Lattice Based Cryptography

Halil İbrahim Kaplan

TDBY / Kripto Analiz Laboratuvarı

halil.kaplan@tubitak.gov.tr

2022

TÜBİTAK
BİLGEM

Overview

1. **Basics Of Lattices**
2. **SVP and CVP**
3. **Sieving Algorithm**
4. **LLL Algorithm**
5. **BKZ Algorithm**
6. **SIS and LWE**
7. **LWE applications**

Basics Of Lattices

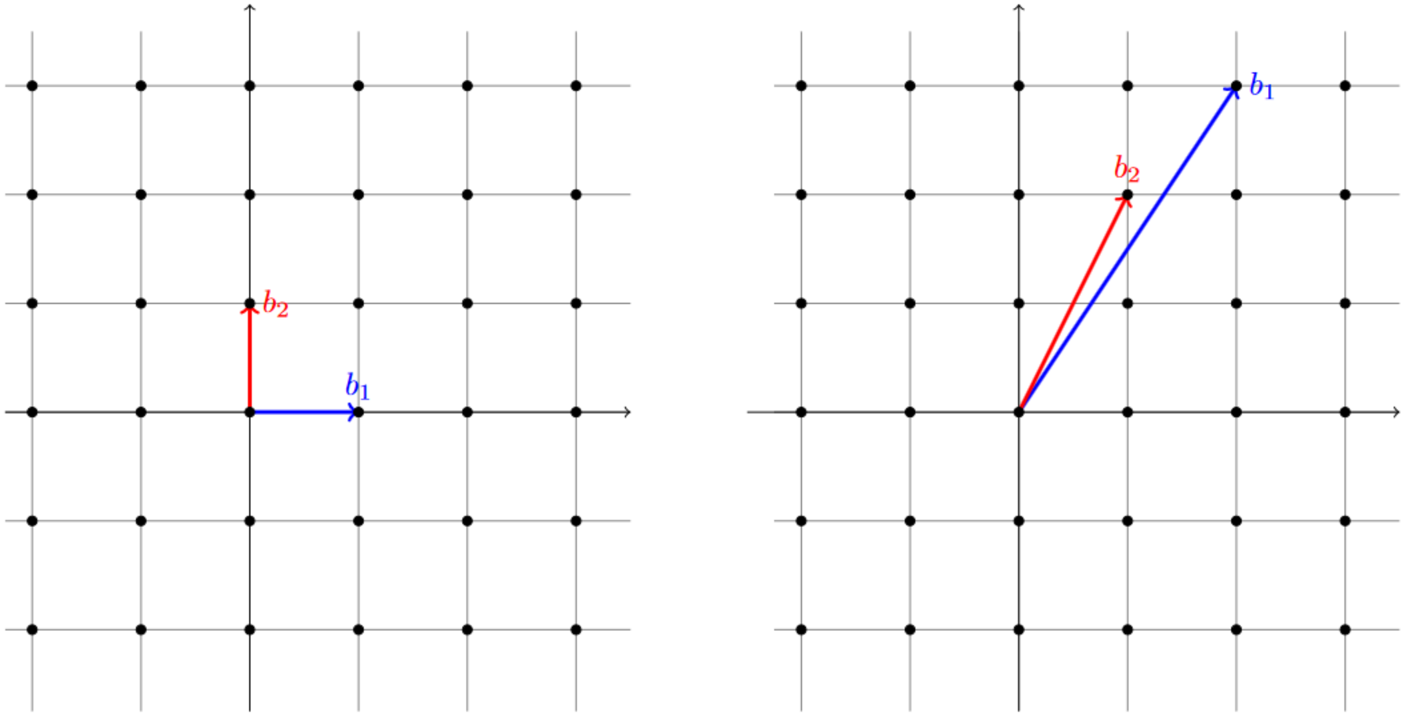
Let $b_1, b_2 \ldots b_n \epsilon \mathbb{R}^m$ be linearly independent vectors.

Lattice generated by them is defined as:

$$\mathcal{L}(\underbrace{\mathbf{b}_1, \ldots, \mathbf{b}_n}_{\text{Basis of Lattice}}) \stackrel{\text{def}}{=} \left\{ \sum_{i=1}^{n} x_i \mathbf{b}_i \mid x_i \in \mathbb{Z} \right\}$$

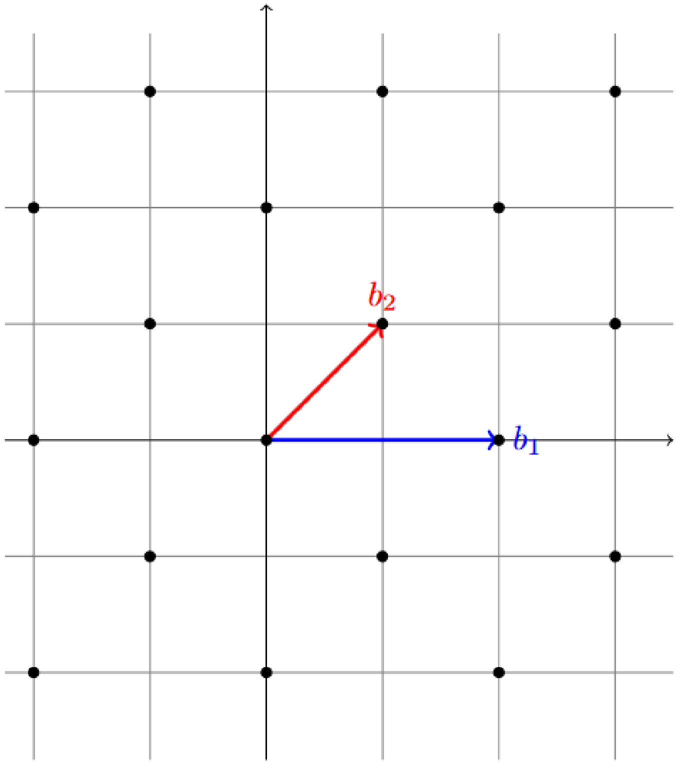
m = Dimention of Lattice

n = Rank of Lattice

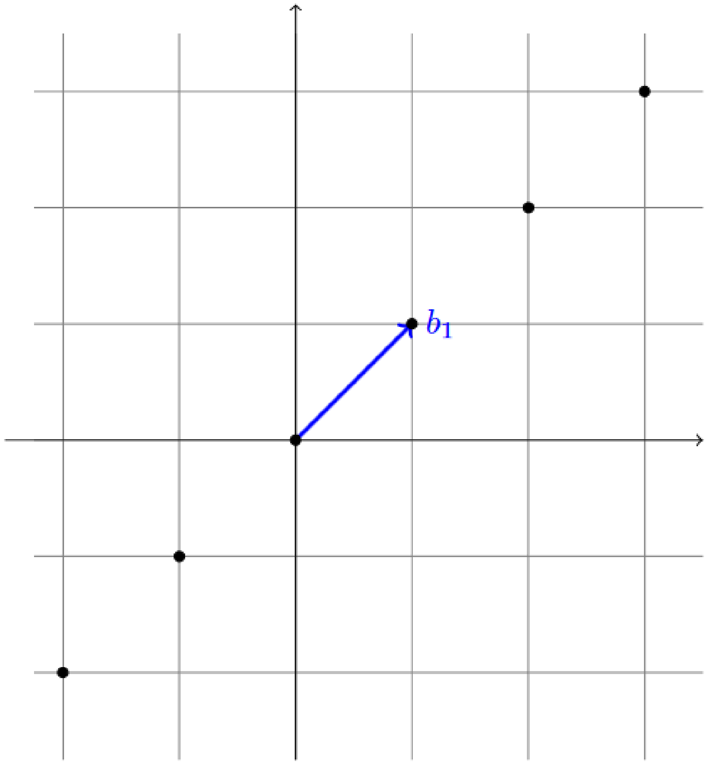


(a) The lattice $\mathbb{Z}^2$ with basis vectors $(0,1)$ and $(1,0)$.

(b) The lattice $\mathbb{Z}^2$ with a different basis consisting of vectors $(1,2)$ and $(2,3)$. In fact, any lattice has infinitely many bases.



(c) A full-rank lattice generated by the basis vectors $(1, 1)$ and $(2, 0)$. Note that this is a sub-lattice of $\mathbb{Z}^2$.



(d) A *non full-rank* lattice with basis vector $(1, 1)$

$m \neq n$

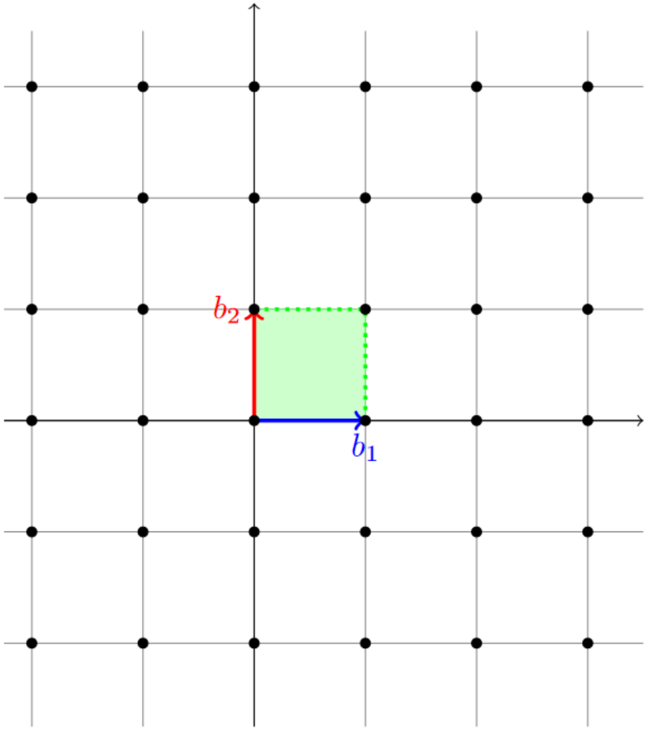
Let $b_1, b_2 \ldots b_n \epsilon \mathbb{R}^m$ be linearly independent vectors.

Their fundamental parallelepiped is defined as:

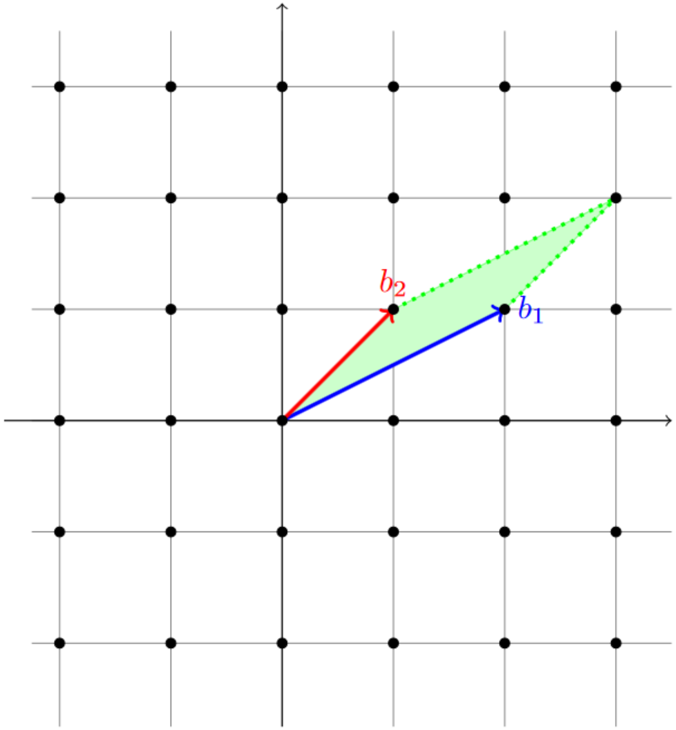
$$\mathcal{P}(\mathbf{b}_1, \ldots, \mathbf{b}_n) \stackrel{\text{def}}{=} \left\{ \sum_{i=1}^{n} x_i \mathbf{b}_i \mid x_i \in \mathbb{R}, 0 \leq x_i < 1 \right\}$$

Different bases of the same lattice generate different fundamental paralellepipeds.

Basics Of Lattices



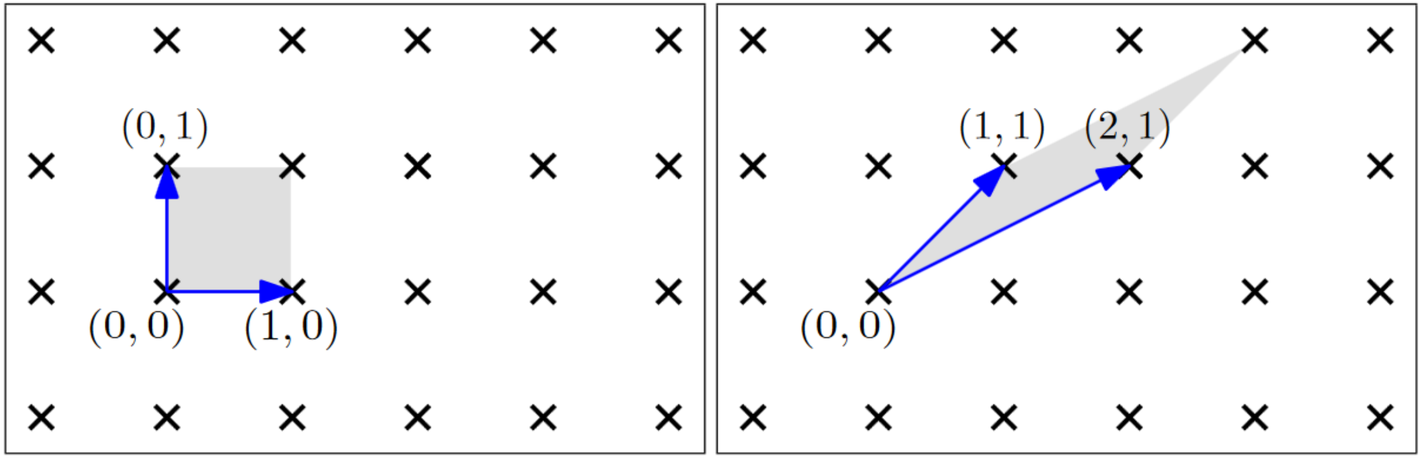
(a) The lattice $\mathbb{Z}^2$ with basis vectors $(0, 1)$ and $(1, 0)$ and the associated fundamental parallelepiped.



(b) The lattice $\mathbb{Z}^2$ with a different basis consisting of vectors $(1, 1)$ and $(2, 1)$, and the associated fundamental parallelepiped.

Determinant of Lattice : n-dimensional volume of its fundamental parallelepiped.

The parallelepipeds associated with different bases of a lattice have the same volume. Thus, the determinant is a lattice invariant.



(a) A basis of $\mathbb{Z}^2$ (b) Another basis of $\mathbb{Z}^2$

Theorem

Let $\mathcal{L}$ be a full-rank n-dimentional lattice, and let $b_1, b_2 \ldots b_n \epsilon \ \mathbb{R}^m$ be linearly independent vectors in $\mathcal{L}$.

$$b_1, b_2 \ldots b_n \text{ forms basis of } \mathcal{L} \iff \mathrm{P}(b_1, b_2 \ldots b_n) \cap \mathcal{L} = \{0\}$$

Proof : Omitted

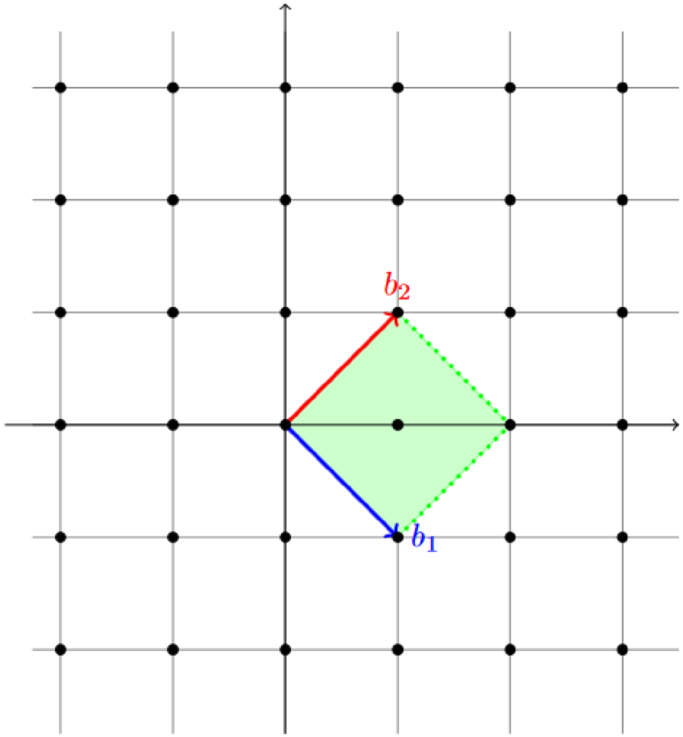


Figure 3: $\mathbf{b}_1$ and $\mathbf{b}_2$ do not form a basis of $\mathbb{Z}^2$. Note that the parallelepiped of $\mathbf{b}_1$ and $\mathbf{b}_2$ contains a non-zero lattice point, namely $(1, 0)$.

Dual Lattices :

Defined as

$$\Lambda^* = \{y \in \text{span}(\Lambda) \mid \forall x \in \Lambda,\ \langle x, y \rangle \in \mathbb{Z}\}.$$

In words, the dual of Λ is the set of all points (in the span of Λ) whose inner product with any of the points in Λ is integer.

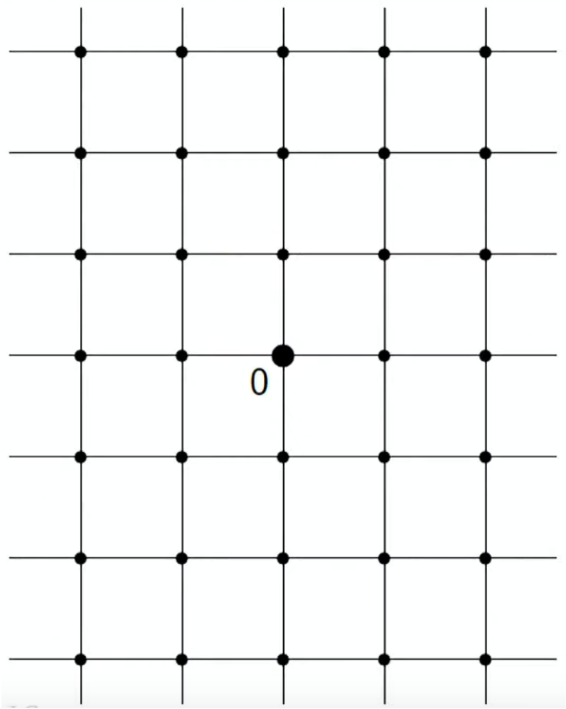
- Λ^* is also a lattice
- ${\Lambda^*}^* = \Lambda$

Dual Lattices :

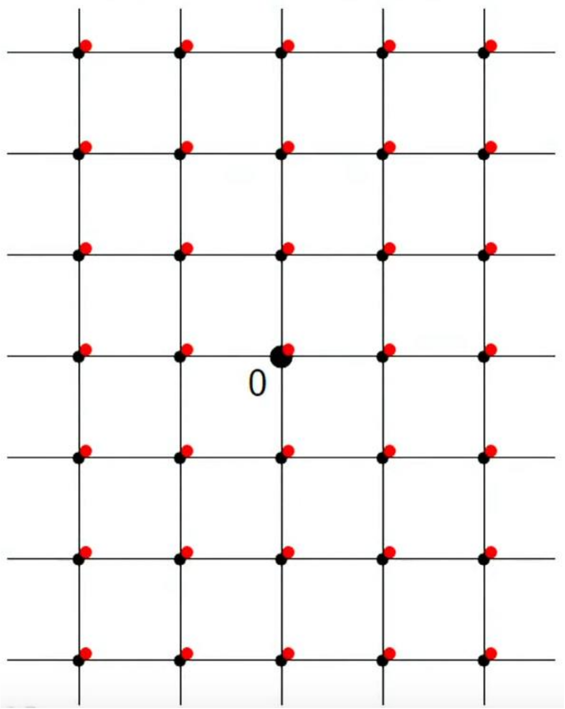
Defined as

$$\Lambda^* = \{y \in \mathrm{span}(\Lambda) \mid \forall x \in \Lambda,\ \langle x, y\rangle \in \mathbb{Z}\}.$$

$\Lambda = \mathbb{Z}^n$



$\Lambda^* = \mathbb{Z}^n$

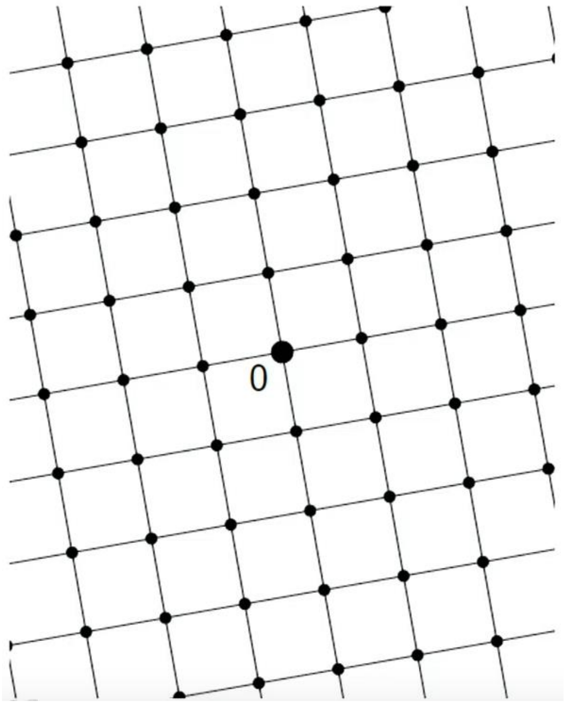


Dual Lattices :

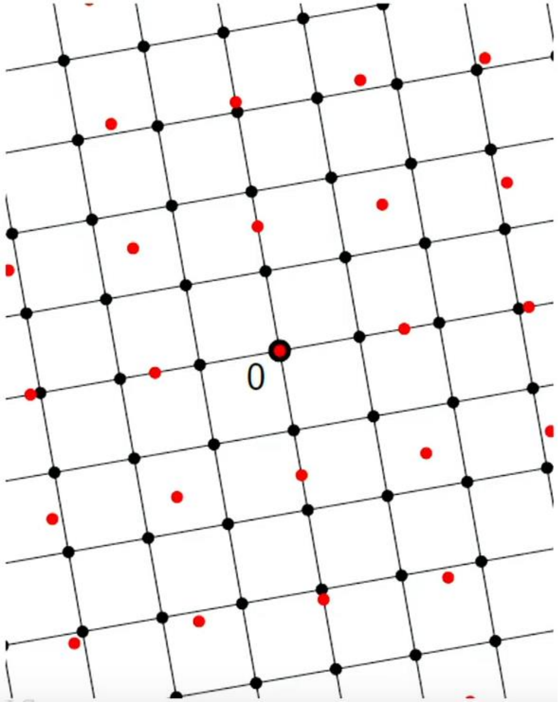
Defined as

$$\Lambda^* = \{y \in \mathrm{span}(\Lambda) \mid \forall x \in \Lambda,\ \langle x, y \rangle \in \mathbb{Z}\}.$$

$\Lambda = 2R\mathbb{Z}^n$



$\Lambda^* = \frac{1}{2}R\mathbb{Z}^n$

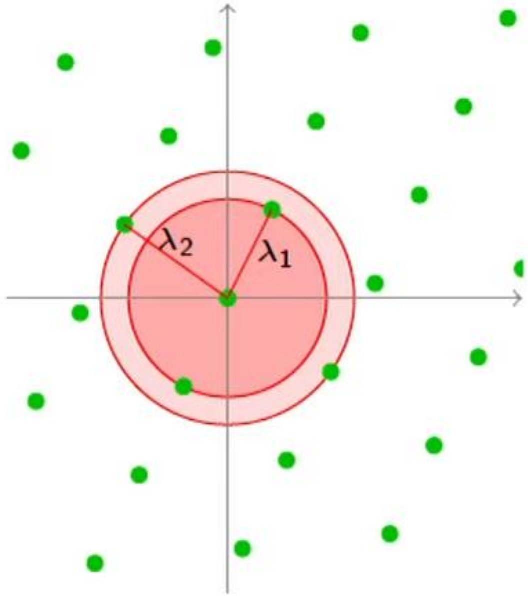


SVP and CVP

Successive Minima:

Suppose that we select vectors in L as:

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



The lengths

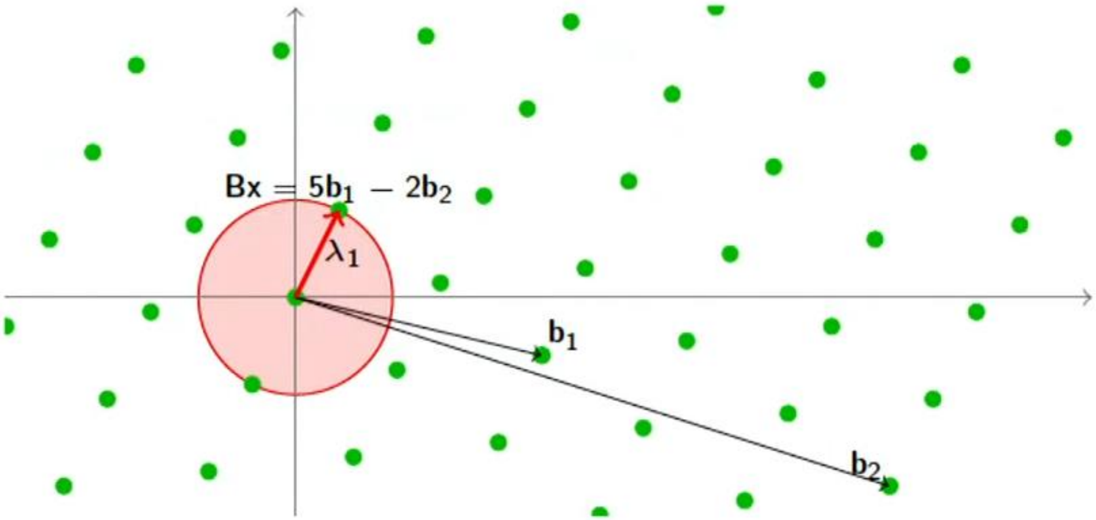
$$\lambda_1 = \|\mathbf{v}_1\|, \ \lambda_2 = \|\mathbf{v}_2\|, \ \ldots, \ \lambda_n = \|\mathbf{v}_n\|$$

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

<u>The Shortest Vector Problem</u> (SVP):

Given a Lattice $\mathcal{L}(B)$, find a (nonzero) Lattice vector **Bx** (with x ϵ $\mathbb{Z}^k$)

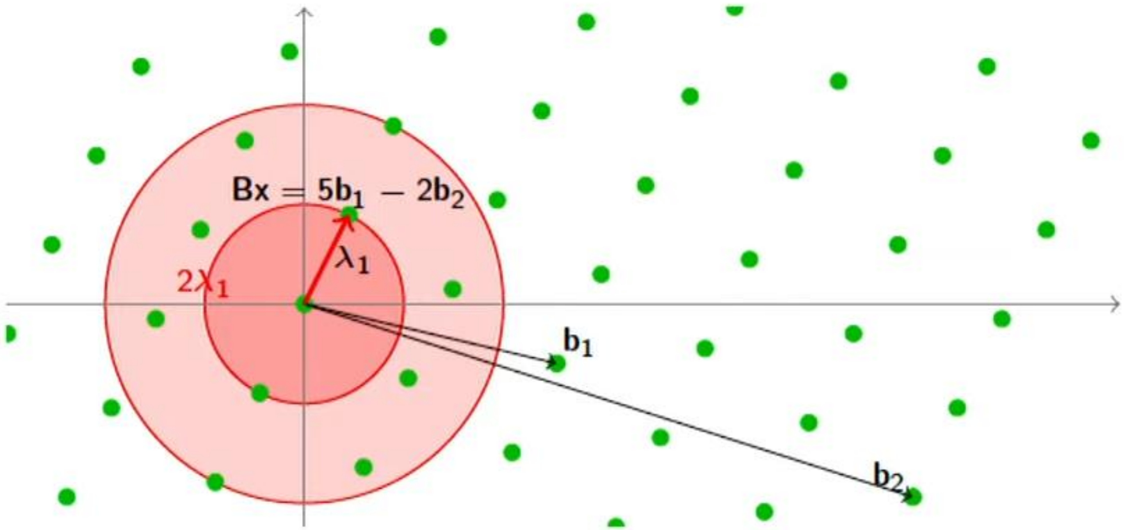
of length (at most) || **Bx** || $\leq \lambda_1$(Successive Minima)



<u>The Shortest Vector Problem</u> (SVP$_\gamma$):

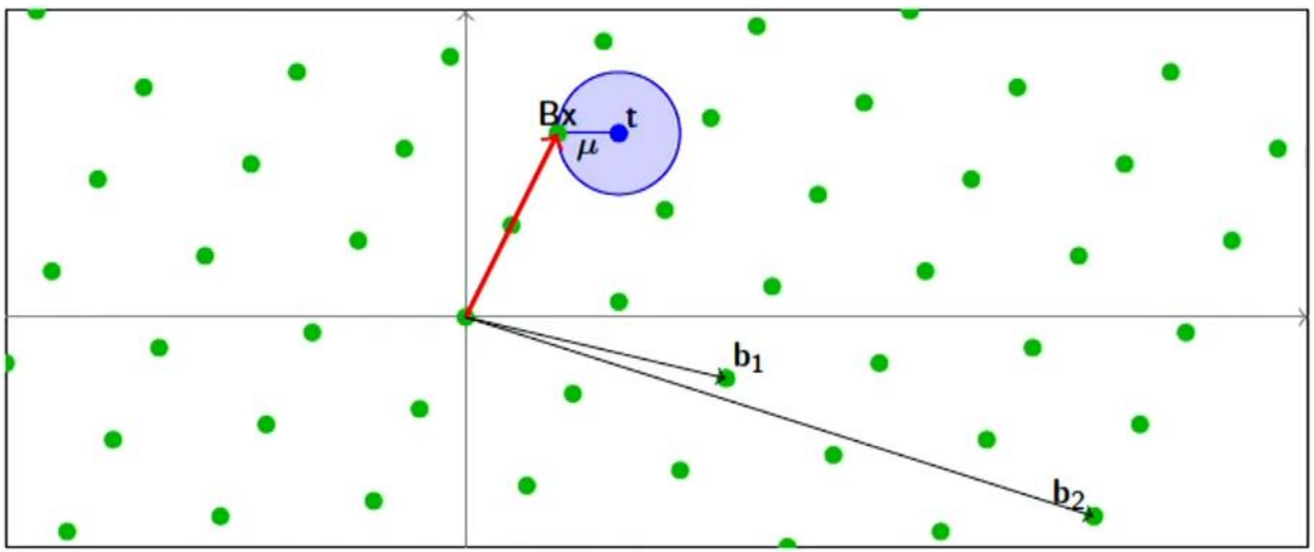
Given a Lattice $\mathcal{L}(B)$, find a (nonzero) Lattice vector **Bx** (with x ϵ $\mathbb{Z}^k$)

of length (at most) $\| \mathbf{Bx} \| \leq \gamma . \lambda_1$



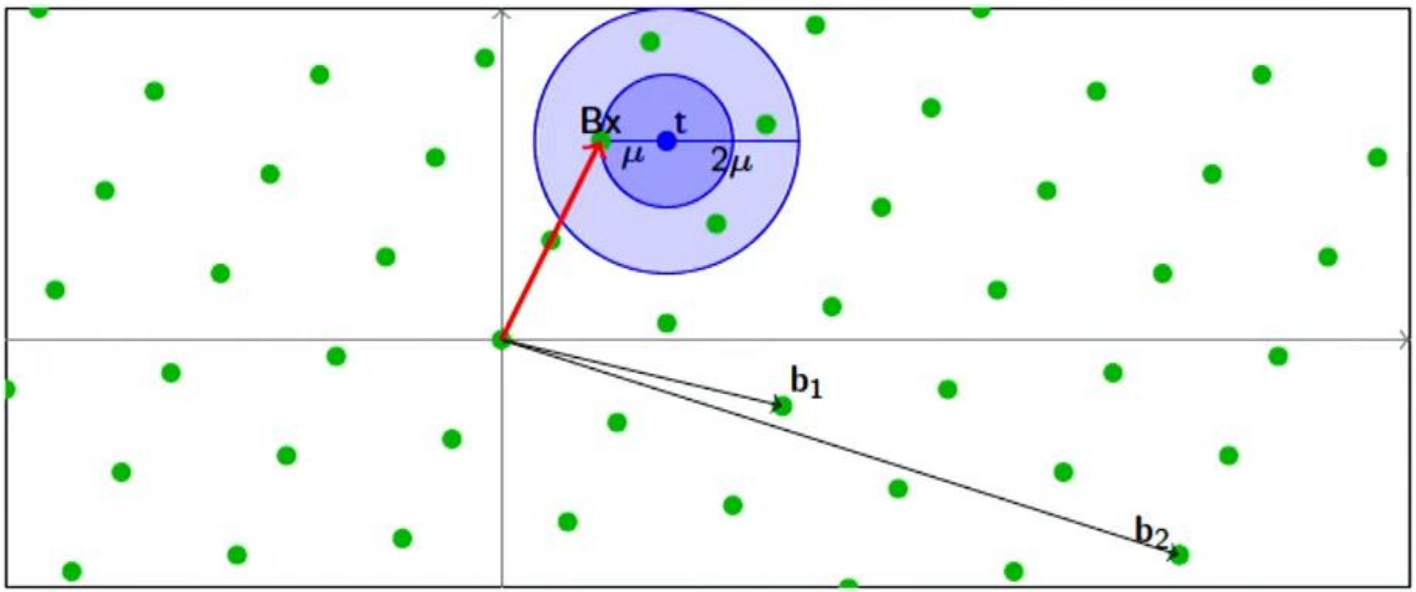
<u>The Closest Vector Problem</u> (CVP):

Given a Lattice $\mathcal{L}(B)$ and a target point **t** , find a Lattice vector **Bx** within distance || **Bx - t** || ≤ μ from the target



The Closest Vector Problem (CVP$_\gamma$):

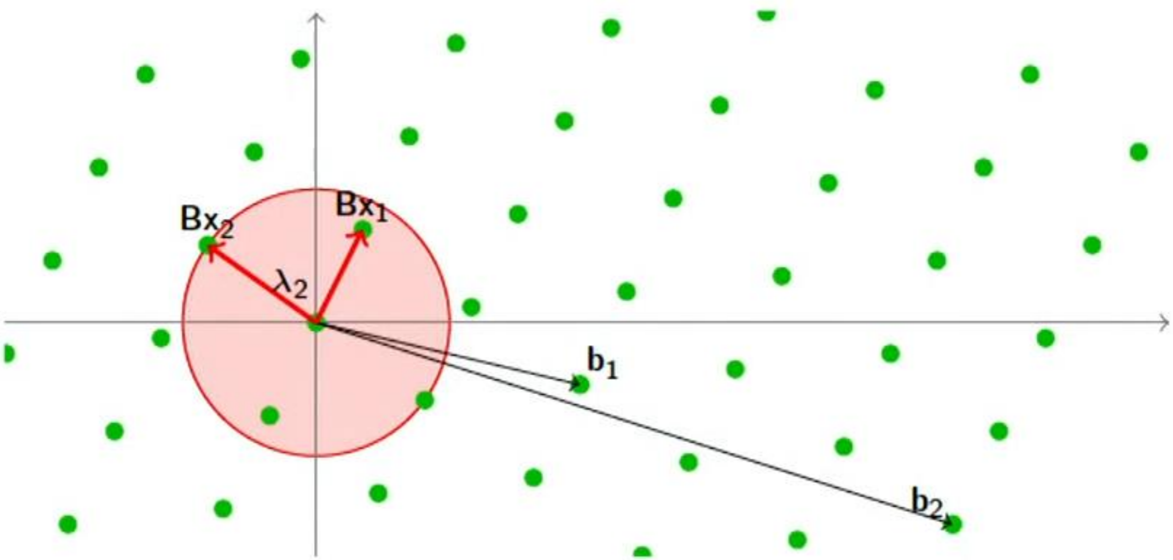
Given a Lattice $\mathcal{L}(\boldsymbol{B})$ and a target point **t** , find a Lattice vector **Bx** within distance || **Bx - t** || ≤ $\gamma.\mu$ from the target



Shortest Independent Vectors Problem (SIVP):

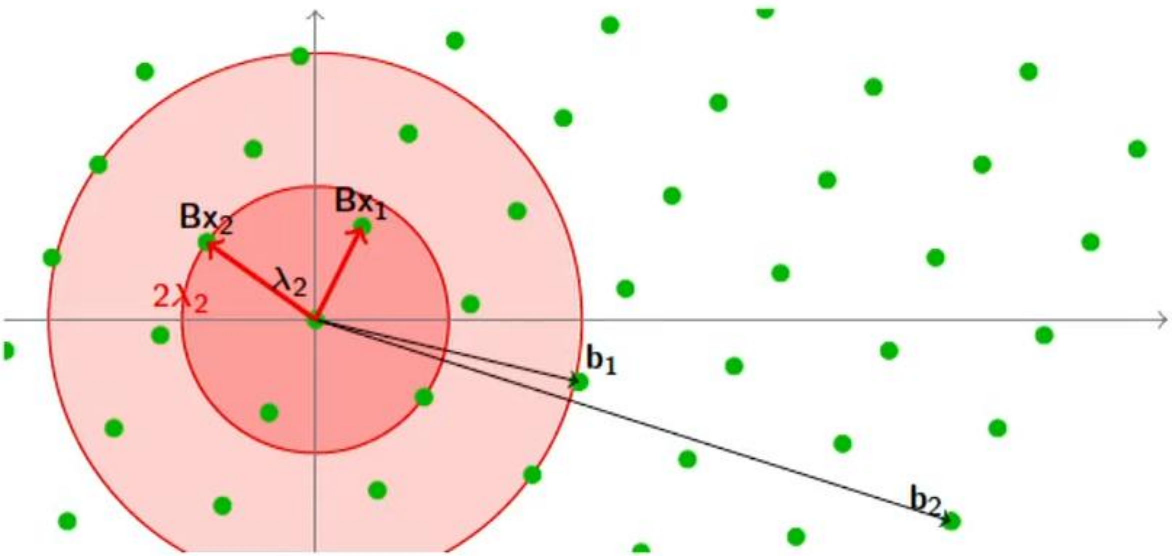
Given a Lattice $\mathcal{L}(\boldsymbol{B})$, find n linearly independent Lattice vectors

$\boldsymbol{Bx_1}, \boldsymbol{Bx_2} \ldots \boldsymbol{Bx_n}$ of length (at most) $max_i \| \boldsymbol{Bx_i} \| \leq \lambda_n$



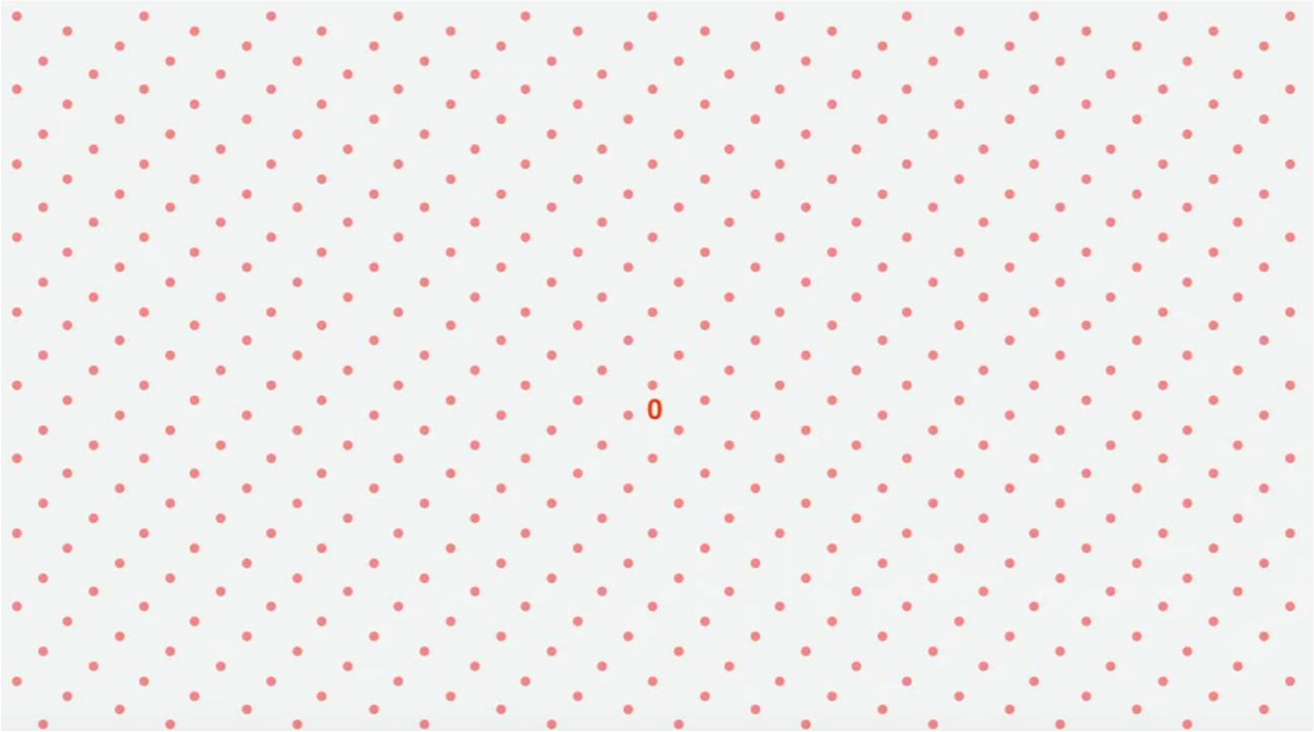
<u>Shortest Independent Vectors Problem</u> (SIVP$_\gamma$):

Given a Lattice $\mathcal{L}(\boldsymbol{B})$, find n linearly independent Lattice vectors $\boldsymbol{Bx_1}, \boldsymbol{Bx_2} \ldots \boldsymbol{Bx_n}$ of length (at most) $max_i \| \boldsymbol{Bx_i} \| \leq \gamma . \lambda_n$



Sieving Algorithm

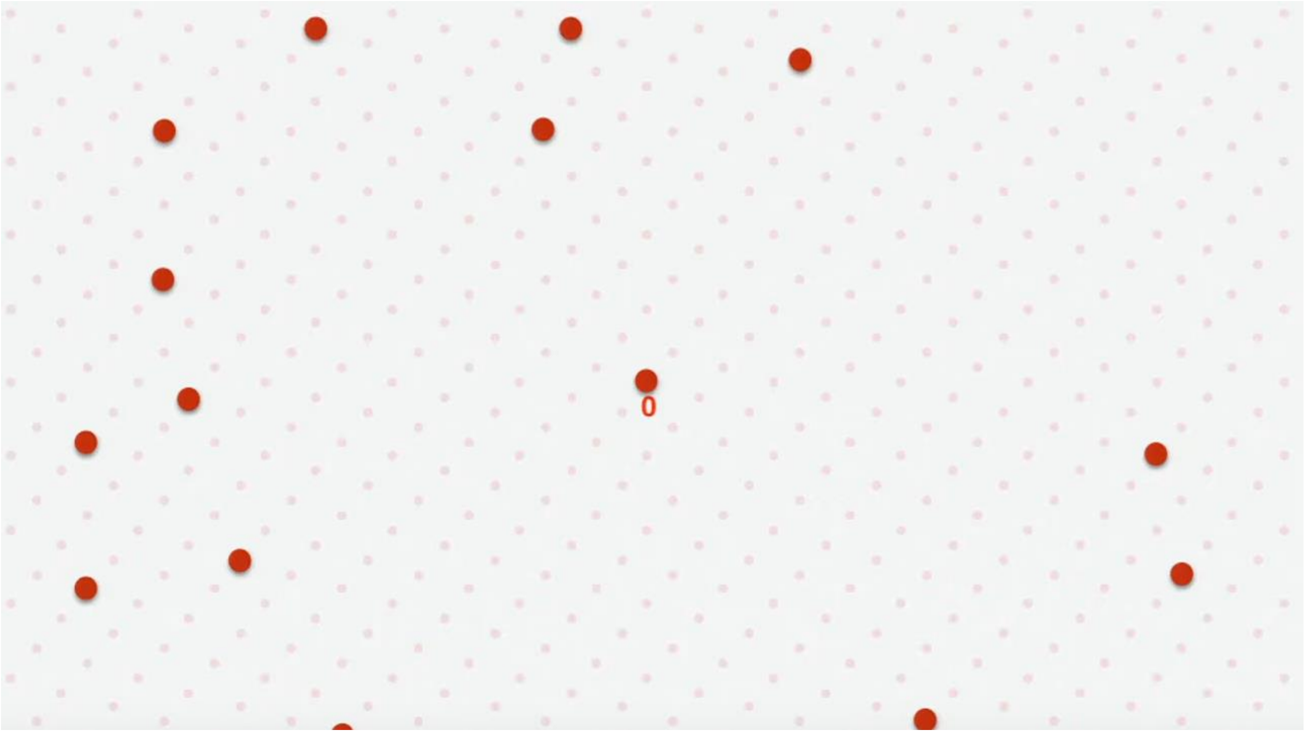
Lets we have a Lattice.



Sieving Algorithm

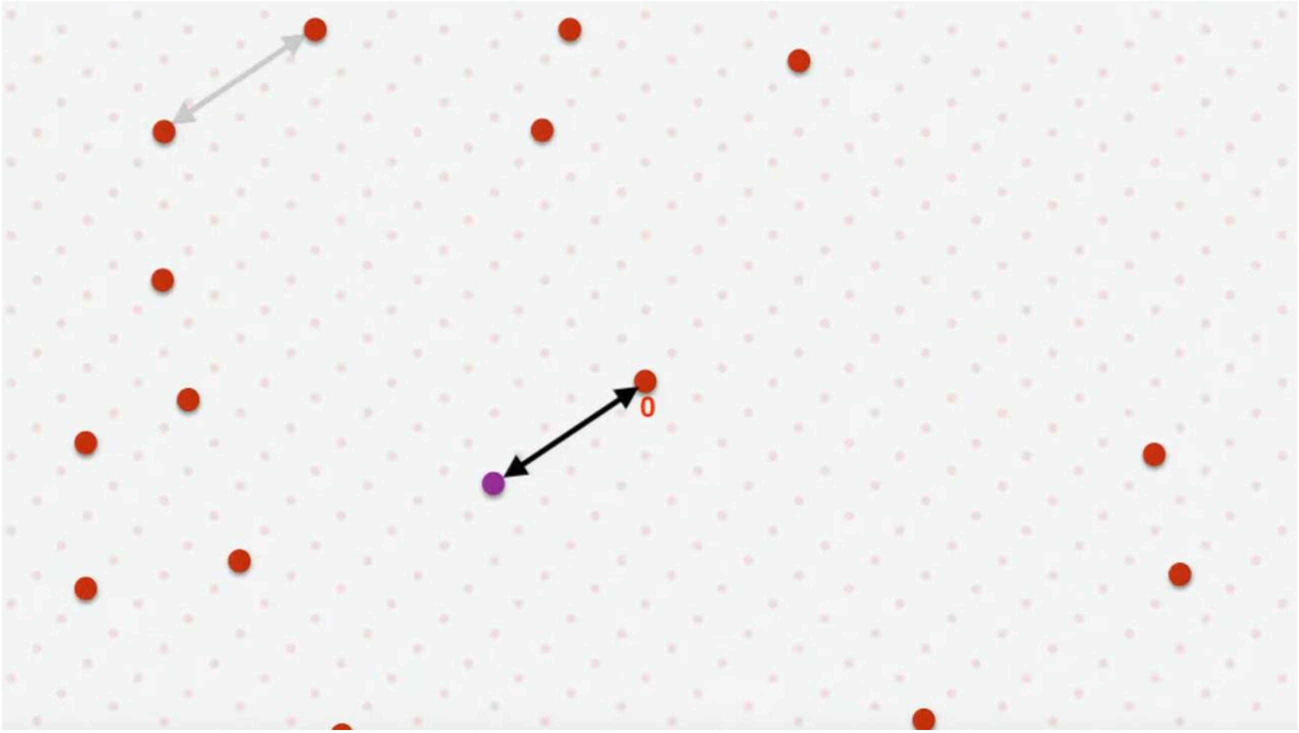
Finding short points in Lattice = HARD

Finding long points in Lattice = EASY



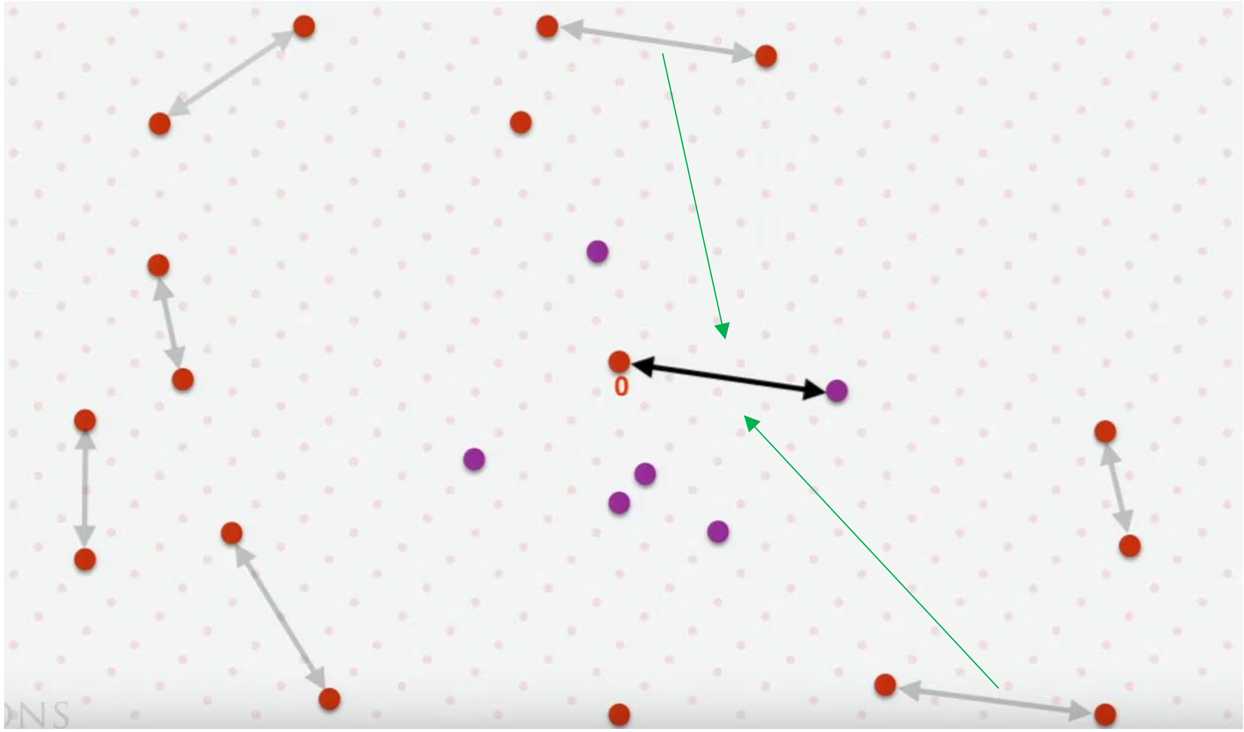
Sieving Algorithm

If you know two Lattice points are close , then take their difference to the origin $\Longrightarrow$ You will get shorter Lattice point

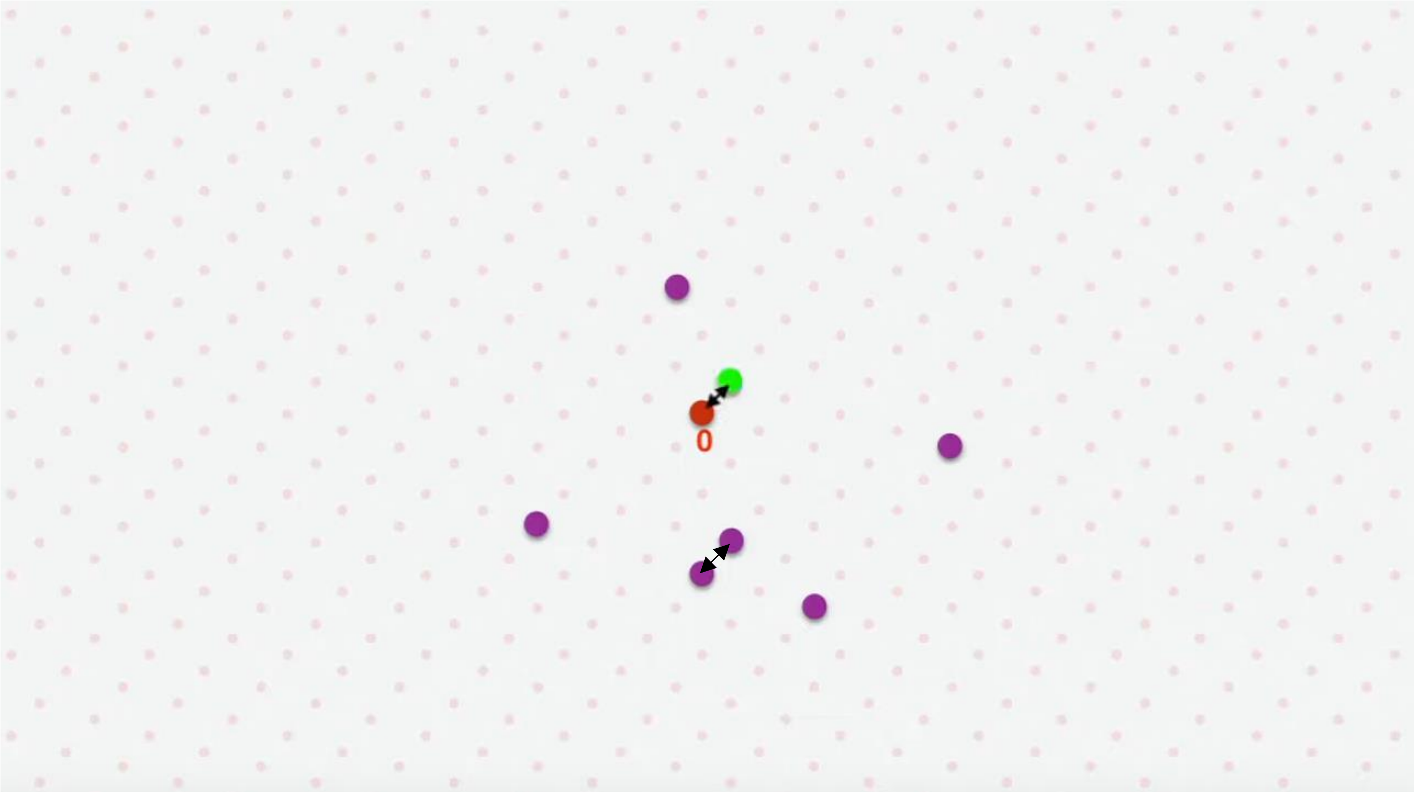


Sieving Algorithm

- We didn’t find the shortest vector.
- We found a collision.



- We continue with new purple points.
- We found the shortest vector.



GOOD, BUT

How do you start ?

How do you find close pairs ?

Which pairs do you choose ?

What is distribution of the vectors at each step ?

How common are collisions ?

UNCLEAR

Sieving Algorithm

Perturbation	$2^{O(n)}$	[AKS01]
Perturbation	$2^{2.5n}$	[NV08, PS09, MV10, …]
Perturbation	$2^{0.802n}$	Approximate [LWXZ11, WLW15, AUV19]
Heuristic (no proof of correctness)	$(3/2)^{n/2+o(n)} \approx 2^{0.29n}$	[NV08, Laa15, BDGL15, BLS16, …]
Sieving by Averages (Discrete Gaussian)	$2^{n+o(n)}$	[ADRS15, ADS15, AS18]
Sieving by Averages (Discrete Gaussian)	$2^{n/2+o(n)}$	• approx decision SVP [ADRS15] • HermiteSVP [ALS20] • Probably SVP [You!20]
????	Fast!	[Veysel01]

LLL Algorithm

Arjen Lenstra

Hendrik Lenstra

Laszlo Lovasz

LLL Algorithm

Lattice Reduction: Practical problem of solving SVP and CVP, or more generally of finding reasonably short vectors and reasonably good bases.

LLL Algorithm :

- Finds moderately short lattice vectors in polynomial time
- Finding very short (or very close) vectors is currently still exponentially hard.

Gram-Schmidt Orthogonalization Algorithm:

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

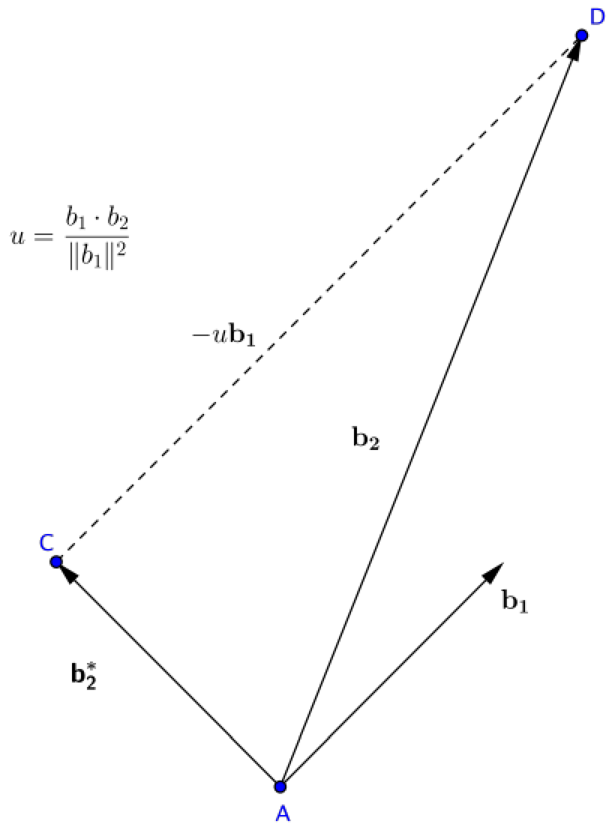


Figure : Orthogonal projection. The set $\{\mathbf{b_1}, \mathbf{b_2^*}\}$ is the orthogonal basis for the lattice generated by basis $\{\mathbf{b_1}, \mathbf{b_2}\}$

We want that basis satisfies 2 conditions:

Size condition : $1 \leq j < i \leq n$: $|\mu_{i,j}| \leq 0.5$

Makes coefficients smaller.

Lovász condition : $\delta \|\mathbf{b}_{k-1}^*\|^2 \leq \|\mathbf{b}_k^*\|^2 + \mu_{k,k-1}^2 \|\mathbf{b}_{k-1}^*\|^2$

Makes basis vectors somewhat orthogonal

LLL Algorithm

```
INPUT
    a lattice basis b0, b1, ..., bn in Z^m
    a parameter δ with 1/4 < δ < 1, most commonly δ = 3/4
PROCEDURE
    B* <- GramSchmidt({b0, ..., bn}) = {b0*, ..., bn*};  and do not normalize
    μi,j <- InnerProduct(bi, bj*)/InnerProduct(bj*, bj*);   using the most current values of bi and bj*
    k <- 1;
    while k <= n do
        for j from k-1 to 0 do
            if |μk,j| > 1/2 then
                bk <- bk - ⌊μk,j⌉bj;
               Update B* and the related μi,j's as needed.
               (The naive method is to recompute B* whenever bi changes:
                B* <- GramSchmidt({b0, ..., bn}) = {b0*, ..., bn*})
            end if
        end for
        if InnerProduct(bk*, bk*) > (δ - μ²k,k-1) InnerProduct(bk-1*, bk-1*) then
            k <- k + 1;
        else
            Swap bk and  bk-1;
            Update B* and the related μi,j's as needed.
            k <- max(k-1, 1);
        end if
    end while
    return B the LLL reduced basis of {b0, ..., bn}
OUTPUT
    the reduced basis b0, b1, ..., bn in Z^m
```

Size condition satisfied

Lovasz condition satisfied

Algorithm 1: LLL Algorithm

Input: $\{\mathbf{b_1}, \mathbf{b_2}, \cdots, \mathbf{b_n}\}$

Repeat two steps until find the LLL reduced basis

Step 1: Gram-Schmidt orthogonalization

```
for i = 1 to n do
    for k = i − 1 to 1 do
        m ← nearest integer of u_{k,i}
        b_i ← b_i − m b_k
    end
end
```

Step 2: Check Condition 2, and swap

```
for i = 1 to n − 1 do
    if ||b_{i+1}* + u_{i,i+1} b_i*||^2 < (3/4)||b_i*||^2 then
        swap b_{i+1} and b_i
        go to step 1
    end
end
```

LLL EXAMPLE :

Find an LLL-reduced basis for $b_1 = (1,0,0), b_2 = (0,1,1), b_3 = (1,0,1)$.
Let $\|b_i^*\|^2 = B_i$

Step 1:

$$b_1^* = b_1 = (1,0,0), B_1 = 1$$

Step 2:

$$\mu_{2,1} = \frac{(0,1,1)(1,0,0)}{1} = 0$$

Condition 1:

$$|\mu_{2,1}| = 0 \leq \frac{1}{2} \quad \checkmark$$

$$b_2^* = b_2 - \mu_{2,1} b_1^* = (0,1,1) - 0(1,0,0) = (0,1,1), \quad B_2 = 2$$

Condition 2:

$$B_2 \geq (\frac{3}{4} - \mu_{2,1}^2) B_1$$

$$2 \geq (\frac{3}{4} - 0)1 \quad \checkmark$$

Step 3:

$$\mu_{3,2} = \frac{\langle b_3, b_2^* \rangle}{\langle b_2^*, b_2^* \rangle} = \frac{(1,0,1)(0,1,1)}{2} = \frac{1}{2}$$

Condition 1:

$$|\mu_{3,2}| = \frac{1}{2} \leq \frac{1}{2} \quad \checkmark$$

$$\mu_{3,1} = \frac{\langle b_3, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} = \frac{(1,0,1)(1,0,0)}{1} = 1$$

Condition 1:

$$|\mu_{3,1}| = 1 \leq \frac{1}{2} \quad \times \text{ Cond. 1 is violated.}$$

Reduce:

$$r = \left\lfloor \frac{1}{2} + \mu_{3,1} \right\rfloor = \left\lfloor \frac{1}{2} + 1 \right\rfloor = 1$$

$$b_3 = b_3 - r b_1 = (1,0,1) - 1(1,0,0) = (0,0,1)$$

$$\mu_{3,1} = \mu_{3,1} - r = 1 - 1 = 0$$

New basis: $b_1 = (1,0,0), b_2 = (0,1,1), b_3 = (0,0,1)$

$$b_3^* = b_3 - \mu_{3,2} b_2^* - \mu_{3,1} b_1^* = (0,0,1) - \frac{1}{2}(0,1,1) - 0 = (0, -\frac{1}{2}, \frac{1}{2}), \quad B_3 = \frac{1}{2}$$

Condition 2:

$$B_3 \geq (\frac{3}{4} - \mu_{3,2}^2) B_2$$

LLL EXAMPLE :

$$\frac{1}{2} \geq 1 \quad \times \text{ Cond. 2 is violated.}$$

Swap: $b_1 = (1, 0, 0), b_2 = (0, 0, 1), b_3 = (0, 1, 1)$
After swap, we start again.
Step 1: ✓
Step 2:

$$\mu_{2,1} = \frac{(0,0,1)(1,0,0)}{1} = 0$$

Condition 1:

$$|\mu_{2,1}| = 0 \leq \frac{1}{2} \quad \checkmark$$

$$b_2^* = b_2 - \mu_{2,1} b_1^* = (0,0,1) - 0 = (0,0,1), \quad B_2 = 1$$

Condition 2:

$$B_2 \geq (\frac{3}{4} - \mu_{2,1}^2) B_1$$

$$1 \geq (\frac{3}{4} - 0)1 \quad \checkmark$$

Step 3:

$$\mu_{3,2} = \frac{\langle b_3, b_2^* \rangle}{\langle b_2^*, b_2^* \rangle} = \frac{(0,1,1)(0,0,1)}{1} = 1$$

Condition 1:

$$|\mu_{3,2}| = 1 \leq \frac{1}{2} \quad \times$$

$$\mu_{3,1} = \frac{\langle b_3, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} = \frac{(0,1,1)(1,1,0)}{1} = 0$$

Condition 1:

$$|\mu_{3,1}| = 0 \leq \frac{1}{2} \quad \checkmark$$

Reduce:

$$r = \left\lfloor \frac{1}{2} + \mu_{3,2} \right\rfloor = \left\lfloor \frac{1}{2} + 1 \right\rfloor = 1$$

$$b_3 = b_3 - r b_2 = (0,1,1) - 1(0,0,1) = (0,1,0)$$

$$\mu_{3,2} = \mu_{3,2} - r = 1 - 1 = 0$$

$$\mu_{3,1} = \mu_{3,1} - r\mu_{2,1} = 0 - 1.0 = 0$$

New basis: $b_1 = (1,0,0), b_2 = (0,0,1), b_3 = (0,1,0)$

$$b_3^* = b_3 - \mu_{3,2} b_2^* - \mu_{3,1} b_1^* = (0,1,0), \quad B_3 = 1$$

Condition 2:

$$B_3 \geq (\frac{3}{4} - \mu_{3,2}^2) B_2$$

$$1 \geq \frac{3}{4} \quad \checkmark$$

Reduced Basis: $b_1 = (1,0,0), b_2 = (0,0,1), b_3 = (0,1,0)$

- 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 $\lambda_1(L)$.

BKZ Algorithm

Assume that we have given

Basis :

b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	…	b_n

Svp oracle(For small number of vectors) :

İnput : $b_1, b_2, \dots, b_n$

Finds shortest vector of Lattice (a_1) genarated by baseses

Returns : $a_1, \underbrace{b_i, b_j, \dots, b_k}_{\text{n-1}}$

BKZ Algorithm

b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	...	b_n

Apply LLL

a_1	?	?	?	?	?	?	?	...	?

Apply SVP oracle

c_1	?	?	?	?	?	?	?	...	?

Apply LLL

d_1	?	?	?	?	?	?	?	...	?

Apply SVP oracle

d_1	e_1	?	?	?	?	?	?	...	?

Apply LLL

f_1	g_2	?	?	?	?	?	?	...	?

Apply SVP oracle

$\vdots$

BKZ Algorithm

Algorithm 1 BKZ reduction (as given by Chen and Nguyen [2])

Input: A basis $B = (b_1, \ldots, b_n)$, the blocksize $\beta \in \{2, \ldots, n\}$, the Gram-Schmidt triangular matrix μ and $\|b_1^*\|^2, \ldots, \|b_n^*\|^2$

Output: A BKZ-β reduced basis for $L(B)$

$z \leftarrow 0$

$j \leftarrow 0$

$LLL(b_1, \ldots, b_n, \mu)$

while $z < n - 1$ **do**

    $j \leftarrow (j \bmod (n-1)) + 1$

    $k \leftarrow \min(j + \beta - 1, n)$

    $h \leftarrow \min(k + 1, n)$

    $v \leftarrow Enum(\mu_{[j,k]}, \|b_j^*\|^2, \ldots, \|b_k^*\|^2)$

    if $v \neq (1, 0, \ldots, 0)$ **then**

        $z \leftarrow 0$

        $LLL(b_1, \ldots, \sum_{i=j}^{k} v_i b_i, b_j, \ldots, b_h, \mu)$ at stage j

    else

        $z \leftarrow z + 1$

        $LLL(b_1, \ldots, b_h, \mu)$ at stage $h - 1$

    end if

end while

SIS and LWE

Given many uniform a_i , find small $z_1, z_2 \dots z_m \in \mathbb{Z}$ so that

$$z_1 \cdot \begin{pmatrix} | \\ \mathbf{a}_1 \\ | \end{pmatrix} + z_2 \cdot \begin{pmatrix} | \\ \mathbf{a}_2 \\ | \end{pmatrix} + \cdots + z_m \cdot \begin{pmatrix} | \\ \mathbf{a}_m \\ | \end{pmatrix} = \begin{pmatrix} | \\ 0 \\ | \end{pmatrix} \in \mathbb{Z}_q^n$$

OR

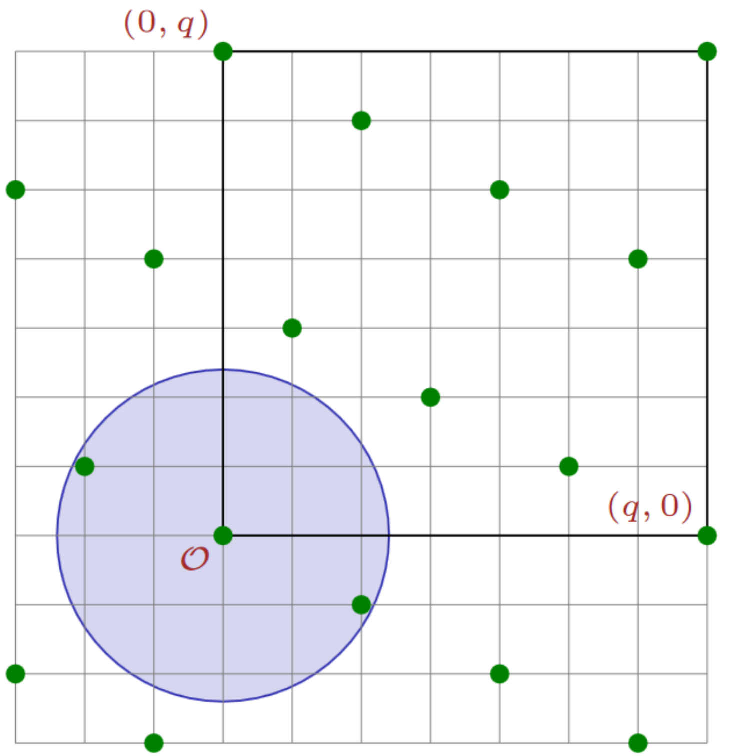
$$\underbrace{\begin{pmatrix} \cdots\cdot & \mathbf{A} & \cdots\cdot \end{pmatrix}}_{m} \begin{pmatrix} \\ \mathbf{z} \\ \\ \end{pmatrix} = \mathbf{0} \in \mathbb{Z}_q^n$$

As a Lattice problem

- Given matrix A= $(a_1, a_2 \ldots a_m) \in \mathbb{Z}_q^{n*m}$

$$\mathcal{L}^{\perp}(\mathrm{A}) = \{\mathrm{z} \in \mathbb{Z}^{m} : \mathrm{Az} = 0 \ \}$$

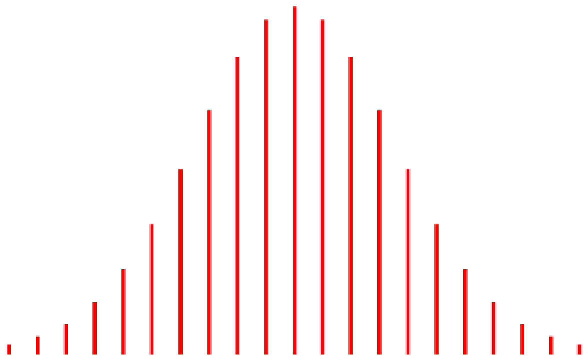
- SIS asks: short solutions z lies inside



SEARCH: Find secret s ϵ $\mathbb{Z}_q^n$ given many noisy inner products

$$\mathbf{a}_1 \leftarrow \mathbb{Z}_q^n \quad , \quad b_1 = \langle \mathbf{s} \, , \, \mathbf{a}_1 \rangle + e_1 \in \mathbb{Z}_q$$
$$\mathbf{a}_2 \leftarrow \mathbb{Z}_q^n \quad , \quad b_2 = \langle \mathbf{s} \, , \, \mathbf{a}_2 \rangle + e_2 \in \mathbb{Z}_q$$
$$\vdots$$

e_i's comes from Gaussian like distributions.



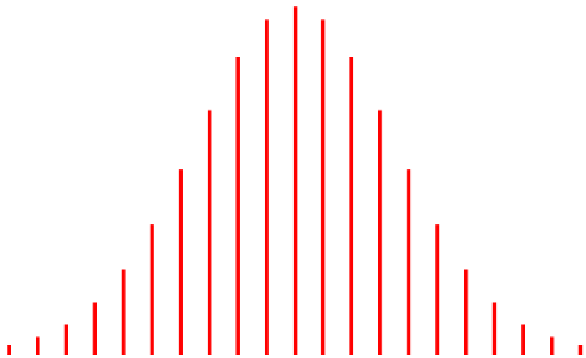
SIS and LWE

SEARCH: Find secret s $\epsilon\ \mathbb{Z}_q^n$ given many noisy inner products

$$\begin{pmatrix} \cdots & \mathbf{A} & \cdots \end{pmatrix}, \quad \begin{pmatrix} \cdots & \mathbf{b}^t & \cdots \end{pmatrix} = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t$$

Given (Public)

e_i's comes from Gaussian like distributions.

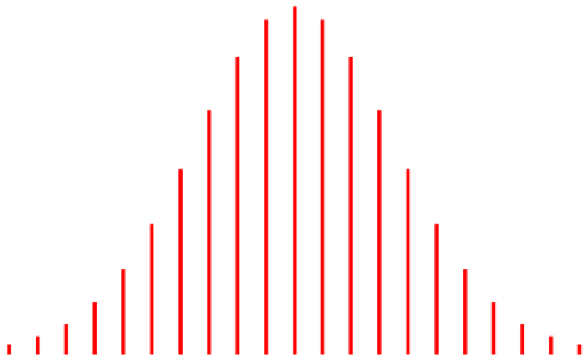


DECİSİON: Distinguish **(A , b)** from uniform **(A , b)**

No error, just uniformly random values

$$\begin{pmatrix} \cdots & \mathbf{A} & \cdots \\ \end{pmatrix} \quad , \quad \begin{pmatrix} \cdots & \mathbf{b}^t & \cdots \end{pmatrix} = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t$$

e_i's comes from Gaussian like distributions.



As a Lattice problem

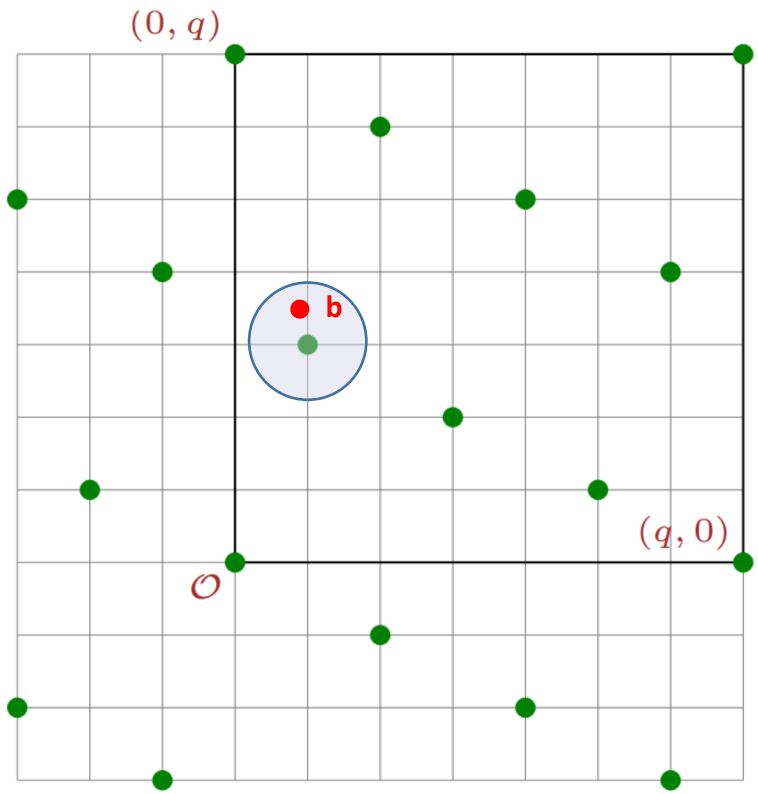
LWE Lattice:

$$\mathcal{L}(A) = \{ z \in \mathbb{Z}^m : z \equiv s * A \bmod q \}$$

LWE is bounded-dist decoding on $\mathcal{L}(A)$:

given $b^t = z^t + e = s^t * A + e$,

find z^t



LWE applications

Public-Key Cryptosystem from LWE (1 bit) (Lindler-Peikert)

$$A \longleftarrow \mathbb{Z}_q^{n*m}$$

(Public)

Choose short int. vector x
(Private key)

$$\mathrm{s} \longleftarrow \mathbb{Z}_q^{n}$$

u = Ax (SIS problem)

(Public key)

$$b^t = s^t A + e^t$$

(Ciphertext)

$$b' = s^t u + e' + bit * q/2$$

(Payload)

b'- b^tx

$$= s^t u - s^t A x - e^t x + e' + bit * q/2$$

X X

much smaller
than q/2

Bit can be obtained

Public-Key Cryptosystem from LWE (Regev 2005)

$$a_1, \ldots, a_m \longleftarrow \mathbb{Z}_p^n$$

(Public)

Choose uniformly rondom s $\in \mathbb{Z}_p^n$
(Private key)

Choose error e from distrubition.

$$S \longleftarrow \{0,1\}^m$$

$(\mathbf{a}_i, b_i)_{i=1}^m$ where $b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i$.

(Public key)

$(\sum_{i\in S} \mathbf{a}_i, \sum_{i\in S} b_i)$ if the bit is 0

$(\sum_{i\in S} \mathbf{a}_i, \lfloor \frac{p}{2} \rfloor + \sum_{i\in S} b_i)$ if the bit is 1

(Ciphertext)

$$\begin{aligned} b - < a, s > &= < a, s > + e - < a, s > \\ &= e \end{aligned}$$

Plaintext bit is 0 if e closer to 0 than to $\lfloor \frac{p}{2} \rfloor$ modulo p

Plaintext bit is 1 otherwise

Basic LWE key agreement

Based on Lindner–Peikert LWE public key encryption scheme

public: "big" A in $\mathbf{Z}_q^{n \times m}$

Alice

secret:
random "small" s, e in $\mathbf{Z}_q^m$

$b = As + e \longrightarrow$

$\longleftarrow b' = s'A + e'$

shared secret:
$b's = s'As + e's \approx s'As$

Bob

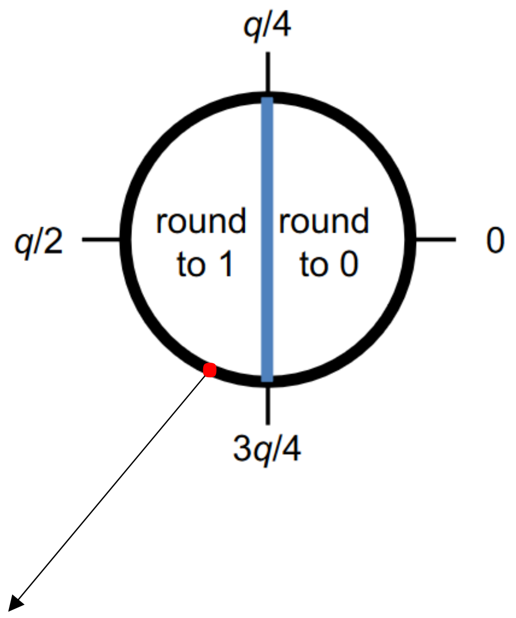
secret:
random "small" s', e' in $\mathbf{Z}_q^n$

shared secret:
$s'b \approx s'As$

These are only approximately equal ⇒ need rounding

Basic LWE key agreement

Treat each coefficient independently

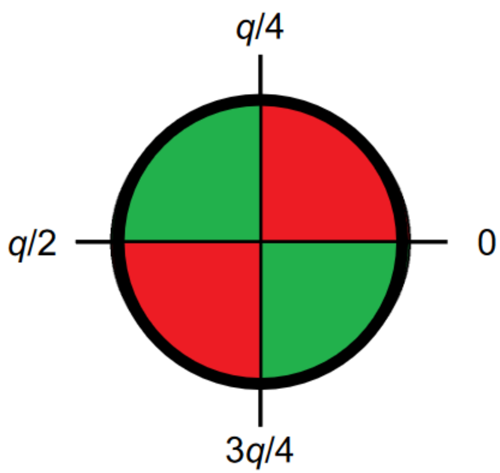


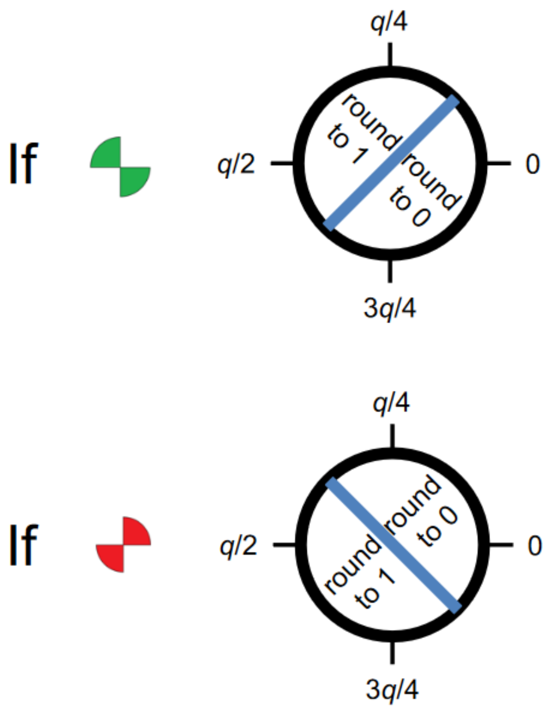
$s'As + s'e$

Not always Works !

Basic LWE key agreement

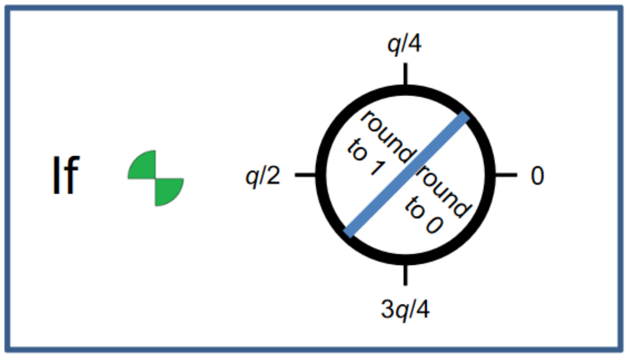
Bob says which of two regions the value is in:  or 

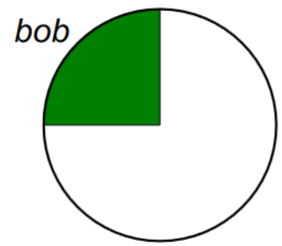


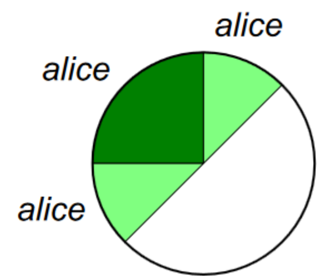


Basic LWE key agreement

If $b'\mathrm{s} - s'b = s'\ As + e's - s'As\ - s'e$
$= e's - s'e < \frac{q}{8}$ then this always works.







- Security not affected: revealing  or  leaks no information

Basic LWE key agreement

public: "big" A in $\mathbf{Z}_q^{n \times m}$

Alice

secret:
random "small" s, e in $\mathbf{Z}_q^{m}$

Bob

secret:
random "small" s', e' in $\mathbf{Z}_q^{n}$

Alice → Bob: $b = As + e$

Bob → Alice: $b' = s'A + e'$,  or 

Alice: shared secret:
round($b's$)

Bob: shared secret:
round($s'b$)

References

1. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. J. ACM,56(6):1–40, 2009. Preliminary version in STOC 2005.

2. C. Peikert & R.Lindner., (2010) Better Key Sizes (and Attacks) for LWE-Based Encryption –Cryptology ePrint Archive, Report 2010/613 . https://ia.cr/2010/613

3. Peikert Lecture noteshttps://web.eecs.umich.edu/~cpeikert/lic15/