v_k \text{ with } \alpha_1, .., \alpha_k \in F$$

The set of all linear combinations of the above is called the **span** of $v_1, ..., v_k$
A set of vectors is linearly independent if for the above, $\alpha_1 v_1 + \alpha_2 v_2 + .. + \alpha_k v_k = 0$ implies $a_1 = a_2 = ... = a_k = 0$

Vector spaces

A **basis** for a vector space V is a set of linearly independent vectors $v_1, ..., v_n \in V$ that span V, such a way that every $w \in V$ is written as a *unique* linear combination (in other words, unique combination of coefficients). Such a basis is said to have **dimension** (or dim) n.
Let $V \subset R^m$ be a vector space. Then:

- There exists a basis for V
- Any two basis for V have the same number of elements.
- The dimension of the basis cannot be larger than the dimension of the coordinate space (Meaning for basis of dimension n and R^m, $n <= m$).

Vector spaces

Defn: The **dot product** of two vectors $v = (v_1, v_2, .., v_n)$ and $u = (u_1, u_2, ..., u_n)$ is defined by the quantity:

$$u \cdot v = \sum_{i=1}^{n} v_i u_i$$

Defn: The length or Euclidean norm (or *length*) of some vector $v = (v_1, v_2, .., v_n)$ is the quantity

$$\|v\| = \sqrt{v_1^2 + v_2^2 + .. + v_n^2}$$

The dot product and the Euclidean norm share a relationship via:

$$v \cdot v = \|v\|^2$$

Vector spaces

Defn (Cauchy-Schwarz inequality): For some vectors v, $w \in V \subset R^n$,

$$|v \cdot w| \leq \|v\|\|w\|$$

Defn A basis B is said to be **orthogonal** if for all $v_i, v_j \in B$ with $i \neq j$, $1 \leq i < j \leq n$ we have:

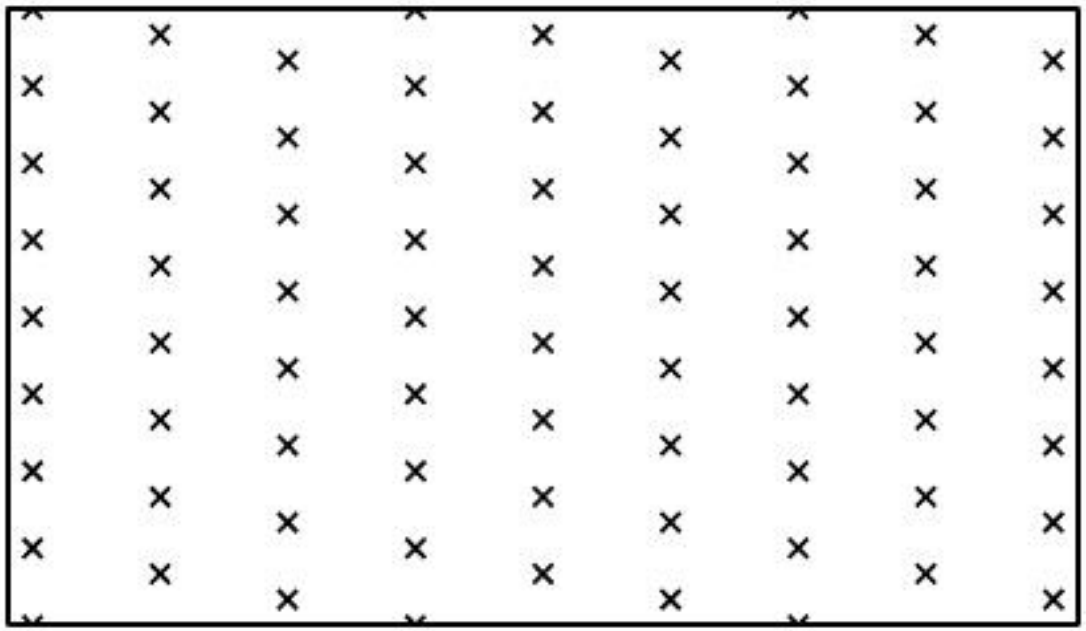
$$v_i \cdot v_j = 0$$

Said basis is *orthonormal* if for any $v_i \in B$, $\|v_i\| = 1$.

1. Vector Spaces
2. **Lattices**
3. GGH
4. Convolution Polynomials and NTRU
5. (BONUS) Lattice-Reductions (LLL)
6. (BONUS) Short Integer Solutions
7. (BONUS) LWE

Lattices

Lattices, geometrically, are a set of points in n-dimensional Euclidean space that exhibit a periodic structure, the easiest of which to envision is, for example, a grid-like structure in two-dimensional space.



Lattices

Let B be a set of linearly independent vectors $B = b_1, b_2, .., b_n$ of dimension n.
The lattice $\mathcal{L}(b_1, b_2, .., b_n)$ is the discrete linear combinator of vectors spanned by the basis with integer coefficients, or:

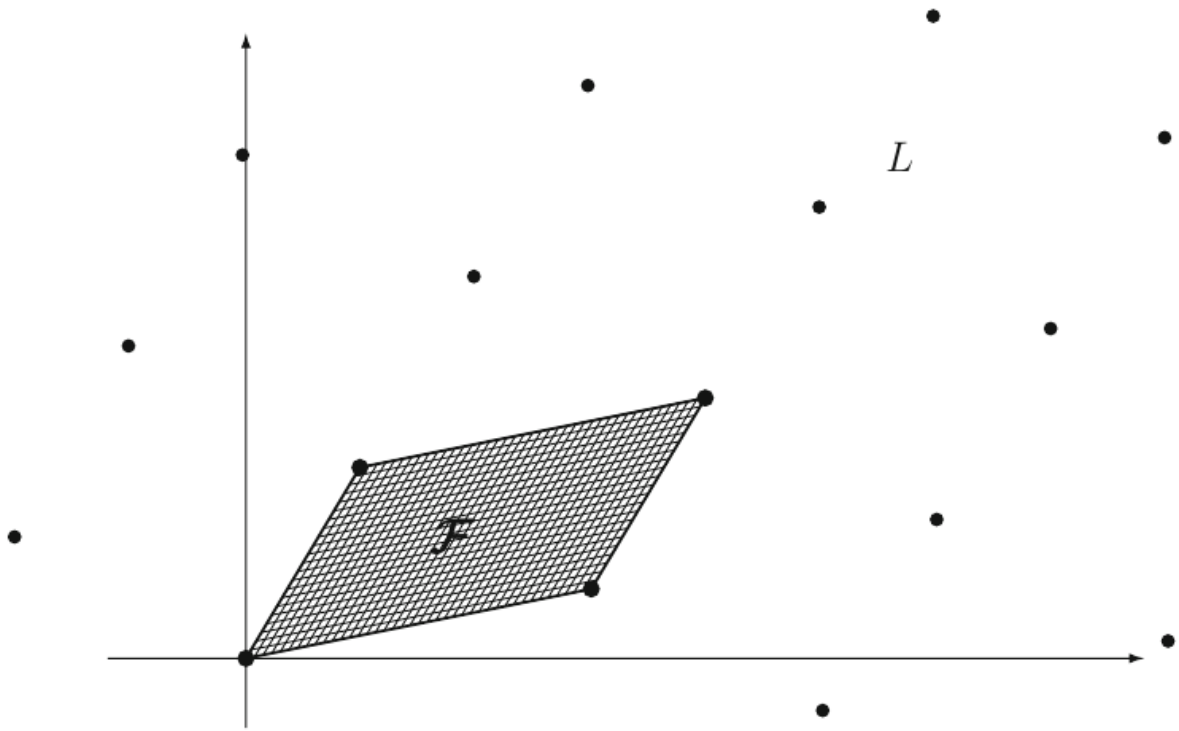
$$\mathcal{L}(b_1, b_2, ..., b_n) = \{ \sum_{i=1}^{n} a_i b_i | a_i \in \mathbb{Z} \}$$

By *discrete*, it means that there exists some $\epsilon > 0$ such that the distance between any two lattice vectors is at least ϵ.
Geometrically, this corresponds to drawing an n-dimensional ball centered around some vector coordinate in R^n.
It turns out that a subset of R^m is a lattice if and only if it is a discrete, additive subgroup (We will come back to the proof of this!).

Lattices

For any lattice of dimension n with basis $b_1, ..., b_n$, we define the *Fundamental Parallelepiped* $\mathcal{F}$ as the set

$$\mathcal{F}(b_1, ..., b_n) = \left\{ \sum x_i b_i \mid x_i \in R^n, 1 \leq i \leq n, 0 \leq x_i < 1 \right\}$$



Lattices

Fundamental Parallelepipeds are interesting to us since, even by naiive inspection, we can observe a few things:

- Fundamental Parallelepipeds give us an "area" to work with when we want to describe how big are the "cells" that make up our n-dimensional lattice.
- It almost looks like they "tile" our lattice.

Unrelated: Paralelepiped. I love that word.

Lattices

With these notions in hand, it turns out that the intuitions are actually correct! Fundamental Parallelepipeds in fact provide us with both a notion of tiling and "volume". We will start with the former.

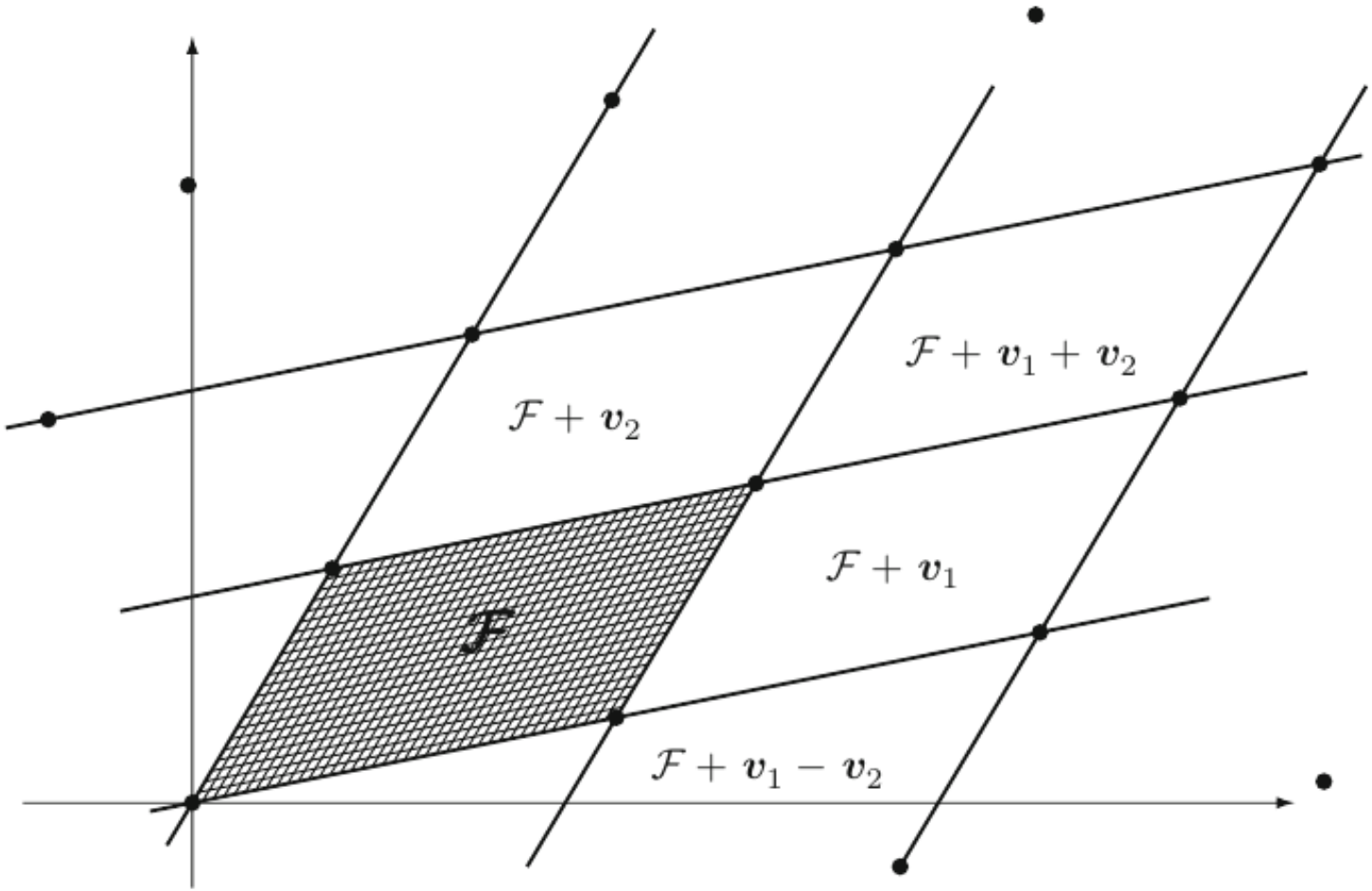
Proposition (L1): Let L is a full rank lattice in R^n, $\mathcal{F}$ a F.P. for L. Then we can write every vector in $w \in R^n$ as:

$$w = u + f, \text{ for some unique } u \in L, f \in \mathcal{F}$$

Equivalently, we can say for all $u \in L$, $u + F = \{u + f \,|\, f \in \mathcal{F}\}$ forms a partition of R^n.

Lattices

We can observe the idea from the previous slide in R^2 as such:



Lattices

Proof (L1): For our full rank lattice L, let the basis of L be $b_1, ..., b_n$. We know since it is a full rank lattice, the basis also spans R^n.
Thus, every vector in $\mathbb{R}^n$ is of the form $a_1 b_1 + ... + a_n b_n$ with $a_i \in \mathbb{R}$.

We can also consider writing every a_i in the form $a_i = \alpha_i + t_i$ where $\alpha_i \in \mathbb{Z}, t_i \in \mathbb{R}, 0 \leq t_i < 1$, which is an equivalent formulation.

WLOG for any $a_i b_i$ we can transform it as:

$$\begin{aligned} a_i b_i &= (\alpha_i + t_i) b_i \\ &= \alpha_i b_i + t_i b_i \end{aligned}$$

Where now $\alpha_i b_i$ is an element of L and $t_i b_i$ is an element of $\mathcal{F}$.

Proof (L1) (Cont): The only thing left now is to prove that each combination of unique representatives will yield a unique vector, which is quick.

WLOG pick one representative $(\alpha_i + t_i)b_i$ and let us assume there is a pair $\alpha_i' + t_i'$ such that $(\alpha_i + t_i)b_i = (\alpha_i' + t_i')b_i$.

Then we can algebraically cancel b_i and rearrange to yield

$$\alpha_i - \alpha_i' = t_i - t_i'$$

However, we know that α_i and α_i' are both integers, so their difference should yield an integer. Thus, the only way t_i and t_i' yield an integer is if $t_i = t_i'$, proving that our vector is unique.

Lattices

Another way to think about (Full rank) lattices is as a set generated by the linear transformation of an nxn matrix A composed of the basis vectors columnwise, and a vector x in $\mathbb{Z}^n$ such that:

$$B = \begin{bmatrix} | & & | \\ b_1 & .. & b_n \\ | & & | \end{bmatrix}$$

And the lattice $\mathcal{L}(B)$ is now defined as

$$\mathcal{L}(B) = \{Bx | x \in \mathbb{Z}^n\}$$

This definition becomes immediately useful when we consider: when do two linearly independent subsets of R^m generate the same lattice?

Lattices

As an example, let's use:

$$B_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, B_2 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$

It turns out that both bases generate the same lattice, but how?
Moreover, the matrix:

$$B_3 = \begin{bmatrix} 41 & 41 \\ 9 & 8 \end{bmatrix}$$

Also has column vectors spanning $\mathbb{R}^2$, but $\mathcal{L}(B_1) = \mathcal{L}(B_2) \neq \mathcal{L}(B_3)$.
Why?

Lattices

Defn A matrix U is unimodular if $det(U) = \pm 1$.
It turns out we relate any two bases $B_1, B_2 \in \mathbb{R}^{nxn}$, such that they generate the same Lattice, if an only if there exists a *unimodular* matrix U such that $B_1 = B_2 U$ (L2):

Proof (L1) ($\implies$):

Assume $\mathcal{L}(B_1) = \mathcal{L}(B_2)$. Then for each column vector b_{1i} of B_1, we have $b_{1i} \in \mathcal{L}(B_2)$.
This implies that there exists some matrix U such that $B_1 = B_2 U$. Apply the same argument backwards to each b_{2i} of B_2 and you get $B_2 = B_1 V$ for some matrix V.

Lattices

Thus we can, via simple algebraic substitution:

$$
\begin{aligned}
B_1 &= B_2 U \\
B_1 &= (B_1 V) U && \text{<substitution on previous slide.>} \\
B_1 &= B_1 (VU) && \text{<matrix multiplication commutativity >} \\
I &= VU && \text{<left cancellation >}
\end{aligned}
$$

(Note: Left cancellation works on B_1 due to having linearly independent column vectors, thus it has an inverse).
Thus VU are matrix inverses of each other, so we have $det(I) = det(VU) = 1$, which means $det(I) = det(V)det(U)$. However given both are integer matrices, the only way this results to 1 is if the determinant of each is ± 1.
We will do the converse together.

Lattices

Now that we've proven that two bases generate the same lattice if they have a unimodular linear transformation, there's a connection to be made with this.

From Linear algebra (this one is easy to forget), we know that the determinant of a matrix refers to the signed volume of the n-dimensional parallelepiped (for our purposes, ignore the sign). So we can simply refer to $Vol(F) = |det(B)|$ for some base B that generates a lattice.

It then connects the dots that the fundamental parallelepiped for some lattice $\mathcal{L}(B)$ should have the same volume no matter your choice of basis, meaning that a unimodular matrix is the only way you would be able to preserve volume while changing basis!

Lattices

In the interests of cryptography, with lattices, we want to somehow create constructions that under some "priviledged information" we can use to manipulate information.
With cryptographic constructions based on fields of discrete points, the diffie-hellman problem comes to mind..
With cryptographic constructions like RSA, we have the prime factoring problem.
For the foundations of lattice cryptography, there exists a similar few problems.

Lattices

The Shortest Vector Problem (SVP) deals with trying to minimize the euclidean norm $\|v\|$.

The Closest Vector Problem(CVP) deals with, for some vector $a \in \mathbb{R}^n$, find the vector b in the lattice L that minimizes the euclidean norm $\|a - b\|$.
From these two, arise many important variants, with two we will see:

Approximate SVP (apprSVP): Don't just minimize Euclidean norm, find a vector v that for some function $\phi(n)$ (with n being the dimension of the lattice), we minimize

$$\|v\| \leq \phi(n)\|v_{shortest}\|$$

Approximate CVP (apprCVP): same as above except approximating the closest vector.

Lattices

An important metric is the shortest nonzero vector, which we can denote as λ_1. We can also think about λ_1, as a closed ball of radius r, which gives a space that only contains one point of the lattice (or equivalently, the lattice points inside the ball span a space of dimension 1).

So for some Lattice L, we have:

$$\lambda_1(L) = inf\{r \mid dim(span(L \cap \overline{B}(0, r))) \geq 1\}$$

It turns out this concept is easy to generalize, and thus we have *successive minima*, defined by

$$\lambda_i(L) = inf\{r \mid dim(span(L \cap \overline{B}(0, r))) \geq i\}$$

Lattices

Successive minima might not be defined as a vector belonging to some lattice L, but due to the property of discreteness of lattices, it turns out that there's a correspondence between successive minima and progressively nth smallest vectors in $\mathbb{R}^n$.
Successive minima give us a lower bound on the length of the shortest vector in L.

Proposition: Let B be the basis of some lattice $\mathcal{L}(B)$ of rank n, and $\widetilde{B}$ it's Gram-Schmidt orthogonalization, then we have

$$\lambda_1 \geq min_{i \leq n} \|\tilde{b}_i\| > 0$$

for some $\tilde{b}_i$ as a column vector of $\widetilde{B}$

Proof: If we consider $x \in \mathbb{Z}^n$ to be an arbitrary non-zero integer, we can show $\|Bx\| \geq min\|b_i\|$. Pick the largest nonzero element of x, denoted as x_j for an integer $j \in [1, n]$. If we take $(Bx) \cdot \tilde{b}_j$, we have:

$$|(Bx) \cdot \tilde{b}_j| = |(\sum_{i=1}^{j} x_i b_i) \cdot \tilde{b}_j|$$

Lattices

But notice that for Gram-Schmidt orthogonalized matrices, not only are the column vectors are orthogonal basis, but forall $i < j$, $b_i \cdot \tilde{b}_j = 0$, so then we have

$$
\begin{aligned}
|(Bx) \cdot \tilde{b}_j| &= |(\sum_{i=1}^{j} x_i b_i) \cdot \tilde{b}_j| \\
&= |(\sum_{i=1}^{j} x_i b_i \cdot \tilde{b}_j)| && \text{(distrib over addition)} \\
&= |x_j (b_j \cdot \tilde{b}_j)| && \text{(orthogonal for all i ¡ j)} \\
&= |x_j||(\tilde{b}_j \cdot \tilde{b}_j)| && \text{equality of dot products} \\
&= |x_j| \|\tilde{b}_j\|^2
\end{aligned}
$$

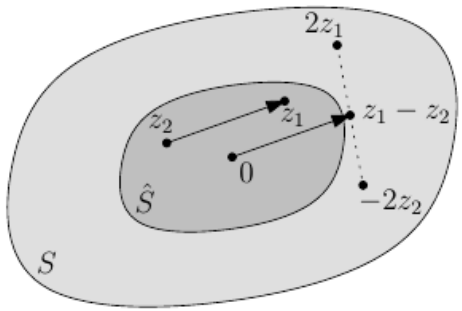
From Cauchy-Schwartz we have: $|(Bx) \cdot \tilde{b}_j| \leq \|Bx\|\|\tilde{b}_j\|$, so we can conclude that

$$\|Bx\|\|\tilde{b}_j\| \geq |x_j|\|b_j\|^2$$
$$\|Bx\| \geq |x_j|\|b_j\| \geq \|b_j\| \geq min\|b_i\|$$

Completing our proof. This proof is quite powerful, as it proves the discreteness property (Try it yourself!)

Lattices

To establish an upper bound on the shortest vector on our lattice, we need a way to relate "balls" (closed symmetric convex sets) to our lattice. Enter **Minkowski's convex body theorem** (We will not prove it): for some lattice L, for some centrally symmetric convex set, if $vol(S) > 2^n det(L)$, then S contains a nonzero lattice point.
In other words: any centrally symmetric convex set whose volume is 2^n bigger than the fundamental parallelepiped must contain a non-zero lattice point.



Lattices

The convex body theorem plus a lower bound on the volume of an $n-$ *dimensionalball* gives you an upper bound on the shortest vector.
This is given by **Minkowki's first theorem**: Every lattice L contains a nonzero vector v that satisfies

$$\|v\| \leq \sqrt{n}(det(L)^{\frac{1}{n}})$$

Which now gives us an interesting bound above and below for our shortest vector!
We can define also the **Hadamard Ratio** as:

$$\mathcal{H}(B) = \left(\frac{detL}{\|v1\|...\|v_n\|}\right)^{\frac{1}{n}}$$

Where $0 < \mathcal{H}(B) \leq 1$ where the closer that value is to 1, the more orthogonal the vector is.

GGH

With bounds on our problem, we can start trying to look at a way we can contruct a lattice-based cryptosystem.
To start, we can notice that solving the SVP and CVP is incredibly easy for for a lattice L with an orthogonal basis, since orthogonality degrades the euclidean norm of every single linear combination of the basis vectors to

$$\|v\| = \sum_{i \leq n} a_i^2 \|v_i\|^2$$

A similar thing happens when we have an orthogonal basis to try and solve the CVP.

The euclidean norm for some two vectors $w \in \mathbb{R}^n, v \in L$ is also expressed as a single sum, except with a difference of coefficients, for $a_i \in \mathbb{Z}$ and $b_i \in \mathbb{R}$:

$$\|w - v\| = \sum_{i \leq n} (a_i - b_i)^2 \|v_i\|^2$$

Since a_i are integers, the easiest way to minimize the coefficient is simply take the closest integer to each b_i.

GGH

It's tempting to attempt to use this same method on any arbitrary basis, but the basis of the lattice needs to be at least close to orthogonality for this to work.
Recall how we proved earlier that the element of a lattice L combined with it's fundamental parallelepiped form a partition of $\mathbb{R}^n$. This means that to solve the CVP, we can attempt to approximate it via the unique decomposition into its corresponding partition, then finding the closest vertex of the Parallelepiped to w. This is fairly easy if you hvae the vector that translates w into the lattice quadrant, and thus comes babai's algorithm:

GGH

Theorem 7.34 (Babai's Closest Vertex Algorithm). *Let $L \subset \mathbb{R}^n$ be a lattice with basis $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$, and let $\boldsymbol{w} \in \mathbb{R}^n$ be an arbitrary vector. If the vectors in the basis are sufficiently orthogonal to one another, then the following algorithm solves* CVP.

Write $\boldsymbol{w} = t_1\boldsymbol{v}_1 + t_2\boldsymbol{v}_2 + \cdots + t_n\boldsymbol{v}_n$ with $t_1, \ldots, t_n \in \mathbb{R}$.

Set $a_i = \lfloor t_i \rceil$ for $i = 1, 2, \ldots, n$.

Return the vector $\boldsymbol{v} = a_1\boldsymbol{v}_1 + a_2\boldsymbol{v}_2 + \cdots + a_n\boldsymbol{v}_n$.

GGH

With this, we can make our first cryptographic construction (Discovered by Goldreich, Goldwasser and Halevi):
Begin choosing a "Good" set of linearly independent vectors $v_1, ..., v_n \in \mathbb{Z}^n$ that are reasonably orthogonal (check with Hadamard). This set is alice's private key, and we can denote the (Column-wise) matrix that this set spans by V. Then, pick a unimodular matrix U such that you hvae $W = UV$ (U can be randomly chosen elementary matrices). The column vectors for W are her public key.

GGH

When Bob wants to send us a message, he chooses a plaintext m in the form of a vector (could be a binary vector), and small random perturbation vector r (think of a nonce, but vector-style!).
To Encrypt, bob computes:

$$e = Wm + r = \sum_{i-1}^{n} m_i w_i + r$$

Which is his ciphertext. For us to decrypt it, we use babai's algorithm combined with the "Good" basis we chose to find the smallest vector in L close to e. Using the good basis with a small r means we recover Wm, which we can multiply with W^{-1} to recover m.
Unfortunately, GGH was proven impractical via Lattice reduction algorithms, which we will get to.

NTRU

The next cryptosystem we will toy with when it comes to lattices requires a bit more algebra to understand.
Let us define the *ring of convolution polynomials* of rank n as the quotient ring

$$R = \frac{\mathbb{Z}[x]}{x^n - 1}$$

You can also have The ring of convolution polynomials over some modulus, by simply using:

$$R_q = \frac{\mathbb{Z}_q[x]}{x^n - 1}$$

NTRU

Without spending too much time on this, we only need a few properties:

- The product of two polynomials in a convolution ring has its coefficients defined, for two polynomials in the ring $a(x)$ and $b(x)$ as

$$c_k = \sum_{i+j \equiv k (mod n)} a_i b_{k-i}$$

  Example: $a(x) = 1 - 2x + x^2$ and $b(x) = 3 - 5x$ with R of rank 3
- Let $a(x) \in R_q$, the **center-lift** of $a(x)$ to R is a unique polynomial $a'(x)$ such that $a'(x) mod q = a(x)$, with coefficients $a_i \in (-\frac{q}{2}, \frac{q}{2}]$
- if q is prime, then $a(x)$ in R_q has a multiplicative inverse iff it is relatively prime to $x^n - 1$.

Onto NTRUEncrypt:

Fix the rank of your convolution polynomial ring to some prime n and pick two relatively prime moduli p and q so you have R, R_p and R_q, that also obey $gcd(n, q) = gcd(p, q) = 1$.

We can observe a polynomial in the different rings using center-lifts (or in R to R_p orR_q, by taking the modulus).

We also define a $T(d_1, d_2)$ as the function that categorizes the set of all ternary polynomials with d_1 coefficients set to 1, and d_2 to -1, and the rest 0.

NTRU

Alice picks some (n, p, q, d) with the constraints outlined above, and $q > (6d + 1)p$. She generates the private key consisting of two ternary polynomials $f(x) \in T(d + 1, d)$ and $g(x) \in T(d, d)$.

Alice computes the inverses of $f(x)$: $F_q(x) \in R_q, F_p(x) \in R_p$, then computes $h(x) = F_q(x)g(x)$. The pair $(f(x), F_p(x))$ is alice's private key. $h(x)$ is alice's public key.

Bob sends a message to alice in the form of the coefficients of some polynomial $m(x)$ that satisfy $-\frac{1}{2}p < m_i \leq \frac{1}{2}p$ (meaning it's the center lift of another polynomial in R_p). Bob chooses a random polynomial $r(x) \in T(d, d)$ and computes

$$e(x) = p(h(x)r(x)) + m(x) \pmod{q}$$

Which makes $e(x)$ bob's ciphertext, in R_q.

Alice decrypts $e(x)$ via the computations:

$$a(x) = f(x)e(x) \pmod{q}$$

$$b(x) = F_p(x)a(x) \pmod{p}$$

We can prove rather easily, algebraically, that $b(x)$ and $m(x)$ match up, what is left is to prove that the coefficient bounds on our congruences makes $b(x)$ and $m(x)$ have the same coefficients.

Alice decrypts $e(x)$ via the computations:

$$a(x) = f(x)e(x) \pmod{q}$$

$$b(x) = F_p(x)a(x) \pmod{p}$$

We can prove rather easily, algebraically, that $b(x)$ and $m(x)$ match up, what is left is to prove that the coefficient bounds on our congruences makes $b(x)$ and $m(x)$ have the same coefficients.

Public parameter creation	
A trusted party chooses public parameters (N, p, q, d) with N and p prime, $\gcd(p, q) = \gcd(N, q) = 1$, and $q > (6d + 1)p$.	
Alice	**Bob**
Key creation	
Choose private $\boldsymbol{f} \in \mathcal{T}(d + 1, d)$ that is invertible in R_q and R_p. Choose private $\boldsymbol{g} \in \mathcal{T}(d, d)$. Compute $\boldsymbol{F}_q$, the inverse of $\boldsymbol{f}$ in R_q. Compute $\boldsymbol{F}_p$, the inverse of $\boldsymbol{f}$ in R_p. Publish the public key $\boldsymbol{h} = \boldsymbol{F}_q \star \boldsymbol{g}$.	
Encryption	
	Choose plaintext $\boldsymbol{m} \in R_p$. Choose a random $\boldsymbol{r} \in \mathcal{T}(d, d)$. Use Alice's public key $\boldsymbol{h}$ to compute $\boldsymbol{e} \equiv p\boldsymbol{r} \star \boldsymbol{h} + \boldsymbol{m} \pmod q$. Send ciphertext $\boldsymbol{e}$ to Alice.
Decryption	
Compute $\boldsymbol{f} \star \boldsymbol{e} \equiv p\boldsymbol{g} \star \boldsymbol{r} + \boldsymbol{f} \star \boldsymbol{m} \pmod q$. Center-lift to $\boldsymbol{a} \in R$ and compute $\boldsymbol{m} \equiv \boldsymbol{F}_p \star \boldsymbol{a} \pmod p$.	

NTRU

Recall $h(x)$ (The public key) from our previous slide, and recall the polynomial coefficients look quite similar to a vector, with:

$$h(x) = h_0 + h_1 x + .. + h_n x^n$$

The NTRU Lattice L_h^{NTRU} is the $2n$-dimensional associated to $h(x)$ by:

$$M_{\boldsymbol{h}}^{\mathrm{NTRU}} = \left(\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N-1} \\ 0 & 1 & \cdots & 0 & h_{N-1} & h_0 & \cdots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & h_2 & \cdots & h_0 \\ \hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array}\right).$$

NTRU

M_h^{NTRU} is composed of:

1. left upper quadrant as the identity matrix
2. left lower quadrant sa the zero matrix
3. right lower quadrant as qI
4. right upper quadrant as the cyclical permutations (think the rotation) of the coefficients of $h(x)$.

Often abbreviated as:

$$M_h^{NTRU} = \begin{bmatrix} I & h \\ 0 & qI \end{bmatrix}$$

NTRU

To try and convince ourselves that NTRUEncrypt truly is a lattice problem, we must show that the polynomial operations truly are lattice operations.
This is where the **NTRUEncrypt Key recovery problem** appears. Despite the coefficients of some $h(x)$ appearing to be random (due to heuristic random distributions over modulus) , there is a relationship:

$$f(x)h(x) \equiv g(x) \pmod{q}$$

Where breaking NTRUEncrypt comes down to solving this congruence. (Note: the relationship is not unique, due to cyclic rotations).

With this in mind, the relationship can be stated equivalently as:

$$f(x)h(x) \equiv g(x) + q(u(x))$$

For some $u(x)$ satisfting that equality.
We can then perform the matrix multiplication:

$$(f, -u)\begin{bmatrix} I & h \\ 0 & qI \end{bmatrix} = (f, hf - qu) = (f, g)$$

LLL (BONUS)

[1] Input a basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ for a lattice L
[2] Set $k = 2$
[3] Set $\boldsymbol{v}_1^* = \boldsymbol{v}_1$
[4] Loop while $k \leq n$
[5] Loop Down $j = k-1, k-2, \ldots, 2, 1$
[6] Set $\boldsymbol{v}_k = \boldsymbol{v}_k - \lfloor \mu_{k,j} \rceil \boldsymbol{v}_j$ [Size Reduction]
[7] End j Loop
[8] If $\|\boldsymbol{v}_k^*\|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) \|\boldsymbol{v}_{k-1}^*\|^2$ [Lovász Condition]
[9] Set $k = k + 1$
[10] Else
[11] Swap $\boldsymbol{v}_{k-1}$ and $\boldsymbol{v}_k$ [Swap Step]
[12] Set $k = \max(k-1, 2)$
[13] End If
[14] End k Loop
[15] Return LLL reduced basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$

Note: At each step, $\boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_k^*$ is the orthogonal set of vectors obtained by applying Gram–Schmidt (Theorem 7.13) to the current values of $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$, and $\mu_{i,j}$ is the associated quantity $(\boldsymbol{v}_i \cdot \boldsymbol{v}_j^*)/\|\boldsymbol{v}_j^*\|^2$.

Figure 7.8: The LLL lattice reduction algorithm

Thank you!

References and further reading:

- Lecture notes by Dr. Chris Peikert for a graduate course on this topic (http://web.eecs.umich.edu/ cpeikert/lic15/index.html).
- Lecture notes by the father of LWE, Dr. Oded Regev, in 2009: (https://cims.nyu.edu/ regev/teaching/lattices_fall_-2009/index.html)
- The book by Dr. Shari Goldwasser and Dr. Daniele Micciancio, *Complexity of Lattice Problems, A Cryptographic Perspective*
- CSC2414 Notes taught by Dr. Vinod Vaikuntanathan (MIT).
- Abstract Algebra (Artin).
- UTM Book: An Introduction to Mathematical Cryptography (Undergraduate Texts in Mathematics), by Hoffstein, Pipher and Silverman.