Evolutionary Sieving in Complex and Integral Lattices

Ahmad Tashfeen, Qi Cheng
tashfeen@ou.edu, cheng@ou.edu
Computer Science, University of Oklahoma

Abstract. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1. Introduction

A lattice is a discrete additive subgroup of a d-dimensional Euclidean space. Intuitively, lattice points are to euclidean spaces what residues modulo d are to the real number line. Consider a basis matrix $\mathcal{B} \in \mathbb{R}^{d\times d}$ with d linearly independent column vectors $\{1 \leq i \leq d : b_i \in \mathbb{R}^d\}$ such that b_i is the i^{th} column of $\mathcal{B}$. We recall $\mathbb{R}^d$ as,

$$\mathbb{R}^d = \{x \in \mathbb{R}^d : \mathcal{B}x\}$$

If we restrict $x \in \mathbb{Z}^d$, i. e., $\mathcal{B}x$ to only the integral linear combinations, we obtain a lattice,

$$\mathcal{L} = \{x \in \mathbb{Z}^d : \mathcal{B}x\}$$

You may also see it more commonly (see [1]) written as,

$$\mathcal{L}(b_1, \ldots, b_d) = \{x_i \in \mathbb{Z} : x_1b_1 + x_2b_2 + x_3b_3 + \cdots + x_db_d\}$$

$$\mathcal{L}(\mathcal{B}) = \left\{x_i \in \mathbb{Z} : \sum_{i=1}^{d} x_ib_i\right\}$$

The dimension of the lattice is $\dim(\mathcal{L}) = d$, i. e., the number of column-vectors in the basis matrix $\mathcal{B}$. And–since $\mathcal{B}$ also has d number of rows, we call $\mathcal{L}$ a full rank lattice. $\mathbb{R}^d$ is now the ambient space for the reduced set of vectors that are in the lattice $\mathcal{L}$. The group operation of a lattice is the vector difference,

$$v_i \in \mathcal{L} \ni v_j, \quad (v_i - v_j) \in \mathcal{L}$$

If we further restrict $\mathcal{B} \in \mathbb{Z}^{d\times d}$, the resulting lattice is known as an integral lattice. We show the plot of a two dimensional lattice ($d = 2$) spanned by the following basis in figure 1.

$$\mathcal{B}_{\text{bad}} = \begin{bmatrix} 95 & 47 \\ 460 & 215 \end{bmatrix}, \quad \mathcal{B}_{\text{good}} = \begin{bmatrix} 1 & 40 \\ 30 & 5 \end{bmatrix}$$

You might imagine another trivial lattice $\mathbb{Z}^2$ spanned by the identity matrix.

In this paper, we will also discuss the lattices formed over the Gaussian integers. Despite the name, these are complex numbers where the real and the imaginary parts are limited to the integers.

$$\mathbb{Z}[i] = \{(a, b) \in \mathbb{Z}^2 : a + bi\}, \quad i^2 = -1$$

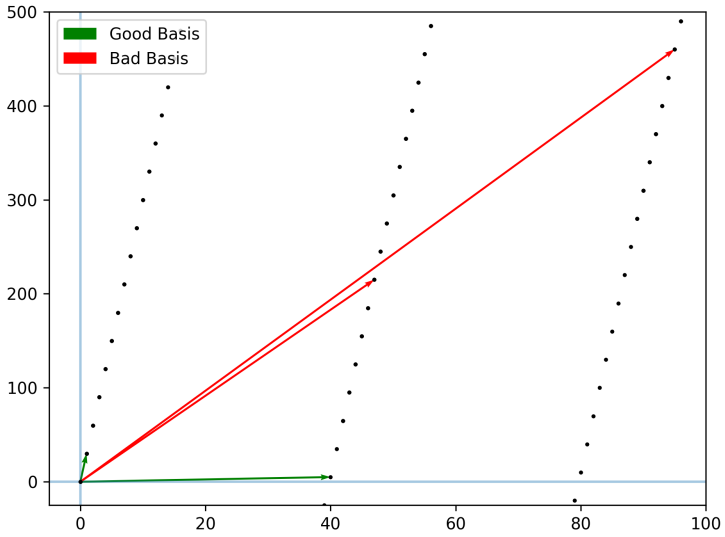

FIGURE 1. An example two dimensional lattice.

We can similarly form a lattice, this time in the ambient space of complex numbers $\mathbb{C}$, by letting $\mathcal{B} \in \mathbb{Z}[i]^{d\times d}$ such that,

$$\mathcal{L} = \{x \in \mathbb{Z}[i]^d : \mathcal{B}x\}$$

1.1. **Lattice Problems.** Under the restrictions of randomised reduction hypothesis [2] both of the problems given bellow (SVP, CVP) are $\mathcal{NP}$-hard. Let $||v||$ be the Euclidean norm of a vector v then,

1.1.1. *Shortest Vector Problem (SVP).* Find the shortest non-zero vector $v_i \in \mathcal{L}$ such that for all $v_j \in \mathcal{L}$ where $i \neq j$,

$$\|v_i\| < \|v_j\|$$

An easier optimisation version of the SVP is stated as follows,

1.1.2. *Approximate Shortest Vector Problem (apprSVP).* Let $\alpha \geq 1$ be an approximation factor and find a $v \in \mathcal{L}$ such that $\|v\|$ is no bigger than α times the length of the shortest vector.

$$\|v\| \leq \alpha \|v_{\text{shortest}}\|$$

1.1.3. *Closest Vector Problem (CVP).* For $w \in \mathbb{R}^d$ where $w \notin \mathcal{L}$ find a vector $v \in \mathcal{L}$ that is closest to w.

$$\min_{v\in\mathcal{L}} \|w - v\| \quad \text{where} \quad \|w - v\| > 0$$

1.1.4. *Shortest Basis Problem (SBP).* Given a bad basis $\mathcal{B}_{\text{bad}}$, reduce it to a good basis $\mathcal{B}_{\text{good}}$. The goodness maybe achieved by minimising the lengths of the basis vectors, e. g.,

$$\sum_{i=1}^{d} \|b_i\|^2 = \|b_1\|^2 + \|b_2\|^2 + \|b_3\|^2 + \cdots + \|b_d\|^2$$

while optionally also requiring $\mathcal{H}(\mathcal{B}_{\text{good}})$ to be closer to 1. Figure 1 shows two basis that span the same lattice. One is labelled as "good" and the other as "bad." A basis gets better

as it's vectors get shorter and more orthogonal. This measure of orthogonality is expressed as the *Hadamard ratio* of a basis $\mathcal{B}$,

$$\mathcal{H}(\mathcal{B}) = \left(\frac{\det \mathcal{L}}{\|b_1\| \, \|b_2\| \, \|b_3\| \cdots \|b_d\|} \right)^{\frac{1}{d}}$$

Note that $0 < \mathcal{H}(\mathcal{B}) < 1$ and $\det \mathcal{L} = |\det \mathcal{B}|$. The closer $\mathcal{H}(\mathcal{B})$ to 1, the better the basis $\mathcal{B}$.

The process of turning a bad basis into a good basis is also sometimes referred to as performing lattice reduction. I. e., SBP maybe solved via the various lattice reduction algorithms. As discussed in the section 2, any solution to the SBP often also uncovers a comparable solution to the apprSVP. Essentially all lattice-cryptography (popular choice for quantum-safe cryptosystems) relay on the inability of lattice reduction algorithms to solve the apprSVP with an approximation of $\alpha \in \mathcal{O}(\sqrt{d})$.

1.2. **Gaussian Heuristic.** It is difficult to verify a solution for the apprSVP and SVP since the length of the shortest vector is unknown in the general case. As the dimension of a lattice increases, we may relay on the Gaussian expected shortest length, also known as the Gaussian heuristic (see [3], [4] for more justification). If $\varepsilon > 0$, then for a sufficiently large dimension d,

$$(1 - \varepsilon)\sigma(\mathcal{L}) \leq \|v_{\text{shortest}}\| \leq (1 + \varepsilon)\sigma(\mathcal{L})$$

whereas,

$$\sigma(\mathcal{L}) = \sqrt{\frac{d}{2\pi e}} (\det \mathcal{L})^{\frac{1}{d}} \quad \text{or} \quad \|v_{\text{shortest}}\| \approx \sigma(\mathcal{L})$$

2. Related Work

If any of the $\mathcal{NP}$-complete problems are shown to be in $\mathcal{P}$ then all of $\mathcal{NP}$-problems are in $\mathcal{P}$. In the same seminal paper [5] where Karp determines a subset of the $\mathcal{NP}$ problems as $\mathcal{NP}$-complete with the aforementioned property, he also gives twenty-one examples of such $\mathcal{NP}$-complete problems. Number 18 on the list is the knapsack problem: Given a set $M \in \mathbb{N}^n, S \in \mathbb{N}$ find $x \in \{0,1\}^n$ such that $Mx = S$. In other words, find a subset of M whose sum is equal to S.

The first cryptosystem to be based on an $\mathcal{NP}$-complete problem uses a disguised knapsack problem [3] and was attempted by Merkle and Hellman [6]. Note that we say a *disguised* knapsack problem since whether a cryptographic system can be as hard to break as an $\mathcal{NP}$-complete problem is an open problem in itself [7].

Lagarias and Odlyzko [8] showed that any knapsack problem can be encoded as an SVP. We state the gist of their idea. Take any knapsack problem $M = \{r_1, r_2, \ldots, r_n\}$ with S and the relevant solution x such that $Mx = S$. Now consider the following lattice basis in $\mathbb{N}^{d \times d}$ with dimension $d = n + 1$,

$$\mathcal{B}_{\text{bad}} = \begin{bmatrix} 2 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 2 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 2 & \cdots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 2 & 1 \\ r_1 & r_2 & r_3 & \cdots & r_n & S \end{bmatrix}$$

The lattice spanned by $\mathcal{B}_{\text{bad}}$ must have a vector t that is the result of an integral linear combination due to x,

$$t = \begin{bmatrix} 2 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 2 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 2 & \cdots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 2 & 1 \\ r_1 & r_2 & r_3 & \cdots & r_n & S \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \\ -1 \end{bmatrix} = \begin{bmatrix} 2x_1 - 1 \\ 2x_2 - 1 \\ 2x_3 - 1 \\ \vdots \\ 2x_n - 1 \\ M \cdot x - S \end{bmatrix} = \begin{bmatrix} 2x_1 - 1 \\ 2x_2 - 1 \\ 2x_3 - 1 \\ \vdots \\ 2x_n - 1 \\ 0 \end{bmatrix}$$

Since $x \in \{0,1\}^n$ then any $(2x_i - 1) \in t$ must be either 1 or -1. Therefore,

$$\|t\| = \sqrt{n}$$

t is very likely to be the shortest vector in $\mathcal{L}(\mathcal{B}_{\text{bad}})$. The shortest vector in the lattice spanned by $\mathcal{B}_{\text{bad}}$ will reveal the solution x to the knapsack problem.

2.1. **Lattice Reduction Algorithms.** If a lattice is expressed in terms of it's good basis then solving the SVP and the CVP becomes fairly easy. This means that if we first solve the SBP then both the SVP and CVP are easy. For example, assume for a certain $\mathcal{B}_{\text{good}}$ that all column vectors b_i are pairwise orthogonal, i. e., for $i \neq j$ we know that $b_i \cdot b_j = 0$. Then for any $x \in \mathbb{Z}^d$,

$$\|x_1 b_1 + x_2 b_2 + x_3 b_3 + \cdots x_d b_d\|^2 = x_1^2 \|b_1\|^2 + x_2^2 \|b_2\|^2 + x_3^2 \|b_3\|^2 + \cdots x_d^2 \|b_d\|^2$$

and the shortest non-zero vector(s) can be found in,

$$\{\pm b_1, \pm b_2, \pm b_3, \cdots \pm b_d\}$$

Similarly for the solution of an instance of the CVP using a good basis, see the Babai's nearest hyperplane algorithm [9] or Theorem 7.34 (pg. 405) of Hoffstein [3].

2.1.1. *Lenstra, Lenstra, and Lovász (LLL) Algorithm.* The first lattice reduction algorithm is by Gauss. It works like the Euclid's greatest common divisor (highest common factor) algorithm but with two, two-dimensional vectors, i. e., $\mathcal{B} = \{b_1, b_2\}$. Assume without the loss of generality that $\|b_1\| < \|b_2\|$ then,

$$b_2 = b_2 - \left\lfloor \frac{b_1 \cdot b_2}{b_1 \cdot b_1} \right\rceil b_1$$

If $\|b_2\|$ is still greater than $\|b_1\|$ we can stop. Otherwise, swap b_2 with b_1 and try again.

The LLL algorithm generalises the Gaussian lattice reduction from two to d dimensions. Just like the Gaussian lattice reduction, it subtracts an integral multiple of a shorter basis vector from a larger basis vector till some size condition is fulfilled. For each $1 \leq k \leq d$, we reduce b_k as the following,

$$b_k = b_k - \left\lfloor \frac{b_j^* \cdot b_k}{b_j^* \cdot b_j^*} \right\rceil b_j$$

b_i^* is the i^{th} basis vector in the Gram-Schmidt orthogonalization of $\mathcal{B}$. The reduction of b_k is carried out using the Gram-Schmidt orthogonalizations of all the already reduced b_j for every $k - 1 > j \geq 1$ where the following *size condition* is met,

$$\left| \frac{b_j^* \cdot b_k}{b_j^* \cdot b_j^*} \right| > \frac{1}{2}$$

If LLL terminates after only this recursive size reduction then the goodness of the reduced basis depends on the order of the original basis vectors in the basis matrix. Therefore, after the size reduction of b_k, another condition, namely the popular Lovász condition is checked. This is given as,

$$\|b_k^*\|^2 \geq \left(\frac{3}{4} - \left(\frac{b_{k-1}^* \cdot b_k}{b_{k-1}^* \cdot b_{k-1}^*}\right)^2\right) \left\|b_{k-1}^*\right\|^2$$

Written another way for $\mu_{k,k-1} = \left(b_{k-1}^* \cdot b_k\right) / \left(b_{k-1}^* \cdot b_{k-1}^*\right)$ and $\delta = 3/4$,

$$\|b_k^*\|^2 \geq \left(\delta - \mu_{k,k-1}^2\right) \left\|b_{k-1}^*\right\|^2$$

If the Lovász condition is met then b_k is considered reduced and k will be incremented. Otherwise, for optimal ordering, we swap b_k and b_{k-1} and decrement k[1]. For a more in depth analysis of LLL, read Deng [10]. The full description is given in algorithm 1.

Input: Lovász condition constant: $0 < \delta < 1$.
Input: Bad basis: $\mathcal{B} = \{b_1, b_2, b_3, \ldots, b_d\}$.
Output: Good/Reduced basis: $\mathcal{B} = \{b_1, b_2, b_3, \ldots, b_d\}$.

$k \leftarrow 2$
$(\mathcal{B}^*, \mu) \leftarrow$ **GramSchmidt**$(\mathcal{B})$
  while $k \leq d$
    for $j \in \{k-1, k-2, k-3, \ldots, 1\}$
      if $\mu_{k,j} > 0.5$
        $b_k \leftarrow b_k - \lfloor \mu_{k,j} \rceil \, b_j$
        $(\mathcal{B}^*, \mu) \leftarrow$ **GramSchmidt**$(\mathcal{B})$
    if $\|b_k^*\|^2 \geq \left(\delta - \mu_{k,k-1}^2\right) \left\|b_{k-1}^*\right\|^2$
      $k \leftarrow k + 1$
    else
      Swap(b_{k-1}, b_k)
      $(\mathcal{B}^*, \mu) \leftarrow$ **GramSchmidt**$(\mathcal{B})$
      $k \leftarrow \max(k-1, 2)$

ALGORITHM 1. The Lenstra, Lenstra, and Lovász (LLL) algorithm.

The authors [11] of the LLL algorithm show that it is a polynomial time lattice reduction algorithm for all $0 < \delta < 1$. The outer loop runs at most in,

$$\mathcal{O}(d^2 \log(d) + d^2 \log(\max \|b_i\|))$$

and that it solves the apprSVP with an approximation factor of $\alpha = 2^{(d-1)/2}$. It is also an open problem whether LLL terminates in polynomial time for $\delta = 1$.

2.1.2. *LLL Variations.* While LLL is the only polynomial time algorithm for lattice reduction, many of it's generalisations tend to perform just as fast during empirical analysis and yield a further reduced basis. Gamma et alia argue by their extensive empirical analysis [12] that there is a gap between what theory is able to prove and what the true power of the reduction algorithms maybe.

[1]More precisely $k = \max(k-1, 2)$

The two possible exponential time variations of LLL are due to Schnorr [13] and Euchner [14]: DEEP (deep insertion method) and BKZ (block Korkin–Zolotarev). DEEP differs from the standard LLL when the Lovász condition fails. In standard LLL we simply swap the b_k with b_{k-1} whereas in DEEP we insert b_k at an optimal place before the k^{th} basis vector. While in BKZ, instead of reducing b_k with only one b_j, the same is done with a block of basis vectors, $b_j, b_{j+1}, b_{j+2}, \ldots b_{j+\beta-1}$ where β is the block size. Note that if we let $\beta = d$ then the shortest vector in the output of BKZ solves the SVP problem and for any $\beta < d$ we solve some version of the apprSVP.

3. Methodology

Apart from the reduction algorithms mentioned in section 2, we now discuss the approach taken in this paper. The method of sieving for the shortest vector first surfaced due to the works of Ajtai, Kumar and Sivakumar [15]. The idea behind sieving is quite simple: if we have two lattice vectors $u \in \mathcal{L} \ni v$ and we want a shorter one, we might try $u - v$. More specifically, repeatedly checks if there exists (u, v) in a fixed subset P of $\mathcal{L}$ such that $\|u - v\| < \|u\|$ then u is replaced with $\|u - v\|$. When $\|u - v\| < \|u\|$ is not true for any pair (u, v) in P^2, we hope to solve some version of the apprSVP with the shortest vector in P. We give our variation of the aforementioned idea as a naïve sieving algorithm 2.

Input: $P \subset \mathcal{L}$.
Output: Reduced version of subset: P.

1: $R \leftarrow \varnothing$
2: **do**
3: $\quad P \leftarrow$ **Select**$(P \cup R)$
4: $\quad R \leftarrow \varnothing$
5: $\quad$ **for** each $(u, v) \in P^2$
6: $\quad\quad t \leftarrow u - v$
7: $\quad\quad$ **if** $(\|t\| > 0) \wedge (t \notin P) \wedge (\|t\| < \|u\| \vee \|t\| < \|v\|)$
8: $\quad\quad\quad R \leftarrow R \cup \{t\}$
9: **while** $R \neq \varnothing$

Algorithm 2. Naïve sieving algorithm.

Algorithm 2 checks the difference of all the possible pairings in P for a shorter non-zero vector not already present in P. These newer and shorter vectors are collected in R. At the start of each iteration, we combine P and R *selecting* out of the combination at most $|P|$ successive shortest vectors to be reassigned as P. The sieving terminates when P^2 no longer contains a pair (u, v) whose difference's length is shorter than any of the ones already in P.

We show an example by reducing the bad basis given in section 1,

$$\mathcal{B}_{\text{bad}} = \begin{bmatrix} 95 & 47 \\ 460 & 215 \end{bmatrix}$$

The initial P is randomly picked as follows,

$$P = \begin{bmatrix} -46 & -94 & -97 & -475 \\ -185 & -430 & -520 & -2300 \end{bmatrix}$$

and figure 2 shows the new P at each iteration of seven iterations ran.

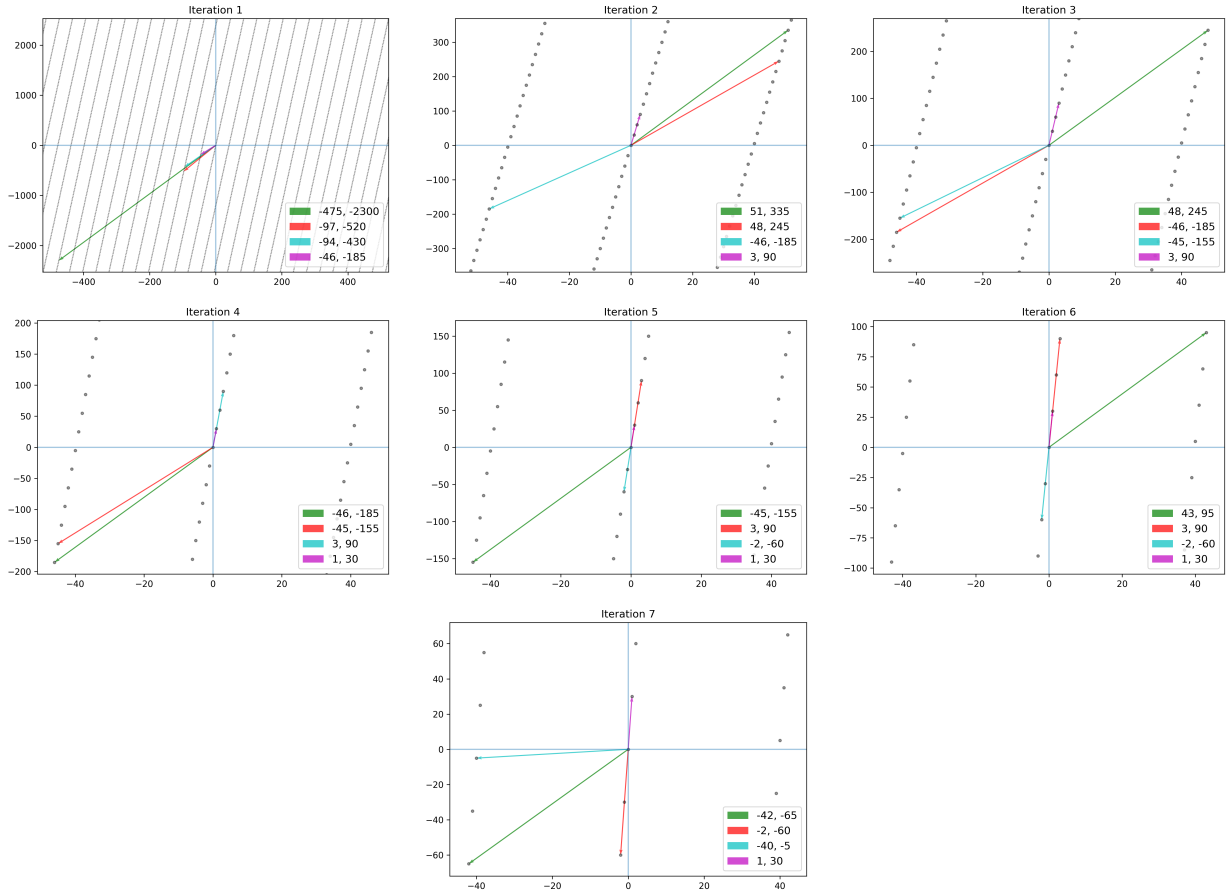

FIGURE 2. Alg. 2 reducing $\mathcal{B}_{\text{bad}}$ with $P = \begin{bmatrix} -46 & -94 & -97 & -475 \\ -185 & -430 & -520 & -2300 \end{bmatrix}$

Note that in this case, we find the shortest vector by solving the SBP exactly. The seventh iteration in figure 2 contains,

$$\mathcal{B}_{\text{good}} = \begin{bmatrix} 1 & -40 \\ 30 & -5 \end{bmatrix}$$

References

[1] Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems: a cryptographic perspective*, volume 671. Springer Science & Business Media, 2002.

[2] Miklós Ajtai. The shortest vector problem in l2 is np-hard for randomized reductions. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 10–19, 1998.

[3] Jeffrey Hoffstein, Jill Pipher, Joseph H Silverman, and Joseph H Silverman. *An introduction to mathematical cryptography*, volume 1, pages 377, 402, 405. Springer, 2008.

[4] Carl Ludwig Siegel. A mean value theorem in geometry of numbers. *Annals of Mathematics*, 46(2):340–347, 1945.

[5] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–104, New York, 1972. Plenum.

[6] Ralph C Merkle and Martin E Hellman. Hiding information and signatures in trapdoor knapsacks. In *Secure communications and asymmetric cryptosystems*, pages 197–215. Routledge, 2019.

[7] Rafael Pass. Parallel repetition of zero-knowledge proofs and the possibility of basing cryptography on np-hardness. In *21st Annual IEEE Conference on Computational Complexity (CCC'06)*, pages 13–pp. IEEE, 2006.

[8] Jeffrey C Lagarias and Andrew M Odlyzko. Solving low-density subset sum problems. *Journal of the ACM (JACM)*, 32(1):229–246, 1985.

[9] László Babai. On lovász'lattice reduction and the nearest lattice point problem. *Combinatorica*, 6:1–13, 1986.

[10] Xinyue Deng. An introduction to lenstra-lenstra-lovasz lattice basis reduction algorithm. *Massachusetts Institute of Technology (MIT)*, 2016.

[11] Arjen K Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische annalen*, 261:515–534, 1982.

[12] Nicolas Gama and Phong Q Nguyen. Predicting lattice reduction. In *Advances in Cryptology–EUROCRYPT 2008: 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey, April 13-17, 2008. Proceedings 27*, pages 31–51. Springer, 2008.

[13] Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical computer science*, 53(2-3):201–224, 1987.

[14] Claus-Peter Schnorr and Martin Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical programming*, 66:181–199, 1994.

[15] Miklós Ajtai, Ravi Kumar, and Dandapani Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 601–610, 2001.

Computer Science, Gallogly College of Engineering, University of Oklahoma