Lattice-based cryptography I

Definitions and LLL

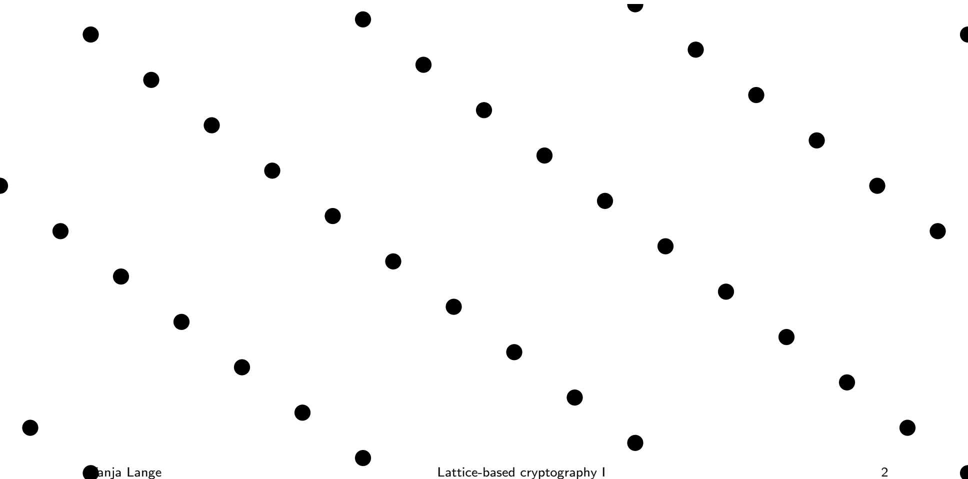
Tanja Lange
(with some slides from Daniel J. Bersntein and Nadia Heninger)

Eindhoven University of Technology

SAC – Post-quantum cryptography

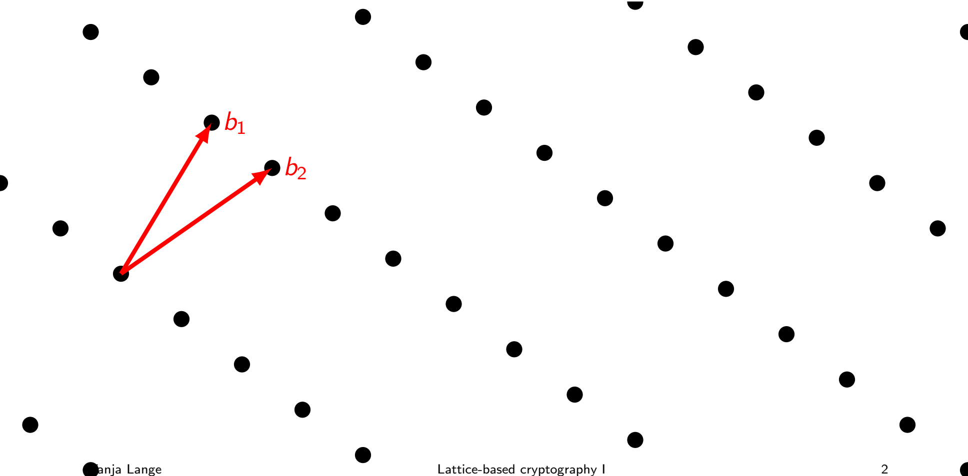
What is a lattice?

A lattice is a discrete subgroup of $\mathbf{R}^n$.



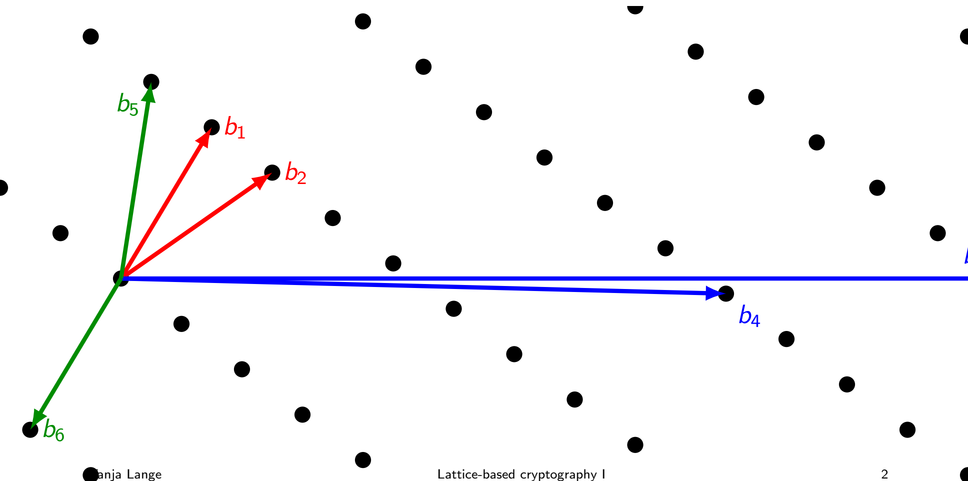
What is a lattice?

We can think of a lattice as being generated by integer multiples of some basis vectors.



What is a lattice?

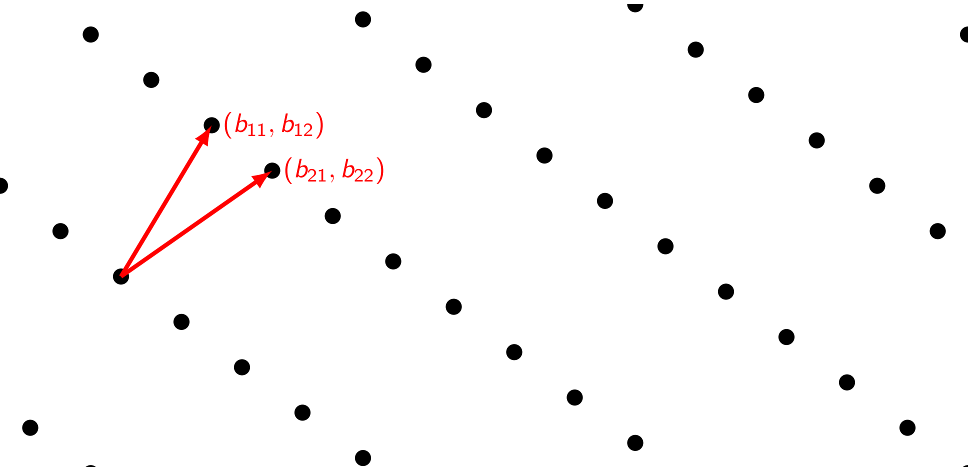
A lattice can have many different bases.



What is a lattice?

We can represent a lattice as a matrix of basis vector coefficients:

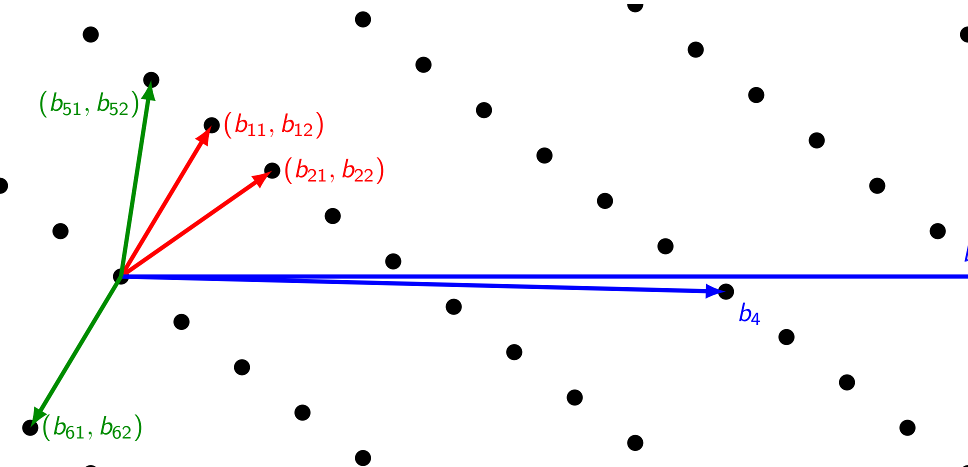
$$B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$



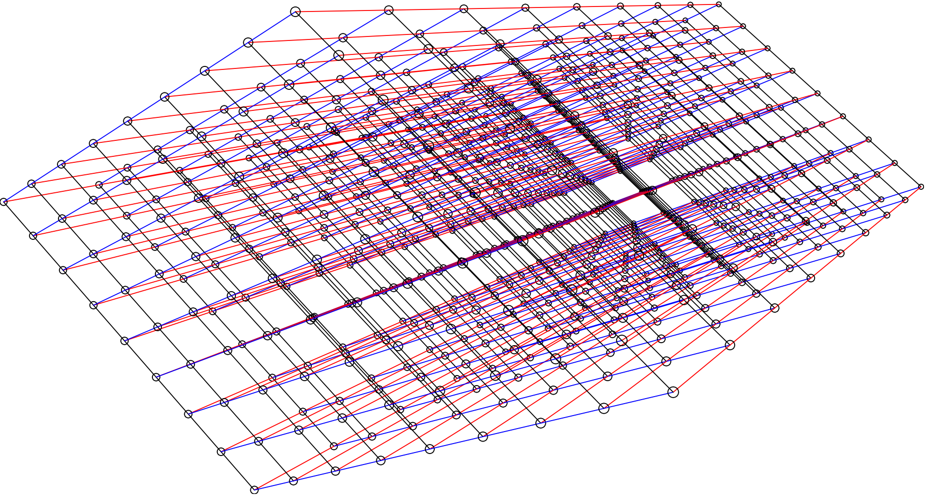
What is a lattice?

A change of basis multiplies B by an integer unimodular matrix U:

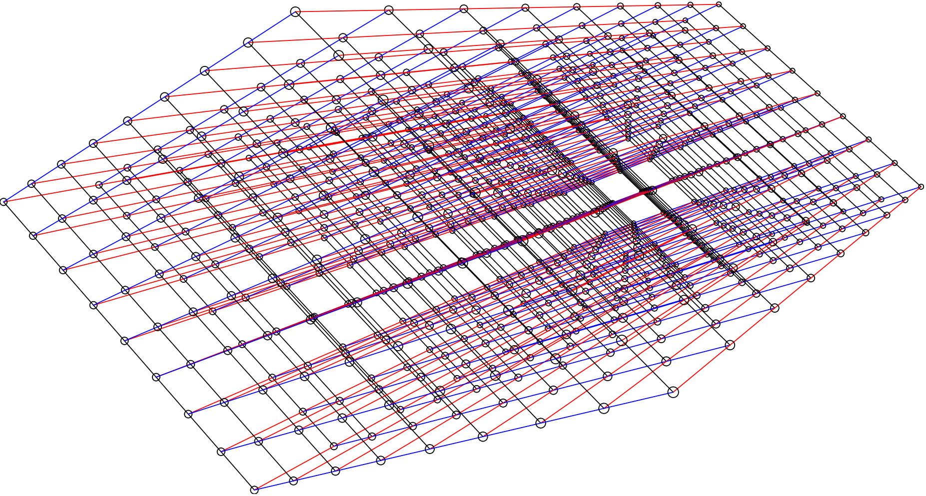
$$B' = \begin{bmatrix} b_{41} & b_{42} \\ b_{51} & b_{52} \end{bmatrix} = U \cdot \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}, \qquad \det(U) \in \{\pm 1\}$$



Here is a lattice in three dimensions



Here is a lattice in three dimensions



Easier to think abstractly of $L = \{\sum_{i=1}^{n} a_i b_i | a_i \in \mathbf{Z}\}$
for a given basis $\{b_1, b_2, \ldots, b_n\}$.
We typically work with lattices of rank n in $\mathbf{R}^n$.

The Shortest Vector Problem (SVP)

How to measure shortness?

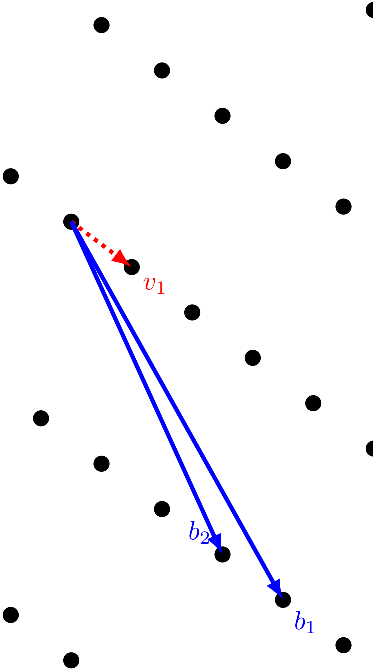
We typically use the Euclidean distance:

$$||(c_1, c_2, \dots, c_n)|| = \sqrt{c_1^2 + c_2^2 + \cdots + c_n^2}$$

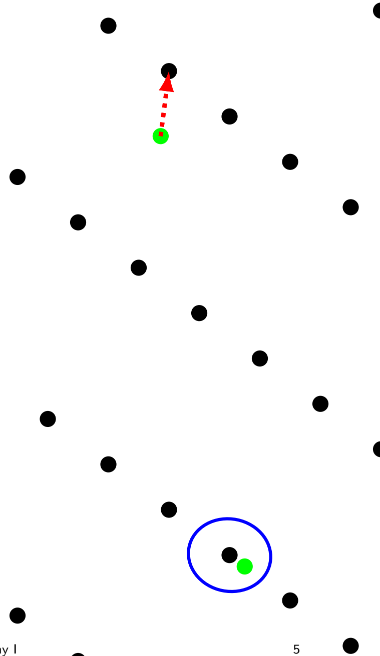
The Shortest Vector Problem (SVP)

Given an arbitrary basis for L, find a shortest nonzero vector v_1 in L.

- Slow algorithm to compute exact solution. (Exponential time!)
- Fast algorithm to compute approximate solution. (Polynomial time!)



Computational problems on lattices: CVP



Closest Vector Problem (CVP)

Given an arbitrary basis for L, and a point x find the vector in L closest to x.

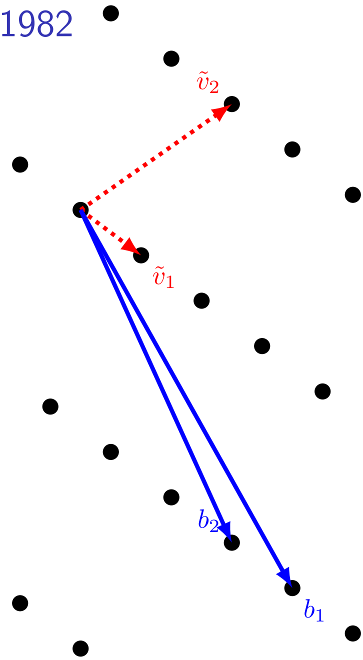
- CVP is NP-hard.

We can find pretty close vectors efficiently.

This gets much easier with a basis of short and close-to-orthogonal vectors:
we can just round.

LLL – Lenstra, Lenstra, and Lovász, 1982

- On input $\{b_1, b_2, \ldots, b_n\}$ as matrix B, output shorter and more orthogonal basis $\{\tilde{v}_1, \tilde{v}_2, \ldots, \tilde{v}_n\}$ with $\tilde{v}_j = \sum a_i b_i, a_i \in \mathbf{Z}$.
- LLL uses many elements from Gram-Schmidt orthogonalization:
  - for $j = 1$ to n
  - for $i = 1$ to $j - 1$
  - $\mu_{ij} = \frac{\langle v_i^*, b_j \rangle}{\langle b_i^*, b_i^* \rangle}$
  - $b_j^* = b_j - \sum_{i=1}^{j-1} \mu_{ij} b_i^*$
- Note that the μ_{ij} are not integers, so the b_j^* are not in the lattice.
- A lattice basis is LLL-reduced for parameter $0.25 < \delta < 1$ if
  - $|\mu_{ij}| \leq 0.5$ for all $1 \leq j < i \leq n$,
  - $(\delta - \mu_{i-1i}^2)||b_{i-1}^*||^2 \leq ||b_i^*||^2$.
- This guarantees $||\tilde{v}_1|| \leq (2/\sqrt{4\delta - 1})^{(n-1)/2} \det(B)^{1/n}$.



LLL algorithm (from Cohen, GTM 138, transposed)

Input: Basis $\{b_1, b_2, \ldots, b_n\}$ of lattice L, $0.25 < \delta < 1$

Output: LLL reduced basis for L with parameter δ

1. $k \leftarrow 2,\ k_{\max} \leftarrow 1,\ b_1^* \leftarrow b_1,\ B_1 = \langle b_1, b_1 \rangle$
2. if $k \leq k_{\max}$ go to step 3
   else $k_{\max} \leftarrow k,\ b_k^* \leftarrow b_k$, for $j = 1$ to $k - 1$
   - put $\mu_{jk} \leftarrow \langle b_j^*, b_k \rangle / B_j$ and $b_k^* \leftarrow b_k^* - \mu_{jk} b_j^*$

   $B_k = \langle b_k, b_k \rangle$
3. Execute RED$(k, k-1)$. If $(\delta - \mu_{i-1i}^2) B_{k-1} > B_k$ execute SWAP(k) and $k \leftarrow \max\{2, k-1\}$; else for $= k-2$ down to 1 execute RED(k, j) and $k \leftarrow k+1$.
4. If $k \leq n$ go to step 2; else output basis $\{b_1, b_2, \ldots, b_n\}$.

- RED(k, j): If $|\mu_{jk}| \leq 0.5$ return; else $q \leftarrow \lfloor \mu_{jk} \rceil$, $b_k \leftarrow b_k - q b_j$, $\mu_{jk} \leftarrow \mu_{jk} - q$, for $i = 1$ to $j - 1$ put $\mu_{ik} \leftarrow \mu_{ik} - q \mu_{ij}$ and return.
- SWAP(k): Swap b_k and b_{k-1}. If $k > 2$ for $j = 1$ to $k - 2$ swap μ_{jk} and μ_{jk-1} and update all variables to match (see p.88 in Cohen)

For a nice visualization see pages 61–66 of
http://thijs.com/docs/lec1.pdf. (Animations only work in acroread.)

Sage has an implementation of LLL, you can call it on matrices.