Lattice basics

Shahed Sharif

November 7, 2019

1 SVP and CVP

We will assume that L is a rank n lattice in $\mathbb{R}^n$. We equip $\mathbb{R}^n$ with the usual inner product. Recall the two fundamental lattice problems:

Problem 1 (SVP)**.** *The* Shortest Vector Problem *is the problem of finding the shortest nonzero $v \in L$. We say the length of such a vector is $\lambda_1(L)$. Given $\gamma \geq 1$, the γ-SVP is the problem of finding nonzero $v \in L$ such that $\|v\| \leq \gamma \cdot \lambda_1(L)$.*

Problem 2 (CVP)**.** *The* Closest Vector Problem *is the problem of, given $w \in \mathbb{R}^n$, finding $v \in L$ which minimizes $\|w - v\|$. Let v_0 be such a vector. Given $\gamma \geq 1$, the γ-CVP is the problem of finding $v \in L$ such that $\|w-v\| \leq \gamma \cdot \|w-v_0\|$.*

The main method for attacking these problems is *lattice reduction*. We assume that L is given by a basis $v_1, \ldots, v_n$. A lattice reduction algorithm takes this basis as input and outputs a better basis. Informally, "better" means more useful for solving the above lattice problems. The ideal situation is if we can find an orthogonal basis. Of course, this may not be possible, so instead we wish to find as close to an orthogonal basis as possible.

2 The Hermite bound

The *Hermite bound* gives an upper bound for $\lambda_1(L)$. The idea is to apply Minkowski's Convex Body Theorem to the lattice L with a convex body consisting of a closed ball of sufficiently large radius. Minkowski's Theorem guarantees that such a ball contains a nonzero lattice point, whose length is bounded by the radius of the ball. For $n >> 0$, the volume of the n-dimensional unit ball is approximately $(\frac{2\pi e}{n})^{n/2}$. From there, it is not hard to show that

$$\lambda_1(L) \leq \sqrt{\frac{2\pi e}{n}}\,\mathrm{covol}(L)^{1/n}.$$

If we instead choose a unit ball whose volume *equals* $\mathrm{covol}(L)$, we are not guaranteed to find a nonzero lattice point, but as we vary our lattice, we expect to find one on average. The *Gaussian heuristic* quantifies this expectation; we compute the radius of the ball with volume equal to covol(L), and obtain the expectation

$$\sigma(L) = \sqrt{\frac{2n}{\pi e}}\,\mathrm{covol}(L)^{1/n}.$$

The use of the Gaussian heuristic is that it gives a benchmark for difficulty of the SVP. If we know that λ_1 is much shorter than $\sigma(L)$, then L has an "unusually" short vector, so SVP algorithms should perform better. Conversely, if $\lambda_1 > \sigma(L)$, we expect SVP algorithms to perform poorly.

3 Lagrange-Gauss reduction

Let $n = 2$. We say a basis b_1, b_2 for L is *Lagrange-Gauss reduced* if

$$\|b_1\| \leq \|b_2\| \leq \|b_2 + qb_1\|$$

for all $q \in \mathbb{Z}$.

Recall the Gram-Schmidt algorithm. Given input $b_1, \ldots, b_n$, write $b_1^*, \ldots, b_n^*$ for the output. While Gram-Schmidt returns a good set of vectors, the new vectors will not generally be a basis for the lattice generated by the b_i. Let

$$\mu_{ij} = \frac{\langle b_j^*, b_i \rangle}{\langle b_j^*, b_j^* \rangle}.$$

We call *integer Gram-Schmidt* the Gram-Schmidt process, but with $\lfloor \mu_{ij} \rceil$ in place of μ_{ij}. Thus, the vectors produced by integer Gram-Schmidt will remain a basis for the given lattice. Of course, the output will not be orthogonal.

To produce a Lagrange-Gauss reduced basis, we start with a basis b_1, b_2, and proceed as follows.

1. Apply integer Gram-Schmidt to b_1, b_2, and replace with the new vectors.
2. If $\|b_2\| < \|b_1\|$, swap them, then return to step 1.
3. Output b_1, b_2.

The motivation behind step 1 should be clear. We certainly need the second step to produced a reduced basis. The geometric justification is as follows. If the fundamental domain defined by b_1, b_2 is extremely skew, integer Gram-Schmidt replaces it with one which is less skew; that is, closer to a rectangle. However, if b_2 is much shorter than b_1, this "unskewing" is very coarse. By swapping the two, we are able to obtain a better basis.

To see that the algorithm outputs a Lagrange-Gauss reduced basis, observe that $\|b_2 + tb_1\|^2$ is quadratic in t, and is minimized when $t = \mu_{21}$. Therefore if $t \in \mathbb{Z}$, it is minimized when $t = \lfloor \mu_{21} \rceil$. To see that the algorithm terminates, observe that b_1, b_2 always form a basis, and that b_1 gets shorter at every iteration.

Proposition 3.1. *Let b_1, b_2 be a Lagrange-Gauss reduced basis for L. Then $\lambda_1 = \|b_1\|$. Furthermore, $r = \|b_2\|$ is the smallest number for which*

$$\dim(\mathrm{Span}\{v \in L \mid \|v\| \le r\}) = 2.$$

Thus, in a sense b_1 and b_2 are the shortest vectors in L.

4 LLL

LLL is the n-dimensional generalization of Lagrange-Gauss. Recall the vectors b_i^* and scalars μ_{ij} from the Gram-Schmidt process. Fix a parameter δ with $\frac{1}{4} < \delta < 1$. We say a basis $b_1, \ldots, b_n$ for our lattice L is δ *LLL-reduced* if

1. (Size reduced) $|\mu_{ij}| \le \frac{1}{2}$ whenever $1 \le j < i \le n$
2. (Lovàsz condition) $\|b_i^* + \mu_{i,i-1} b_{i-1}^*\|^2 \ge \delta \|b_{i-1}^*\|^2$

Lenstra-Lenstra-Lovàsz set $\delta = 3/4$ as a default value, and that choice is now considered the conventional one. The greater the value of δ, the better the basis, at a cost in computational time.

The first condition makes sure that the basis is fairly close to orthogonal. For the second condition, we compare to the Lagrange-Gauss condition. If we translated that condition directly, we would get something like

$$\|b_i + q b_{i-1}\|^2 \ge \|b_{i-1}\|^2$$

for all $q \in \mathbb{Z}$. The replacement of q with $\mu_{i,i-1}$ is easy to justify: if we allow q to be real, then the left-hand side is minimized when $q = \mu_{i,i-1}$. The replacements of b_i with b_i^*, as well as the introduction of δ, are technical changes that do not greatly impact the quality of the resulting basis, but do permit a polynomial-time algorithm to find an LLL-reduced basis.

Such an algorithm is given as follows. First, we compute all values b_i^* and μ_{ij} as above. Then we proceed:

1. Initialize a counter $k = 2$.
2. Apply integer Gram-Schmidt to b_k. (This should be done first with b_{k-1}, then b_{k-2}, etc. Note that order matters for integer Gram-Schmidt.)
3. Check the Lovàsz condition for b_k.
   (a) If it passes, go to the next step.
   (b) If it fails,
      - swap b_k and b_{k-1},
      - recalculate all b_i^* and μ_{ij},
      - decrement k, and
      - go back to step 2.

4. Increment k. If $k \leq n$, go back to step 2. Otherwise, output the b_i.

Just as with Lagrange-Gauss reduction, the output is certainly LLL-reduced. Termination is trickier to prove, but the idea is similar: find a quantity which decreases with each iteration, but is also bounded below.

An LLL-reduced basis has the following nice properties:

Theorem 4.1. *Suppose $b_1, \ldots, b_n$ is an LLL-reduced basis with $\delta = 3/4$ for the lattice L. Then $2^{(1-i)/2}\lambda_i(L) \leq \|b_i\| \leq 2^{(n-1)/2}\lambda_i(L)$ for all i.*

Proposition 4.2. *Let $b_1, \ldots, b_n$ be a δ LLL-reduced basis. Then*

$$\|b_1\| \leq \left(\frac{2}{\sqrt{4\delta - 1}}\right)^{n-1} \lambda_1(L).$$

Thus we can use LLL to solve γ-SVP when $\gamma \geq (2/\sqrt{3})^{n-1}$.

Proposition 4.3. *Let $b_1, \ldots, b_n$ be the input to LLL for fixed parameter δ. Suppose $X \geq \|b_i\|$ for all b_i. Then the running time of LLL is polynomial in n and $\log X$.*

The dependence on δ is at most multiplicative in $-\frac{1}{\log \delta}$. This looks bad for δ close to 1, but if one chooses (for example)

$$\delta = \frac{1}{4} + \left(\frac{3}{4}\right)^{n/(n-1)}$$

then the running time is still polynomial in n.

5 Babai's CVP algorithms

5.1 Rounding algorithm

The algorithm is as follows: we give as input a basis $b_1, \ldots, b_n$ for L and $w \in \mathbb{R}^n$. We wish to output $v \in L$ which is close to w.

1. Compute $\alpha_i \in \mathbb{R}$ such that $w = \alpha_1 b_1 + \cdots + \alpha_n b_n$.
2. Set $v = \lfloor \alpha_1 \rceil b_1 + \cdots + \lfloor \alpha_n \rceil b_n$.

It is not hard to see that if the basis is orthogonal, then this algorithm gives an exact answer to CVP. However, if the basis is skew, then the output v can be relatively far from w.

Theorem 5.1. *If $b_1, \cdots, b_n$ is LLL-reduced with $\delta = 3/4$, $w \in \mathbb{R}^n$, v is the output of the rounding algorithm, and v_0 is the actual solution to CVP, then*

$$\|w - v\| \leq (1 + 2n(9/2)^{n/2})\|w - v_0\|.$$

In other words, the rounding algorithm solves γ-CVP with $\gamma = (1+2n(9/2)^{n/2})$.

Babai also introduced a better algorithm, called the *nearest plane* algorithm. It efficiently solves the γ-CVP problem with $\gamma = 2^{n/2}$. Briefly, the algorithm works recursively on $\dim(L)$. Let U be the span of $b_1, \ldots, b_{n-1}$. The algorithm first finds $v_1 \in L$ which minimizes the distance from w to $v_1 + U$. Then we replace w with the projection w_1 of $w - v_1$ to U. Let $L_1 = L \cap U$. We now repeat with L_1, w_1 to obtain v_2, etc. Our final output is $\sum v_i$.