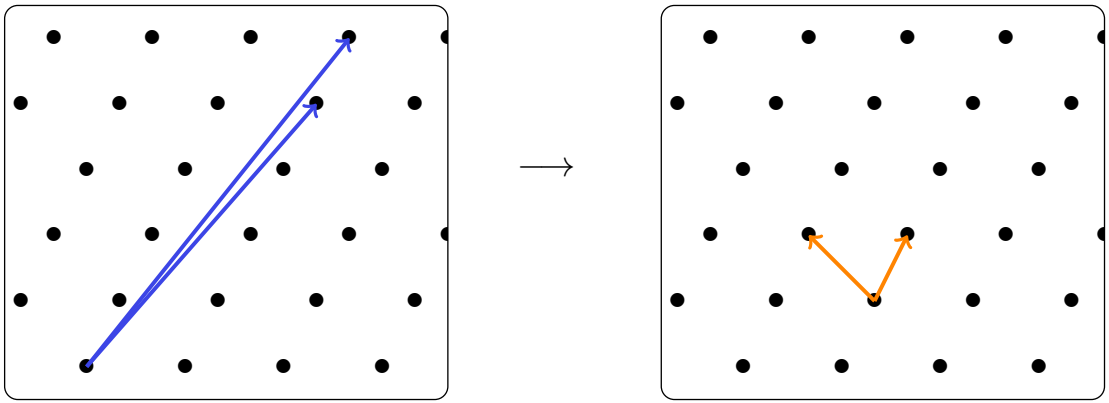
Lattice-based cryptography

Support slides

Alice Pellet-Mary

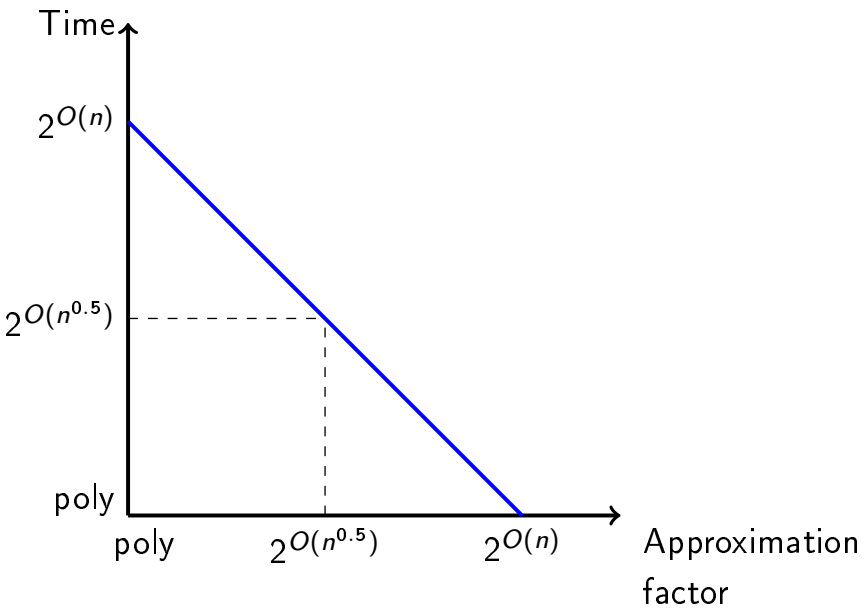
CIMPA school in Pondicherry

Lattice reduction algorithms



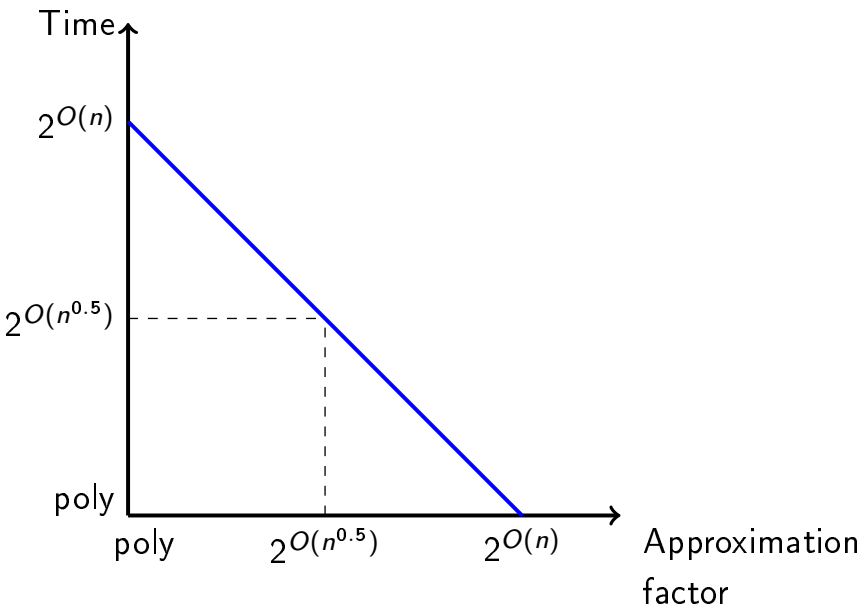
Various SVP algorithms

BKZ trade-offs



Various SVP algorithms

BKZ trade-offs

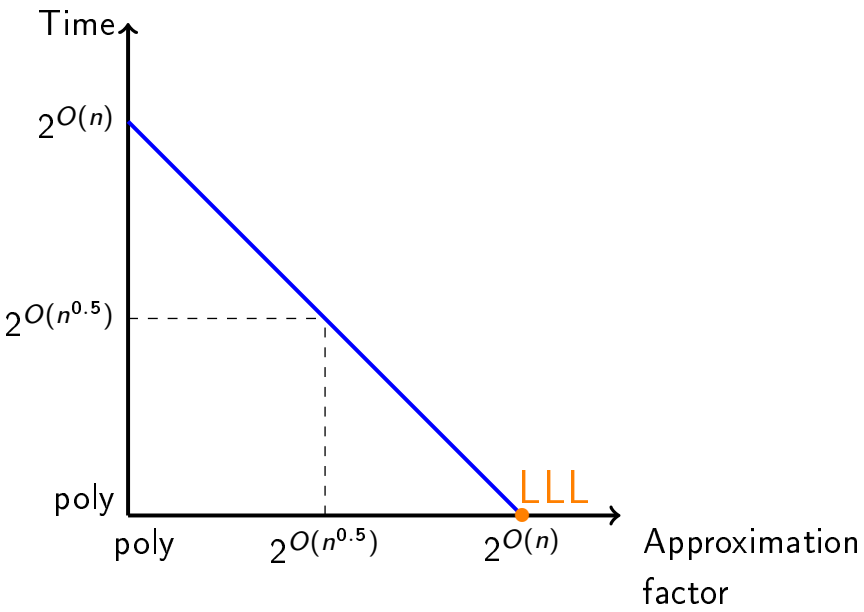


Lagrange-Gauss algorithm: dim 2

- exact SVP/SIVP
- polynomial time

Various SVP algorithms

BKZ trade-offs



Lagrange-Gauss algorithm: dim 2

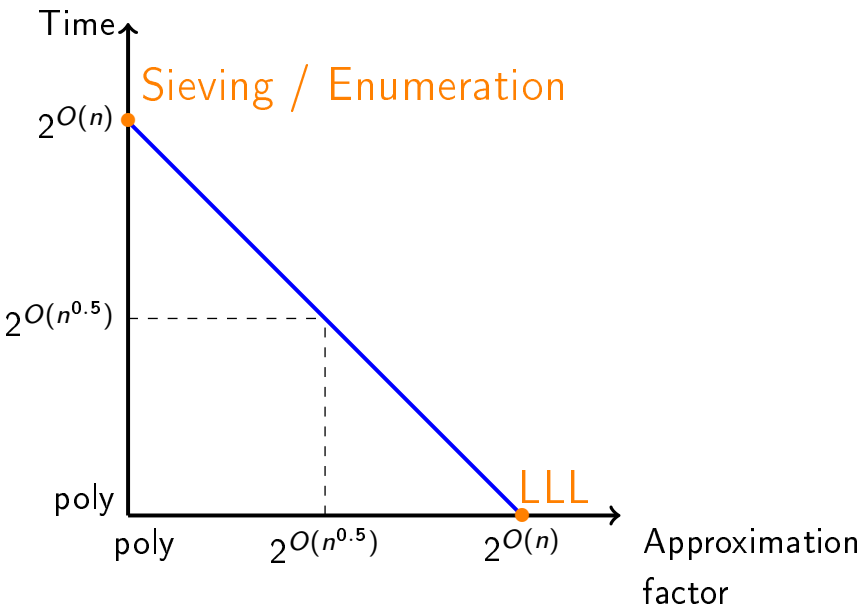
- exact SVP/SIVP
- polynomial time

LLL algorithm: dim n

- γ-SVP / γ-SIVP with $\gamma = 2^n$
- polynomial time

Various SVP algorithms

BKZ trade-offs



Lagrange-Gauss algorithm: dim 2

- exact SVP/SIVP
- polynomial time

LLL algorithm: dim n

- γ-SVP / γ-SIVP with $\gamma = 2^n$
- polynomial time

Sieving algorithm: dim n

- exact SVP
- time $2^{O(n)}$

Lagrange-Gauss algorithm

video

Lagrange-Gauss algorithm

video

Theorem: the algorithm

- solves γ-SVP / γ-SIVP in L for $\gamma = 1$
- 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 $B = (b_1, \dots, b_n)$

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 $B = (b_1, \dots, b_n)$

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

Algorithm:

- while there exist i such that $\|b_i\|_2 > \lambda_1(L_i)$
  (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 $B = (b_1, \dots, b_n)$

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

Algorithm:

- while there exist i such that $\|b_i\|_2 > \lambda_1(L_i)$
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- solves γ-SVP / γ-SIVP in L for $\gamma = 2^n$

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

The LLL algorithm [LLL82]

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

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

Algorithm:

- while there exist i such that $\|b_i\|_2 > \lambda_1(L_i)$
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- solves γ-SVP / γ-SIVP in L for $\gamma = 2^n$
- does not run in polynomial time

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

The LLL algorithm [LLL82]

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

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

Algorithm:

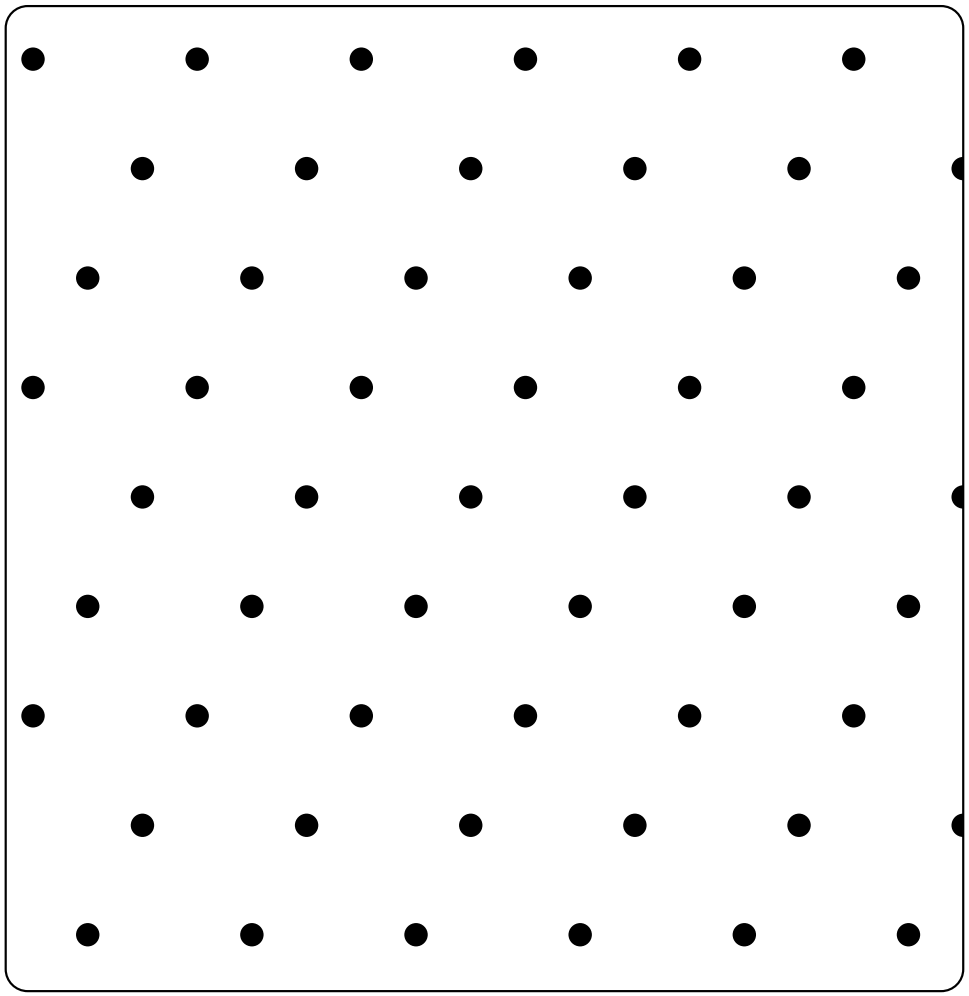
- while there exist i such that $\|b_i\|_2 > 4/3 \cdot \lambda_1(L_i)$
  (L_i is roughly the lattice spanned by (b_i, b_{i+1}))
  - run Lagrange-Gauss on L_i

This algorithm

- solves γ-SVP / γ-SIVP in L for $\gamma = 2^n$
- runs in polynomial time

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

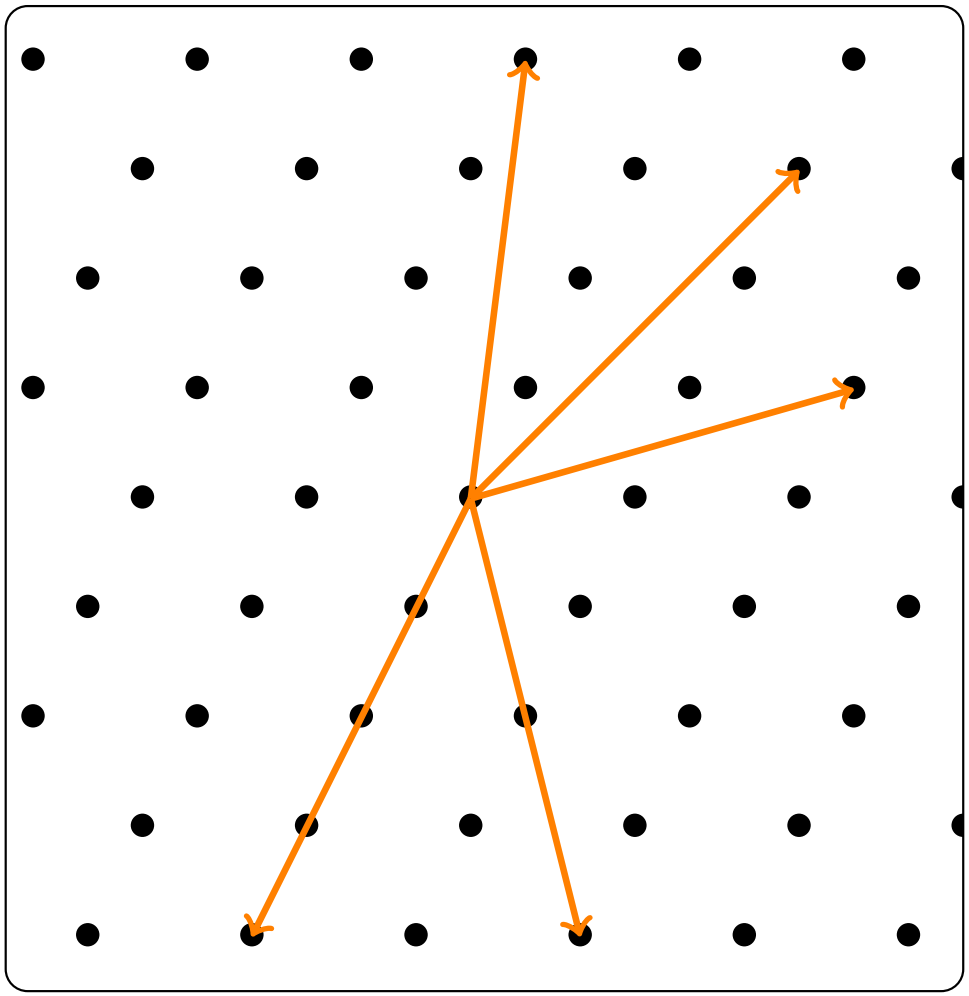
Sieving algorithm [AKS01]



Sieving:

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

Sieving algorithm [AKS01]

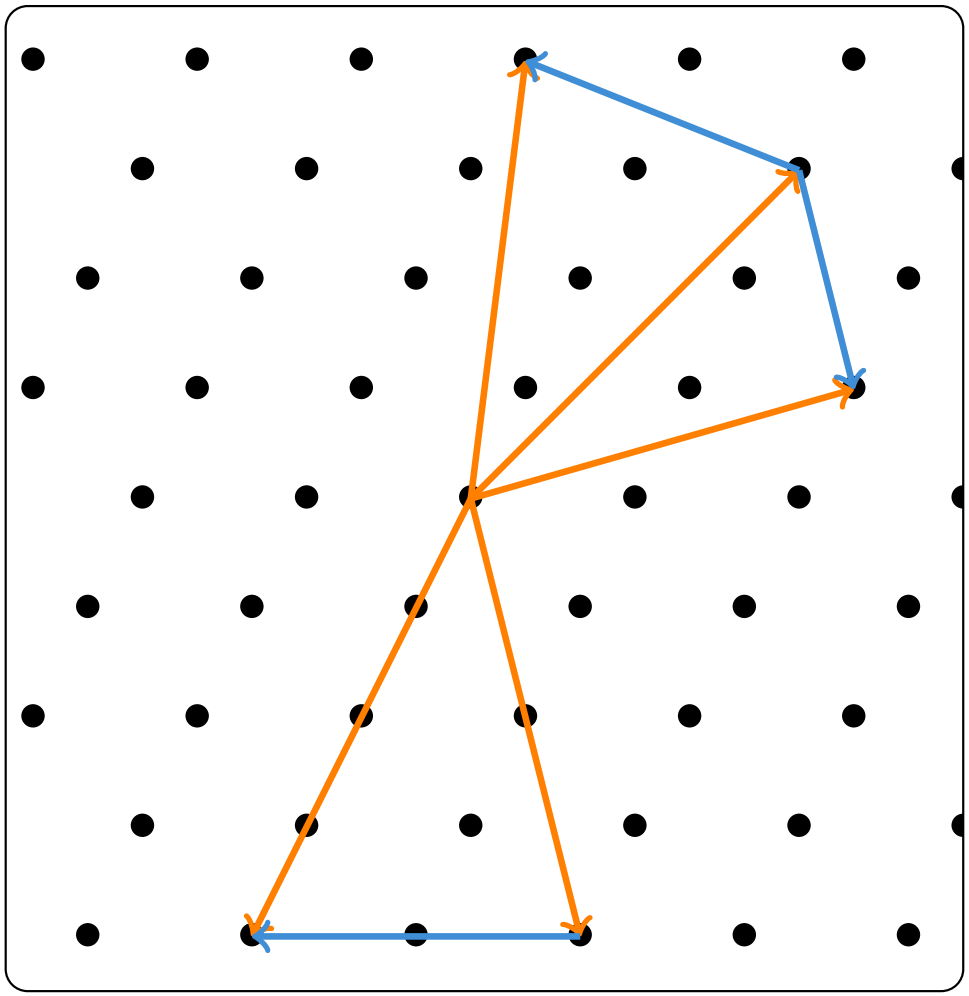


Sieving:

- Create many large vectors

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

Sieving algorithm [AKS01]

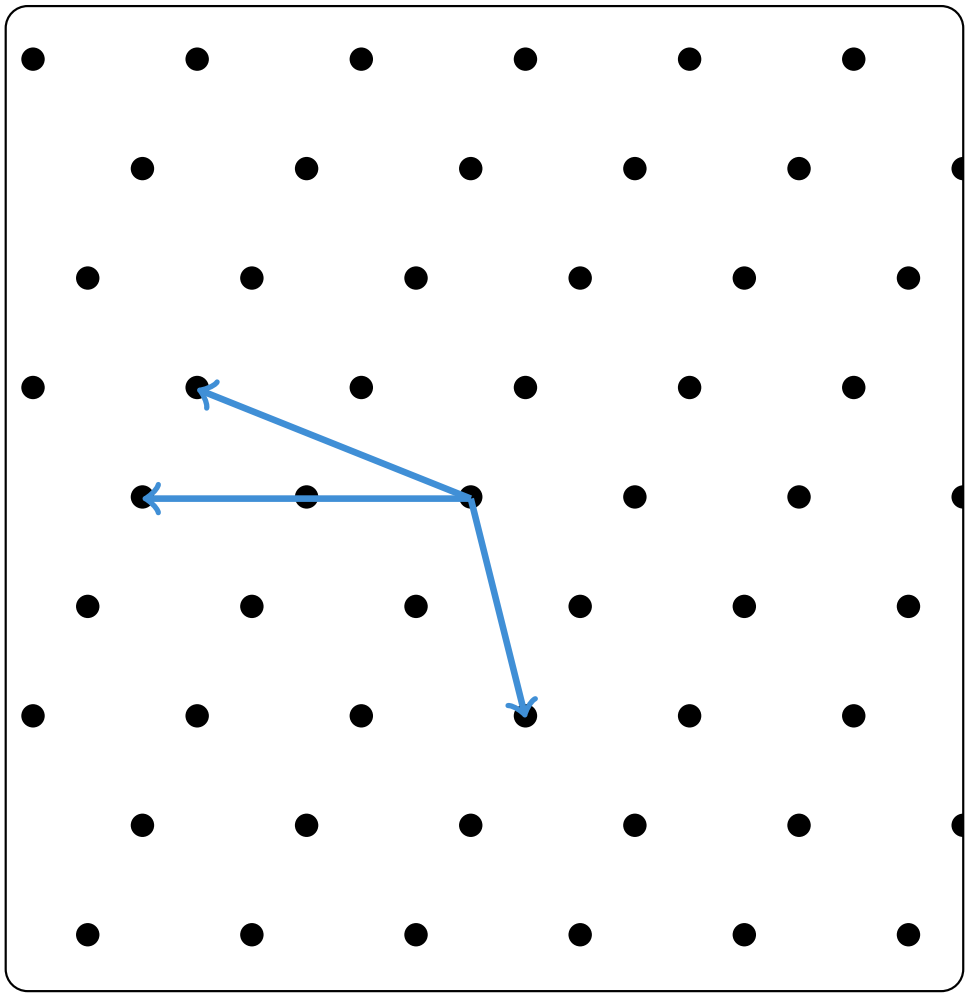


Sieving:

- 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]

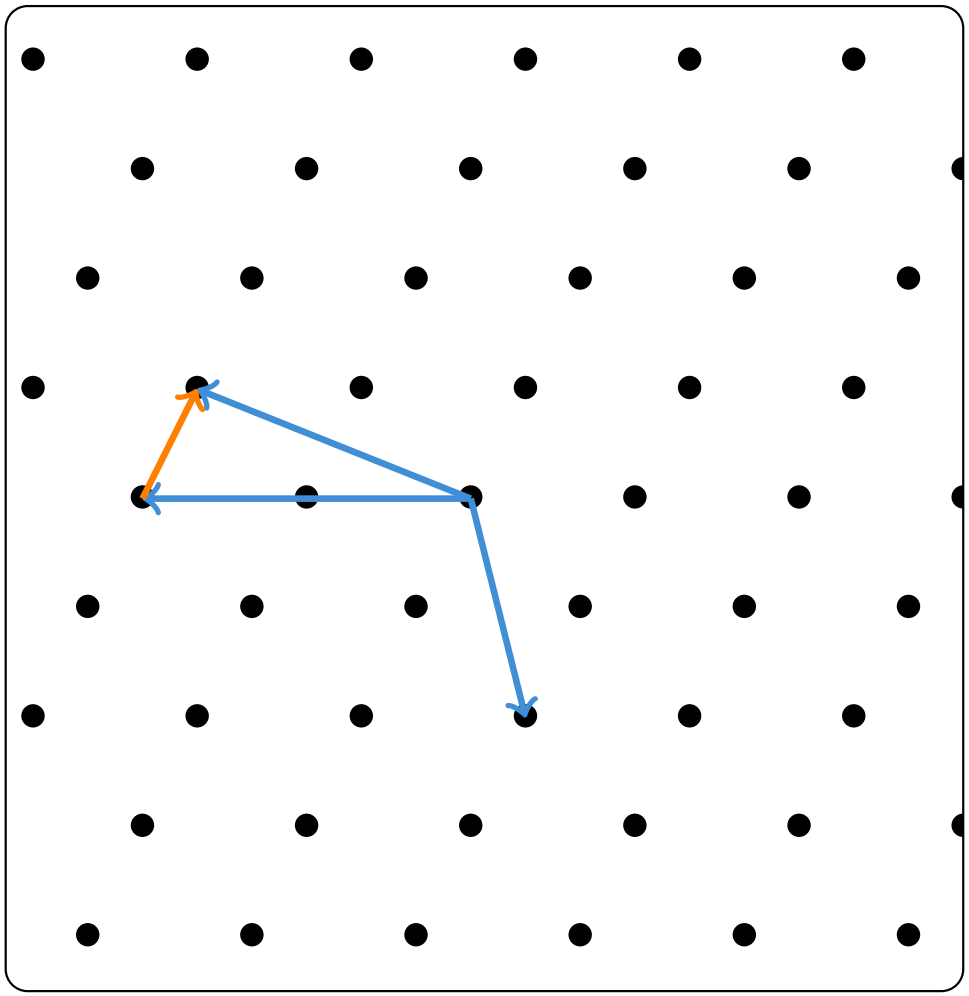


Sieving:

- 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]

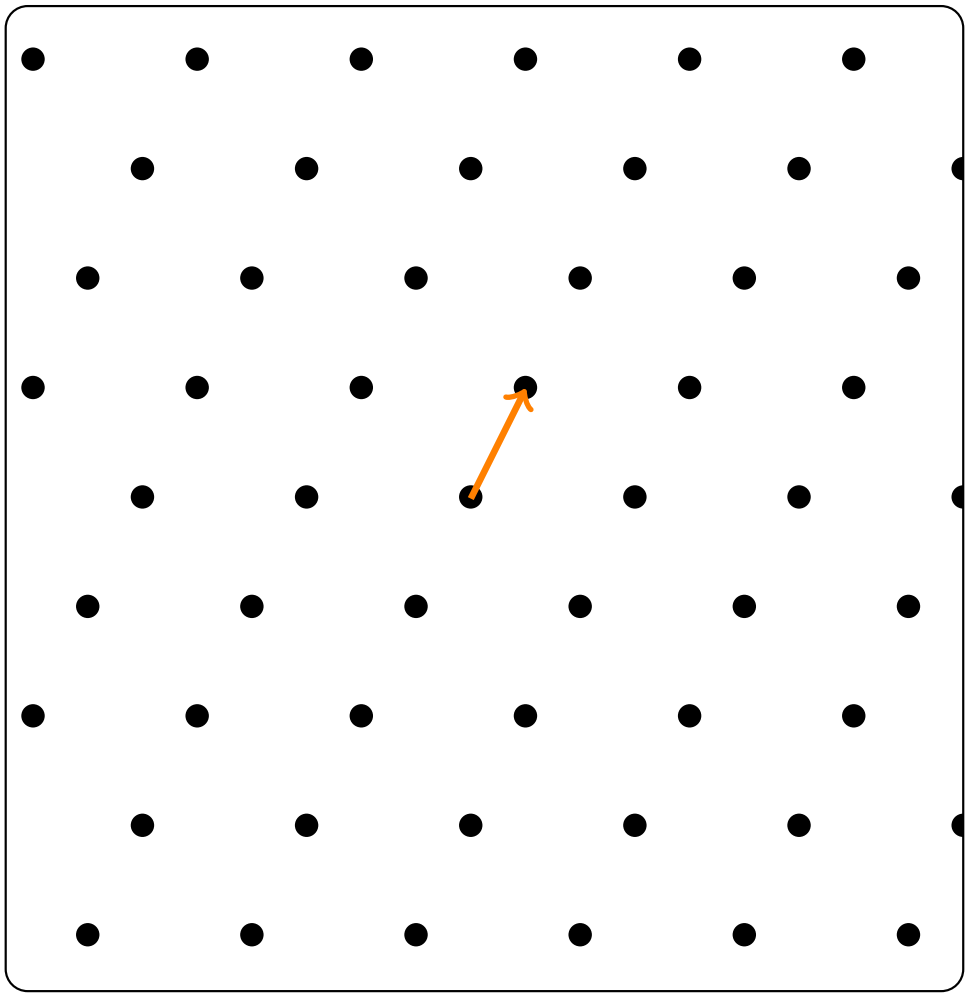


Sieving:

- 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]

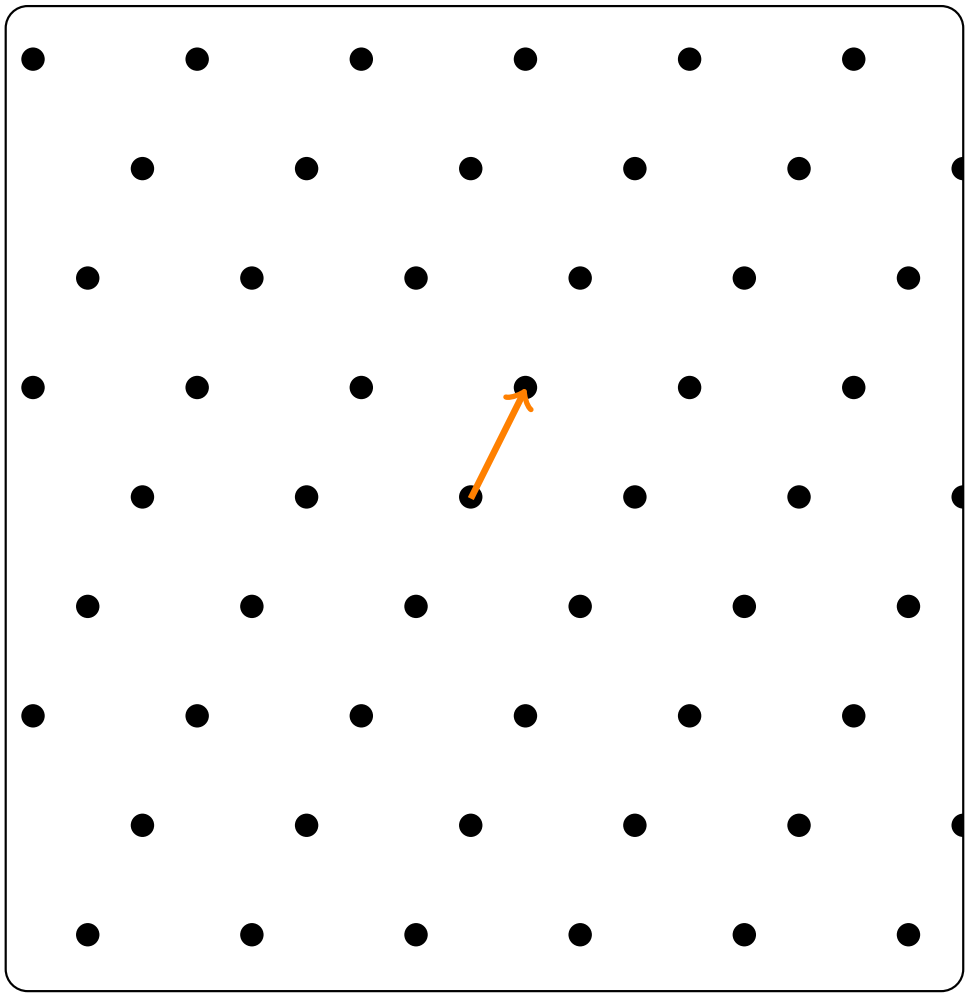


Sieving:

- 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]



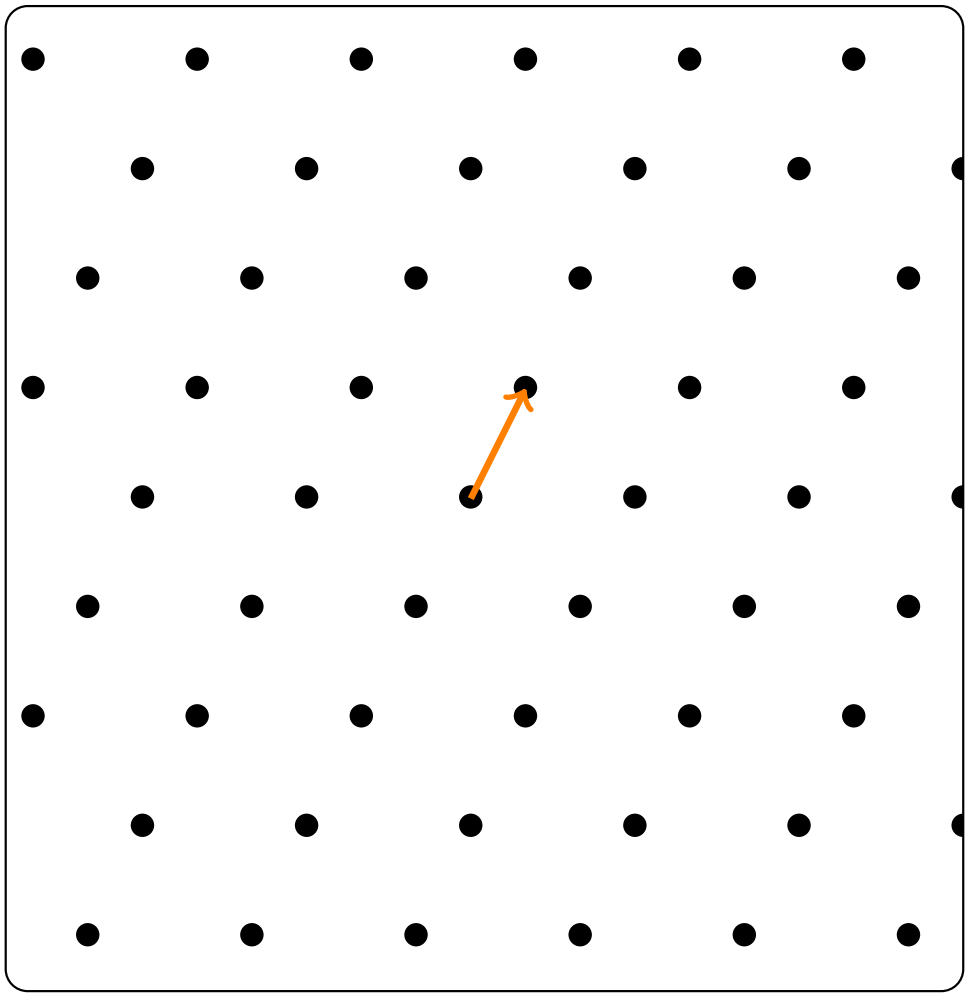
Sieving:

- 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

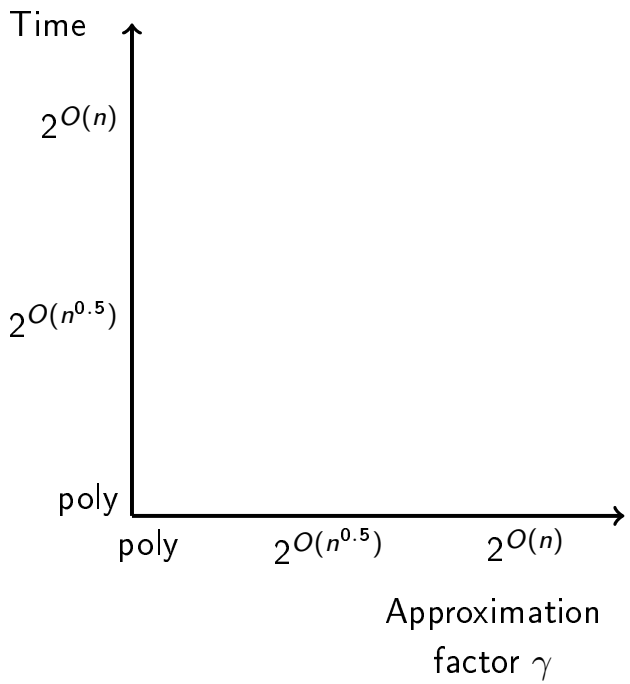
Size of the initial list: $2^{O(n)}$

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

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

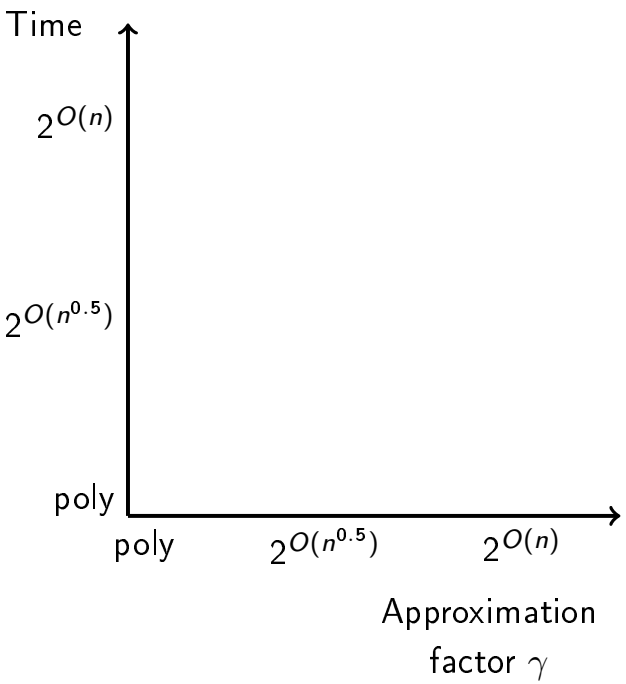
Summing up and BKZ

Algorithms for for SVP_γ



Summing up and BKZ

Algorithms for for SVP_γ

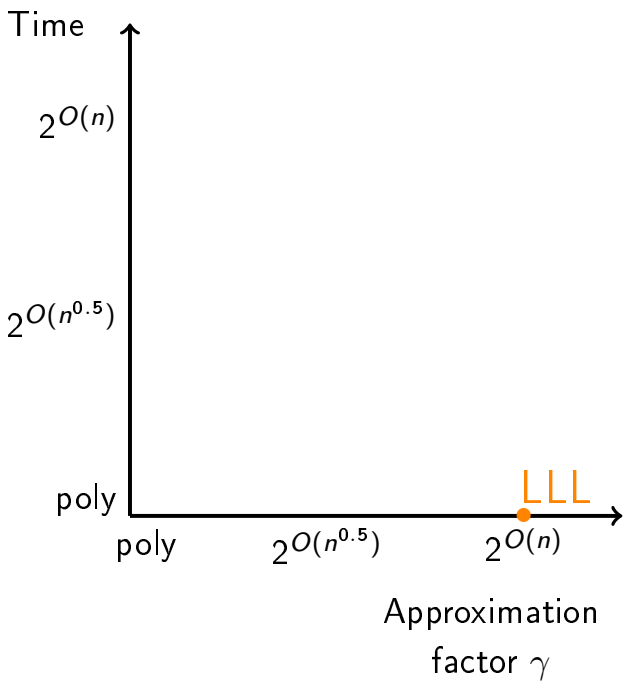


Lagrange-Gauss algorithm: dim 2

- $\gamma = 1$
- polynomial time

Summing up and BKZ

Algorithms for for SVP_γ



Lagrange-Gauss algorithm: dim 2

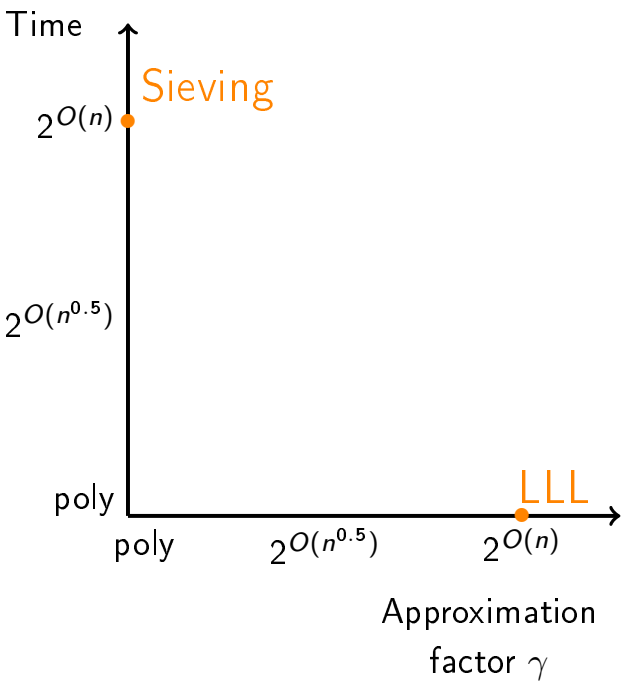
- $\gamma = 1$
- polynomial time

LLL algorithm: dim n

- $\gamma = 2^n$
- polynomial time

Summing up and BKZ

Algorithms for for SVP_γ



Lagrange-Gauss algorithm: dim 2

- $\gamma = 1$
- polynomial time

LLL algorithm: dim n

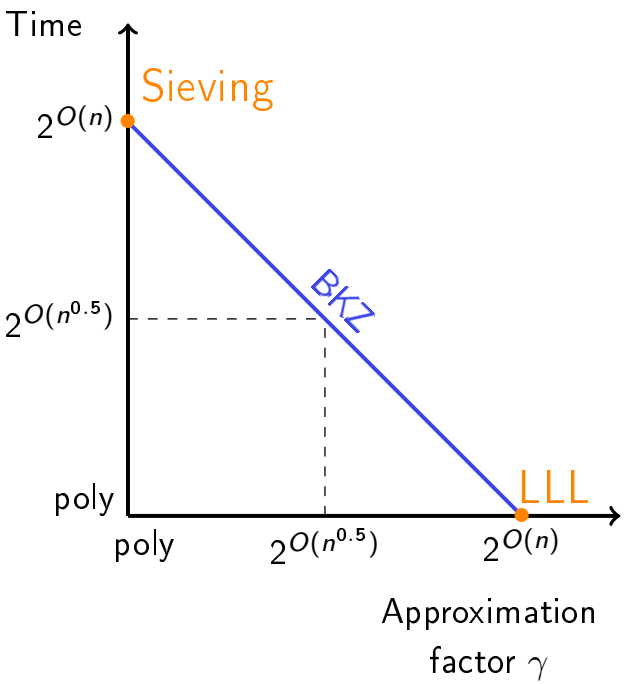
- $\gamma = 2^n$
- polynomial time

Sieving algorithm: dim n

- $\gamma = 1$
- time $2^{O(n)}$

Summing up and BKZ

Algorithms for for SVP_γ



Lagrange-Gauss algorithm: dim 2

- $\gamma = 1$
- polynomial time

LLL algorithm: dim n

- $\gamma = 2^n$
- polynomial time

Sieving algorithm: dim n

- $\gamma = 1$
- time $2^{O(n)}$

BKZ algorithm: combine LLL + Sieving

$\Rightarrow$ various trade-offs

Some concrete numbers

Solving SVP_γ in practice for $\gamma = 1$:

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

Some concrete numbers

Solving SVP_γ in practice for $\gamma = 1$:

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

Some concrete numbers

Solving SVP_γ in practice for $\gamma = 1$:

- $n = 2 \rightsquigarrow$ 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

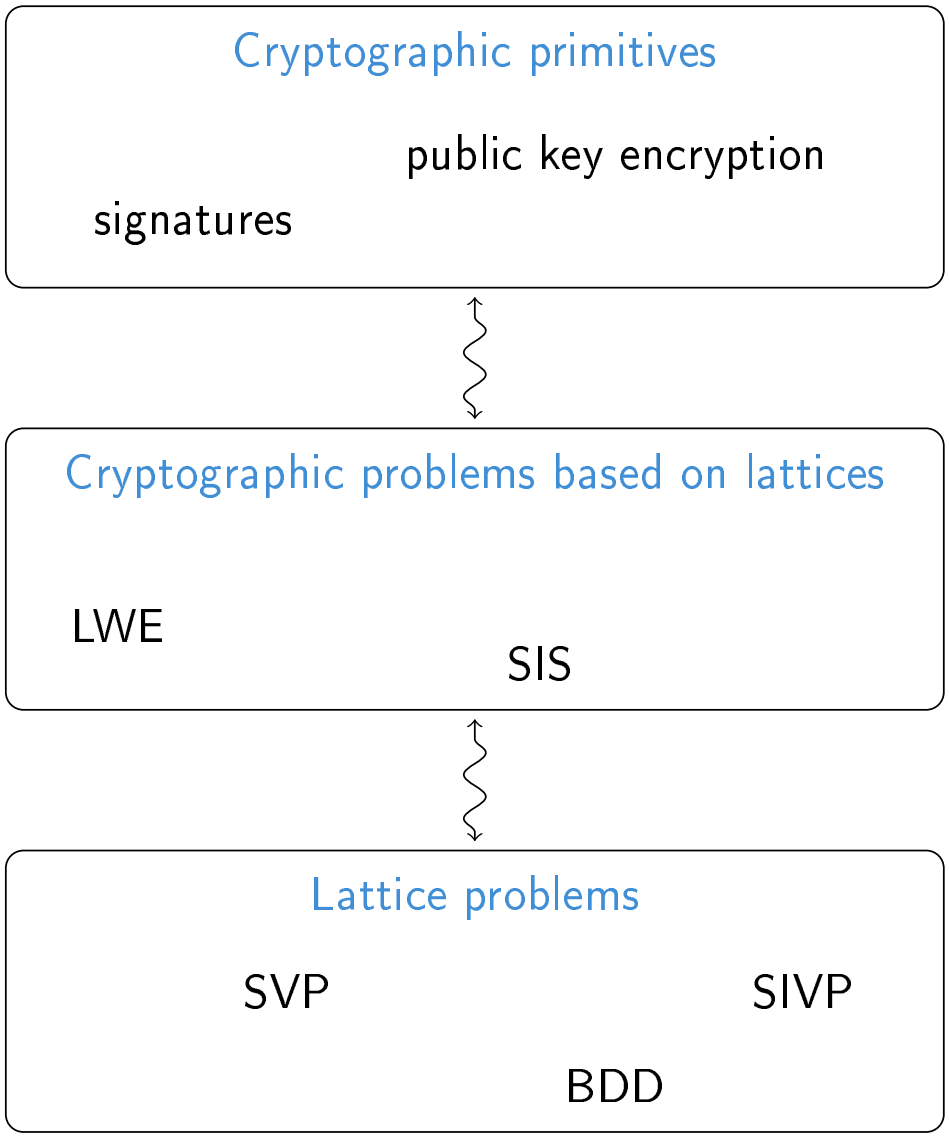
Solving SVP_γ in practice for $\gamma = 1$:

- $n = 2 \rightsquigarrow$ 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

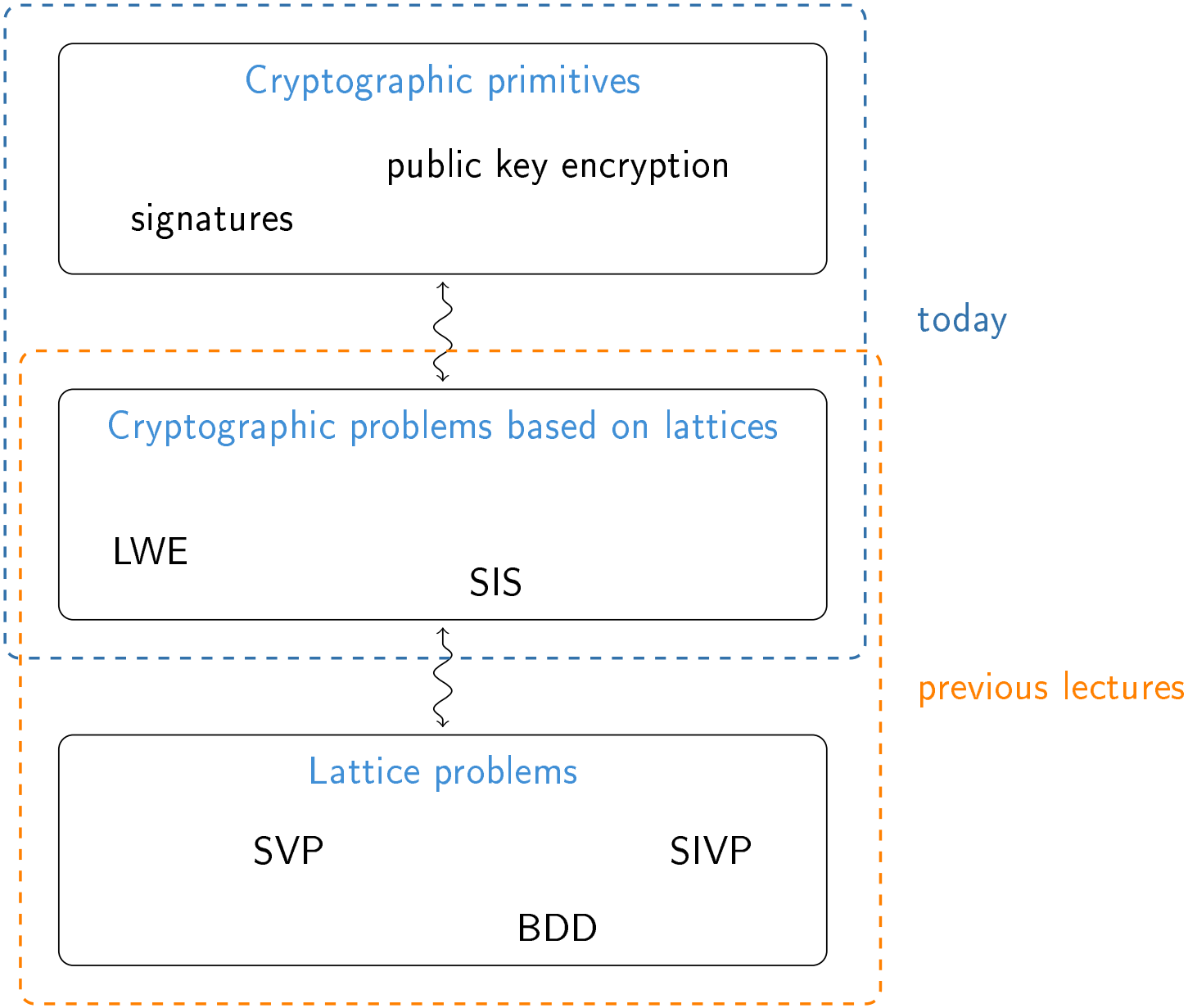
[DSW21] Ducas, Stevens, van Woerden. Advanced Lattice Sieving on GPUs, with Tensor Cores.

Cryptographic constructions

Context


Cryptographic primitives
public key encryption
signatures
Cryptographic problems based on lattices
LWE
SIS
Lattice problems
SVP
SIVP
BDD

Context


Cryptographic primitives
public key encryption
signatures
today
Cryptographic problems based on lattices
LWE
SIS
previous lectures
Lattice problems
SVP
SIVP
BDD

Reminder

What we have seen: LWE and SIS problems

- average case problems
- expressed using simple linear algebra
- best known algorithm takes time $2^{\Omega(n)}$ (if well chosen parameters)
  - even quantumly
- practical hardness quite well understood

Disclaimer

So far we have seen:

- that LWE/SIS is as hard as worst-case lattice problems

  (i.e., if we can solve LWE/SIS with good proba, we can solve some lattice problem over all lattices)

Disclaimer

So far we have seen:

- that LWE/SIS is as hard as worst-case lattice problems
  (i.e., if we can solve LWE/SIS with good proba, we can solve some lattice problem over all lattices)

Today we will see:

- cryptographic schemes based on LWE and SIS

Disclaimer

So far we have seen:

- that LWE/SIS is as hard as worst-case lattice problems
  (i.e., if we can solve LWE/SIS with good proba, we can solve some lattice problem over all lattices)

Today we will see:

- cryptographic schemes based on LWE and SIS

But...

- for practical constructions, we choose parameters for which the reductions to worst-case problem do not hold
  - e.g., binary noise, small modulus q, ...

Disclaimer

So far we have seen:

- that LWE/SIS is as hard as worst-case lattice problems
  (i.e., if we can solve LWE/SIS with good proba, we can solve some lattice problem over all lattices)

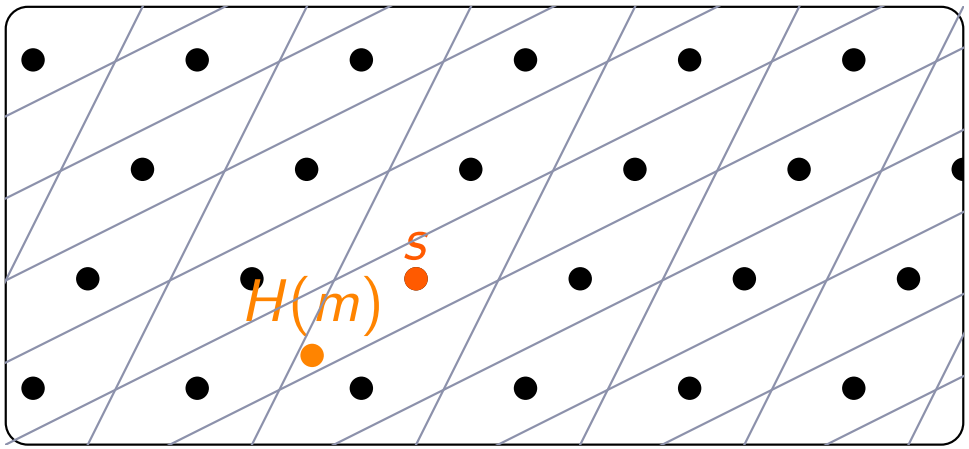
Today we will see:

- cryptographic schemes based on LWE and SIS

But...

- for practical constructions, we choose parameters for which the reductions to worst-case problem do not hold
  - e.g., binary noise, small modulus q, ...
- reductions are used to show that there is no fundamental flaw in the design
  - taking larger parameters, we can prove that the schemes are as secure as worst case lattice problems

Hash-and-sign signature



Trapdoors

Two (related) notions of trapdoors for lattices:

- short basis of $\mathcal{L}$
- gadget-based

Trapdoors

Two (related) notions of trapdoors for lattices:

- short basis of $\mathcal{L}$
- gadget-based

Short basis

Idea: construct a lattice $\mathcal{L}$ with a good basis B_0 and a bad basis B_1

- given B_1, decoding in $\mathcal{L}$ is hard
- given B_0, decoding in $\mathcal{L}$ is easy

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

Short basis

Idea: construct a lattice $\mathcal{L}$ with a good basis B_0 and a bad basis B_1

- given B_1, decoding in $\mathcal{L}$ is hard
- given B_0, decoding in $\mathcal{L}$ is easy

Lemma [Ajt99]

One can efficiently sample a uniform matrix $A \in \mathbb{Z}_q^{m \times n}$ together with a short basis of the kernel lattice $\mathcal{L} := \Lambda^{\perp}(A)$.

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

Short basis

Idea: construct a lattice $\mathcal{L}$ with a good basis B_0 and a bad basis B_1

- given B_1, decoding in $\mathcal{L}$ is hard
- given B_0, decoding in $\mathcal{L}$ is easy

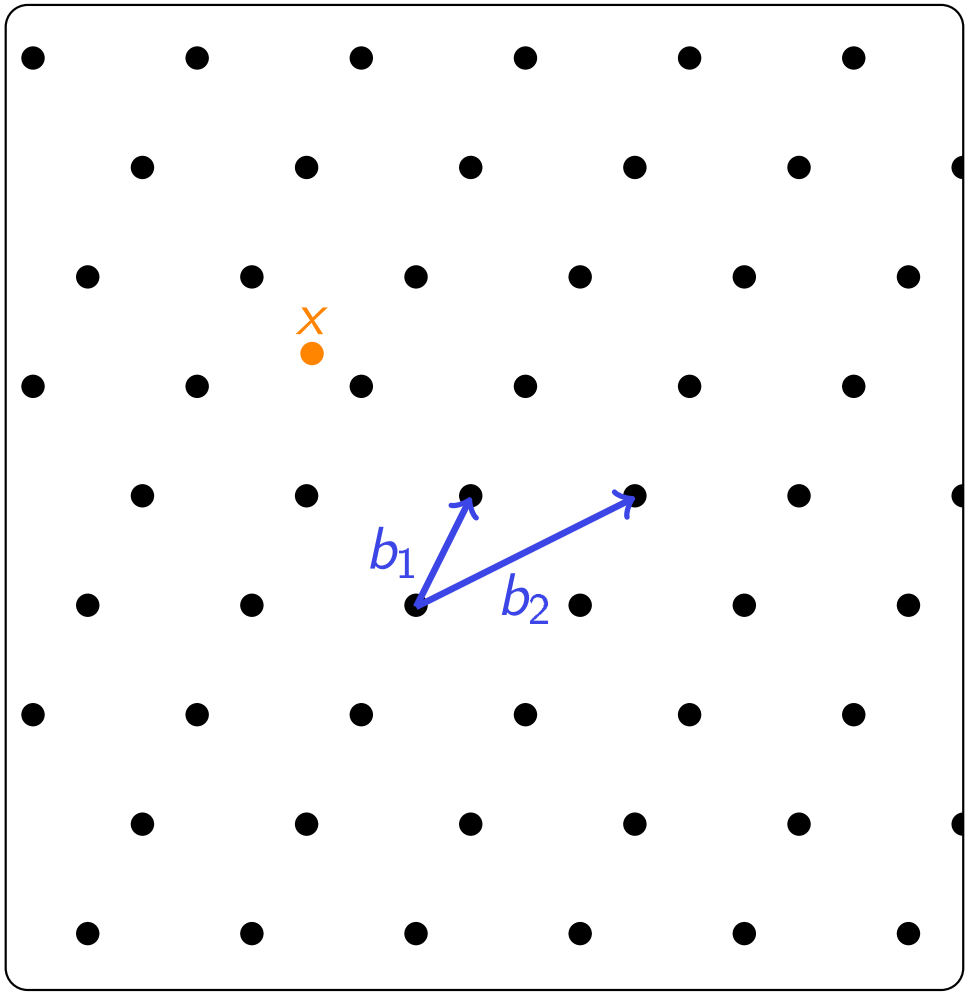
Lemma [Ajt99]

One can efficiently sample a uniform matrix $A \in \mathbb{Z}_q^{m \times n}$ together with a short basis of the kernel lattice $\mathcal{L} := \Lambda^{\perp}(A)$.

- decoding in $\mathcal{L}$ is hard if SIS is hard
- the short basis enables to decode efficiently

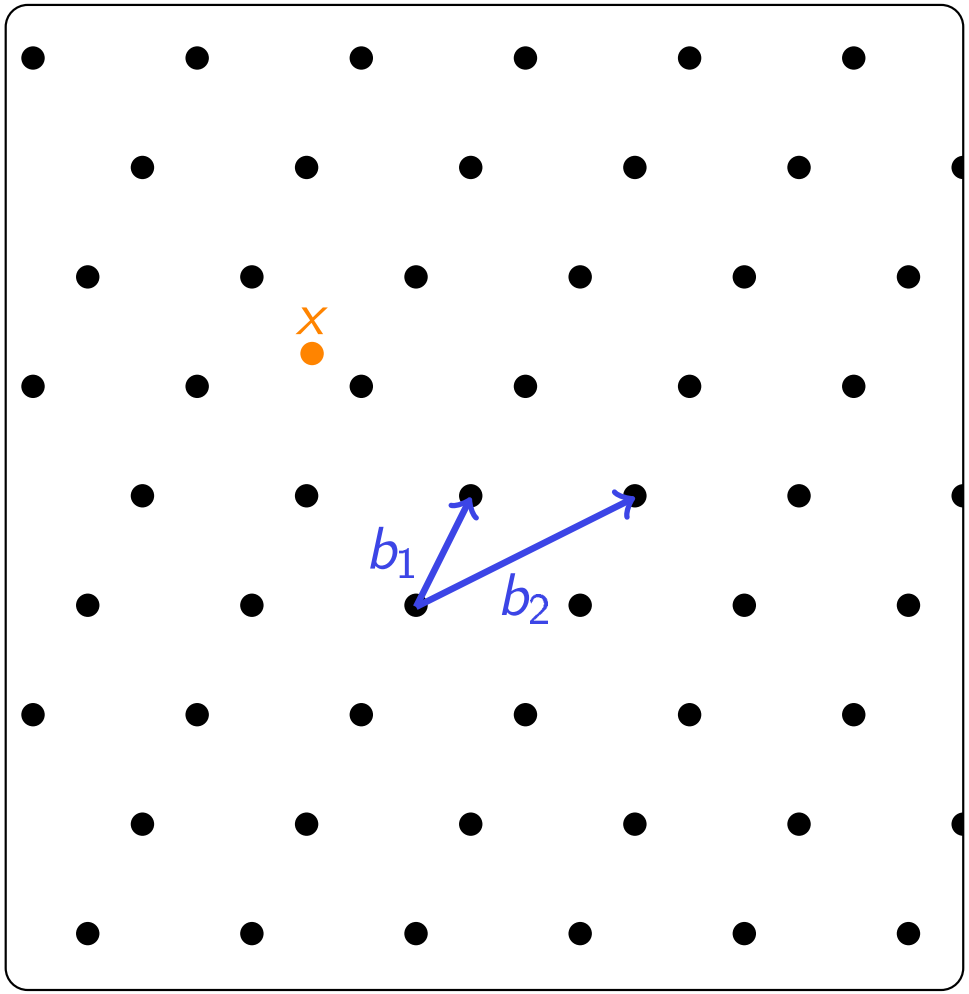
[Ajt99] Ajtai. Generating hard instances of the short basis problem. ICALP.

Decoding in a lattice using a short basis



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

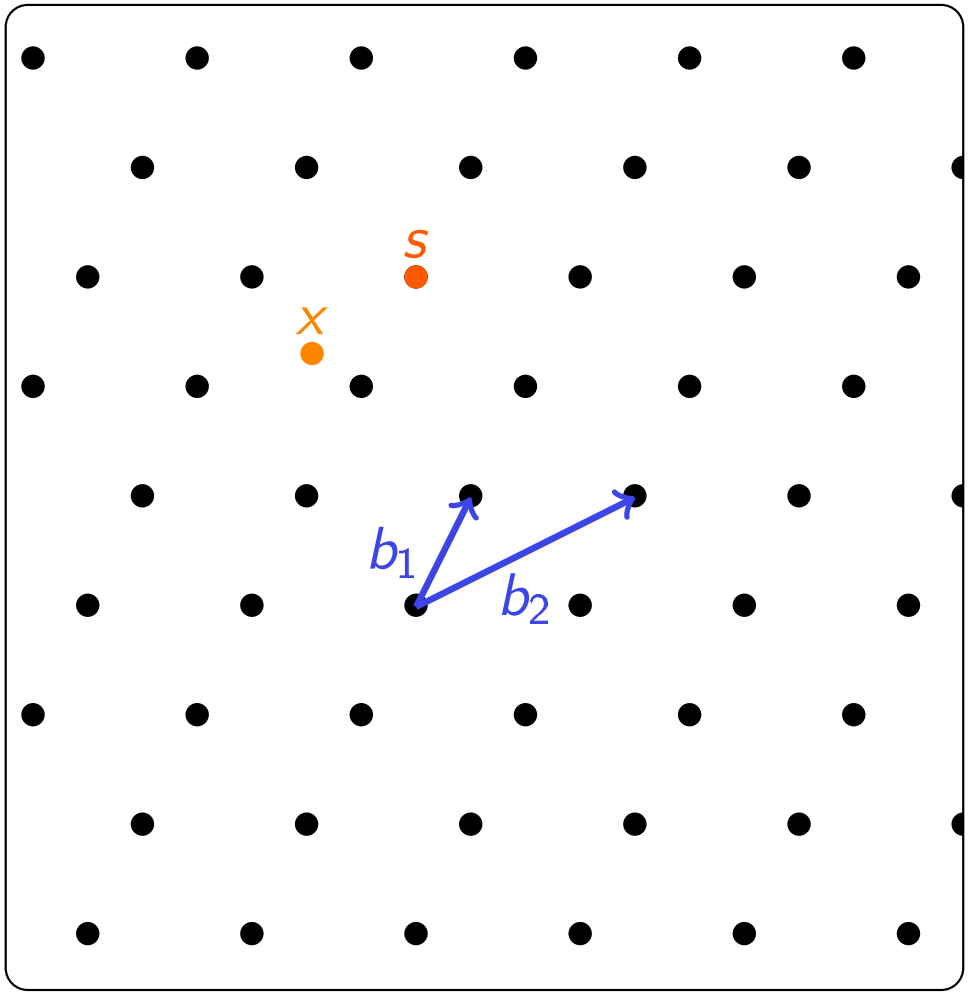
Decoding in a lattice using a short basis



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

Algo: round each coordinate

Decoding in a lattice using a short basis

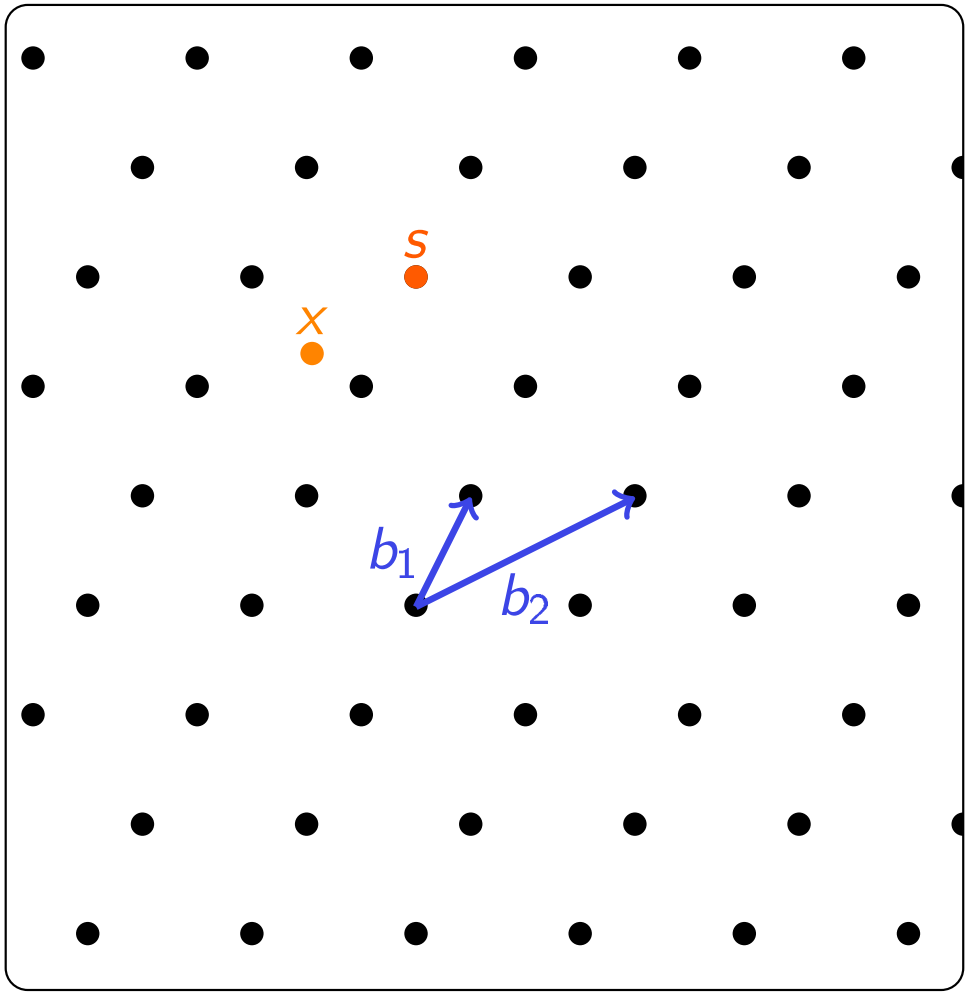


Input: $x = 3.7 \cdot b_1 - 1.4 \cdot b_2$

Algo: round each coordinate

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

Decoding in a lattice using a short basis



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

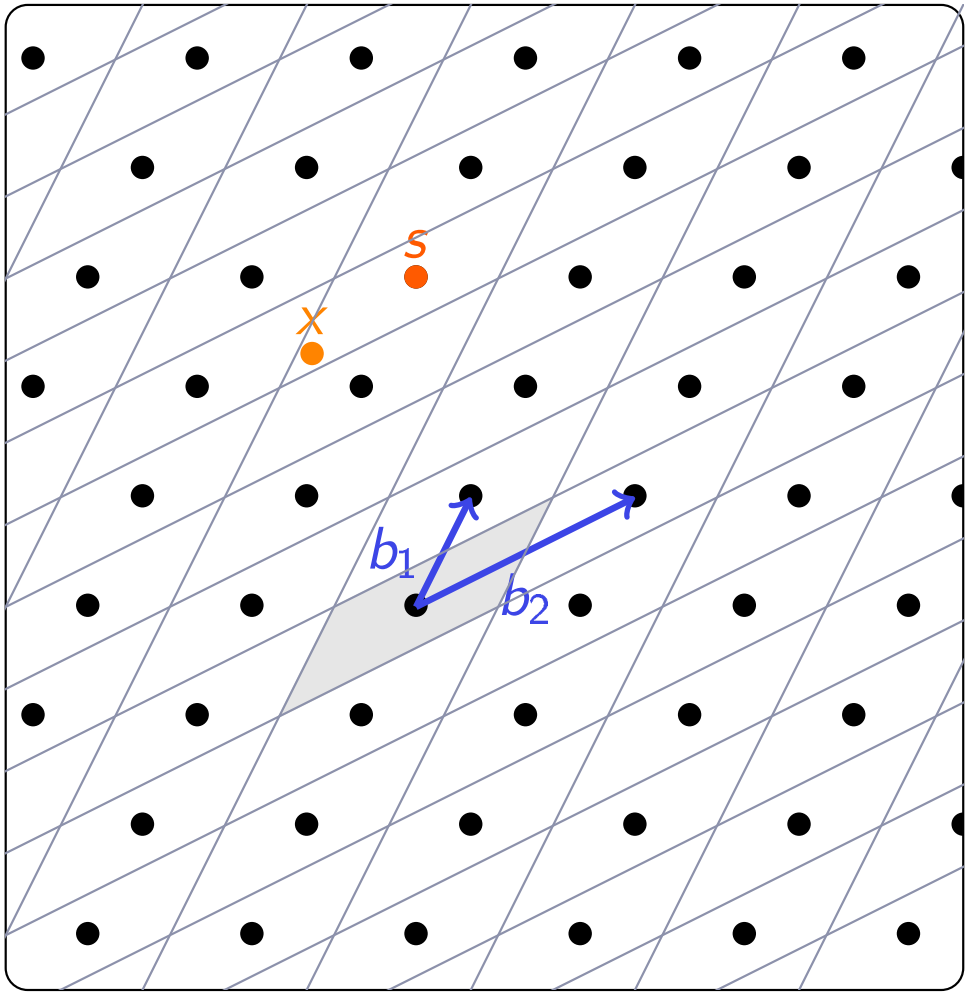
Algo: round each coordinate

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

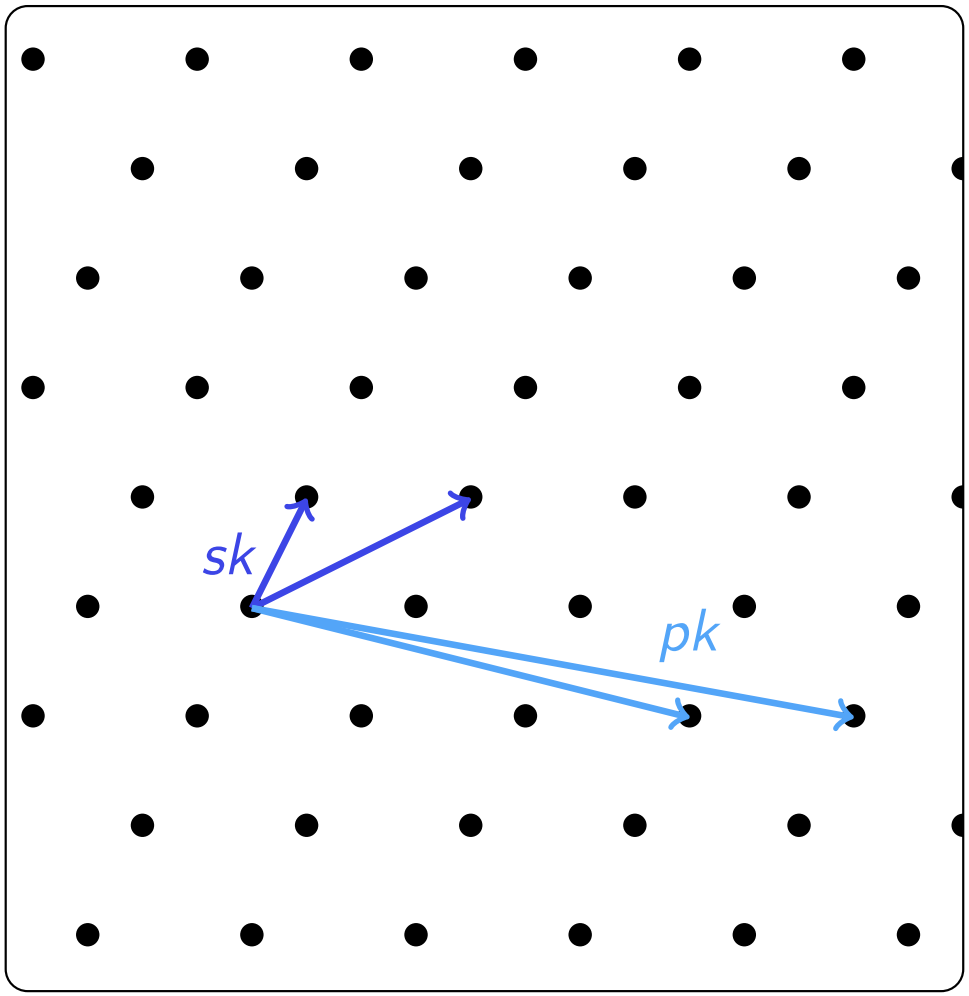
Output: $s = 4 \cdot b_1 - 1 \cdot b_2$

The smaller the basis, the closer the solution

(called Babai's round-off algorithm)

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

Hash-and-sign: first idea [GGH97]

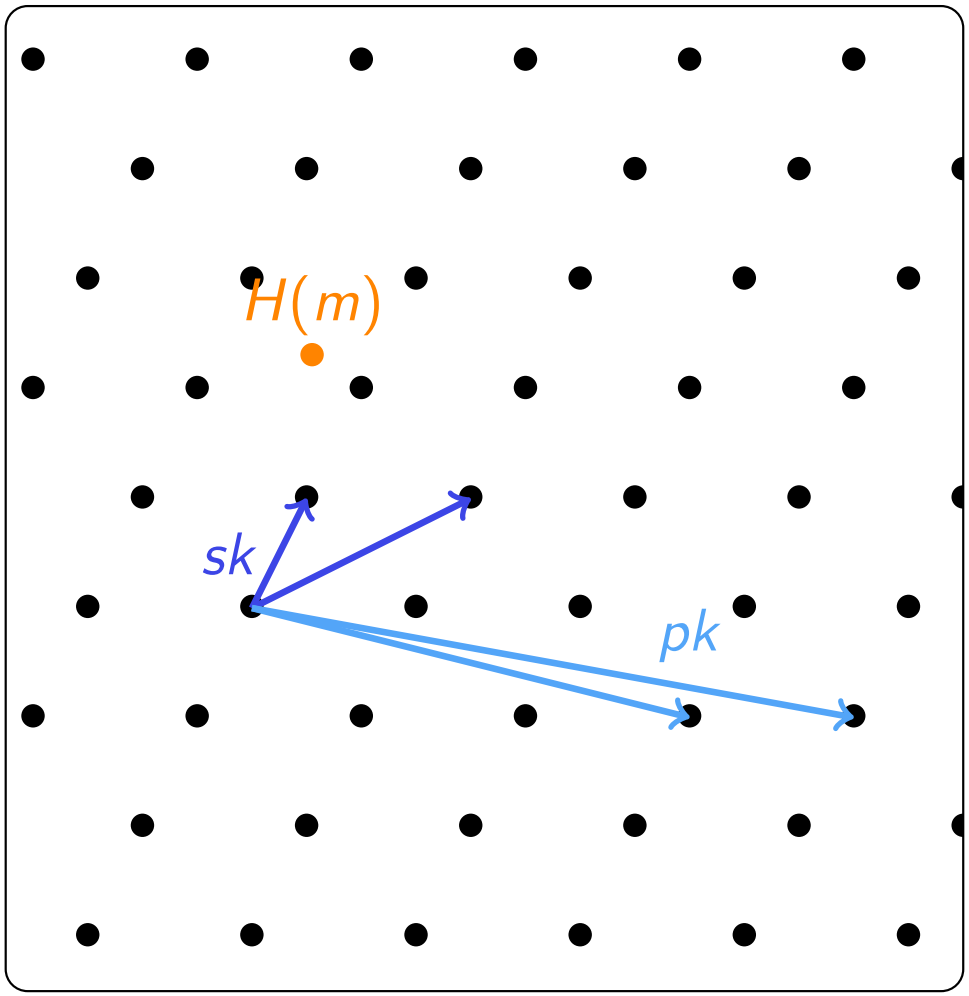


KeyGen:

- 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:

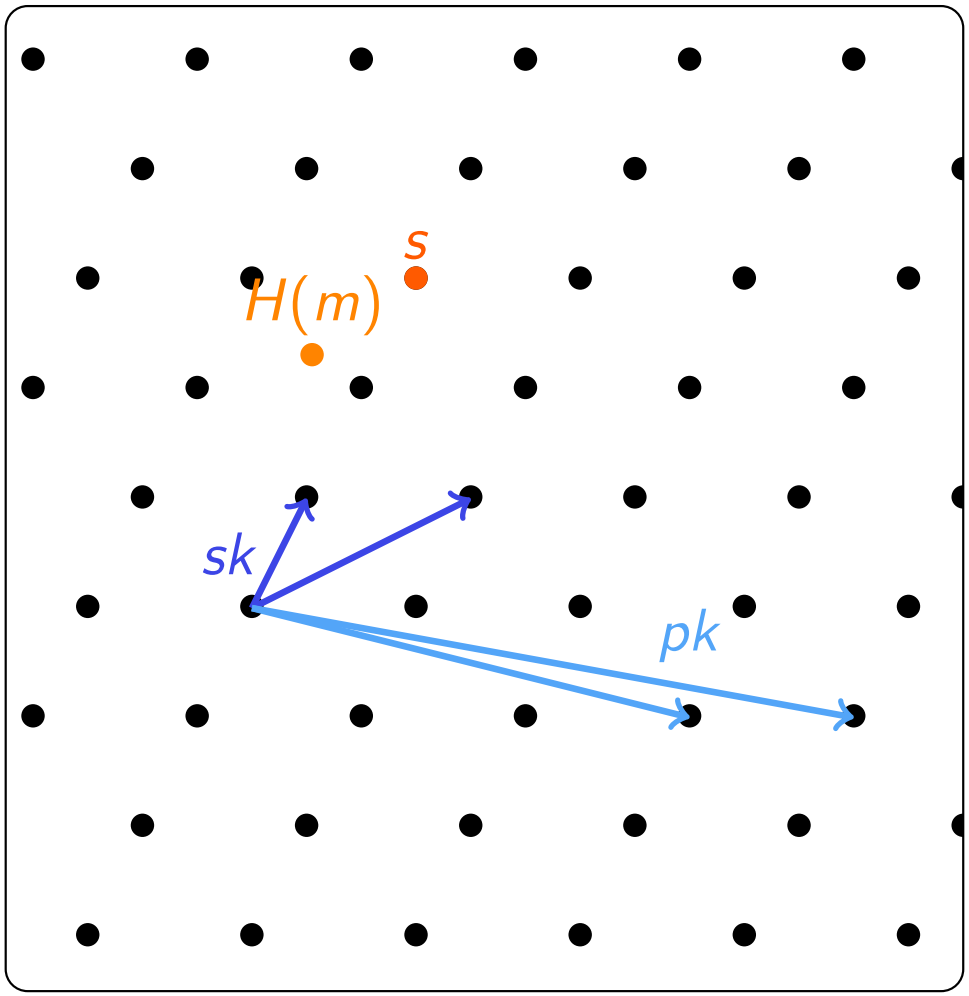
- pk = bad basis of $\mathcal{L}$
- sk = short basis of $\mathcal{L}$

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:

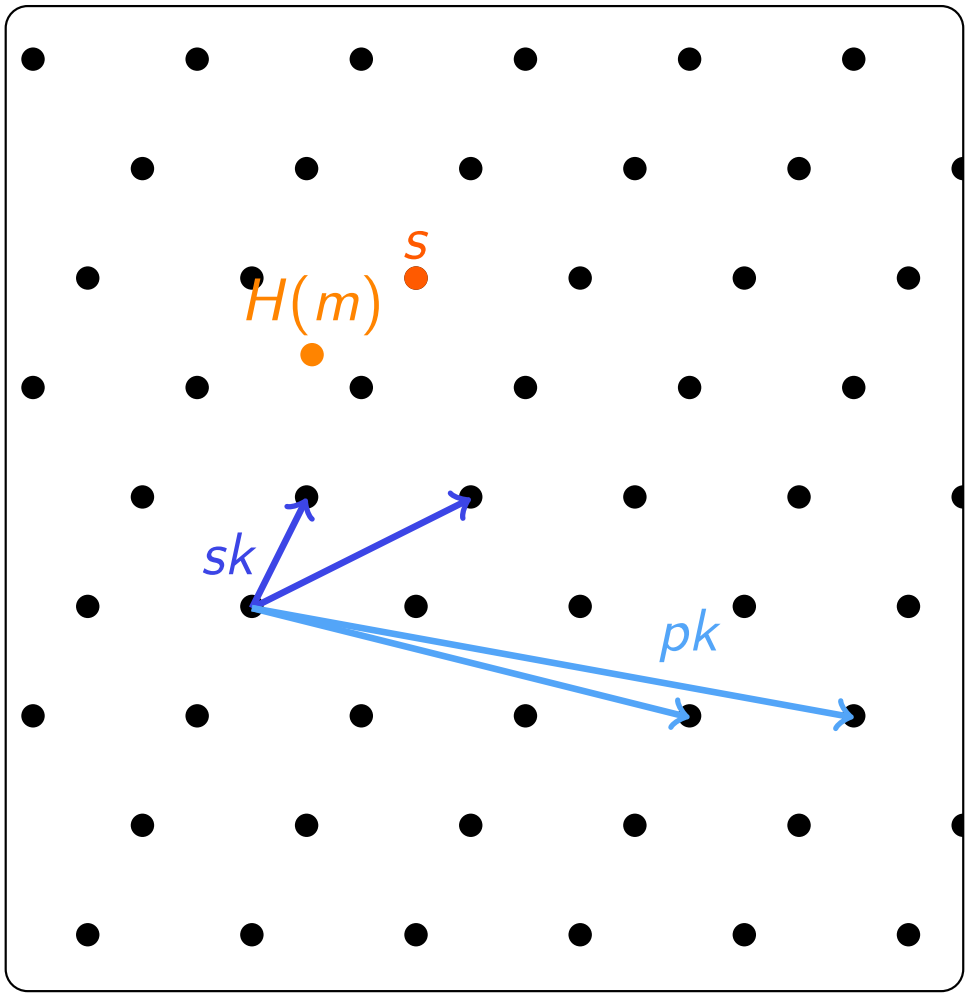
- pk = bad basis of $\mathcal{L}$
- sk = short basis of $\mathcal{L}$

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):

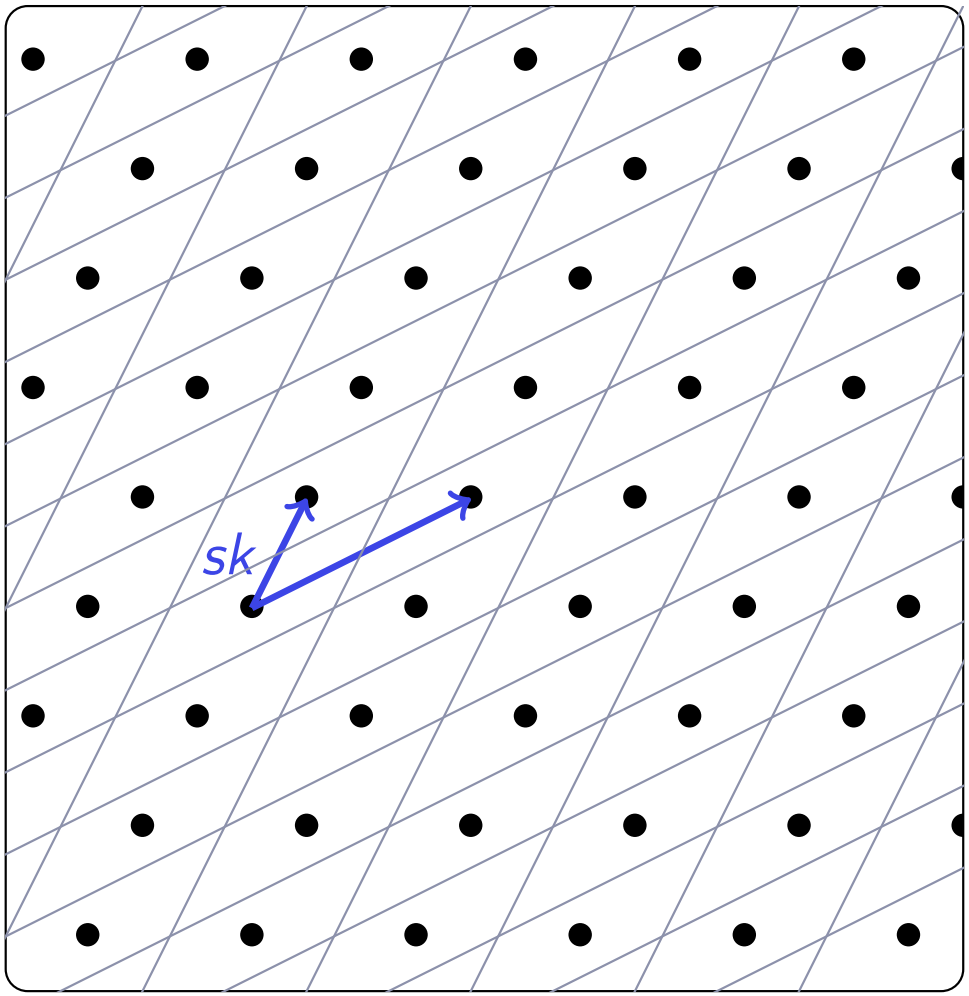
- $x = H(m)$ (hash the message)
- output $s \in \mathcal{L}$ close to x

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

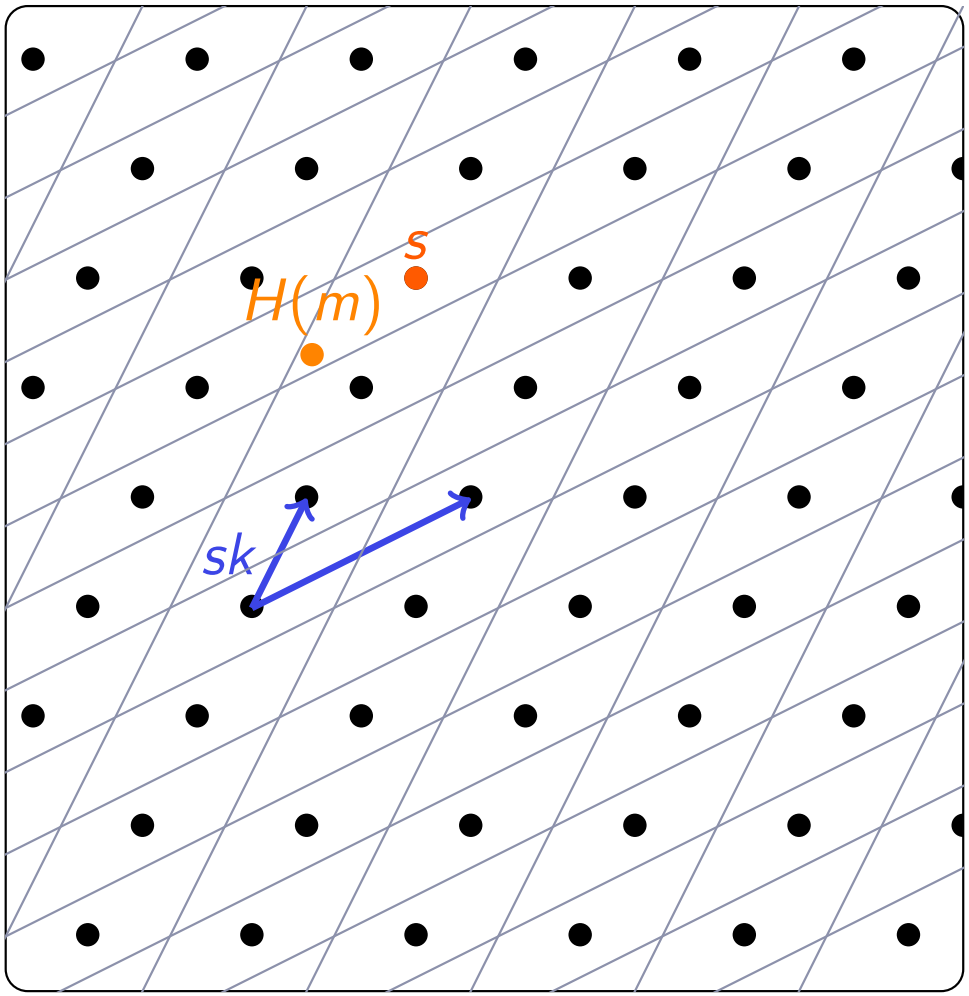
Attack on this first idea [NR06]



Parallelepiped attack:

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

Attack on this first idea [NR06]

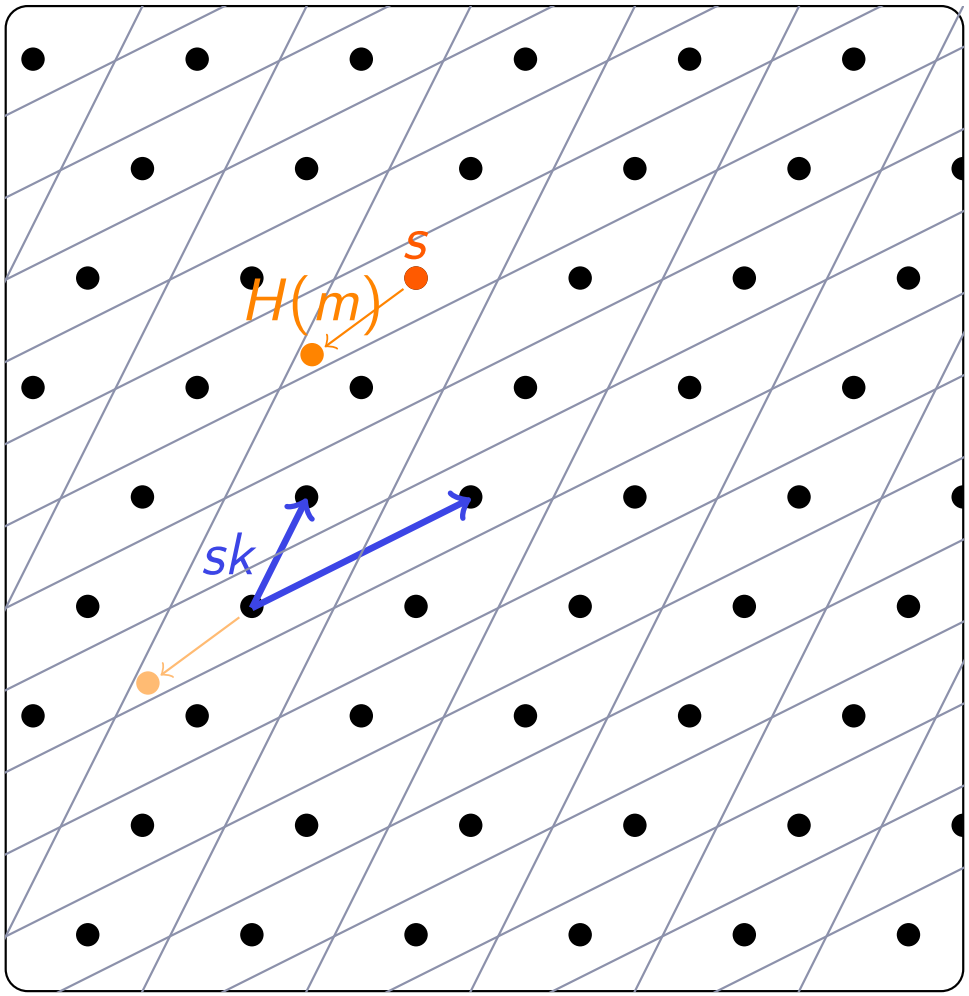


Parallelepiped attack:

- 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]

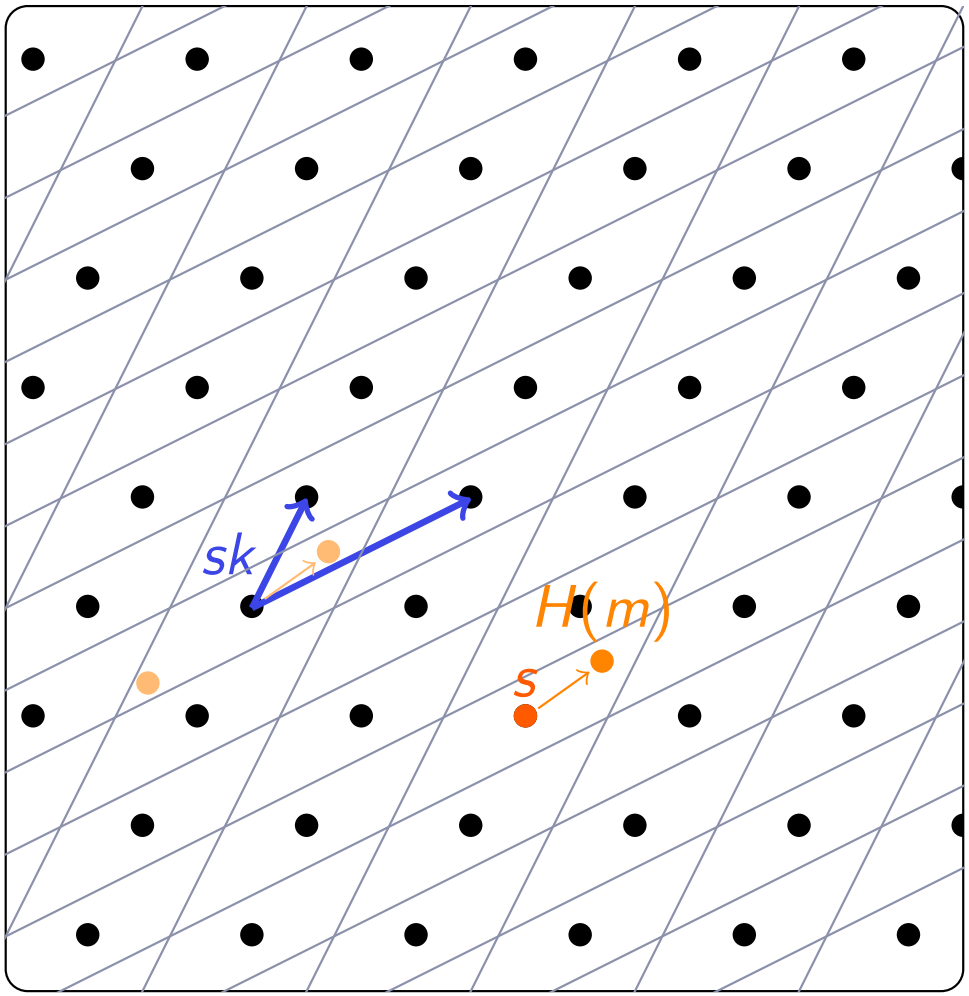


Parallelepiped attack:

- 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]

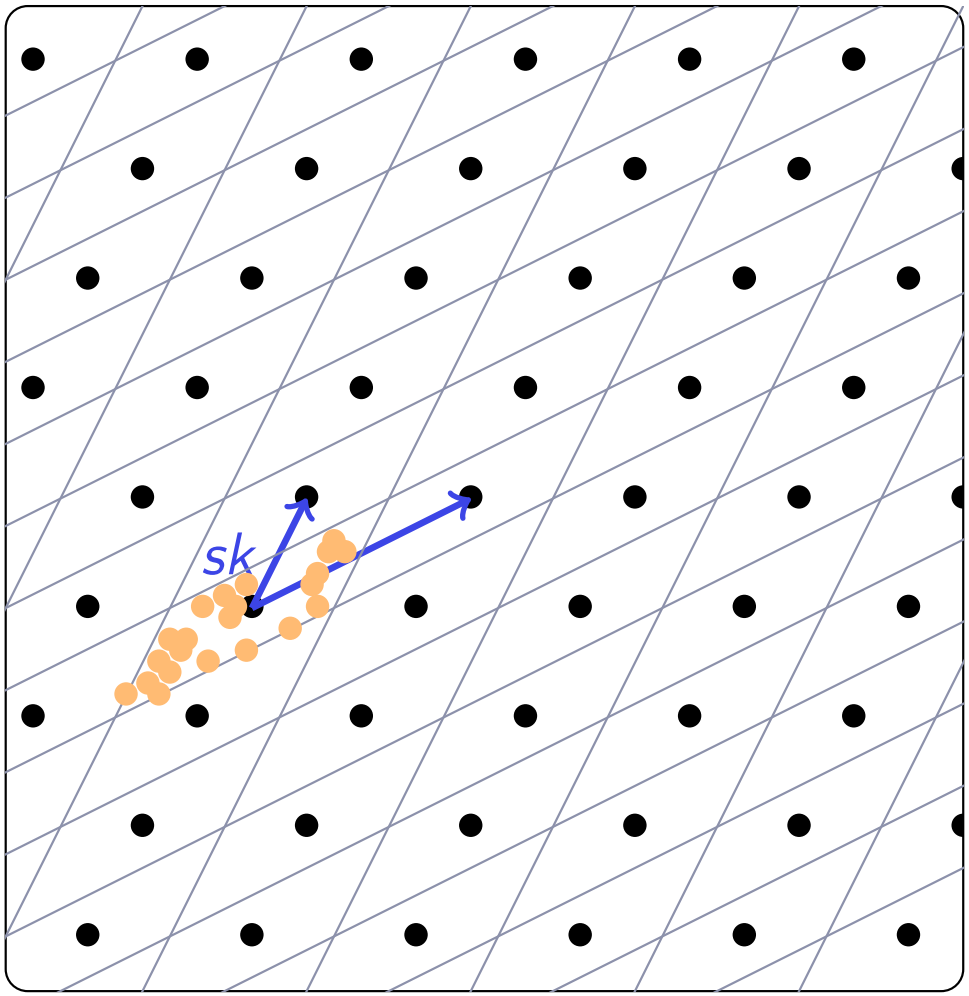


Parallelepiped attack:

- 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]

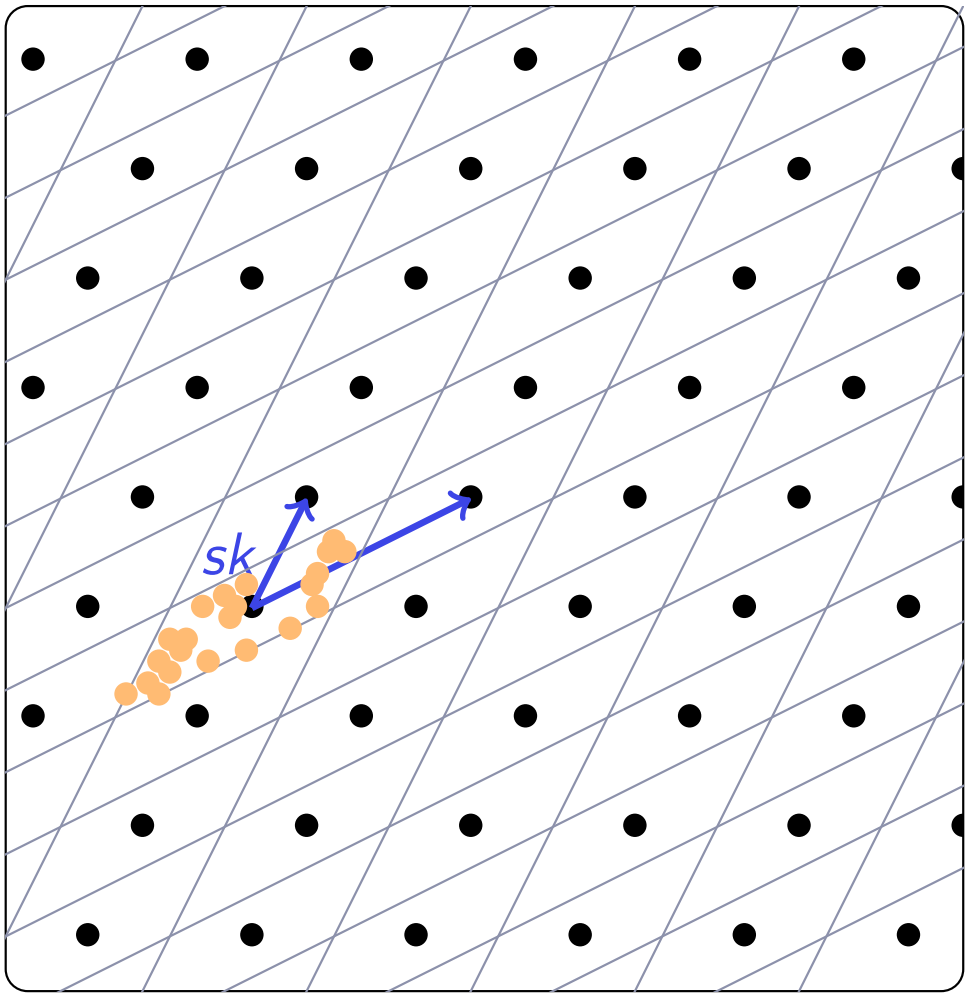


Parallelepiped attack:

- 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]



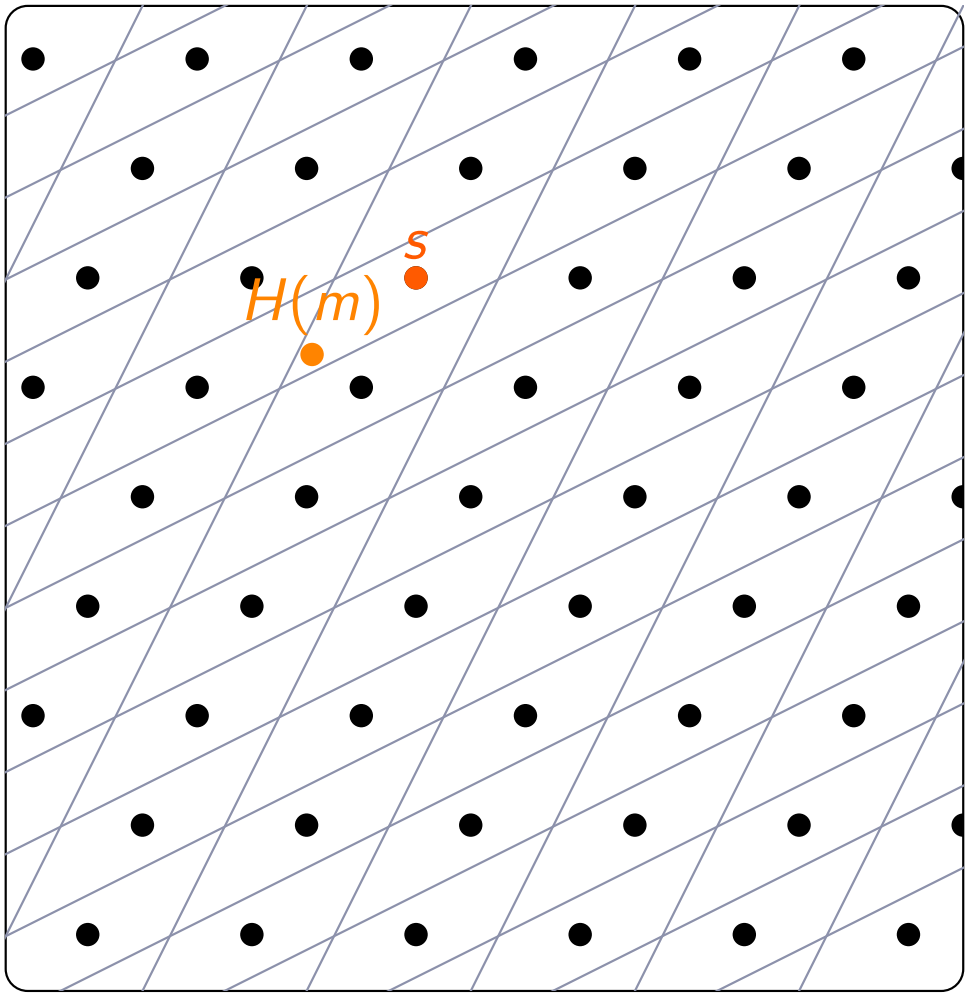
Parallelepiped attack:

- 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

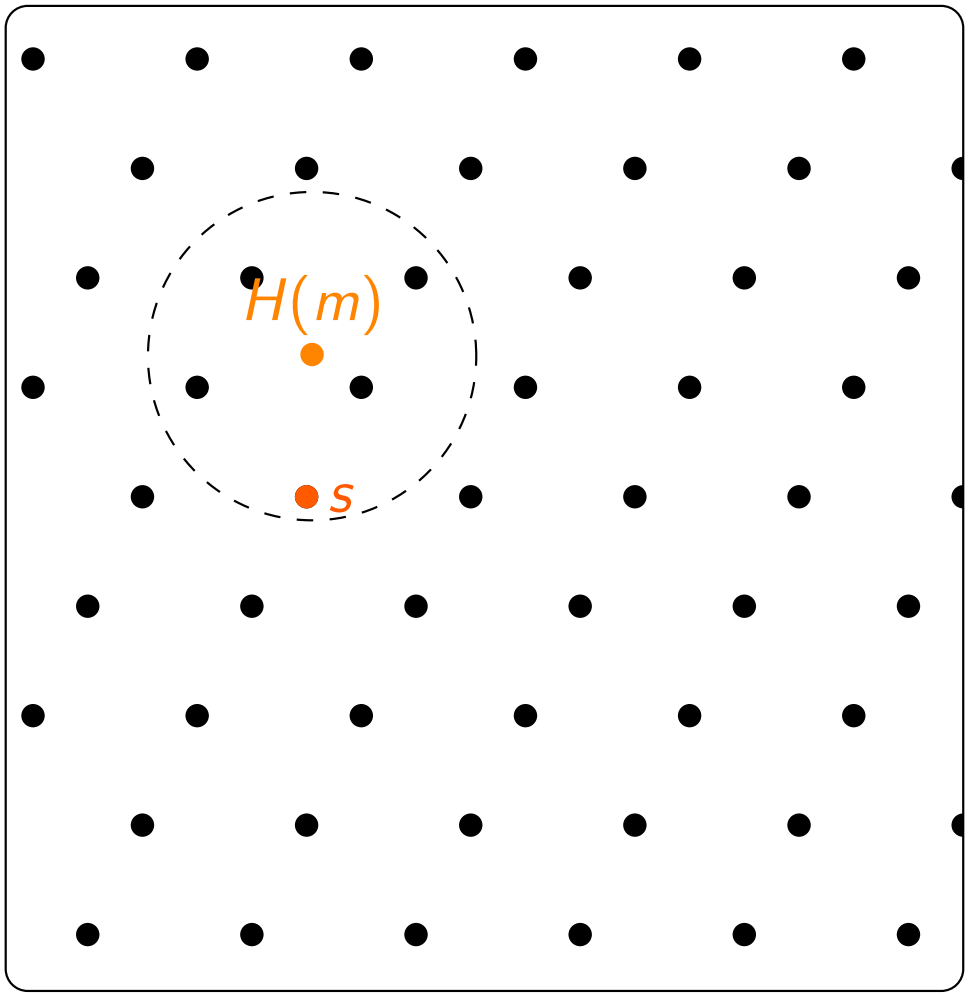
Preventing the attack [GPV08]



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]



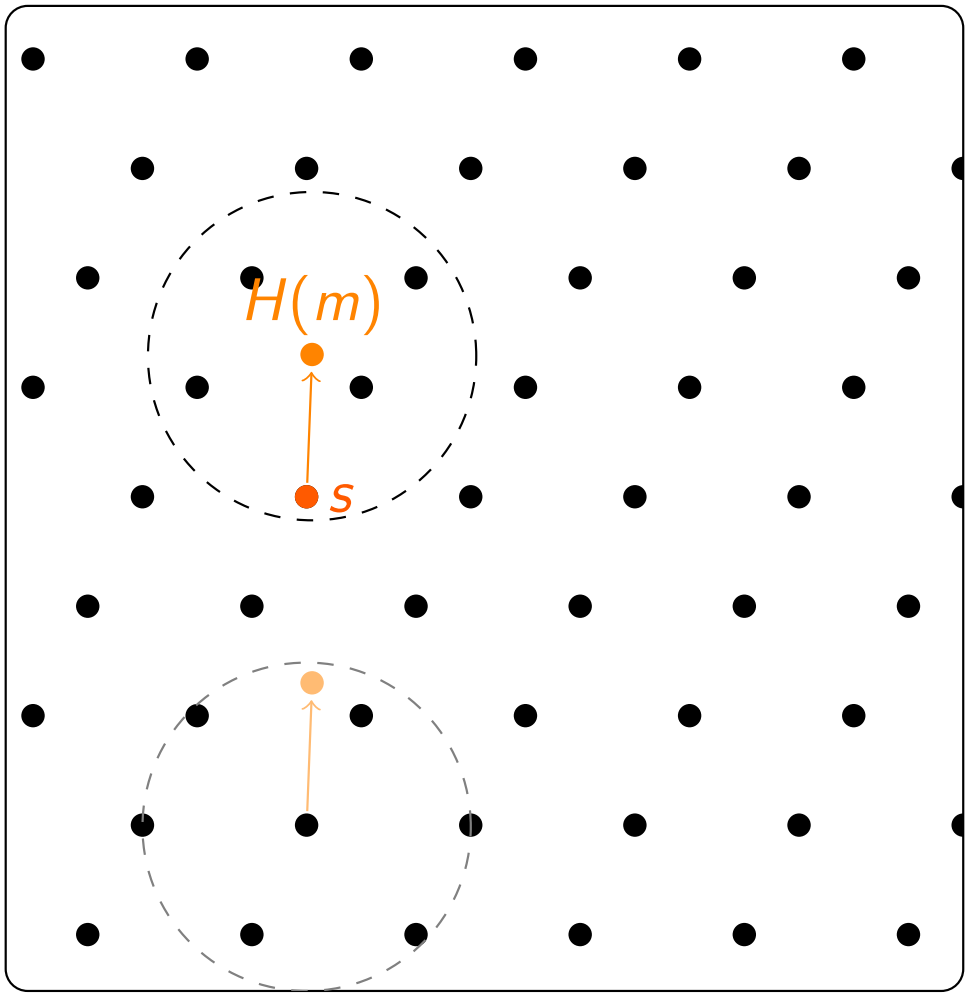
Idea: do not decode deterministically but randomly

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]



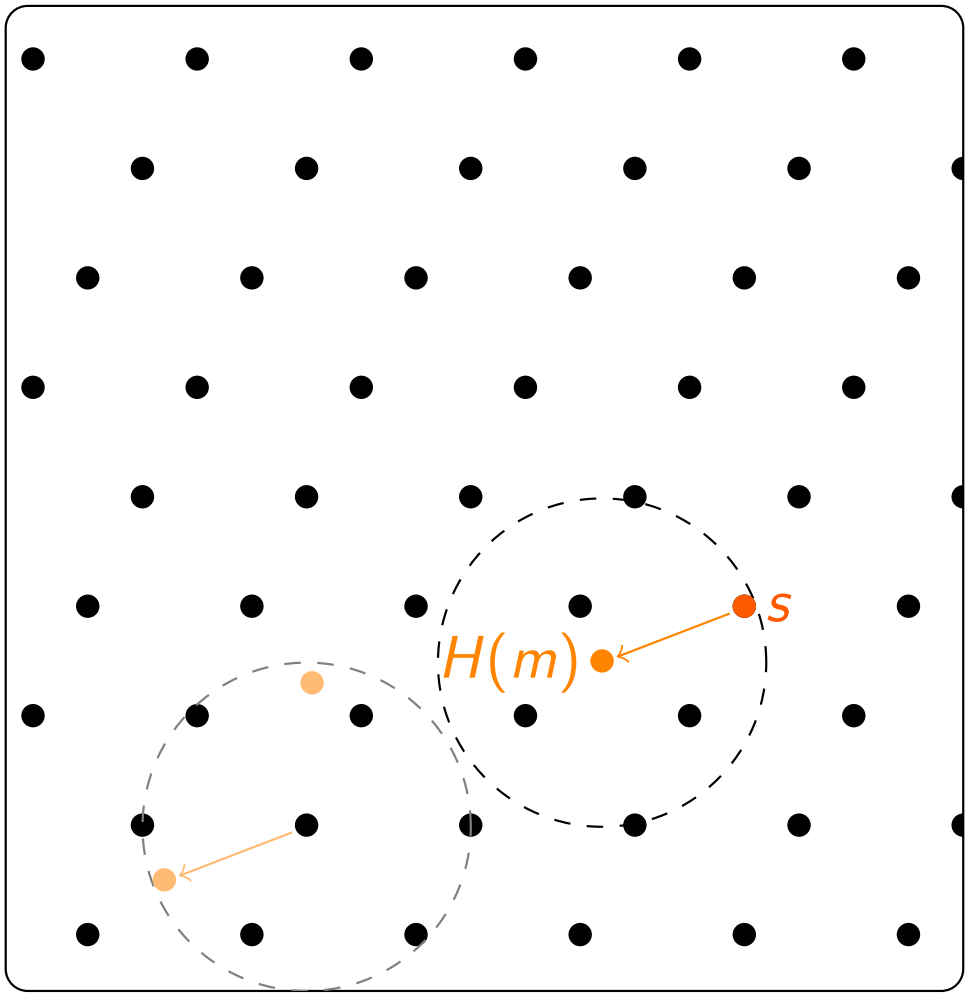
Idea: do not decode deterministically but randomly

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]



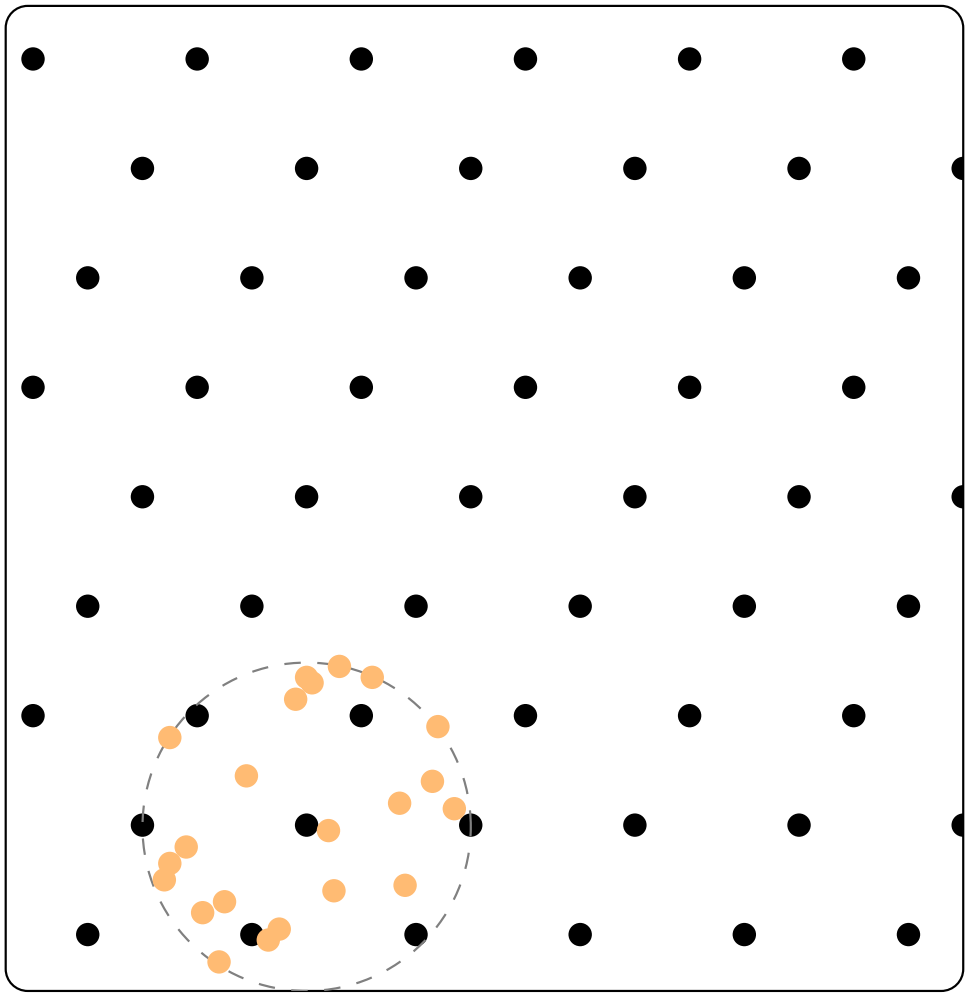
Idea: do not decode deterministically but randomly

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]



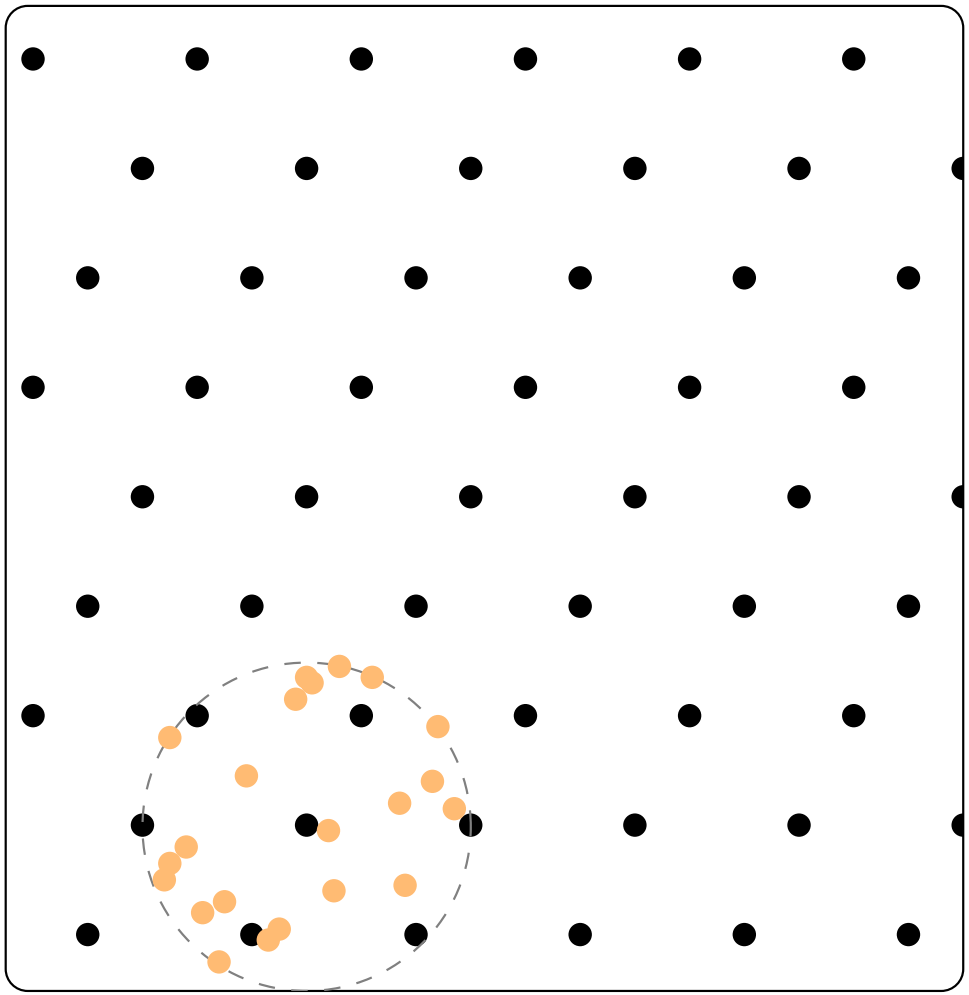
Idea: do not decode deterministically but randomly

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 SIS is hard, the signature scheme is secure (in the ROM)

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

Advanced constructions

One can construct many advanced primitives from lattices:

- (fully) homomorphic encryption
- identity based encryption
- functional encryption for linear functions
- ...