Day 4: LLL Reduction

> "Computational aspects of geometry of numbers have been revolutionized by the Lenstra-Lenstra-Lovász lattice reduction algorithm (LLL), which has led to breakthroughs in fields as diverse as computer algebra, cryptology, and algorithmic number theory. After its publication in 1982, LLL was immediately recognized as one of the most important algorithmic achievements of the twentieth century, because of its broad applicability and apparent simplicity."
>
> ∼ Phong Q. Nguyen and Brigitte Vallée (2009)

We need a better notion of "reduced" bases for dimensions above 2. For instance, let's use our example from before:

$$\mathbf{b}_1 = \begin{pmatrix} 1 \\ 0 \\ -10^{-6} \end{pmatrix}, \mathbf{b}_2 = \begin{pmatrix} 1/2 \\ \sqrt{3}/2 \\ 2 \cdot 10^{-6} \end{pmatrix}, \mathbf{b}_3 = \begin{pmatrix} -1/2 \\ \sqrt{3}/2 \\ -3 \cdot 10^{-6} \end{pmatrix}, \mathbf{b}_2 - \mathbf{b}_1 - \mathbf{b}_3 = \begin{pmatrix} 0 \\ 0 \\ 6 \cdot 10^{-6} \end{pmatrix}.$$

How could we have predicted the existence of a short vector? One way (discussed yesterday) is Minkowski's Theorem, but the theorem only predicts that it exists; it doesn't tell us how to *find* it.

Our next approach is to observe that $\mathbf{b}_3$ is *extremely close* to the plane spanned by $\mathbf{b}_1$ and $\mathbf{b}_2$. In other words, the only contribution from $\mathbf{b}_3$ in a *new* direction is very very small — and we might be able to isolate this short piece by subtracting an appropriate combination of $\mathbf{b}_1$ and $\mathbf{b}_2$.

To quantify this idea (that a vector is close to the space spanned by the previous vectors), we can use *projections.*

Definitions

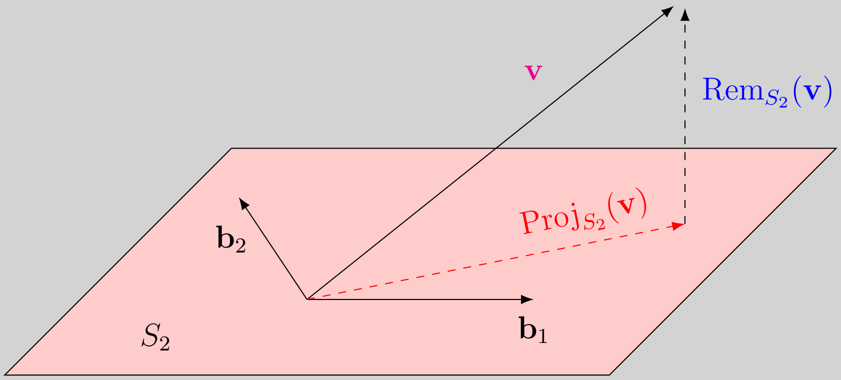
Suppose $\mathbf{b}_1, \ldots, \mathbf{b}_n$ is a basis for $\mathbb{R}^n$. For each i from 1 to n, let S_i denote the **subspace spanned by the first i vectors**:

$$S_i = \{r_1\mathbf{b}_1 + \ldots + r_i\mathbf{b}_i \mid r_1, \ldots, r_i \in \mathbb{R}\}.$$

Given any vector $\mathbf{v} \in \mathbb{R}^n$, and any i, $\mathbf{v}$ can be decomposed in a unique way as

$$\mathbf{v} = \mathrm{Proj}_{S_i}(\mathbf{v}) + \mathrm{Rem}_{S_i}(\mathbf{v}),$$

where $\mathrm{Proj}_{S_i}(\mathbf{v})$ (the **projection** of $\mathbf{v}$ onto S_i) is contained in S_i, and $\mathrm{Rem}_{S_i}(\mathbf{v})$ (the **remainder** of projecting $\mathbf{v}$ onto S_i) is orthogonal to every vector in S_i. The functions Proj_{S_i} and Rem_{S_i} are linear maps.


$\mathbf{v}$
$\mathrm{Rem}_{S_2}(\mathbf{v})$
$\mathrm{Proj}_{S_2}(\mathbf{v})$
$\mathbf{b}_2$
$\mathbf{b}_1$
S_2

(We also set $S_0 = \{\mathbf{0}\}$, so $\mathrm{Proj}_{S_0}(\mathbf{v}) = \mathbf{0}$ and $\mathrm{Rem}_{S_0}(\mathbf{v}) = \mathbf{v}$ for all $\mathbf{v}$.)

The remainder function $\mathrm{Rem}_{S_i}(\mathbf{v})$ tells us what is *new* about the vector $\mathbf{v}$; it describes what $\mathbf{v}$ is contributing to the world that hasn't already been done by vectors in S_i. For the proof-type problems in this handout, you don't need to know how to compute Proj and Rem; you only need the properties listed above. For the computation-type problems, you can use **THIS LINK** to compute Proj and Rem for you.

Let

$$\mathbf{b}_k^* = \mathrm{Rem}_{S_{k-1}}(\mathbf{b}_k)$$

denote the "new contribution" from $\mathbf{b}_k$ (the remainder after projecting onto all the previous basis vectors). The vectors $\mathbf{b}_k^*$ will usually *not* be lattice vectors (except for $\mathbf{b}_1^*$), but they can be used to find a lower bound on the length of the shortest vector in a lattice!

Exploration 1

Given a lattice vector $\mathbf{v} = a_1\mathbf{b}_1 + \cdots + a_n\mathbf{b}_n$, let a_j be the last nonzero coefficient. What is the remainder after projecting $\mathbf{v}$ onto S_{j-1}? Use this to prove Proposition 2.

Proposition 2

Let a lattice $\mathcal{L}$ be generated by $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$. Then every nonzero vector in $\mathcal{L}$ is longer than the minimum value of $\|\mathbf{b}_k^*\|$ over $k \in \{1, \ldots, n\}$.

Exploration 3

Using the basis $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$ from the start of this sheet, compute $\mathbf{b}_1^*$, $\mathbf{b}_2^*$, and $\mathbf{b}_3^*$. Use Proposition 2 to find a lower bound for the length of the shortest nonzero lattice vector.

LLL-Reduced Bases

Remember this definition?

Definition

We will say that a 2-dimensional basis $\{\mathbf{u}, \mathbf{v}\}$ is **reduced** if $\|\mathbf{u}\| \leq \|\mathbf{v}\|$ and $\|\mathbf{v}\| \leq \|\mathbf{v} + n\mathbf{u}\|$ for all integers n.

For higher dimensions, instead of comparing pairs of basis vectors, we will will compare the "new contributions" of basis vectors. That is, we will be comparing the *remainders* of basis vectors $\mathbf{b}_i$ and $\mathbf{b}_j$, after projection onto the vectors that come before both i and j.

Definition

An n-dimensional basis $\{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ is **LLL-reduced** (with parameter $\gamma = 1$) if the following are true for all $k = 2, \ldots, n$:

- $\|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_{k-1})\| \leq \|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_k)\|$ ("$\mathbf{b}_k$ has *non-decreasing remainder*"),[a] and
- $\|\mathrm{Rem}_{S_{i-1}}(\mathbf{b}_k)\| \leq \|\mathrm{Rem}_{S_{i-1}}(\mathbf{b}_k + n\mathbf{b}_i)\|$ for all $1 \leq i < k$ and all $n \in \mathbb{Z}$ ("the pair $\{\mathbf{b}_i, \mathbf{b}_k\}$ is *size-reduced*").

[a] Careful: the first term is $\mathbf{b}_{k-1}^*$, but the second term is not $\mathbf{b}_k^*$. We're only projecting away the first $k-2$ vectors, not the first $k-1$.

Exploration 4

Is the basis $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ from the beginning of this sheet LLL-reduced? Why or why not?

Exploration 5

If a basis $\mathbf{b}_1, \ldots, \mathbf{b}_n$ is LLL-reduced, prove that $\frac{\sqrt{3}}{2}\|\mathbf{b}^*_{k-1}\| \le \|\mathbf{b}^*_k\|$. Use this to prove Proposition 6.
(Hint: Plot $\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_{k-1})$ and $\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_k)$ in a 2-D plane. Where is $\mathbf{b}^*_k$?)

Proposition 6

Suppose $\mathbf{b}_1, \ldots, \mathbf{b}_n$ is an LLL-reduced basis for a lattice $\mathcal{L}$. If λ_1 denotes the length of the shortest nonzero vector of $\mathcal{L}$, then $\|\mathbf{b}_1\| \le \left(\frac{2}{\sqrt{3}}\right)^{n-1} \lambda_1$.

For example, in three dimensions, this tells us that that $\mathbf{b}_1$ will be no more than 1.54 times the length of the shortest vector. So unlike the basis $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$ we've been working with, an LLL-reduced basis will not have any "surprise" vectors that are extremely short!

The Reduction Algorithm

The basic idea will be the same as in the 2-D case. We have two operations:

1. "Swap:" If $\|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_{k-1})\| > \|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_k)\|$, swap $\mathbf{b}_{k-1}$ and $\mathbf{b}_k$.
2. "Slide:" for $i < k$, add multiples of $\mathbf{b}_i$ to $\mathbf{b}_k$ until $\{\mathbf{b}_i, \mathbf{b}_k\}$ is size-reduced.

We can apply these operations to try and achieve the LLL-reduced conditions. It's not clear, though, what order we should apply these steps in. Our attempts to fix one condition may break another!

Exploration 7

Suppose we start with the following basis:

$$\mathbf{u}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \mathbf{u}_2 = \begin{pmatrix} 1/4 \\ 1 \\ 0 \end{pmatrix}, \mathbf{u}_3 = \begin{pmatrix} 0 \\ 5 \\ 1/2 \end{pmatrix}.$$

Show that $\{\mathbf{u}_1, \mathbf{u}_2\}$ and $\{\mathbf{u}_1, \mathbf{u}_3\}$ are size-reduced, but $\{\mathbf{u}_2, \mathbf{u}_3\}$ is not. Now slide $\mathbf{u}_3$ along $\mathbf{u}_2$ (that is, replace $\mathbf{u}_3$ with $\mathbf{u}_3' := \mathbf{u}_3 + n\mathbf{u}_2$ so that $\{\mathbf{u}_2, \mathbf{u}_3'\}$ is size-reduced). Is $\{\mathbf{u}_1, \mathbf{u}_3'\}$ size-reduced?

Exploration 8

Following up with the same example, now slide $\mathbf{u}_3'$ along $\mathbf{u}_1$ (replace $\mathbf{u}_3'$ with $\mathbf{u}_3'' := \mathbf{u}_3' + n\mathbf{u}_1$ for some n so that $\{\mathbf{u}_1, \mathbf{u}_3''\}$ is size-reduced). Is $\{\mathbf{u}_2, \mathbf{u}_3''\}$ size-reduced?

Exploration 9

Show that if you work *backwards* — that is, start by sliding $\mathbf{b}_k$ along $\mathbf{b}_{k-1}$, then along $\mathbf{b}_{k-2}$, all the way down to $\mathbf{b}_1$ — then all pairs $\{\mathbf{b}_i, \mathbf{b}_k\}$ for $i < k$ will be size-reduced. (Hint: If $j < i$, and you add a multiple of $\mathbf{b}_j$ to $\mathbf{b}_k$, what is the effect on $\mathrm{Rem}_{S_{i-1}}(\mathbf{b}_k)$?)

Here is a summary of the LLL algorithm. Start with $k = 2$, and do the following:

1. Slide $\mathbf{b}_k$ along $\mathbf{b}_{k-1}$, then along $\mathbf{b}_{k-2}$, and so on down to $\mathbf{b}_1$, until the pairs $\{\mathbf{b}_i, \mathbf{b}_k\}$ are size-reduced for all $i < k$.
2. If $\|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_{k-1})\| > \|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_k)\|$, swap $\mathbf{b}_{k-1}$ and $\mathbf{b}_k$. Replace k with $k-1$ (unless $k = 2$, in which case leave it at $k = 2$) and go back to step 1.
3. If $\|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_{k-1})\| \leq \|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_k)\|$, replace k with $k+1$ (unless $k = n$, in which case end the algorithm) and go back to to step 1.

Exploration 10

Continue applying the LLL operations to the basis $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$ until you get an LLL-reduced basis.

Exploration 11

Consider our original example $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$. Does LLL reduction find the short vector?

Exploration 12 (Optional)

Apply LLL reduction to the basis from Day 2, Exploration 12. What is a quadratic polynomial that $\alpha \approx 0.4708709$ may be a root of? (LLL reduction might take a little bit longer in this case; doing some coding to automate steps may be helpful.)

Optional Exploration: Polynomial-Time LLL

We've defined an algorithm for LLL reduction. But does it actually work in practice? Unfortunately, the way we've described it, the algorithm will often be far too slow to be useful. A major breakthrough was to realize that including an "approximation factor" γ can drastically speed up the algorithm.

Definition

An n-dimensional basis $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ is **LLL-reduced** with parameter[a] $\gamma \in (\frac{1}{2}, 1]$ if the following are true for all $k = 2, \dots, n$:

- $\gamma \|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_{k-1})\| \leq \|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_k)\|$ ("the pair $\{\mathbf{b}_{k-1}, \mathbf{b}_k\}$ satisfies the *Lovász condition*"), and
- $\|\mathrm{Rem}_{S_{i-1}}(\mathbf{b}_k)\| \leq \|\mathrm{Rem}_{S_{i-1}}(\mathbf{b}_k) + n\mathrm{Rem}_{S_{i-1}}(\mathbf{b}_i)\|$ for all $1 \leq i < k$ and all $n \in \mathbb{Z}$ (same as before; "the pair $\{\mathbf{b}_i, \mathbf{b}_k\}$ is *size-reduced*").

[a] Most sources use $\delta = \gamma^2$ instead of γ, which results in more square roots.

Exploration 13

Prove Proposition 14, which is a modification of Proposition 6 to allow for the γ factor.

Proposition 14

Suppose $\mathbf{b}_1, \dots, \mathbf{b}_n$ is an LLL-reduced basis (with parameter γ) for a lattice $\mathcal{L}$. If λ_1 denotes the length of the shortest nonzero vector of $\mathcal{L}$, then $\|\mathbf{b}_1\| \leq \left(\gamma^2 - \frac{1}{4}\right)^{-(n-1)/2} \lambda_1$.

The LLL reduction algorithm is modified in a similar way: we swap two basis vectors if they don't satisfy the Lovász condition, that is, if $\gamma \|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_{k-1})\| > \|\mathrm{Rem}_{S_{k-2}}(\mathbf{b}_k)\|$. This means that we are a bit more lenient than before: even if $\mathbf{b}_k$ has shorter remainder $\mathbf{b}_{k-1}$, as long as it's not too much shorter, we leave it alone.

If $\gamma < 1$, then we can find an upper bound on the number of steps needed to run LLL reduction:

Theorem 15

Suppose $\mathbf{b}_1, \dots, \mathbf{b}_n$ is a basis for a lattice $\mathcal{L}$ which is contained in $\mathbb{Z}^n$, and let $m = \max_i \|\mathbf{b}_i\|$. Then the LLL algorithm with parameter $\gamma < 1$ will find a basis that is LLL-reduced with parameter γ in at most

$$\frac{n^2+n}{2}\left(\frac{\log m}{\log(1/\gamma)}\right) \text{ swap steps and } \frac{n^3-n}{2}\left(\frac{\log m}{\log(1/\gamma)}\right) \text{ slide steps.}$$

While this proposition only applies to lattices in $\mathbb{Z}^n$, we can fairly easily apply it to lattices in $\mathbb{Q}^n$ by scaling the lattice in order to clear denominators. (We can also apply it to lattices with irrational entries by taking rational approximations, but we need to be careful to ensure that the approximations are close enough; small errors can accumulate quickly.)

The main idea behind the proof is to consider the following quantities:

$$\begin{aligned}
D_1 &:= \|\mathbf{b}_1^*\| \\
D_2 &:= \|\mathbf{b}_1^*\|\|\mathbf{b}_2^*\| \\
D_3 &:= \|\mathbf{b}_1^*\|\|\mathbf{b}_2^*\|\|\mathbf{b}_3^*\| \\
&\ \,\vdots \\
D_n &:= \|\mathbf{b}_1^*\| \cdots \|\mathbf{b}_n^*\|, \\
D &:= D_1 D_2 \cdots D_n.
\end{aligned}$$

As we update the basis $\{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ using the LLL algorithm, these quantities will change. Specifically, we can show the following:

- "Sliding" steps do not change any of the vectors $\mathbf{b}_i^*$, so they leave all the D_i unchanged.
- Suppose $\{\mathbf{b}_{k-1}, \mathbf{b}_k\}$ is size-reduced but does not satisfy the Lovász condition. If we swap $\mathbf{b}_{k-1}$ and $\mathbf{b}_k$, then all the D_i are unchanged besides D_{k-1}, and the new value of D_{k-1} is at most γ times the old value.

This implies that D shrinks at every swap step, by a factor at most γ, so that

$$(\text{final } D) \leq \gamma^{\#\text{ swap steps}}(\text{starting } D).$$

This can be rearranged to give an upper bound for the number of swap steps, and there are at most $n-1$ slide steps for every swap step. All that remains is to compute an upper bound for the starting value of D and a lower bound for the final value of D.

Exploration 16

Fill in the details in the proof sketched above. (You should have all the tools you need, except for one fact which you are free to assume: given $1 \leq i \leq n$, and i vectors $\mathbf{b}_1, \dots, \mathbf{b}_i \in \mathbb{Z}^n$, the i-dimensional volume of the parallelotope generated by $\mathbf{b}_1, \dots, \mathbf{b}_i$ is the square root of an integer.[a] This will be relevant when interpreting what the values D_i represent.)

[a] Wondering why this is true? Look up the "Gram determinant."