Lattice Reduction

The Algorithmic Ant and the Sandpile

Léo Ducas

Cryptology Group, CWI, Amsterdam, The Netherlands

Mathematics of Public Key Cryptography
Aussois, FR, March 2019

The Algorithmic Ant and the Sandpile

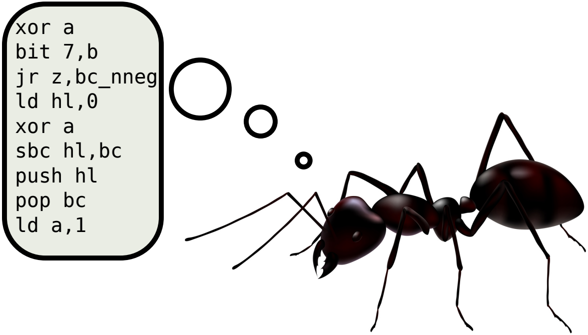
Once upon a time ...

Once upon a time . . .

. . . there was an ant.

Once upon a time . . .

. . . there was an ant.

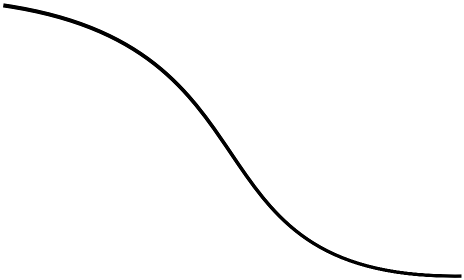


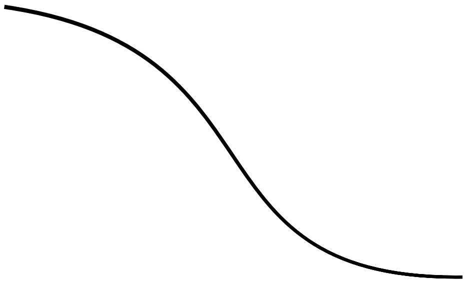
An algorithmic ant.

The Queen of ant ant,

The Queen of ant
ant,

"See this sand pile."





"I want it *flat* !"

Looking clo

the algorithmic ant ponders.

"One grain at the time,
I shall pull the sand downhill."

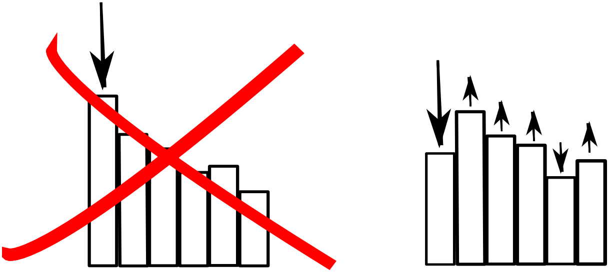
"One grain at the time,
I shall pull the sand downhill."

So ho

But the sandpile is whimsical,

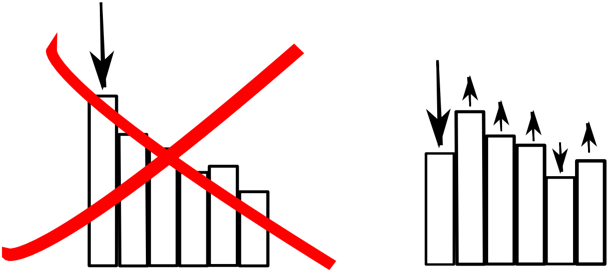
each excavation is a puzzle of it

But the sandpile is whimsical,
each excavation is a puzzle of it



But the sandpile is whimsical,

each excavation is a puzzle of it

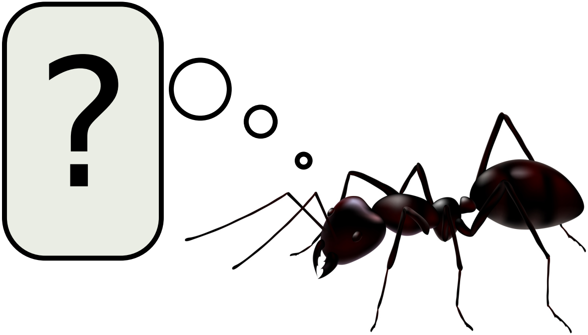


Columns are tied in my

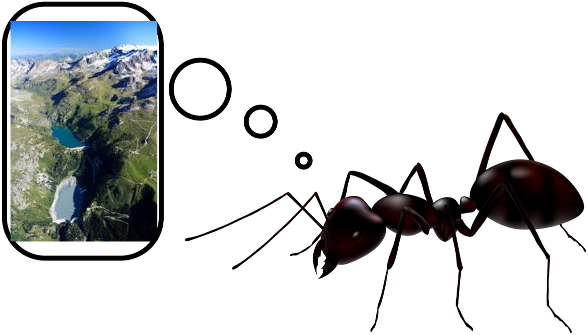
to push one down, one must find

the right combination.

Unsure how to proceed,



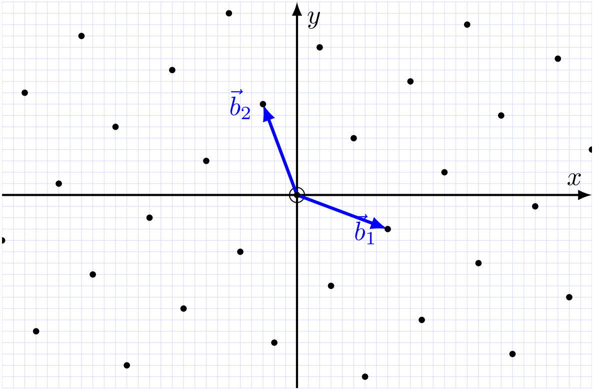
Unsure how to proceed,



The ant joins Aussois Winter school to learn about lattice reduction.

Lattices and Bases

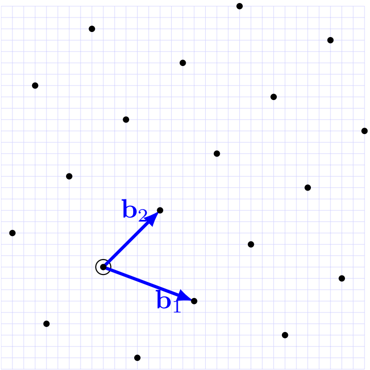
Lattices!



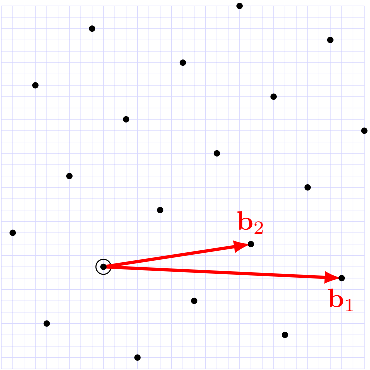
Definition

A lattice L is a discrete subgroup of a finite-dimensional Euclidean vector space.

The Good-basis/Bad-basis Routine



Good Basis $\mathbf{G}$ of L



Bad Basis $\mathbf{B}$ of L

G $\rightarrow$ **B** : easy (randomization);

B $\rightarrow$ **G** : hard (LLL, BKZ, Lattice Sieve...).

Lattice reduction: a pedantic summary

- A basis: $\mathbf{B} \in GL_n(\mathbb{R})$
- A lattice change of basis: $\mathbf{Z} \in GL_n(\mathbb{Z})$
- A lattice: $\Lambda \in GL_n(\mathbb{R})/GL_n(\mathbb{Z})$
- Lattice reduction: finding a good representative $\mathbf{M} \in GL_n(\mathbb{R})$ of the class of $\Lambda \in GL_n(\mathbb{R})/GL_n(\mathbb{Z})$

Remarks:

1. We will not achieve canonical representation.
2. We will typically want invariance by rotations $\mathbf{Q} \in O_n(\mathbb{R})$:

$$\text{if } \mathbf{B} \rightsquigarrow \mathbf{G}, \text{ then } \mathbf{QB} \rightsquigarrow \mathbf{QG}$$

Lattice reduction: a pedantic summary

- A basis: $\mathbf{B} \in GL_n(\mathbb{R})$
- A lattice change of basis: $\mathbf{Z} \in GL_n(\mathbb{Z})$
- A lattice: $\Lambda \in GL_n(\mathbb{R})/GL_n(\mathbb{Z})$
- Lattice reduction: finding a good representative $\mathbf{M} \in GL_n(\mathbb{R})$ of the class of $\Lambda \in GL_n(\mathbb{R})/GL_n(\mathbb{Z})$

Remarks:

1. We will not achieve canonical representation.
2. We will typically want invariance by rotations $\mathbf{Q} \in O_n(\mathbb{R})$:

$$\text{if } \mathbf{B} \rightsquigarrow \mathbf{G}, \text{ then } \mathbf{QB} \rightsquigarrow \mathbf{QG}$$

Lattice reduction: a pedantic summary

- A basis: $\mathbf{B} \in GL_n(\mathbb{R})$
- A lattice change of basis: $\mathbf{Z} \in GL_n(\mathbb{Z})$
- A lattice: $\Lambda \in GL_n(\mathbb{R})/GL_n(\mathbb{Z})$
- Lattice reduction: finding a good representative $\mathbf{M} \in GL_n(\mathbb{R})$ of the class of $\Lambda \in GL_n(\mathbb{R})/GL_n(\mathbb{Z})$

Remarks:

1. We will not achieve canonical representation.
2. We will typically want invariance by rotations $\mathbf{Q} \in O_n(\mathbb{R})$:

$$\text{if } \mathbf{B} \rightsquigarrow \mathbf{G}, \text{ then } \mathbf{QB} \rightsquigarrow \mathbf{QG}$$

An important invariant: the Volume

For any two bases **G**, **B** of the same lattice Λ:

$$\det(\mathbf{G}\mathbf{G}^t) = \det(\mathbf{B}\mathbf{B}^t).$$

We can therefore define:

$$\mathrm{vol}(\Lambda) = \sqrt{\det(\mathbf{G}\mathbf{G}^t)}.$$

Geometrically: the volume of any **fundamental domain of** Λ.

Let $\mathbf{G}^\star$ be the Gram-Schmidt Orthogonalization of $\mathbf{G}$

$\mathbf{G}^\star$ is **not** a basis of Λ, nevertheless:

$$\mathrm{vol}(\Lambda) = \sqrt{\det(\mathbf{G}^\star \mathbf{G}^{\star t})} = \prod \|\mathbf{g}_i^\star\|.$$

An important invariant: the Volume

For any two bases $\mathbf{G}, \mathbf{B}$ of the same lattice Λ:

$$\det(\mathbf{G}\mathbf{G}^t) = \det(\mathbf{B}\mathbf{B}^t).$$

We can therefore define:

$$\mathrm{vol}(\Lambda) = \sqrt{\det(\mathbf{G}\mathbf{G}^t)}.$$

Geometrically: the volume of any **fundamental domain of** Λ.

Let $\mathbf{G}^\star$ be the Gram-Schmidt Orthogonalization of $\mathbf{G}$

$\mathbf{G}^\star$ is **not** a basis of Λ, nevertheless:

$$\mathrm{vol}(\Lambda) = \sqrt{\det(\mathbf{G}^\star \mathbf{G}^{\star t})} = \prod \|\mathbf{g}_i^\star\|.$$

What is a "Good" basis

Recall that, independently of the basis $\mathbf{G}$ it holds that:

$$\mathrm{vol}(\Lambda) = \prod \|\mathbf{g}_i^\star\|.$$

Therefore, it is somehow equivalent that

- $\max_i \|\mathbf{g}_i^\star\|$ is small
- $\min_i \|\mathbf{g}_i^\star\|$ is large
- $\kappa(\mathbf{G}) = \max_i \|\mathbf{g}_i^\star\| / \min_i \|\mathbf{g}_i^\star\|$ is small

Good basis

$$\max \|\mathbf{b}_i^*\| \approx \min \|\mathbf{b}_i^*\|, \qquad (\text{equivalently}: \forall i, \|\mathbf{b}_i^*\| \approx \mathrm{vol}(\Lambda)^{1/n})$$

Bad basis

$$\max \|\mathbf{b}_i^*\| \gg \min \|\mathbf{b}_i^*\|$$

What is a "Good" basis

Recall that, independently of the basis $\mathbf{G}$ it holds that:

$$\mathrm{vol}(\Lambda) = \prod \|\mathbf{g}_i^\star\|.$$

Therefore, it is somehow equivalent that

- $\max_i \|\mathbf{g}_i^\star\|$ is small
- $\min_i \|\mathbf{g}_i^\star\|$ is large
- $\kappa(\mathbf{G}) = \max_i \|\mathbf{g}_i^\star\| / \min_i \|\mathbf{g}_i^\star\|$ is small

Good basis

$$\max \|\mathbf{b}_i^*\| \approx \min \|\mathbf{b}_i^*\|, \qquad (\text{equivalently}: \forall i, \|\mathbf{b}_i^*\| \approx \mathrm{vol}(\Lambda)^{1/n})$$

Bad basis

$$\max \|\mathbf{b}_i^*\| \gg \min \|\mathbf{b}_i^*\|$$

What is a "Good" basis

Recall that, independently of the basis $\mathbf{G}$ it holds that:

$$\mathrm{vol}(\Lambda) = \prod \|\mathbf{g}_i^\star\|.$$

Therefore, it is somehow equivalent that

- $\max_i \|\mathbf{g}_i^\star\|$ is small
- $\min_i \|\mathbf{g}_i^\star\|$ is large
- $\kappa(\mathbf{G}) = \max_i \|\mathbf{g}_i^\star\| / \min_i \|\mathbf{g}_i^\star\|$ is small

Good basis

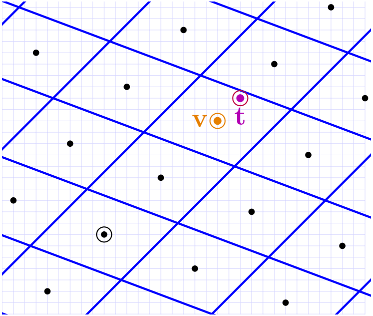
$$\max \|\mathbf{b}_i^*\| \approx \min \|\mathbf{b}_i^*\|, \qquad (\text{equivalently} : \forall i, \|\mathbf{b}_i^*\| \approx \mathrm{vol}(\Lambda)^{1/n})$$

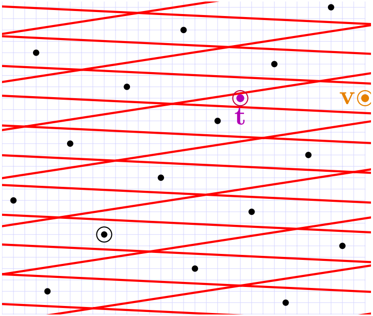
Bad basis

$$\max \|\mathbf{b}_i^*\| \gg \min \|\mathbf{b}_i^*\|$$

Bases and Fundamental Domains

Each basis defines a **parallelepipedic tiling**.

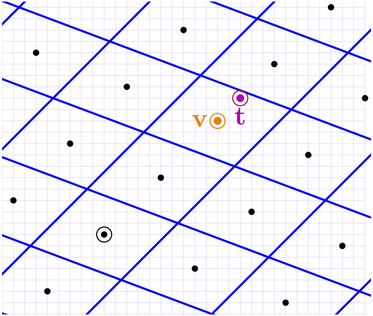


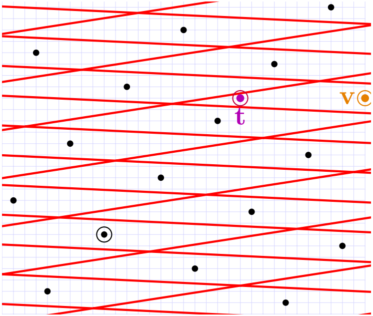


Round'off Algorithm [Lenstra, Babai]:

- Given a target **t**
- Find's **v** $\in L$ at the center the tile.

Each basis defines a **parallelepipedic tiling**.



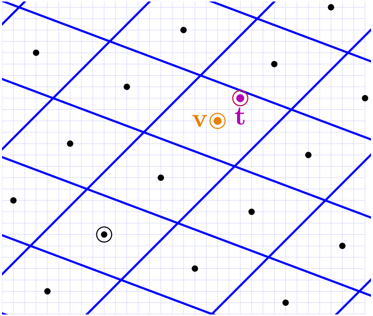


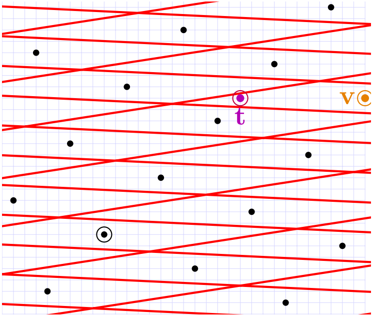
Round'off Algorithm [Lenstra, Babai]:

- Given a target **t**
- Find's **v** $\in L$ at the center the tile.

Bases and Fundamental Domains

Each basis defines a **parallelepipedic tiling**.

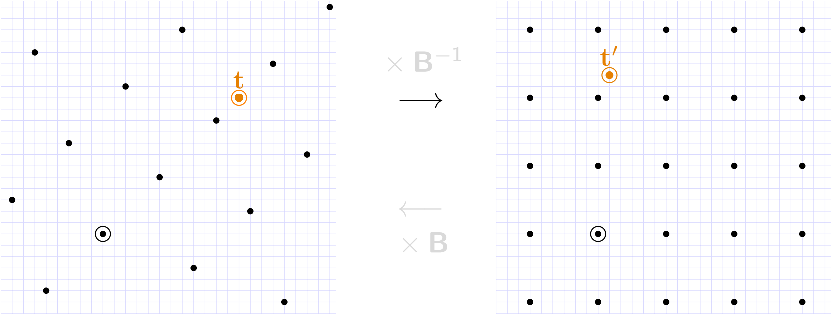




Round'off Algorithm [Lenstra, Babai]:

- Given a target **t**
- Find's **v** $\in L$ at the center the tile.

Round'off Algorithm

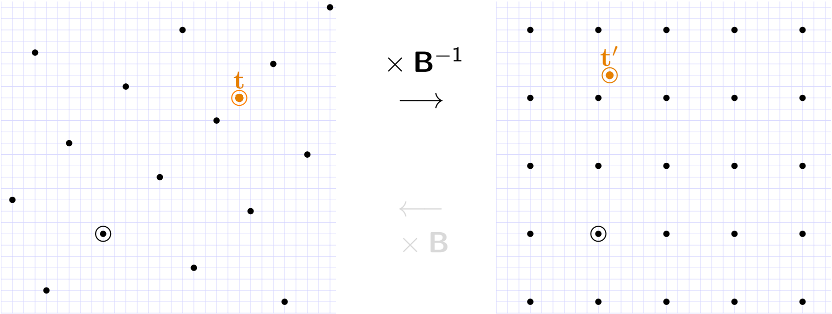


ROUNDOFF Algorithm [Lenstra,Babai]:

- Use $\mathbf{B}$ to switch to the lattice $\mathbb{Z}^n$ ($\times \mathbf{B}^{-1}$)
- round each coordinate (square tiling)
- switch back to L ($\times \mathbf{B}$)

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

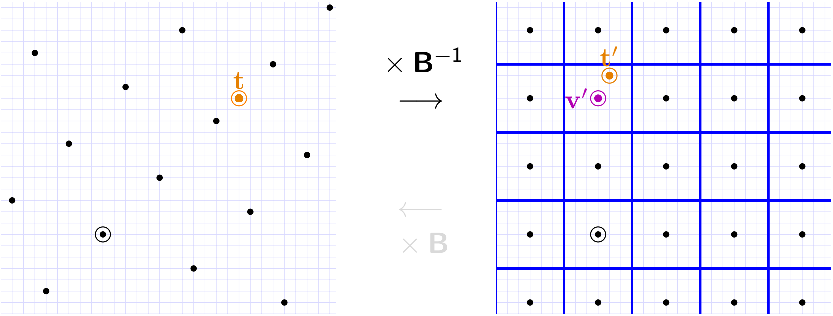
Round'off Algorithm



RoundOff Algorithm [Lenstra,Babai]:

- Use $\mathbf{B}$ to switch to the lattice $\mathbb{Z}^n$ $(\times \mathbf{B}^{-1})$
- round each coordinate (square tiling)
- switch back to L $(\times \mathbf{B})$

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

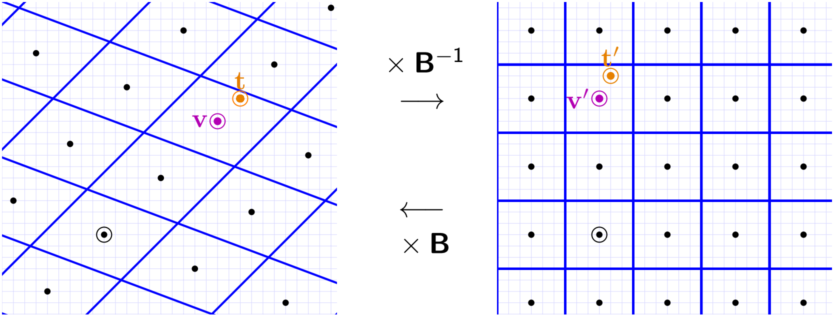


RoundOff Algorithm [Lenstra,Babai]:

- Use $\mathbf{B}$ to switch to the lattice $\mathbb{Z}^n$ ($\times\mathbf{B}^{-1}$)
- round each coordinate (square tiling)
- switch back to L ($\times\mathbf{B}$)

$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Round'off Algorithm



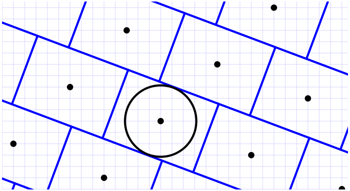
RoundOff Algorithm [Lenstra,Babai]:

- Use $\mathbf{B}$ to switch to the lattice $\mathbb{Z}^n$ $(\times \mathbf{B}^{-1})$
- round each coordinate (square tiling)
- switch back to L $(\times \mathbf{B})$

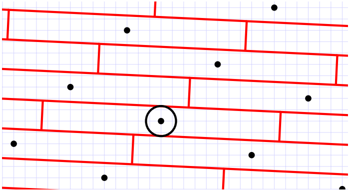
$$\mathbf{t}' = \mathbf{B}^{-1} \cdot \mathbf{t}; \quad \mathbf{v}' = \lfloor \mathbf{t}' \rceil; \quad \mathbf{v} = \mathbf{B} \cdot \mathbf{v}'$$

Nearest-Plane Algorithm

There is a better algorithm (NEARESTPLANE) based on Gram-Schmidt Orth. $\mathbf{B}^\star$ of a basis $\mathbf{B}$:



Decoding radius with $\mathbf{G}^\star$



Decoding radius with $\mathbf{B}^\star$

- Worst-case distance: $\frac{1}{2}\sqrt{\sum \|\mathbf{b}_i^\star\|^2}$ (Approx-CVP)
- Correct decoding of $\mathbf{t} = \mathbf{v} + \mathbf{e}$ where $\mathbf{v} \in \Lambda$ if (BDD)

$$\|\mathbf{e}\| \leq \frac{1}{2} \min \|\mathbf{b}_i^\star\|$$

Profile of a Basis

Good basis

$$\max \|\mathbf{b}_i^*\| \approx \min \|\mathbf{b}_i^*\|$$

Bad basis

$$\max \|\mathbf{b}_i^*\| \gg \min \|\mathbf{b}_i^*\|$$

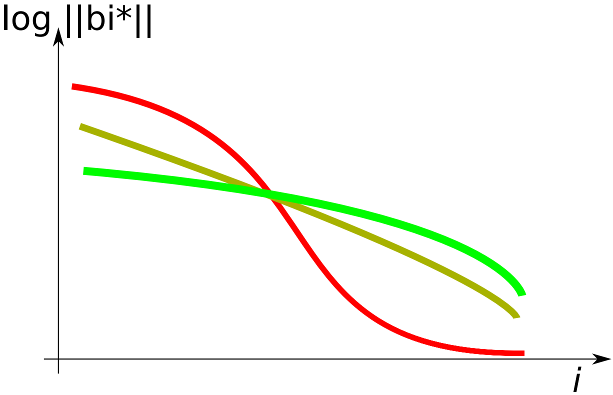
Profile of a Basis

Good basis

$$\max \|\mathbf{b}_i^*\| \approx \min \|\mathbf{b}_i^*\|$$

Bad basis

$$\max \|\mathbf{b}_i^*\| \gg \min \|\mathbf{b}_i^*\|$$



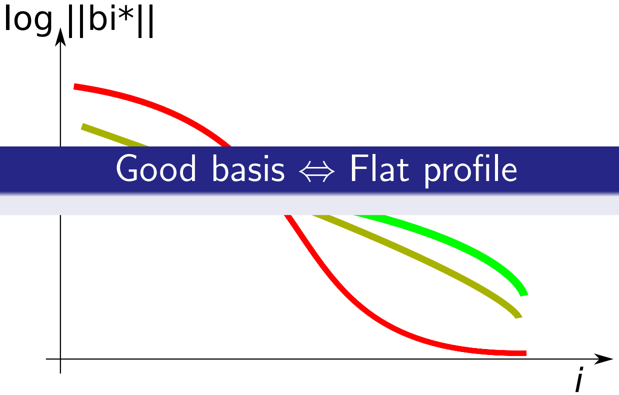
Profile of a Basis

Good basis

$$\max \|\mathbf{b}_i^*\| \approx \min \|\mathbf{b}_i^*\|$$

Bad basis

$$\max \|\mathbf{b}_i^*\| \gg \min \|\mathbf{b}_i^*\|$$



Good basis $\Leftrightarrow$ Flat profile

How Good can a Basis be ?

Theorem (Minkowski)

Let $C \in \mathbb{R}^n$ be a symmetric[a], convex and measurable set.
Let $\Lambda \in \mathbb{R}^n$ be a full dimensional lattice.
If $\mathrm{vol}(C) > 2^n \mathrm{vol}(\Lambda)$ then $\Lambda \cap C$ contains a non-zero point.

[a] $C = -C$

Applied to a Euclidean ball, we get

Corollary

Denoting $\lambda_1(\Lambda)$ the length of a shortest non-zero vector of the n-dimensional lattice Λ, and $\mathfrak{B}_n$ the unit euclidean ball

$$\lambda_1(\Lambda) \leq 2(\mathrm{vol}(\Lambda)/\mathrm{vol}(\mathfrak{B}_n))^{1/n} \qquad \approx \sqrt{2n/\pi e} \cdot \mathrm{vol}(\Lambda)^{1/n}$$

How Good can a Basis be ?

This shows that there exists a basis where $\|\mathbf{b}_1\| = \mathrm{vol}(\Lambda)^{1/n} \cdot O(\sqrt{n})$.
However if λ_1 gets too small, no good basis can exists !

Example (Layered lattice)

Consider the lattice generated by the following basis:

$$\begin{bmatrix} \epsilon & 0 \\ 0 & 1/\epsilon \end{bmatrix}$$

We have $\mathrm{vol}(\Lambda) = 1$, but all its basis $\mathbf{B}$ must have a vector of length $\geq 1/\epsilon$.

Solution:

1. Ignore such corner cases and focus on random lattices

OR 2. State quality of a basis wrt the property of the lattice

Dimension 2

The Wristwatch Lemma

Theorem (Wristwatch lemma)

Let Λ be a 2-dimensional lattice. Then there exists a basis $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2)$ such that

- $\mathbf{b}_1$ *is a shortest vector of* Λ.
- $|\langle \mathbf{b}_1, \mathbf{b}_2 \rangle| \leq \frac{1}{2} \|\mathbf{b}_1\|^2$.

$$\Rightarrow \|\mathbf{b}_2^\star\| \geq \sqrt{3/4} \cdot \|\mathbf{b}_1\|$$

$$\Rightarrow \mathrm{vol}(\Lambda) \geq \sqrt{3/4} \cdot \|\mathbf{b}_1\|^2$$

$$\alpha^2 + \beta^2 \geq 1$$

$$|\alpha| \leq 1/2$$

The Wristwatch Lemma

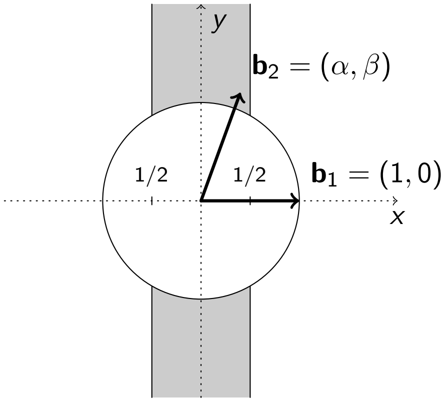
Theorem (Wristwatch lemma)

Let Λ be a 2-dimensional lattice. Then there exists a basis $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2)$ such that

- $\mathbf{b}_1$ *is a shortest vector of* Λ.
- $|\langle \mathbf{b}_1, \mathbf{b}_2 \rangle| \leq \frac{1}{2}\|\mathbf{b}_1\|^2$.

$$\Rightarrow \quad \|\mathbf{b}_2^\star\| \geq \sqrt{3/4} \cdot \|\mathbf{b}_1\|$$

$$\Rightarrow \quad \mathrm{vol}(\Lambda) \geq \sqrt{3/4} \cdot \|\mathbf{b}_1\|^2$$



$$\alpha^2 + \beta^2 \geq 1$$

$$|\alpha| \leq 1/2$$

Proof by Algorithm (Lagrange)

Require: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ of a lattice Λ.
Ensure: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ as in the Wristwatch lemma.

repeat
  swap $\mathbf{b}_1 \leftrightarrow \mathbf{b}_2$
  $k \leftarrow \lceil \langle \mathbf{b}_1, \mathbf{b}_2 \rangle / \|\mathbf{b}_1\|^2 \rfloor$
  $\mathbf{b}_2 \leftarrow \mathbf{b}_2 - k\mathbf{b}_1$
until $\|\mathbf{b}_1\| \leq \|\mathbf{b}_2\|$

Proof:

- Termination: $\|\mathbf{b}_1\|$ strictly decreasing within a discrete set
- $\mathbf{b}_1, \mathbf{b}_2$ is a basis: each step is a transformation in $GL_2(\mathbb{Z})$
- $|\langle \mathbf{b}_1, \mathbf{b}_2 \rangle| \leq \frac{1}{2}\|\mathbf{b}_1\|^2$: because we made sure of that
- $\mathbf{b}_1$ is the shortest lattice vector: left as exercise

Proof by Algorithm (Lagrange)

Require: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ of a lattice Λ.
Ensure: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ as in the Wristwatch lemma.

```
repeat
    swap b1 ↔ b2
    k ← ⌈⟨b1, b2⟩/‖b1‖²⌋
    b2 ← b2 − k b1
until ‖b1‖ ≤ ‖b2‖
```

$$\textbf{repeat}\quad \text{swap } \mathbf{b}_1 \leftrightarrow \mathbf{b}_2;\quad k \leftarrow \lceil \langle \mathbf{b}_1, \mathbf{b}_2\rangle / \|\mathbf{b}_1\|^2 \rfloor;\quad \mathbf{b}_2 \leftarrow \mathbf{b}_2 - k\mathbf{b}_1 \quad \textbf{until } \|\mathbf{b}_1\| \le \|\mathbf{b}_2\|$$

Proof:

- Termination: $\|\mathbf{b}_1\|$ strictly decreasing within a discrete set
- $\mathbf{b}_1, \mathbf{b}_2$ is a basis: each step is a transformation in $GL_2(\mathbb{Z})$
- $|\langle \mathbf{b}_1, \mathbf{b}_2\rangle| \le \frac{1}{2}\|\mathbf{b}_1\|^2$: because we made sure of that
- $\mathbf{b}_1$ is the shortest lattice vector: left as exercise

Proof by Algorithm (Lagrange)

Require: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ of a lattice Λ.
Ensure: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ as in the Wristwatch lemma.

```
repeat
    swap b_1 ↔ b_2
    k ← ⌈⟨b_1, b_2⟩/‖b_1‖^2⌋
    b_2 ← b_2 − k b_1
until ‖b_1‖ ≤ ‖b_2‖
```

Proof:

- Termination: $\|\mathbf{b}_1\|$ strictly decreasing within a discrete set
- $\mathbf{b}_1, \mathbf{b}_2$ is a basis: each step is a transformation in $GL_2(\mathbb{Z})$
- $|\langle \mathbf{b}_1, \mathbf{b}_2 \rangle| \leq \frac{1}{2}\|\mathbf{b}_1\|^2$: because we made sure of that
- $\mathbf{b}_1$ is the shortest lattice vector: left as exercise

Proof by Algorithm (Lagrange)

Require: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ of a lattice Λ.
Ensure: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ as in the Wristwatch lemma.

```
repeat
    swap b1 ↔ b2
    k ← ⌈⟨b1, b2⟩/‖b1‖²⌋
    b2 ← b2 − k b1
until ‖b1‖ ≤ ‖b2‖
```

Proof:

- Termination: $\|\mathbf{b}_1\|$ strictly decreasing within a discrete set
- $\mathbf{b}_1, \mathbf{b}_2$ is a basis: each step is a transformation in $GL_2(\mathbb{Z})$
- $|\langle \mathbf{b}_1, \mathbf{b}_2 \rangle| \leq \frac{1}{2}\|\mathbf{b}_1\|^2$: because we made sure of that
- $\mathbf{b}_1$ is the shortest lattice vector: left as exercise

The Algorithm is (very) Fast

Require: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ of a lattice Λ.
Ensure: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ as in the Wristwatch lemma.

repeat
  swap $\mathbf{b}_1 \leftrightarrow \mathbf{b}_2$
  $k \leftarrow \lceil \langle \mathbf{b}_1, \mathbf{b}_2 \rangle / \|\mathbf{b}_1\|^2 \rfloor$
  $\mathbf{b}_2 \leftarrow \mathbf{b}_2 - k\mathbf{b}_1$
until $\|\mathbf{b}_1\| \leq \|\mathbf{b}_2\|$

Lemma

The Lagrange reduction algorithm terminates after $O\left(\log \frac{\|\mathbf{b}_1\|}{\sqrt{\det \Lambda}}\right)$ *iterations.*

Proof idea: $\mathbf{b}_1$ decrease by a factor 2 at each step, expect maybe for the last 25 steps.

The Algorithm is (very) Fast

Require: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ of a lattice Λ.
Ensure: A basis $(\mathbf{b}_1, \mathbf{b}_2)$ as in the Wristwatch lemma.

```
repeat
    swap b1 ↔ b2
    k ← ⌈⟨b1, b2⟩/‖b1‖^2⌋
    b2 ← b2 − k b1
until ‖b1‖ ≤ ‖b2‖
```

repeat
- swap $\mathbf{b}_1 \leftrightarrow \mathbf{b}_2$
- $k \leftarrow \lceil \langle \mathbf{b}_1, \mathbf{b}_2 \rangle / \|\mathbf{b}_1\|^2 \rfloor$
- $\mathbf{b}_2 \leftarrow \mathbf{b}_2 - k\mathbf{b}_1$

until $\|\mathbf{b}_1\| \leq \|\mathbf{b}_2\|$

Lemma

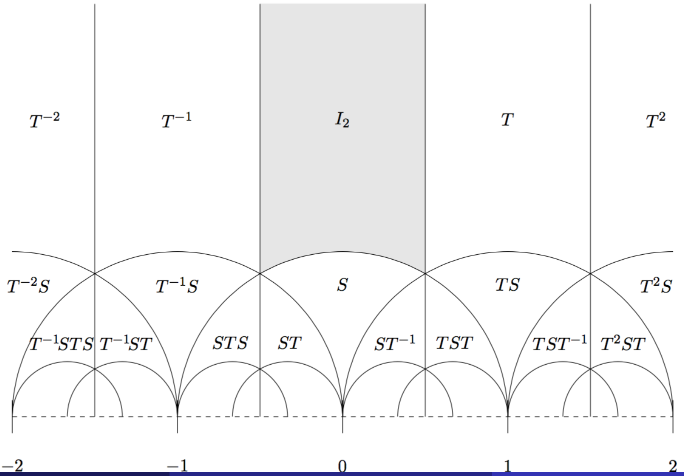
The Lagrange reduction algorithm terminates after $O\left(\log \frac{\|\mathbf{b}_1\|}{\sqrt{\det \Lambda}}\right)$ *iterations.*

Proof idea: $\mathbf{b}_1$ decrease by a factor 2 at each step, expect maybe for the last 25 steps.

Tesselation

Figure: Action of $GL_2(\mathbb{Z})$ over $GL_2(\mathbb{R})$ (modulo rotations and scaling).

$$T : \mathbf{b}_2 \leftarrow \mathbf{b}_2 + \mathbf{b}_1; \qquad S : \mathbf{b}_1 \leftrightarrow \mathbf{b}_2$$



LLL

Projected Sublattices

For a given basis $\mathbf{B}$ of the lattice L

- Define π_i: the projection orthogonally to $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$.
- Define $\mathbf{B}_{[i:j]} = (\pi_i(\mathbf{b}_i), \ldots \pi_i(\mathbf{b}_j))$
- $L_{[i:j]}$ is the *projected sublattice* of L spanned by $\mathbf{B}_{[i:j]}$

Identities:

- For $i \leq i' \leq j' \leq j$: $(\mathbf{B}_{[i:j]})_{[i':j']} = \mathbf{B}_{[i':j']}$
- $(\mathbf{B}_{[i:j]})^\star = (\mathbf{b}_i^\star, \ldots, \mathbf{b}_j^\star)$

Projected Sublattices

For a given basis $\mathbf{B}$ of the lattice L

- Define π_i: the projection orthogonally to $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$.
- Define $\mathbf{B}_{[i:j]} = (\pi_i(\mathbf{b}_i), \ldots \pi_i(\mathbf{b}_j))$
- $L_{[i:j]}$ is the *projected sublattice* of L spanned by $\mathbf{B}_{[i:j]}$

Identities:

- For $i \leq i' \leq j' \leq j$: $(\mathbf{B}_{[i:j]})_{[i':j']} = \mathbf{B}_{[i':j']}$
- $(\mathbf{B}_{[i:j]})^\star = (\mathbf{b}_i^\star, \ldots, \mathbf{b}_j^\star)$

Projected Sublattices

For a given basis $\mathbf{B}$ of the lattice L

- Define π_i: the projection orthogonally to $\mathbf{b}_1, \ldots, \mathbf{b}_{i-1}$.
- Define $\mathbf{B}_{[i:j]} = (\pi_i(\mathbf{b}_i), \ldots \pi_i(\mathbf{b}_j))$
- $L_{[i:j]}$ is the *projected sublattice* of L spanned by $\mathbf{B}_{[i:j]}$

Identities:

- For $i \leq i' \leq j' \leq j$: $(\mathbf{B}_{[i:j]})_{[i':j']} = \mathbf{B}_{[i':j']}$
- $(\mathbf{B}_{[i:j]})^\star = (\mathbf{b}_i^\star, \ldots, \mathbf{b}_j^\star)$

Everybody loves commutative diagrams

Relations between the projected sublattices:

$$
\begin{array}{ccccccccccc}
L = & L_{[1:n]} & \supset & L_{[1:n-1]} & \supset & \dots & \supset & L_{[1:2]} & \supset & L_{[1:1]} \\
& \downarrow & & \downarrow & & & & \downarrow & & \\
& L_{[2:n]} & \supset & L_{[2:n-1]} & \supset & \dots & \supset & L_{[2:2]} & & \\
& \downarrow & & \downarrow & & & & & & \\
& \vdots & & \vdots & & \cdot & & & & \\
& \downarrow & & \downarrow & & & & & & \\
& L_{[n-1:n]} & \supset & L_{[n-1:n-1]} & & & & & & \\
& \downarrow & & & & & & & & \\
& L_{[n:n]} & & & & & & & &
\end{array}
$$

$\downarrow$: Projection orthogonally to $\mathbf{b}_i$ for some i.

LLL-reduced basis

Definition (ϵ-Lovasz reduced)

Let $\epsilon \geq 0$. A 2-dimensional basis $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2)$ is said ϵ-Lovasz reduced if

$$\|\mathbf{b}_2^\star\| \geq (\sqrt{3/4} + \epsilon) \cdot \|\mathbf{b}_1\|.$$

Definition (ϵ-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

An n-dim basis $\mathbf{B}$ is said ϵ-LLL reduced if $\mathbf{B}_{[i:i+1]}$ is ϵ-Lovasz reduced for all $i \in \{1, \ldots n-1\}$. Equivalently, for all i,

$$\|\mathbf{b}_{i+1}^\star\| \geq (\sqrt{3/4} + \epsilon) \cdot \|\mathbf{b}_i\|.$$

LLL-reduced : " Gram-Schmidt norms do not deacrease too fast. "

LLL-reduced basis

Definition (ϵ-Lovasz reduced)

Let $\epsilon \geq 0$. A 2-dimensional basis $\mathbf{B} = (\mathbf{b}_1, \mathbf{b}_2)$ is said ϵ-Lovasz reduced if

$$\|\mathbf{b}_2^\star\| \geq (\sqrt{3/4} + \epsilon) \cdot \|\mathbf{b}_1\|.$$

Definition (ϵ-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

An n-dim basis $\mathbf{B}$ is said ϵ-LLL reduced if $\mathbf{B}_{[i:i+1]}$ is ϵ-Lovasz reduced for all $i \in \{1, \ldots n-1\}$. Equivalently, for all i,

$$\|\mathbf{b}_{i+1}^\star\| \geq (\sqrt{3/4} + \epsilon) \cdot \|\mathbf{b}_i\|.$$

LLL-reduced : " Gram-Schmidt norms do not deacrease too fast. "

Properties of LLL-reduced basis

Theorem

Let $B = (\mathbf{b}_1, \ldots, \mathbf{b}_n)$ *be an* ϵ*-LLL reduced basis. Set* $\Lambda = \mathcal{L}(B)$ *and* $\alpha = \sqrt{4/3} + \epsilon$*. Then, we have*

1. $\|\mathbf{b}_1\| \leq \alpha^{\frac{n-1}{2}} \cdot \det(\Lambda)^{1/n}$ — *Root Hermite-factor Bound*
2. $\|\mathbf{b}_1\| \leq \alpha^{n-1} \cdot \lambda_1(\Lambda)$ — *Approximation factor Bound*
3. $\|\mathbf{b}_i^\star\| \leq \alpha^{n-i} \cdot \lambda_i(\Lambda)$.
4. $\|\mathbf{b}_i\| \leq \alpha^{i-1} \cdot \|\mathbf{b}_i^\star\| \leq \alpha^{n-1} \cdot \lambda_i(\Lambda)$
5. $\prod_{i=1}^n \|\mathbf{b}_i\| \leq \alpha^{\frac{n(n-1)}{2}} \cdot \det(\Lambda)$.

LLL Algorithm

Definition (ϵ-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

An n-dim basis $\mathbf{B}$ is said ϵ-LLL reduced if $B_{[i:i+1]}$ is ϵ-Lovasz reduced for all $i \in \{1, \ldots n-1\}$

The LLL algorithm

Require: A basis $\mathbf{B}$ of a lattice Λ.
Ensure: An ϵ-LLL reduced basis $\mathbf{B}$ of the same lattice.
  while $\exists i, \mathbf{B}_{[i:i+1]}$ is not ϵ-Lovasz reduced **do**
    Lagrange reduce it...
  end while

LLL Algorithm

Definition (ϵ-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

An n-dim basis $\mathbf{B}$ is said ϵ-LLL reduced if $B_{[i:i+1]}$ is ϵ-Lovasz reduced for all $i \in \{1, \ldots n-1\}$

The LLL algorithm

Require: A basis $\mathbf{B}$ of a lattice Λ.

Ensure: An ϵ-LLL reduced basis $\mathbf{B}$ of the same lattice.

  while $\exists i, \mathbf{B}_{[i:i+1]}$ is not ϵ-Lovasz reduced **do**

    Lagrange reduce it...

  end while

LLL Algorithm

Definition (ϵ-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

An n-dim basis $\mathbf{B}$ is said ϵ-LLL reduced if $B_{[i:i+1]}$ is ϵ-Lovasz reduced for all $i \in \{1, \ldots n-1\}$

The LLL algorithm

Require: A basis $\mathbf{B}$ of a lattice Λ.
Ensure: An ϵ-LLL reduced basis $\mathbf{B}$ of the same lattice.
  while $\exists i, \mathbf{B}_{[i:i+1]}$ is not ϵ-Lovasz reduced **do**
    Lagrange reduce it...
  end while

LLL Efficiency

Lemma (Efficiency of LLL)

For a fixed $\epsilon > 0$, and for integral inputs $\mathbf{B} \in \mathbb{Z}^{n \times n}$, the LLL algorithm runs in polynomial time.

Proof:

- Define a potential

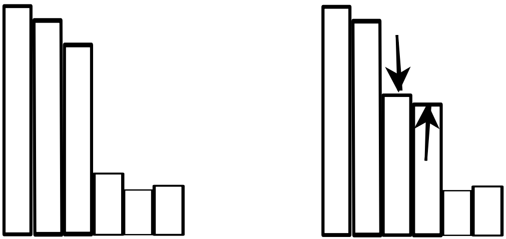
$$P(\mathbf{B}) = \sum (n-i) \log(\|\mathbf{b}_i^\star\|) = \sum \log(\mathrm{vol}(L_{[1:i]}))$$

.

- Each loop of the LLL algorithm decreases $P(\mathbf{B})$ by $\log(1+\epsilon)$
- Note that for integral $\mathbf{B} \in \mathbb{Z}^{n \times n}$, $P(\mathbf{B}) \geq 0$.
- Concludes

$$\#loops \leq \frac{P(\mathbf{B}_{init})}{\log(1+\epsilon)} \leq \frac{n^2 \cdot log(\|\mathbf{B}_{init}\|_\infty)}{\log(1+\epsilon)}.$$

LLL : a 2-legged ant



- Columns height : $\log(\|\mathbf{b}_i^\star\|)$
- Volume invariant: mass of sand is preserved
- Lagrange reduction : almost level 2 consecutive columns
- Potential decreases : the mass of sand moves toward the right.

Remarks

Note the integrality condition

By scaling, can also wrok over $\mathbb{Q}$. However, even assuming perfect computation over $\mathbb{R}$, the above proof fails for real inputs.

The above proof is incomplete

$\#loops = poly$. Ok. But how large are the intermediate values ?

$\epsilon = 0$

The algorithm still terminates, but not poly-time.

Output quality

On most inputs, LLL behave much better in practice than in theory.

- Theory: $\alpha = \sqrt{4/3} + \epsilon \approx 1.155$
- Practice: $\alpha \approx 1.022$

Remarks

Note the integrality condition

By scaling, can also wrok over $\mathbb{Q}$. However, even assuming perfect computation over $\mathbb{R}$, the above proof fails for real inputs.

The above proof is incomplete

$\#loops = poly$. Ok. But how large are the intermediate values ?

$\epsilon = 0$

The algorithm still terminates, but not poly-time.

Output quality

On most inputs, LLL behave much better in practice than in theory.

- Theory: $\alpha = \sqrt{4/3} + \epsilon \approx 1.155$
- Practice: $\alpha \approx 1.022$

Remarks

Note the integrality condition

By scaling, can also wrok over $\mathbb{Q}$. However, even assuming perfect computation over $\mathbb{R}$, the above proof fails for real inputs.

The above proof is incomplete

$\#loops = poly$. Ok. But how large are the intermediate values ?

$\epsilon = 0$

The algorithm still terminates, but not poly-time.

Output quality

On most inputs, LLL behave much better in practice than in theory.

- Theory: $\alpha = \sqrt{4/3} + \epsilon \approx 1.155$
- Practice: $\alpha \approx 1.022$

Remarks

Note the integrality condition

By scaling, can also wrok over $\mathbb{Q}$. However, even assuming perfect computation over $\mathbb{R}$, the above proof fails for real inputs.

The above proof is incomplete

$\#loops = poly$. Ok. But how large are the intermediate values ?

$\epsilon = 0$

The algorithm still terminates, but not poly-time.

Output quality

On most inputs, LLL behave much better in practice than in theory.

- Theory: $\alpha = \sqrt{4/3} + \epsilon \approx 1.155$
- Practice: $\alpha \approx 1.022$

BKZ: a b-legged ant

Bigger Local Improvement

- Focus on a block $[i:j]$, of dimension $b = j - i - 1$
- Find the shortest vector v of the projected sublattice $L_{[i:j]}$

"a puzzle of it
"the right combination."

- Construct a unimodular matrix $\mathbf{U}$ such that $\mathbf{T}_{[i:j]} \cdot \mathbf{U} = [\mathbf{v}, *, *, \ldots]$. Apply $\mathbf{U}$ (locally).
- The new $\mathbf{b}_i^* = v$ got shorter!
- The other $\mathbf{b}_{i+1}^*, \ldots, \mathbf{b}_j^*$ will change as well

Bigger Local Improvement

- Focus on a block $[i:j]$, of dimension $b = j - i - 1$
- Find the shortest vector v of the projected sublattice $L_{[i:j]}$

"a puzzle of it
"the right combination."

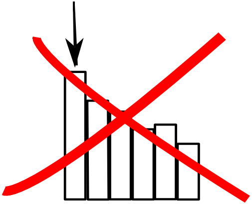
- Construct a unimodular matrix $\mathbf{U}$ such that $\mathbf{T}_{[i:j]} \cdot \mathbf{U} = [\mathbf{v}, *, *, \dots]$. Apply $\mathbf{U}$ (locally).
- The new $\mathbf{b}_i^* = v$ got shorter!
- The other $\mathbf{b}_{i+1}^*, \dots, \mathbf{b}_j^*$ will change as well

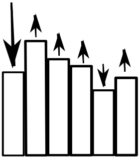
Bigger Local Improvement

- Focus on a block $[i : j]$, of dimension $b = j - i - 1$
- Find the shortest vector v of the projected sublattice $L_{[i:j]}$

"a puzzle of it
"the right combination."

- Construct a unimodular matrix $\mathbf{U}$ such that $\mathbf{T}_{[i:j]} \cdot \mathbf{U} = [\mathbf{v}, *, *, \dots]$. Apply $\mathbf{U}$ (locally).
- The new $\mathbf{b}_i^* = v$ got shorter!
- The other $\mathbf{b}_{i+1}^*, \dots, \mathbf{b}_j^*$ will change as well





b: Blocksize

Run the local improvements for consecutive blocks:

$[1 : b], [2 : b+1], [3 : b+2], \ldots, [n-b : n], [n-b+1 : n], \ldots \; [n-1 : n]$

This is called a tour.

Repeat tours until satisfication (or convergence).

BKZ-b

b: Blocksize

Run the local improvements for consecutive blocks:

$[1:b], [2:b+1], [3:b+2], \ldots, [n-b:n], [n-b+1:n], \ldots\ [n-1:n]$

This is called a tour.

Repeat tours until satisfication (or convergence).

BKZ prediction

At the end of BKZ, $\mathbf{b}_i^*$ is the shortest vector of $L_{[i:i+b]}$.

Worse case bound (Minkowski)

$$\|\mathbf{b}_i^\star\| \lessapprox \sqrt{\frac{2n}{\pi e}} \cdot \mathrm{vol}(L_{[i:i+b]})^{1/n} = \sqrt{\frac{2n}{\pi e}} \cdot \prod_{k=i}^{i+b} \|\mathbf{b}_k^\star\|^{1/n}$$

Average case (for $b \geq 50$)

$$\|\mathbf{b}_i^\star\| \approx \sqrt{\frac{n}{2\pi e}} \cdot \prod_{k=i}^{i+b} \|\mathbf{b}_k^\star\|^{1/n}$$

Take logs, get a linear recurrence $\Rightarrow$ shape is linear, slope is a known function of blocksize b. Known as the Geometric Series Assumption (GSA).

BKZ time vs. approximation

- Solving SVP-b cost

$$2^{\tilde{\Theta}(b)}$$

.

- BKZ can run for very long. But stopping after poly(n, b) steps is sufficient
- The total cost of BKZ is therefore

$$n^{O(1)} \cdot 2^{\tilde{\Theta}(b)}$$

- Achieve an approximation factor:

$$\|b_1\| \approx (b/2\pi e)^{n/2b} \cdot \mathrm{vol}(L)$$

Enumeration

Enumeration in 1 slide

Goal: enumerate all the lattice points in a ball of radius r : $r\mathfrak{B} \cap L$.
Remember the chain of lattices (by projections)

$$L = L_{[1:n]} \xrightarrow{\pi_1} L_{[2:n]} \xrightarrow{\pi_2} L_{[3:n]} \to \ldots \xrightarrow{\pi_{n-1}} L_{[n:n]} \xrightarrow{\pi_n} \{0\}$$

Projection only decrease length: $\pi_i(r\mathfrak{B} \cap L_{[i:n]}) \subset r\mathfrak{B} \cap L_{[i+1:n]}$.

Enumeration algorithm

Compute sets $S_i = B \cap L_{[i+1:n]}$ using the recursion

$$S_n = \{0\}; \qquad S_{i-1} = \pi_i^{-1}(S_i) \cap r\mathfrak{B}$$

Cost of Enumeration

The cost of the algorithm is essentially proportional to $\sum |S_i|$.

$$|S_i| \approx r^{n-i} \cdot \mathrm{vol}(\mathfrak{B}_{n-i}) / \mathrm{vol}(L_{[i:n]}).$$

Depends projected sublattice volumes: SVP faster if the basis is well reduced !

- LLL reduced : $2^{\Theta(n^2)}$
- BKZ-b where $b = n - O(\log n)$: $n^{n/2e+o(n)}$ (Michael ?)

Optimal complexity is reached using a mixed recursion:

$$\text{BKZ-}b_1 \looparrowright \text{SVP-}b_1 \looparrowright \text{BKZ-}b_2 \looparrowright \cdots \looparrowright \text{SVP-60} \looparrowright \text{LLL}$$