On a recursive decoding algorithm for lattices

Annika Meyer

Workshop on lattices, codes and modular forms
Aachen, 27.09.2011

Overview

1. Introduction
2. Iterative lattice decoding
3. Upper bounds on the number of lattice points in a small sphere
4. Examples

Lattice Decoding: The Closest Vector Problem (CVP)

- Given a lattice L in $\mathbb{R}^n$ and $x \in \mathbb{R}^n$, the CVP consists in finding $\ell \in L$ such that

$$|x - \ell| = \min_{\ell' \in L} |x - \ell'|,$$

where $|\cdot|$ denotes the usual Euclidian length.

Lattice Decoding: The Closest Vector Problem (CVP)

- Given a lattice L in $\mathbb{R}^n$ and $x \in \mathbb{R}^n$, the CVP consists in finding $\ell \in L$ such that

$$|x - \ell| = \min_{\ell' \in L} |x - \ell'|,$$

where $|\cdot|$ denotes the usual Euclidian length.

- The CVP is NP hard in its *exact* version.

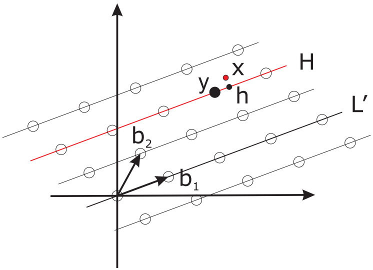
Lattice Decoding: The Closest Vector Problem (CVP)

- Given a lattice L in $\mathbb{R}^n$ and $x \in \mathbb{R}^n$, the CVP consists in finding $\ell \in L$ such that
$$|x-\ell| = \min_{\ell' \in L} |x-\ell'|,$$
where $|\cdot|$ denotes the usual Euclidian length.
- The CVP is NP hard in its *exact* version.
- Solving the CVP *with approximation factor* $\delta \geq 1 \in \mathbb{R}$ means finding $\ell \in L$ such that, for all $\ell' \in L$,
$$|x-\ell| \leq \delta \cdot |x-\ell'|.$$

Lattice Decoding: The Closest Vector Problem (CVP)

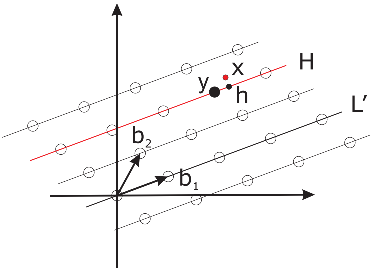
- Given a lattice L in $\mathbb{R}^n$ and $x \in \mathbb{R}^n$, the CVP consists in finding $\ell \in L$ such that
$$|x - \ell| = \min_{\ell' \in L} |x - \ell'|,$$
where $|\cdot|$ denotes the usual Euclidian length.
- The CVP is NP hard in its *exact* version.
- Solving the CVP *with approximation factor* $\delta \geq 1 \in \mathbb{R}$ means finding $\ell \in L$ such that, for all $\ell' \in L$,
$$|x - \ell| \leq \delta \cdot |x - \ell'|.$$
- The best known approximation factor for a deterministic polynomial time algorithm to solve the CVP approximately is $2^{n(\log\log n)^2/2\log n}$ (Schnorr 1985).

Babai's Nearest Plane Procedure (BNPP)



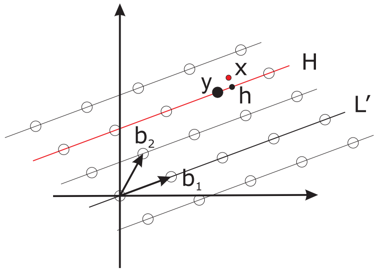
Given a basis $\mathcal{B} = (b_1, \ldots, b_n)$ of L and $x \in \mathbb{R}^n$, BNPP *approximates* x in L.

Babai's Nearest Plane Procedure (BNPP)



Given a basis $\mathcal{B} = (b_1, \ldots, b_n)$ of L and $x \in \mathbb{R}^n$, BNPP *approximates* x in L.
An approximation factor $2^{n/2}$ is achieved if $\mathcal{B}$ is LLL reduced.

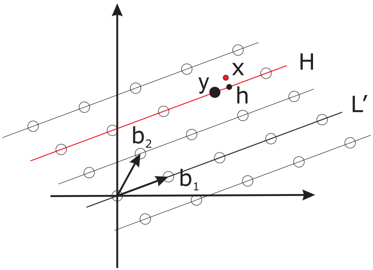
Babai's Nearest Plane Procedure (BNPP)



Given a basis $\mathcal{B} = (b_1, \ldots, b_n)$ of L and $x \in \mathbb{R}^n$, BNPP *approximates* x in L.
An approximation factor $2^{n/2}$ is achieved if $\mathcal{B}$ is LLL reduced.

(1) Let $L' = \langle b_1, \ldots, b_{n-1} \rangle_{\mathbb{Z}}$, then $L = \cup_{z \in \mathbb{Z}} z \cdot b_1 + L'$.

Babai's Nearest Plane Procedure (BNPP)

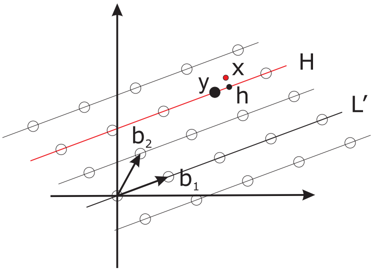


Given a basis $\mathcal{B} = (b_1, \ldots, b_n)$ of L and $x \in \mathbb{R}^n$, BNPP *approximates* x in L.
An approximation factor $2^{n/2}$ is achieved if $\mathcal{B}$ is LLL reduced.

(1) Let $L' = \langle b_1, \ldots, b_{n-1} \rangle_{\mathbb{Z}}$, then $L = \cup_{z \in \mathbb{Z}} z \cdot b_1 + L'$.

(2) Choose $H = zb_2 + L' \otimes \mathbb{R}$ closest to x and $h \in H$ closest to x.

Babai's Nearest Plane Procedure (BNPP)



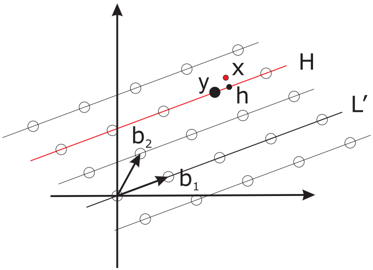
Given a basis $\mathcal{B} = (b_1, \ldots, b_n)$ of L and $x \in \mathbb{R}^n$, BNPP *approximates* x in L.
An approximation factor $2^{n/2}$ is achieved if $\mathcal{B}$ is LLL reduced.

(1) Let $L' = \langle b_1, \ldots, b_{n-1} \rangle_{\mathbb{Z}}$, then $L = \cup_{z \in \mathbb{Z}} z \cdot b_1 + L'$.

(2) Choose $H = zb_2 + L' \otimes \mathbb{R}$ closest to x and $h \in H$ closest to x.

(3) Iteratively, find an approximation y' of $h - zb_2$ in L'.

Babai's Nearest Plane Procedure (BNPP)



Given a basis $\mathcal{B} = (b_1, \ldots, b_n)$ of L and $x \in \mathbb{R}^n$, BNPP *approximates* x in L.
An approximation factor $2^{n/2}$ is achieved if $\mathcal{B}$ is LLL reduced.

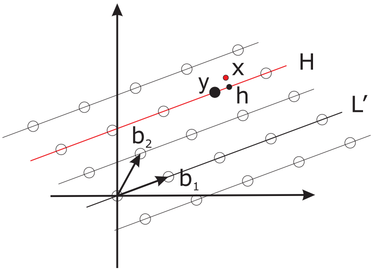
(1) Let $L' = \langle b_1, \ldots, b_{n-1} \rangle_{\mathbb{Z}}$, then $L = \cup_{z \in \mathbb{Z}} z \cdot b_1 + L'$.

(2) Choose $H = zb_2 + L' \otimes \mathbb{R}$ closest to x and $h \in H$ closest to x.

(3) Iteratively, find an approximation y' of $h - zb_2$ in L'.

(4) Output the approximation $y = y' + zb_2$.

Babai's Nearest Plane Procedure (BNPP)



Given a basis $\mathcal{B} = (b_1, \ldots, b_n)$ of L and $x \in \mathbb{R}^n$, BNPP *approximates* x in L.
An approximation factor $2^{n/2}$ is achieved if $\mathcal{B}$ is LLL reduced.

(1) Let $L' = \langle b_1, \ldots, b_{n-1} \rangle_{\mathbb{Z}}$, then $L = \cup_{z \in \mathbb{Z}} z \cdot b_1 + L'$.

(2) Choose $H = zb_2 + L' \otimes \mathbb{R}$ closest to x and $h \in H$ closest to x.

(3) Iteratively, find an approximation y' of $h - zb_2$ in L'.

(4) Output the approximation $y = y' + zb_2$.

BNPP as an iterative decoding algorithm

BNPP as an iterative decoding algorithm

Let $\mathcal{B}' = (b'_1, \dots, b'_n)$ be the Gram Schmidt orthonormalisation of $\mathcal{B}$ and define an isometry $\varphi : b'_i \mapsto e_{n-i+1}$, where $(e_1, \dots, e_n)$ is the standard basis of $\mathbb{R}^n$. Write

BNPP as an iterative decoding algorithm

Let $\mathcal{B}' = (b'_1, \dots, b'_n)$ be the Gram Schmidt orthonormalisation of $\mathcal{B}$ and define an isometry $\varphi : b'_i \mapsto e_{n-i+1}$, where $(e_1, \dots, e_n)$ is the standard basis of $\mathbb{R}^n$. Write

$$\begin{pmatrix} \varphi(b_n) \\ \vdots \\ \varphi(b_1) \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \dots & \alpha_{1,n} \\ & \ddots & \vdots \\ 0 & & \alpha_{n,n} \end{pmatrix}.$$

BNPP as an iterative decoding algorithm

Let $\mathcal{B}' = (b'_1, \ldots, b'_n)$ be the Gram Schmidt orthonormalisation of $\mathcal{B}$ and define an isometry $\varphi : b'_i \mapsto e_{n-i+1}$, where $(e_1, \ldots, e_n)$ is the standard basis of $\mathbb{R}^n$. Write

$$\begin{pmatrix} \varphi(b_n) \\ \vdots \\ \varphi(b_1) \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \cdots & \alpha_{1,n} \\ & \ddots & \vdots \\ 0 & & \alpha_{n,n} \end{pmatrix}.$$

With $\varphi(x) = (u_1, \ldots, u_n)$, BNPP is the following:

BNPP as an iterative decoding algorithm

Let $\mathcal{B}' = (b'_1, \ldots, b'_n)$ be the Gram Schmidt orthonormalisation of $\mathcal{B}$ and define an isometry $\varphi : b'_i \mapsto e_{n-i+1}$, where $(e_1, \ldots, e_n)$ is the standard basis of $\mathbb{R}^n$. Write

$$\begin{pmatrix} \varphi(b_n) \\ \vdots \\ \varphi(b_1) \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \ldots & \alpha_{1,n} \\ & \ddots & \vdots \\ 0 & & \alpha_{n,n} \end{pmatrix}.$$

With $\varphi(x) = (u_1, \ldots, u_n)$, BNPP is the following:

(1) Find the optimal approximation $\ell_1 = z\alpha_{1,1}$ of u_1 in $\mathbb{Z}\alpha_{1,1}$.

BNPP as an iterative decoding algorithm

Let $\mathcal{B}' = (b_1', \ldots, b_n')$ be the Gram Schmidt orthonormalisation of $\mathcal{B}$ and define an isometry $\varphi : b_i' \mapsto e_{n-i+1}$, where $(e_1, \ldots, e_n)$ is the standard basis of $\mathbb{R}^n$. Write

$$\begin{pmatrix} \varphi(b_n) \\ \vdots \\ \varphi(b_1) \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \ldots & \alpha_{1,n} \\ & \ddots & \vdots \\ 0 & & \alpha_{n,n} \end{pmatrix}.$$

With $\varphi(x) = (u_1, \ldots, u_n)$, BNPP is the following:

(1) Find the optimal approximation $\ell_1 = z\alpha_{1,1}$ of u_1 in $\mathbb{Z}\alpha_{1,1}$.

(2) Iteratively, approximate $(u_2 - z\alpha_{1,2}, \ldots, u_n - z\alpha_{1,n}) \in \mathbb{R}^{n-1}$ in $L' = \langle \varphi(b_2), \ldots, \varphi(b_n) \rangle_{\mathbb{Z}}$ with ℓ'.

BNPP as an iterative decoding algorithm

Let $\mathcal{B}' = (b'_1, \ldots, b'_n)$ be the Gram Schmidt orthonormalisation of $\mathcal{B}$ and define an isometry $\varphi : b'_i \mapsto e_{n-i+1}$, where $(e_1, \ldots, e_n)$ is the standard basis of $\mathbb{R}^n$. Write

$$\begin{pmatrix} \varphi(b_n) \\ \vdots \\ \varphi(b_1) \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \ldots & \alpha_{1,n} \\ & \ddots & \vdots \\ 0 & & \alpha_{n,n} \end{pmatrix}.$$

With $\varphi(x) = (u_1, \ldots, u_n)$, BNPP is the following:

(1) Find the optimal approximation $\ell_1 = z\alpha_{1,1}$ of u_1 in $\mathbb{Z}\alpha_{1,1}$.

(2) Iteratively, approximate $(u_2 - z\alpha_{1,2}, \ldots, u_n - z\alpha_{1,n}) \in \mathbb{R}^{n-1}$ in $L' = \langle \varphi(b_2), \ldots, \varphi(b_n) \rangle_{\mathbb{Z}}$ with ℓ'.

(3) Form $\ell = \ell' + z(\alpha_{1,1}, \ldots, \alpha_{1,n})$.

BNPP as an iterative decoding algorithm

Let $\mathcal{B}' = (b'_1, \ldots, b'_n)$ be the Gram Schmidt orthonormalisation of $\mathcal{B}$ and define an isometry $\varphi : b'_i \mapsto e_{n-i+1}$, where $(e_1, \ldots, e_n)$ is the standard basis of $\mathbb{R}^n$. Write

$$\begin{pmatrix} \varphi(b_n) \\ \vdots \\ \varphi(b_1) \end{pmatrix} = \begin{pmatrix} \alpha_{1,1} & \ldots & \alpha_{1,n} \\ & \ddots & \vdots \\ 0 & & \alpha_{n,n} \end{pmatrix}.$$

With $\varphi(x) = (u_1, \ldots, u_n)$, BNPP is the following:

(1) Find the optimal approximation $\ell_1 = z\alpha_{1,1}$ of u_1 in $\mathbb{Z}\alpha_{1,1}$.

(2) Iteratively, approximate $(u_2 - z\alpha_{1,2}, \ldots, u_n - z\alpha_{1,n}) \in \mathbb{R}^{n-1}$ in $L' = \langle \varphi(b_2), \ldots, \varphi(b_n) \rangle_{\mathbb{Z}}$ with ℓ'.

(3) Form $\ell = \ell' + z(\alpha_{1,1}, \ldots, \alpha_{1,n})$.

Idea: Generalise BNPP, changing from lattices $\alpha\mathbb{Z}$ to higher dimensional lattices.

Iterative lattice decoding

- Let W_i be lattices of dimension n_i, $i \in \{1, \ldots, t\}$, and let $f_i : \mathbb{R}^{n_1+\cdots+n_i} \to \mathbb{R}^{n_{i+1}}$ linear maps, for $i \in \{1, \ldots, t-1\}$.

Iterative lattice decoding

- Let W_i be lattices of dimension n_i, $i \in \{1, \ldots, t\}$, and let $f_i : \mathbb{R}^{n_1+\cdots+n_i} \to \mathbb{R}^{n_{i+1}}$ linear maps, for $i \in \{1, \ldots, t-1\}$.
- Form a lattice $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_t)$ of dimension $n = n_1 + \cdots + n_t$ by

$$\mathcal{L} = \{(\ell_1, \ldots, \ell_t) \in \mathbb{R}^n \mid \ell_1 \in W_1, \ell_i - f_{i-1}(\ell_1, \ldots, \ell_{i-1}) \in W_i,\ i \in \{1, \ldots, t\}\}.$$

Iterative lattice decoding

- Let W_i be lattices of dimension n_i, $i \in \{1, \dots, t\}$, and let $f_i : \mathbb{R}^{n_1+\cdots+n_i} \to \mathbb{R}^{n_{i+1}}$ linear maps, for $i \in \{1, \dots, t-1\}$.
- Form a lattice $\mathcal{L} = \mathcal{L}(W_1, \dots, W_t, f_2, \dots, f_t)$ of dimension $n = n_1 + \cdots + n_t$ by

$$\mathcal{L} = \{(\ell_1, \dots, \ell_t) \in \mathbb{R}^n \mid \ell_1 \in W_1, \ell_i - f_{i-1}(\ell_1, \dots, \ell_{i-1}) \in W_i,\ i \in \{1, \dots, t\}\}.$$

- Decoding algorithm $\mathcal{A}$ for $\mathcal{L}$: Let $x = (x_1, \dots, x_t) \in \mathbb{R}^n$.

Iterative lattice decoding

- Let W_i be lattices of dimension n_i, $i \in \{1, \ldots, t\}$, and let $f_i : \mathbb{R}^{n_1+\cdots+n_i} \to \mathbb{R}^{n_{i+1}}$ linear maps, for $i \in \{1, \ldots, t-1\}$.
- Form a lattice $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_t)$ of dimension $n = n_1 + \cdots + n_t$ by

$$\mathcal{L} = \{(\ell_1, \ldots, \ell_t) \in \mathbb{R}^n \mid \ell_1 \in W_1, \ell_i - f_{i-1}(\ell_1, \ldots, \ell_{i-1}) \in W_i,\ i \in \{1, \ldots, t\}\}.$$

- Decoding algorithm $\mathcal{A}$ for $\mathcal{L}$: Let $x = (x_1, \ldots, x_t) \in \mathbb{R}^n$.
  (1) Let $\ell_1 \in W_1$ be the lattice point closest to x_1.

Iterative lattice decoding

- Let W_i be lattices of dimension n_i, $i \in \{1, \ldots, t\}$, and let $f_i : \mathbb{R}^{n_1+\cdots+n_i} \to \mathbb{R}^{n_{i+1}}$ linear maps, for $i \in \{1, \ldots, t-1\}$.
- Form a lattice $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_t)$ of dimension $n = n_1 + \cdots + n_t$ by

$$\mathcal{L} = \{(\ell_1, \ldots, \ell_t) \in \mathbb{R}^n \mid \ell_1 \in W_1, \ell_i - f_{i-1}(\ell_1, \ldots, \ell_{i-1}) \in W_i,\ i \in \{1, \ldots, t\})\}.$$

- Decoding algorithm $\mathcal{A}$ for $\mathcal{L}$: Let $x = (x_1, \ldots, x_t) \in \mathbb{R}^n$.
  (1) Let $\ell_1 \in W_1$ be the lattice point closest to x_1.
  (2) For $2 \leq i \leq t$, approximate $x_i - f_{i-1}(\ell_1, \ldots, \ell_{i-1})$ by $w_i \in W_i$ and put $\ell_i := w_i + f_{i-1}(\ell_1, \ldots, \ell_{i-1})$.

Iterative lattice decoding

- Let W_i be lattices of dimension n_i, $i \in \{1, \ldots, t\}$, and let $f_i : \mathbb{R}^{n_1+\cdots+n_i} \to \mathbb{R}^{n_{i+1}}$ linear maps, for $i \in \{1, \ldots, t-1\}$.
- Form a lattice $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_t)$ of dimension $n = n_1 + \cdots + n_t$ by

$$\mathcal{L} = \{(\ell_1, \ldots, \ell_t) \in \mathbb{R}^n \mid \ell_1 \in W_1, \ell_i - f_{i-1}(\ell_1, \ldots, \ell_{i-1}) \in W_i,\ i \in \{1, \ldots, t\}\}.$$

- Decoding algorithm $\mathcal{A}$ for $\mathcal{L}$: Let $x = (x_1, \ldots, x_t) \in \mathbb{R}^n$.
  (1) Let $\ell_1 \in W_1$ be the lattice point closest to x_1.
  (2) For $2 \leq i \leq t$, approximate $x_i - f_{i-1}(\ell_1, \ldots, \ell_{i-1})$ by $w_i \in W_i$ and put $\ell_i := w_i + f_{i-1}(\ell_1, \ldots, \ell_{i-1})$.
  (3) Output the approximation $\ell = (\ell_1, \ldots, \ell_t) \in \mathcal{L}$.

Algorithm $\mathcal{A}'$ - some questions and remarks

- Algorithm $\mathcal{A}'$ depends on the chosen lattice basis.

Algorithm $\mathcal{A}'$ - some questions and remarks

- Algorithm $\mathcal{A}'$ depends on the chosen lattice basis.
- For every isometry class of lattices, there are many ways to give an upper triangular (block) basis matrix for a representative.

Algorithm $\mathcal{A}'$ - some questions and remarks

- Algorithm $\mathcal{A}'$ depends on the chosen lattice basis.
- For every isometry class of lattices, there are many ways to give an upper triangular (block) basis matrix for a representative.
  - There should exist good decoding algorithms for $W_1, \ldots, W_t$ (specific decoding algorithms exist for many well known lattices).

Algorithm $\mathcal{A}'$ - some questions and remarks

- Algorithm $\mathcal{A}'$ depends on the chosen lattice basis.
- For every isometry class of lattices, there are many ways to give an upper triangular (block) basis matrix for a representative.
  - There should exist good decoding algorithms for $W_1, \ldots, W_t$ (specific decoding algorithms exist for many well known lattices).
  - When do we obtain a good approximation?

Algorithm $\mathcal{A}'$ - some questions and remarks

- Algorithm $\mathcal{A}'$ depends on the chosen lattice basis.
- For every isometry class of lattices, there are many ways to give an upper triangular (block) basis matrix for a representative.
  - There should exist good decoding algorithms for $W_1, \ldots, W_t$ (specific decoding algorithms exist for many well known lattices).
  - When do we obtain a good approximation?
- What do we gain when if we consider *all* the elements of

$$B_r(x_1) \cap W_1 = \{w \in W_1 \mid |w - x_1| \leq r\}$$

in the first step of Algorithm $\mathcal{A}'$?

Algorithm $\mathcal{A}'$ - some questions and remarks

- Algorithm $\mathcal{A}'$ depends on the chosen lattice basis.
- For every isometry class of lattices, there are many ways to give an upper triangular (block) basis matrix for a representative.
  - There should exist good decoding algorithms for $W_1, \ldots, W_t$ (specific decoding algorithms exist for many well known lattices).
  - When do we obtain a good approximation?
- What do we gain when if we consider *all* the elements of

$$B_r(x_1) \cap W_1 = \{w \in W_1 \mid |w - x_1| \leq r\}$$

  in the first step of Algorithm $\mathcal{A}'$?
- *Sphere decoding* (Fincke, Pohst) can be used to compute $B_r(x_1) \cap W_1$.

Approximation factors for Algorithm $\mathcal{A}'$

Definition

- The *packing radius* of a lattice L in $\mathbb{R}^n$ is $\rho_L := \frac{1}{2}\sqrt{\min(L)}$, where $\min(L) := \min_{0 \neq \ell \in L} |\ell|^2$.
- The *covering radius* of L is $\gamma_L := \sqrt{\mu(L)}$, where $\mu(L) = \max_{v \in \mathbb{R}^n} \min_{\ell \in L} |v - \ell|^2$.

Approximation factors for Algorithm $\mathcal{A}'$

Definition

- The *packing radius* of a lattice L in $\mathbb{R}^n$ is $\rho_L := \frac{1}{2}\sqrt{\min(L)}$, where $\min(L) := \min_{0\neq\ell\in L} |\ell|^2$.
- The *covering radius* of L is $\gamma_L := \sqrt{\mu(L)}$, where $\mu(L) = \max_{v\in\mathbb{R}^n} \min_{\ell\in L} |v-\ell|^2$.

Theorem

Algorithm $\mathcal{A}'$ achieves an approximation factor $\sqrt{\delta_t}$, definded recursively by

$$\delta_1 = 4\frac{\mu(W_t)}{\min(W_t)}, \quad \delta_j = \max\{4\frac{\sum_{i=t-j+1}^{t} \mu(W_i)}{\min(W_{t-j+1})}, \delta_{j-1}+1\}, \quad j = 2,\dots,t.$$

Approximation factors for Algorithm $\mathcal{A}'$

Definition

- The *packing radius* of a lattice L in $\mathbb{R}^n$ is $\rho_L := \frac{1}{2}\sqrt{\min(L)}$, where $\min(L) := \min_{0\neq\ell\in L} |\ell|^2$.
- The *covering radius* of L is $\gamma_L := \sqrt{\mu(L)}$, where $\mu(L) = \max_{v\in\mathbb{R}^n} \min_{\ell\in L} |v-\ell|^2$.

Theorem

Algorithm $\mathcal{A}'$ achieves an approximation factor $\sqrt{\delta_t}$, definded recursively by

$$\delta_1 = 4\frac{\mu(W_t)}{\min(W_t)}, \quad \delta_j = \max\{4\frac{\sum_{i=t-j+1}^{t} \mu(W_i)}{\min(W_{t-j+1})}, \delta_{j-1}+1\}, \quad j = 2, \dots, t.$$

A modification of Algorithm $\mathcal{A}'$

Algorithm $\mathcal{A}$:

A modification of Algorithm $\mathcal{A}'$

Algorithm $\mathcal{A}$:
Let $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_{t-1})$, $x = (x_1, \ldots, x_t) \in \mathbb{R}^n$ and $r > 0$.

A modification of Algorithm $\mathcal{A}'$

Algorithm $\mathcal{A}$:
Let $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_{t-1})$, $x = (x_1, \ldots, x_t) \in \mathbb{R}^n$ and $r > 0$.

- Use sphere decoding to find all the points in $W_1 \cap B_r(x_1)$.

A modification of Algorithm $\mathcal{A}'$

Algorithm $\mathcal{A}$:
Let $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_{t-1})$, $x = (x_1, \ldots, x_t) \in \mathbb{R}^n$ and $r > 0$.

- Use sphere decoding to find all the points in $W_1 \cap B_r(x_1)$.
- For every point found in step 1, perform steps 2 and 3 of Algorithm $\mathcal{A}'$.

A modification of Algorithm $\mathcal{A}'$

Algorithm $\mathcal{A}$:
Let $\mathcal{L} = \mathcal{L}(W_1, \dots, W_t, f_2, \dots, f_{t-1})$, $x = (x_1, \dots, x_t) \in \mathbb{R}^n$ and $r > 0$.

- Use sphere decoding to find all the points in $W_1 \cap B_r(x_1)$.
- For every point found in step 1, perform steps 2 and 3 of Algorithm $\mathcal{A}'$.
- Among all the approximations found, choose the best one.

A modification of Algorithm $\mathcal{A}'$

Algorithm $\mathcal{A}$:
Let $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_{t-1})$, $x = (x_1, \ldots, x_t) \in \mathbb{R}^n$ and $r > 0$.

- Use sphere decoding to find all the points in $W_1 \cap B_r(x_1)$.
- For every point found in step 1, perform steps 2 and 3 of Algorithm $\mathcal{A}'$.
- Among all the approximations found, choose the best one.

Theorem

With $\delta_1, \ldots, \delta_{t-1}$ as above, Algorithm $\mathcal{A}$ achieves an approximation factor of

$$\max\{1 + \delta_{t-1}, r^{-1} \sum_{i=1}^{t} \mu(W_i)\}^{\frac{1}{2}}.$$

A modification of Algorithm $\mathcal{A}'$

Algorithm $\mathcal{A}$:

Let $\mathcal{L} = \mathcal{L}(W_1, \ldots, W_t, f_2, \ldots, f_{t-1})$, $x = (x_1, \ldots, x_t) \in \mathbb{R}^n$ and $r > 0$.

- Use sphere decoding to find all the points in $W_1 \cap B_r(x_1)$.
- For every point found in step 1, perform steps 2 and 3 of Algorithm $\mathcal{A}'$.
- Among all the approximations found, choose the best one.

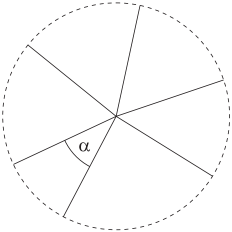
Theorem

With $\delta_1, \ldots, \delta_{t-1}$ as above, Algorithm $\mathcal{A}$ achieves an approximation factor of

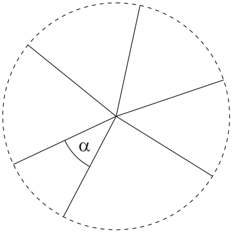
$$\max\{1 + \delta_{t-1}, r^{-1} \sum_{i=1}^{t} \mu(W_i)\}^{\frac{1}{2}}.$$

Question: Can we upper bound $|B_r(x_1) \cap W_1|$?

Bounds on $|B_r(x) \cap W|$ via spherical codes



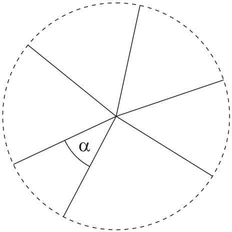
Bounds on $|B_r(x) \cap W|$ via spherical codes



Definition

A *spherical code* in $\mathbb{R}^s$ is a set $\mathcal{C}$ of vectors of length 1. The minimum angle of $\mathcal{C}$ is $\alpha_{\min}(\mathcal{C}) := \min_{c \neq c' \in \mathcal{C}} \angle(c, c')$.

Bounds on $|B_r(x) \cap W|$ via spherical codes

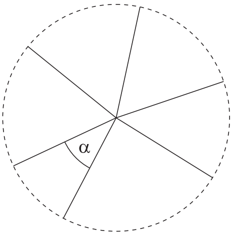


Definition

A *spherical code* in $\mathbb{R}^s$ is a set $\mathcal{C}$ of vectors of length 1. The minimum angle of $\mathcal{C}$ is $\alpha_{\min}(\mathcal{C}) := \min_{c \neq c' \in \mathcal{C}} \angle(c, c')$.

- If $\mathcal{C}$ is a spherical code at a positive minimum angle, then $|\mathcal{C}| < \infty$.

Bounds on $|B_r(x) \cap W|$ via spherical codes

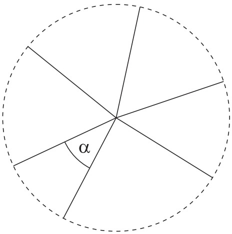


Definition

A *spherical code* in $\mathbb{R}^s$ is a set $\mathcal{C}$ of vectors of length 1. The minimum angle of $\mathcal{C}$ is $\alpha_{\min}(\mathcal{C}) := \min_{c \neq c' \in \mathcal{C}} \angle(c, c')$.

- If $\mathcal{C}$ is a spherical code at a positive minimum angle, then $|\mathcal{C}| < \infty$.
- In this case, good upper bounds on $|\mathcal{C}|$ can be derived using *linear programs*, whose variables are the coefficients of the *weight distribution* of $\mathcal{C}$ (Kabatiansky, Levenshtein).

Bounds on $|B_r(x) \cap W|$ via spherical codes



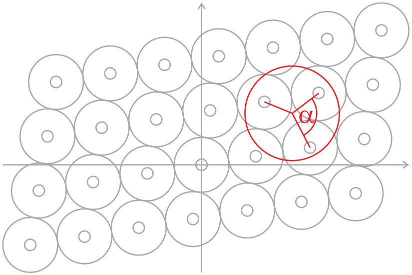
Definition

A *spherical code* in $\mathbb{R}^s$ is a set $\mathcal{C}$ of vectors of length 1. The minimum angle of $\mathcal{C}$ is $\alpha_{\min}(\mathcal{C}) := \min_{c \neq c' \in \mathcal{C}} \angle(c, c')$.

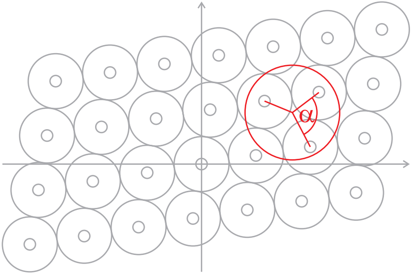
- If $\mathcal{C}$ is a spherical code at a positive minimum angle, then $|\mathcal{C}| < \infty$.
- In this case, good upper bounds on $|\mathcal{C}|$ can be derived using *linear programs*, whose variables are the coefficients of the *weight distribution* of $\mathcal{C}$ (Kabatiansky, Levenshtein).

Lattices and spherical codes

Lattices and spherical codes



Lattices and spherical codes



Theorem

Let L be a lattice in $\mathbb{R}^s$. If r is a real number with $0 < r \leq 2\rho_L$ then the set

$$\{|x - z|^{-1}(x - z) \mid z \in B_r(x) \cap L\}$$

is a spherical code with minimum angle $\alpha = \cos^{-1}(1 - \frac{\rho_L}{r})$, for every $x \in \mathbb{R}^s$.

Examples: Bounds obtained for A_n, E_n, Λ_{24}, $r = \gamma_L$

Type	n	θ	$A(n,\theta)$	Gaussian bound	for deep holes
A	2	$\frac{2}{3}\pi$	3	3	3
	3	$\frac{\pi}{2}$	6	7	6
	4	$\cos^{-1}(\frac{1}{6})$	10	12	10
	5	$\cos^{-1}(\frac{1}{3})$	≤ 24	26	20
	6	$\cos^{-1}(\frac{5}{12})$	≤ 54	47	35
	7	$\frac{\pi}{3}$	≤ 140	99	70
	8	-	-	188	126
	9	-	-	391	252
E	6	$\cos^{-1}(\frac{1}{4})$	27	37	27
	7	$\cos^{-1}(\frac{1}{3})$	56	84	56
	8	$\frac{\pi}{2}$	16	77	16
Leech	24	$\frac{\pi}{2}$	48	974	48

Example: Nebe's extremal even unimodular lattice Λ_{72}

- Λ_{72} is obtained from a polarisation $(\alpha(\Lambda_{24}), \beta(\Lambda_{24}))$ of the Leech lattice Λ_{24}, where $\alpha, \beta \in \mathrm{End}(\Lambda_{24})$ such that $\alpha^2 - \alpha + 2 = 0$, $\beta = 1 - \alpha$ and $(\alpha(x), y) = (x, \beta(y))$ for all $x, y \in \mathbb{R}^{24}$:

Example: Nebe's extremal even unimodular lattice Λ_{72}

- Λ_{72} is obtained from a polarisation $(\alpha(\Lambda_{24}), \beta(\Lambda_{24}))$ of the Leech lattice Λ_{24}, where $\alpha, \beta \in \mathrm{End}(\Lambda_{24})$ such that $\alpha^2 - \alpha + 2 = 0$, $\beta = 1 - \alpha$ and $(\alpha(x), y) = (x, \beta(y))$ for all $x, y \in \mathbb{R}^{24}$:

$$\Lambda_{72} \quad = \quad \{(\ell_1, \ell_2, \ell_3) \in \perp_{i=1}^{3} \Lambda_{24} \mid \ell_1 - \ell_2 \in \beta(\Lambda_{24}),\ \ell_2 - \beta(\ell_1) - \ell_3 \in 2\Lambda_{24})\}$$

Example: Nebe's extremal even unimodular lattice Λ_{72}

- Λ_{72} is obtained from a polarisation $(\alpha(\Lambda_{24}), \beta(\Lambda_{24}))$ of the Leech lattice Λ_{24}, where $\alpha, \beta \in \mathrm{End}(\Lambda_{24})$ such that $\alpha^2 - \alpha + 2 = 0$, $\beta = 1 - \alpha$ and $(\alpha(x), y) = (x, \beta(y))$ for all $x, y \in \mathbb{R}^{24}$:

$$\Lambda_{72} = \{(\ell_1, \ell_2, \ell_3) \in \perp_{i=1}^{3} \Lambda_{24} \mid \ell_1 - \ell_2 \in \beta(\Lambda_{24}),\ \ell_2 - \beta(\ell_1) - \ell_3 \in 2\Lambda_{24})\}$$

- Algorithm $\mathcal{A}'$: Decoding $(x_1, x_2, x_3) \in \perp_{i=1}^{3} \mathbb{R}^{24}$ in Λ_{72} with approximation factor $\sqrt{14}$:

Example: Nebe's extremal even unimodular lattice Λ_{72}

- Λ_{72} is obtained from a polarisation $(\alpha(\Lambda_{24}), \beta(\Lambda_{24}))$ of the Leech lattice Λ_{24}, where $\alpha, \beta \in \mathrm{End}(\Lambda_{24})$ such that $\alpha^2 - \alpha + 2 = 0$, $\beta = 1 - \alpha$ and $(\alpha(x), y) = (x, \beta(y))$ for all $x, y \in \mathbb{R}^{24}$:

$$\Lambda_{72} = \{(\ell_1, \ell_2, \ell_3) \in \perp_{i=1}^{3} \Lambda_{24} \mid \ell_1 - \ell_2 \in \beta(\Lambda_{24}),\ \ell_2 - \beta(\ell_1) - \ell_3 \in 2\Lambda_{24})\}$$

- Algorithm $\mathcal{A}'$: Decoding $(x_1, x_2, x_3) \in \perp_{i=1}^{3} \mathbb{R}^{24}$ in Λ_{72} with approximation factor $\sqrt{14}$:
  (1) Approximate x_1 with $y_1 \in \Lambda_{24}$.

Example: Nebe's extremal even unimodular lattice Λ_{72}

- Λ_{72} is obtained from a polarisation $(\alpha(\Lambda_{24}), \beta(\Lambda_{24}))$ of the Leech lattice Λ_{24}, where $\alpha, \beta \in \mathrm{End}(\Lambda_{24})$ such that $\alpha^2 - \alpha + 2 = 0$, $\beta = 1 - \alpha$ and $(\alpha(x), y) = (x, \beta(y))$ for all $x, y \in \mathbb{R}^{24}$:

$$\Lambda_{72} = \{(\ell_1, \ell_2, \ell_3) \in \perp_{i=1}^{3} \Lambda_{24} \mid \ell_1 - \ell_2 \in \beta(\Lambda_{24}),\ \ell_2 - \beta(\ell_1) - \ell_3 \in 2\Lambda_{24})\}$$

- Algorithm $\mathcal{A}'$: Decoding $(x_1, x_2, x_3) \in \perp_{i=1}^{3} \mathbb{R}^{24}$ in Λ_{72} with approximation factor $\sqrt{14}$:
  (1) Approximate x_1 with $y_1 \in \Lambda_{24}$.
  (2) Approximate $x_2 - y_1$ with $y_2 \in \beta(\Lambda_{24})$.

Example: Nebe's extremal even unimodular lattice Λ_{72}

- Λ_{72} is obtained from a polarisation $(\alpha(\Lambda_{24}), \beta(\Lambda_{24}))$ of the Leech lattice Λ_{24}, where $\alpha, \beta \in \mathrm{End}(\Lambda_{24})$ such that $\alpha^2 - \alpha + 2 = 0$, $\beta = 1 - \alpha$ and $(\alpha(x), y) = (x, \beta(y))$ for all $x, y \in \mathbb{R}^{24}$:

$$\Lambda_{72} = \{(\ell_1, \ell_2, \ell_3) \in \perp_{i=1}^{3} \Lambda_{24} \mid \ell_1 - \ell_2 \in \beta(\Lambda_{24}),\ \ell_2 - \beta(\ell_1) - \ell_3 \in 2\Lambda_{24})\}$$

- Algorithm $\mathcal{A}'$: Decoding $(x_1, x_2, x_3) \in \perp_{i=1}^{3} \mathbb{R}^{24}$ in Λ_{72} with approximation factor $\sqrt{14}$:
  (1) Approximate x_1 with $y_1 \in \Lambda_{24}$.
  (2) Approximate $x_2 - y_1$ with $y_2 \in \beta(\Lambda_{24})$.
  (3) Approximate $x_3 - \alpha(y_1) - y_2$ with $y_3 \in 2\Lambda_{24}$.

Example: Nebe's extremal even unimodular lattice Λ_{72}

- Λ_{72} is obtained from a polarisation $(\alpha(\Lambda_{24}), \beta(\Lambda_{24}))$ of the Leech lattice Λ_{24}, where $\alpha, \beta \in \mathrm{End}(\Lambda_{24})$ such that $\alpha^2 - \alpha + 2 = 0$, $\beta = 1 - \alpha$ and $(\alpha(x), y) = (x, \beta(y))$ for all $x, y \in \mathbb{R}^{24}$:

$$\Lambda_{72} \quad = \quad \{(\ell_1, \ell_2, \ell_3) \in \perp_{i=1}^{3} \Lambda_{24} \mid \ell_1 - \ell_2 \in \beta(\Lambda_{24}),\ \ell_2 - \beta(\ell_1) - \ell_3 \in 2\Lambda_{24})\}$$

- Algorithm $\mathcal{A}'$: Decoding $(x_1, x_2, x_3) \in \perp_{i=1}^{3} \mathbb{R}^{24}$ in Λ_{72} with approximation factor $\sqrt{14}$:
  (1) Approximate x_1 with $y_1 \in \Lambda_{24}$.
  (2) Approximate $x_2 - y_1$ with $y_2 \in \beta(\Lambda_{24})$.
  (3) Approximate $x_3 - \alpha(y_1) - y_2$ with $y_3 \in 2\Lambda_{24}$.
  (4) Output the approximation $(\ell_1, \ell_2, \ell_3) = (y_1, y_1 + y_2, \alpha(y_1) + y_2 + y_3)$

Example: Nebe's extremal even unimodular lattice Λ_{72}

- Λ_{72} is obtained from a polarisation $(\alpha(\Lambda_{24}), \beta(\Lambda_{24}))$ of the Leech lattice Λ_{24}, where $\alpha, \beta \in \mathrm{End}(\Lambda_{24})$ such that $\alpha^2 - \alpha + 2 = 0$, $\beta = 1 - \alpha$ and $(\alpha(x), y) = (x, \beta(y))$ for all $x, y \in \mathbb{R}^{24}$:

$$\Lambda_{72} \quad = \quad \{(\ell_1, \ell_2, \ell_3) \in \perp_{i=1}^{3} \Lambda_{24} \mid \ell_1 - \ell_2 \in \beta(\Lambda_{24}),\ \ell_2 - \beta(\ell_1) - \ell_3 \in 2\Lambda_{24})\}$$

- Algorithm $\mathcal{A}'$: Decoding $(x_1, x_2, x_3) \in \perp_{i=1}^{3} \mathbb{R}^{24}$ in Λ_{72} with approximation factor $\sqrt{14}$:
  (1) Approximate x_1 with $y_1 \in \Lambda_{24}$.
  (2) Approximate $x_2 - y_1$ with $y_2 \in \beta(\Lambda_{24})$.
  (3) Approximate $x_3 - \alpha(y_1) - y_2$ with $y_3 \in 2\Lambda_{24}$.
  (4) Output the approximation $(\ell_1, \ell_2, \ell_3) = (y_1, y_1 + y_2, \alpha(y_1) + y_2 + y_3)$
- Algorithm $\mathcal{A}$: time increased by at most $|B_{\sqrt{2}}(x_1) \cap \Lambda_{24}| \leq 48$, approximation factor of $\sqrt{7}$, using sphere decoding with $r = \sqrt{2} = \sqrt{\mu(\Lambda_{24})}$.

Thank you very much for your attention!