The Lenstra-Lenstra-Lovasz basis reduction algorithm for lattices

Jeremy Porter

CSCI-6101

April 4, 2011

Lattice basics

Concepts & definitions

A *lattice* in $\mathbb{R}^n$ is a free $\mathbb{Z}$-module of rank n.

Lattice basics

Concepts & definitions

A *lattice* in $\mathbb{R}^n$ is a free $\mathbb{Z}$-module of rank n.

Better yet: a lattice L is a subgroup of n-dimensional space such that

$$L = \left\{ \sum_{i=1}^{n} a_i b_i : a_i \in \mathbb{Z} \right\} = L(b_1, \ldots, b_n)$$

where the b_i are linearly independent and form the *basis* of the lattice.

Lattice basics

Concepts & definitions

A *lattice* in $\mathbb{R}^n$ is a free $\mathbb{Z}$-module of rank n.

Better yet: a lattice L is a subgroup of n-dimensional space such that

$$L = \left\{ \sum_{i=1}^{n} a_i b_i : a_i \in \mathbb{Z} \right\} = L(b_1, \ldots, b_n)$$

where the b_i are linearly independent and form the *basis* of the lattice.

In other words, L is "almost" a vector space on $\mathbb{R}^n$, except with coefficients limited to $\mathbb{Z}$.

Lattice basics

Concepts & definitions

- Lattices form tilings, diving $\mathbb{R}^n$ into infinitely many copies of its *fundamental region*

Lattice basics

Concepts & definitions

- Lattices form tilings, diving $\mathbb{R}^n$ into infinitely many copies of its *fundamental region*
- The basis of a lattice is not unique; however, the volume vol(L) of its fundamental region is independent of choice of basis

Lattice basics

Concepts & definitions

- Lattices form tilings, diving $\mathbb{R}^n$ into infinitely many copies of its *fundamental region*
- The basis of a lattice is not unique; however, the volume $\text{vol}(L)$ of its fundamental region is independent of choice of basis
- Also independent of basis is the *determinant* of the lattice, $\det(L) = \text{vol}(L)^2$

Lattice basics

Typical problems on lattices

Given a basis $b_1, \ldots, b_n$ for an n-dimensional lattice L...

Given a basis $b_1, \ldots, b_n$ for an n-dimensional lattice L...

- Closest Vector Problem (CVP): given a target $t \in \mathbb{R}^n$, find $\ell \in L$ closest to t

Given a basis $b_1, \ldots, b_n$ for an n-dimensional lattice L...

- Closest Vector Problem (CVP): given a target $t \in \mathbb{R}^n$, find $\ell \in L$ closest to t
- Shortest Vector Problem (SVP): find the $\ell \in L$ of minimum, non-zero length

Given a basis $b_1, \ldots, b_n$ for an n-dimensional lattice L...

- Closest Vector Problem (CVP): given a target $t \in \mathbb{R}^n$, find $\ell \in L$ closest to t
- Shortest Vector Problem (SVP): find the $\ell \in L$ of minimum, non-zero length
- Shortest Independent Vector Problem (SIVP): find n linearly independent $\ell_i \in L$ with the smallest $\max_i ||\ell_i||$

Lattice basics

Typical problems on lattices

- (Ajtai, 1996)
  the SVP is provably NP-Hard

Lattice basics

Typical problems on lattices

- (Ajtai, 1996)
  the SVP is provably NP-Hard
- (Aharonov & Regev, 2005)
  approximating the SVP is in NP $\cap$ co-NP

- (Ajtai, 1996)
  the SVP is provably NP-Hard
- (Aharonov & Regev, 2005)
  approximating the SVP is in NP $\cap$ co-NP
- (Ajtai & Dwork, 1997)
  the SVP is used to construct cryptosystems with worst-case/average-case equivalence

Lattice basics

Typical problems on lattices

- (Ajtai, 1996)
  the SVP is provably NP-Hard
- (Aharonov & Regev, 2005)
  approximating the SVP is in NP $\cap$ co-NP
- (Ajtai & Dwork, 1997)
  the SVP is used to construct cryptosystems with worst-case/average-case equivalence
- (Lenstra, Lenstra, & Lovasz, 1982)
  the LLL algorithm can be used as an approximation algorithm for solving the SVP within a factor of $2^{O(n)}$ in polynomial time

Gram-Schmidt orthogonalization

A review of linear algebra

Gram-Schmidt orthogonalization

A review of linear algebra

Given a basis $b_1, \ldots, b_n$, we inductively define

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*, \quad \text{where } \mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$$

Gram-Schmidt orthogonalization

A review of linear algebra

Given a basis $b_1, \ldots, b_n$, we inductively define

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*, \quad \text{where } \mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$$

So for instance:

$$b_1^* = b_1$$

Gram-Schmidt orthogonalization

A review of linear algebra

Given a basis $b_1, \ldots, b_n$, we inductively define

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*, \quad \text{where } \mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$$

So for instance:

$$b_1^* = b_1$$

$$b_2^* = b_2 - \frac{\langle b_2, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} b_1^*$$

Gram-Schmidt orthogonalization

A review of linear algebra

Given a basis $b_1, \ldots, b_n$, we inductively define

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*, \quad \text{where } \mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$$

So for instance:

$$b_1^* = b_1$$

$$b_2^* = b_2 - \frac{\langle b_2, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} b_1^*$$

$$b_3^* = b_3 - \frac{\langle b_3, b_2^* \rangle}{\langle b_2^*, b_2^* \rangle} b_2^* - \frac{\langle b_3, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} b_1^*$$

$$\vdots$$

Gram-Schmidt orthogonalization

A review of linear algebra

Given a basis $b_1, \ldots, b_n$, we inductively define

$$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{i,j} b_j^*, \quad \text{where } \mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle}$$

So for instance:

$$b_1^* = b_1$$

$$b_2^* = b_2 - \frac{\langle b_2, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} b_1^*$$

$$b_3^* = b_3 - \frac{\langle b_3, b_2^* \rangle}{\langle b_2^*, b_2^* \rangle} b_2^* - \frac{\langle b_3, b_1^* \rangle}{\langle b_1^*, b_1^* \rangle} b_1^*$$

$$\vdots$$

Orthogonal basis $\uparrow$

Hadamard's inequality

Theorem (Hadamard)

For an n-dimensional lattice L with basis vectors $b_1, \ldots, b_n$ and determinant $d(L)$,

$$d(L) \leq \prod_{i=1}^{n} |b_i|.$$

Hadamard's inequality

Theorem (Hadamard)

For an n-dimensional lattice L with basis vectors $b_1, \ldots, b_n$ and determinant $d(L)$,

$$d(L) \leq \prod_{i=1}^{n} |b_i|.$$

Corollary

If the basis is orthogonal, say $b_1^*, \ldots, b_n^*$, then this becomes the equality

$$d(L) = \prod_{i=1}^{n} |b_i^*|.$$

The LLL algorithm

Two LLL conditions

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

The LLL algorithm

Two LLL conditions

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

B) $|b_i^* + \mu_{i,i-1} b_{i-1}^*|^2 \geq \frac{3}{4}|b_{i-1}^*|^2$ for $1 < i \leq n$

The LLL algorithm

Two LLL conditions

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

B) $|b_i^* + \mu_{i,i-1} b_{i-1}^*|^2 \geq \frac{3}{4}|b_{i-1}^*|^2$ for $1 < i \leq n$

B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

The LLL algorithm

Two LLL conditions

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

B) $|b_i^* + \mu_{i,i-1} b_{i-1}^*|^2 \geq \frac{3}{4}|b_{i-1}^*|^2$ for $1 < i \leq n$

B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

$\rightarrow$ Recall that the triangle inequality implies:

$$|b_i^*|^2 + |\mu_{i,i-1} b_{i-1}^*|^2 \geq |b_i^* + \mu_{i,i-1} b_{i-1}^*|^2$$

The LLL algorithm

Two LLL conditions

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

B) $|b_i^* + \mu_{i,i-1} b_{i-1}^*|^2 \geq \frac{3}{4}|b_{i-1}^*|^2$ for $1 < i \leq n$

B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

$\rightarrow$ [Micciancio, 2010] The $\frac{3}{4}$ factor may be replaced by $\delta \in (\frac{1}{4}, 1)$

Theorem (Lenstra, Lenstra, Lovasz)

A lattice with basis vectors $b_1, \ldots, b_n$ satisfying both conditions (A) and (B) has the following properties:

- $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$
- $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, *for all non-zero vectors* $x \in L$

Theorem (Lenstra, Lenstra, Lovasz)

A lattice with basis vectors $b_1, \ldots, b_n$ satisfying both conditions (A) and (B) has the following properties:

- $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$
- $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, *for all non-zero vectors* $x \in L$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$

(B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

1. $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

1. $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$
   The LHS is the Hadamard inequality.

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

1. $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$
   The RHS is from the two LLL conditions. Taken together,

$$|b_i^*|^2 \geq \frac{1}{2}|b_{i-1}^*|^2$$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

1. $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$
   Thus, by induction

$$|b_j^*|^2 \leq 2^{i-j}|b_i^*|^2, \quad \text{for } i \geq j.$$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

1. $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$
   Thus, by induction

$$|b_j^*|^2 \leq 2^{i-j}|b_i^*|^2, \quad \text{for } i \geq j.$$

Now Gram-Schmidt gives

$$|b_i|^2 \leq |b_i^*|^2 + \sum_{j=1}^{i-1} \frac{1}{4} 2^{i-j}|b_i^*|^2$$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

1. $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$

$$|b_i|^2 \leq |b_i^*|^2 + \sum_{j=1}^{i-1} \frac{1}{4} 2^{i-j} |b_i^*|^2 \quad = \quad |b_i^*|^2 \left(1 + \frac{1}{4} \sum_{j=1}^{i-1} 2^j \right)$$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

1. $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$

$$
\begin{aligned}
|b_i|^2 \leq |b_i^*|^2 + \sum_{j=1}^{i-1} \frac{1}{4} 2^{i-j} |b_i^*|^2 &= |b_i^*|^2 \left(1 + \frac{1}{4} \sum_{j=1}^{i-1} 2^j\right) \\
&= |b_i^*|^2 \left(1 + \frac{1}{4}\left[\frac{2^i - 1}{2-1} - 1\right]\right) = |b_i^*|^2 \left(1 + \frac{1}{4}\left[2^i - 2\right]\right) \\
&\leq 2^{i-1} \cdot |b_i^*|^2
\end{aligned}
$$

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

1. $d(L) \leq \prod_{i=1}^{n} |b_i| \leq 2^{\frac{n(n-1)}{4}} \cdot d(L)$

   With $|b_i|^2 \leq 2^{i-1} \cdot |b_i^*|^2$ and $|b_i| \leq 2^{\frac{i-1}{2}} \cdot |b_i^*|$, we can write

$$\prod_{i=1}^{n} |b_i| \leq \prod_{i=1}^{n} 2^{\frac{i-1}{2}} \cdot |b_i^*|$$

$$= 2^{\sum_{i=1}^{n} \frac{i-1}{2}} \prod_{i=1}^{n} |b_i^*|$$

$$= 2^{\frac{n(n-1)}{4}} \prod_{i=1}^{n} |b_i^*|$$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

2. $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, for all non-zero vectors $x \in L$

$\rightarrow$ We just showed that $|b_j|^2 \leq 2^{i-1}|b_i^*|^2$, for $1 \leq j \leq i \leq n$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

2. $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, for all non-zero vectors $x \in L$

$\rightarrow$ We just showed that $|b_j|^2 \leq 2^{i-1}|b_i^*|^2$, for $1 \leq j \leq i \leq n$

- Write $x = \sum_{j=1}^{i} s_j b_j = \sum_{j=1}^{i} t_j b_j^*$,
  $s_j \in \mathbb{Z}$, $t_j \in \mathbb{R}$, and i the largest index having $t_i \neq 0$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

2. $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, for all non-zero vectors $x \in L$

$\rightarrow$ We just showed that $|b_j|^2 \leq 2^{i-1}|b_i^*|^2$, for $1 \leq j \leq i \leq n$

- Write $x = \sum_{j=1}^{i} s_j b_j = \sum_{j=1}^{i} t_j b_j^*$,

  $s_j \in \mathbb{Z}$, $t_j \in \mathbb{R}$, and i the largest index having $t_i \neq 0$
- The RHS equality is

$$x = t_i \cdot b_i^* + t_{i-1} \cdot b_{i-1}^* + \cdots + t_1 \cdot b_1^* = t_i \left(b_i - \sum_{k=1}^{i-1} \mu_{i,k} b_j^* \right) + t_{i-1}(\dots) + \dots$$

so that $t_i = s_i$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

2. $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, for all non-zero vectors $x \in L$

$\rightarrow$ We just showed that $|b_j|^2 \leq 2^{i-1}|b_i^*|^2$, for $1 \leq j \leq i \leq n$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

2. $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, for all non-zero vectors $x \in L$

$\rightarrow$ We just showed that $|b_j|^2 \leq 2^{i-1}|b_i^*|^2$, for $1 \leq j \leq i \leq n$

- Since $t_i \in \mathbb{Z}$, then

$$|x|^2 \geq t_i \cdot |b_i^*|^2 \geq |b_i^*|^2$$

$$\Rightarrow |b_1|^2 \leq 2^{n-1}|b_i^*|^2 \leq 2^{n-1}|x|^2$$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

2. $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, for all non-zero vectors $x \in L$

$\rightarrow$ We just showed that $|b_j|^2 \leq 2^{i-1}|b_i^*|^2$, for $1 \leq j \leq i \leq n$

- Since $t_i \in \mathbb{Z}$, then

$$|x|^2 \geq t_i \cdot |b_i^*|^2 \geq |b_i^*|^2$$

$$\Rightarrow |b_1|^2 \leq 2^{n-1}|b_i^*|^2 \leq 2^{n-1}|x|^2$$

The LLL algorithm

Proof of main theorem

(A) $|\mu_{i,j}| \leq \frac{1}{2}, \quad 1 \leq j < i \leq n$ (B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}^2)|b_{i-1}^*|^2$

2. $|b_1| \leq 2^{\frac{n-1}{2}} \cdot |x|$, for all non-zero vectors $x \in L$

$\rightarrow$ This is the $2^{O(n)}$ approximation bound we promised earlier!

The LLL algorithm

Idea behind the algorithm

- an "induction"-style algorithm, where the index k is a moving target
- begin with $k = 2$

The LLL algorithm

Idea behind the algorithm

- an "induction"-style algorithm, where the index k is a moving target
- begin with $k = 2$
- ensure that (A) is satisfied for all b_i with index $i \leq k$

$$(\mathrm{A}) \quad \mu_{i,j} \leq \frac{1}{2} \quad \text{for } 1 \leq j < i \leq k$$

The LLL algorithm

Idea behind the algorithm

- an "induction"-style algorithm, where the index k is a moving target
- begin with $k = 2$
- ensure that (A) is satisfied for all b_i with index $i \leq k$
- ensure that (B) is satisfied for b_k and b_{k-1}

$$\text{(B)} \quad |b_k^*|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$$

The LLL algorithm

Idea behind the algorithm

- an "induction"-style algorithm, where the index k is a moving target
- begin with $k = 2$
- ensure that (A) is satisfied for all b_i with index $i \leq k$
- ensure that (B) is satisfied for b_k and b_{k-1}
- increment k until we reach $k = n$

$$\text{(B)} \quad |b_k^*|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$$

The LLL algorithm

The gory details

LLL-REDUCE(L):
// Outputs basis $b_1, \ldots, b_n$ with short b_1

$k := 2$
while $k \leq n$
    A-SAT(k)
    if($|b_k^*|^2 < \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$)
        SWAP(b_k, b_{k-1})
        $k := max(2, k-1)$
    else
        $k := k+1$
end

The LLL algorithm

The gory details

LLL-REDUCE(L):
// Outputs basis $b_1, \ldots, b_n$ with short b_1

$k := 2$
while $k \leq n$
    A-SAT(k)
    if($|b_k^*|^2 < \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$)
        SWAP(b_k, b_{k-1})
        $k := max(2, k-1)$
    else
        $k := k+1$
end

The LLL algorithm

The gory details

A-SAT(k):

// Invariant: (A) is satisfied for $\mu_{k,j}$ with $\ell < j < k$

The LLL algorithm

The gory details

A-SAT(k):

// Invariant: (A) is satisfied for $\mu_{k,j}$ with $\ell < j < k$

for $\ell = k - 1$ to 1

$r := \lceil \mu_{k,\ell} \rfloor$

$b_k := b_k - r \cdot b_\ell$

$\mu_{k,\ell} := \mu_{k,\ell} - r$

end

The LLL algorithm

The gory details

A-SAT(k):
// Invariant: (A) is satisfied for $\mu_{k,j}$ with $\ell < j < k$

for $\ell = k - 1$ to 1
    $r := \lceil \mu_{k,\ell} \rfloor$
    $b_k := b_k - r \cdot b_\ell$
    $\mu_{k,\ell} := \mu_{k,\ell} - r$
end

The LLL algorithm

The gory details

A-SAT(k):
// Invariant: (A) is satisfied for $\mu_{k,j}$ with $\ell < j < k$

for $\ell = k - 1$ to 1
    $r := \lceil \mu_{k,\ell} \rfloor$
    $b_k := b_k - r \cdot b_\ell$
    $\mu_{k,\ell} := \mu_{k,\ell} - r$
end

The LLL algorithm

The gory details

A-SAT(k):
// Invariant: (A) is satisfied for $\mu_{k,j}$ with $\ell < j < k$

for $\ell = k - 1$ to 1
    $r := \lceil \mu_{k,\ell} \rfloor$
    $b_k := b_k - r \cdot b_\ell$
    $\mu_{k,\ell} := \mu_{k,\ell} - r$
end

$$\rightarrow \quad \mu_{k,\ell} = \frac{\langle b_k, b_\ell^* \rangle}{\langle b_\ell^*, b_\ell^* \rangle}$$

The LLL algorithm

The gory details

A-SAT(k):
// Invariant: (A) is satisfied for $\mu_{k,j}$ with $\ell < j < k$

for $\ell = k - 1$ to 1
    $r := \lceil \mu_{k,\ell} \rfloor$
    $b_k := b_k - r \cdot b_\ell$
    $\mu_{k,\ell} := \mu_{k,\ell} - r$
end

$$\rightarrow \quad \mu_{k,\ell} = \frac{\langle b_k, b_\ell^* \rangle}{\langle b_\ell^*, b_\ell^* \rangle}$$

$$\mu'_{k,\ell} := \frac{\langle b_k - r \cdot b_\ell, b_\ell^* \rangle}{\langle b_\ell^*, b_\ell^* \rangle} = \frac{\langle b_k, b_\ell^* \rangle - r \cdot \langle b_\ell^*, b_\ell \rangle}{\langle b_\ell^*, b_\ell^* \rangle}$$

The LLL algorithm

The gory details

A-SAT(k):
// Invariant: (A) is satisfied for $\mu_{k,j}$ with $\ell < j < k$

for $\ell = k - 1$ to 1
    $r := \lceil \mu_{k,\ell} \rfloor$
    $b_k := b_k - r \cdot b_\ell$
    $\mu_{k,\ell} := \mu_{k,\ell} - r$
end

$$\rightarrow \quad \mu_{k,\ell} = \frac{\langle b_k, b_\ell^* \rangle}{\langle b_\ell^*, b_\ell^* \rangle}$$

$$\mu'_{k,\ell} := \frac{\langle b_k - r \cdot b_\ell, b_\ell^* \rangle}{\langle b_\ell^*, b_\ell^* \rangle} = \frac{\langle b_k, b_\ell^* \rangle - r \cdot \langle b_\ell^*, b_\ell \rangle}{\langle b_\ell^*, b_\ell^* \rangle}$$

$$= \mu_{k,\ell} - r \leq \frac{1}{2} \quad \checkmark$$

The LLL algorithm

The gory details

LLL-REDUCE(L):
// Outputs basis $b_1, \ldots, b_n$ with short b_1

$k := 2$
while $k \leq n$
    A-SAT(k)
    if($|b_k^*|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$)
        $k := k + 1$
    else
        SWAP(b_k, b_{k-1})
        $k := max(2, k - 1)$
end

The LLL algorithm

The gory details

LLL-REDUCE(L):
// Outputs basis $b_1, \ldots, b_n$ with short b_1

$k := 2$
while $k \leq n$
    A-SAT(k)
    if($|b_k^*|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$)
        $k := k + 1$
    else
        SWAP(b_k, b_{k-1})
        $k := max(2, k - 1)$
end

The LLL algorithm

The gory details

LLL-REDUCE(L):
// Outputs basis $b_1, \ldots, b_n$ with short b_1

$k := 2$
while $k \leq n$
    A-SAT(k)
    if($|b_k^*|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$)
        $k := k + 1$
    else
        SWAP(b_k, b_{k-1})
        $k := max(2, k - 1)$
end

The LLL algorithm

The gory details

LLL-REDUCE(L):
// Outputs basis $b_1, \ldots, b_n$ with short b_1

$k := 2$
while $k \leq n$
    A-SAT(k)
    if($|b_k^*|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$)
        $k := k + 1$
    else
        SWAP(b_k, b_{k-1})
        $k := max(2, k - 1)$
end

- condition (A) may no longer be satisfied!

The LLL algorithm

The gory details

LLL-REDUCE(L):
// Outputs basis $b_1, \ldots, b_n$ with short b_1

$k := 2$
while $k \leq n$
    A-SAT(k)
    if($|b_k^*|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) |b_{k-1}^*|^2$)
        $k := k + 1$
    else
        SWAP(b_k, b_{k-1})
        $k := max(2, k - 1)$
end

- condition (A) may no longer be satisfied!
- backtrack to fix

The LLL algorithm

Correctness of the algorithm

Correctness is "obvious!"

The LLL algorithm

Correctness of the algorithm

Correctness is "obvious!"

Upon termination:

- Condition (A) is met
- Condition (B) is met

The LLL algorithm

Correctness of the algorithm

Correctness is "obvious!"

Upon termination:

- Condition (A) is met
- Condition (B) is met
- $\Rightarrow$ The basis is LLL-reduced, so b_1 within $2^{\frac{n-1}{2}}$ of the shortest vector

The LLL algorithm

Termination of the algorithm

- Less "obvious": whether LLL ever stops!

The LLL algorithm

Termination of the algorithm

- Less "obvious": whether LLL ever stops!
- We investigate with a potential function argument

Define

$$d_i = \det(L(b_1, \ldots, b_i))^2 = \prod_{1 \leq j \leq i} |b_j^*|^2$$

The LLL algorithm

Termination of the algorithm

- Less "obvious": whether LLL ever stops!
- We investigate with a potential function argument

Define

$$d_i = \det(L(b_1, \ldots, b_i))^2 = \prod_{1 \leq j \leq i} |b_j^*|^2$$

and

$$D = \prod_{i=1}^{n} d_i.$$

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

- The `A-SAT()` routine never affects the values of $|b_i^*|^2$
- Only the `SWAP()` routine affects the values of $|b_i^*|^2$

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

- The `A-SAT()` routine never affects the values of $|b_i^*|^2$
- Only the `SWAP()` routine affects the values of $|b_i^*|^2$
- Observe that d_i is unaffected if $i \neq (k-1)$:

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

- The `A-SAT()` routine never affects the values of $|b_i^*|^2$
- Only the `SWAP()` routine affects the values of $|b_i^*|^2$
- Observe that d_i is unaffected if $i \neq (k-1)$:
  - $i < (k-1)$: no basis vectors in $L(b_1, \ldots, b_i)$ are changed

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

- The `A-SAT()` routine never affects the values of $|b_i^*|^2$
- Only the `SWAP()` routine affects the values of $|b_i^*|^2$
- Observe that d_i is unaffected if $i \neq (k-1)$:
  - $i < (k-1)$: no basis vectors in $L(b_1, \ldots, b_i)$ are changed
  - $i > (k-1)$: $L(b_1, \ldots, b_{k-1}, b_k, \ldots, b_i) = L(b_1, \ldots, b_k, b_{k-1}, \ldots, b_i)$

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

- The `A-SAT()` routine never affects the values of $|b_i^*|^2$
- Only the `SWAP()` routine affects the values of $|b_i^*|^2$
- Observe that d_i is unaffected if $i \neq (k-1)$:
  - $i < (k-1)$: no basis vectors in $L(b_1, \ldots, b_i)$ are changed
  - $i > (k-1)$: $L(b_1, \ldots, b_{k-1}, b_k, \ldots, b_i) = L(b_1, \ldots, b_k, b_{k-1}, \ldots, b_i)$
- However when $i = (k-1)$, then

$$L(b_1, \ldots, b_{k-2}, b_{k-1})$$

becomes

$$L(b_1, \ldots, b_{k-2}, b_k)$$

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

Say D is pre-swap, D' is post-swap, then:

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

Say D is pre-swap, D' is post-swap, then:

$$\frac{D'}{D} = \frac{\left(\prod_{i=1}^{k-2} d_i \right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_k))^2}{\left(\prod_{i=1}^{k-2} d_i \right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_{k-1}))^2}$$

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

Say D is pre-swap, D' is post-swap, then:

$$\frac{D'}{D} = \frac{\left(\prod_{i=1}^{k-2} d_i\right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_k))^2}{\left(\prod_{i=1}^{k-2} d_i\right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_{k-1}))^2}$$

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

Say D is pre-swap, D' is post-swap, then:

$$\frac{D'}{D} = \frac{\left(\prod_{i=1}^{k-2} d_i\right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_k))^2}{\left(\prod_{i=1}^{k-2} d_i\right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_{k-1}))^2}$$

$$= \frac{\left(\prod_{j=1}^{k-2} |b_j^*|^2\right) \cdot |b_k^*|^2}{\prod_{j=1}^{k-1} |b_j^*|^2}$$

The LLL algorithm

Termination of the algorithm

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

Say D is pre-swap, D' is post-swap, then:

$$\frac{D'}{D} = \frac{\left(\prod_{i=1}^{k-2} d_i\right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_k))^2}{\left(\prod_{i=1}^{k-2} d_i\right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_{k-1}))^2}$$

$$= \frac{\left(\prod_{j=1}^{k-2} |b_j^*|^2\right) \cdot |b_k^*|^2}{\prod_{j=1}^{k-1} |b_j^*|^2}$$

$$D = \prod_{i=1}^{n} d_i = \prod_{i=1}^{n} \left(\prod_{j=1}^{i} |b_j^*|^2 \right)$$

Say D is pre-swap, D' is post-swap, then:

$$\begin{aligned}
\frac{D'}{D} &= \frac{\left(\prod_{i=1}^{k-2} d_i\right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_k))^2}{\left(\prod_{i=1}^{k-2} d_i\right) \cdot \det(L(b_1, \ldots, b_{k-2}, b_{k-1}))^2} \\
&= \frac{\left(\prod_{j=1}^{k-2} |b_j^*|^2\right) \cdot |b_k^*|^2}{\prod_{j=1}^{k-1} |b_j^*|^2} \\
&= \frac{|b_k^*|^2}{|b_{k-1}^*|^2}
\end{aligned}$$

The LLL algorithm

Termination of the algorithm

$$\frac{D'}{D} = \frac{|b_k^*|^2}{|b_{k-1}^*|^2}$$

Recall the LLL conditions:

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

B) $|b_i^*|^2 \geq (\frac{3}{4} - {\mu_{i,i-1}}^2)|b_{i-1}^*|^2$

The LLL algorithm

Termination of the algorithm

$$\frac{D'}{D} = \frac{|b_k^*|^2}{|b_{k-1}^*|^2}$$

Recall the LLL conditions:

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}{}^2)|b_{i-1}^*|^2$

Taken together, they imply

$$\frac{|b_i^*|^2}{|b_{i-1}^*|^2} \leq \frac{3}{4}$$

The LLL algorithm

Termination of the algorithm

$$\frac{D'}{D} = \frac{|b_k^*|^2}{|b_{k-1}^*|^2}$$

Recall the LLL conditions:

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}{}^2)|b_{i-1}^*|^2$

Taken together, they imply

$$\frac{|b_i^*|^2}{|b_{i-1}^*|^2} \leq \frac{3}{4} \quad \text{so} \quad D' \leq \frac{3}{4} \cdot D$$

$$\frac{D'}{D} = \frac{|b_k^*|^2}{|b_{k-1}^*|^2}$$

Recall the LLL conditions:

A) $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$

B) $|b_i^*|^2 \geq (\frac{3}{4} - \mu_{i,i-1}{}^2)|b_{i-1}^*|^2$

Taken together, they imply

$$\frac{|b_i^*|^2}{|b_{i-1}^*|^2} \leq \frac{3}{4} \quad \text{so} \quad D' \leq \frac{3}{4} \cdot D$$

After m swaps, this becomes

$$0 \leq D^{(m)} \leq (3/4)^m \cdot D$$

The LLL algorithm

Complexity analysis

- Complexity of LLL on an n-dimensional lattice is $O(n^6 \log^3(B))$, where $B \geq |b_i|^2$ for $1 \leq i \leq n$
- This also assumes operating on integers of bit-length $O(n \log(B))$

The LLL algorithm

Complexity analysis

- Complexity of LLL on an n-dimensional lattice is $O(n^6 \log^3(B))$, where $B \geq |b_i|^2$ for $1 \leq i \leq n$
- This also assumes operating on integers of bit-length $O(n \log(B))$
- Replacing Gram-Schmidt by *Householder orthogonalization* gives small improvements

The LLL algorithm

Complexity analysis

- Complexity of LLL on an n-dimensional lattice is $O(n^6 \log^3(B))$, where $B \geq |b_i|^2$ for $1 \leq i \leq n$
- This also assumes operating on integers of bit-length $O(n \log(B))$
- Replacing Gram-Schmidt by *Householder orthogonalization* gives small improvements
- Nguyen, Stehlè (2007) modify LLL to run in $O(n^6 \log B + n^5 \log^2 B))$, which is only quadratic w.r.t $\log B$