Algorithmic Geometry of Numbers

New and old algorithms and open problems around shortest and closest lattice vectors

EuroCG, March 29, 2011

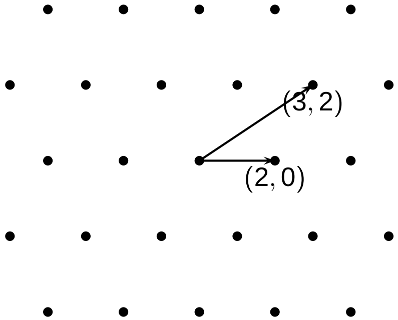
Friedrich Eisenbrand
EPFL

1. Geometry of Numbers

Lattices

Lattice

$\Lambda(A) = \{Ax : x \in \mathbb{Z}^n\}$, where $A \in \mathbb{R}^{n \times n}$ is nonsingular matrix



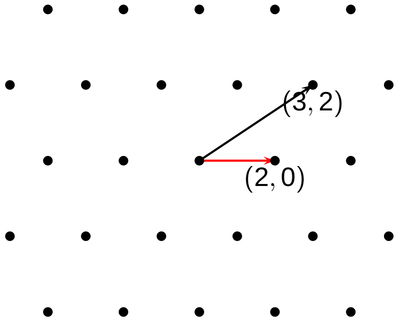
A Central Problem : Shortest Vector

Given basis A, find shortest nonzero vector of $\Lambda(A)$.

Lattices

Lattice

$\Lambda(A) = \{Ax : x \in \mathbb{Z}^n\}$, where $A \in \mathbb{R}^{n \times n}$ is nonsingular matrix



A Central Problem : Shortest Vector

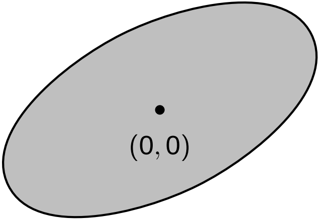
Given basis A, find shortest nonzero vector of $\Lambda(A)$.

Minkowski's theorem

Existence of short vectors

Let $K \subseteq \mathbb{R}^n$ be convex body which is symmetric around origin ($x \in K$ implies $-x \in K$).

$$\text{If } \mathrm{vol}(K) > 2^n \text{, then } K \text{ contains } v \in \mathbb{Z}^n \setminus \{0\}.$$

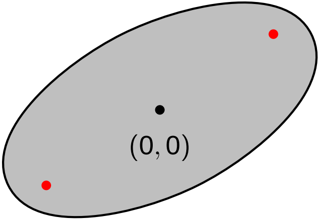


Minkowski's theorem

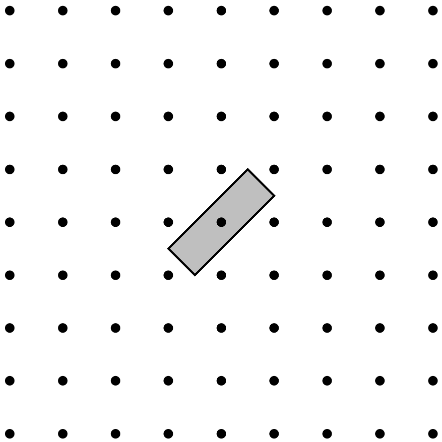
Existence of short vectors

Let $K \subseteq \mathbb{R}^n$ be convex body which is symmetric around origin ($x \in K$ implies $-x \in K$).

$$\text{If } \mathrm{vol}(K) > 2^n \text{, then } K \text{ contains } v \in \mathbb{Z}^n \setminus \{0\}.$$

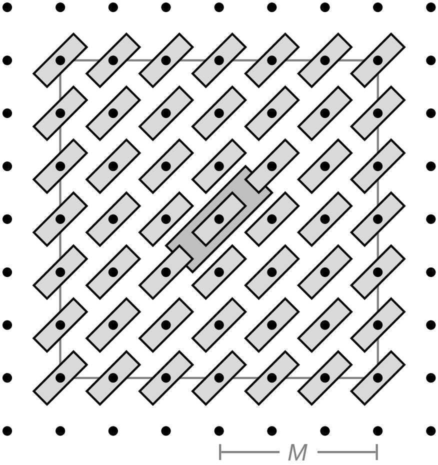


Proof



To show : If $K \cap \mathbb{Z} = \{0\}$, then $\mathrm{vol}(K) \leqslant 2^n$.

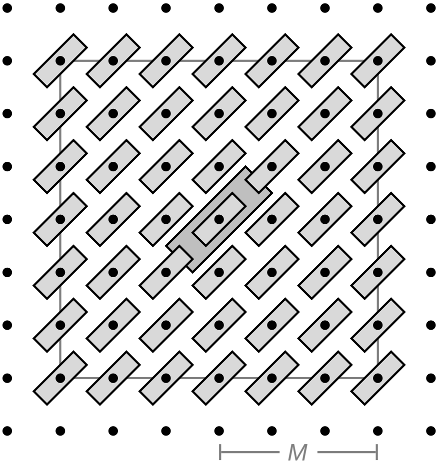
Proof



Translates $(1/2) \cdot K + v$, $v \in \mathbb{Z}^n$, $\|v\|_\infty \leqslant M$ cannot intersect!

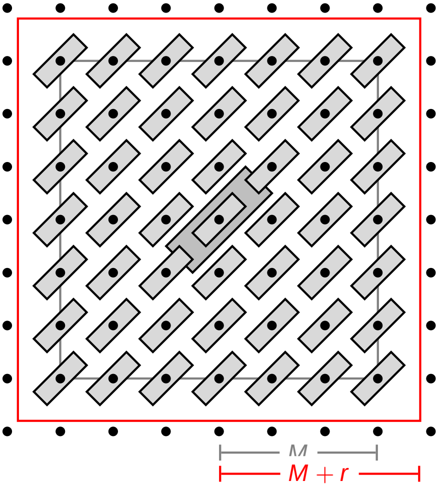
$(1/2)k_1 + v_1 = (1/2)k_2 + v_2$ implies $0 \neq v_1 - v_2 = (1/2)k_2 - (1/2)k_1 \in K$.

Proof



Union of translates : $\mathrm{vol} = (\mathbf{2} \cdot \mathbf{M} + \mathbf{1})^{\mathbf{n}} \cdot \mathrm{vol}(K)/2^n$.

Proof

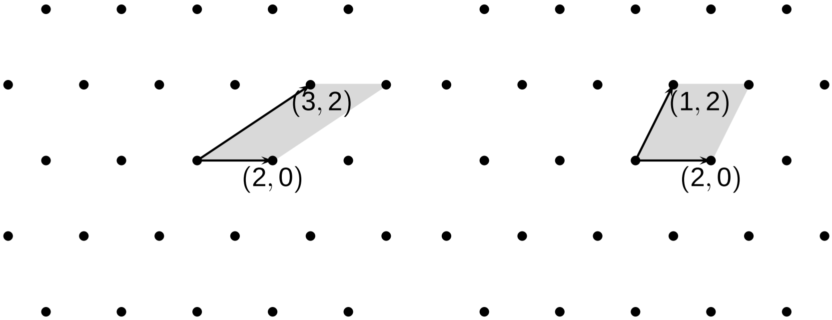


Union of translates : $\mathrm{vol} = \mathbf{(2 \cdot M + 1)^n} \cdot \mathrm{vol}(K)/2^n$.
Outer box : $\mathrm{vol} = \mathbf{(2M + 2r)^n}$.
As $M \to \infty$, ratio of blue numbers $\to 1$ and thus $\mathrm{vol}(K)/2^n \leqslant 1$.

Lattice determinant

Determinant of Λ

A basis of Λ, then $|\det(A)|$ is invariant of Λ called lattice determinant

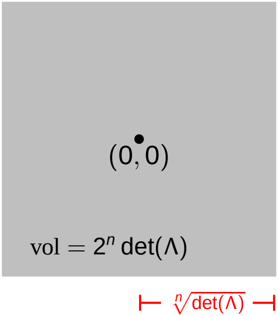


Minkowski's convex body theorem (V 2.0)

Let $\Lambda \subseteq \mathbb{R}^n$ be a lattice and $K \subseteq \mathbb{R}^n$ be a convex body of volume $\mathrm{vol}(K) \geqslant 2^n \det(\Lambda)$ that is symmetric about the origin. K contains nonzero lattice point.

Existence of short vectors

Λ has nonzero lattice point v with $\|v\|_\infty \leqslant \sqrt[n]{\det(\Lambda)}$

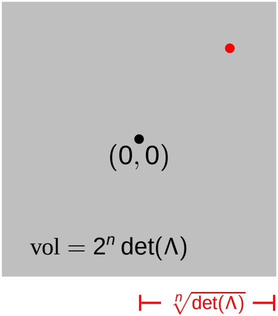


Minkowski's convex body theorem (V 2.0)

Let $\Lambda \subseteq \mathbb{R}^n$ be a lattice and $K \subseteq \mathbb{R}^n$ be a convex body of volume $\mathrm{vol}(K) \geqslant 2^n \det(\Lambda)$ that is symmetric about the origin. K contains nonzero lattice point.

Existence of short vectors

Λ has nonzero lattice point v with $\|v\|_\infty \leqslant \sqrt[n]{\det(\Lambda)}$



2. Application : Diophantine Approximation

Rounding a vector

Theorem (Dirichlet)

Given : $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and $Q \in \mathbb{N}$
There exist : $q \in \mathbb{N}$ and $p_1, \ldots, p_n \in \mathbb{Z}$ with

$$1 \leqslant q \leqslant Q^n \quad \text{and} \quad |q\,\alpha_i - p_i| < 1/Q.$$

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & \alpha(1) \\ 0 & 1 & \cdots & 0 & \alpha(2) \\ & & \ddots & & \\ 0 & 0 & \cdots & 1 & \alpha(n) \\ 0 & 0 & \cdots & 0 & 1/Q^{n+1} \end{pmatrix}$$

$$\sqrt[n+1]{\det(\Lambda)} = 1/Q$$

Rounding a vector

Theorem (Dirichlet)

Given : $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and $Q \in \mathbb{N}$
There exist : $q \in \mathbb{N}$ and $p_1, \ldots, p_n \in \mathbb{Z}$
with

$$1 \leqslant q \leqslant Q^n \quad \text{and} \quad |q\,\alpha_i - p_i| < 1/Q.$$

$$\begin{array}{c} \begin{array}{ccccc} \mathbf{p}_1 & \mathbf{p}_2 & \cdots & \mathbf{p}_n & \mathbf{q} \end{array} \\ \begin{pmatrix} 1 & 0 & \cdots & 0 & \alpha(1) \\ 0 & 1 & \cdots & 0 & \alpha(2) \\ & & \ddots & & \\ 0 & 0 & \cdots & 1 & \alpha(n) \\ 0 & 0 & \cdots & 0 & 1/Q^{n+1} \end{pmatrix} \end{array}$$

$$\sqrt[n+1]{\det(\Lambda)} = 1/Q$$

Rounding a vector

Theorem (Dirichlet)

Given : $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and $Q \in \mathbb{N}$
There exist : $q \in \mathbb{N}$ and $p_1, \ldots, p_n \in \mathbb{Z}$
with

$$1 \leqslant q \leqslant Q^n \quad \text{and} \quad |q\,\alpha_i - p_i| < 1/Q.$$

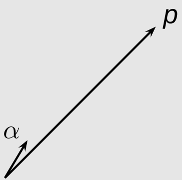
$$\begin{array}{c}
\begin{array}{ccccc} \mathbf{p}_1 & \mathbf{p}_2 & \cdots & \mathbf{p}_n & \mathbf{q} \end{array} \\
\begin{pmatrix}
1 & 0 & \cdots & 0 & \alpha(1) \\
0 & 1 & \cdots & 0 & \alpha(2) \\
 & & \ddots & & \\
0 & 0 & \cdots & 1 & \alpha(n) \\
0 & 0 & \cdots & 0 & 1/Q^{n+1}
\end{pmatrix}
\end{array}$$

$$\sqrt[n+1]{\det(\Lambda)} = 1/Q$$

Rounding $c \in \mathbb{Z}^n$

Set $\alpha := c/\|c\|_\infty$.

1. p is integer vector with small angle to c
2. $\|p\|_\infty \leqslant Q^n$



Strongly polynomial algorithms

Weakly polynomial 0/1 optimization problem

Feasible points : $\mathscr{F} \subseteq 2^{\{0,1\}^n}$
Problem : $\max_{x \in \mathscr{F}} c^T x$

- Optimization problem is polynomial.
- Running time depends on n **and** binary encoding length of c.

Strongly polynomial algorithms

Weakly polynomial 0/1 optimization problem

Feasible points : $\mathscr{F} \subseteq 2^{\{0,1\}^n}$
Problem : $\max_{x \in \mathscr{F}} c^T x$

- Optimization problem is polynomial.
- Running time depends on n **and** binary encoding length of c.

Weakly and strongly polynomial are equivalent for $0/1$ problems

With Dirichlet : Replace c by a d such that

- $x \in \mathscr{F}$ is optimal w.r.t. c if and only if x is optimal w.r.t. d.
- Binary encoding length of d is polynomial in the dimension n.

(Frank and Tardos 1987)

Some more detailed explanations

Round c

Set $\alpha := c/\|c\|_\infty$ and $Q := n$
Apply Dirichlet and obtain $q \in \mathbb{Z}$ and $p \in \mathbb{Z}^n$

Some more detailed explanations

Round c

Set $\alpha := c/\|c\|_\infty$ and $Q := n$
Apply Dirichlet and obtain $q \in \mathbb{Z}$ and $p \in \mathbb{Z}^n$

Norm of p :

i) $1 \leqslant q \leqslant n^n$
ii) $|q\,\alpha_i - p_i| < 1/n$

Together with $\|\alpha\|_\infty = 1$ implies

$$\|\mathbf{p}\|_\infty \leqslant \mathbf{n^n}.$$

Some more detailed explanations

Round c

Set $\alpha := c/\|c\|_\infty$ and $Q := n$
Apply Dirichlet and obtain $q \in \mathbb{Z}$ and $p \in \mathbb{Z}^n$

Norm of p :

i) $1 \leqslant q \leqslant n^n$
ii) $|q\,\alpha_i - p_i| < 1/n$

Together with $\|\alpha\|_\infty = 1$ implies

$$\|\mathbf{p}\|_\infty \leqslant \mathbf{n}^{\mathbf{n}}.$$

Polynomial encoding length !

Some more detailed explanations

Round c

Set $\alpha := c/\|c\|_\infty$ and $Q := n$
Apply Dirichlet and obtain $q \in \mathbb{Z}$ and $p \in \mathbb{Z}^n$

Norm of p :

i) $1 \leqslant q \leqslant n^n$
ii) $|q\,\alpha_i - p_i| < 1/n$

Together with $\|\alpha\|_\infty = 1$ implies

$$\|\mathbf{p}\|_\infty \leqslant \mathbf{n}^{\mathbf{n}}.$$

Polynomial encoding length !

How good is p in place of c (or α) ?

- $\overline{x} \in \mathscr{F}$ is optimal solution if and only if $\alpha^T(\overline{x} - x) \geqslant 0$ for all $x \in \mathscr{F}$.

Some more detailed explanations

Round c

Set $\alpha := c/\|c\|_\infty$ and $Q := n$
Apply Dirichlet and obtain $q \in \mathbb{Z}$ and $p \in \mathbb{Z}^n$

Norm of p :

i) $1 \leqslant q \leqslant n^n$
ii) $|q\,\alpha_i - p_i| < 1/n$

Together with $\|\alpha\|_\infty = 1$ implies

$$\|\mathbf{p}\|_\infty \leqslant \mathbf{n}^{\mathbf{n}}.$$

Polynomial encoding length !

How good is p in place of c (or α) ?

- $\overline{x} \in \mathscr{F}$ is optimal solution if and only if $\alpha^T(\overline{x} - x) \geqslant 0$ for all $x \in \mathscr{F}$.
- Optimal solutions w.r.t. α and p are the same if

$$\mathrm{sign}(p^T y) = \mathrm{sign}(\alpha^T y) \quad \text{for all} \quad y \in \{0, \pm 1\}^n$$

p is almost as good as α

Claim

$\forall y \in \{0, \pm 1\}^n$ one has

1. $p^T y > 0 \Longrightarrow c^T y > 0$
2. $p^T y < 0 \Longrightarrow c^T y < 0$

p is almost as good as α

Claim

$\forall y \in \{0, \pm 1\}^n$ one has

1. $p^T y > 0 \Longrightarrow c^T y > 0$
2. $p^T y < 0 \Longrightarrow c^T y < 0$

Reminder

i) $1 \leqslant q \leqslant n^n$

ii) $|q\,\alpha_i - p_i| < 1/n$

p is almost as good as α

Claim

$\forall y \in \{0, \pm 1\}^n$ one has

1. $p^T y > 0 \Longrightarrow c^T y > 0$
2. $p^T y < 0 \Longrightarrow c^T y < 0$

Reminder

i) $1 \leqslant q \leqslant n^n$

ii) $|q\,\alpha_i - p_i| < 1/n$

Proof of claim

Suppose $p^T y > 0$

- $p^T y \geqslant 1$, since p and y integral.
- $|q\,\alpha_i - p_i| < 1/n$
- Thus $|q\,\alpha^T y - p^T y| = |(q\,\alpha - p)^T y| < n/n = 1$.
- Therefore $\alpha^T y > 0$

p is almost as good as α

Claim

$\forall y \in \{0, \pm 1\}^n$ one has

1. $p^T y > 0 \Longrightarrow c^T y > 0$
2. $p^T y < 0 \Longrightarrow c^T y < 0$

Reminder

i) $1 \leqslant q \leqslant n^n$

ii) $|q\,\alpha_i - p_i| < 1/n$

What if $\mathbf{p^T y = 0}$? : Apply recursion

- Recursively find $v \in \mathbb{Z}^n$ such that $\forall y \in \{0, \pm 1\}^n$ $\operatorname{sign}(v^T y) = \operatorname{sign}((q \cdot \alpha - p)^T y)$.
- Let M be a large weight, then

$$\forall y \in \{0, \pm 1\}^n: \ \operatorname{sign}(c^T y) = \operatorname{sign}\left((M \cdot p + v)^T y\right)$$

- Binary encoding length of $\mathbf{M} \cdot \mathbf{p} + \mathbf{v}$ is polynomial even if exponential approximation of shortest vector (LLL) is used.
- Weakly and strongly polynomial are equivalent notions for $0/1$ optimization problems. (Frank and Tardos 1987)

Complexity of simultaneous diophantine approximation

Best denominator problem

Given : $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$, $\varepsilon > 0$
Find : Minimal $Q \in \mathbb{N}_{\geqslant 1}$ with

$$|Q \cdot \alpha_i - \lfloor Q \cdot \alpha_i \rceil| < \varepsilon \text{ for all } i = 1, \ldots, n.$$

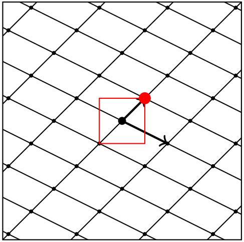
- NP-hard (Lagarias 1985)
- Hard to approximate within $2^{n/2}$ unless $\mathrm{NP} = \mathrm{co} - \mathrm{NP}$ (Lagarias 1985)
- Hard to approximate within 2^n unless $\mathrm{P} = \mathrm{NP}$ (E. & Rothvoß 2009)

3. New Algorithms for Shortest Vector and Closest Vector

Two classics

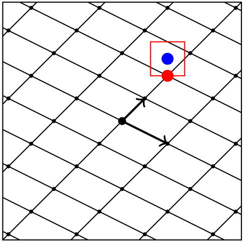
Shortest Vector

Given : Lattice basis $A \in \mathbb{Z}^{n \times n}$.
Task : Find a shortest nonzero vector in $\Lambda(A)$



Closest Vector

Given : Lattice basis $A \in \mathbb{Z}^{n \times n}$, target vector $t \in \mathbb{Z}^n$.
Task : Find a vector in Λ closest to t.



History of Shortest Vector

Quest for a singly exponential algorithm

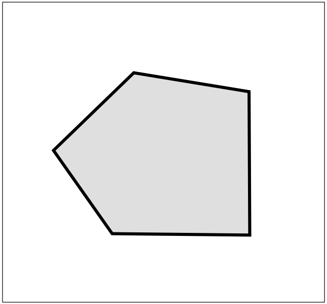
- First algorithms : Lagrange (1775), Gauss (1801), Hermite (1850)
- Lenstra, Lenstra & Lovász (1982) : Polynomial 2^n-approximation algorithm
- Lenstra (1983) : $2^{O(n^3)}$- algorithm
- Kannan (1987) : $n^{O(n)}$-algorithm
- Ajtai, Kumar and Sievakumar (2001) : $2^{O(n)}$-randomized algorithm for any ℓ_p-norm (Blömer & Naewe 2007)
- Micciancio & Voulgaris (2010) : $2^{O(n)}$-deterministic algorithm for $\ell_\mathbf{2}$ only !
- Singly exp. algorithms use $\Omega(\mathbf{2^n})$-space.

History of closest vector

Quest for a singly exponential algorithm

- Lenstra (1983) : $2^{O(n^3)}$-algorithm for closest vector and
- Kannan (1987) : $n^{O(n)}$-algorithm for closest vector and integer programming
- Blömer & Naewe (2007) : $(1+\varepsilon)$-approx. CVP for all ℓ_p-norms in time $O(1/\varepsilon)^n$ (randomized)
- Micciancio & Voulgaris (2010) : $2^{O(n)}$-deterministic algorithm for ℓ_2 only !
- E., Hähnle & Niemeier (2011) : $(1+\varepsilon)$-approx. CVP for ℓ_∞-norm only in time $O(\log(1/\varepsilon))^n$ (randomized)

Transference bounds and the $n^{\Omega(n)}$-barrier

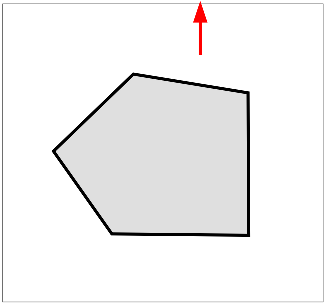


If $K \cap \mathbb{Z}^n = \emptyset$ then there exists $d \in \mathbb{Z}^n - \{0\}$ with

$$\max_{x \in K} d^T x - \min_{x \in K} d^T x = O(n^{3/2})$$

(Banaszczyk 1996)

Transference bounds and the $n^{\Omega(n)}$-barrier

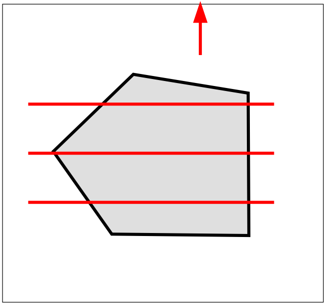


Compute direction $d \in \mathbb{Z}^d - \{0\}$ minimizing

$$\max_{x \in K} d^T x - \min_{x \in K} d^T x$$

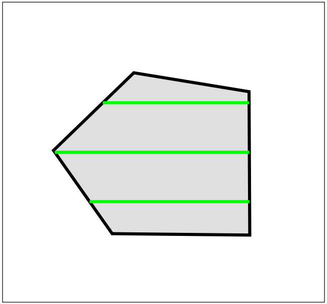
If width too large, then $K \cap \mathbb{Z}^n \neq \emptyset$

Transference bounds and the $n^{\Omega(n)}$-barrier



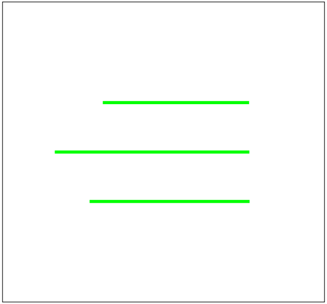
Otherwise search for integer point recursively on one of the $O(n^{3/2}$ hyperplanes $(d^T x = \delta) \cap P$, $\delta \in \mathbb{Z}$

Transference bounds and the $n^{\Omega(n)}$-barrier



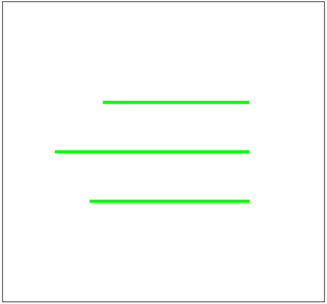
Otherwise search for integer point recursively on one of the $O(n^{3/2}$ hyperplanes $(d^T x = \delta) \cap P$, $\delta \in \mathbb{Z}$

Transference bounds and the $n^{\Omega(n)}$-barrier



Otherwise search for integer point recursively on one of the $O(n^{3/2}$ hyperplanes $(d^T x = \delta) \cap P$, $\delta \in \mathbb{Z}$

Transference bounds and the $n^{\Omega(n)}$-barrier



Otherwise search for integer point recursively on one of the $O(n^{3/2})$ hyperplanes $(d^T x = \delta) \cap P$, $\delta \in \mathbb{Z}$

Analysis

$T(n) = n^{3/2} \cdot T(n-1) \leqslant (n^{3/2})^n = n^{O(n)}$. (Kannan 1987)
Advantage : Polynomial space

4. Detailed Explanation

of the randomized singly exponential algorithm by Ajtai et al. (2001)

Singly Exponential Algorithm for SV

Ajtai, Kumar and Sivakumar (2001)

There exists a randomized $2^{O(n)}$ algorithm for shortest vector.

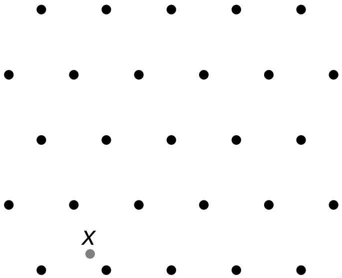
Simplifying assumption on basis $B = (b_1, \ldots, b_n)$

- $2 \leqslant SV(\Lambda) \leqslant 3$
- Achievable via scaling with powers of $2/3$

Promise direction

A promise direction of $x \in \mathbb{R}^n$, $\mathrm{prom}(x) \in \mathbb{R}^n$, is vector satisfying

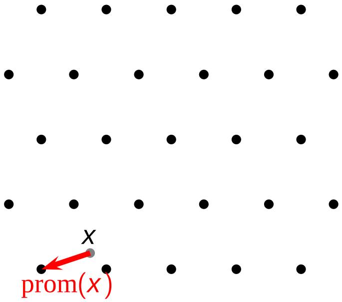
$$x + \mathrm{prom}(x) \in \Lambda.$$



Promise direction

A promise direction of $x \in \mathbb{R}^n$, $\mathrm{prom}(x) \in \mathbb{R}^n$, is vector satisfying

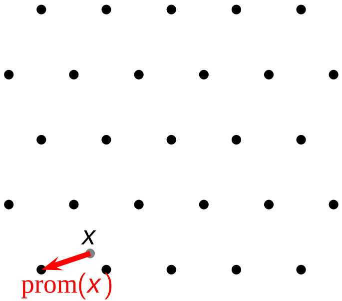
$$x + \mathrm{prom}(x) \in \Lambda.$$



Promise direction

A promise direction of $x \in \mathbb{R}^n$, $\mathrm{prom}(x) \in \mathbb{R}^n$, is vector satisfying

$$x + \mathrm{prom}(x) \in \Lambda.$$



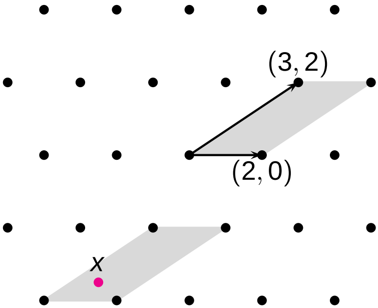
Desired property of promise direction

$$\forall u \in \Lambda, x \in \mathbb{R}^n : \mathrm{prom}(x) = \mathrm{prom}(x + u)$$

Example of Promise Direction

For $x \in \mathbb{R}^n$ write $x = \sum_{i=1}^{n} \lambda_i b_i$ (B is basis of $\mathbb{R}^n$) an define

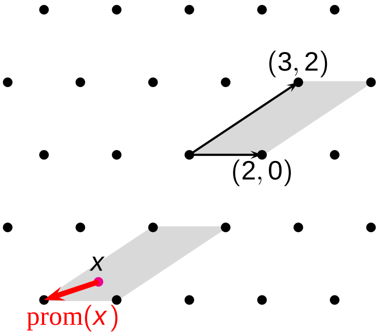
$$\mathrm{prom}(x) := \sum_{i=1}^{n} (\lfloor \lambda_i \rfloor - \lambda_i) b_i .$$



Example of Promise Direction

For $x \in \mathbb{R}^n$ write $x = \sum_{i=1}^n \lambda_i b_i$ (B is basis of $\mathbb{R}^n$) an define

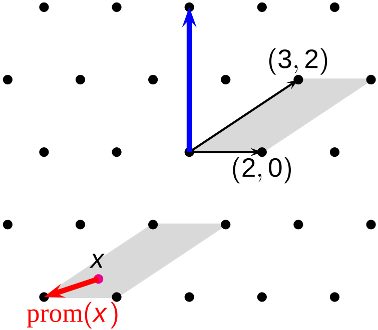
$$\operatorname{prom}(x) := \sum_{i=1}^{n} (\lfloor \lambda_i \rfloor - \lambda_i) b_i .$$



Example of Promise Direction

For $x \in \mathbb{R}^n$ write $x = \sum_{i=1}^{n} \lambda_i b_i$ (B is basis of $\mathbb{R}^n$) an define

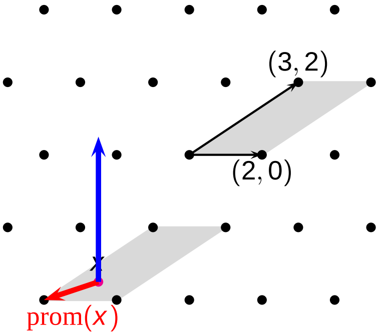
$$\mathrm{prom}(x) := \sum_{i=1}^{n} (\lfloor \lambda_i \rfloor - \lambda_i) b_i .$$



Example of Promise Direction

For $x \in \mathbb{R}^n$ write $x = \sum_{i=1}^n \lambda_i b_i$ (B is basis of $\mathbb{R}^n$) an define

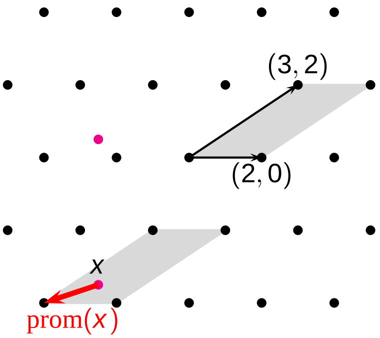
$$\mathrm{prom}(x) := \sum_{i=1}^{n} (\lfloor \lambda_i \rfloor - \lambda_i) b_i .$$



Example of Promise Direction

For $x \in \mathbb{R}^n$ write $x = \sum_{i=1}^{n} \lambda_i b_i$ (B is basis of $\mathbb{R}^n$) an define

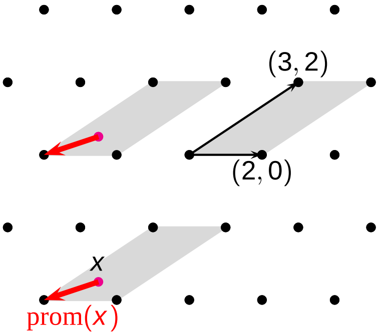
$$\text{prom}(x) := \sum_{i=1}^{n} (\lfloor \lambda_i \rfloor - \lambda_i) b_i.$$



Example of Promise Direction

For $x \in \mathbb{R}^n$ write $x = \sum_{i=1}^n \lambda_i b_i$ (B is basis of $\mathbb{R}^n$) an define

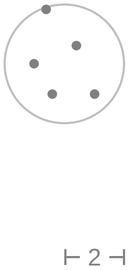
$$\mathrm{prom}(x) := \sum_{i=1}^{n} (\lfloor \lambda_i \rfloor - \lambda_i) b_i .$$



Sampling and Sieving

Initialization of Algorithm

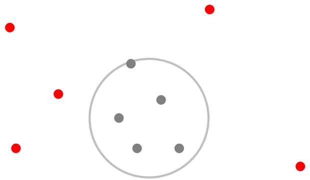
- Sample exponentially many points $x_1, \ldots, x_N \in B_2(0)$



Sampling and Sieving

Initialization of Algorithm

- Sample exponentially many points $x_1, \ldots, x_N \in B_2(0)$
- Compute promise directions $\mathrm{prom}(x_1), \ldots, \mathrm{prom}(x_N)$ and store list $\{(x_1, \mathrm{prom}(x_1)), \ldots, (x_N, \mathrm{prom}(x_N))\}$

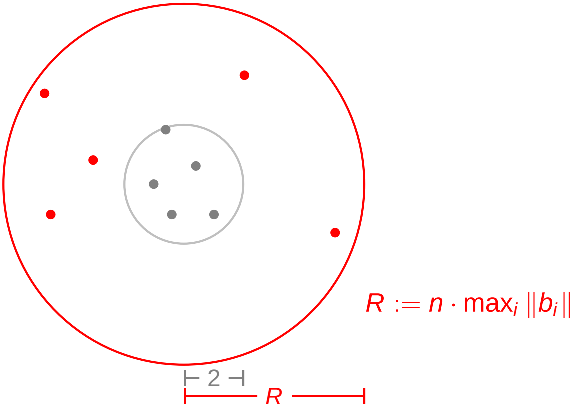


$\vdash 2 \dashv$

Sampling and Sieving

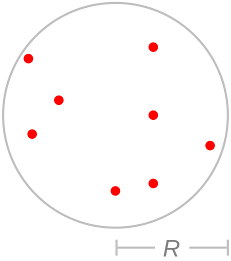
Initialization of Algorithm

- Sample exponentially many points $x_1, \ldots, x_N \in B_2(0)$
- Compute promise directions $\mathrm{prom}(x_1), \ldots, \mathrm{prom}(x_N)$ and store list $\{(x_1, \mathrm{prom}(x_1)), \ldots, (x_N, \mathrm{prom}(x_N))\}$



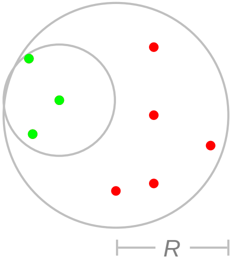
Sieving

- Consider promise directions $\mathrm{prom}(x_1), \ldots, \mathrm{prom}(x_N)$. In the beginning they are red



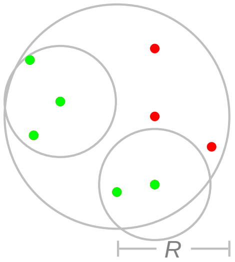
Sieving

- Consider promise directions $\mathrm{prom}(x_1), \ldots, \mathrm{prom}(x_N)$. In the beginning they are red
- Pick arbitrary red point and color all points within distance $R/2$ to this point green



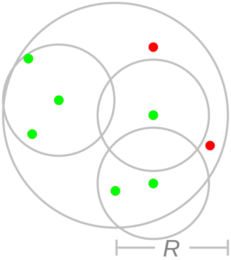
Sieving

- Consider promise directions $\mathrm{prom}(x_1), \ldots, \mathrm{prom}(x_N)$. In the beginning they are red
- Pick arbitrary red point and color all points within distance $R/2$ to this point green
- Repeat until no red points left



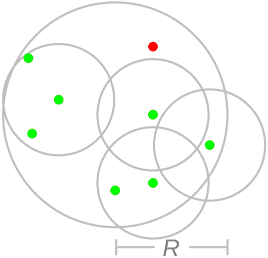
Sieving

- Consider promise directions $\mathrm{prom}(x_1), \dots, \mathrm{prom}(x_N)$. In the beginning they are red
- Pick arbitrary red point and color all points within distance $R/2$ to this point green
- Repeat until no red points left



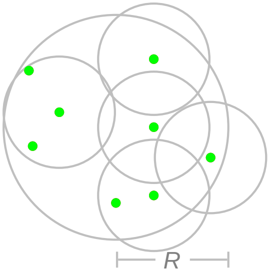
Sieving

- Consider promise directions $\mathrm{prom}(x_1), \ldots, \mathrm{prom}(x_N)$. In the beginning they are red
- Pick arbitrary red point and color all points within distance $R/2$ to this point green
- Repeat until no red points left



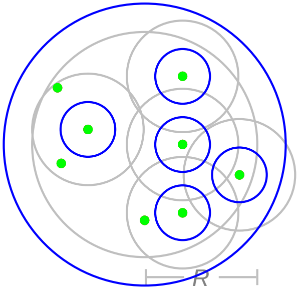
Sieving

- Consider promise directions $\mathrm{prom}(x_1), \ldots, \mathrm{prom}(x_N)$. In the beginning they are red
- Pick arbitrary red point and color all points within distance $R/2$ to this point green
- Repeat until no red points left
- How many centers of smaller balls ?



Sieving

- Consider promise directions $\mathrm{prom}(x_1), \ldots, \mathrm{prom}(x_N)$. In the beginning they are red
- Pick arbitrary red point and color all points within distance $R/2$ to this point green
- Repeat until no red points left
- How many centers of smaller balls ?



Number of centers bounded by

$$\frac{\mathrm{vol}(B_{5/4}(0))}{\mathrm{vol}(B_{1/4}(0))} = 5^n$$

The Algorithm

While $R > 6$

- Apply sieving algorithm to the vectors $\mathrm{prom}(x_i)$ for each $(x_i, \mathrm{prom}(x_i))$ in list
- Delete from list all tuples $(x_i, \mathrm{prom}(x_i))$, where $\mathrm{prom}(x_i)$ is a center of the sieving procedure
- Replace $(x_j, \mathrm{prom}(x_j))$ with

  $$(x_j, \mathrm{prom}(x_j) - (\mathrm{prom}(x_i) + x_i)$$

  where $\mathrm{prom}(x_i)$ was center of $\mathrm{prom}(x_j)$
- $R \leftarrow R/2 + 2$.

The Algorithm

While $R > 6$

- Apply sieving algorithm to the vectors $\mathrm{prom}(x_i)$ for each $(x_i, \mathrm{prom}(x_i))$ in list
- Delete from list all tuples $(x_i, \mathrm{prom}(x_i))$, where $\mathrm{prom}(x_i)$ is a center of the sieving procedure
- Replace $(x_j, \mathrm{prom}(x_j))$ with

$$(x_j, \mathrm{prom}(x_j) - (\mathrm{prom}(x_i) + x_i)$$

  where $\mathrm{prom}(x_i)$ was center of $\mathrm{prom}(x_j)$
- $R \leftarrow R/2 + 2$.

Output

For each remaining $(x_i, \mathrm{prom}(x_i))$ compute lattice vector $x_i + \mathrm{prom}(x_i)$ and output shortest nonzero one

Invariants and Number of Iterations

$\|\text{prom}(x_j)\| \leqslant R$

$$\begin{aligned}
\|\text{prom}_{new}(x_j)\| &= \|\text{prom}(x_j) - \text{prom}(x_i) - x_i\| \\
&\leqslant \|\text{prom}(x_j) - \text{prom}(x_i)\| + \|x_i\| \\
&\leqslant R/2 + 2 = R_{new}
\end{aligned}$$

Invariants and Number of Iterations

$\|\mathrm{prom}(x_j)\| \leqslant R$

$$\begin{aligned}\|\mathrm{prom}_{new}(x_j)\| &= \|\mathrm{prom}(x_j) - \mathrm{prom}(x_i) - x_i\| \\ &\leqslant \|\mathrm{prom}(x_j) - \mathrm{prom}(x_i)\| + \|x_i\| \\ &\leqslant R/2 + 2 = R_{new}\end{aligned}$$

Number of iterations

Bounded by $O(\log R_0)$ with $R_0 = n \cdot \max_i \|b_i\|$

Invariants and Number of Iterations

$\|\mathrm{prom}(x_j)\| \leqslant R$

$$
\begin{aligned}
\|\mathrm{prom}_{new}(x_j)\| &= \|\mathrm{prom}(x_j) - \mathrm{prom}(x_i) - x_i\| \\
&\leqslant \|\mathrm{prom}(x_j) - \mathrm{prom}(x_i)\| + \|x_i\| \\
&\leqslant R/2 + 2 = R_{new}
\end{aligned}
$$

Number of iterations

Bounded by $O(\log R_0)$ with $R_0 = n \cdot \max_i \|b_i\|$

Number of deleted tuples

$O(\log R_0 \cdot 5^n)$

Invariants and Number of Iterations

$\|\mathrm{prom}(x_j)\| \leqslant R$

$$\begin{aligned} \|\mathrm{prom}_{new}(x_j)\| &= \|\mathrm{prom}(x_j) - \mathrm{prom}(x_i) - x_i\| \\ &\leqslant \|\mathrm{prom}(x_j) - \mathrm{prom}(x_i)\| + \|x_i\| \\ &\leqslant R/2 + 2 = R_{new} \end{aligned}$$

Number of iterations

Bounded by $O(\log R_0)$ with $R_0 = n \cdot \max_i \|b_i\|$

Number of deleted tuples

$O(\log R_0 \cdot 5^n)$

Length of generated lattice points

$\|x_i + \mathrm{prom}(x_i)\| \leqslant 2 + 6 = 8$. Short!

Invariants and Number of Iterations

$\|\mathrm{prom}(x_j)\| \leqslant R$

$$\begin{aligned} \|\mathrm{prom}_{new}(x_j)\| &= \|\mathrm{prom}(x_j) - \mathrm{prom}(x_i) - x_i\| \\ &\leqslant \|\mathrm{prom}(x_j) - \mathrm{prom}(x_i)\| + \|x_i\| \\ &\leqslant R/2 + 2 = R_{new} \end{aligned}$$

Number of iterations

Bounded by $O(\log R_0)$ with $R_0 = n \cdot \max_i \|b_i\|$

Number of deleted tuples

$O(\log R_0 \cdot 5^n)$

Zero is short

How can one guarantee that not all $x_j + \mathrm{prom}(x_j)$ are zero ?

A crucial observation

- Consider tuple $(x_i, \mathrm{prom}(x_i))$ before the output phase of the algorithm.

A crucial observation

- Consider tuple $(x_i, \mathrm{prom}(x_i))$ before the output phase of the algorithm.
- Alg. never queried x_i itself.

A crucial observation

- Consider tuple $(x_i, \mathrm{prom}(x_i))$ before the output phase of the algorithm.
- Alg. never queried x_i itself.
- Alg. queried $\mathrm{prom}(x_i)$ instead.

A crucial observation

- Consider tuple $(x_i, \mathrm{prom}(x_i))$ before the output phase of the algorithm.
- Alg. never queried x_i itself.
- Alg. queried $\mathrm{prom}(x_i)$ instead.
- Alg. would behave just the same until this point if x_i was replaced by

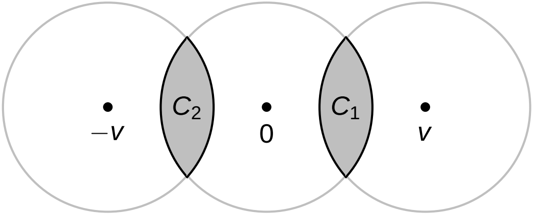
$$x_i + v \ \text{ for any } v \in \Lambda(B),$$

after initialization Step.

Gedankenexperiment

Sets C_1 and C_2

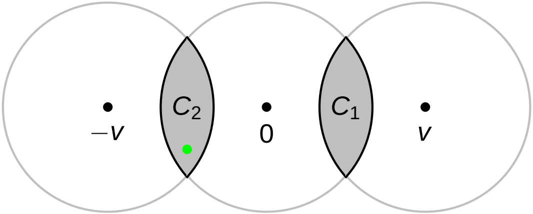
- Let $v \in \Lambda(B)$ be shortest vector ($2 \leqslant \|v\| \leqslant 3$)
- $C_1 := B_2(0) \cap B_2(v)$; $C_2 = B_2(0) \cap B_2(-v)$.



Gedankenexperiment

Sets C_1 and C_2

- Let $v \in \Lambda(B)$ be shortest vector ($2 \leqslant \|v\| \leqslant 3$)
- $C_1 := B_2(0) \cap B_2(v)$; $C_2 = B_2(0) \cap B_2(-v)$.



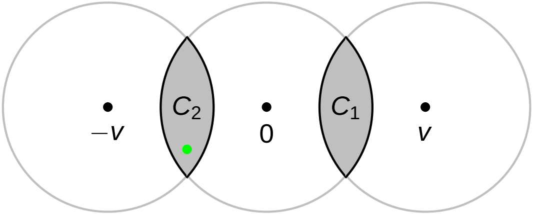
Toss coin

- If $x_i \in C_1 \cup C_2$ and tuple $(x_i, \mathrm{prom}(x_i))$ has survived

Gedankenexperiment

Sets C_1 and C_2

- Let $v \in \Lambda(B)$ be shortest vector ($2 \leqslant \|v\| \leqslant 3$)
- $C_1 := B_2(0) \cap B_2(v)$; $C_2 = B_2(0) \cap B_2(-v)$.



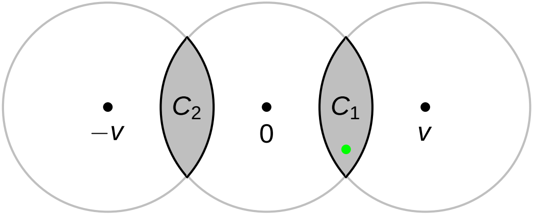
Toss coin

- If $x_i \in C_1 \cup C_2$ and tuple $(x_i, \mathrm{prom}(x_i))$ has survived
- If coin shows head, flip x_i to other side

Gedankenexperiment

Sets C_1 and C_2

- Let $v \in \Lambda(B)$ be shortest vector ($2 \leqslant \|v\| \leqslant 3$)
- $C_1 := B_2(0) \cap B_2(v)$; $C_2 = B_2(0) \cap B_2(-v)$.



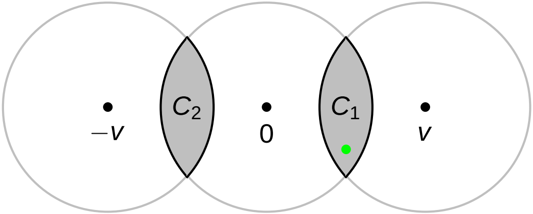
Toss coin

- If $x_i \in C_1 \cup C_2$ and tuple $(x_i, \mathrm{prom}(x_i))$ has survived
- If coin shows head, flip x_i to other side

Gedankenexperiment

Sets C_1 and C_2

- Let $v \in \Lambda(B)$ be shortest vector ($2 \leqslant \|v\| \leqslant 3$)
- $C_1 := B_2(0) \cap B_2(v)$; $C_2 = B_2(0) \cap B_2(-v)$.



Toss coin

- If $x_i \in C_1 \cup C_2$ and tuple $(x_i, \mathrm{prom}(x_i))$ has survived
- If coin shows head, flip x_i to other side
- Prob. of $x_i + \mathrm{prom}(x_i) = 0$ is $\leqslant 1/2$

Many sampled points will be in $C_1 \cup C_2$

Volume of C_1 and C_2

$\mathrm{vol}(C_1)/\mathrm{vol}(B_2(0)) = \mathrm{vol}(C_2)/\mathrm{vol}(B_2(0)) \geqslant 2^{-2n}$

Sample size

If number of sampled points is $\Omega(\log(R_0) \cdot 5^{2n})$ then many points will survive in $C_1 \cup C_2$ and short nonzero vector is computed whp.

Many sampled points will be in $C_1 \cup C_2$

Volume of C_1 and C_2

$\mathrm{vol}(C_1)/\mathrm{vol}(B_2(0)) = \mathrm{vol}(C_2)/\mathrm{vol}(B_2(0)) \geqslant 2^{-2n}$

Sample size

If number of sampled points is $\Omega(\log(R_0) \cdot 5^{2n})$ then many points will survive in $C_1 \cup C_2$ and short nonzero vector is computed whp.

Theorem (Ajtai, Kumar and Sivakumar (2001))

There exists a simply exponential randomized algorithm for shortest vector.

Extensions

More recent results

- Blömer & Naewe (2007, 2009) generalize to arbitrary ℓ_p-norm
  Derandomization : (Dadush, Peikert & Vempala 2011)
- Blömer & Naewe (2007, 2009) also provide $(1+\varepsilon)$-approximation alg. for CVP for any ℓ_p-norm. Running time $O(1/\varepsilon)^n$
- E., Hähnle & Niemeier (2011) : $(1+\varepsilon)$-approximation alg. for CVP for ℓ_∞-norm. Running time $O\log(1/\varepsilon)^n$

Faster approximation alg. for CVP_∞

Why CVP_∞ is particularly interesting

- Integer programming : Decide whether $P = \{x \in \mathbb{R}^n \ : \ Ax \leqslant u\}$ contains integer point
- Reduce to IP-feasibility of $l \leqslant Ax \leqslant u$ (standard technique)
- Rescale : $u - l = \mathbf{1}$
- Define $t := \frac{l+u}{2}$: P contains integer point iff there exists $v \in \Lambda(A)$ with $\|v - t\|_\infty \leqslant \frac{1}{2}$

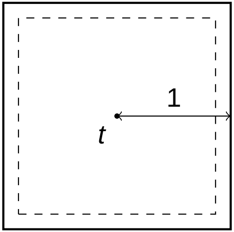
$(1+\varepsilon)$-approximate CVP$_\infty$

Given : Λ and t
Consider cubes

- $B := \{x \in \mathbb{R}^n \ : \ \|x - t\|_\infty \leqslant 1\}$
- $B' := \{x \in \mathbb{R}^n \ : \ \|x - t\|_\infty \leqslant (1-\varepsilon)\}$

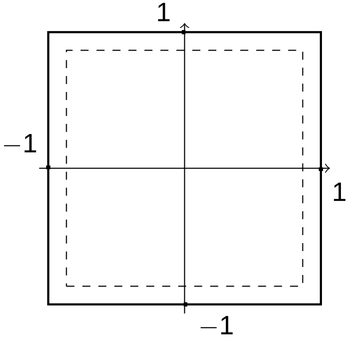
Task :

- Either : Find $v \in \Lambda \cap B$
- Or : Assert $\Lambda \cap B' = \emptyset$.



Boosting a 2-approximation algorithm

- Consider the unit hypercube $H := \{x \in \mathbb{R}^n \ : \ \|x\|_\infty \leqslant 1\}$,
- and a scaled cube $H' := \{x \in \mathbb{R}^n \ : \ \|x\|_\infty \leqslant 1 - \varepsilon\}$.

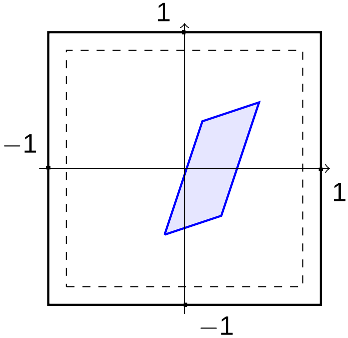


Question

How many parallelepipeds, if scaled by 2 from their centers of gravity are contained in H, are needed to cover H' ?

Boosting a 2-approximation algorithm

- Consider the unit hypercube $H := \{x \in \mathbb{R}^n \ : \ \|x\|_\infty \leqslant 1\}$,
- and a scaled cube $H' := \{x \in \mathbb{R}^n \ : \ \|x\|_\infty \leqslant 1 - \varepsilon\}$.

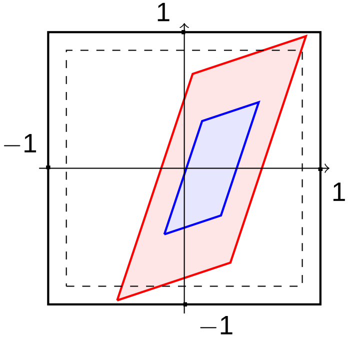


Question

How many parallelepipeds, if scaled by 2 from their centers of gravity are contained in H, are needed to cover H'?

Boosting a 2-approximation algorithm

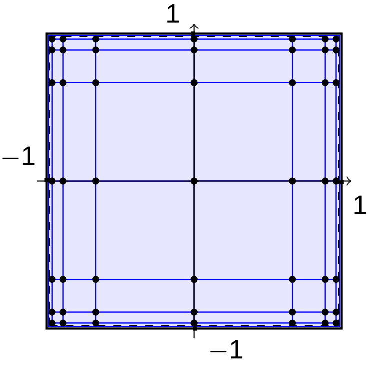
- Consider the unit hypercube $H := \{x \in \mathbb{R}^n \ : \ \|x\|_\infty \leqslant 1\}$,
- and a scaled cube $H' := \{x \in \mathbb{R}^n \ : \ \|x\|_\infty \leqslant 1 - \varepsilon\}$.



Question

How many parallelepipeds, if scaled by 2 from their centers of gravity are contained in H, are needed to cover H'?

An $O(\log(1/\varepsilon))^n$-algorithm



Number of parallelepipeds

At most $2^n(\log 1/\varepsilon)^n$

Theorem (E., Hähnle & Niemeier 2011)

There is a randomized algorithm to solve $(1+\varepsilon)$-gap CVP in time $(\log 1/\varepsilon)^n$.

Future Challenge

Open Problem

- Is there a simply exponential time and polynomial space alg. for SV, CVP and integer programming ?