Hermite’s Inequality and the LLL Algorithm

Phong Nguyênn
Gram-Schmidt and Size-Reduction
Recall Gram-Schmidt

- Let $b_1, \ldots, b_n \in \mathbb{R}^m$.

- Its Gram-Schmidt Orthogonalization is $b_1^*, \ldots, b_n^* \in \mathbb{R}^m$ defined as:
  - $b_1^* = b_1$
  - For $2 \leq i \leq n$, $b_i^* = \text{projection of } b_i \text{ over } \text{span}(b_1, \ldots, b_{i-1})^\perp$
Linearly Independent Vectors
Let \( b_1, \ldots, b_n \in \mathbb{R}^m \) be linearly independent.

Then all \( b_j^* \neq 0 \).

For \( 1 \leq j < i \leq n \), let

\[ \mu_{i,j} = \frac{\langle \vec{b}_i, \vec{b}_j^* \rangle}{\| \vec{b}_j^* \|^2} . \]

Then:

\[ \vec{b}_1^* = \vec{b}_1 \]

\[ \vec{b}_i^* = \vec{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \vec{b}_j^* \]
Induction Formulas

\[
\|\vec{b}_i^*\|^2 = \|\vec{b}_i\|^2 - \sum_{j=1}^{i-1} \mu_{i,j}^2 \|\vec{b}_j^*\|^2
\]

\[
\mu_{i,j} = \frac{\langle \vec{b}_i, \vec{b}_j \rangle - \sum_{k=1}^{j-1} \mu_{j,k} \mu_{i,k} \|\vec{b}_k^*\|^2}{\|\vec{b}_j^*\|^2}
\]

- This gives an algorithm, but not necessarily efficient: we want cheap operations on reasonably-sized numbers.
Efficient Computations

- We only deal with integers, so assume that \( b_1, \ldots, b_n \in \mathbb{Z}^m \) and let \( M = \max_{1 \leq i \leq n} \|b_i\| \).

- Define the following integers:
  - \( d_0 = 1 \)
  - \( d_i = \text{Gram}(b_1, \ldots, b_i) = \|b_1^*\|^2 \times \cdots \times \|b_i^*\|^2 \)
    for \( 1 \leq i \leq m \). Thus: \( 1 \leq d_i \leq M^{2i} \)
  - Then \( \mu_{i,j}, \|b_i^*\|^2 \in \mathbb{Q} \) and \( b_j^* \in \mathbb{Q}^m \)
Lemma: Let $b_1, \ldots, b_n \in \mathbb{Z}^m$ be linearly independent. Then for all $1 \leq j < i \leq n$:

- $d_{i-1}b_i^* \in L(b_1, \ldots, b_i) \subseteq \mathbb{Z}^m$ with $\|d_{i-1}b_i^*\| \leq M^{2i-1}$

- $d_j \mu_{i,j} \in \mathbb{Z}$ with $|d_j \mu_{i,j}| \leq M^{2j}$
Proof

- $B = \mu B^*$ for some lower-triangular matrix $\mu$ with unit diagonal: $B^* = \nu B$ where $\nu = \mu^{-1}$ is lower-triangular with unit diagonal.

\[ b_i^* = b_i + \sum_{j=1}^{i-1} \nu_{i,j} b_j \]

\[ \Rightarrow \langle b_i, b_k \rangle = -\sum_{j=1}^{i-1} \nu_{i,j} \langle b_j, b_k \rangle, \text{ if } k < i \]

- Thus, $\text{Gram}(b_1, \ldots, b_i-1) \forall i, j \in \mathbb{Z}$ therefore $d_{i-1}b_i^* \in \mathbb{Z}^m$
Alternative Proof by Duality

- Let \( L = L(b_1, \ldots, b_i) \) and denote by \( L^\times \) its dual lattice. Then \( [L^\times : L] = \text{covol}(L)^2 = d_i \).

- Note that: \( b_i^*/\|b_i^*\|^2 \in L^\times \)

- Therefore \( [L^\times : L] b_i^*/\|b_i^*\|^2 \in L \),
  i.e. \( d_{i-1} b_i^* \in L(b_1, \ldots, b_i) \subseteq \mathbb{Z}^m \).
Gram-Schmidt Algorithm

- Induction formulas can be rewritten with integers, giving an efficient algorithm.

- Let \( \lambda_{i,j} = d_j \mu_{i,j} \in \mathbb{Z} \).

\[
d_i = d_{i-1} \|\vec{b}_i\|^2 - \sum_{j=1}^{i-1} \frac{\lambda_{i,j}^2}{d_j d_{j-1}}
\]

\[
\lambda_{i,j} = d_{j-1} \langle \vec{b}_i, \vec{b}_j \rangle - \sum_{k=1}^{j-1} \frac{d_{j-1} \lambda_{j,k} \lambda_{i,k}}{d_k d_{k-1}}
\]

- Could also derive \( b_i^* \), but usually not needed
Recap

- If \( b_1, \ldots, b_n \in \mathbb{Z}^m \) are linearly independent, we can compute efficiently all the integers
  \[ d_i = \text{Gram}(b_1, \ldots, b_i) = \| b_1^* \|^2 \times \ldots \times \| b_i^* \|^2 \]
  and
  \[ \lambda_{i,j} = d_j \lambda_{i,j} = d_j \frac{\langle b_i, b_j^* \rangle}{\| b_j^* \|^2} \].

\[
\overrightarrow{b_i^*} = \overrightarrow{b_i} - \sum_{j=1}^{i-1} \frac{\lambda_{i,j}}{d_j} \overrightarrow{b_j^*}
\]

\[ \| \overrightarrow{b_i^*} \|^2 = \frac{d_i}{d_{i-1}} \]
Application: Lattice Membership

- Let $b_1,\ldots,b_n \in \mathbb{Z}^m$ be linearly independent: let $L = L(b_1,\ldots,b_n)$.

- Given $t \in \mathbb{Z}^m$, decide if $t \in L$, and if so, find its integer coefficients in the decomposition $t = x_1 b_1 + \ldots + x_n b_n$. 
Lattice Membership

- Let $b_1, ..., b_n \in \mathbb{Z}^m$ be linearly independent.
- Assume that $t = x_1 b_1 + ... + x_n b_n$.
- Then $\langle t, b_n^* \rangle = x_n \langle b_n, b_n^* \rangle = x_n \|b_n^*\|^2$
- Letting $b_{n+1} = t$, then $x_n = \mu_{n+1,n}$:
  - Derive $x_n$ from Gram-Schmidt over $(b_1, ..., b_n, t)$,
  - Repeat with $t - x_n b_n$ and $L(b_1, ..., b_{n-1})$, etc.
Lattice Membership

- Let $b_1, \ldots, b_n \in \mathbb{Z}^m$ be linearly independent.
- Assume that $t = x_1 b_1 + \ldots + x_n b_n$.
- Then we can find efficiently $x_n, x_{n-1}, \ldots, x_1 \in \mathbb{Z}$ using Gram-Schmidt.
- By checking if $t = x_1 b_1 + \ldots + x_n b_n$, we can decide if $t \in \mathbb{L}$.
- Hence: we can decide lattice membership efficiently.
Application: Size-reduction

Let $b_1, \ldots, b_d \in \mathbb{Z}^m$ be linearly independent.

$B=(b_1, \ldots, b_d)$ is size-reduced if all $|\mu_{i,j}| \leq \frac{1}{2}$

Th: There is an efficient algorithm to size-reduce $B$, without changing the Gram-Schmidt vectors.
If we take an appropriate orthonormal basis, the matrix of the lattice basis becomes triangular.

\[
\begin{pmatrix}
\|\vec{b}_1^*\| & 0 & 0 & \cdots & 0 \\
\mu_{2,1}\|\vec{b}_1^*\| & \|\vec{b}_2^*\| & 0 & \cdots & 0 \\
\mu_{3,1}\|\vec{b}_1^*\| & \mu_{3,2}\|\vec{b}_2^*\| & \|\vec{b}_3^*\| & \cdots & 0 \\
\vdots & \cdots & \cdots & \cdots & \vdots \\
\mu_{d,1}\|\vec{b}_1^*\| & \mu_{d,2}\|\vec{b}_2^*\| & \cdots & \mu_{d,d-1}\|\vec{b}_{d-1}^*\| & \|\vec{b}_d^*\| \\
\end{pmatrix}
\]
Size-reduction Algorithm

- For $i = 2$ to $d$
  - For $j = i-1$ downto 1
    - Size-reduce $b_i$ with respect to $b_j$:
      make $|\mu_{i,j}| \leq 1/2$ by $b_i := b_i - \text{round}(\mu_{i,j})b_j$
    - Update all $\mu_{i',j'}$ for $j' \leq j$.
- The translation does not affect the previous $\mu_{i',j'}$ where $i' < i$, or $i' = i$ and $j' > j$. 
Linearly Dependent Vectors
Reminder

- Let $b_1, \ldots, b_n \in \mathbb{R}^m$.

- Its Gram-Schmidt Orthogonalization is $b_1^*, \ldots, b_n^* \in \mathbb{R}^m$ defined as:
  - $b_1^* = b_1$
  - For $2 \leq i \leq n$, $b_i^* =$ projection of $b_i$ over $\text{span}(b_1, \ldots, b_{i-1})^\perp$
Generalization

- Let \( b_1, \ldots, b_n \in \mathbb{R}^m \) possibly linearly dependent.
- Then not all \( b_j^* \neq 0 \).
- For \( 1 \leq j < i \leq n \), let
  \[
  \mu_{i,j} = \frac{\langle \vec{b}_i, \vec{b}_j^* \rangle}{\|\vec{b}_j^*\|^2}
  \]
  if \( b_j^* \neq 0 \), and 0 otherwise.
- Then we still have:
  \[
  \vec{b}_1^* = \vec{b}_1 \quad \text{and} \quad \vec{b}_i^* = \vec{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \vec{b}_j^*
  \]
Induction Formulas

\[
\|\vec{b}_i^*\|^2 = \|\vec{b}_i\|^2 - \sum_{j=1}^{i-1} \mu_{i,j}^2 \|\vec{b}_j^*\|^2
\]

\[
\mu_{i,j} = \frac{\langle \vec{b}_i, \vec{b}_j \rangle - \sum_{k=1}^{j-1} \mu_{j,k} \mu_{i,k} \|\vec{b}_k\|^2}{\|\vec{b}_j^*\|^2}
\]

- If \( b_j^* = 0 \), then we let \( \mu_{i,j} = 0 \).
Efficient Computations

We only deal with integers, so assume that \( b_1, \ldots, b_n \in \mathbb{Z}^m \) and let \( \|B\| = \max_{1 \leq i \leq n} \|b_i\| \).

Define the following integers:

1. \( d_0 = 1 \)
2. \( d_i = \text{Gram}(b_j) \) over \( 1 \leq j \leq i, \ b_j \neq 0 = \prod_{1 \leq j \leq i} \\text{non-zero} \ \|b_j^*\|^2 \). Still: \( 1 \leq d_i \leq \|B\|^{2i} \)

Then \( \mu_{i,j}, \|b_i^*\|^2 \in \mathbb{Q} \) and \( b_j^* \in \mathbb{Q}^m \)
Generalized Integral Gram-Schmidt

Lemma: Let \( b_1, \ldots, b_n \in \mathbb{Z}^m \). Then for all \( 1 \leq j < i \leq n \):

- \( d_{i-1}b_i^* \in L(b_1, \ldots, b_i) \subseteq \mathbb{Z}^m \) with \( ||d_{i-1}b_i^*|| \leq M^{2i-1} \)

- \( d_j \mu_{i,j} \in \mathbb{Z} \) with \( |d_j \mu_{i,j}| \leq M^{2j} \)
Recap

- If $b_1, ..., b_n \in \mathbb{Z}^m$, we can compute efficiently (polynomial time) all the generalized integers $d_i$ and $\lambda_{i,j} = d_j \mu_{i,j}$ and decide which $b_i^*$ are zero.
A Non-Trivial Lattice Algorithm
Euclid with Vectors

- If $b_1, ..., b_n \in \mathbb{Z}^m$, $L(b_1, ..., b_n)$ is a lattice: Find an efficient algorithm to find a lattice basis.

- If $n=2$ and $m=1$, this is exactly the $\gcd$ problem, so we are trying to generalize Euclid's algorithm.
Overview on Lattice Algorithms
Insight

- The most classical problem is to prove the existence of short lattice vectors.

- All known upper bounds on Hermite's constant have an algorithmic analogue:
  - Hermite's inequality: the LLL algorithm.
  - Mordell's inequality: Blockwise generalizations of LLL.
  - Mordell's proof of Minkowski's inequality: worst-case to average-case reductions for SIS and sieve algorithms [BJN14, ADRS15]
SVP Algorithms

- Poly-time approximation algorithms.
  - The LLL algorithm [1982].
  - Block generalizations by [Schnorr1987], [GHKN06], [GamaN08], [MiWa16].

- Exponential exact algorithms.
  - Exp-space sieving [AKS01, MV10, ADRS15].
Hermite’s Inequality and LLL
Hermite’s Inequality

- Hermite proved in 1850:
  \[ \gamma_d \leq \gamma_2^{d-1} = \left( \frac{4}{3} \right)^{(d-1)/2} \]

- [LLL82] finds in polynomial time a non-zero lattice vector of norm \( \leq (4/3 + \varepsilon)^{(d-1)/4} \text{vol}(L)^{1/d} \).
  It is an algorithmic version of Hermite’s inequality.
Proof of Hermite’s Inequality

- Induction over $d$: obvious for $d=1$.

- Let $b_1$ be a shortest vector of $L$, and $\pi$ the projection over $b_1 \perp$.

- Let $\pi(b_2)$ be a shortest vector of $\pi(L)$.

- We can make sure by lifting that:
  $$\|b_2\|^2 \leq \|\pi(b_2)\|^2 + \|b_1\|^2/4$$  \hspace{1cm} \text{(size-reduction)}

- On the other hand, $\|b_1\| \leq \|b_2\|$ and $\text{vol}(\pi(L)) = \text{vol}(L)/\|b_1\|$.
Hermite’s Reduction

- Hermite proved the existence of bases such that:

  \[ |\mu_{i,j}| \leq \frac{1}{2} \quad \text{and} \quad \frac{\|\mathbf{b}_i^*\|^2}{\|\mathbf{b}_{i+1}^*\|^2} \leq \frac{4}{3} \]

- Such bases approximate SVP to an exp factor:

  \[
  \|\mathbf{b}_1\| \leq \left[ \left( \frac{4}{3} \right)^{1/4} \right]^{d-1} \text{vol}(L)^{1/d} \quad \gamma_d \leq \left( \frac{4}{3} \right)^{(d-1)/2} \\
  \|\mathbf{b}_i\| \leq \left[ \left( \frac{4}{3} \right)^{1/2} \right]^{d-1} \lambda_i(L)
  \]
Graphically

- Condition 1 is over off-diagonal coeffs: size-reduction.
- Condition 2 is over diagonal coeffs.

\[
\begin{pmatrix}
\|\mathbf{b}_1\| & 0 & 0 & \cdots & 0 \\
\mu_{2,1} \|\mathbf{b}_1\| & \|\mathbf{b}_2\| & 0 & \cdots & 0 \\
\mu_{3,1} \|\mathbf{b}_1\| & \mu_{3,2} \|\mathbf{b}_2\| & \|\mathbf{b}_3\| & \cdots & 0 \\
\vdots & \cdots & \cdots & \cdots & \vdots \\
\mu_{d,1} \|\mathbf{b}_1\| & \mu_{d,2} \|\mathbf{b}_2\| & \cdots & \mu_{d,d-1} \|\mathbf{b}_{d-1}\| & \|\mathbf{b}_d\| \\
\end{pmatrix}
\]
Question

- Is the proof constructive?

- Does it build a non-zero lattice vector satisfying Hermite’s inequality:

\[ \|\vec{b}_1\| \leq \left( \frac{4}{3} \right)^{(d-1)/4} \text{vol}(L)^{1/d} \]
An Algorithmic Proof

- Let $b_1$ be a primitive vector of $L$, and $\pi$ the projection over $b_1^\perp$.
- Find recursively $\pi(b_2) \in \pi(L)$ satisfying Hermite's inequality.
- Size-reduce so that $\|b_2\|^2 \leq \|\pi(b_2)\|^2 + \|b_1\|^2 / 4$
- If $\|b_2\| < \|b_1\|$, swap($b_1, b_2$) and restart, otherwise stop.
An Algorithmic Proof

- This algorithm will terminate and output a non-zero lattice vector satisfying Hermite's inequality:

\[ \| \vec{b}_1 \| \leq \left( \frac{4}{3} \right)^{(d-1)/4} \frac{\text{vol}(L)^{1/d}}{4^{(d-1)/4}} \]

- But it may not be efficient: LLL does better by strengthening the test \( \|b_2\| < \|b_1\| \).
Computing Hermite reduction

- Hermite proved the existence of bases s.t.:
  \[ |\mu_{i,j}| \leq \frac{1}{2} \quad \text{and} \quad \frac{\|b_i^*\|^2}{\|b_{i+1}^*\|^2} \leq \frac{4}{3} \]

- By relaxing the 4/3, [LLL1982] obtained a provably polynomial-time algorithm.
LLL is an elegant divide-and-conquer based on 2-dim reduction.
Lenstra-Lenstra-Lovász

\[ \vec{b}_i^* = \vec{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \vec{b}_j^* \]

where \( \mu_{i,j} = \frac{\langle \vec{b}_i, \vec{b}_j^* \rangle}{\| \vec{b}_j^* \|^2} \)

○ A basis is LLL-reduced for \( \varepsilon > 0 \) if and only if

○ it is size-reduced \( |\mu_{i,j}| \leq \frac{1}{2} \)

○ Lovász' conditions are satisfied

\[
(1 - \varepsilon) \| \vec{b}_{i-1}^* \|^2 \leq \| \vec{b}_i^* + \mu_{i,i-1} \vec{b}_{i-1}^* \|^2
\]

\[
\Rightarrow \| \vec{b}_{i-1}^* \|^2 \leq \left( \frac{4}{3} + \varepsilon' \right) \| \vec{b}_i^* \|^2
\]
Description of the LLL Algorithm

- While the basis is not LLL-reduced
  - Size-reduce the basis
  - If Lovász’ condition does not hold for some pair \((i-1,i)\): swap \(b_{i-1}\) and \(b_i\).
Recursive LLL

- **Input:** \((b_1, b_2, \ldots, b_d)\) basis of \(L\) and \(\varepsilon > 0\).
- **LLL-reduce** \((\pi(b_2), \ldots, \pi(b_d))\) where \(\pi\) is the projection over \(b_1\).
- **Size-reduce** so that \(\|b_i\|^2 \leq \|\pi(b_i)\|^2 + \|b_1\|^2/4\)
- If \(\|b_2\| \leq (1 - \varepsilon)\|b_1\|\), swap \((b_1, b_2)\) and restart, otherwise stop.
Evolution of Gram-Schmidt

- During LLL reduction:
  - $\min_i \|b^*_i\|$ never decreases.
  - $\max_i \|b^*_i\|$ never increases.
  - Each $\text{vol}(b_1, \ldots, b_i)$ never increases.
  - The only LLL operations that modify the $b^*_i$'s are swaps.
Evolution of Gram-Schmidt

- We swap $b_{i-1}$ and $b_i$ whenever $(1 - \varepsilon) \|b^*_i\|^2 > \|b^*_i + \mu_{i,i-1}b^*_{i-1}\|^2$

- What happens to $b^*_{i-1}$ and $b^*_i$?
  - $\text{New}(b^*_{i-1}) = b^*_i + \mu_{i,i-1}b^*_{i-1}$ has norm between $\|b^*_i\|$ and $\sqrt{1 - \varepsilon} \|b^*_{i-1}\|$, hence $\geq \sqrt{1 - \varepsilon}$ shorter.
  - $\text{New}(b^*_i)$ has norm between $\|b^*_i\|/\sqrt{1 - \varepsilon}$ and $\|b^*_{i-1}\|$, hence $\geq 1/\sqrt{1 - \varepsilon}$ longer.

- $[\text{new}(\|b^*_i\|), \text{new}(\|b^*_{i-1}\|)] \subseteq [\|b^*_i\|, \|b^*_{i-1}\|]$
Why LLL is polynomial

- Consider the quantity
\[ P = \prod_{i=1}^{d} \| \vec{b}_i^* \|^2(d-i+1) \]

- If the \( b_i \)'s have integral coordinates, then \( P \) is a positive integer.
  - Size-reduction does not modify \( P \).
  - But each swap of LLL makes \( P \) decrease by a factor \( \leq 1 - \varepsilon \)

- This implies that the number of swaps is polynomially bounded.
Remarks

- We described a simple version of LLL, which is not optimized for implementation.

- We did not fully prove that LLL is polynomial time, because we did not pay attention to the size of all temporary variables.
Recap of LLL

- The LLL algorithm finds in polynomial time a basis such that:
  \[ |\mu_{i,j}| \leq \frac{1}{2} \quad \text{and} \quad \frac{\|\vec{b}_i^*\|^2}{\|\vec{b}_{i+1}^*\|^2} \leq \frac{4}{3} + \varepsilon \]
- Such bases approximate SVP to an exp factor:
  \[ \|\vec{b}_1\| \leq \left[(4/3 + \varepsilon)^{1/4}\right]^{d-1} \text{vol}(L)^{1/d} \]
  \[ \|\vec{b}_i\| \leq \left[(4/3 + \varepsilon)^{1/2}\right]^{d-1} \lambda_i(L) \]
  \[ \gamma_d \leq (4/3)^{(d-1)/2} \]
Hermite's inequality and LLL are based on two key ideas:

- Projection
- Lifting projected vectors aka size-reduction.
LLL in Practice
The Magic of LLL

- One of the main reasons behind the popularity of LLL is that it performs “much better” than what the worst-case bounds suggest, especially in low dimension.

- This is another example of worst-case vs. “average-case” and the difficulty of security estimates.
LLL: Theory vs Practice

- The approx factors $(4/3 + \varepsilon)^{(d-1)/4}$ is tight in the worst case and for uniformly random LLL bases [KiVe16].

- Experimentally, $4/3 + \varepsilon \approx 1.33$ can be replaced by a smaller constant $\approx 1.08$, for any lattice, by randomizing the input basis.

- No good explanation for this phenomenon, and no known formula for the experimental constant $\approx 1.08$. 
Illustration

Log(Hermite Factor)

theoretical worst-case bound

experimental value
Random Bases

- There is no natural probability space over the infinite set of bases.

- Folklore: generate a « random » unimodular matrix and multiply by a fixed basis. But distribution not so good.

- Better method:
  - Generate say n+20 random long lattice points
  - Extract a basis, e.g. using LLL.
Random LLL

- Surprisingly, [KiVe16] showed that most LLL bases of a random lattice have a $\|b_1\|$ close to the worst case. Note: in fixed dimension, the number of LLL bases can be bounded, independently of the lattice.

- This means that LLL biases the output distribution: it is not the uniform distribution.
Open problems

- Take a random integer lattice $L$.
- Let $B$ be the Hermite normal form of $L$, or a «random» basis from the discrete Gaussian distribution.
- Is it true that with overwhelming probability, after LLL-reducing $B$, $\|b_1\| \leq c^{d-1} \text{vol}(L)^{1/d}$ for some $c < (4/3)^{1/4}$?
- Can we guess the distribution of $\|b_1\|$ and the running time?