An Introduction to Lattices, Lattice Reduction, and Lattice-Based Cryptography

Joseph H. Silverman

Brown University

PCMI Lecture Series

July 6–10, 2020

Lecture 2. Lattice Reduction: The Practical Problem of Solving Hard Lattice Problems

The Shortest/Closest Vector Problems

Let L be a lattice of dimension n. We recall that the two most important computational problems are:

Shortest Vector Problem (SVP)

Find a shortest nonzero vector in L.

Closest Vector Problem (CVP)

Given a target vector $\boldsymbol{t} \in \mathbb{R}^n$, find a vector in L that is closest to $\boldsymbol{t}$.

- The shortest vector problem (SVP) and the closest vector problems (CVP) are clearly closely related.
- In practice, CVP seems slightly harder than SVP, but in any case, if the dimension of the lattice L is large, both SVP and CVP are very difficult to solve.
- In full generality, CVP is known to be NP-hard and SVP is NP-hard under a randomized reduction hypothesis. So we settle for less.

What is an Approximate Solution to SVP and CVP?

The

$\boldsymbol{\kappa}$-Approximate Shortest Vector Problem

abbreviated, $\boldsymbol{\kappa}$-apprSVP, is to find a vector $\boldsymbol{v} \in L$ so that

$$\|\boldsymbol{v}\| \le \kappa \min_{\boldsymbol{w} \in L \smallsetminus \mathbf{0}} \|\boldsymbol{w}\|.$$

And similarly, the

$\boldsymbol{\kappa}$-Approximate Closest Vector Problem

($\boldsymbol{\kappa}$-apprCVP) is to find a vector $\boldsymbol{v} \in L$ so that

$$\|\boldsymbol{v} - \boldsymbol{t}\| \le \kappa \min_{\boldsymbol{w} \in L} \|\boldsymbol{w} - \boldsymbol{t}\|.$$

The goal is to solve κ-apprSVP and κ-apprCVP for some $\kappa = \kappa(n)$ that's not too large.

Algorithms to Solve apprSVP and apprCVP

- The best lattice reduction methods currently known are based on the **LLL Algorithm** of Lenstra, Lenstra, and Lovász, orginally described in *Mathematische Annalen* **261** (1982), 515-534
- LLL finds moderately short lattice vectors in polynomial time. This suffices for many applications. Explicitly, LLL always solves the $2^{n/2-1}$-apprSVP problem, and often does better.
- However, finding very short (or very close) vectors, by which we mean solving the n^c-apprSVP for some small'ish c, still takes exponential time.
- Current lattice reduction algorithms such as LLL are highly sequential. Thus they are not easily distributable, although somewhat parallelizable. Also, there are no known quantum algorithms that are able to solve n^c-apprSVP in less than exponential time.

Variants and Improvements to LLL

Many methods have been proposed to improve LLL. Often they sacrifice provable polynomial time performance for improved output on most lattices.

One of the most important uses

Block Korkine–Zolotareff (BKZ) Reduction.

Roughly speaking, instead of increasing the angle between *pairs* of vectors, as is done by LLL, the BKZ method takes β-dimensional subspaces and finds reasonably orthogonal ("KZ-reduced") bases for them.

An advantage of BKZ-LLL is that as one increases the block size β, the output gets better. Indeed, taking $\beta = n$ gives a full KZ reduced basis for L, so it solves SVP. Of course, the improved output comes at a cost of increased running time.

Operating Characteristics of BKZ-LLL

For a moderately large block size β, one can prove that BKZ-LLL finds a nonzero vector $\boldsymbol{v} \in L$ satisfying

$$\|\boldsymbol{v}\| \le \left(\frac{\beta}{\pi e}\right)^{\frac{n-1}{\beta-1}} \lambda_1(L).$$

In other words, BKZ with blocksize β solves κ-apprSVP with $\kappa \approx (\beta/\pi e)^{\frac{n-1}{\beta-1}}$.

This improved output comes at a cost. The running time LLL is increased by a factor of (at least) C^β for some constant C.

Experimentally one finds this borne out: For a fixed (small'ish) constant c, the time for LLL-BKZ to find a $\boldsymbol{v} \in L$ satisfying

$$\|\boldsymbol{v}\| \le n^c \lambda_1(L) \quad \text{is exponential in } n.$$

Gram-Schmidt Orthogonalization

Before getting to the nitty-gritty of how LLL works, we start with an easier problem:

$$\boxed{\begin{array}{l}\text{How do you transform a basis } \boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \\ \text{for } L \text{ into an } \textit{orthogonal} \text{ basis of } \mathbb{R}^n?\end{array}}$$

You learned how to do that when you took linear algebra.

Gram-Schmidt Orthogonalization Algorithm

$$\begin{aligned}
\boldsymbol{v}_1^* &= \boldsymbol{v}_1 \\
\boldsymbol{v}_2^* &= \boldsymbol{v}_2 - \frac{\boldsymbol{v}_2 \cdot \boldsymbol{v}_1^*}{\|\boldsymbol{v}_1^*\|^2}\boldsymbol{v}_1^* \\
\boldsymbol{v}_3^* &= \boldsymbol{v}_3 - \frac{\boldsymbol{v}_3 \cdot \boldsymbol{v}_2^*}{\|\boldsymbol{v}_2^*\|^2}\boldsymbol{v}_2^* - \frac{\boldsymbol{v}_3 \cdot \boldsymbol{v}_1^*}{\|\boldsymbol{v}_1^*\|^2}\boldsymbol{v}_1^* \\
&\vdots \qquad\qquad\qquad \ddots \\
\boldsymbol{v}_n^* &= \boldsymbol{v}_n - \frac{\boldsymbol{v}_n \cdot \boldsymbol{v}_{n-1}^*}{\|\boldsymbol{v}_{n-1}^*\|^2}\boldsymbol{v}_{n-1}^* - \frac{\boldsymbol{v}_n \cdot \boldsymbol{v}_{n-2}^*}{\|\boldsymbol{v}_{n-2}^*\|^2}\boldsymbol{v}_{n-2}^* \cdots - \frac{\boldsymbol{v}_n \cdot \boldsymbol{v}_1^*}{\|\boldsymbol{v}_1^*\|^2}\boldsymbol{v}_1^*
\end{aligned}$$

Intuition:

$$\boldsymbol{v}_i^* = \text{Projection of } \boldsymbol{v}_i \text{ onto } \operatorname{Span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{i-1})^\perp.$$

Lattice Reduction in Dimension 2

Idea (Gauss): Alternately subtract multiples of one basis vector from the other until further improvement is not possible. So we start with

$$L = \mathrm{Span}\{\boldsymbol{v}_1, \boldsymbol{v}_2\}.$$

Step 1: If $\|\boldsymbol{v}_2\| < \|\boldsymbol{v}_1\|$, then **Swap** $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$.

Step 2: Let $m = \dfrac{\boldsymbol{v}_1 \cdot \boldsymbol{v}_2}{\|\boldsymbol{v}_1\|^2}$.

Step 3: If $\lfloor m \rceil = 0$, output $\{\boldsymbol{v}_1, \boldsymbol{v}_2\}$.

Step 4: Else replace $\boldsymbol{v}_2$ with $\boldsymbol{v}_2 - \lfloor m \rceil \boldsymbol{v}_1$, go to Step 1.

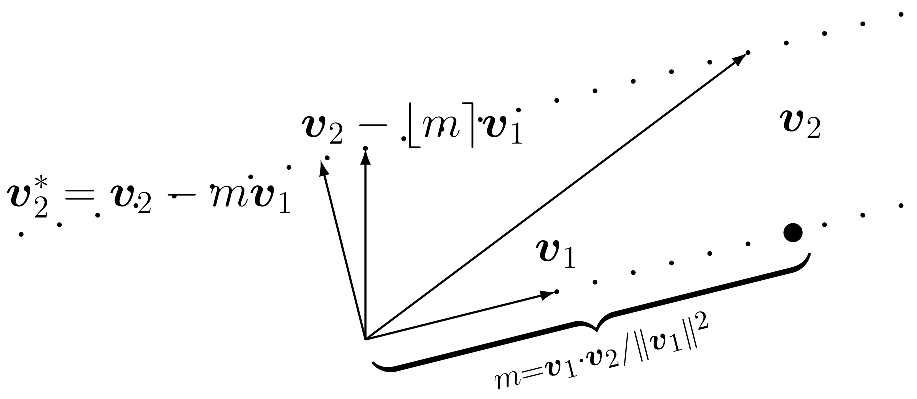
Intuition: If we remove the closest integer sign in Step 4, then $\boldsymbol{v}_2 - m\boldsymbol{v}_1$ is actually orthogonal to $\boldsymbol{v}_1$. So Step 4 does the best that it can, subject to the requirement that the new $\boldsymbol{v}_2$ must be in L.

Theorem. $\boldsymbol{v}_1$ solves SVP, and $\dfrac{\pi}{3} \le \theta(\boldsymbol{v}_1, \boldsymbol{v}_2) \le \dfrac{2\pi}{3}$.

Lattice Reduction in Dimension 2 Illustrated

Here is a picture illustrating a single reduction step when

$$\|\boldsymbol{v}_1\| < \|\boldsymbol{v}_2\|.$$


$\boldsymbol{v}_2 - \lfloor m \rceil \boldsymbol{v}_1$
$\boldsymbol{v}_2$
$\boldsymbol{v}_2^* = \boldsymbol{v}_2 - m\boldsymbol{v}_1$
$\boldsymbol{v}_1$
$m = \boldsymbol{v}_1 \cdot \boldsymbol{v}_2 / \|\boldsymbol{v}_1\|^2$

- $m = \boldsymbol{v}_1 \cdot \boldsymbol{v}_2 / \|\boldsymbol{v}_1\|^2$ is the exact factor to make

$$(\boldsymbol{v}_2 - m\boldsymbol{v}_1) \perp \boldsymbol{v}_1.$$

- In other words,

$$\boldsymbol{v}_2^* := \boldsymbol{v}_2 - m\boldsymbol{v}_1 = \text{projection of } \boldsymbol{v}_2 \text{ onto } \boldsymbol{v}_1^\perp.$$

- Rounding m to the nearest integer yields

$$\lfloor m \rceil \neq 0 \implies \boldsymbol{v}_2 - \lfloor m \rceil \boldsymbol{v}_1 \text{ is better than } \boldsymbol{v}_2.$$

The Size and Quasiorthogonality Conditions

If some coefficient in the Gram-Schmidt process satisfies

$$\frac{|\boldsymbol{v}_i \cdot \boldsymbol{v}_j^*|}{\|\boldsymbol{v}_j^*\|^2} > \frac{1}{2},$$

then we can improve the basis, i.e., make it more orthogonal, by replacing $\boldsymbol{v}_i$ with $\boldsymbol{v}_i - a\boldsymbol{v}_j$ for an appropriate $a \in \mathbb{Z}$. A basis satisfies the Size Condition if

$$\textsf{Size Condition:} \qquad \frac{|\boldsymbol{v}_i \cdot \boldsymbol{v}_j^*|}{\|\boldsymbol{v}_j^*\|^2} \le \frac{1}{2} \quad \text{for all } j < i.$$

To balance this, we want the basis vectors to be somewhat orthogonal to one another, so we impose the

$$\textsf{QuasiOrthogonality Condition:} \quad \|\boldsymbol{v}_{i+1}^*\| \ge \frac{\sqrt{3}}{2}\|\boldsymbol{v}_i^*\|.$$

The Lovász Condition

Theorem. (Hermite) Every lattice has a basis satisfying both the Size Condition and the QuasiOrthogonality Condition.

Unfortunately, the best known algorithms to find such a basis are exponential in the dimension.

So we relax the QuasiOrthogonality Condition to

Lovász Condition: $\|\boldsymbol{v}_{i+1}^*\| \geq \sqrt{\frac{3}{4} - \frac{|\boldsymbol{v}_{i+1} \cdot \boldsymbol{v}_i^*|^2}{\|\boldsymbol{v}_i^*\|^4}}\, \|\boldsymbol{v}_i^*\|.$

What a mess, right! But geometrically the Lovász Condition says that

$$\begin{aligned}&\text{Projection of } \boldsymbol{v}_{i+1} \text{ onto } \operatorname{Span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{i-1})^\perp \\ &\qquad \geq \frac{3}{4} \cdot \text{Projection of } \boldsymbol{v}_i \text{ onto } \operatorname{Span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{i-1})^\perp.\end{aligned}$$

The LLL Algorithm

Theorem. (Lenstra,Lenstra,Lovász) There is a polynomial time algorithm that finds a basis for L satisfying both the Size Condition and the Lovász Condition. Such bases are called **LLL Reduced Bases**.

```
[1]  $k = 2$
[2]  LOOP WHILE $k < n$
[3]    Replace $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ with linear combi-
       nations so the Size Condition is true
[4]    If the Lovász Condition is false
[5]       Swap $\boldsymbol{v}_k \leftrightarrow \boldsymbol{v}_{k-1}$ and set $k = k - 1$
[6]    Else
[7]       Set $k = k + 1$
[8]    If $k = n$, then basis is LLL reduced
[9]  END LOOP
```

The Basic LLL Algorithm

Operating Characteristics of LLL

- If $k = n$ in Step 8, then the basis is LLL reduced.
- Step 7 helps us by incrementing k. But there is a countervaling force because the Swapping Step (Step 5) decrements k.
- One can prove that Step 5 is executed only finitely many times and that the number of executions is bounded by a polynomial in n.
- The LLL algorithm is guaranteed to find a $\boldsymbol{v} \in L$ satisfying

  $$0 < \|\boldsymbol{v}\| \le 2^{n/2-1}\lambda_1(L).$$

  Thus LLL is a polynomial-time algorithm that solves the $2^{n/2-1}$-apprCVP problem.
- In practice, LLL generally does better than $2^{n/2-1}$. But also in practice, if n is large, then LLL will not solve the n^c-apprCVP.

The LLL Algorithm (unoptimized)

```
[1]   Input: A basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ for a lattice $L$
[2]   Set $k = 2$
[3]   Loop while $k \leq n$
[4]       Loop Down $j = k-1, k-2, \ldots, 2, 1$
[5]           Set $\boldsymbol{v}_k = \boldsymbol{v}_k - \lfloor \mu_{k,j} \rceil \boldsymbol{v}_j$   [Size Reduction]
[6]       End $j$ Loop
[7]       If $\|\boldsymbol{v}_k^*\|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) \|\boldsymbol{v}_{k-1}^*\|^2$   [Lovász Condition]
[8]           Set $k = k+1$
[9]       Else
[10]          Swap $\boldsymbol{v}_{k-1}$ and $\boldsymbol{v}_k$   [Swap Step]
[11]          Set $k = \max(k-1, 2)$
[12]      End If
[13]  End $k$ Loop
[14]  Output: The LLL reduced basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$
```

The LLL Algorithm — Notes

- At each step, $\boldsymbol{v}_1^*, \dots, \boldsymbol{v}_k^*$ is the orthogonal set of vectors in $\mathbb{R}^k$ obtained by applying the Gram–Schmidt algorithm to the current values of $\boldsymbol{v}_1, \dots, \boldsymbol{v}_k$, and
$$\mu_{i,j} = (\boldsymbol{v}_i \cdot \boldsymbol{v}_j^*)/\|\boldsymbol{v}_j^*\|^2 \quad \text{for } i > j.$$
- When the loop at Steps 4–6 ends, the vectors $\boldsymbol{v}_1, \dots, \boldsymbol{v}_k$ satisfy the Size Condition.
- At Step 7, we know that $\boldsymbol{v}_1, \dots, \boldsymbol{v}_{k-1}$ satisfy the Lovász Condition, so we check if the Lovász Condition is true when we adjoin $\boldsymbol{v}_k$ to the list. If it is, we move on to $\boldsymbol{v}_{k+1}$. If not, then $\boldsymbol{v}_k$ is "better" than $\boldsymbol{v}_{k-1}$, so we swap them.
- The big outer k-loop from Steps 3 to 13 ends when $k = n$, at which point $\boldsymbol{v}_1, \dots, \boldsymbol{v}_n$ has been size reduced and passed the Lováscz condition, to it is an LLL reduced basis.

Proof Sketch that LLL Works as Advertised

The goal is to show that the LLL algorithm terminates, and to estimate the number of steps that it takes. For ease of exposition, we assume that $L \subseteq \mathbb{Z}^n$.

Proof Idea: Define a function

$$D : \{\text{bases } \boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \text{ for } L\} \longrightarrow \mathbb{R}_{\geq 1}$$

that measures the "complexity" of a basis. Prove that

$$\begin{matrix}\text{Lovászc cond-}\\ \text{ition false}\end{matrix} \implies D\begin{bmatrix}\text{basis af-}\\ \text{ter swap}\end{bmatrix} \leq \frac{\sqrt{3}}{2} D\begin{bmatrix}\text{basis be-}\\ \text{fore swap}\end{bmatrix}$$

It follows that

$$D\begin{bmatrix}\text{basis after swap step has}\\ \text{been executed } k \text{ times}\end{bmatrix} \leq \left(\frac{\sqrt{3}}{2}\right)^k D\begin{bmatrix}\text{original}\\ \text{basis}\end{bmatrix}$$

But $D(\mathcal{B}) \geq 1$ for any basis, so the swap step is executed

$$\leq \frac{\log D[\text{original basis}]}{\log 2/\sqrt{3}} \quad \text{times}$$

Measuring the Complexity of a Basis

For a basis $\mathcal{B} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ for L, let

$$L_\ell := \operatorname{Span}\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_\ell\}.$$

Note that $L_\ell \subset \mathbb{Z}^n$ implies that $\operatorname{Det}(L_\ell)^2 \in \mathbb{N}$. Let

$$D[\mathcal{B}] := \prod_{\ell=1}^{n} \operatorname{Det}(L_\ell).$$

Let $\boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_n^*$ be the associated Gram-Schmidt vectors.

$$\operatorname{Det}(L_\ell) = \prod_{i=1}^{\ell} \|\boldsymbol{v}_i^*\|, \qquad D[\mathcal{B}] = \prod_{i=1}^{n} \|\boldsymbol{v}_i^*\|^{n+1-i}.$$

Then $D[\mathcal{B}] \geq 1$ for all bases, and

$$D[\mathcal{B}] \leq \left(\max_{1 \leq i \leq n} \log \|\boldsymbol{v}_i^*\|\right)^{\frac{n(n+1)}{2}} \leq \left(\max_{\boldsymbol{v} \in \mathcal{B}} \log \|\boldsymbol{v}\|\right)^{\frac{n(n+1)}{2}}.$$

Hence LLL terminates in at most $O(n^2 \log \|\mathcal{B}\|)$ steps.

Lováscz False Implies Swap Reduces Complexity

Assume Loáscz is false at Step $\boldsymbol{k}$:

$$\|\boldsymbol{v}_k^*\| < \sqrt{\frac{3}{4} - \mu_{k,k-1}^2} \cdot \|\boldsymbol{v}_{k-1}^*\| \leq \frac{\sqrt{3}}{2}\|\boldsymbol{v}_{k-1}^*\|.$$

Swapping $\boldsymbol{v}_k$ and $\boldsymbol{v}_{k-1}$ has the effect:

$$\begin{aligned}
\mathrm{Det}(L_{k-1}^{\text{new}}) &= \|\boldsymbol{v}_1^*\| \cdot \|\boldsymbol{v}_2^*\| \cdots \|\boldsymbol{v}_{k-2}^*\| \cdot \|\boldsymbol{v}_k^*\| \\
&= \|\boldsymbol{v}_1^*\| \cdot \|\boldsymbol{v}_2^*\| \cdots \|\boldsymbol{v}_{k-2}^*\| \cdot \|\boldsymbol{v}_{k-1}^*\| \cdot \frac{\|\boldsymbol{v}_k^*\|}{\|\boldsymbol{v}_{k-1}^*\|} \\
&= \mathrm{Det}(L_{k-1}^{\text{old}}) \cdot \frac{\|\boldsymbol{v}_k^*\|}{\|\boldsymbol{v}_{k-1}^*\|} \\
&\leq \frac{\sqrt{3}}{2}\mathrm{Det}(L_{k-1}^{\text{old}}).
\end{aligned}$$

$$\text{Hence } D[\mathcal{B}^{\text{new}}] = \left(\prod_{i \neq k-1} \mathrm{Det}(L_i^{\text{old}})\right) \cdot \mathrm{Det}(L_{k-1}^{\text{new}})$$

$$\leq \frac{\sqrt{3}}{2} \prod_{1 \leq i \leq n} \mathrm{Det}(L_i^{\text{old}}) = \frac{\sqrt{3}}{2} D[\mathcal{B}^{\text{old}}].$$

LLL Reduced Bases Are Pretty Good

Okay, now we can find an LLL reduced basis in polynomial time. How good is an LLL reduced basis?

Theorem. (LLL) Let $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ be an LLL reduced basis for L.

(a) $\|\boldsymbol{v}_1\| \le 2^{(n-1)/2}\lambda_1(L)$.

(b) $\displaystyle\prod_{i=1}^{n} \|\boldsymbol{v}_i\| \le 2^{n(n-1)/4}\,\mathrm{Det}(L)$.

Proof Sketch of (b):

The Lovász condition and $|\mu_{i,i-1}| \le \frac{1}{2}$ give

$$\|\boldsymbol{v}_i^*\|^2 \ge \left(\frac{3}{4} - \mu_{i,i-1}^2\right)\|\boldsymbol{v}_{i-1}^*\|^2 \ge \frac{1}{2}\|\boldsymbol{v}_{i-1}^*\|^2.$$

Applying repeatedly yields

$$\|\boldsymbol{v}_j^*\|^2 \le 2^{i-j}\|\boldsymbol{v}_i^*\|^2.$$

LLL Reduced Bases Are Pretty Good (continued)

We now compute

$$\begin{aligned}
\|\boldsymbol{v}_i\|^2 &= \left\| \boldsymbol{v}_i^* + \sum_{j=1}^{i-1} \mu_{i,j} \boldsymbol{v}_j^* \right\|^2 && \text{from Gram–Schmidt,} \\
&= \|\boldsymbol{v}_i^*\|^2 + \sum_{j=1}^{i-1} \mu_{i,j}^2 \|\boldsymbol{v}_j^*\|^2 && \text{since } \boldsymbol{v}_1^*, \ldots, \boldsymbol{v}_n^* \text{ are orthogonal,} \\
&\le \|\boldsymbol{v}_i^*\|^2 + \sum_{j=1}^{i-1} \frac{2^{i-j} \|\boldsymbol{v}_i^*\|^2}{4} && \text{since } |\mu_{i,j}| \le \tfrac{1}{2} \text{ and } \|\boldsymbol{v}_j^*\|^2 \le 2^{i-j} \|\boldsymbol{v}_i^*\|^2, \\
&\le 2^{i-1} \|\boldsymbol{v}_i^*\|^2.
\end{aligned}$$

Multiplying over $1 \le i \le n$ yields

$$\prod_{i=1}^{n} \|\boldsymbol{v}_i\|^2 \le \prod_{i=1}^{n} 2^{i-1} \|\boldsymbol{v}_i^*\|^2 = 2^{n(n-1)/2} \operatorname{Det}(L)^2.$$

An Introduction to Lattices, Lattice Reduction, and Lattice-Based Cryptography

Joseph H. Silverman

Brown University

PCMI Lecture Series

July 6–10, 2020