Lattice algorithms for variants of LWE

Shi Bai

Florida Atlantic University, Boca Raton.

Microsoft Research Redmond Cryptography Colloquium, 2019.

Outline

- LWE and SIS problems
- Lattices
- Solving LWE
    - Strategies
    - uSVP/BDD solver (BKZ: quality and time estimates)
- Questions & future work

1. Introduction to LWE/SIS.

LWE

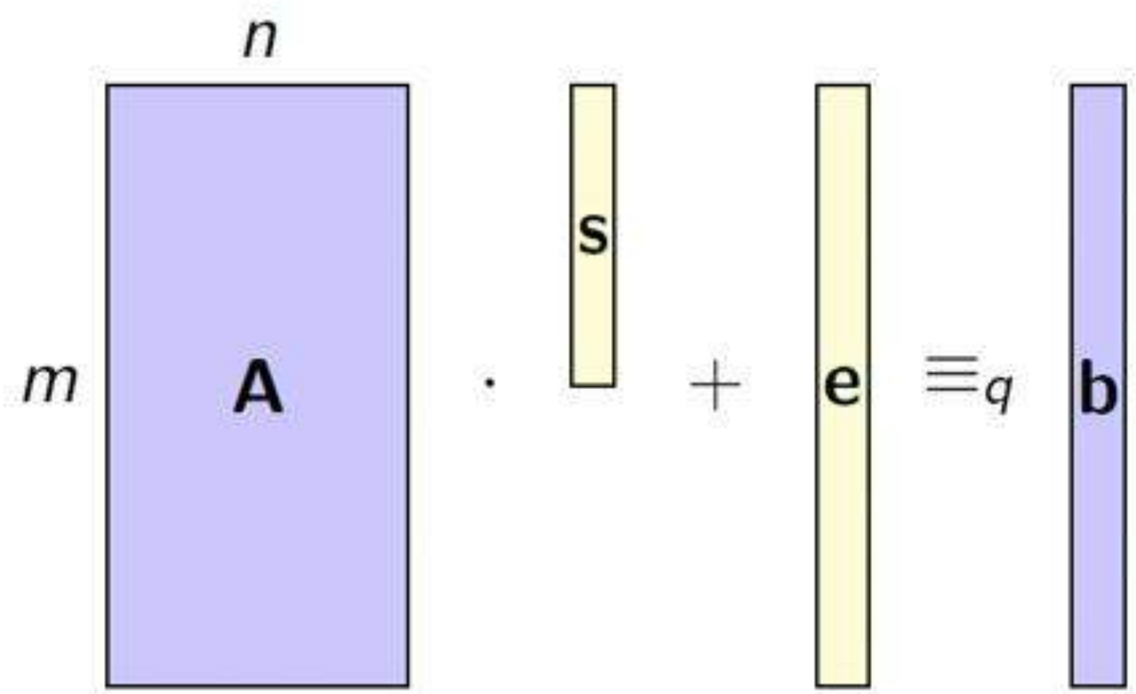
Matrix-form LWE: input security parameter λ, choose parameters n, q, m and two distribution $D_{\mathbf{s}}$, $D_{\mathbf{e}}$.

- sample uniform $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$.
- sample $\mathbf{s}$ according to $D_{\mathbf{s}}$.
- sample "small" $\mathbf{e}$ according to $D_{\mathbf{e}}$.

Problem: given $(\mathbf{A}, \mathbf{b})$ where $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \pmod{\mathbf{q}}$, recover $\mathbf{s}$ (or $\mathbf{e}$).

E.g. $n = \theta(\lambda)$; $q = n^{\Theta(1)}$; $m = \Theta(n \log q)$; $D_{\mathbf{s}}$ uniform on $\mathbb{Z}_q^n$; $D_{\mathbf{e}}$ discrete Gaussian with $\alpha = 2\sqrt{n}/q$.

LWE



- Search version: Given $(\mathbf{A}, \mathbf{b})$, find $\mathbf{s}$ (or $\mathbf{e}$).
- Decisional version: Given samples $(\mathbf{A}, \mathbf{b})$ (either LWE or uniform), decide whether they are LWE samples or uniformly random samples.

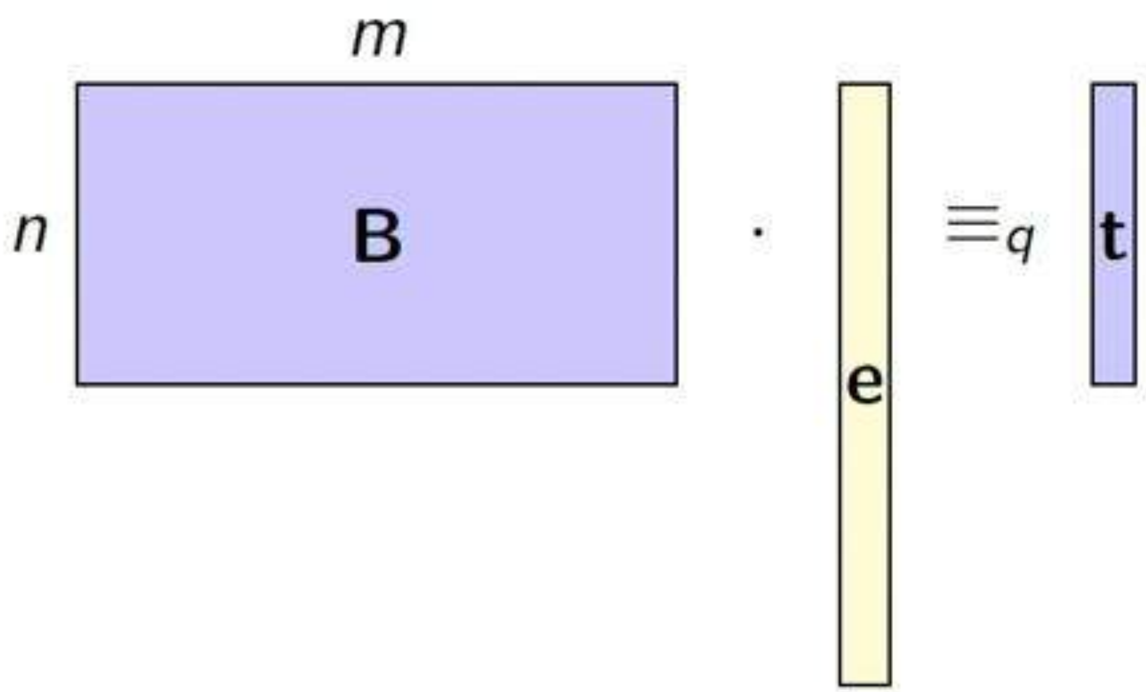
ISIS (inhomogeneous short integer solution)

Matrix-form ISIS: input security parameter λ, choose parameters n, q, m and a distribution $D_{\mathbf{e}}$.

- sample uniform $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$.
- sample "small" $\mathbf{e}$ according to $D_{\mathbf{e}}$.

Problem: given $(\mathbf{B}, \mathbf{t})$ where $\mathbf{t} = \mathbf{Be}$, recover $\mathbf{e}$.

ISIS



LWE variants

Variants of LWE $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e}$.

Distribution of $\mathbf{s}$, $\mathbf{e}$ (“small” means standard):

- $\mathbf{s}$ uniform, $\mathbf{e}$ small.
- Normal form: $\mathbf{s}$ small, $\mathbf{e}$ small.
- variant 1: $\mathbf{s}$ uniform, $\mathbf{e}$ tiny. (e.g. binary error LWE)
- variant 2: $\mathbf{s}$ tiny, $\mathbf{e}$ tiny. (e.g. binary secret-error LWE)
- variant 3: $\mathbf{s}$ tiny, $\mathbf{e}$ small. (e.g. binary secret LWE)
- variant 4: $\mathbf{s}$ sparse, $\mathbf{e}$ small. (e.g. sparse secret LWE)
- More variants: # samples restricted; Modulus q large.

Lots of applications: signatures (Dilithium, qTESLA), KE (Newhope, Kyber) and HE schemes (HElib, SEAL) and many others.

Combinations of distributions, number of samples, e.g. give variants with various difficulty – **concrete security level** needs to be carefully analyzed.

Three types of algorithms

- Algebraic: Arora-Ge algorithm and variants.
- Combinatoric: Blum-Kalai-Wasserman (BKW) algorithm and variants.
- Geometric: phrase the LWE instance as some lattice problem and solve this problem using lattice solvers (e.g. lattice reduction or sieving).

More attacks on structured LWE problem (of the aforementioned variants) using algebraic structure: Elias-Lauter-Ozman-Stange '15; Chen-Lauter-Stange '16, '17; Castryck-Iliashenko-Vercauteren '16; Peikert '16.

We will focus on LWE in general q-ary lattice in this talk.

Algebraic algorithms

Algebraic attacks, e.g. Arora-Ge algorithm and variants.

- Binary error LWE:
  - Poly. with samples $m = O(n^2)$.
  - Subexp. with samples $m = O(n \log\log n)$ (Albrecht-Cid-Faugere-Perret 14').
  - $m \gtrapprox n$ for the reduction to SIVP-γ with poly. γ in dimension $\Theta(n/\log n)$ (Micciancio-Peikert '13).
- Tiny error LWE:
  - $2^{\tilde{O}(\alpha^2 q^2)}$; thus already subexp for $\alpha q < \sqrt{n}$ with enough (same) samples.
- Small (standard) error LWE:
  - $2^{O(n \log\log n)}$ for $\alpha q = \sqrt{n}$. Slower than sieving/BKW.
  - $2^{O(n)}$ using Gröbner basis (Albrecht-Cid-Faugere-Perret 14').

For full power, need # Samples $\approx n^{2w}$ where w bounds the width of error.

Combinatoric algorithms

Combinatoric attacks: BKW-like algorithms (Blum-Kalai-Wasserman '99).

- LPN ($q = 2$):
  - $2^{O(n/\log n)}$ samples/time.
  - $2^{O(n/\log\log n)}$ time with $n^{1+\epsilon}$ samples (Lyubashevsky '05).
- Binary secret-error LWE:
  - $2^{O(n/\log\log n)}$ time with n samples (Kirchner-Fouque '15).
- LWE: $2^{O(n)}$ (Albrecht-Faugere-Fitzpatric-Perret '14).

Also meet-in-the-middle type algorithms: useful for sparse $\mathbf{s}$ or $\mathbf{e}$.

Geometric algorithms

Geometric methods: turn the LWE into a problem on lattices (tools: lattice reduction and lattice sieving).

Feature: # LWE samples are usually small.

Quick summary (asymptotic running-time):

- LWE: $2^{O(n)}$.
- Binary secret-error LWE: $2^{O(n)}$.
- Binary secret but small error LWE: $2^{O(n)}$.
- Binary error but small/uniform secret LWE: $2^{O(n)}$.
- Tiny error but larger modulus, polynomial time (Laine-Lauter '15).

Thus, these lattice algorithms are mostly relevant in terms of concrete security levels.

2. Lattices

Euclidean lattice

An integral lattice can be defined as the $\mathbb{Z}$-linear combination of n independent vectors $\mathbf{b}_i \in \mathbb{Z}^n$

$$L(\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_n) = \left\{ \sum_{i=1}^{n} \mathbb{Z}\mathbf{b}_i \right\}.$$

Let $\mathbf{B} = [\mathbf{b}_1\, \mathbf{b}_2\, \cdots\, \mathbf{b}_n]$ then $\Lambda = L(\mathbf{B}) = \{\mathbf{B}\mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^n\}$.

Euclidean lattice

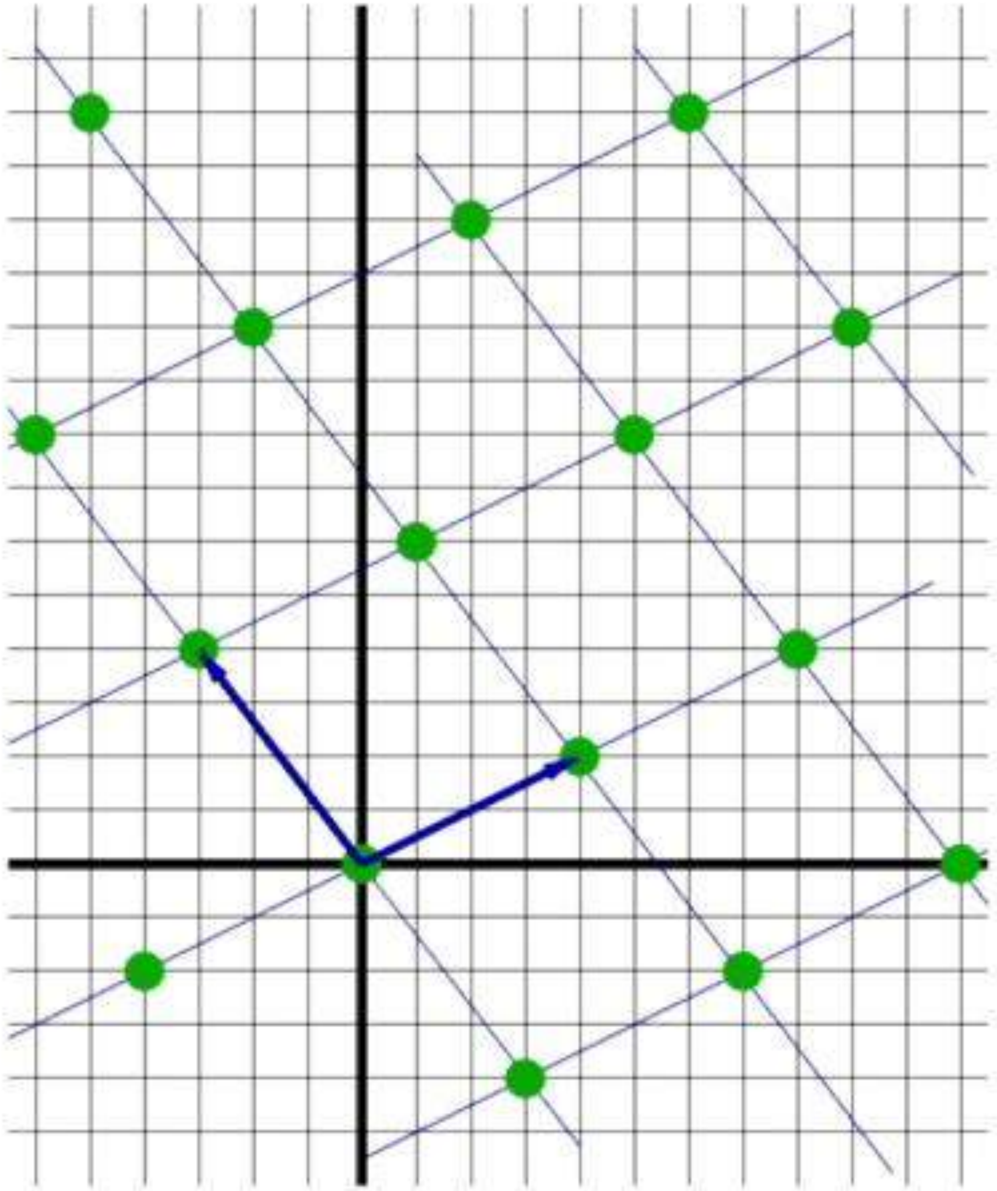
An integral lattice can be defined as the $\mathbb{Z}$-linear combination of n independent vectors $\mathbf{b}_i \in \mathbb{Z}^n$

$$L(\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_n) = \left\{ \sum_{i=1}^{n} \mathbb{Z}\mathbf{b}_i \right\}.$$

Let $\mathbf{B} = [\mathbf{b}_1 \, \mathbf{b}_2 \, \cdots \, \mathbf{b}_n]$ then $\Lambda = L(\mathbf{B}) = \{\mathbf{B}\mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^n\}$.

The volume of a lattice Λ is $|\det(\mathbf{B})|$, which is independent of the choice of the basis.

Lattice minimum

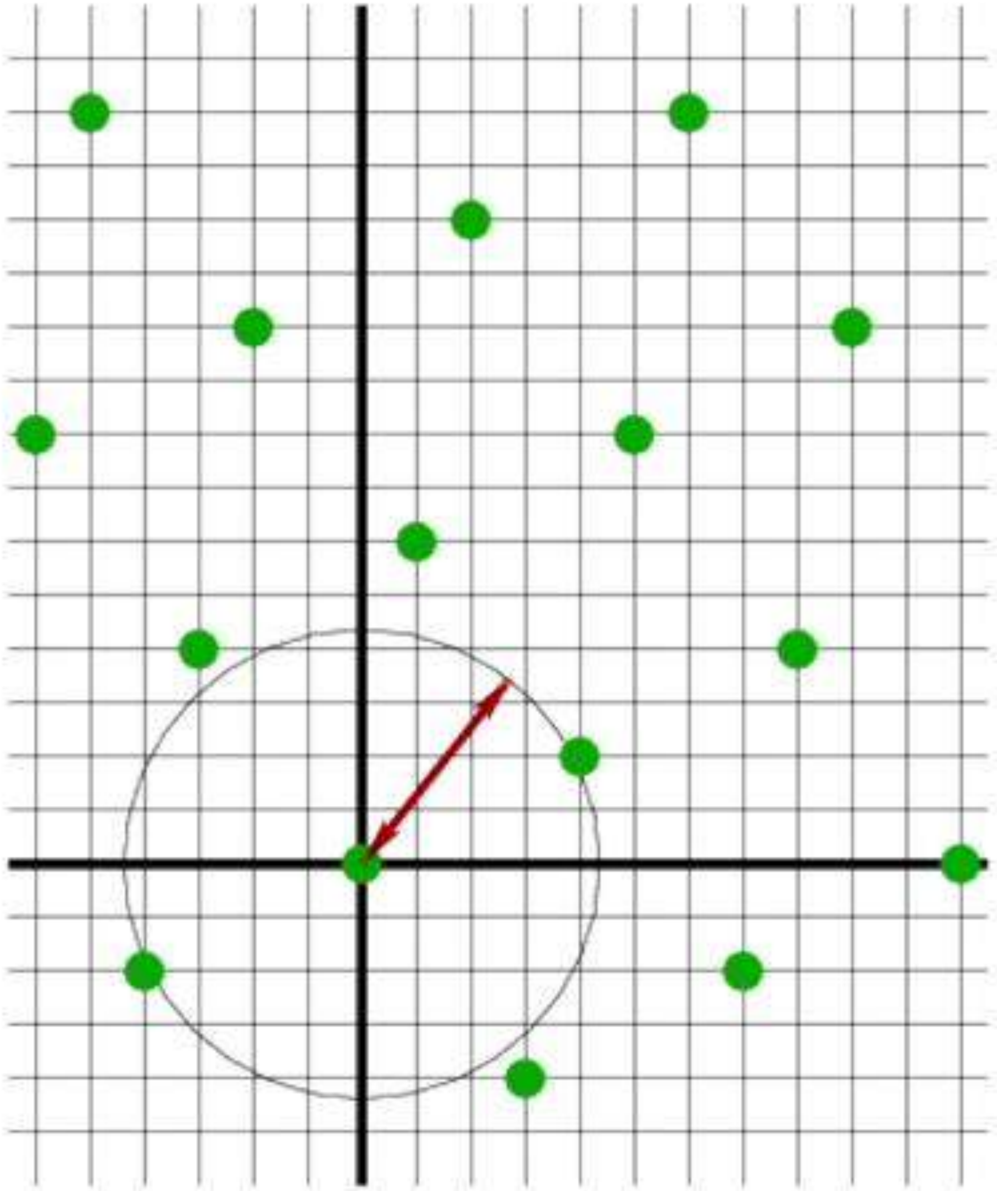


Lattice minimum

$\lambda_1(\Lambda) = \min\ (\ \|\mathbf{b}\|\colon \mathbf{b} \in \Lambda \setminus \mathbf{0}\ )$

Lattice minimum

Lattice minimum
$\lambda_1(\Lambda) = \min\ (\ \|\mathbf{b}\|\colon \mathbf{b} \in \Lambda \setminus \mathbf{0}\ )$

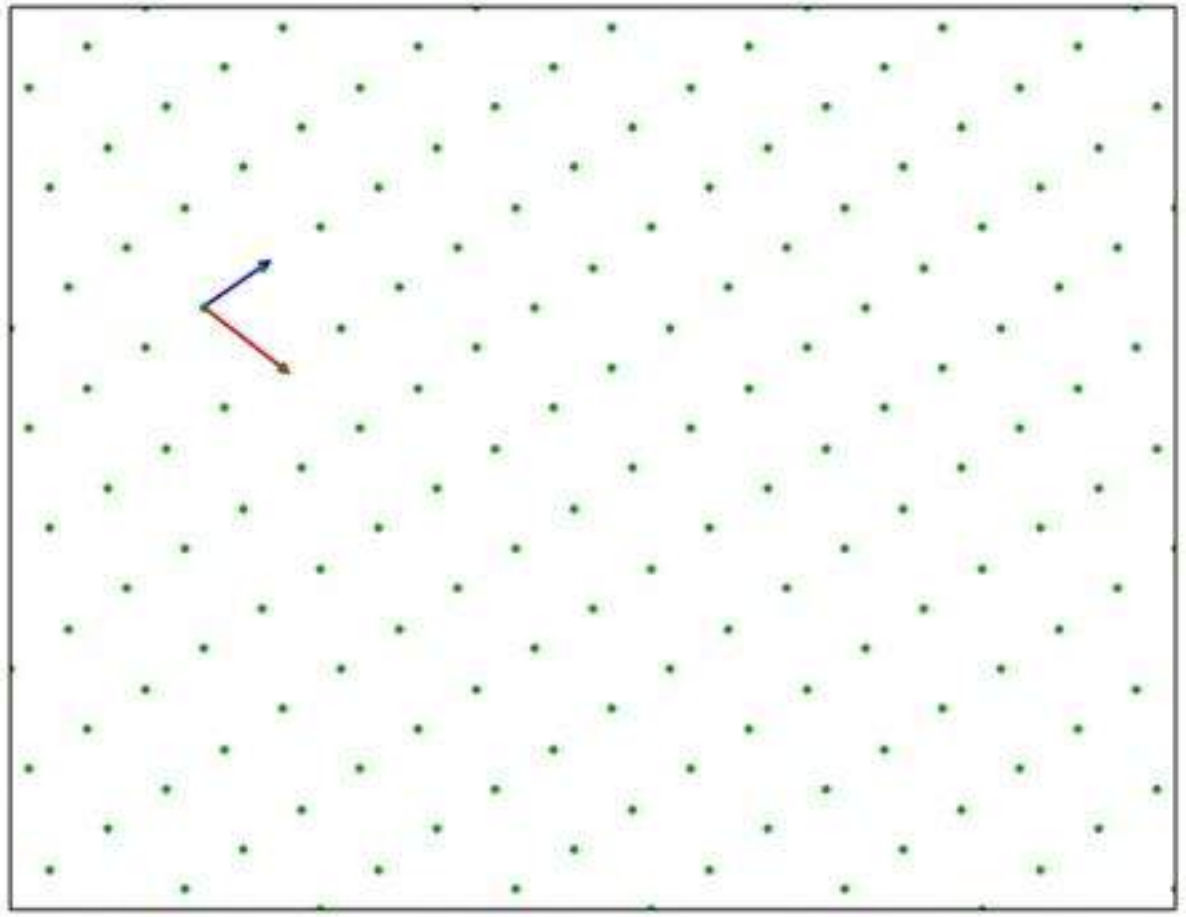


Computational problems for lattices

Shortest vector problem (SVP)

Input: $\mathbf{B} \in \mathbb{Z}^{n \times n}$ a basis matrix of Λ.
Output: $\mathbf{s} \in \Lambda \setminus \mathbf{0}$ shortest.

The difficulty **heavily** depends on the "shape" of the input basis **B**. In cryptography, a "bad" **B** is given.

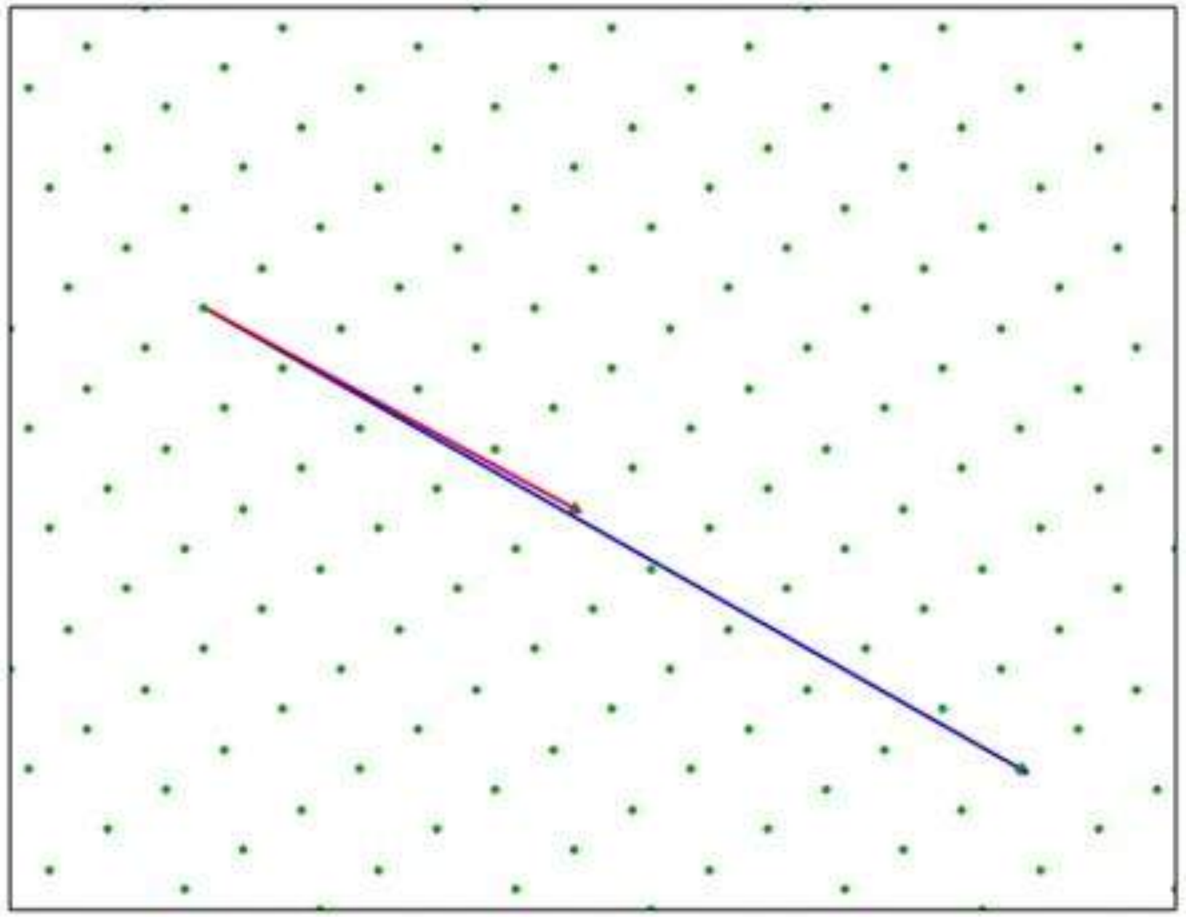


Computational problems for lattices

Shortest vector problem (SVP)

Input: $\mathbf{B} \in \mathbb{Z}^{n \times n}$ a basis matrix of Λ.
Output: $\mathbf{s} \in \Lambda \setminus \mathbf{0}$ shortest.

The difficulty **heavily** depends on the "shape" of the input basis **B**. In cryptography, a "bad" **B** is given.

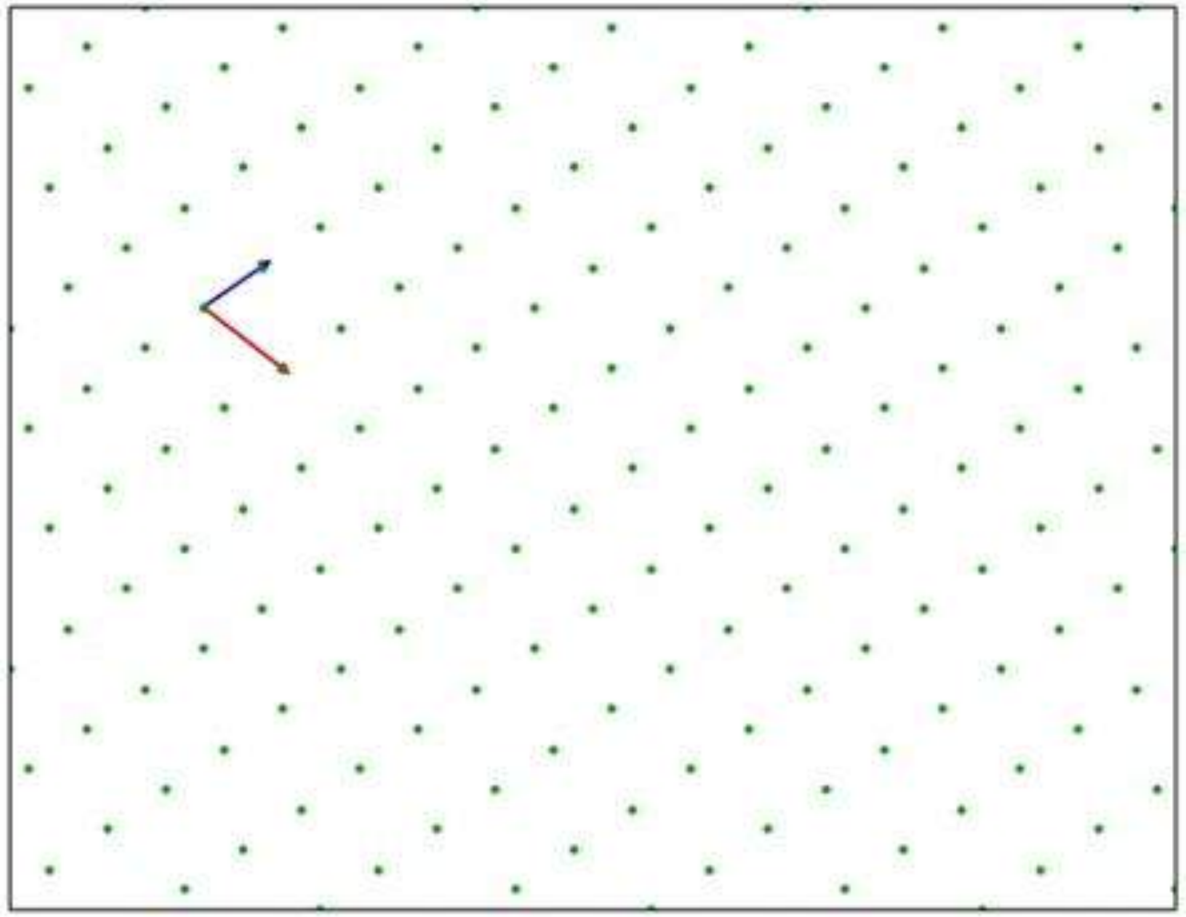


Computational problems for lattices

Shortest vector problem (SVP)

Input: $\mathbf{B} \in \mathbb{Z}^{n \times n}$ a basis matrix of Λ.
Output: $\mathbf{s} \in \Lambda \setminus \mathbf{0}$ shortest.

The difficulty **heavily** depends on the “shape” of the input basis **B**. In cryptography, a “bad” **B** is given.

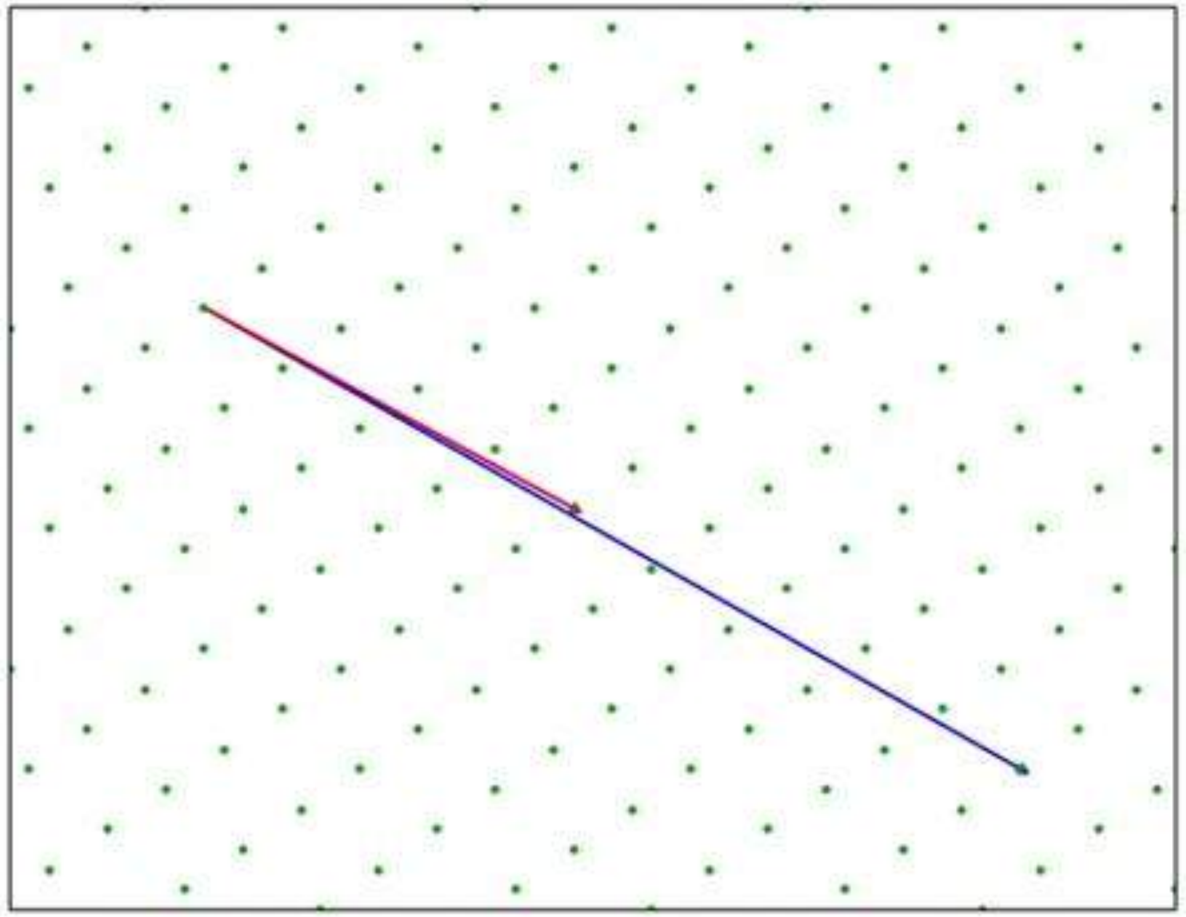


Computational problems for lattices

Shortest vector problem (SVP)

Input: $\mathbf{B} \in \mathbb{Z}^{n \times n}$ a basis matrix of Λ.
Output: $\mathbf{s} \in \Lambda \setminus \mathbf{0}$ shortest.

The difficulty **heavily** depends on the "shape" of the input basis **B**. In cryptography, a "bad" **B** is given.



SVP-γ

Cryptography needs to use a relaxed version: SVP-γ s.t. γ depends on n.

- SVP-1: enumeration or sieving.
- SVP-γ: Block Korkine-Zolotarev (BKZ) reduction.
  - γ is exponential in n: Lenstra-Lenstra-Lovász (LLL) algorithm.
  - γ is polynomial/sub-exponential in n: cryptography.

Algorithms for SVP-γ and algorithms for SVP-1 are reciprocal.

SVP-γ

Cryptography needs to use a relaxed version: SVP-γ s.t. γ depends on n.

- SVP-1: enumeration or sieving.
- SVP-γ: Block Korkine-Zolotarev (BKZ) reduction.
  - γ is exponential in n: Lenstra-Lenstra-Lovász (LLL) algorithm.
  - γ is polynomial/sub-exponential in n: cryptography.

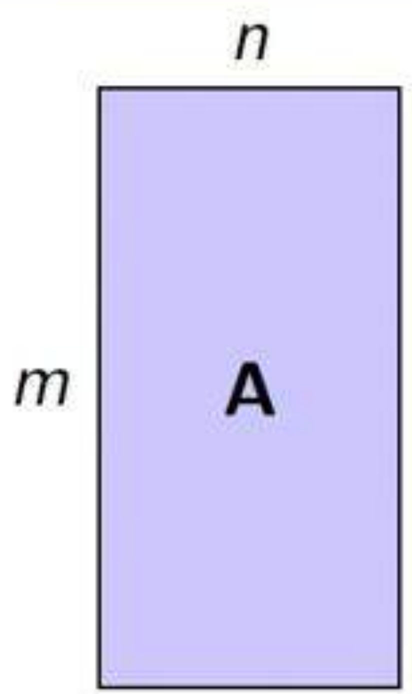
Algorithms for SVP-γ and algorithms for SVP-1 are reciprocal.

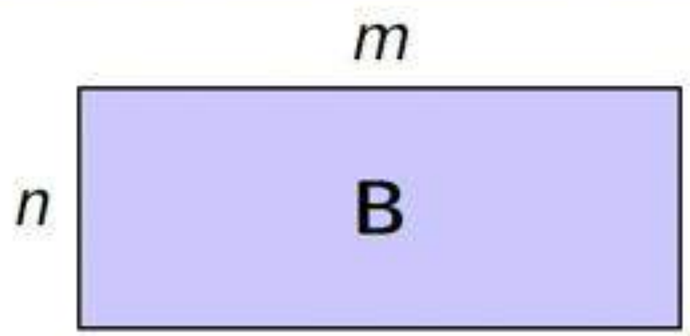
(1) Best algorithm for SVP-1:

- Enumeration $2^{O(n \log n)}$ (Kannan-Fincke-Pohst '83).
- Sieving $2^{O(n)}$ (Ajtai-Kumar-Sivakumar '01).

(2) Best algorithm for SVP-γ: BKZ whose complexity is dominated by SVP-1 in smaller dimensions.

Two q-ary lattices: $m > n$





$$\Lambda_q^\perp(\mathbf{B}) = \{\mathbf{e} \in \mathbb{Z}^m \mid \mathbf{Be} = \mathbf{0} \pmod{q}$$

$$\Lambda_q(\mathbf{A}) = \{\mathbf{y} \mid \mathbf{y} = \mathbf{Ax} \pmod{q}, \forall \mathbf{x} \in \mathbb{Z}^n\}$$

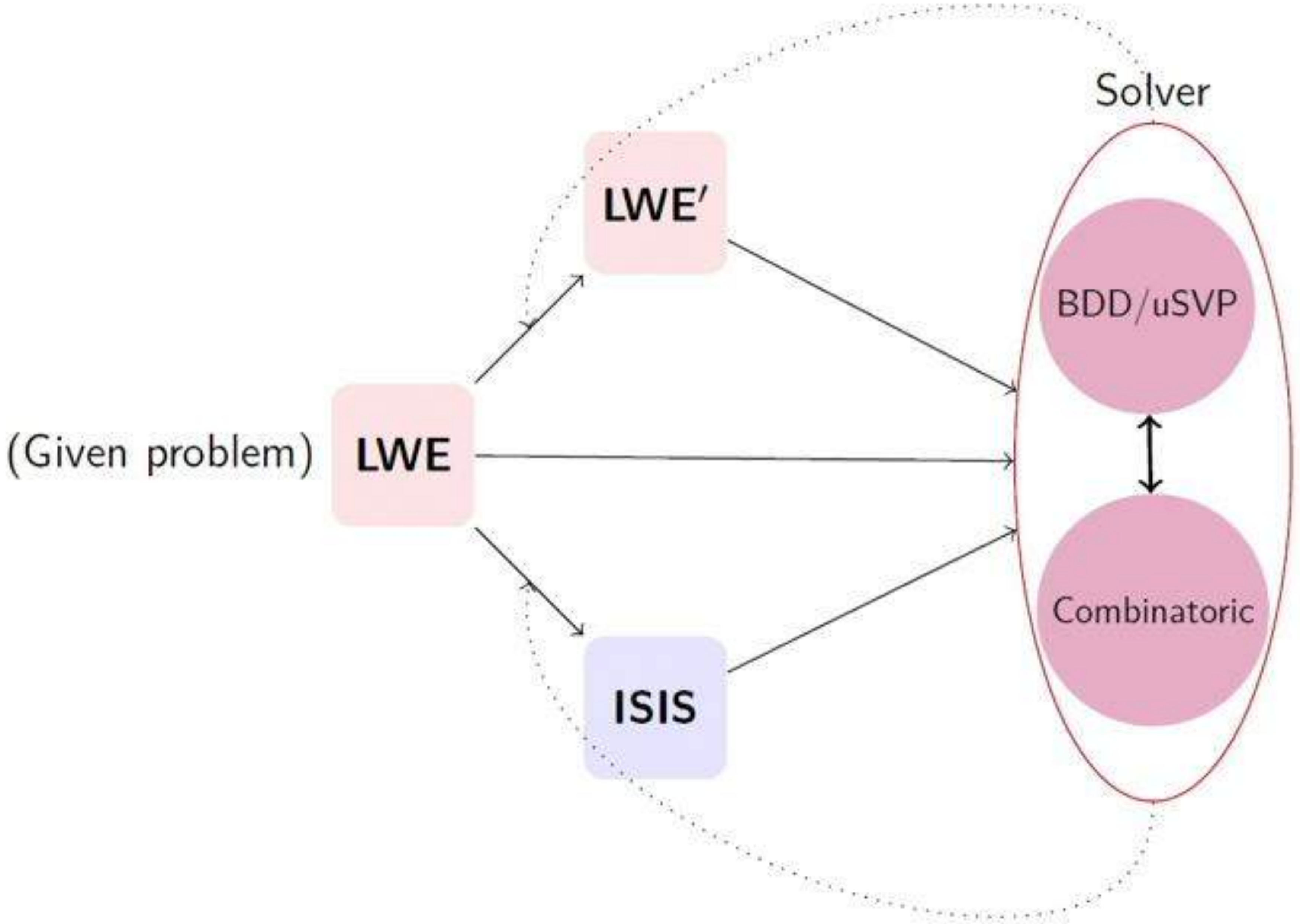
Sometimes, $\mathbf{B} = \mathbf{A}^T$. In such case,

$$\Lambda_q^\perp(\mathbf{A}^T) = q \cdot \Lambda_q(\mathbf{A})^*$$

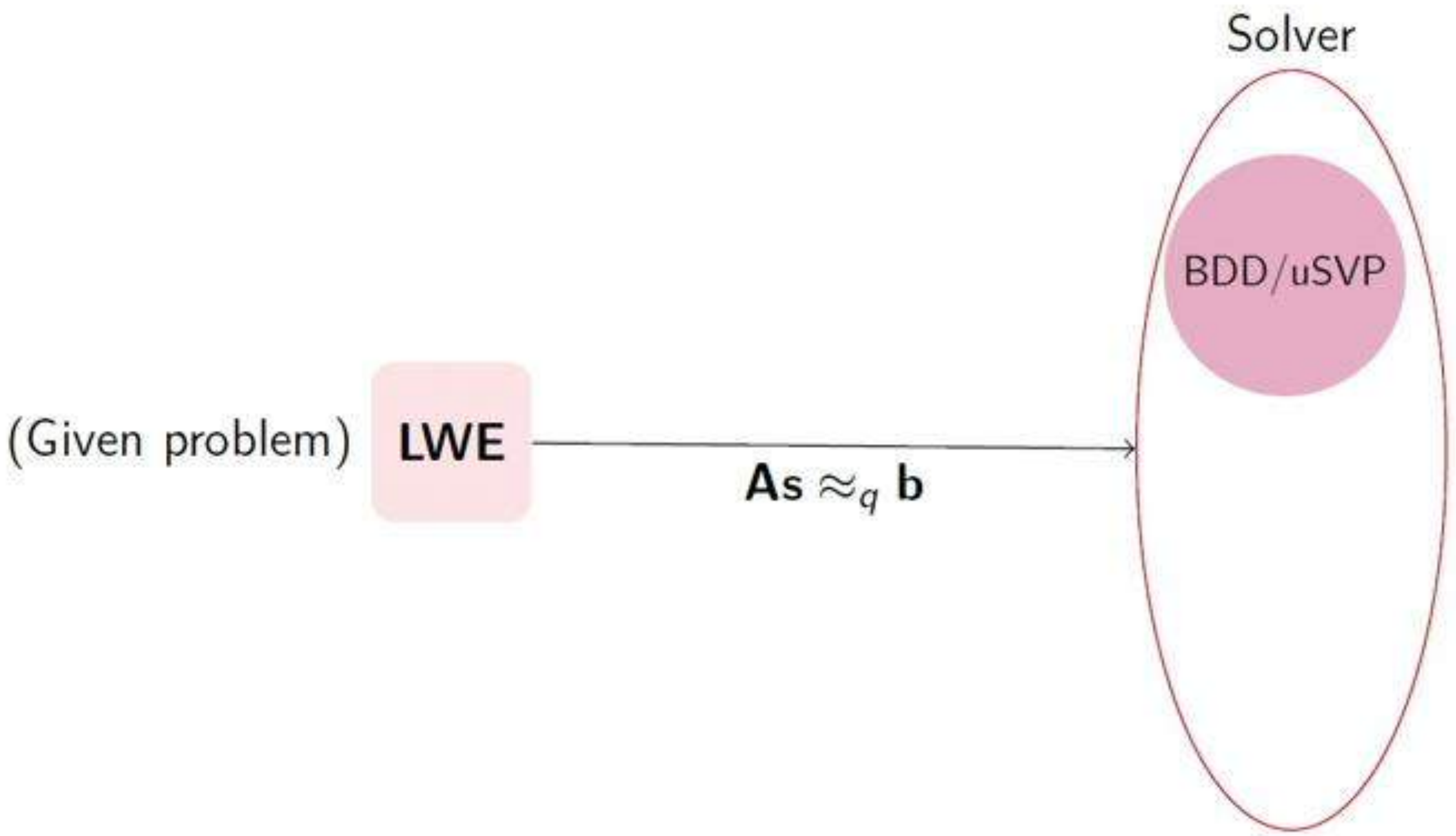
where $\Lambda_q(\mathbf{A})^*$ is the dual lattice of $\Lambda_q(\mathbf{A})$.

Refer them as: image/column lattice and kernel lattice.

3. Lattice algorithms for LWE: summary of strategies.


Solver
LWE′
BDD/uSVP
(Given problem)
LWE
Combinatoric
ISIS

First strategy (primal): direct decoding attack.



First strategy (primal): direct decoding attack.

Main idea: $\mathbf{e}$ is short.

The image lattice $\Lambda_q(\mathbf{A}) = \{\mathbf{y} \mid \mathbf{y} = \mathbf{A}\mathbf{x} \pmod{q}, \forall \mathbf{x} \in \mathbb{Z}^n\}$.

- Solve CVP/BDD on $\Lambda_q(\mathbf{A})$ given target point $\mathbf{b}$.
- The lattice has rank m and volume q^{m-n}.
- Convert to uSVP using Kannan's embedding.
- The concrete security depends on the number of samples m given.
- For best asymptotics, $m \approx -\frac{n \log q}{\log \alpha}$.

Asymptotic running-time with above m:

$$\left(\frac{n \log q}{\log^2 \alpha}\right)^{O\left(\frac{n \log q}{\log^2 \alpha}\right)}.$$

Note: binary error LWE does not change the asymptotics $2^{O(n)}$.

First strategy (primal): direct decoding attack.

Kannan's embedding: BDD → uSVP.

In LWE/BDD $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} + \mathbf{c}q$, thus $\mathbf{b}$ is close to the lattice point $\mathbf{A}\mathbf{s} + \mathbf{c}q$ in $\Lambda_q(\mathbf{A})$ where $\mathbf{e}$ is the small "shift". Let $\mathbf{L}$ be the basis of $\Lambda_q(\mathbf{A})$.

Construct

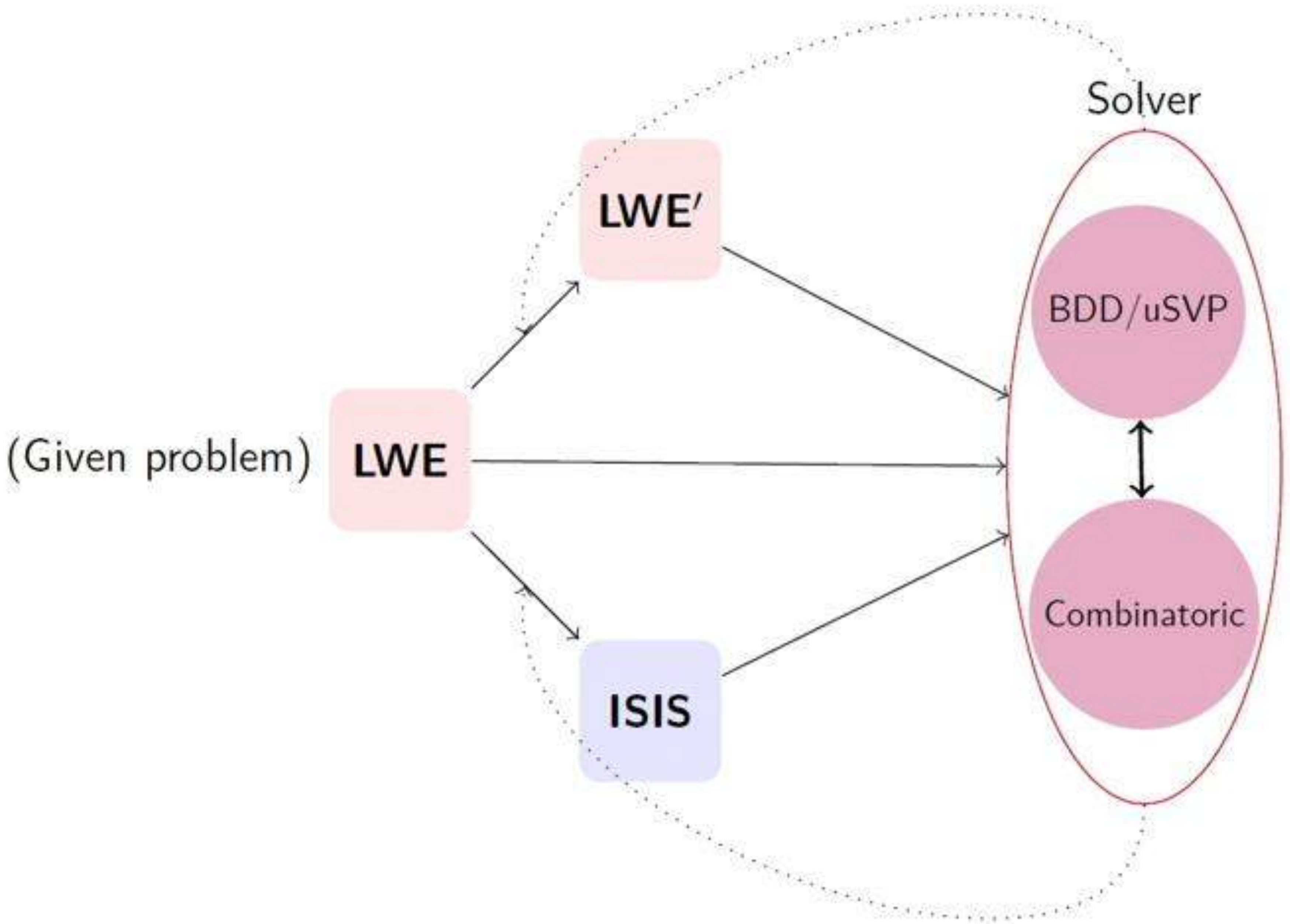
$$\mathbf{L}' = \begin{pmatrix} \mathbf{L} & \mathbf{b} \\ \mathbf{0} & 1 \end{pmatrix}.$$

We have

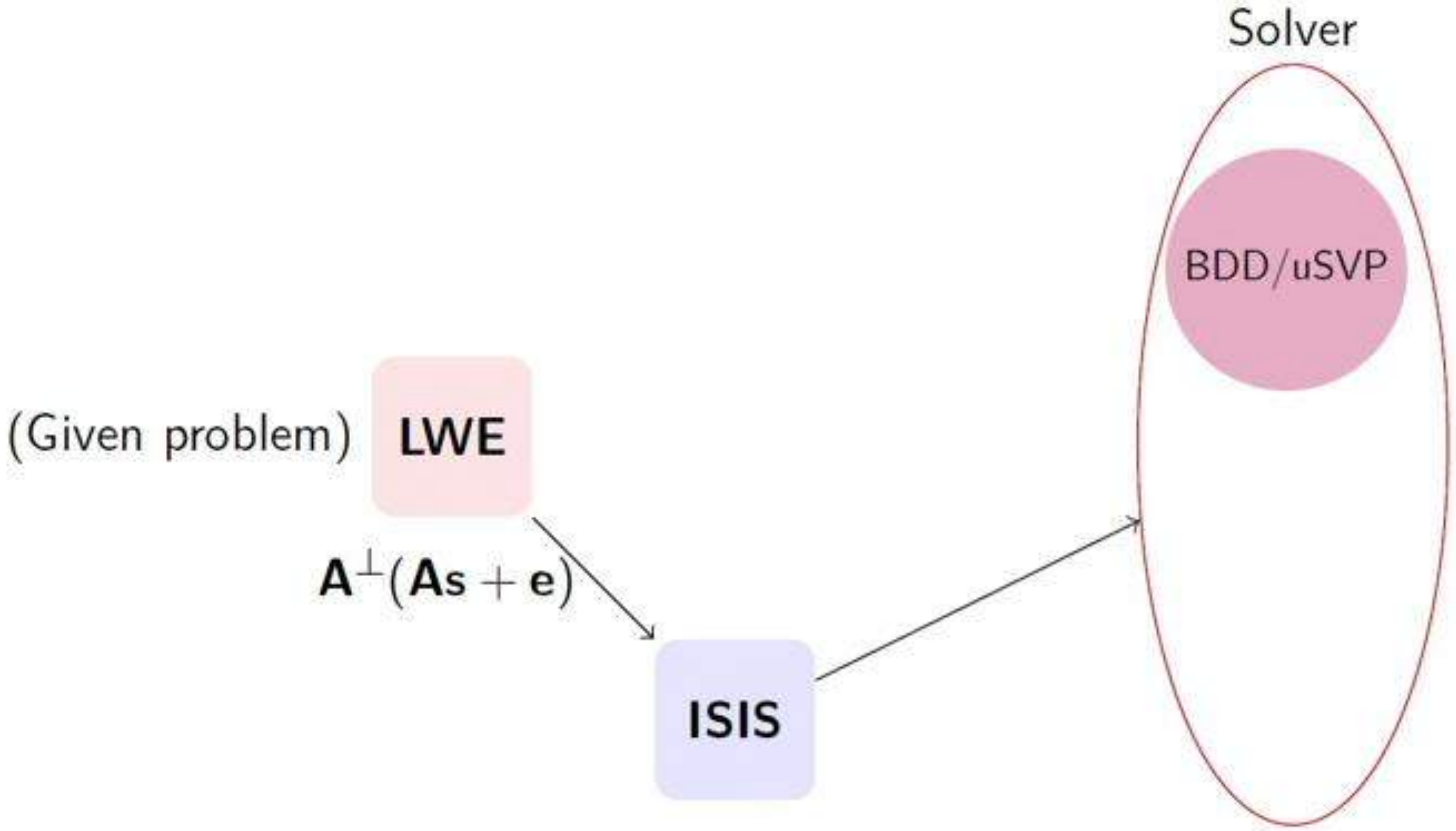
$$\mathbf{L}' \cdot \begin{pmatrix} * \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{e} \\ 1 \end{pmatrix}$$

where * is (negative) coefficients in generating the lattice point $\mathbf{A}\mathbf{s} + \mathbf{c}q$.

Lattice algorithms for LWE: summary of strategies.



Second strategy (dual): convert to ISIS-like using left-kernel.



Solver

BDD/uSVP

(Given problem) **LWE**

$\mathbf{A}^{\perp}(\mathbf{A}\mathbf{s} + \mathbf{e})$

ISIS

Second strategy (dual): convert to ISIS-like using left-kernel.

LWE → ISIS-like → BDD → uSVP, solve by BKZ.

$$\mathbf{A}^{\perp}\mathbf{b} = \mathbf{A}^{\perp}(\mathbf{A}\mathbf{s} + \mathbf{e}) = \mathbf{A}^{\perp} \cdot \mathbf{e} \pmod{q}.$$

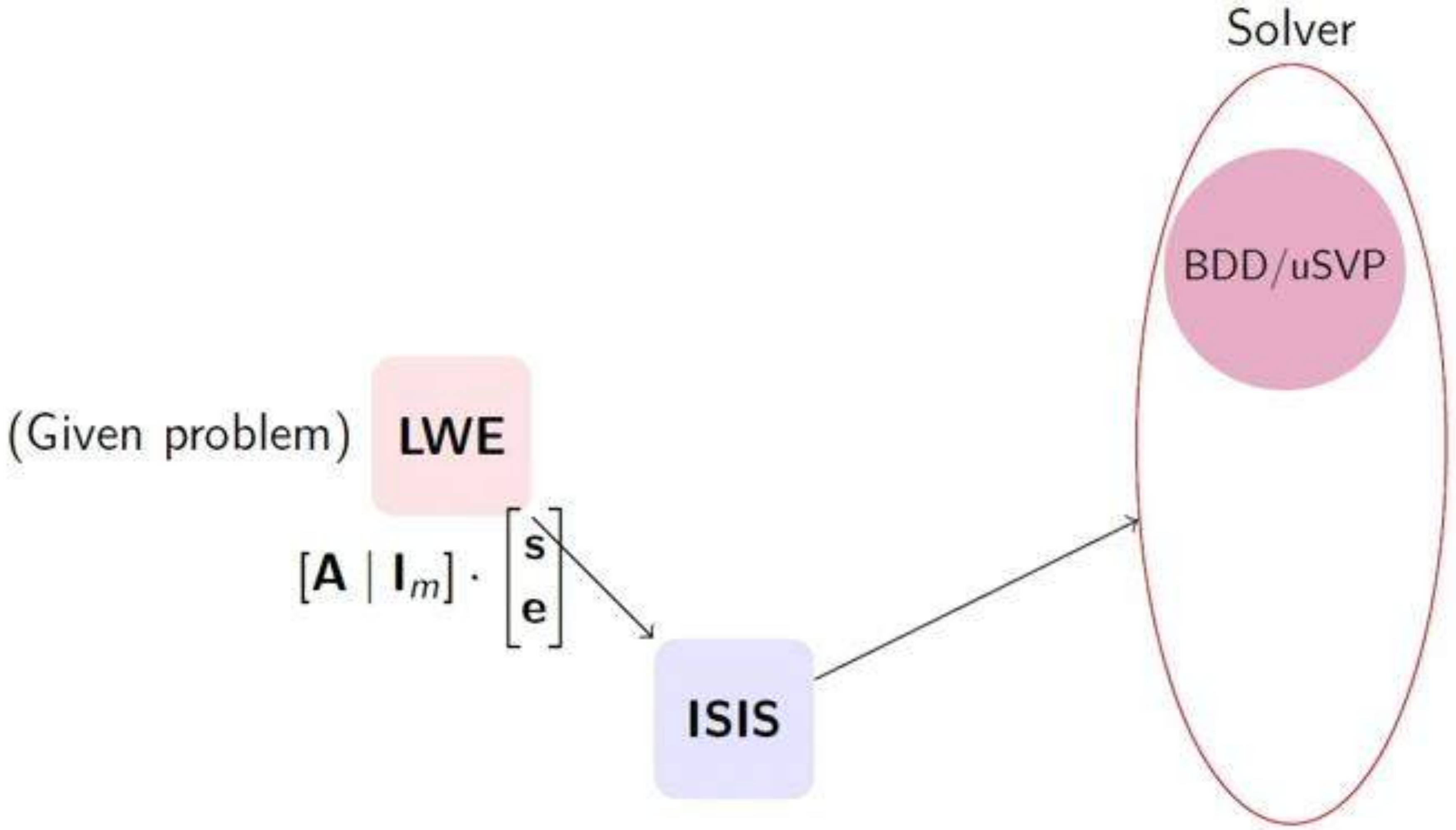
Find **e** by solving a ISIS-like problem.

General way to solve ISIS: $\mathbf{B} \cdot \mathbf{e} = \mathbf{t} \pmod{q}$.

- Find arbitrary (not necessarily short) **y** such that $\mathbf{B} \cdot \mathbf{y} = \mathbf{t} \pmod{q}$.
- Kernel lattice $\Lambda_q^{\perp}(\mathbf{B}) = \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{B}\mathbf{x} = \mathbf{0} \pmod{q}\}$.
- Call BDD/CVP with target point **y** in the kernel lattice. This gives **v** closest to **y**.
- $\mathbf{B}\mathbf{v} - \mathbf{B}\mathbf{y} = \mathbf{B}\mathbf{e} = \mathbf{t} \pmod{q}$.

The lattice has rank m and volume q^{m-n}. Only **e** is used.

Third strategy (primal): another way to convert to ISIS-like.



Solver

BDD/uSVP

(Given problem) **LWE**

$[\mathbf{A} \mid \mathbf{I}_m] \cdot \begin{bmatrix} \mathbf{s} \\ \mathbf{e} \end{bmatrix}$

ISIS

Third strategy (primal): another way to convert to ISIS-like.

Given $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \pmod{q}$, re-write as

$$\mathbf{b} = [\mathbf{A}|\mathbf{I}_m] \cdot \begin{bmatrix} \mathbf{s} \\ \mathbf{e} \end{bmatrix} = \mathbf{A}' \cdot \mathbf{s}' \pmod{q}.$$

- Find short $\begin{bmatrix} \mathbf{s} \\ \mathbf{e} \end{bmatrix}$ in the kernel lattice of $\mathbf{A}'$.
- The information of $\mathbf{s}$ is retained.
- The lattice has rank $m + n$ and volume q^m.
- If $\mathbf{s}$ and $\mathbf{e}$ are not balanced, re-balance the lattice (B.-Galbraith, '14).

These methods are sometimes equivalent, but not always, depending on the parameters given.

Fourth strategy: distinguishing attack using the dual.

Fourth strategy: distinguishing attack using the dual.

The previous dual/ISIS method:

$$\mathbf{A}^{\perp}\mathbf{b} = \mathbf{A}^{\perp}(\mathbf{A}\mathbf{s} + \mathbf{e}) = \mathbf{A}^{\perp} \cdot \mathbf{e} \pmod{q}.$$

Consider the kernel lattice of $\mathbf{A}^T$: $\{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}^T\mathbf{x} = \mathbf{0} \pmod{q}\}$.

- Find short vectors **w** in the kernel lattice.
- Then $\langle \mathbf{w}, \mathbf{b} \rangle = \langle \mathbf{w}, \mathbf{e} \rangle$ is much smaller than q.
- Repeat this for many **w** for higher confidence.

Note this removes the information on **s**.

Fourth strategy: distinguishing attack using the dual.

To keep $\mathbf{s}$, consider the solution $\mathbf{x}$ to $\{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}^T\mathbf{x} = \mathbf{y} \pmod{q}\}$ for any short $\mathbf{y}$. Equivalently this is,

$$[\mathbf{A}^T | \mathbf{I}_n] \cdot \begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix} = \mathbf{0} \pmod{q}.$$

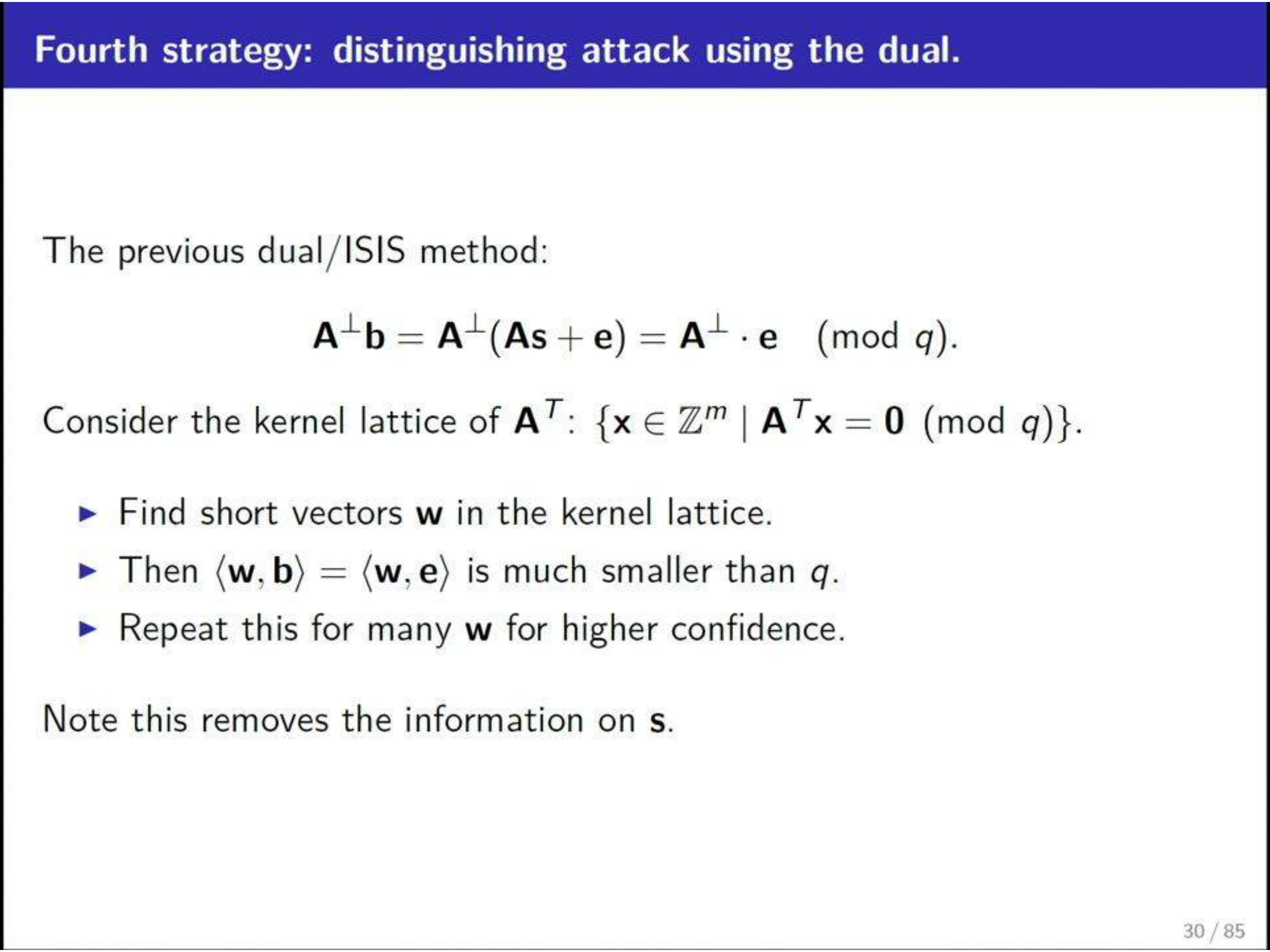
Let $(\mathbf{w}, \mathbf{v})$ be a short solution. Then

$$\langle \mathbf{w}, \mathbf{b} \rangle = \langle \mathbf{v}, \mathbf{s} \rangle + \langle \mathbf{w}, \mathbf{e} \rangle \ll q.$$

When $\mathbf{s}$ and $\mathbf{e}$ are not balanced, one can re-balance the lattice (Albrecht '17). This leads to

$$\langle c \cdot \mathbf{v}, \mathbf{s} \rangle + \langle \mathbf{w}, \mathbf{e} \rangle$$

where two parts contribute similar.



Fourth strategy: distinguishing attack using the dual.

The previous dual/ISIS method:

$$\mathbf{A}^{\perp}\mathbf{b} = \mathbf{A}^{\perp}(\mathbf{A}\mathbf{s} + \mathbf{e}) = \mathbf{A}^{\perp} \cdot \mathbf{e} \pmod{q}.$$

Consider the kernel lattice of $\mathbf{A}^T$: $\{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}^T\mathbf{x} = \mathbf{0} \pmod{q}\}$.

- Find short vectors $\mathbf{w}$ in the kernel lattice.
- Then $\langle \mathbf{w}, \mathbf{b} \rangle = \langle \mathbf{w}, \mathbf{e} \rangle$ is much smaller than q.
- Repeat this for many $\mathbf{w}$ for higher confidence.

Note this removes the information on $\mathbf{s}$.

Fourth strategy: distinguishing attack using the dual.

To keep $\mathbf{s}$, consider the solution $\mathbf{x}$ to $\{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}^T\mathbf{x} = \mathbf{y} \pmod{q}\}$ for any short $\mathbf{y}$. Equivalently this is,

$$[\mathbf{A}^T | \mathbf{I}_n] \cdot \begin{bmatrix} \mathbf{x} \\ -\mathbf{y} \end{bmatrix} = \mathbf{0} \pmod{q}.$$

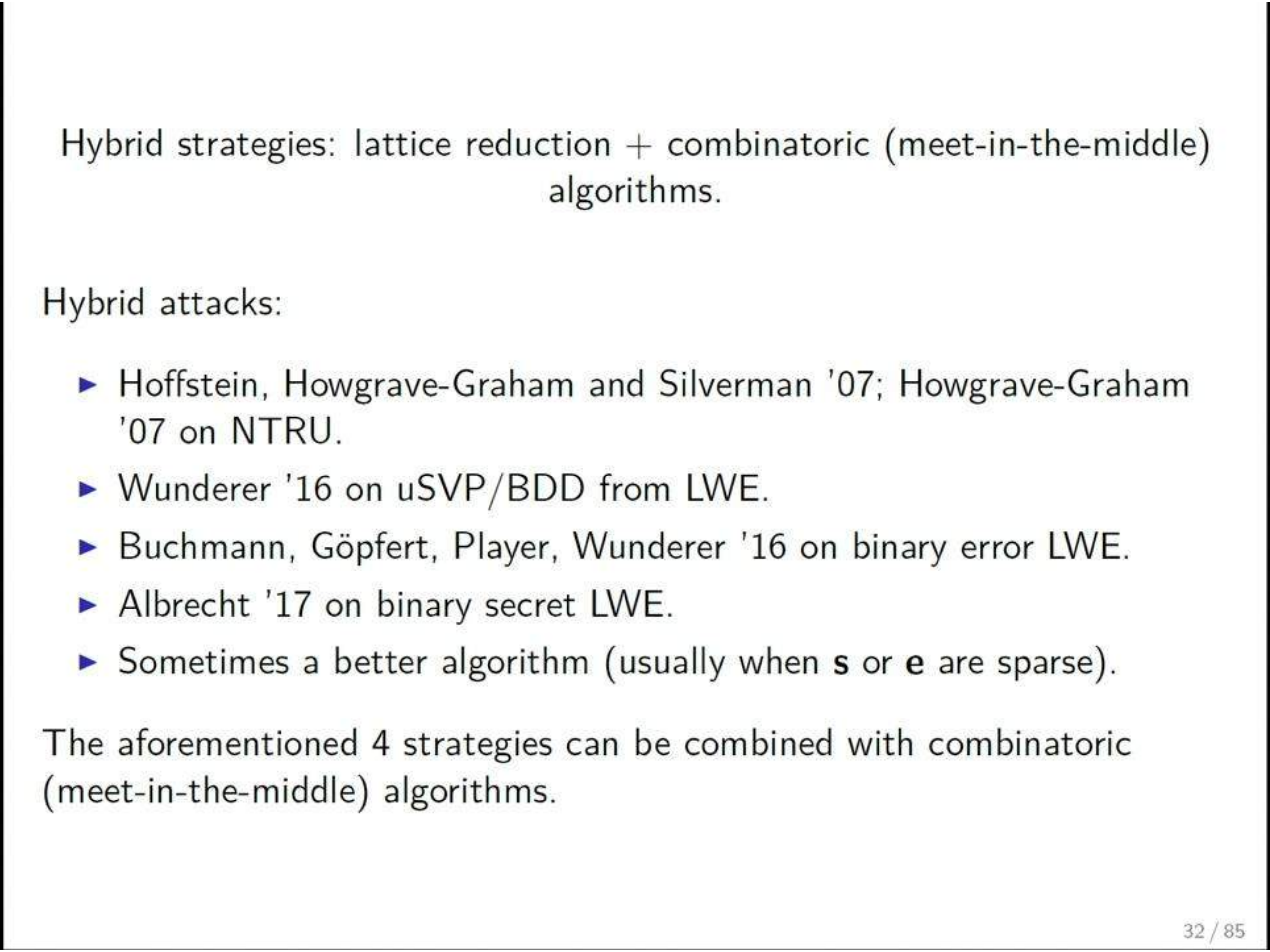
Let $(\mathbf{w}, \mathbf{v})$ be a short solution. Then

$$\langle \mathbf{w}, \mathbf{b} \rangle = \langle \mathbf{v}, \mathbf{s} \rangle + \langle \mathbf{w}, \mathbf{e} \rangle \ll q.$$

When $\mathbf{s}$ and $\mathbf{e}$ are not balanced, one can re-balance the lattice (Albrecht '17). This leads to

$$\langle c \cdot \mathbf{v}, \mathbf{s} \rangle + \langle \mathbf{w}, \mathbf{e} \rangle$$

where two parts contribute similar.

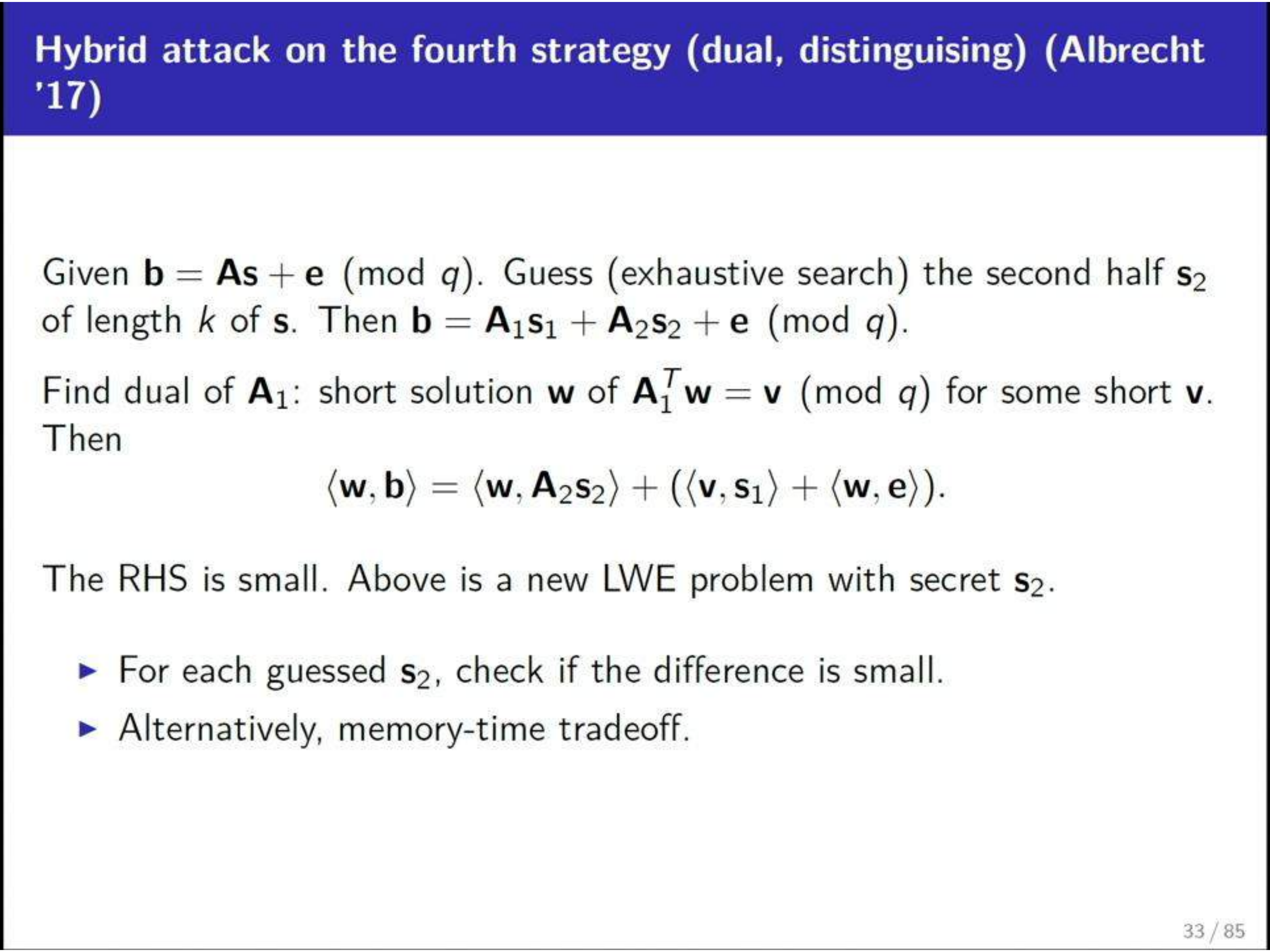


Hybrid strategies: lattice reduction + combinatoric (meet-in-the-middle) algorithms.

Hybrid attacks:

- Hoffstein, Howgrave-Graham and Silverman '07; Howgrave-Graham '07 on NTRU.
- Wunderer '16 on uSVP/BDD from LWE.
- Buchmann, Göpfert, Player, Wunderer '16 on binary error LWE.
- Albrecht '17 on binary secret LWE.
- Sometimes a better algorithm (usually when **s** or **e** are sparse).

The aforementioned 4 strategies can be combined with combinatoric (meet-in-the-middle) algorithms.



Hybrid attack on the fourth strategy (dual, distinguising) (Albrecht '17)

Given $\mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \pmod q$. Guess (exhaustive search) the second half $\mathbf{s}_2$ of length k of $\mathbf{s}$. Then $\mathbf{b} = \mathbf{A}_1\mathbf{s}_1 + \mathbf{A}_2\mathbf{s}_2 + \mathbf{e} \pmod q$.

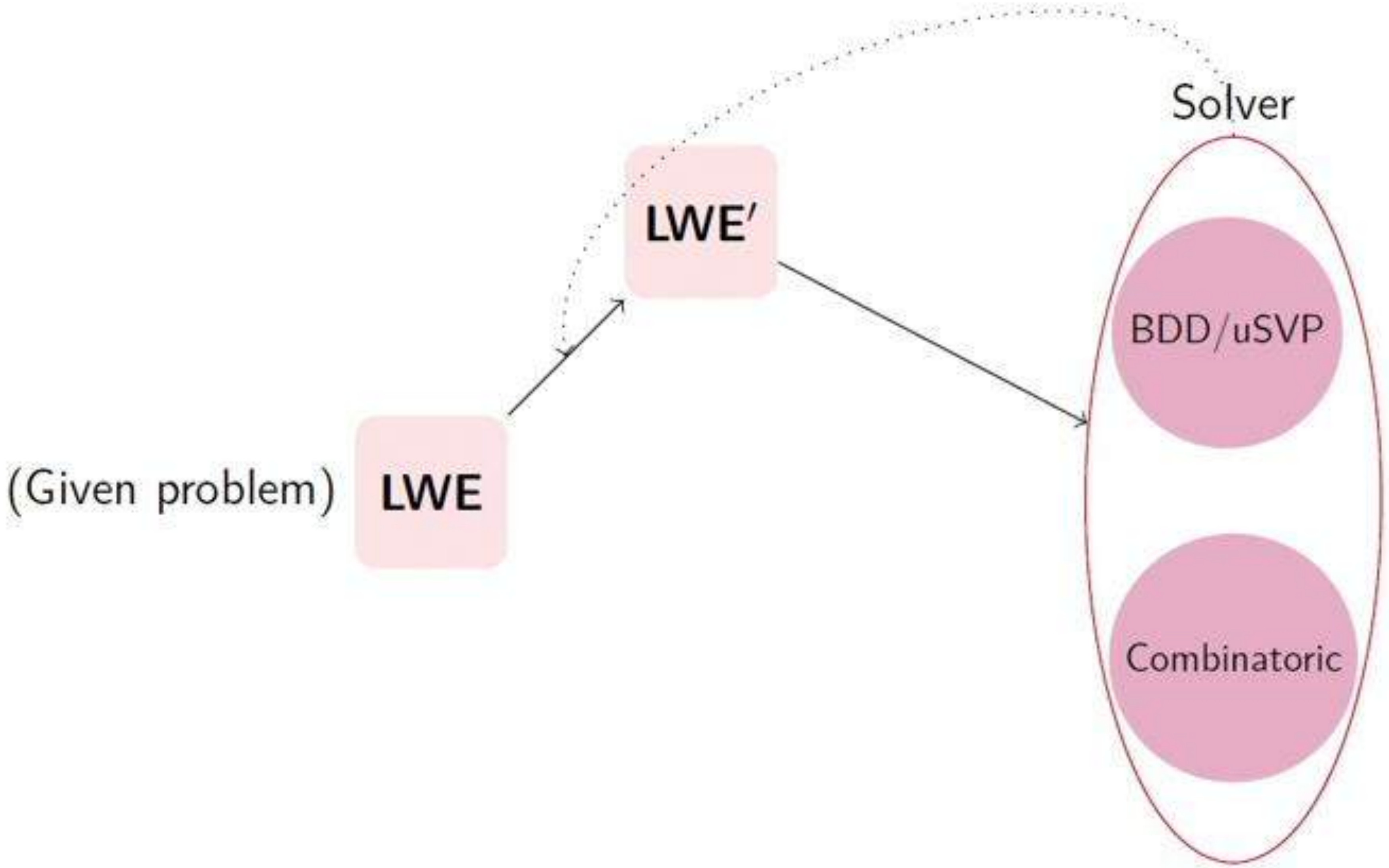
Find dual of $\mathbf{A}_1$: short solution $\mathbf{w}$ of $\mathbf{A}_1^T\mathbf{w} = \mathbf{v} \pmod q$ for some short $\mathbf{v}$. Then

$$\langle \mathbf{w}, \mathbf{b} \rangle = \langle \mathbf{w}, \mathbf{A}_2\mathbf{s}_2 \rangle + (\langle \mathbf{v}, \mathbf{s}_1 \rangle + \langle \mathbf{w}, \mathbf{e} \rangle).$$

The RHS is small. Above is a new LWE problem with secret $\mathbf{s}_2$.

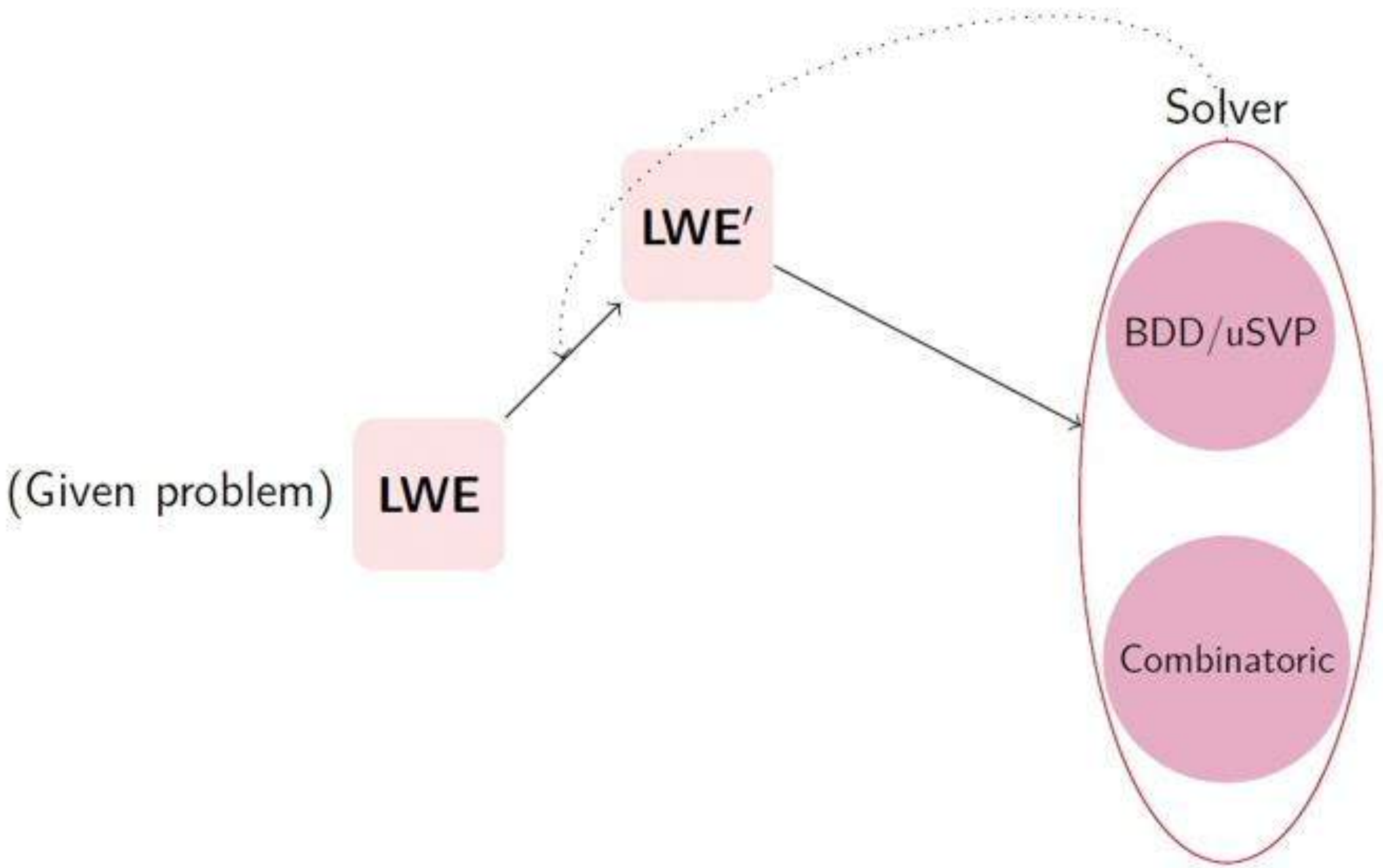
- For each guessed $\mathbf{s}_2$, check if the difference is small.
- Alternatively, memory-time tradeoff.

Strategy of dual, distinguishing, combinatoric.



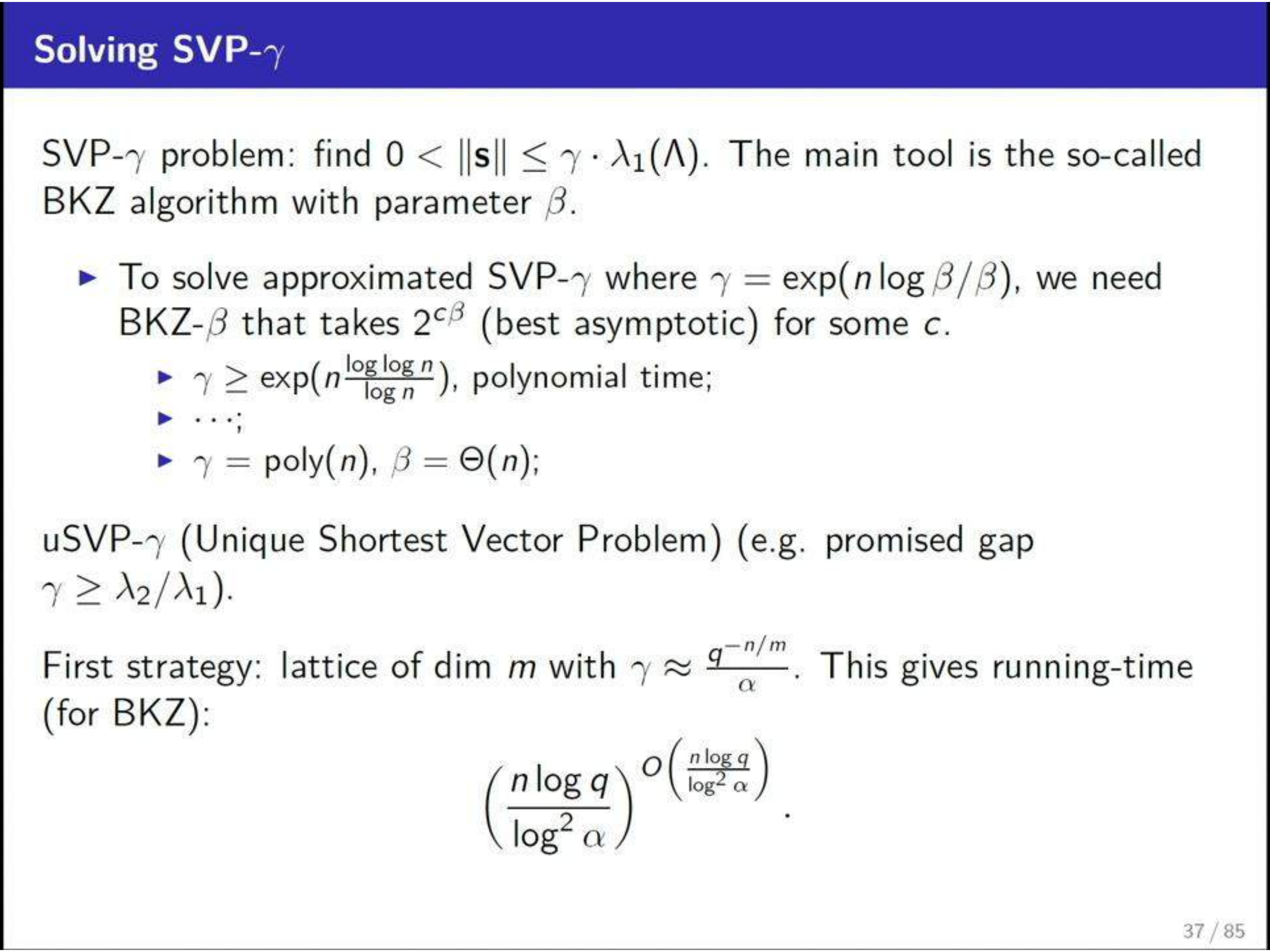
So far, we used the SVP-solvers as oracles.

Strategy of dual, distinguishing, combinatoric.



So far, we used the SVP-solvers as oracles.

4. uSVP/BDD solver.



Solving SVP-γ

SVP-γ problem: find $0 < \|\mathbf{s}\| \leq \gamma \cdot \lambda_1(\Lambda)$. The main tool is the so-called BKZ algorithm with parameter β.

- To solve approximated SVP-γ where $\gamma = \exp(n \log \beta / \beta)$, we need BKZ-$\beta$ that takes $2^{c\beta}$ (best asymptotic) for some c.
  - $\gamma \geq \exp(n \frac{\log \log n}{\log n})$, polynomial time;
  - $\cdots$;
  - $\gamma = \text{poly}(n)$, $\beta = \Theta(n)$;

uSVP-γ (Unique Shortest Vector Problem) (e.g. promised gap $\gamma \geq \lambda_2/\lambda_1$).

First strategy: lattice of dim m with $\gamma \approx \frac{q^{-n/m}}{\alpha}$. This gives running-time (for BKZ):

$$\left(\frac{n \log q}{\log^2 \alpha}\right)^{O\left(\frac{n \log q}{\log^2 \alpha}\right)}.$$

SVP-β tours in BKZ-β

BKZ-β "smooth" over the basis vectors with many invokes of SVP-β.

The building elements for BKZ-like reductions are block-wise-β SVP reductions on projected sublattices,

$$L_i = [\pi_i(\mathbf{b}_i), \cdots, \pi_i(\mathbf{b}_{i+\beta-1})].$$

If the block is SVP-reduced, then

$$\|\mathbf{b}_i^*\| \leq \sqrt{\gamma_\beta} \cdot \det(L_i)^{1/\beta}.$$

If the block is dual-SVP-reduced, then

$$\det(L_i)^{1/\beta} \leq \sqrt{\gamma_\beta} \cdot \left\|\mathbf{b}_{i+\beta-1}^*\right\|.$$

The BKZ, Slide reduction (Gama-Nguyen '08), self-dual BKZ reduction (Micciancio-Walter '15) use a combination of above strategies locally; and update these changes globally by "tours" over the whole basis.

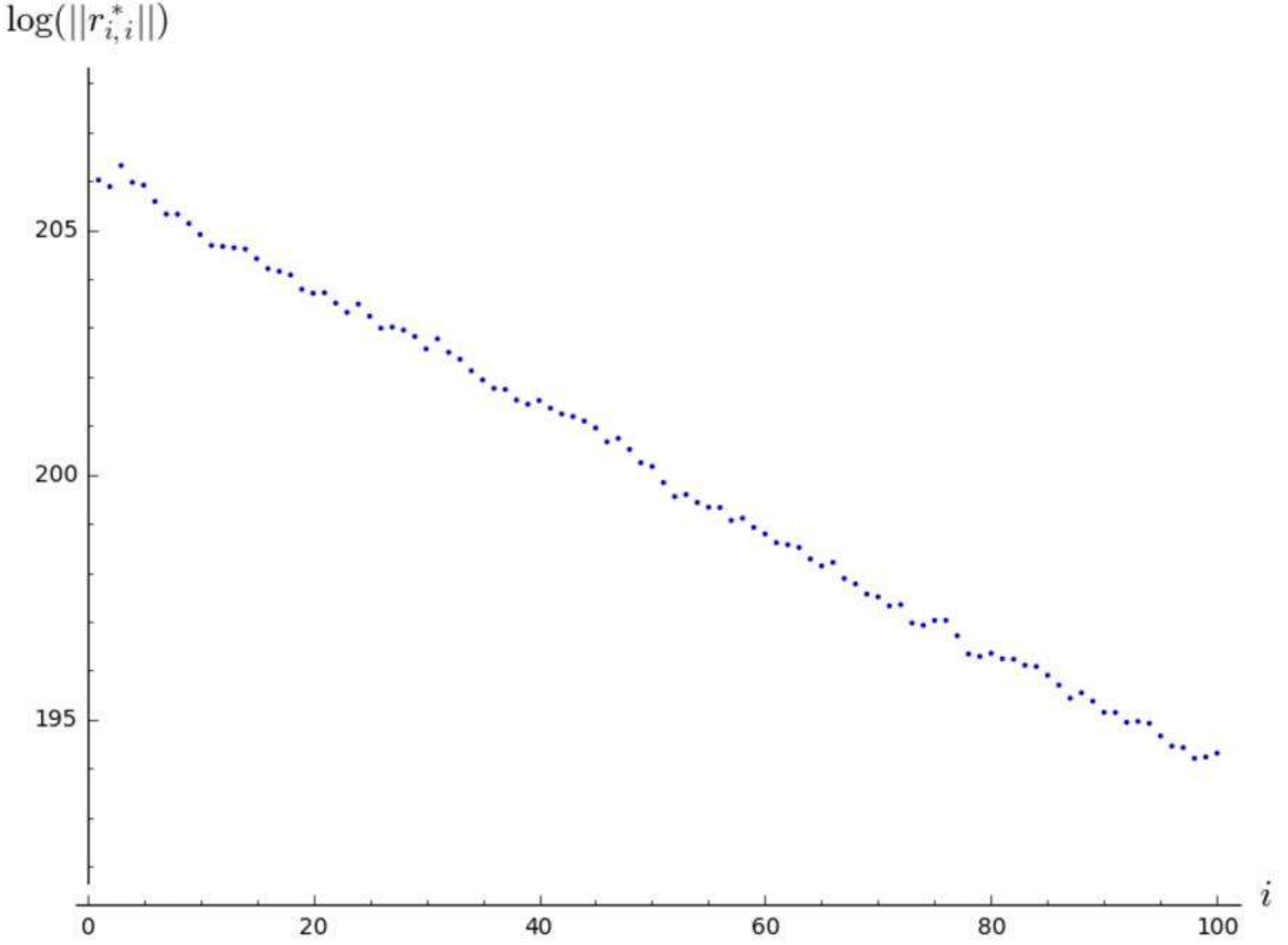
Simple BKZ

Algorithm 1: BKZ-β

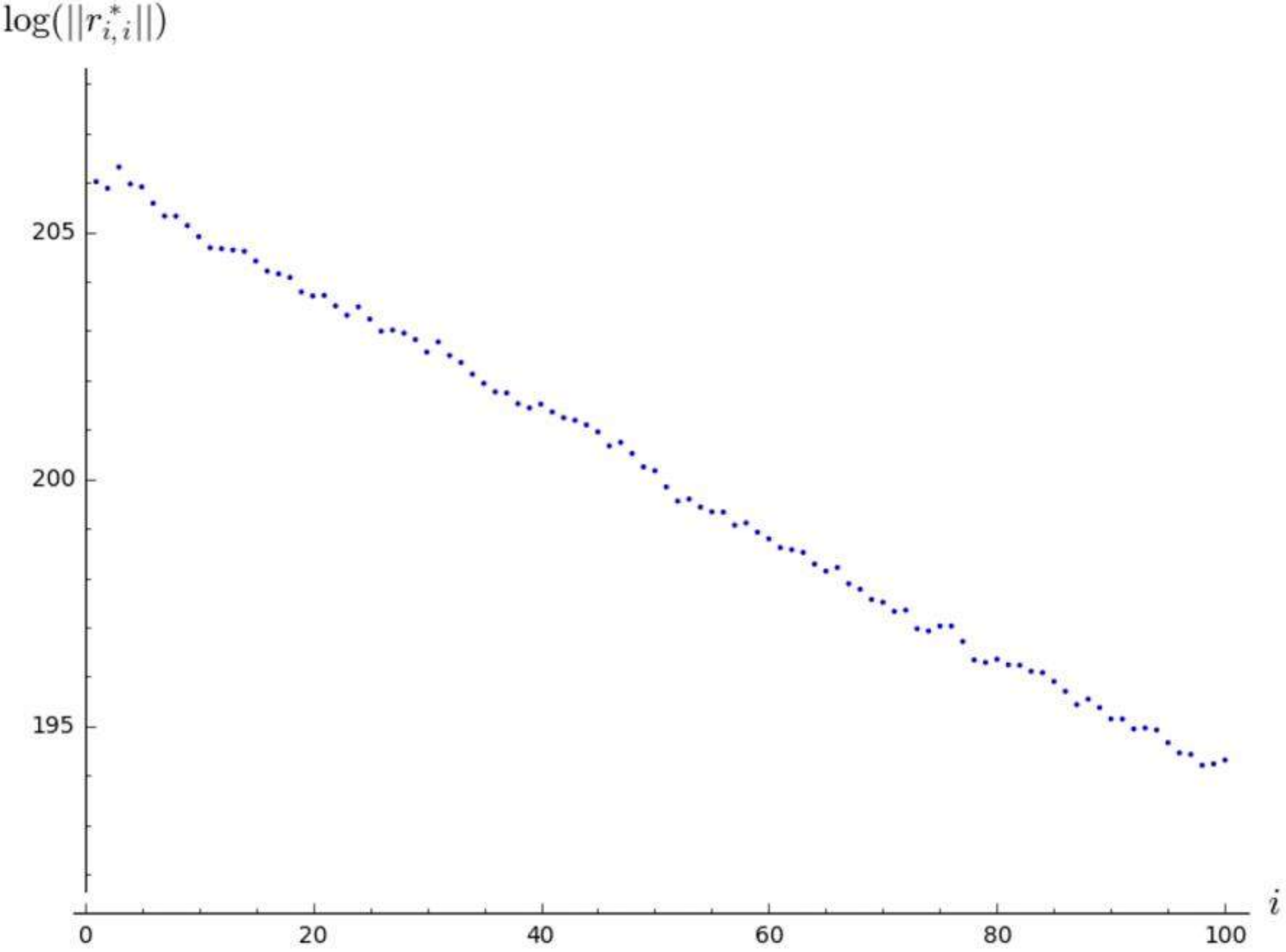
```
while Changes during SVP process or reached a threshold do
    for $i = 1$ to $n - \beta + 1$ do
        Solve SVP on block $L_i$;
    for $i = n - \beta + 2$ to $n - 1$ do
        Solve SVP on tail blocks;
```

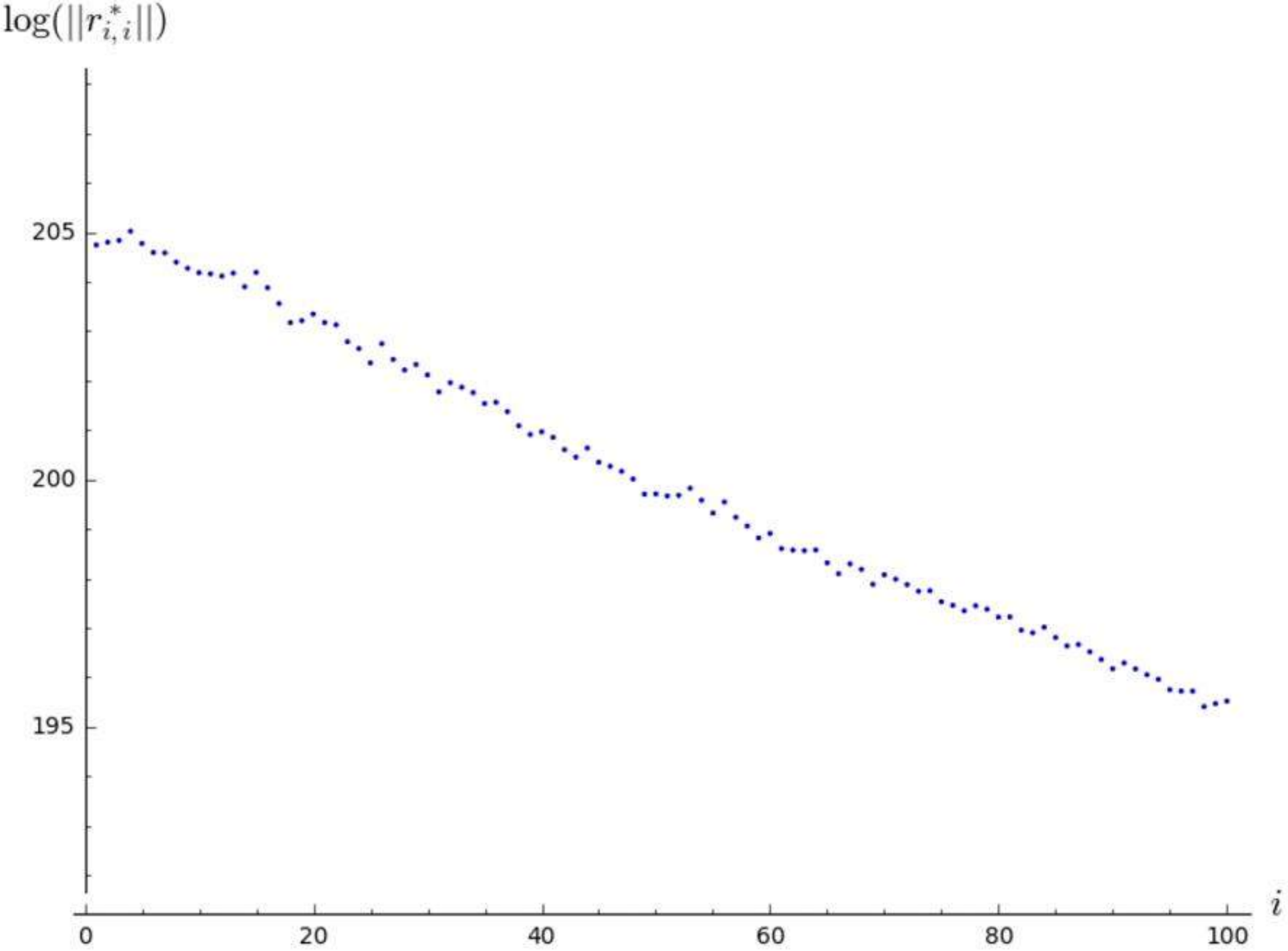
After each tour, the basis is “smoothed” by bringing the “shorter” vectors to the front.

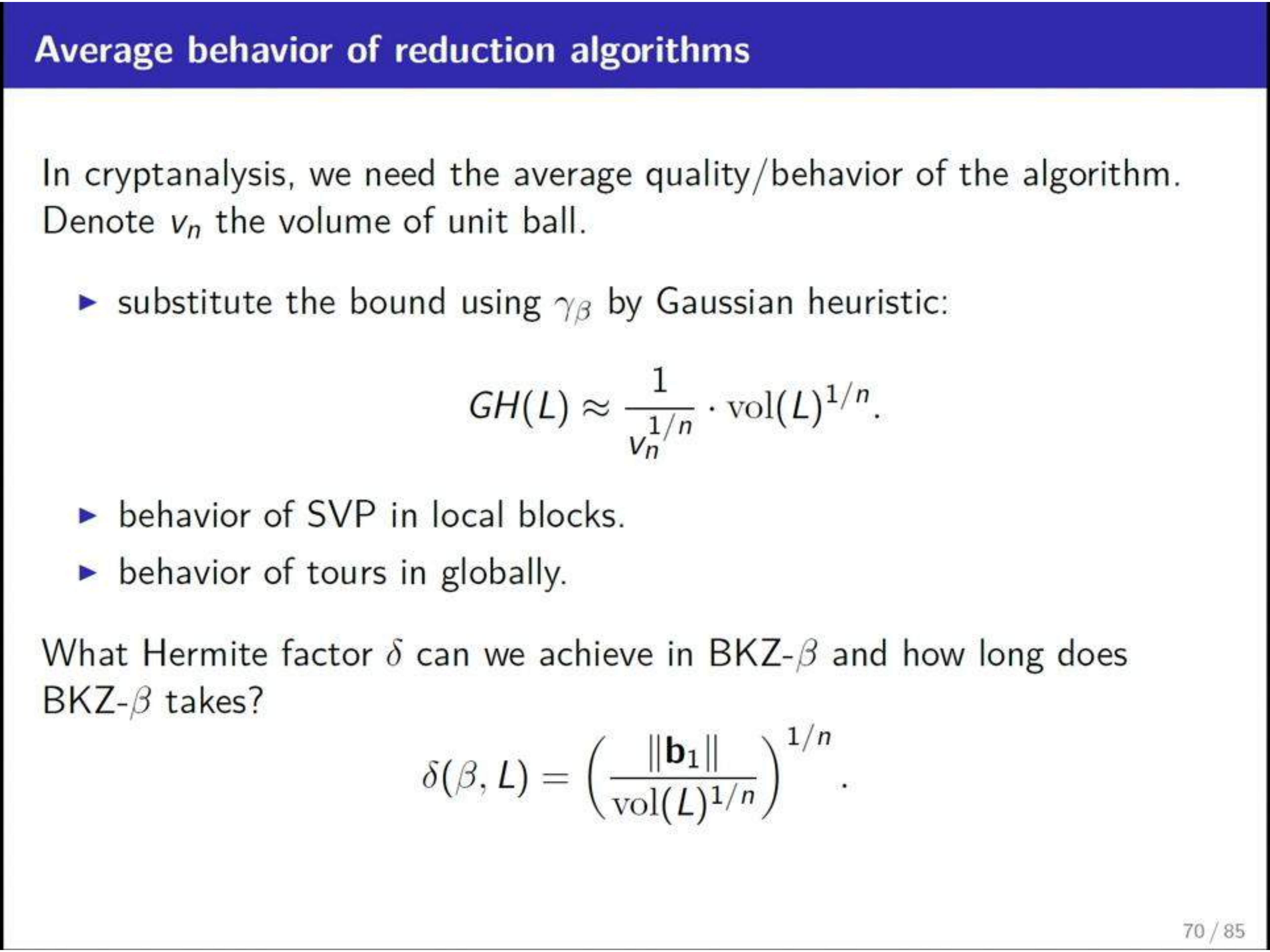
Example: BKZ-40


$\log(||r^*_{i,i}||)$
205
200
195
0
20
40
60
80
100
i

Example: BKZ-40


$\log(||r^*_{i,i}||)$
205
200
195
i
0
20
40
60
80
100


$\log(||r^*_{i,i}||)$
205
200
195
0
20
40
60
80
100
i



Average behavior of reduction algorithms

In cryptanalysis, we need the average quality/behavior of the algorithm. Denote v_n the volume of unit ball.

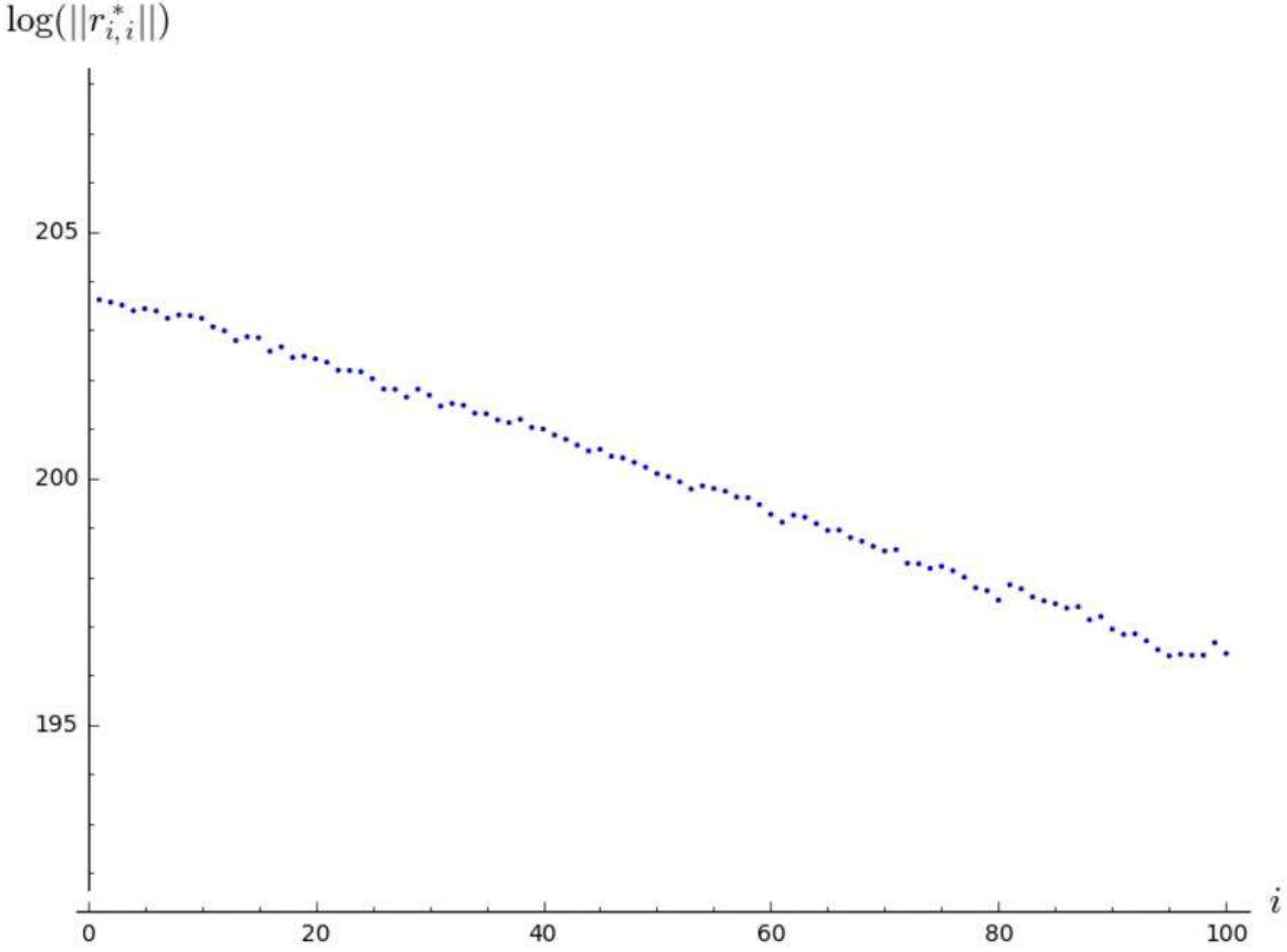
- substitute the bound using γ_β by Gaussian heuristic:

$$GH(L) \approx \frac{1}{v_n^{1/n}} \cdot \mathrm{vol}(L)^{1/n}.$$

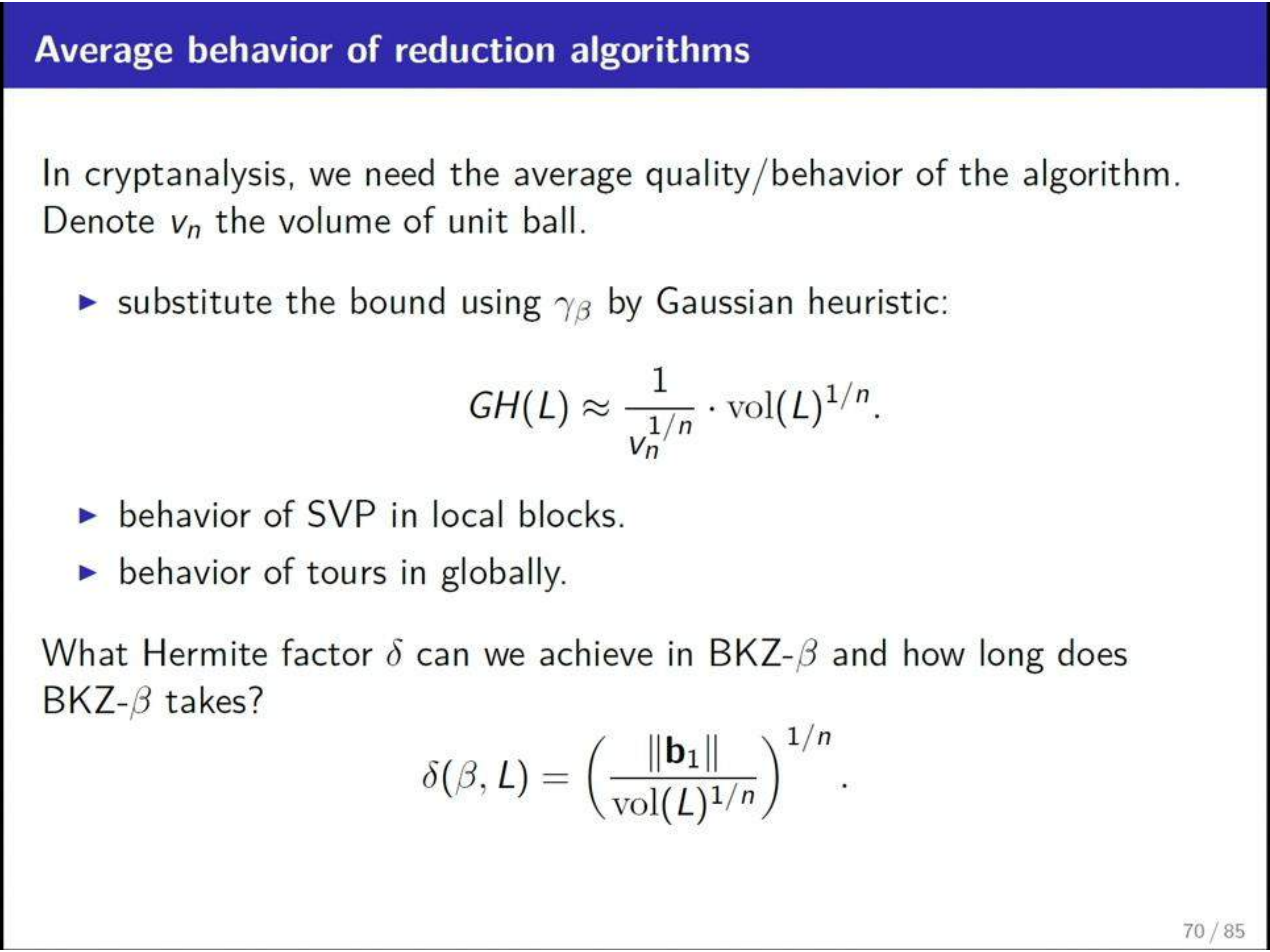
- behavior of SVP in local blocks.
- behavior of tours in globally.

What Hermite factor δ can we achieve in BKZ-β and how long does BKZ-β takes?

$$\delta(\beta, L) = \left(\frac{\|\mathbf{b}_1\|}{\mathrm{vol}(L)^{1/n}}\right)^{1/n}.$$


$\log(||r^*_{i,i}||)$
205
200
195
0
20
40
60
80
100
i

Quantifying the quality after BKZ.



Average behavior of reduction algorithms

In cryptanalysis, we need the average quality/behavior of the algorithm. Denote v_n the volume of unit ball.

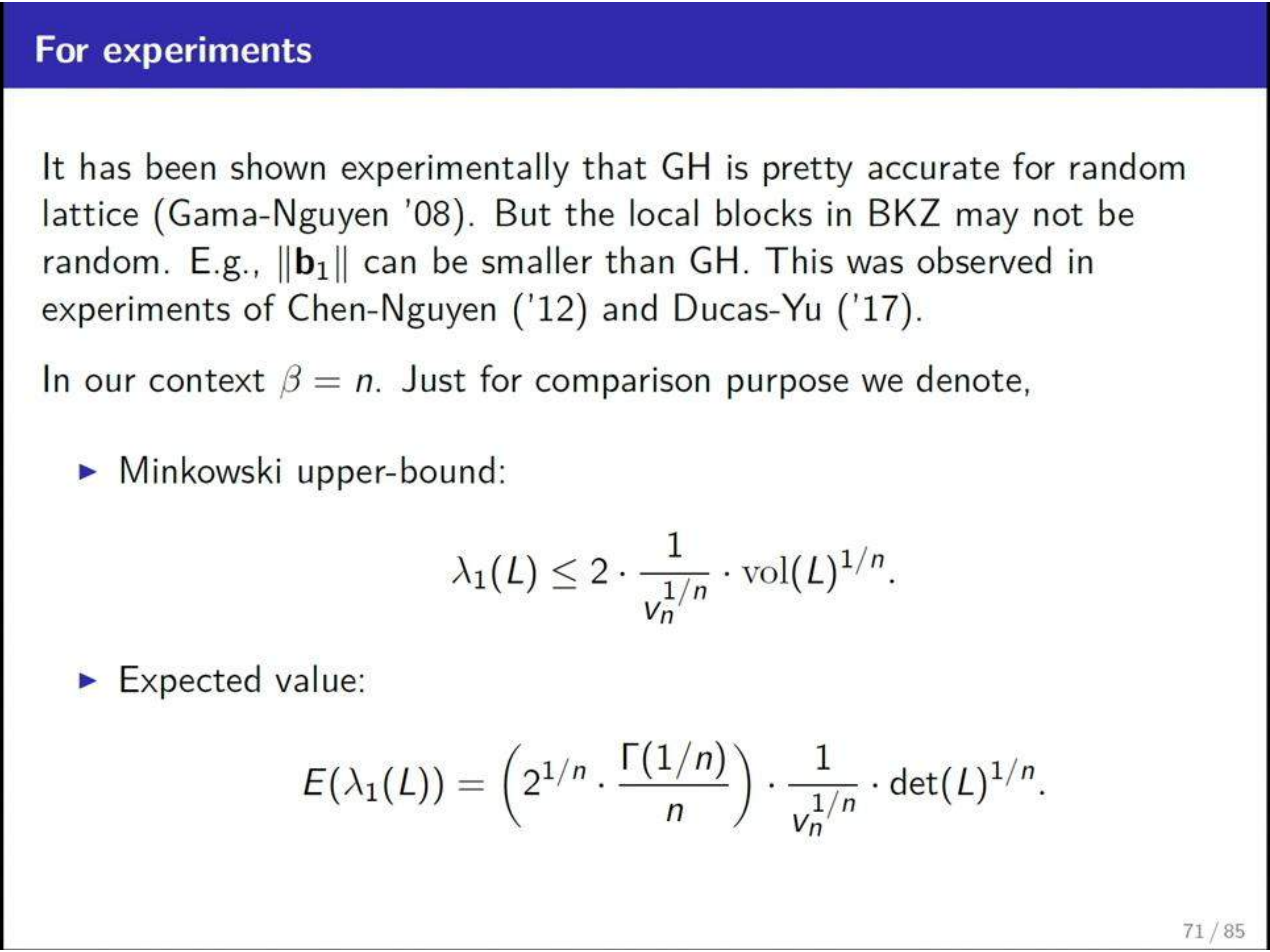
- substitute the bound using γ_β by Gaussian heuristic:

$$GH(L) \approx \frac{1}{v_n^{1/n}} \cdot \mathrm{vol}(L)^{1/n}.$$

- behavior of SVP in local blocks.
- behavior of tours in globally.

What Hermite factor δ can we achieve in BKZ-β and how long does BKZ-β takes?

$$\delta(\beta, L) = \left(\frac{\|\mathbf{b}_1\|}{\mathrm{vol}(L)^{1/n}}\right)^{1/n}.$$



For experiments

It has been shown experimentally that GH is pretty accurate for random lattice (Gama-Nguyen '08). But the local blocks in BKZ may not be random. E.g., $\|\mathbf{b}_1\|$ can be smaller than GH. This was observed in experiments of Chen-Nguyen ('12) and Ducas-Yu ('17).

In our context $\beta = n$. Just for comparison purpose we denote,

- Minkowski upper-bound:

$$\lambda_1(L) \leq 2 \cdot \frac{1}{v_n^{1/n}} \cdot \mathrm{vol}(L)^{1/n}.$$

- Expected value:

$$E(\lambda_1(L)) = \left(2^{1/n} \cdot \frac{\Gamma(1/n)}{n}\right) \cdot \frac{1}{v_n^{1/n}} \cdot \det(L)^{1/n}.$$

Average behavior of reduction algorithms

In cryptanalysis, we need the average quality/behavior of the algorithm. Denote v_n the volume of unit ball.

- substitute the bound using γ_β by Gaussian heuristic:

$$GH(L) \approx \frac{1}{v_n^{1/n}} \cdot \mathrm{vol}(L)^{1/n}.$$

- behavior of SVP in local blocks.
- behavior of tours in globally.

What Hermite factor δ can we achieve in BKZ-β and how long does BKZ-β takes?

$$\delta(\beta, L) = \left(\frac{\|\mathbf{b}_1\|}{\mathrm{vol}(L)^{1/n}}\right)^{1/n}.$$

For experiments

It has been shown experimentally that GH is pretty accurate for random lattice (Gama-Nguyen '08). But the local blocks in BKZ may not be random. E.g., $\|\mathbf{b}_1\|$ can be smaller than GH. This was observed in experiments of Chen-Nguyen ('12) and Ducas-Yu ('17).

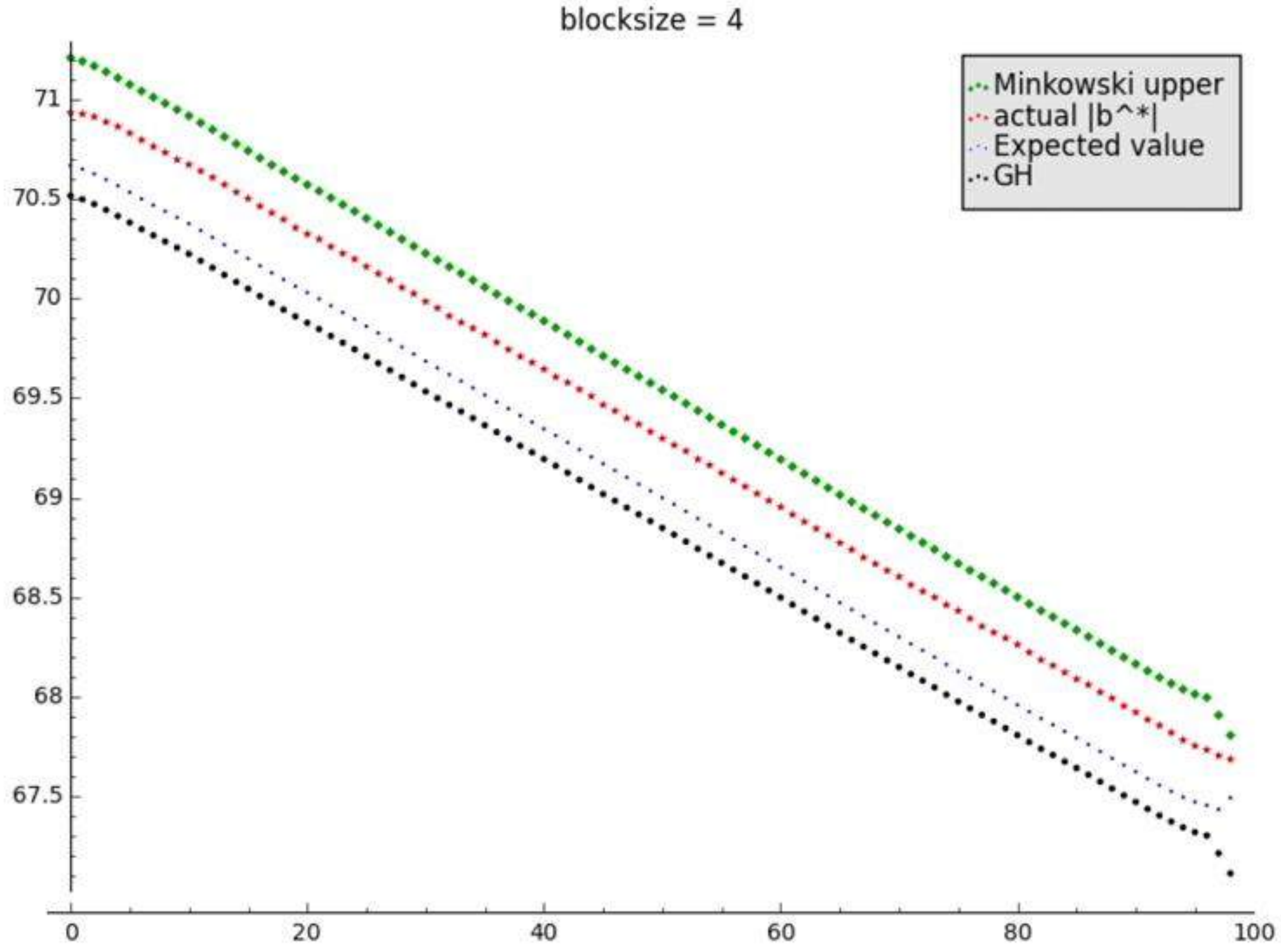
In our context $\beta = n$. Just for comparison purpose we denote,

- Minkowski upper-bound:

$$\lambda_1(L) \leq 2 \cdot \frac{1}{v_n^{1/n}} \cdot \mathrm{vol}(L)^{1/n}.$$

- Expected value:

$$E(\lambda_1(L)) = \left(2^{1/n} \cdot \frac{\Gamma(1/n)}{n}\right) \cdot \frac{1}{v_n^{1/n}} \cdot \det(L)^{1/n}.$$


blocksize = 4
Minkowski upper
actual |b^*|
Expected value
GH
71
70.5
70
69.5
69
68.5
68
67.5
0
20
40
60
80
100

It seems that the average behavior BKZ algorithm is even better than the Gaussian heuristic estimate: the $\|\mathbf{b}_i\|$'s are smaller than GH predicates for first several indices.

Better simulating the quality of BKZ

We model this using the distribution of lengths of lattice vectors in a random lattice (B.-Stehlé-Wen, '18). For a random n-dimensional unit-volume lattice $\mathcal{L}$, the $\lambda_1(\mathcal{L})$ (Södergren, '11) follows

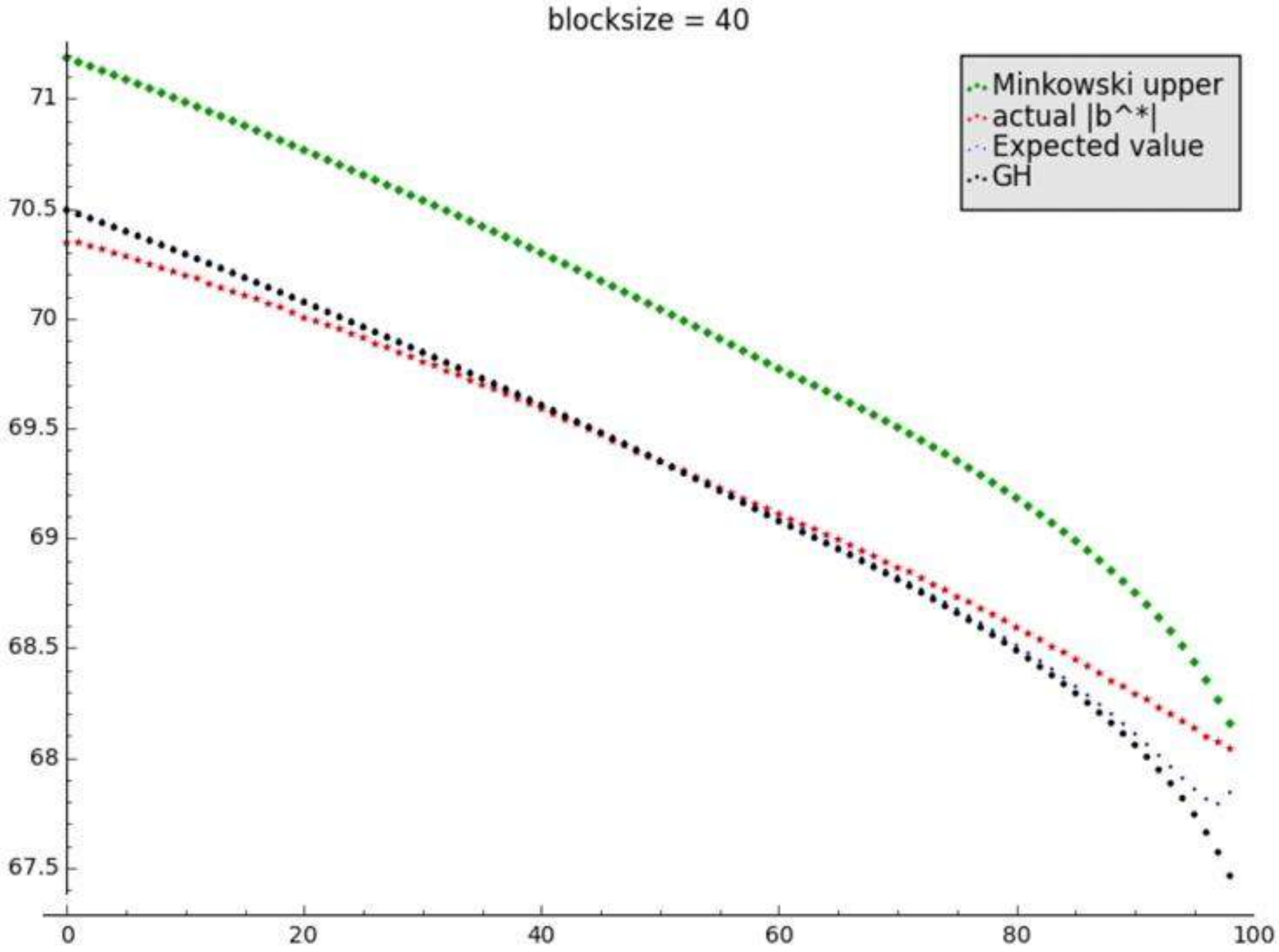
$$Y = X^{1/n} \cdot \mathrm{GH}(\mathcal{L})$$

where X is a random variable distributed as the exponential distribution with parameter $1/2$. We used this idea to improve the simulator of Chen-Nguyen '12.

- Sample X according to $\mathrm{Expo}[1/2]$.
- If $\|\mathbf{b}_i^*\| > X^{1/\beta} \cdot \mathrm{GH}(\mathcal{L}(\pi_i(\mathbf{b}_i), \cdots, \pi_i(\mathbf{b}_{\min(n,i+\beta-1)})))$; update this $\|\mathbf{b}_i^*\|$.

Heuristic analysis using order statistics (after K SVPs):

$$\mathbb{E}(Y_{K,min}) = (2/K)^{1/\beta} \cdot \Gamma(1 + 1/\beta) = \mathbb{E}(\lambda_1(\mathcal{L}))/K^{1/\beta}.$$


blocksize = 40
Minkowski upper
actual |b^*|
Expected value
GH
71
70.5
70
69.5
69
68.5
68
67.5
0
20
40
60
80
100

Better simulating the quality of BKZ

We model this using the distribution of lengths of lattice vectors in a random lattice (B.-Stehlé-Wen, '18). For a random *n*-dimensional unit-volume lattice $\mathcal{L}$, the $\lambda_1(\mathcal{L})$ (Södergren, '11) follows

$$Y = X^{1/n} \cdot \mathrm{GH}(\mathcal{L})$$

where X is a random variable distributed as the exponential distribution with parameter $1/2$. We used this idea to improve the simulator of Chen-Nguyen '12.

- Sample X according to $\mathrm{Expo}[1/2]$.
- If $\|\mathbf{b}_i^*\| > X^{1/\beta} \cdot \mathrm{GH}(\mathcal{L}(\pi_i(\mathbf{b}_i), \cdots, \pi_i(\mathbf{b}_{\min(n, i+\beta-1)})))$; update this $\|\mathbf{b}_i^*\|$.

Heuristic analysis using order statistics (after K SVPs):

$$\mathbb{E}(Y_{K,min}) = (2/K)^{1/\beta} \cdot \Gamma(1 + 1/\beta) = \mathbb{E}(\lambda_1(\mathcal{L}))/K^{1/\beta}.$$

Quality (seems fine) and time (tricky)

The aforementioned simulator seems to provide a good quality estimate to the BKZ algorithm.

However, simulating the concrete cost of BKZ seems tricky. There are various models:

- Local SVP-β cost;
  - By sieving or,
  - By pruned enumeration;
- Number of tours (thus the number of local SVP-β).
- Alternative reduction strategies may change the number of tours.

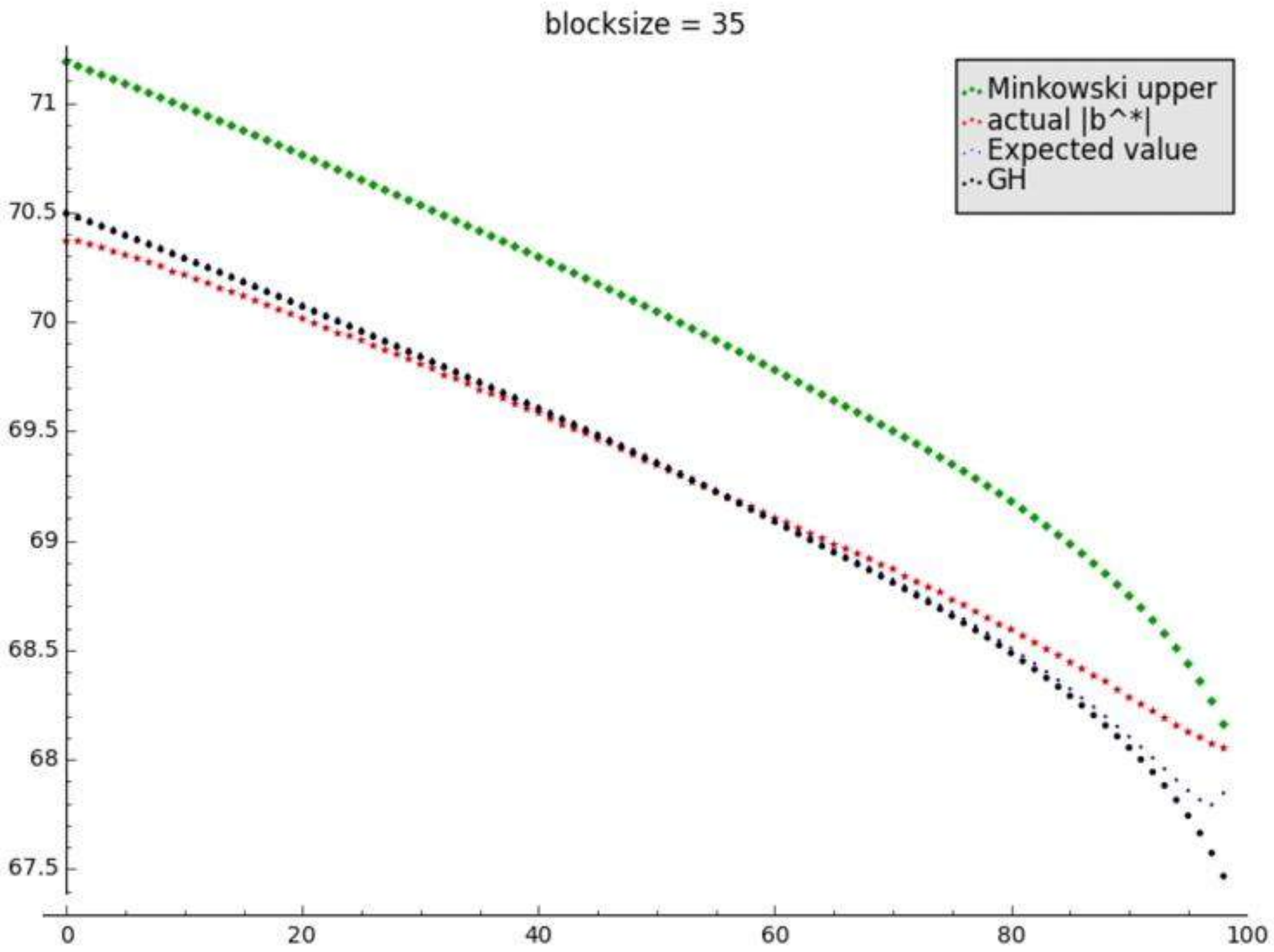
Some questions

- A more extensive study of hybrid attacks to LWE using the aforementioned strategies.
- Removing gaps between reduction/best-attacks in variants of LWE.
- Estimating the running-time of BKZ-β more precisely.
- Sieving v.s. enumeration. Sieving outperforms enum. in practice recently (but memory is the bottleneck in real world).
- Better algorithm (strategies) for BKZ-like reduction?
- How to better use the algebraic structures in lattice reduction ?

THANK
YOU

Some questions

- A more extensive study of hybrid attacks to LWE using the aforementioned strategies.
- Removing gaps between reduction/best-attacks in variants of LWE.
- Estimating the running-time of BKZ-β more precisely.
- Sieving v.s. enumeration. Sieving outperforms enum. in practice recently (but memory is the bottleneck in real world).
- Better algorithm (strategies) for BKZ-like reduction?
- How to better use the algebraic structures in lattice reduction ?



blocksize = 35

Better simulating the quality of BKZ

We model this using the distribution of lengths of lattice vectors in a random lattice (B.-Stehlé-Wen, '18). For a random *n*-dimensional unit-volume lattice $\mathcal{L}$, the $\lambda_1(\mathcal{L})$ (Södergren, '11) follows

$$Y = X^{1/n} \cdot \mathrm{GH}(\mathcal{L})$$

where X is a random variable distributed as the exponential distribution with parameter $1/2$. We used this idea to improve the simulator of Chen-Nguyen '12.

- Sample X according to $\mathrm{Expo}[1/2]$.
- If $\|\mathbf{b}_i^*\| > X^{1/\beta} \cdot \mathrm{GH}(\mathcal{L}(\pi_i(\mathbf{b}_i), \cdots, \pi_i(\mathbf{b}_{\min(n,i+\beta-1)})))$; update this $\|\mathbf{b}_i^*\|$.

Heuristic analysis using order statistics (after K SVPs):

$$\mathbb{E}(Y_{K,min}) = (2/K)^{1/\beta} \cdot \Gamma(1 + 1/\beta) = \mathbb{E}(\lambda_1(\mathcal{L}))/K^{1/\beta}.$$

Solving SVP-γ

SVP-γ problem: find $0 < \|\mathbf{s}\| \leq \gamma \cdot \lambda_1(\Lambda)$. The main tool is the so-called BKZ algorithm with parameter β.

- To solve approximated SVP-γ where $\gamma = \exp(n \log \beta / \beta)$, we need BKZ-$\beta$ that takes $2^{c\beta}$ (best asymptotic) for some c.
    - $\gamma \geq \exp(n \frac{\log \log n}{\log n})$, polynomial time;
    - $\cdots$;
    - $\gamma = \text{poly}(n)$, $\beta = \Theta(n)$;

uSVP-γ (Unique Shortest Vector Problem) (e.g. promised gap $\gamma \geq \lambda_2 / \lambda_1$).

First strategy: lattice of dim m with $\gamma \approx \frac{q^{-n/m}}{\alpha}$. This gives running-time (for BKZ):

$$\left(\frac{n \log q}{\log^2 \alpha}\right)^{O\left(\frac{n \log q}{\log^2 \alpha}\right)}.$$