

## The LLL Algorithm

---

### Motivation

The rows of the following matrices form bases for lattices in  $\mathbb{R}^3$ :

$$X = \begin{bmatrix} 4 & 5 & 1 \\ 4 & 8 & 2 \\ 6 & 2 & 6 \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & 3 & 1 \\ 4 & 1 & -1 \\ 2 & -3 & 5 \end{bmatrix}.$$

The rows of  $X$  and the rows of  $Y$  actually span the *same* lattice. Intuitively, the rows of  $X$  seem to be a “worse” basis for  $L$  than those of  $Y$ . Here we make precise the notion of a “nice” basis and introduce a polynomial-time algorithm that transforms a “bad” basis into a “good” one.

### Basis Reduction and the LLL Algorithm [1], [2]

A basis is “nice” if its vectors are short and orthogonal to one another. The Gram-Schmidt process transforms a given basis into an orthogonal basis, but when working in a lattice  $L$ , this Gram-Schmidt basis need not live in  $L$ .

**Definition 1.** Let  $x_1, \dots, x_n$  be an ordered basis for a lattice  $L$  in  $\mathbb{R}^n$ , and let  $x_1^*, \dots, x_n^*$  be its Gram-Schmidt orthogonalization (GSO):

$$x_i^* = x_i - \sum_{j=1}^{i-1} \mu_{ij} x_j^*, \quad \text{where } \mu_{ij} = \frac{x_i \cdot x_j^*}{x_j^* \cdot x_j^*}, \text{ and } x_1^* = x_1.$$

Let  $\alpha$  be a real number with  $\frac{1}{4} < \alpha < 1$ . We say that the basis  $x_1, \dots, x_n$  is  **$\alpha$ -reduced** if it satisfies

(I) (size condition)  $|\mu_{ij}| \leq \frac{1}{2}$  for all  $1 \leq j < i \leq n$ ,

(II) (Lovász condition)  $|x_i^*|^2 \geq (\alpha - \mu_{i,i-1}^2) |x_{i-1}^*|^2$  for  $2 \leq i \leq n$ .

In the Gram-Schmidt process we build  $x_i^*$ , the projection of  $x_i$  onto  $\text{span}\{x_1^*, \dots, x_{i-1}^*\}^\perp$ , by subtracting each  $\mu_{ij} x_j^*$  from  $x_i$ . Since  $\mu_{ij}$  need not be an integer, this vector generally won't be an element of  $L$ . If we instead subtract off the integer multiple of  $x_j$  closest to  $\mu_{ij}$  then we stay in  $L$  and end up nearly orthogonal to  $x_j$ . Condition (I) then says that the closest integer to  $\mu_{ij}$  is zero:  $x_i$  is already nearly orthogonal to  $x_j$  for each  $j$ .

Condition (II) states that while the GSO vectors may get shorter, they do not decrease in length too quickly. In particular, if  $\beta = \frac{1}{\alpha-1/4}$  then repeatedly applying conditions (I) and (II) gives the estimate

$$|x_1| = |x_1^*| \leq \beta^{(n-1)/2} \min_{1 \leq i \leq n} |x_i^*|.$$

The LLL algorithm, named after its creators, Arjen Lenstra, Hendrik Lenstra Jr., and László Lovász, takes a basis  $x_1, \dots, x_n$  for a lattice  $L \subset \mathbb{R}^n$  and returns an  $\alpha$ -reduced basis  $y_1, \dots, y_n$  for  $L$ . The algorithm, which runs in time polynomial in  $n$  and  $\log \max(|x_1|, \dots, |x_n|)$ , proceeds as follows.

1. Copy the basis elements  $x_1, \dots, x_n$  into  $y_1, \dots, y_n$ .
2. For each vector  $y_i$ ,  $2 \leq i \leq n$  do the following:
  - (a) Reduce  $y_i$  with the previous basis vectors,  $y_j$ ,  $j < i$ :  $y_i \leftarrow y_i - \lceil \mu_{ij} \rceil y_j$ .
  - (b) If  $y_i$  does not satisfy the Lovász condition, then swap  $y_i$  and  $y_{i-1}$  and return to step 2(a).
3. Return the reduced basis  $y_1, \dots, y_n$ .

### An Application: Finding Small Roots of Polynomials mod $M$ [3], [4]

Suppose we're given  $f(x) \in \mathbb{Z}[x]$  and we know that it has a “small” root modulo  $M \in \mathbb{Z}$  that we want to find,  $x_0$ . The root is small in the sense that  $|x_0| < X$  for some specified integer  $X$ . Approximating roots of polynomials in  $\mathbb{Q}[x]$  is easy, but we might not have  $f(x_0) = 0$  unless all of the coefficients of  $f(x)$  are small. The idea is to use the LLL algorithm to build a polynomial  $g(x) \in \mathbb{Z}[x]$  that has the same root  $x_0$  modulo  $M$ , but whose coefficients are small enough that  $g(x_0) = 0$  as well.

Write  $f(x) = a_0 + a_1x + \dots + a_dx^d$  with  $a_i \in \mathbb{Z}$  and consider the matrix

$$B = \begin{bmatrix} M & 0 & \dots & 0 & 0 \\ 0 & MX & \dots & 0 & 0 \\ \vdots & & & \vdots & \vdots \\ 0 & 0 & \dots & MX^{d-1} & 0 \\ a_0 & a_1X & \dots & a_{d-1}X^{d-1} & a_dX^d \end{bmatrix}.$$

Any vector in the lattice spanned by the rows of  $B$  is of the form  $(b_0, b_1X, \dots, b_dX^d)$ ,  $b_i \in \mathbb{Z}$ . We can identify such a vector with a polynomial in  $\mathbb{Z}[x]$  via  $(b_0, b_1X, b_2X^2, \dots, b_dX^d) \mapsto b_0 + b_1x + \dots + b_dx^d$ . Under this identification, every vector in the lattice generated by  $B$  corresponds to a polynomial  $F(x) \in \mathbb{Z}[x]$  with  $F(x_0) \equiv 0 \pmod{M}$ , since this is true for every row of  $B$ .

Running the LLL algorithm on the rows of  $B$  will give a reduced basis for this lattice. Let  $g(x) \in \mathbb{Z}[x]$  be the first element in this reduced basis. If  $X < 2^{-1/2}(d+1)^{-1/d}M^{2/d(d+1)}$ , then  $g(x_0) = 0$  and we can use numerical methods to find  $x_0$ .

## References

- [1] A. K. Lenstra, H. W. Lenstra Jr., and L. Lovász. “Factoring polynomials with rational coefficients”. In: *Mathematische Annalen* 261 (1982), pp. 515–534.
- [2] J. Hoffstein, J. Pipher, and J. Silverman. “An Introduction to Mathematical Cryptography”. In: Springer-Verlag New York, 2014. Chap. 7.
- [3] D. Coppersmith. “Finding a small root of a univariate modular equation”. In: *Eurocrypt 1996: Advances in Cryptology*. Lecture Notes in Computer Science, 1070. Springer, 1996, pp. 155–165.

- [4] S. Galbraith. *Mathematics of Public Key Cryptography*. 2018. Chap. 19. URL: <https://www.math.auckland.ac.nz/~sgal018/crypto-book/main.pdf>.