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.

xor a
bit 7,b
jr z,bc_nneg
ld hl,0
xor a
sbc hl,bc
push hl
pop bc
ld a,1

An algorithmic ant.
The Queen of ants summoned the algorithmic ant,
The Queen of ants summoned the algorithmic ant,

"See this sand pile."
"I want it flat!"
Looking closer at the Sandpile,
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 hoped the ant.
But the sandpile is whimsical, each excavation is a puzzle of its own.
But the sandpile is whimsical, each excavation is a puzzle of its own.
But the sandpile is whimsical, each excavation is a puzzle of its own.

Columns are tied in mysterious ways: 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
**Definition**

A lattice $L$ is a discrete subgroup of a finite-dimensional Euclidean vector space.
The Good-basis/Bad-basis Routine

Good Basis $G$ of $L$  

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

Bad Basis $B$ of $L$  

$B \rightarrow G$ : hard (LLL, BKZ, Lattice Sieve...).
Lattice reduction: a pedantic summary

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

   \[
   \text{if } B \sim G, \text{ then } QB \sim QG
   \]
Lattice reduction: a pedantic summary

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

\[
\text{if } B \leadsto G, \text{ then } QB \leadsto QG
\]
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 $\mathbf{Q} \in O_n(\mathbb{R})$: if $\mathbf{B} \leadsto \mathbf{G}$, then $\mathbf{Q}\mathbf{B} \leadsto \mathbf{QG}$
An important invariant: the Volume

For any two bases $G, B$ of the same lattice $\Lambda$:

$$\det(GG^t) = \det(BB^t).$$

We can therefore define:

$$\text{vol}(\Lambda) = \sqrt{\det(GG^t)}.$$

Geometrically: the volume of any fundamental domain of $\Lambda$.

Let $G^*$ be the Gram-Schmidt Orthogonalization of $G$

$G^*$ is not a basis of $\Lambda$, nevertheless:

$$\text{vol}(\Lambda) = \sqrt{\det(G^*G^{*t})} = \prod \|g^*_i\|. $$
An important invariant: the Volume

For any two bases $G$, $B$ of the same lattice $\Lambda$:

$$\det(GG^t) = \det(BB^t).$$

We can therefore define:

$$\text{vol}(\Lambda) = \sqrt{\det(GG^t)}.$$

Geometrically: the volume of any fundamental domain of $\Lambda$.

Let $G^*$ be the Gram-Schmidt Orthogonalization of $G$

$G^*$ is not a basis of $\Lambda$, nevertheless:

$$\text{vol}(\Lambda) = \sqrt{\det(G^*G^{*t})} = \prod ||g^*_i||.$$
What is a “Good” basis

Recall that, independently of the basis $G$ it holds that:

$$\text{vol}(\Lambda) = \prod \|g_i^*\|.$$  

Therefore, it is somehow equivalent that

- $\max_i \|g_i^*\|$ is small
- $\min_i \|g_i^*\|$ is large
- $\kappa(G) = \max_i \|g_i^*\| / \min_i \|g_i^*\|$ is small

**Good basis**

$$\max \|b_i^*\| \approx \min \|b_i^*\|, \quad \text{(equivalently: } \forall i, \|b_i^*\| \approx \text{vol}(\Lambda)^{1/n})$$

**Bad basis**

$$\max \|b_i^*\| \gg \min \|b_i^*\|$$
What is a “Good” basis

Recall that, independently of the basis $G$ it holds that:

$$\text{vol}(\Lambda) = \prod \|g_i^*\|.$$ 

Therefore, it is somehow equivalent that

- $\max_i \|g_i^*\|$ is small
- $\min_i \|g_i^*\|$ is large
- $\kappa(G) = \max_i \|g_i^*\|/ \min_i \|g_i^*\|$ is small

### Good basis

$$\max \|b_i^*\| \approx \min \|b_i^*\|, \quad \text{(equivalently: } \forall i, \|b_i^*\| \approx \text{vol}(\Lambda)^{1/n})$$

### Bad basis

$$\max \|b_i^*\| \gg \min \|b_i^*\|$$
What is a “Good” basis

Recall that, independently of the basis $G$ it holds that:

$$\text{vol}(\Lambda) = \prod \|g_i^*\|.$$  

Therefore, it is somehow equivalent that

- $\max_i \|g_i^*\|$ is small
- $\min_i \|g_i^*\|$ is large
- $\kappa(G) = \max_i \|g_i^*\| / \min_i \|g_i^*\|$ is small

**Good basis**

$$\max \|b_i^*\| \approx \min \|b_i^*\|,$$  

(equivalently: $\forall i, \|b_i^*\| \approx \text{vol}(\Lambda)^{1/n}$)

**Bad basis**

$$\max \|b_i^*\| \gg \min \|b_i^*\|.$$
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.
Each basis defines a **parallelepipedic tiling**.

Round’off Algorithm [Lenstra, Babai]:

- Given a target $\mathbf{t}$
- Find’s $\mathbf{v} \in L$ at the center the tile.
Round’off Algorithm

**RoundOff Algorithm** [Lenstra, Babai]:

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

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

Léo Ducas (CWI, Amsterdam)
Lattice Reduction
Math of PKC. Mar. 2019 17 / 43
Round’off Algorithm [Lenstra, Babai]:

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

$t' = B^{-1} \cdot t; \quad v' = \lceil t' \rceil; \quad v = B \cdot v'$
Round’off Algorithm

**RoundOff Algorithm** [Lenstra,Babai]:

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

$t' = B^{-1} \cdot t; \quad v' = \lfloor t' \rfloor; \quad v = B \cdot v'$
Round’off Algorithm

**RoundOff Algorithm** [Lenstra, Babai]:
- Use $B$ to switch to the lattice \( \mathbb{Z}^n \times B^{-1} \)
- round each coordinate (square tiling)
- switch back to $L \times B$

\[
t' = B^{-1} \cdot t; \quad v' = \lfloor t' \rfloor; \quad v = B \cdot v'
\]
Nearest-Plane Algorithm

There is a better algorithm (NEARESTPLANE) based on Gram-Schmidt Orth. $B^*$ of a basis $B$:

- Decoding radius with $G^*$
- Decoding radius with $B^*$

- Worst-case distance: $\frac{1}{2} \sqrt{\sum \|b^*_i\|^2}$ (Approx-CVP)
- Correct decoding of $t = v + e$ where $v \in \Lambda$ if
  $$\|e\| \leq \frac{1}{2} \min \|b^*_i\|$$ (BDD)
### Profile of a Basis

<table>
<thead>
<tr>
<th>Good basis</th>
<th>Bad basis</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ \max | b_i^* | \approx \min | b_i^* | ]</td>
<td>[ \max | b_i^* | \gg \min | b_i^* | ]</td>
</tr>
</tbody>
</table>
Profile of a Basis

**Good basis**

\[ \max \| b_i^* \| \approx \min \| b_i^* \| \]

**Bad basis**

\[ \max \| b_i^* \| \gg \min \| b_i^* \| \]

\[
\log \| b_i^* \|
\]

\[ i \]
Profile of a Basis

**Good basis**
\[
\max \|b_i^*\| \approx \min \|b_i^*\|
\]

**Bad basis**
\[
\max \|b_i^*\| \gg \min \|b_i^*\|
\]

**Good basis ⇔ Flat profile**

\[
\log \|b_i^*\|
\]
How Good can a Basis be?

**Theorem (Minkowski)**

Let $C \in \mathbb{R}^n$ be a symmetric, convex and measurable set.
Let $\Lambda \in \mathbb{R}^n$ be a full dimensional lattice.

If $\text{vol}(C) > 2^n \text{vol}(\Lambda)$ then $\Lambda \cap C$ contains a non-zero point.

\[ C = -C \]

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 $\Lambda$, and $\mathcal{B}_n$ the unit euclidean ball

\[ \lambda_1(\Lambda) \leq 2(\text{vol}(\Lambda)/\text{vol}(\mathcal{B}_n))^{1/n} \approx \sqrt{2n/\pi e} \cdot \text{vol}(\Lambda)^{1/n} \]
This shows that there exists a basis where $\|b_1\| = \text{vol}(\Lambda)^{1/n} \cdot O(\sqrt{n})$. However if $\lambda_1$ gets too small, no good basis can exist!

**Example (Layered lattice)**

Consider the lattice generated by the following basis:

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

We have $\text{vol}(\Lambda) = 1$, but all its basis $B$ 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 \( \Lambda \) be a 2-dimensional lattice. Then there exists a basis \( B = (b_1, b_2) \) such that:

- \( b_1 \) is a shortest vector of \( \Lambda \).
- \( |\langle b_1, b_2 \rangle| \leq \frac{1}{2} \|b_1\|^2 \).

\[
\Rightarrow \|b_2^*\| \geq \sqrt{3/4} \cdot \|b_1\| \\
\Rightarrow \text{vol}(\Lambda) \geq \sqrt{3/4} \cdot \|b_1\|^2
\]
The Wristwatch Lemma

Let \( \Lambda \) be a 2-dimensional lattice. Then there exists a basis \( B = (b_1, b_2) \) such that

- \( b_1 \) is a shortest vector of \( \Lambda \).
- \( |\langle b_1, b_2 \rangle| \leq \frac{1}{2} \| b_1 \|^2 \).

\[ \Rightarrow \| b_2^\perp \| \geq \sqrt{\frac{3}{4}} \cdot \| b_1 \| \]
\[ \Rightarrow \text{vol}(\Lambda) \geq \sqrt{\frac{3}{4}} \cdot \| b_1 \|^2 \]

\[ \alpha^2 + \beta^2 \geq 1 \]
\[ |\alpha| \leq 1/2 \]
Proof by Algorithm (Lagrange)

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

repeat
    swap \(\mathbf{b}_1 \leftrightarrow \mathbf{b}_2\)
    \(k \leftarrow \lceil \langle \mathbf{b}_1, \mathbf{b}_2 \rangle / \| \mathbf{b}_1 \|^2 \rceil\)
    \(\mathbf{b}_2 \leftarrow \mathbf{b}_2 - k \mathbf{b}_1\)
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 \(\Lambda\).

**Ensure:** A basis \((\mathbf{b}_1, \mathbf{b}_2)\) as in the Wristwatch lemma.

\[
\text{repeat}
\]
\[
\text{swap } \mathbf{b}_1 \leftrightarrow \mathbf{b}_2
\]
\[
k \leftarrow \left\lceil \langle \mathbf{b}_1, \mathbf{b}_2 \rangle / \|\mathbf{b}_1\|^2 \right\rceil
\]
\[
\mathbf{b}_2 \leftarrow \mathbf{b}_2 - k \mathbf{b}_1
\]

\text{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 \((b_1, b_2)\) of a lattice \(\Lambda\).

**Ensure:** A basis \((b_1, b_2)\) as in the Wristwatch lemma.

```
repeat
    swap \(b_1 \leftrightarrow b_2\)
    \(k \leftarrow \left\lceil \frac{\langle b_1, b_2 \rangle}{\|b_1\|^2} \right\rceil\)
    \(b_2 \leftarrow b_2 - kb_1\)
until \(\|b_1\| \leq \|b_2\|\)
```

**Proof:**

- **Termination:** \(\|b_1\|\) strictly decreasing within a discrete set
- \(b_1, b_2\) is a basis: each step is a transformation in \(GL_2(\mathbb{Z})\)
- \(\langle b_1, b_2 \rangle \leq \frac{1}{2}\|b_1\|^2\) : because we made sure of that
- \(b_1\) is the shortest lattice vector: left as exercise
**Proof by Algorithm (Lagrange)**

**Require:** A basis \((b_1, b_2)\) of a lattice \(\Lambda\).

**Ensure:** A basis \((b_1, b_2)\) as in the Wristwatch lemma.

```
repeat
  swap \(b_1 \leftrightarrow b_2\)
  \(k \leftarrow \lceil \langle b_1, b_2 \rangle / \|b_1\|^2 \rceil\)
  \(b_2 \leftarrow b_2 - kb_1\)
until \(\|b_1\| \leq \|b_2\|\)
```

**Proof:**

- **Termination:** \(\|b_1\|\) strictly decreasing within a discrete set
- \(b_1, b_2\) is a basis: each step is a transformation in \(GL_2(\mathbb{Z})\)
- \(\langle b_1, b_2 \rangle \leq \frac{1}{2}\|b_1\|^2\) : because we made sure of that
- \(b_1\) is the shortest lattice vector: left as exercise
The Algorithm is (very) Fast

**Require:** A basis \((b_1, b_2)\) of a lattice \(\Lambda\).

**Ensure:** A basis \((b_1, b_2)\) as in the Wristwatch lemma.

\[
\begin{align*}
\text{repeat} & \\
\quad & \text{swap } b_1 \leftrightarrow b_2 \\
\quad & k \leftarrow \left\lfloor \frac{\langle b_1, b_2 \rangle}{\|b_1\|^2} \right\rfloor \\
\quad & b_2 \leftarrow b_2 - kb_1 \\
\text{until } & \|b_1\| \leq \|b_2\|
\end{align*}
\]

**Lemma**

*The Lagrange reduction algorithm terminates after* \(O\left(\log \frac{\|b_1\|}{\sqrt{\det \Lambda}}\right)\) *iterations.*

Proof idea: \(b_1\) decrease by a factor 2 at each step, expect maybe for the last 25 steps.
The Algorithm is (very) Fast

\textbf{Require: } A basis \((b_1, b_2)\) of a lattice \(\Lambda\).

\textbf{Ensure: } A basis \((b_1, b_2)\) as in the Wristwatch lemma.

\begin{verbatim}
repeat
  swap \(b_1 \leftrightarrow b_2\)
  \(k \leftarrow \lceil \langle b_1, b_2 \rangle / \|b_1\|^2 \rceil \)
  \(b_2 \leftarrow b_2 - kb_1\)
until \(\|b_1\| \leq \|b_2\|\)
\end{verbatim}

**Lemma**

\textit{The Lagrange reduction algorithm terminates after} \(O\left(\log \frac{\|b_1\|}{\sqrt{\det \Lambda}}\right)\) \textit{iterations.}

Proof idea: \(b_1\) 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).

$T : b_2 \leftarrow b_2 + b_1; \quad S : b_1 \leftrightarrow b_2$

Diagram: A tessellation of the plane under the action of $GL_2(\mathbb{Z})$ over $GL_2(\mathbb{R})$, showing the transformations $T$ and $S$.
Projected Sublattices

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

- Define $\pi_i$: 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 $i \leq i' \leq j' \leq j$: $(\mathbf{B}[i:j])[i':j'] = \mathbf{B}[i':j']$
- $(\mathbf{B}[i:j])^* = (\mathbf{b}_i^*, \ldots, \mathbf{b}_j^*)$
Projected Sublattices

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

- Define $\pi_i$: the projection orthogonally to $b_1, \ldots, b_{i-1}$.
- Define $\mathbf{B}_{[i:j]} = (\pi_i(b_i), \ldots, \pi_i(b_j))$
- $L_{[i:j]}$ is the projected sublattice of $L$ spanned by $\mathbf{B}_{[i:j]}$

Identities:

- For $i \leq i' \leq j' \leq j$: $(\mathbf{B}_{[i:j]})_{[i':j']} = \mathbf{B}_{[i':j']}$
- $(\mathbf{B}_{[i:j]})^* = (b_i^*, \ldots, b_j^*)$
Projected Sublattices

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

- Define $\pi_i$: the projection orthogonally to $b_1, \ldots, b_{i-1}$.
- Define $\mathbf{B}_{[i:j]} = (\pi_i(b_i), \ldots \pi_i(b_j))$
- $L_{[i:j]}$ is the projected sublattice of $L$ spanned by $\mathbf{B}_{[i:j]}$

Identities:

- For $i \leq i' \leq j' \leq j$: $(\mathbf{B}_{[i:j]})_{[i':j']} = \mathbf{B}_{[i':j']}$
- $\left(\mathbf{B}_{[i:j]}\right)^* = (b_i^*, \ldots, b_j^*)$
Everybody loves commutative diagrams

Relations between the projected sublattices:

\[
\begin{align*}
L &= L_{[1:n]} \supset L_{[1:n-1]} \supset \cdots \supset L_{[1:2]} \supset L_{[1:1]} \\
&\quad \downarrow \quad \quad \downarrow \quad \quad \downarrow \\
L_{[2:n]} &= L_{[2:n-1]} \supset \cdots \supset L_{[2:2]} \\
&\quad \downarrow \quad \quad \downarrow \\
&\quad \vdots \quad \quad \vdots \\
&\quad \downarrow \quad \quad \downarrow \\
L_{[n-1:n]} &= L_{[n-1:n-1]} \\
&\quad \downarrow \\
L_{[n:n]} \\
\downarrow
\end{align*}
\]

\[\downarrow: \text{Projection orthogonally to } b_i \text{ for some } i.\]
LLL-reduced basis

**Definition (ε-Lovasz reduced)**

Let \( \epsilon \geq 0 \). A 2-dimensional basis \( B = (b_1, b_2) \) is said \( \epsilon \)-Lovasz reduced if

\[
\|b_2^*\| \geq \left( \sqrt{3/4} + \epsilon \right) \cdot \|b_1\|.
\]

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

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

\[
\|b_{i+1}^*\| \geq \left( \sqrt{3/4} + \epsilon \right) \cdot \|b_i\|.
\]

LLL-reduced: “Gram-Schmidt norms do not decrease too fast.”
LLL-reduced basis

Definition ($\epsilon$-Lovasz reduced)

Let $\epsilon \geq 0$. A 2-dimensional basis $B = (b_1, b_2)$ is said $\epsilon$-Lovasz reduced if

$$\|b_2^*\| \geq (\sqrt{3/4} + \epsilon) \cdot \|b_1\|.$$  

Definition ($\epsilon$-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

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

$$\|b_{i+1}^*\| \geq (\sqrt{3/4} + \epsilon) \cdot \|b_i\|.$$  

LLL-reduced: “Gram-Schmidt norms do not decrease too fast.”
Properties of LLL-reduced basis

**Theorem**

Let $B = (b_1, \ldots, b_n)$ be an $\epsilon$-LLL reduced basis. Set $\Lambda = L(B)$ and $\alpha = \sqrt{4/3 + \epsilon}$. Then, we have

1. $\|b_1\| \leq \alpha^{n-1}/2 \cdot \det(\Lambda)^{1/n}$  
   *Root Hermite-factor Bound*

2. $\|b_1\| \leq \alpha^{n-1} \cdot \lambda_1(\Lambda)$  
   *Approximation factor Bound*

3. $\|b_i^*\| \leq \alpha^{n-i} \cdot \lambda_i(\Lambda)$.

4. $\|b_i\| \leq \alpha^{i-1} \cdot \|b_i^*\| \leq \alpha^{n-1} \cdot \lambda_i(\Lambda)$

5. $\prod_{i=1}^n \|b_i\| \leq \alpha^{n(n-1)/2} \cdot \det(\Lambda)$. 

Léo Ducas (CWI, Amsterdam)
LLL Algorithm

Definition ($\varepsilon$-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

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

The LLL algorithm

Require: A basis $B$ of a lattice $\Lambda$.
Ensure: An $\varepsilon$-LLL reduced basis $B$ of the same lattice.

while $\exists i, B_{[i:i+1]}$ is not $\varepsilon$-Lovasz reduced do
    Lagrange reduce it...
end while
Definition (\(\epsilon\)-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

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

The LLL algorithm

**Require:** A basis \(B\) of a lattice \(\Lambda\).

**Ensure:** An \(\epsilon\)-LLL reduced basis \(B\) of the same lattice.

\[\text{while } \exists i, B_{[i:i+1]} \text{ is not } \epsilon\text{-Lovasz reduced do}\]

\[\text{Lagrange reduce it...}\]

\[\text{end while}\]
Definition (ε-LLL reduced (Lenstra-Lenstra-Lovasz 1982))

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

The LLL algorithm

Require: A basis $B$ of a lattice $\Lambda$.

Ensure: An $\epsilon$-LLL reduced basis $B$ of the same lattice.

while $\exists i, B[i:i+1]$ is not $\epsilon$-Lovasz reduced do

Lagrange reduce it...

end while
Lemma (Efficiency of LLL)

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

Proof:

- Define a potential
  
  $P(B) = \sum (n - i) \log(\|b_i^*\|) = \sum \log(\text{vol}(L_{[1:i]}))$

- Each loop of the LLL algorithm decreases $P(B)$ by $\log(1 + \epsilon)$

- Note that for integral $B \in \mathbb{Z}^{n \times n}$, $P(B) \geq 0$.

- Concludes

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

Léo Ducas (CWI, Amsterdam)
LLL : a 2-legged ant

- Columns height : $\log(\|b_i^*\|)$
- 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 work 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 work over $\mathbb{Q}$. However, even assuming perfect computation over $\mathbb{R}$, the above proof fails for real inputs.

The above proof is incomplete

$\#\text{loops} = \text{poly}$. 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 work over $\mathbb{Q}$. However, even assuming perfect computation over $\mathbb{R}$, the above proof fails for real inputs.

The above proof is incomplete
$\#\text{loops} = poly$. 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 work over $\mathbb{Q}$. However, even assuming perfect computation over $\mathbb{R}$, the above proof fails for real inputs.

The above proof is incomplete

$\#\text{loops} = \text{poly}$. 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 its own."
  
  "the right combination."

- Construct a unimodular matrix \(U\) such that \(T_{[i:j]} \cdot U = [v, *, *, \ldots]\).
  
  Apply \(U\) (locally).
- The new \(b_i^* = v\) got shorter!
- The other \(b_{i+1}^*, \ldots, b_j^*\) will change as well
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 its own."

"the right combination."

Construct a unimodular matrix \(U\) such that \(T_{[i:j]} \cdot U = [v, *, *, \ldots]\).

Apply \(U\) (locally).

The new \(b_i^* = v\) got shorter!

The other \(b_{i+1}^*, \ldots, b_j^*\) 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 its own."
"the right combination."

- Construct a unimodular matrix $U$ such that $T_{[i:j]} \cdot U = [v, *, *, \ldots]$. Apply $U$ (locally).
- The new $b_i^* = v$ got shorter!
- The other $b_{i+1}^*, \ldots, b_j^*$ 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 satisfaction (or convergence).
$b$: Blocksize

Run the local improvements for consecutive blocks:


This is called a tour.

Repeat tours until satisfaction (or convergence).
BKZ prediction

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

Worse case bound (Minkowski)

\[
\|b_i^*\| \approx \sqrt{\frac{2n}{\pi e}} \cdot \text{vol}(L_{[i:i+b]})^{1/n} = \sqrt{\frac{2n}{\pi e}} \cdot \prod_{k=i}^{i+b} \|b_k^*\|^{1/n}
\]

Average case (for $b \geq 50$)

\[
\|b_i^*\| \approx \sqrt{\frac{n}{2\pi e}} \cdot \prod_{k=i}^{i+b} \|b_k^*\|^{1/n}
\]

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:
  
  \[\|b_1\| \approx \left(\frac{b}{2\pi e}\right)^{n/2b} \cdot \text{vol}(L)\]
Enumeration
Goal: enumerate all the lattice points in a ball of radius $r : r \mathcal{B} \cap L$. Remember the chain of lattices (by projections)

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

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

**Enumeration algorithm**

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

$$S_n = \{0\}; \quad S_{i-1} = \pi_i^{-1}(S_i) \cap r \mathcal{B}$$
The cost of the algorithm is essentially proportional to $\sum |S_i|$. 

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

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:

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