Lattices in cryptography: cryptanalysis, constructions and reductions

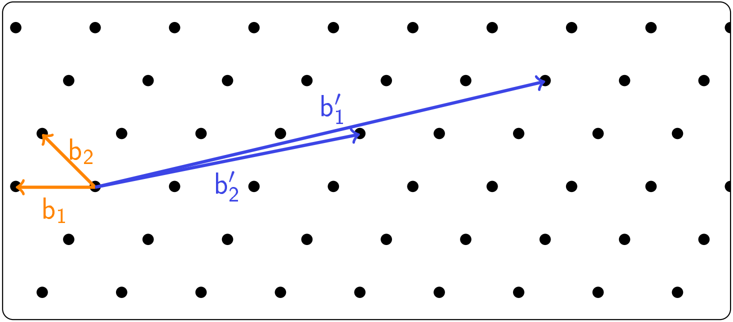
Alice Pellet--Mary

CNRS and Université de Bordeaux

Journées C2, 2023

Najac

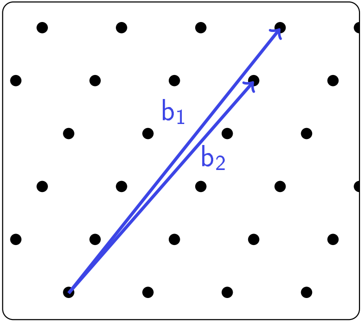
Lattices



- $\mathcal{L} = \{\sum_{i=1}^{n} x_i \mathrm{b}_i \mid \forall i,\, x_i \in \mathbb{Z}\}$ is a lattice
- $(\mathrm{b}_1, \ldots, \mathrm{b}_n) =: B \in \mathrm{GL}_n(\mathbb{R})$ is a basis (not unique)

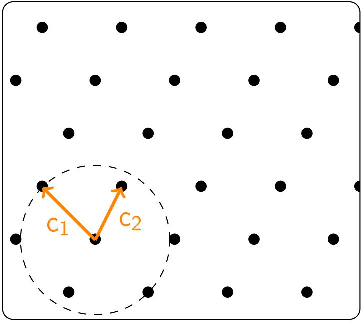
Short basis problem

Input:



$\longrightarrow$

Output:

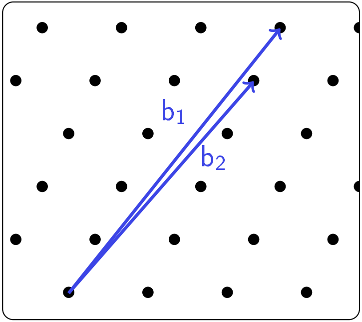


Shortest basis problem

$$\max_i \|c_i\| \leq \min_{B' \text{ basis of } L} \left(\max_i \|b'_i\| \right)$$

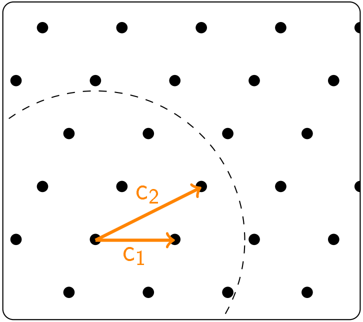
Short basis problem

Input:



$\longrightarrow$

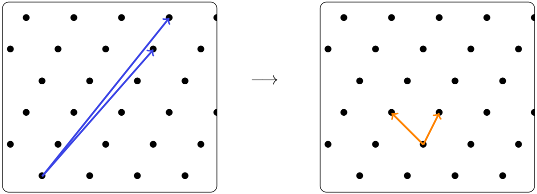
Output:



Approximate short basis problem

$$\max_i \|c_i\| \leq \gamma \cdot \min_{B' \text{ basis of } L} \left(\max_i \|b'_i\| \right)$$

Lattice reduction algorithms



Dimension 2: Lagrange-Gauss algorithm

video

Dimension 2: Lagrange-Gauss algorithm

video

Theorem: The algorithm

- finds a shortest basis
- runs in polynomial time

The LLL algorithm [LLL82]

Input: basis $B = (b_1, \ldots, b_n)$

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (\mathrm{b}_1, \ldots, \mathrm{b}_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

Algorithm:

- while there exists i such that (b_i, b_{i+1}) is not a shortest basis of L_i
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
    - run Lagrange-Gauss on L_i

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (\mathrm{b}_1, \ldots, \mathrm{b}_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

Algorithm:

- while there exists i such that (b_i, b_{i+1}) is not a shortest basis of L_i
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- finds an approximate short basis with $\gamma = 2^n$

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (\mathrm{b}_1, \ldots, \mathrm{b}_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

Algorithm:

- while there exists i such that (b_i, b_{i+1}) is not a shortest basis of L_i
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- finds an approximate short basis with $\gamma = 2^n$
- does not run in polynomial time

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

The LLL algorithm [LLL82]

Input: basis $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$

Main idea: improve the basis locally on blocks of dimension 2
(using Lagrange-Gauss algorithm)

Algorithm:

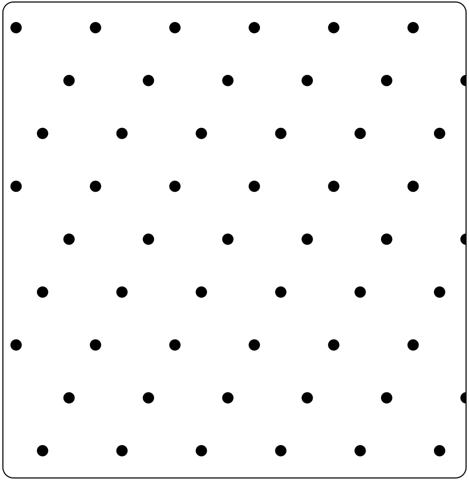
- while there exists i such that (b_i, b_{i+1}) is not a γ'-short basis of L_i with $\gamma' = 4/3$
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- finds an approximate short basis with $\gamma = 2^n$
- runs in polynomial time

[LLL82] Lenstra, Lenstra, and Lovász. Factoring polynomials with rational coefficients. Mathematische annalen.

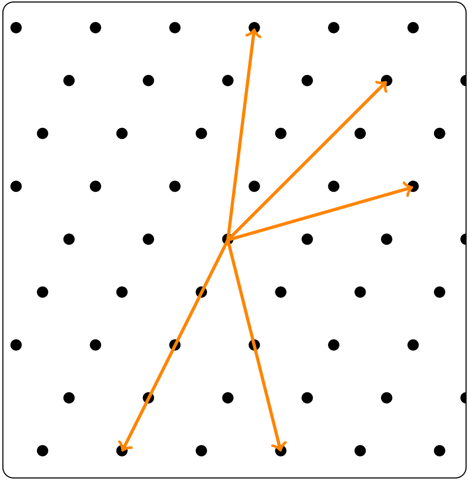
Sieving algorithm [AKS01]



Sieving:

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

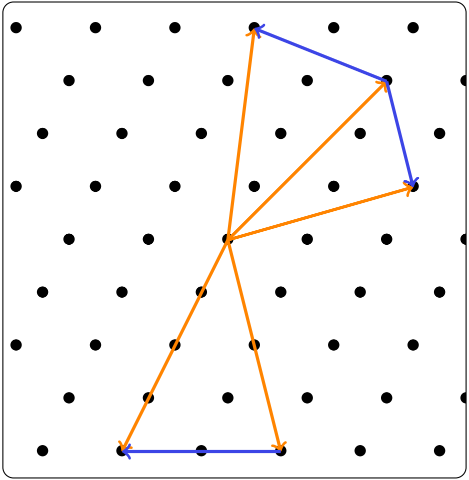


Sieving:

- Create many large vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

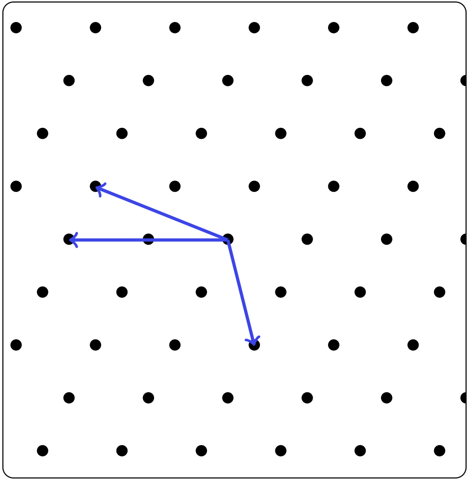


Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

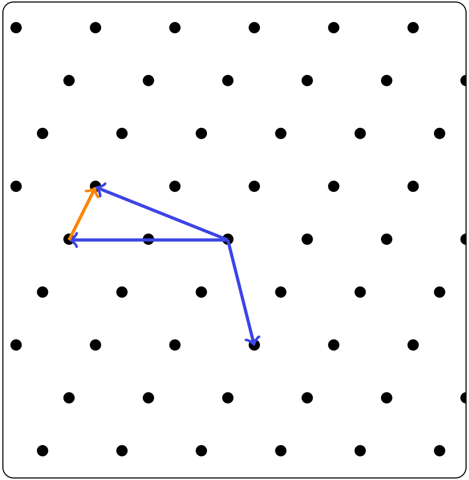


Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

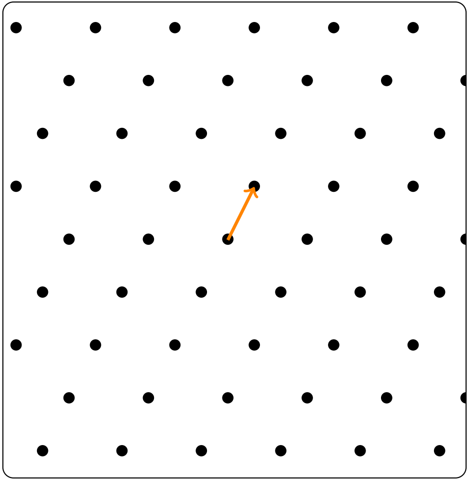


Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]

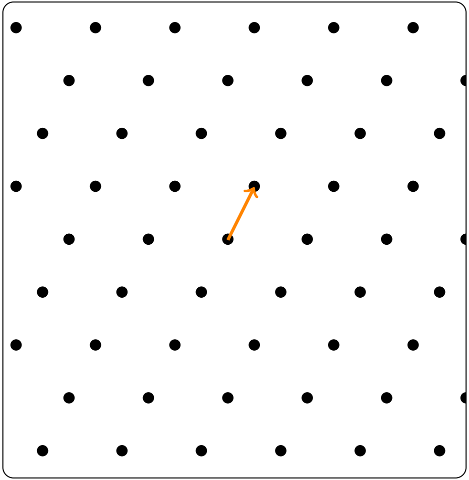


Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]



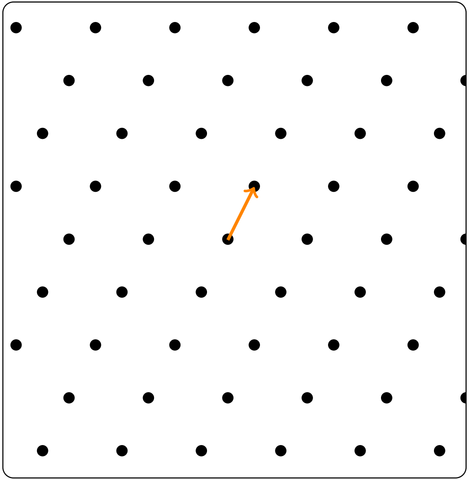
Sieving:

- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

Size of the initial list: $2^{O(n)}$

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]



Sieving:

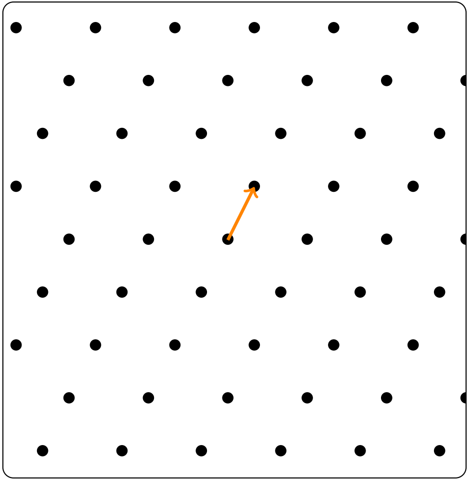
- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

Size of the initial list: $2^{O(n)}$

- finds a shortest basis

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Sieving algorithm [AKS01]



Sieving:

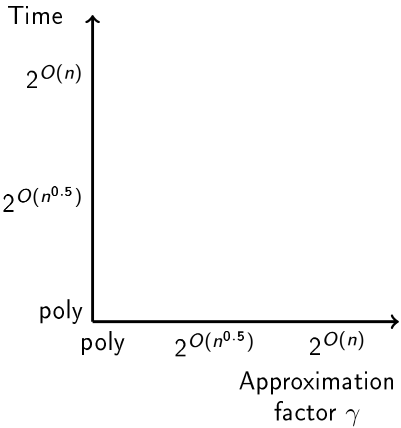
- Create many large vectors
- Subtract close ones to create shorter vectors
- Repeat with the shorter vectors

Size of the initial list: $2^{O(n)}$

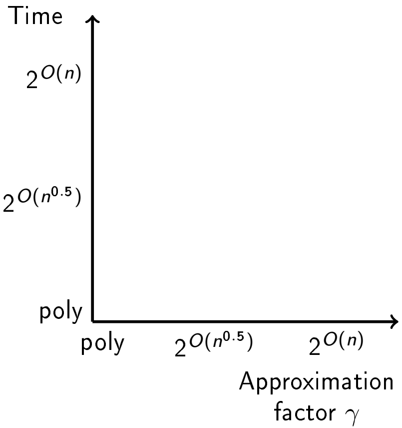
- finds a shortest basis
- runs in time $2^{O(n)}$

[AKS01] Ajtai, Kumar, and Sivakumar. A sieve algorithm for the shortest lattice vector problem. STOC

Summary and BKZ

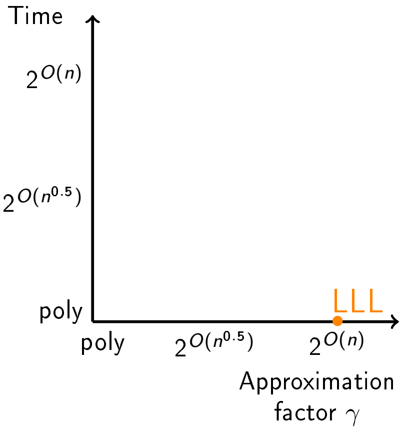


Summary and BKZ



Lagrange-Gauss algorithm: dim 2

- shortest basis
- polynomial time



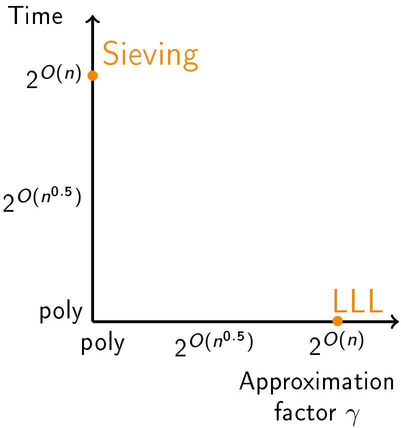
Lagrange-Gauss algorithm: dim 2

- shortest basis
- polynomial time

LLL algorithm: dim n

- γ-short basis with $\gamma = 2^n$
- polynomial time

Summary and BKZ



Lagrange-Gauss algorithm: dim 2

- shortest basis
- polynomial time

LLL algorithm: dim n

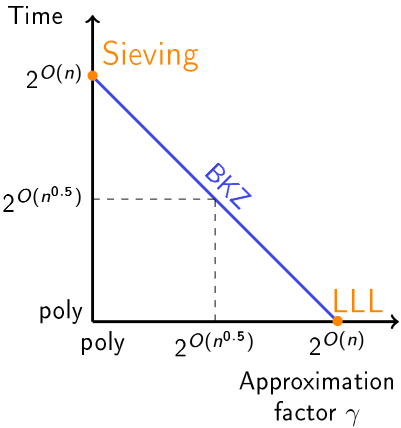
- γ-short basis with $\gamma = 2^n$
- polynomial time

Sieving algorithm: dim n

- shortest basis
- time $2^{O(n)}$

Summary and BKZ

BKZ trade-offs



Lagrange-Gauss algorithm: dim 2

- shortest basis
- polynomial time

LLL algorithm: dim n

- γ-short basis with $\gamma = 2^n$
- polynomial time

Sieving algorithm: dim n

- shortest basis
- time $2^{O(n)}$

BKZ algorithm: combine LLL + Sieving $\Rightarrow$ various trade-offs

Some concrete numbers

Finding a shortest basis in practice:

- $n = 2 \rightsquigarrow$ easy, very efficient in practice

Some concrete numbers

Finding a shortest basis in practice:

- $n = 2 \rightsquigarrow$ easy, very efficient in practice
- up to $n = 60$ or $n = 80 \rightsquigarrow$ a few minutes on a personal laptop

Some concrete numbers

Finding a shortest basis in practice:

- $n = 2 \rightsquigarrow$ easy, very efficient in practice
- up to $n = 60$ or $n = 80 \rightsquigarrow$ a few minutes on a personal laptop
- up to $n = 180 \rightsquigarrow$ few days on big computers with good code [DSW21]

[DSW21] Ducas, Stevens, van Woerden. Advanced Lattice Sieving on GPUs, with Tensor Cores.

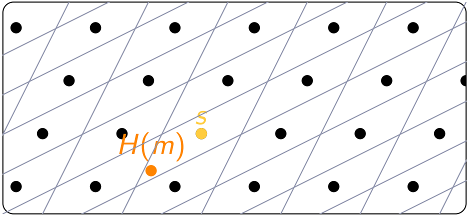
Some concrete numbers

Finding a shortest basis in practice:

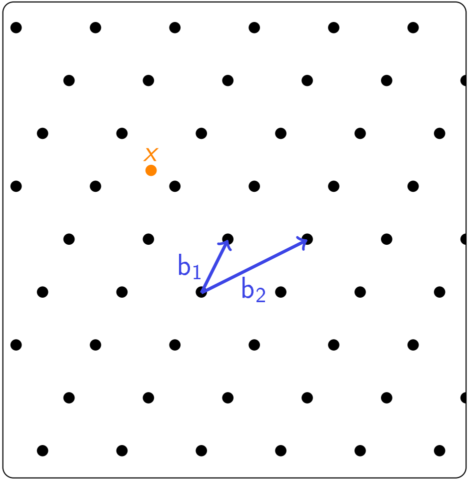
- $n = 2 \rightsquigarrow$ easy, very efficient in practice
- up to $n = 60$ or $n = 80 \rightsquigarrow$ a few minutes on a personal laptop
- up to $n = 180 \rightsquigarrow$ few days on big computers with good code [DSW21]
- from $n = 500$ to $n = 1000 \rightsquigarrow$ cryptography

[DSW21] Ducas, Stevens, van Woerden. Advanced Lattice Sieving on GPUs, with Tensor Cores.

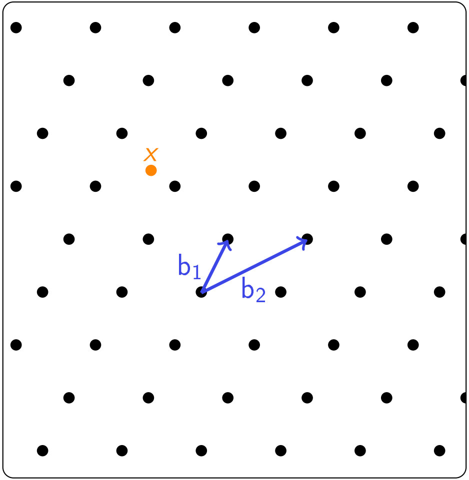
Hash-and-sign signature



Decoding in a lattice using a short basis



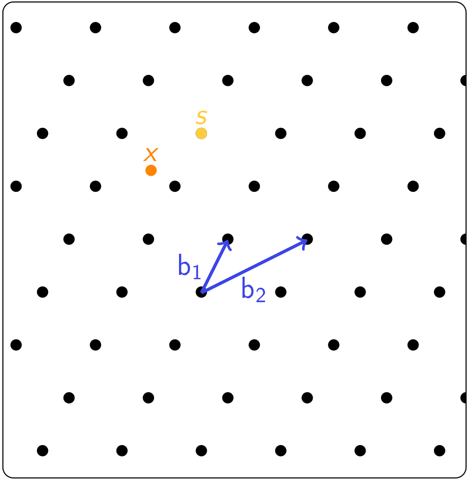
Input: $x = 3.7 \cdot \mathrm{b}_1 - 1.4 \cdot \mathrm{b}_2$



Input: $x = 3.7 \cdot b_1 - 1.4 \cdot b_2$

Algo: round each coordinate

Decoding in a lattice using a short basis

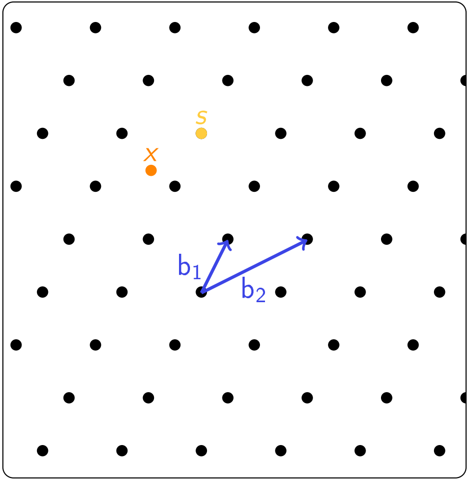


Input: $x = 3.7 \cdot \mathrm{b}_1 - 1.4 \cdot \mathrm{b}_2$

Algo: round each coordinate

Output: $s = 4 \cdot \mathrm{b}_1 - 1 \cdot \mathrm{b}_2$

Decoding in a lattice using a short basis



Input: $x = 3.7 \cdot b_1 - 1.4 \cdot b_2$

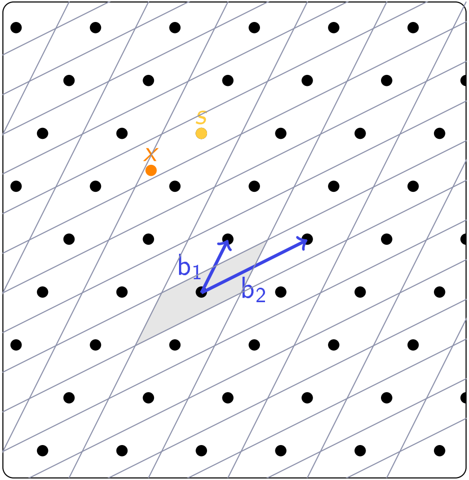
Algo: round each coordinate

Output: $s = 4 \cdot b_1 - 1 \cdot b_2$

The smaller the basis, the closer the solution

(called Babai's round-off algorithm)

Decoding in a lattice using a short basis



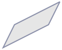
Input: $x = 3.7 \cdot b_1 - 1.4 \cdot b_2$

Algo: round each coordinate

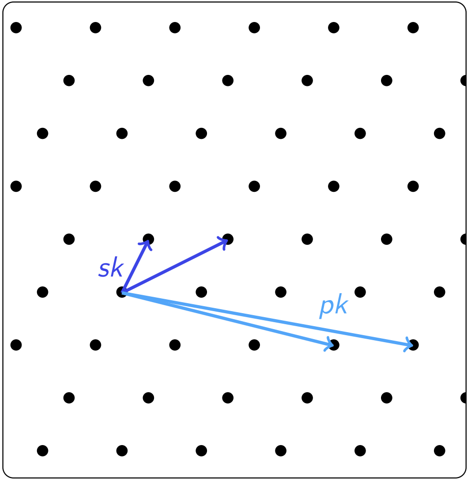
Output: $s = 4 \cdot b_1 - 1 \cdot b_2$

The smaller the basis, the closer the solution

(called Babai's round-off algorithm)

 $= \left\{ x_1 b_1 + x_2 b_2 \,\middle|\, |x_i| \leq \frac{1}{2} \right\}$

Hash-and-sign: first idea [GGH97]

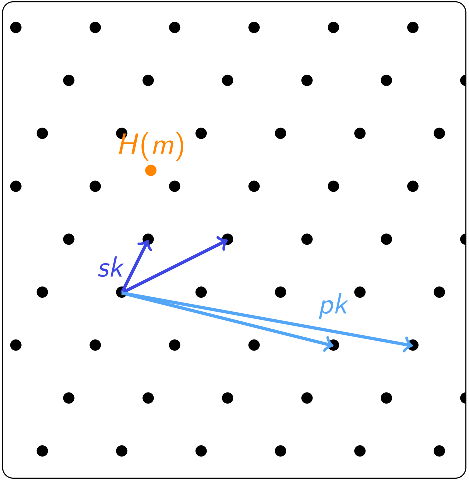


KeyGen:

- pk = bad basis of $\mathcal{L}$
- sk = short basis of $\mathcal{L}$

[GGH97] Goldreich, Goldwasser, and Halevi. Public-key cryptosystems from lattice reduction problems. CRYPTO

Hash-and-sign: first idea [GGH97]



KeyGen:

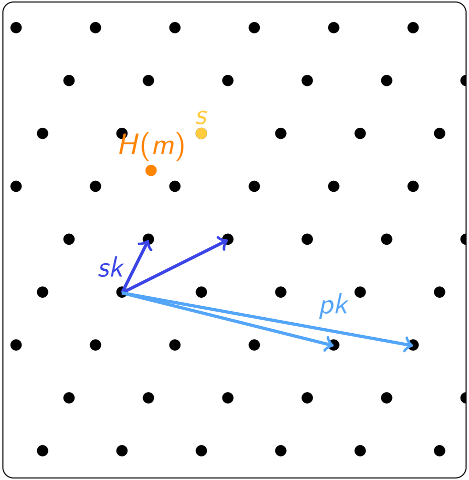
- pk = bad basis of $\mathcal{L}$
- sk = short basis of $\mathcal{L}$

Sign(m, sk):

- $x = H(m)$ (hash the message)

[GGH97] Goldreich, Goldwasser, and Halevi. Public-key cryptosystems from lattice reduction problems. CRYPTO

Hash-and-sign: first idea [GGH97]



KeyGen:

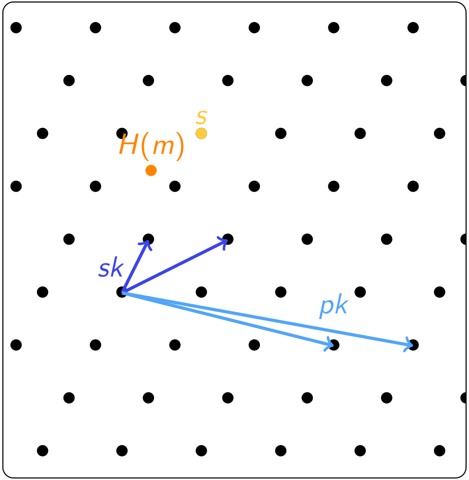
- pk = bad basis of $\mathcal{L}$
- sk = short basis of $\mathcal{L}$

Sign(m, sk):

- $x = H(m)$ (hash the message)
- output $s \in \mathcal{L}$ close to x

[GGH97] Goldreich, Goldwasser, and Halevi. Public-key cryptosystems from lattice reduction problems. CRYPTO

Hash-and-sign: first idea [GGH97]



KeyGen:

- pk = bad basis of $\mathcal{L}$
- sk = short basis of $\mathcal{L}$

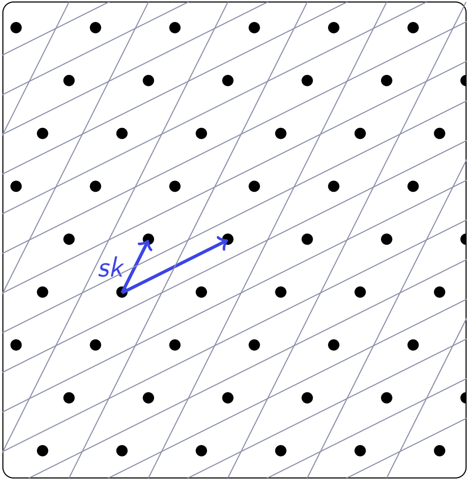
Sign(m, sk):

- $x = H(m)$ (hash the message)
- output $s \in \mathcal{L}$ close to x

Verify(s, pk):

- check that $s \in \mathcal{L}$
- check that $H(m) - s$ is small

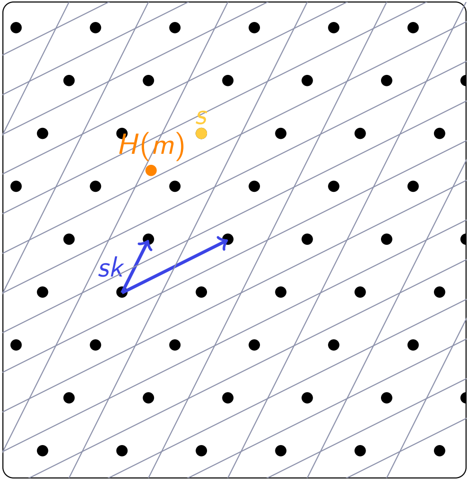
[GGH97] Goldreich, Goldwasser, and Halevi. Public-key cryptosystems from lattice reduction problems. CRYPTO



Parallelepiped attack:

[NR06] Nguyen and Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J. Cryptology

Attack on this first idea [NR06]

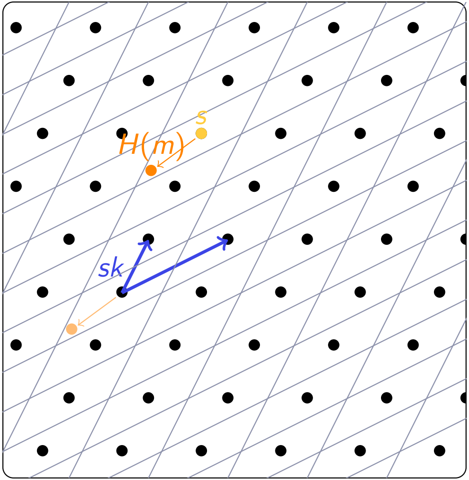


Parallelepiped attack:

- ask for a signature s on m

[NR06] Nguyen and Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J. Cryptology

Attack on this first idea [NR06]

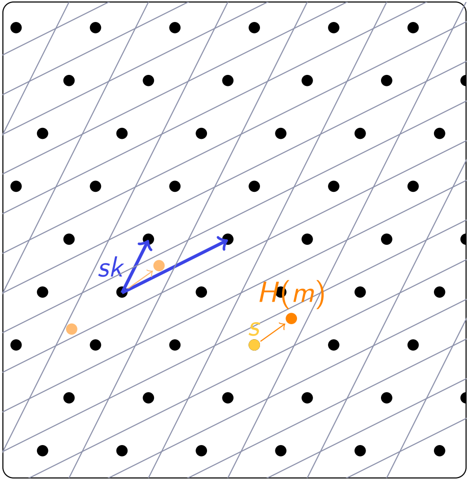


Parallelepiped attack:

- ask for a signature s on m
- plot $H(m) - s$

[NR06] Nguyen and Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J. Cryptology

Attack on this first idea [NR06]

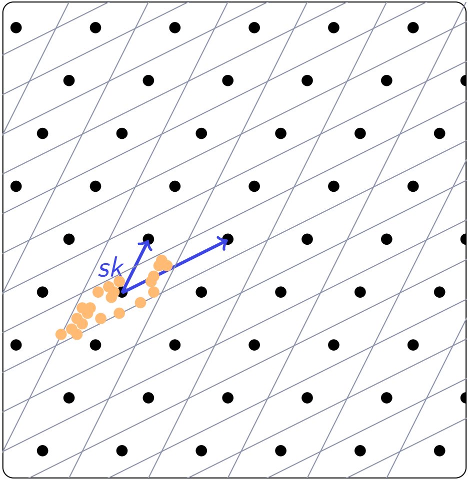


Parallelepiped attack:

- ask for a signature s on m
- plot $H(m) - s$
- repeat

[NR06] Nguyen and Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J. Cryptology

Attack on this first idea [NR06]

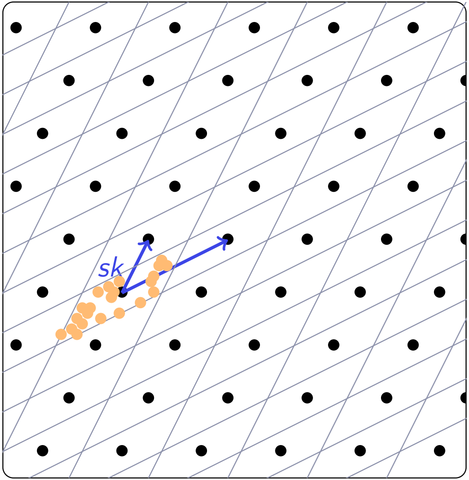


Parallelepiped attack:

- ask for a signature s on m
- plot $H(m) - s$
- repeat

[NR06] Nguyen and Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J. Cryptology

Attack on this first idea [NR06]



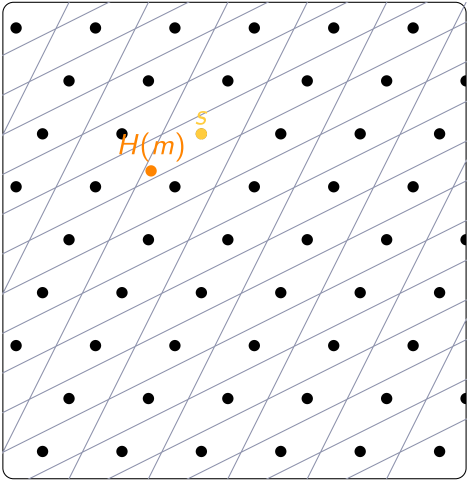
Parallelepiped attack:

- ask for a signature s on m
- plot $H(m) - s$
- repeat

From the shape of the parallelepiped, one can recover the short basis

[NR06] Nguyen and Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J. Cryptology

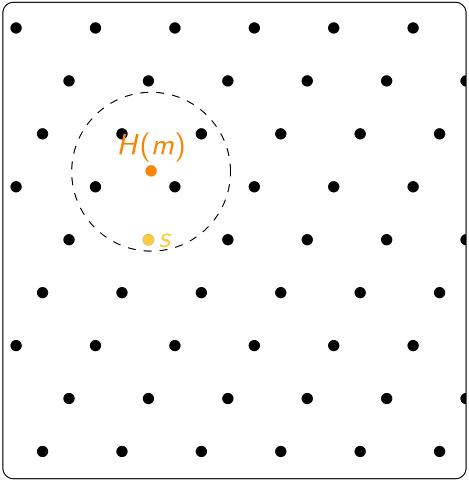
Preventing the attack [GPV08]



Idea: do not decode deterministically but randomly

[GPV08] Gentry, Peikert, and Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC.

Preventing the attack [GPV08]



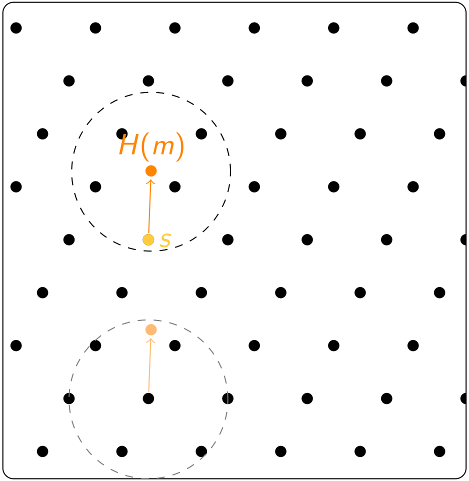
Idea: do not decode deterministically but randomly

Sign(m, sk):

- $x = H(m)$ (hash the message)
- sample $s \in \mathcal{L} \cap \mathcal{B}_r(x)$ (small radius r)

[GPV08] Gentry, Peikert, and Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC.

Preventing the attack [GPV08]



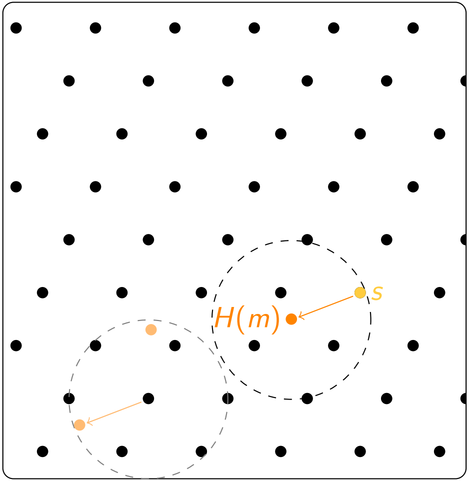
Idea: do not decode deterministically but randomly

Sign(m, sk):

- $x = H(m)$ (hash the message)
- sample $s \in \mathcal{L} \cap \mathcal{B}_r(x)$ (small radius r)

[GPV08] Gentry, Peikert, and Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC.

Preventing the attack [GPV08]



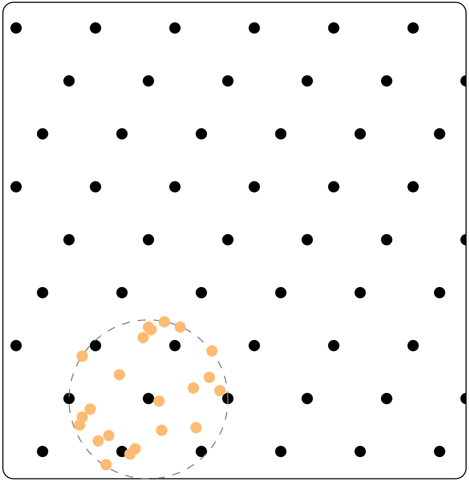
Idea: do not decode deterministically but randomly

Sign(m, sk):

- $x = H(m)$ (hash the message)
- sample $s \in \mathcal{L} \cap \mathcal{B}_r(x)$ (small radius r)

[GPV08] Gentry, Peikert, and Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC.

Preventing the attack [GPV08]



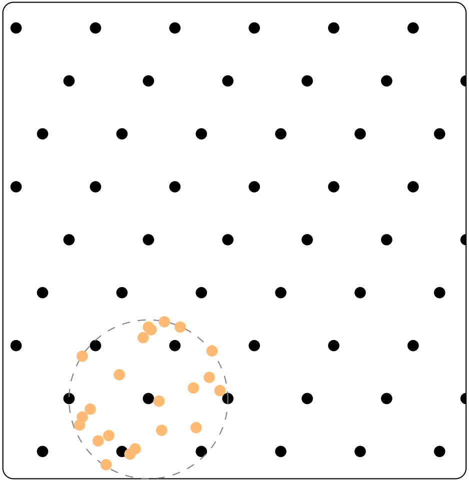
Idea: do not decode deterministically but randomly

Sign(m, sk):

- $x = H(m)$ (hash the message)
- sample $s \in \mathcal{L} \cap \mathcal{B}_r(x)$ (small radius r)

[GPV08] Gentry, Peikert, and Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC.

Preventing the attack [GPV08]



Idea: do not decode deterministically but randomly

Sign(m, sk):

- $x = H(m)$ (hash the message)
- sample $s \in \mathcal{L} \cap \mathcal{B}_r(x)$ (small radius r)

Lemma: if an adversary can forge signatures, then she can recover a short basis of $\mathcal{L}$ using only pk (in the ROM)

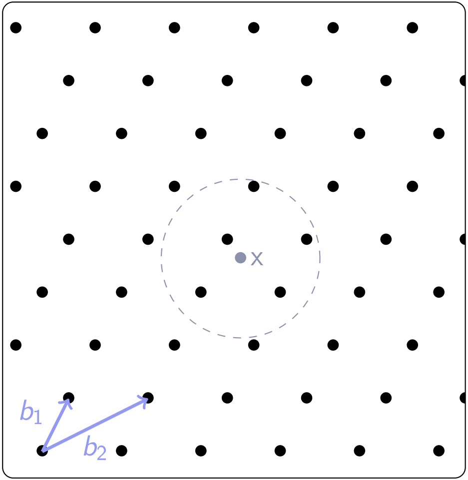
[GPV08] Gentry, Peikert, and Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC.

Sampling uniformly in a ball [PP21]

Input: center x, radius r

(and a short basis $(b_1, \ldots, b_n)$)

Output: $s \leftarrow \mathcal{U}\left(\mathcal{L} \cap \mathcal{B}_r(x)\right)$



[PP21] Plançon and Prest. Exact Lattice Sampling from Non-Gaussian Distributions. PKC.

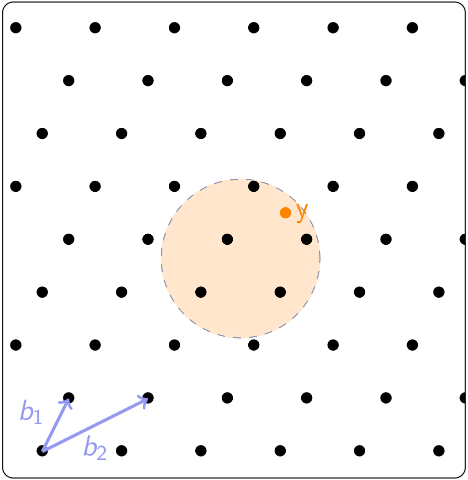
Input: center x, radius r

(and a short basis $(b_1, \ldots, b_n)$)

Output: $s \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r(x))$

Algo:

- Sample $y \leftarrow \mathcal{U}(\mathcal{B}_r(x))$
  (continuous distribution)



[PP21] Plançon and Prest. Exact Lattice Sampling from Non-Gaussian Distributions. PKC.

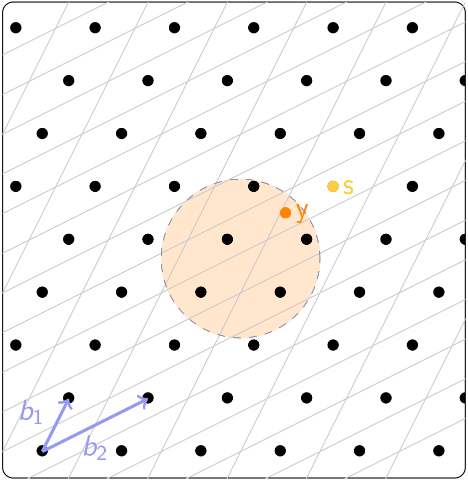
Input: center x, radius r

(and a short basis $(b_1, \ldots, b_n)$)

Output: $s \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r(x))$

Algo:

- Sample $y \leftarrow \mathcal{U}(\mathcal{B}_r(x))$
  (continuous distribution)
- s ← Babai_decoding(y)



[PP21] Plançon and Prest. Exact Lattice Sampling from Non-Gaussian Distributions. PKC.

Sampling uniformly in a ball [PP21]

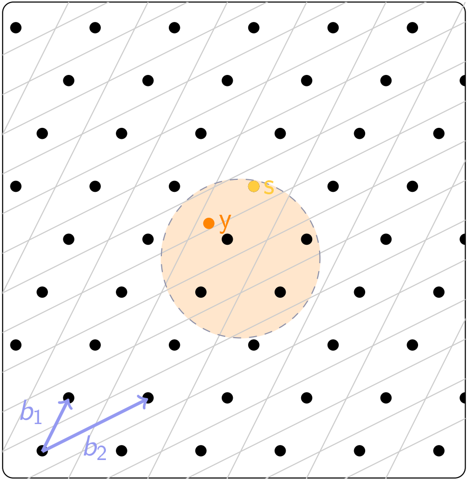
Input: center x, radius r

(and a short basis $(b_1, \ldots, b_n)$)

Output: $s \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r(x))$

Algo:

- Sample $y \leftarrow \mathcal{U}(\mathcal{B}_r(x))$
  (continuous distribution)
- $s \leftarrow$ `Babai_decoding(y)`
- repeat until $s \in \mathcal{B}_r(x)$



[PP21] Plançon and Prest. Exact Lattice Sampling from Non-Gaussian Distributions. PKC.

Sampling uniformly in a ball [PP21]

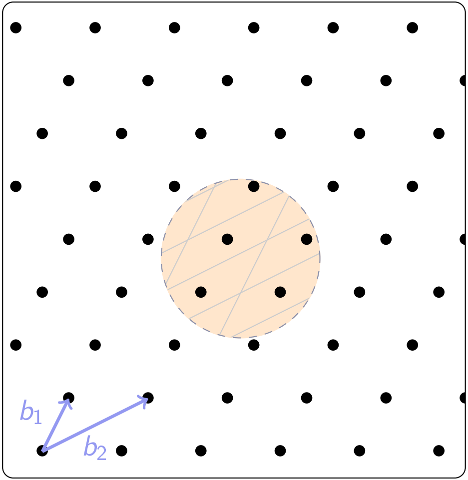
Input: center x, radius r

(and a short basis $(b_1, \ldots, b_n)$)

Output: $s \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r(x))$

Algo:

- Sample $y \leftarrow \mathcal{U}(\mathcal{B}_r(x))$
  (continuous distribution)
- $s \leftarrow$ `Babai_decoding(y)`
- repeat until $s \in \mathcal{B}_r(x)$



[PP21] Plançon and Prest. Exact Lattice Sampling from Non-Gaussian Distributions. PKC.

Sampling uniformly in a ball [PP21]

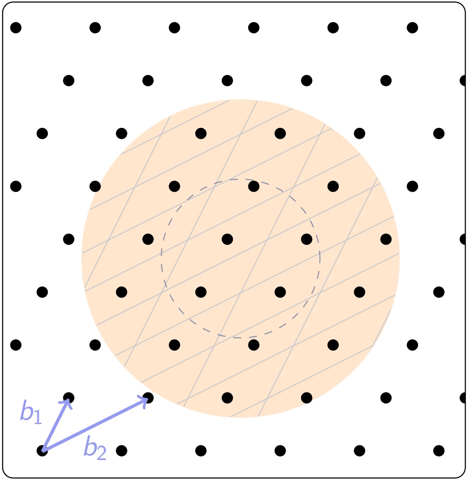
Input: center x, radius r

(and a short basis $(b_1, \ldots, b_n)$)

Output: $s \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r(x))$

Algo:

- Sample $y \leftarrow \mathcal{U}(\mathcal{B}_{r'}(x))$
  (continuous distribution)
- $s \leftarrow$ `Babai_decoding(y)`
- repeat until $s \in \mathcal{B}_r(x)$



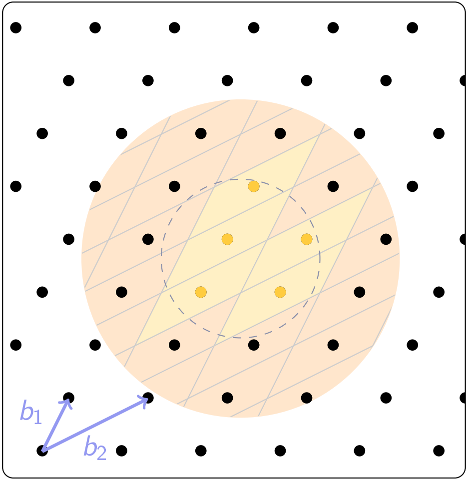
[PP21] Plançon and Prest. Exact Lattice Sampling from Non-Gaussian Distributions. PKC.

Sampling uniformly in a ball [PP21]

Input: center x, radius r
(and a short basis $(b_1, \ldots, b_n)$)
Output: $s \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r(x))$

Algo:

- Sample $y \leftarrow \mathcal{U}(\mathcal{B}_{r'}(x))$
  (continuous distribution)
- $s \leftarrow$ `Babai_decoding(y)`
- repeat until $s \in \mathcal{B}_r(x)$



[PP21] Plançon and Prest. Exact Lattice Sampling from Non-Gaussian Distributions. PKC.

Sampling uniformly in a ball [PP21]

Input: center x, radius r

(and a short basis $(b_1, \ldots, b_n)$)

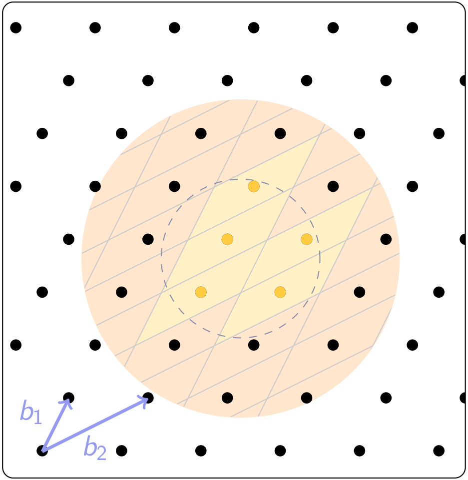
Output: $s \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r(x))$

Algo:

- Sample $y \leftarrow \mathcal{U}(\mathcal{B}_{r'}(x))$
  (continuous distribution)
- $s \leftarrow$ `Babai_decoding(y)`
- repeat until $s \in \mathcal{B}_r(x)$

polynomial time if

$r \geq 2n^2 \cdot \max_i \|b_i\|$



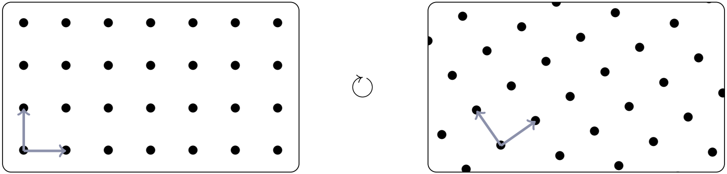
[PP21] Plançon and Prest. Exact Lattice Sampling from Non-Gaussian Distributions. PKC.

Summary

Hash-and-sign signature scheme:

- requires a lattice $\mathcal{L}$ + a short basis B_s + a bad basis B_p;
- provably secure if recovering a short basis from B_p is hard.

How to generate a hard lattice?



Objective

What we want: An algorithm `KeyGen` such that

- `KeyGen` computes
  - a random lattice $\mathcal{L}$
  - a short basis B_s of $\mathcal{L}$ (sk)
  - a bad basis B_p of $\mathcal{L}$ (pk)

Objective

What we want: An algorithm `KeyGen` such that

- `KeyGen` computes
  - a random lattice $\mathcal{L}$
  - a short basis B_s of $\mathcal{L}$ (sk)
  - a bad basis B_p of $\mathcal{L}$ (pk)
- computing a short basis of $\mathcal{L}$ from B_p is hard with overwhelming probability

Worst-possible basis

There is a basis B_0 of $\mathcal{L}$ that can be computed in poly time from any other basis B

$\Rightarrow$ B_0 is a worst possible basis

Worst-possible basis

There is a basis B_0 of $\mathcal{L}$ that can be computed
in poly time from any other basis B
$\Rightarrow B_0$ is a worst possible basis

Input: any basis B of $\mathcal{L}$

Algorithm:

- Compute LLL-reduced basis $C = (c_1, \ldots, c_n)$
  - poly time
  - $\max_i \|c_i\| \leq 2^n \cdot \min_{C'} \max_i \|c'_i\|$ (C' ranging over all bases of $\mathcal{L}$)

Worst-possible basis

There is a basis B_0 of $\mathcal{L}$ that can be computed
in poly time from any other basis B
$\Rightarrow$ B_0 is a worst possible basis

Input: any basis B of $\mathcal{L}$

Algorithm:

- Compute LLL-reduced basis $C = (c_1, \ldots, c_n)$
  - poly time
  - $\max_i \|c_i\| \leq 2^n \cdot \min_{C'} \max_i \|c'_i\|$ (C' ranging over all bases of $\mathcal{L}$)
- sample many vectors $v_j \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r)$ (with $r = 2n^2 \cdot 2^n \cdot \min_{C'} \max_i \|c'_i\|$)
  - until they generate $\mathcal{L}$
  - poly time because $r \geq 2n^2 \cdot \max_i \|c_i\|$

Worst-possible basis

There is a basis B_0 of $\mathcal{L}$ that can be computed in poly time from any other basis B

$\Rightarrow$ B_0 is a worst possible basis

Input: any basis B of $\mathcal{L}$

Algorithm:

- Compute LLL-reduced basis $C = (c_1, \ldots, c_n)$
  - poly time
  - $\max_i \|c_i\| \leq 2^n \cdot \min_{C'} \max_i \|c'_i\|$ (C' ranging over all bases of $\mathcal{L}$)
- sample many vectors $v_j \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r)$ (with $r = 2n^2 \cdot 2^n \cdot \min_{C'} \max_i \|c'_i\|$)
  - until they generate $\mathcal{L}$
  - poly time because $r \geq 2n^2 \cdot \max_i \|c_i\|$
- extract a basis B_0 from the v_j's
  - linear algebra $\Rightarrow$ poly time

Worst-possible basis

There is a random basis B_0 of $\mathcal{L}$ that can be computed in poly time from any other basis B

$\Rightarrow$ B_0 is a worst possible distribution over bases

Input: any basis B of $\mathcal{L}$

Algorithm:

- Compute LLL-reduced basis $C = (c_1, \ldots, c_n)$
  - poly time
  - $\max_i \|c_i\| \leq 2^n \cdot \min_{C'} \max_i \|c'_i\|$ (C' ranging over all bases of $\mathcal{L}$)
- sample many vectors $v_j \leftarrow \mathcal{U}(\mathcal{L} \cap \mathcal{B}_r)$ (with $r = 2n^2 \cdot 2^n \cdot \min_{C'} \max_i \|c'_i\|$)
  - until they generate $\mathcal{L}$
  - poly time because $r \geq 2n^2 \cdot \max_i \|c_i\|$
- extract a basis B_0 from the v_j's
  - linear algebra $\Rightarrow$ poly time

Objective

What we want: An algorithm `KeyGen` such that

- `KeyGen` computes
  - a random lattice $\mathcal{L}$
  - a short basis B_s of $\mathcal{L}$ (sk)
  - a bad basis B_p of $\mathcal{L}$ (pk)

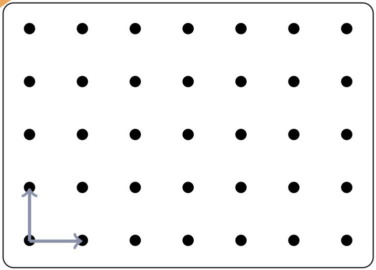
- computing a short basis from B_p is hard with overwhelming probability

Objective

What we want: An algorithm `KeyGen` such that

- `KeyGen` computes
  - a random lattice $\mathcal{L}$
  - a short basis B_s of $\mathcal{L}$ (sk)
  - a worst possible basis B_p of $\mathcal{L}$ (pk)
- computing a short basis from B_p is hard with overwhelming probability

Using lattice isomorphism [DW22,BGPS23]

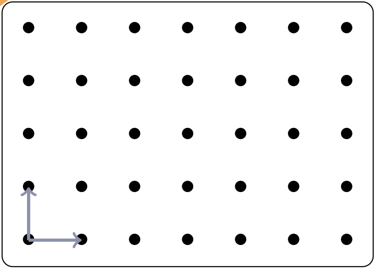


$\mathcal{L}_0 = \mathbb{Z}^n$

[DW22] Ducas and van Woerden. On the lattice isomorphism problem, quadratic forms [...] Eurocrypt

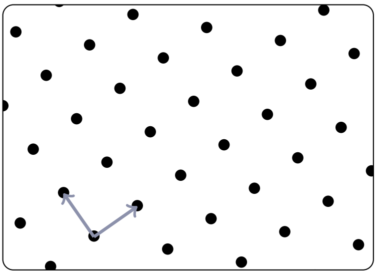
[BGPS23] Bennett, Ganju, Peetathawatchai, Stephens-Davidowitz. Just how hard are rotations of $\mathbb{Z}^n$? [...] Eurocrypt

Using lattice isomorphism [DW22,BGPS23]



$\mathcal{L}_0 = \mathbb{Z}^n$

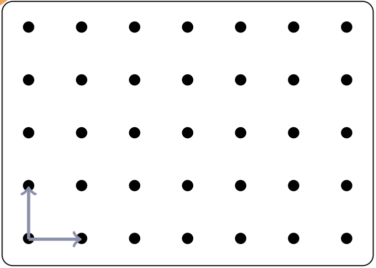
rotate
$\longrightarrow$
(choose O orthogonal matrix)



$\mathcal{L} = O \cdot \mathbb{Z}^n$

[DW22] Ducas and van Woerden. On the lattice isomorphism problem, quadratic forms [...] Eurocrypt
[BGPS23] Bennett, Ganju, Peetathawatchai, Stephens-Davidowitz. Just how hard are rotations of $\mathbb{Z}^n$? [...] Eurocrypt

Using lattice isomorphism [DW22,BGPS23]

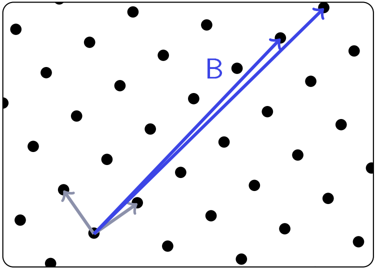


$\mathcal{L}_0 = \mathbb{Z}^n$

rotate

$\longrightarrow$

(choose O orthogonal matrix)



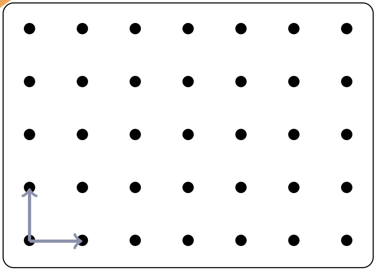
$\mathcal{L} = O \cdot \mathbb{Z}^n$

B worst-possible basis of $\mathcal{L}$

[DW22] Ducas and van Woerden. On the lattice isomorphism problem, quadratic forms [...] Eurocrypt

[BGPS23] Bennett, Ganju, Peetathawatchai, Stephens-Davidowitz. Just how hard are rotations of $\mathbb{Z}^n$? [...] Eurocrypt

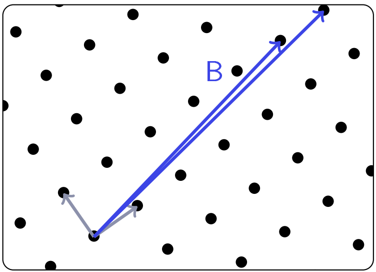
Using lattice isomorphism [DW22,BGPS23]



rotate

$\longrightarrow$

(choose O orthogonal matrix)



$\mathcal{L} = O \cdot \mathbb{Z}^n$

B worst-possible basis of $\mathcal{L}$

Lattice Isomorphism Problem (LIP) assumption

recovering O from B is hard

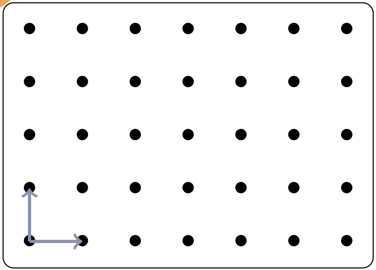
$\Leftrightarrow$ computing a shortest basis of $\mathcal{L}$ is hard

---

[DW22] Ducas and van Woerden. On the lattice isomorphism problem, quadratic forms [...] Eurocrypt

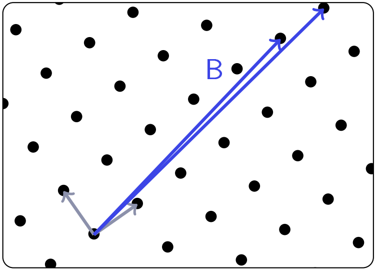
[BGPS23] Bennett, Ganju, Peetathawatchai, Stephens-Davidowitz. Just how hard are rotations of $\mathbb{Z}^n$? [...] Eurocrypt

Using lattice isomorphism [DW22,BGPS23]



$\mathcal{L}_0 = \mathbb{Z}^n$

rotate
$\longrightarrow$
(choose O orthogonal matrix)



$\mathcal{L} = O \cdot \mathbb{Z}^n$

B worst-possible basis of $\mathcal{L}$

Lattice Isomorphism Problem (LIP) assumption

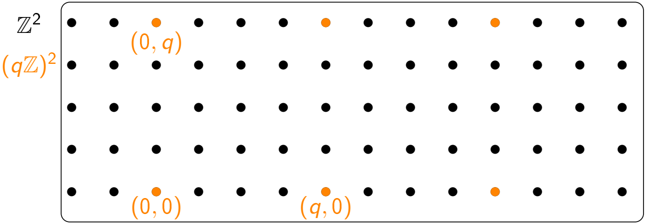
recovering O from B is hard

$\Leftrightarrow$ computing a shortest basis of $\mathcal{L}$ is hard

- Hawk: hash-and-sign + (module) LIP [DPPW23]

[DPPW23] Ducas, Postlethwaite, Pulles, van Woerden. Hawk: Module LIP makes lattice signatures [...] Asiacrypt

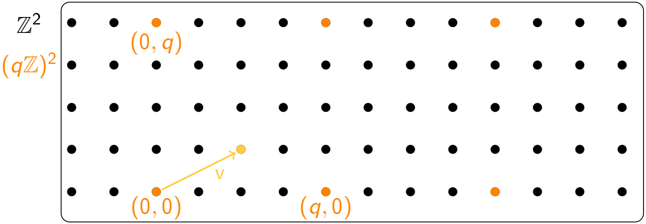
NTRU [HPS98]



- Start with $(q\mathbb{Z})^2$

[HPS98] Hoffstein, Pipher, and Silverman. NTRU: a ring based public key cryptosystem. ANTS.

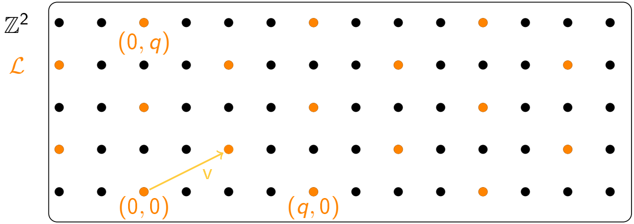
NTRU [HPS98]



- Start with $(q\mathbb{Z})^2$
- sample random short $\mathrm{v} \in \mathbb{Z}^2$
  $(\|\mathrm{v}\| \approx \sqrt{q})$

[HPS98] Hoffstein, Pipher, and Silverman. NTRU: a ring based public key cryptosystem. ANTS.

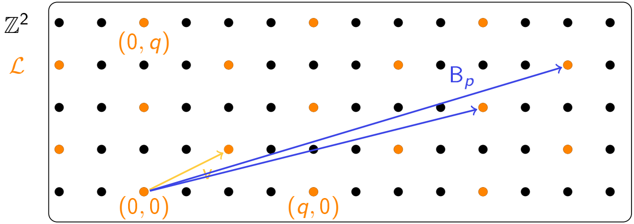
NTRU [HPS98]



- Start with $(q\mathbb{Z})^2$
- sample random short $\mathrm{v} \in \mathbb{Z}^2$
  ($\|\mathrm{v}\| \approx \sqrt{q}$)
- $\mathcal{L}$ spanned by v and $(q\mathbb{Z})^2$

[HPS98] Hoffstein, Pipher, and Silverman. NTRU: a ring based public key cryptosystem. ANTS.

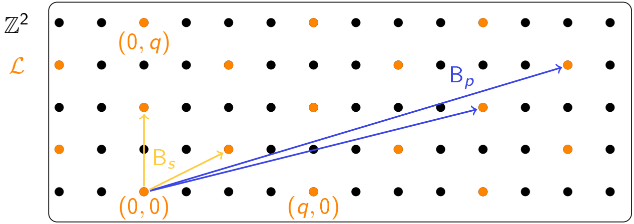
NTRU [HPS98]



- Start with $(q\mathbb{Z})^2$
- sample random short $\mathrm{v} \in \mathbb{Z}^2$ ($\|\mathrm{v}\| \approx \sqrt{q}$)
- $\mathcal{L}$ spanned by v and $(q\mathbb{Z})^2$
- B_p worst-possible basis of $\mathcal{L}$

[HPS98] Hoffstein, Pipher, and Silverman. NTRU: a ring based public key cryptosystem. ANTS.

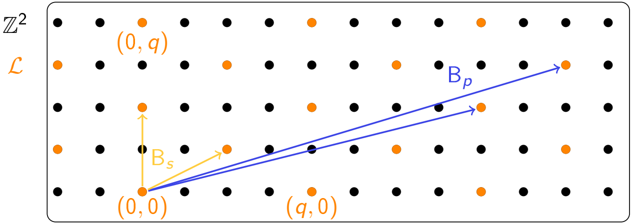
NTRU [HPS98]



- Start with $(q\mathbb{Z})^2$
- sample random short $v \in \mathbb{Z}^2$
  ($\|v\| \approx \sqrt{q}$)
- $\mathcal{L}$ spanned by v and $(q\mathbb{Z})^2$
- B_p worst-possible basis of $\mathcal{L}$
- B_s short basis
  (using knowledge of v short)

[HPS98] Hoffstein, Pipher, and Silverman. NTRU: a ring based public key cryptosystem. ANTS.

NTRU [HPS98]



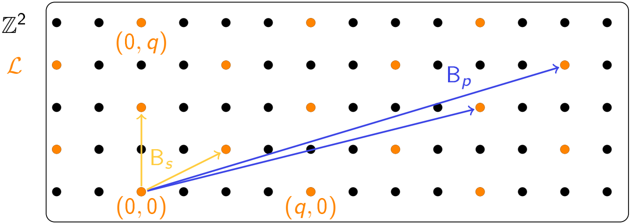
- Start with $(q\mathbb{Z})^2$
- sample random short $v \in \mathbb{Z}^2$
  $(\|v\| \approx \sqrt{q})$
- $\mathcal{L}$ spanned by v and $(q\mathbb{Z})^2$
- B_p worst-possible basis of $\mathcal{L}$
- B_s short basis
  (using knowledge of v short)

Issue: dimension 2

$\Rightarrow$ short basis problem is easy

[HPS98] Hoffstein, Pipher, and Silverman. NTRU: a ring based public key cryptosystem. ANTS.

NTRU [HPS98]



- Start with $(q\mathbb{Z})^2$
- sample random short $v \in \mathbb{Z}^2$
  $(\|v\| \approx \sqrt{q})$
- $\mathcal{L}$ spanned by v and $(q\mathbb{Z})^2$
- B_p worst-possible basis of $\mathcal{L}$
- B_s short basis
  (using knowledge of v short)

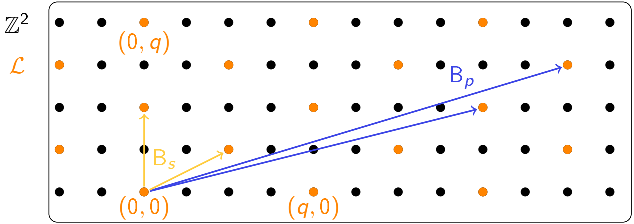
Issue: dimension 2
$\Rightarrow$ short basis problem is easy

Solution: use polynomials in $\mathbb{Z}[X]/(X^d + 1)$ instead of integers

- module lattice of dimension $2d$

[HPS98] Hoffstein, Pipher, and Silverman. NTRU: a ring based public key cryptosystem. ANTS.

NTRU [HPS98]



- Start with $(q\mathbb{Z})^2$
- sample random short $v \in \mathbb{Z}^2$
  ($\|v\| \approx \sqrt{q}$)
- $\mathcal{L}$ spanned by v and $(q\mathbb{Z})^2$
- B_p worst-possible basis of $\mathcal{L}$
- B_s short basis
  (using knowledge of v short)

Issue: dimension 2
$\Rightarrow$ short basis problem is easy

Solution: use polynomials in $\mathbb{Z}[X]/(X^d + 1)$ instead of integers

- module lattice of dimension $2d$
- Falcon: hash-and-sign + NTRU

[Falcon] Fouque, Hoffstein, Kirchner, Lyubashevsky, Pornin, Prest, Ricosset, Seiler, Whyte, Zhang. NIST standard

Short Integer Solution [Ajt96]

Short Integer Solution (SIS) assumption

Let $A \leftarrow \mathcal{U}(\mathbb{Z}_q^{m \times n})$ $(m > n \log q)$ and

$$\mathcal{L}(A) := \{x \in \mathbb{Z}^m \,|\, xA = 0 \bmod q\}.$$

Finding a short basis of $\mathcal{L}(A)$ is hard with overwhelming probability.

[Ajt96] Ajtai. Generating hard instances of lattice problems. STOC.

Short Integer Solution [Ajt96]

Short Integer Solution (SIS) assumption

Let $\boxed{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{m \times n})$ $(m > n \log q)$ and

$$\mathcal{L}(A) := \{\mathrm{x} \in \mathbb{Z}^m \,|\, \mathrm{x}A = 0 \bmod q\}.$$

Finding a short basis of $\mathcal{L}(A)$ is hard with overwhelming probability.

Lemma: if there exists one lattice for which the short basis problem is hard, then the SIS assumption holds. [Ajt96]

[Ajt96] Ajtai. Generating hard instances of lattice problems. STOC.

Short Integer Solution [Ajt96]

Short Integer Solution (SIS) assumption

Let $A \leftarrow \mathcal{U}(\mathbb{Z}_q^{m\times n})$ $(m > n \log q)$ and

$$\mathcal{L}(A) := \{\mathrm{x} \in \mathbb{Z}^m \,|\, \mathrm{x}A = 0 \bmod q\}.$$

Finding a short basis of $\mathcal{L}(A)$ is hard with overwhelming probability.

Lemma: if there exists one lattice for which the short basis problem is hard, then the SIS assumption holds. [Ajt96]

Lemma: one can sample A uniformly + a short basis B_s of $\mathcal{L}(A)$ in polynomial time [Ajt99]

[Ajt99] Ajtai. Generating hard instances of the short basis problem. ICALP.

Short Integer Solution [Ajt96]

Short Integer Solution (SIS) assumption

Let $\boxed{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{m \times n})$ $(m > n \log q)$ and

$$\mathcal{L}(A) := \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{x}A = 0 \bmod q\}.$$

Finding a short basis of $\mathcal{L}(A)$ is hard with overwhelming probability.

Lemma: if there exists one lattice for which the short basis problem is hard, then the SIS assumption holds. [Ajt96]

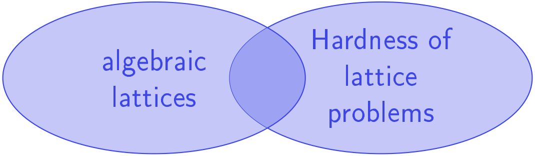
Lemma: one can sample A uniformly + a short basis $\mathbf{B}_s$ of $\mathcal{L}(A)$ in polynomial time [Ajt99]

- GPV: hash-and-sign + SIS [GPV08]

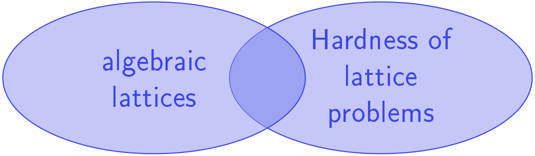
[GPV08] Gentry, Peikert, Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. STOC

Conclusion

Some questions that interest me



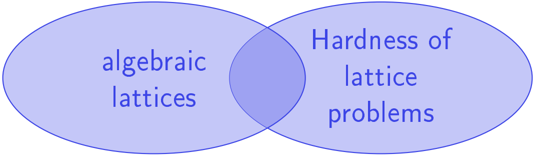
Some questions that interest me



Some concrete questions: (come ask me if you want to know more)

- can we generate a random prime ideal $\mathfrak{p}$ in a number field K together with a short element in it?
- can we re-randomize an NTRU instance?

Some questions that interest me

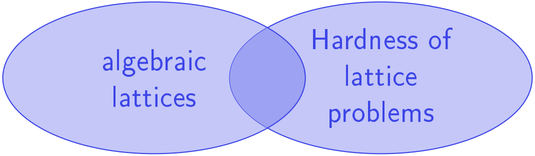


Some concrete questions: (come ask me if you want to know more)

- can we generate a random prime ideal $\mathfrak{p}$ in a number field K together with a short element in it?
- can we re-randomize an NTRU instance?

Open position: 2 years post-doc on quantum cryptanalysis (Bordeaux)

Some questions that interest me



Some concrete questions: (come ask me if you want to know more)

- can we generate a random prime ideal $\mathfrak{p}$ in a number field K together with a short element in it?
- can we re-randomize an NTRU instance?

Open position: 2 years post-doc on quantum cryptanalysis (Bordeaux)

Thank you