Research Article

Masaya Yasuda*, Kazuhiro Yokoyama, Takeshi Shimoyama, Jun Kogure and Takeshi Koshiba

Analysis of decreasing squared-sum of Gram–Schmidt lengths for short lattice vectors

DOI: 10.1515/jmc-2016-0008
Received February 1, 2016; revised December 7, 2016; accepted February 2, 2017

Abstract: In 2015, Fukase and Kashiwabara proposed an efficient method to find a very short lattice vector. Their method has been applied to solve Darmstadt shortest vector problems of dimensions 134 to 150. Their method is based on Schnorr's random sampling, but their preprocessing is different from others. It aims to decrease the sum of the squared lengths of the Gram–Schmidt vectors of a lattice basis, before executing random sampling of short lattice vectors. The effect is substantiated from their statistical analysis, and it implies that the smaller the sum becomes, the shorter sampled vectors can be. However, no guarantee is known to strictly decrease the sum. In this paper, we study Fukase–Kashiwabara's method in both theory and practice, and give a heuristic but practical condition that the sum is strictly decreased. We believe that our condition would enable one to monotonically decrease the sum and to find a very short lattice vector in fewer steps.

Keywords: SVP, LLL algorithm, random sampling

MSC 2010: Primary 68R01; secondary 06B99

Communicated by: Kristin Lauter

1 Introduction

Given n linearly independent column vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the set of all integral linear combinations of the $\mathbf{b}_i$'s defines a *lattice* of dimension n. The matrix $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n] \in \mathbb{R}^{m \times n}$ is called a *basis* of the lattice. Given a lattice basis, the shortest vector problem (SVP) is to find a non-zero shortest lattice vector, and it has been a landmark problem in complexity theory for a long time. No efficient algorithm is currently known to find very short vectors in high dimensional lattices. Ajtai [1] proved that SVP is NP-hard under randomized reduction (see [13] for the NP-hardness of approximate SVP). In cryptography, the computational hardness of SVP assures the security of lattice-based cryptography such as [9, 10], to which has recently been paid attention as a candidate of post-quantum cryptography. There are four main approaches for solving SVP: *lattice reduction*, *enumeration*, *sieving*, and finally *random sampling*. Given a basis of a lattice L, lattice

***Corresponding author: Masaya Yasuda:** Institute of Mathematics for Industry, Kyushu University, 744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan, e-mail: yasuda@imi.kyushu-u.ac.jp
Kazuhiro Yokoyama: Department of Mathematics, Rikkyo University, Nishi-Ikebukuro, Tokyo 171-850, Japan, e-mail: kazuhiro@rikkyo.ac.jp
Takeshi Shimoyama, Jun Kogure: Fujitsu Laboratories Ltd., 1-1, Kamikodanaka 4-chome, Nakahara-ku, Kawasaki, Kanagawa 211-8588, Japan, e-mail: shimo-shimo@jp.fujitsu.com, kogure@jp.fujitsu.com
Takeshi Koshiba: Division of Mathematics, Electronics and Informatics, Graduate School of Science and Engineering, Saitama University, 255 Shimo-Okubo, Sakura, Saitama 338-8570, Japan, e-mail: koshiba@mail.saitama-u.ac.jp

reduction finds a basis with short and nearly orthogonal vectors; e.g., the Lenstra–Lenstra–Lovász (LLL) [11] and the block Korkine–Zolotarev (BKZ) [18] algorithms. Enumeration performs to enumerate all lattice points within a sphere S around a target vector; e.g., Schnorr–Euchner's enumeration [18] and Gama–Nguyen–Regev's pruned enumeration [8]. Sieving aims to do a randomized sampling of $L \cap S$ while enumeration performs an exhaustive search of $L \cap S$; e.g., the Ajtai–Kumar–Sivakumar algorithm [2]. Random sampling randomly enumerates a number of short lattice vectors until a very short lattice vector is found.

We focus on random sampling. Schnorr [17] first proposed a random sampling algorithm, called *random sampling reduction* (RSR). Given a basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$, the sampling algorithm (SA) in RSR generates a number of short lattice vectors to find $\mathbf{v}$ with $\|\mathbf{v}\|^2 < 0.99\|\mathbf{b}_1\|^2$. Buchmann and Ludwig [4] proposed *simple sampling reduction* (SSR) to make RSR practical. In 2015, Fukase and Kashiwabara [5] proposed a method for SVP. Their method has been applied to solve Darmstadt SVP problems of dimensions 134 to 150 by Kashiwabara and Teruya (see http://www.latticechallenge.org/svp-challenge/). Their method is based on RSR, but their preprocessing is different from others. Before describing their method, we define the following notation.

Definition 1.1. Given a lattice basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$, let $[\mathbf{b}_1^*, \dots, \mathbf{b}_n^*]$ denote its Gram–Schmidt vectors. Then we define by

$$\mathrm{SS}(\mathbf{B}) = \sum_{i=1}^{n} \|\mathbf{b}_i^*\|^2$$

the sum of the squared lengths of the Gram–Schmidt vectors $[\mathbf{b}_1^*, \dots, \mathbf{b}_n^*]$.

Fukase and Kashiwabara [5] decrease $\mathrm{SS}(\mathbf{B})$ before executing random sampling. The *smaller* the squared-sum $\mathrm{SS}(\mathbf{B})$ becomes, the *shorter* lattice vectors can be sampled. To decrease $\mathrm{SS}(\mathbf{B})$, Fukase and Kashiwabara insert a short lattice vector $\mathbf{v}$ into $\mathbf{B}$ to obtain a new basis $\mathbf{C}$, and then reduce $\mathbf{C}$ by LLL to obtain a basis $\mathbf{B}' = [\mathbf{b}_1', \dots, \mathbf{b}_n']$. Then we *sometimes* have

$$\mathrm{SS}(\mathbf{B}') < \mathrm{SS}(\mathbf{B}). \tag{1.1}$$

By repeating the procedures, they attempt to decrease $\mathrm{SS}(\mathbf{B})$ as much as possible. However, there is no guarantee to strictly decrease $\mathrm{SS}(\mathbf{B})$.

In this paper, we analyze which lattice vectors $\mathbf{v}$ can decrease the squared-sum $\mathrm{SS}(\mathbf{B})$, and give a condition of $\mathbf{v}$ such that the squared-sum $\mathrm{SS}(\mathbf{B})$ is strictly decreased. Specifically, we consider the following: Given an LLL-reduced basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ of a lattice L and a vector $\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{b}_i^* \in L$ with $v_n = 1$ and insertion index k, we consider

$$\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n] \xrightarrow{\text{insertion of } \mathbf{v}} \mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_n] \xrightarrow{\text{LLL reduction}} \mathbf{B}' = [\mathbf{b}_1', \dots, \mathbf{b}_n'], \tag{1.2}$$

where $\mathbf{C} = [\mathbf{b}_1, \dots, \mathbf{b}_{k-1}, \mathbf{v}, \mathbf{b}_k, \dots, \mathbf{b}_{n-1}]$. Note that $\mathbf{C}$ is a basis of the whole lattice L due to $v_n = 1$ (see Proposition 4.3 below). We focus on the LLL-reduction for $\mathbf{C}$ as in [5]. Our main contributions are as follows:

(i) We compute the Gram–Schmidt vectors $[\mathbf{c}_1^*, \dots, \mathbf{c}_n^*]$ of $\mathbf{C}$, and give their explicit lengths $\|\mathbf{c}_i^*\|$ for $1 \le i \le n$. Thus we obtain the explicit gap between two squared-sums $\mathrm{SS}(\mathbf{B})$ and $\mathrm{SS}(\mathbf{C})$.

(ii) We study the behavior of the LLL algorithm for a general basis $\mathbf{S}$, and show that swaps of the LLL algorithms can strictly decrease $\mathrm{SS}(\mathbf{S})$. In particular, we estimate how much $\mathrm{SS}(\mathbf{S})$ is decreased by one time swap, and study the total number of swaps in the LLL algorithm.

(iii) We estimate the gap between $\mathrm{SS}(\mathbf{C})$ and $\mathrm{SS}(\mathbf{B}')$, by applying the estimates of (ii) for the basis $\mathbf{C}$. Specifically, we estimate the average of decreasing values of $\mathrm{SS}(\mathbf{C})$ by one time swap in the LLL algorithm, and the number of swaps. We give a heuristic but practical condition of a candidate lattice vector $\mathbf{v}$ satisfying condition (1.1). We call such $\mathbf{v}$ a *mutant vector*.

(iv) We also verify our analysis by experiments. Our experimental results imply that the condition of mutant vectors gives a good criterion to strictly decrease $\mathrm{SS}(\mathbf{B})$. Thereby we expect that mutant vectors could help to monotonically decrease $\mathrm{SS}(\mathbf{B})$, and it would make it easier to find a very short lattice vector.

The paper is organized as follows: In Section 2, we review lattices and lattice reduction. In Section 3, we review random sampling algorithms and present some results by Fukase and Kashiwabara [5]. In Section 4,

we compute the Gram–Schmidt vectors of the basis $\mathbf{C}$, and give the explicit gap between SS($\mathbf{B}$) and SS($\mathbf{C}$). In Section 5, we review some basic properties of the LLL algorithm for a general basis $\mathbf{S}$, and study the behavior of the LLL algorithm for $\mathbf{S}$. In Section 6, we analyze the LLL-reduction for $\mathbf{C}$ by using results obtained in Section 5. In Section 7, we estimate the gap between SS($\mathbf{B}$) and SS($\mathbf{B}'$), and define mutant vectors to strictly decrease SS($\mathbf{B}$). In Section 8, we verify our analysis by experiments. In Section 9, we conclude this work and outline some future issues.

Notation. For a vector $\mathbf{a} = (a_1, \dots, a_n) \in \mathbb{R}^n$, let $\|\mathbf{a}\|$ denote its Euclidean norm defined by $\|\mathbf{a}\|^2 = \sum_{i=1}^n a_i^2$. For two vectors $\mathbf{a} = (a_1, \dots, a_n)$ and $\mathbf{b} = (b_1, \dots, b_n) \in \mathbb{R}^n$, let $\langle \mathbf{a}, \mathbf{b} \rangle$ denote the inner product $\sum_{i=1}^n a_i b_i$.

2 Preliminaries

In this section, we briefly review lattices and lattice reduction.

2.1 Lattices

For two positive integers m and n, let $\mathbf{b}_1, \dots, \mathbf{b}_n$ be n column vectors of $\mathbb{Z}^m$ (we only consider integral vectors). Set $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n] \in \mathbb{Z}^{m \times n}$, and let

$$L = \mathcal{L}(\mathbf{B}) := \left\{ \sum_{i=1}^n x_i \mathbf{b}_i \mid x_i \in \mathbb{Z},\ 1 \le i \le n \right\}$$

denote the set of all integral linear combinations of the $\mathbf{b}_i$'s. The set L gives a subgroup of $\mathbb{R}^m$. We say that L is a *lattice* of dimension n if all the $\mathbf{b}$'s are linearly independent over $\mathbb{R}$. When $n = m$, the lattice L is called *full-dimensional* or *full-rank* (in this paper, we only consider full-rank lattices). In this case, the matrix $\mathbf{B}$ is called a *basis* of L. Every lattice has infinitely many bases. If $\mathbf{B}_1$ and $\mathbf{B}_2$ are two bases, then there exists a unimodular matrix $\mathbf{V} \in \mathrm{GL}_n(\mathbb{Z})$ such that $\mathbf{B}_1 = \mathbf{B}_2 \cdot \mathbf{V}$. The volume of L, denoted by $\mathrm{vol}(L)$, is defined as

$$\mathrm{vol}(L) = \left(\det(\langle \mathbf{b}_i, \mathbf{b}_j \rangle)_{1 \le i,j \le n}\right)^{1/2} > 0,$$

where $(\langle \mathbf{b}_i, \mathbf{b}_j \rangle)_{1 \le i,j \le n}$ denotes the $n \times n$ Gram-matrix of a basis $\mathbf{B}$. The volume $\mathrm{vol}(L)$ is independent of the choice of the bases.

The Gram–Schmidt orthogonalization of a basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ is the orthogonal family $[\mathbf{b}_1^*, \dots, \mathbf{b}_n^*]$, recursively defined by

$$\mathbf{b}_i^* = \mathbf{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \mathbf{b}_j^*, \quad \text{where } \mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\|\mathbf{b}_j^*\|^2} \text{ for } 1 \le j < i \le n. \tag{2.1}$$

Let $\mathbf{B}^* = [\mathbf{b}_1^*, \dots, \mathbf{b}_n^*] \in \mathbb{R}^{m \times n}$ and $\mathbf{U} = (\mu_{i,j})_{1 \le i,j \le n} \in \mathbb{R}^{n \times n}$, where $\mu_{i,i} = 1$ for all i and $\mu_{i,j} = 0$ for all $j > i$. Then $\mathbf{B} = \mathbf{B}^* \cdot \mathbf{U}^T$ and

$$\mathrm{vol}(L) = \prod_{i=1}^n \|\mathbf{b}_i^*\|.$$

2.2 Lattice reduction

Given a basis of a lattice L, lattice reduction outputs a basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ of L with short and nearly orthogonal vectors $\mathbf{b}_1, \dots, \mathbf{b}_n$. Lattice reduction gives a powerful tool to break lattice-based cryptosystems such as [9, 10]. The *Hermite factor* γ of a lattice reduction algorithm is defined by

$$\gamma = \frac{\|\mathbf{b}_1\|}{\mathrm{vol}(L)^{1/n}}$$

with the output basis $[\mathbf{b}_1, \dots, \mathbf{b}_n]$ (we assume $\|\mathbf{b}_1\| \le \|\mathbf{b}_2\| \le \cdots$). This factor is a good index to measure the output quality of a lattice reduction algorithm. Note that the output quality becomes better as γ is smaller.

Here we introduce two practical algorithms: LLL is a polynomial-time algorithm [11]. Gama–Nguyen's experimental results [7, Figure 4] show that the Hermite factor of LLL is practically 1.022^n on average in high dimension $n \geq 100$. BKZ is a blockwise generalization of LLL with sub-exponential complexity [18]. No good upper bound on the complexity is currently known. BKZ uses a blockwise parameter β, and larger β improves the output quality but increases the running time. In practice, $\beta \approx 20$ can achieve the best time/quality compromise. It follows from [7, Section 5.2] that the Hermite factor with $\beta = 20$ is 1.0128^n on average. Currently, BKZ 2.0 is known as the state-of-the-art implementation of the BKZ algorithm.

Geometric Series Assumption (GSA). Except when a lattice has a special structure, practical reduction algorithms output a basis $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ such that

$$\frac{\|\mathbf{b}_i^*\|}{\|\mathbf{b}_{i+1}^*\|} \approx q \quad \text{for any } 1 \leq i \leq n-1$$

(i.e. $\|\mathbf{b}_i^*\| \approx q^{1-i}\|\mathbf{b}_1\|$ for $1 \leq i \leq n$), where the constant q depends on algorithms (this assumption was first introduced in [17]). Under GSA, the values $\log_2(\|\mathbf{b}_1\|/\|\mathbf{b}_i^*\|)$ are on a straight line (see, e.g., [17, Figure 1]). According to [11], we have $q \approx 1.02^2 \approx 1.04$ (resp. $q \approx 1.025$) for LLL (resp. BKZ with $\beta = 20$) for random lattices in practice.

3 Random sampling of short lattice vectors

In this section, we mainly present some results by Fukase and Kashiwabara [5]. Before presenting their work, let us review previous work on random sampling.

3.1 Review of previous work

For a lattice L of dimension n, fix a constant u of search space bound with $1 \leq u < n$. Let a basis $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ of L be given. As a main subroutine of Schnorr's RSR [17], SA samples $\mathbf{v} = \sum_{i=1}^n v_i \mathbf{b}_i^* \in L$ satisfying

$$v_i \in \begin{cases} (-\frac{1}{2}, \frac{1}{2}] & \text{if } 1 \leq i < n-u, \\ (-1, 1] & \text{if } n-u \leq i < n, \\ \{1\} & \text{if } i = n. \end{cases} \tag{3.1}$$

Let $S_{u,\mathbf{B}}$ denote the set of $\mathbf{v} = \sum_{i=1}^n v_i \mathbf{b}_i^*$ satisfying condition (3.1). Since the number of candidates for v_i with $|v_i| \leq \frac{1}{2}$ (resp. $|v_i| \leq 1$) is 1 (resp. 2), there are 2^u lattice vectors in $S_{u,\mathbf{B}}$. By calling SA up to 2^u times, RSR generates $\mathbf{v}$ satisfying $\|\mathbf{v}\|^2 < 0.99\|\mathbf{b}_1\|^2$ (see [17, Theorem 1]). In [4], Buchmann and Ludwig proposed SSR to get rid of two RSR assumptions, namely, the *randomness assumption* (RA)[1] and GSA (they claim that both RA and GSA do not hold strictly in practice). Ludwig [12] gave a more detailed view about the behavior of SSR. Schneider and Göttert [16] presented a GPU implementation of SSR with the BKZ algorithm.

In 2015, Fukase and Kashiwabara [5] proposed a method for SVP. Their method is based on Schnorr's RSR, and it has two extensions: The first one is to represent a lattice vector by a sequence of natural numbers via the Gram–Schmidt orthogonalization, and to sample lattice vectors on an appropriate distribution of the representation. The second one is to decrease the sum of the squared lengths of Gram–Schmidt vectors to make it easier to sample very short lattice vectors. The effectiveness of their extensions is guaranteed by their statistical analysis on lattices, which we shall describe in the next subsection.

1 RA states that the coefficients v_i of $\mathbf{v} = \sum_{i=1}^n v_i \mathbf{b}_i^*$ sampled by SA are uniformly distributed in $[-\frac{1}{2}, \frac{1}{2}]$ for $1 \leq i < n-u$ and in $[-1, 1]$ for $n-u \leq i < n$.

3.2 Statistical analysis of Fukase and Kashiwabara on lattices

In [5, Definiton 3], Fukase and Kashiwabara extend Schnorr's search space $S_{u,\mathbf{B}}$ to define a wider search space $V_{\mathbf{B}}(\mathbf{s},\mathbf{t})$ for $\mathbf{s},\mathbf{t}\in\mathbb{N}^c$ with some c. Given $\mathbf{v}=\sum_{i=1}^n v_i\mathbf{b}_i^*\in V_{\mathbf{B}}(\mathbf{s},\mathbf{t})$, they first assume

$$E[\|\mathbf{v}\|^2]\approx\frac{1}{12}\sum_{i=1}^n\|\mathbf{b}_i^*\|^2=\frac{1}{12}\mathrm{SS}(\mathbf{B})$$

(see [5, Assumption 2]). Under this assumption, they apply the generalized central limit theorem to obtain the following assumption on the distribution of $\|\mathbf{v}\|^2$ (see [5, Assumption 3]).

Assumption 3.1. *The distribution of the length* $\|\mathbf{v}\|^2=\sum_{i=1}^n v_i^2\|\mathbf{b}_i^*\|^2$ *with* $\mathbf{v}\in V_{\mathbf{B}}(\mathbf{s},\mathbf{t})$ *follows the normal distribution* $\mathcal{N}(\mu,\sigma^2)$ *with*

$$\mu=\frac{\sum_{i=1}^n\|\mathbf{b}_i^*\|^2}{12}=\frac{\mathrm{SS}(\mathbf{B})}{12}\quad\text{and}\quad\sigma=\Big(\frac{\sum_{i=1}^n\|\mathbf{b}_i^*\|^4}{180}\Big)^{1/2}.$$

Assumption 3.1 shows that *shorter* lattice vectors are sampled as the squared-sum $\mathrm{SS}(\mathbf{B})$ becomes *smaller*. Fukase and Kashiwabara verified Assumption 3.1 by experiments, and showed that it does not hold strictly, but it is close enough for finding very short vectors (see [5, Figure 1]). Their experiments were performed over a random lattice of dimension 120. Assumption 3.1 enables one to estimate the probability of finding a lattice vector shorter than a given constant η. Specifically, the probability of finding $\mathbf{v}=\sum_{i=1}^n v_i\mathbf{b}_i^*\in V_{\mathbf{B}}(\mathbf{s},\mathbf{t})$ shorter than η is

$$\frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{\eta^2}\exp\Big(-\frac{(x-\mu)^2}{2\sigma^2}\Big)dx=\frac{1}{2}\Big(1+\mathrm{erf}\Big(\frac{\eta^2-\mu}{\sqrt{2}\sigma}\Big)\Big),$$

where $\mathrm{erf}(x)=\frac{2}{\sqrt{\pi}}\int_0^x\exp(-t^2)dt$ is the error function.

3.3 Basic strategy of Fukase and Kashiwabara for finding short lattice vectors

Assumption 3.1 implies an importance of decreasing the sum of the squared lengths of Gram–Schmidt vectors to find a short lattice vector. Once we obtain a basis $\mathbf{B}$ of L with *smaller* squared-sum $\mathrm{SS}(\mathbf{B})$, we can generate a short vector $\mathbf{v}=\sum_{i=1}^n v_i\mathbf{b}_i^*\in L$ with *higher* probability. The basic strategy in [5] for finding a short vector consists of the following two steps:

- *Step 1*. Given a basis of a lattice L, we first decrease the sum of the squared lengths of its Gram–Schmidt vectors as much as possible, and obtain a basis $\mathbf{B}$ of L with small $\mathrm{SS}(\mathbf{B})$.
- *Step 2*. With such basis $\mathbf{B}$, we find a short lattice vector by randomly sampling $\mathbf{v}=\sum_{i=1} v_i\mathbf{b}_i^*\in V_{\mathbf{B}}(\mathbf{s},\mathbf{t})$.

To decrease $\mathrm{SS}(\mathbf{B})$ in Step 1, Fukase and Kashiwabara insert a certain lattice vector $\mathbf{v}$ into a given basis $\mathbf{B}=[\mathbf{b}_1,\dots,\mathbf{b}_n]$ at a certain position k to obtain a new basis $\mathbf{C}$, as in (1.2). The insertion index k is determined as follows (see [5, Definition 4]).

Definition 3.2 (Insertion index). Let $\mathbf{B}=[\mathbf{b}_1,\dots,\mathbf{b}_n]$ be a basis of L. For a fixed constant $0<\alpha\le 1$, the insertion index k of a vector $\mathbf{v}\in L$ is defined by

$$\min\{1\le j\le n\mid\|\pi_j(\mathbf{v})\|^2<\alpha\|\mathbf{b}_j^*\|^2\},$$

where for $2\le j\le n$ we let $\pi_j:\mathbb{R}^n\to V_{j-1}^{\perp}$ denote the orthogonal projection over the orthogonal supplement of $V_{j-1}=\langle\mathbf{b}_1,\dots,\mathbf{b}_{j-1}\rangle_{\mathbb{R}}=\langle\mathbf{b}_1^*,\dots,\mathbf{b}_{j-1}^*\rangle_{\mathbb{R}}$. In particular, let π_1 denote the identity map. If $\|\pi_j(\mathbf{v})\|^2\ge\alpha\|\mathbf{b}_j^*\|^2$ for all $1\le j\le n$, we do not insert $\mathbf{v}$ into $\mathbf{B}$.

4 Gram–Schmidt orthogonalization for C

In this section, we compute the Gram–Schmidt vectors of the basis $\mathbf{C}$ defined in (1.2) and their explicit lengths. Given a basis $\mathbf{B}=[\mathbf{b}_1,\dots,\mathbf{b}_n]$ of a lattice L, let $[\mathbf{b}_1^*,\dots,\mathbf{b}_n^*]$ be its Gram–Schmidt vectors. The basis $\mathbf{B}$ is not

necessarily LLL-reduced. Let $\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{b}_i^* \in L$. At the beginning of this section, we do not assume $v_n = 1$. Let k be the insertion index of $\mathbf{v}$. We consider the Gram–Schmidt orthogonalization for the $n+1$ vectors $[\mathbf{b}_1, \dots, \mathbf{b}_{k-1}, \mathbf{v}, \mathbf{b}_k, \dots, \mathbf{b}_n]$. Let $[\mathbf{b}_1^*, \dots, \mathbf{b}_{k-1}^*, \mathbf{c}_k^*, \mathbf{c}_{k+1}^*, \dots, \mathbf{c}_{n+1}^*]$ denote its Gram–Schmidt vectors. By formula (2.1), it is clear that the first $k-1$ vectors are the same as the first $k-1$ vectors of $[\mathbf{b}_1^*, \dots, \mathbf{b}_n^*]$.

Lemma 4.1. *For the vectors $\mathbf{c}_k^*, \mathbf{c}_{k+1}^*, \dots, \mathbf{c}_{n+1}^*$, we have the following:*
(i) *The vectors $[\mathbf{c}_k^*, \mathbf{c}_{k+1}^*, \dots, \mathbf{c}_{n+1}^*]$ are the same as the Gram–Schmidt orthogonalization for*

$$[\pi_k(\mathbf{v}), \pi_k(\mathbf{b}_k), \dots, \pi_k(\mathbf{b}_n)].$$

(ii) *The Gram–Schmidt orthogonalization for $[\pi_k(\mathbf{v}), \pi_k(\mathbf{b}_k), \dots, \pi_k(\mathbf{b}_n)]$ is the same as that for*

$$[\pi_k(\mathbf{v}), \mathbf{b}_k^*, \dots, \mathbf{b}_n^*].$$

Proof. Assertion (i) is clear from recursive formula (2.1). For (ii), we let

$$[\mathbf{a}_0, \mathbf{a}_1, \dots, \mathbf{a}_{n-k+1}] = [\pi_k(\mathbf{v}), \pi_k(\mathbf{b}_k), \dots, \pi_k(\mathbf{b}_n)],$$
$$[\mathbf{a}_0', \mathbf{a}_1', \dots, \mathbf{a}_{n-k+1}'] = [\pi_k(\mathbf{v}), \mathbf{b}_k^*, \dots, \mathbf{b}_n^*].$$

It is sufficient to show $\mathbf{a}_i^* = \mathbf{a}_i'^*$ for any $0 \le i \le n-k+1$. The case $i = 0$ is clear since $\mathbf{a}_0 = \mathbf{a}_0'$. The case $i = 1$ is also clear since $\pi_k(\mathbf{b}_k) = \mathbf{b}_k^*$, and we have $\mathbf{b}_k^* \in \langle \mathbf{a}_0^*, \mathbf{a}_1^* \rangle_{\mathbb{R}}$. For some $1 \le \ell \le n-k$, we assume $\mathbf{a}_i^* = \mathbf{a}_i'^*$ and $\mathbf{b}_{k+i-1}^* \in W_i$ for all $1 \le i \le \ell$, where $W_i = \langle \mathbf{a}_0^*, \dots, \mathbf{a}_i^* \rangle_{\mathbb{R}}$ for each $0 \le i \le n-k+1$. Now we consider the case $i = \ell + 1$. By the assumption, we have

$$\mathbf{a}_{\ell+1}^* - \mathbf{a}_{\ell+1}'^* = (\mathbf{a}_{\ell+1} - \mathbf{a}_{\ell+1}')^* = \pi_k(\mathbf{b}_{k+\ell}) - \mathbf{b}_{k+\ell}^* - \sum_{j=0}^{\ell} \frac{\langle \pi_k(\mathbf{b}_{k+\ell}) - \mathbf{b}_{k+\ell}^*, \mathbf{a}_j^* \rangle}{\|\mathbf{a}_j^*\|^2} \mathbf{a}_j^*.$$

Since $\pi_k(\mathbf{b}_{k+\ell}) - \mathbf{b}_{k+\ell}^* \in \langle \mathbf{b}_k^*, \dots, \mathbf{b}_{k+\ell-1}^* \rangle_{\mathbb{R}}$ and $\mathbf{b}_k^*, \dots, \mathbf{b}_{k+\ell-1}^* \in W_\ell = \langle \mathbf{a}_0^*, \dots, \mathbf{a}_\ell^* \rangle_{\mathbb{R}}$, we have

$$\mathbf{a}_{\ell+1}^* - \mathbf{a}_{\ell+1}'^* \in W_\ell.$$

We also have $\mathbf{a}_{\ell+1}^* - \mathbf{a}_{\ell+1}'^* \in W_\ell^\perp$, and hence $\mathbf{a}_{\ell+1}^* = \mathbf{a}_{\ell+1}'^*$ since $W_\ell \cap W_\ell^\perp = \{\mathbf{0}\}$. Furthermore, $\mathbf{b}_{k+\ell}^* \in W_{\ell+1}$, which completes the proof by induction. □

By Lemma 4.1, we obtain the following result on $\mathbf{c}_k^*, \mathbf{c}_{k+1}^*, \dots, \mathbf{c}_{n+1}^*$.

Proposition 4.2. *Set $m = \max\{k \le i \le n \mid v_i \ne 0\}$. Then we have*

$$\mathbf{c}_j^* = \begin{cases} \displaystyle\sum_{i=k}^{m} v_i \mathbf{b}_i^* & \text{for } j = k, \\ \displaystyle\frac{D_j}{D_{j-1}} \mathbf{b}_{j-1}^* - \sum_{i=j}^{m} \frac{v_i v_{j-1} \|\mathbf{b}_{j-1}^*\|^2}{D_{j-1}} \mathbf{b}_i^* & \text{for } k+1 \le j \le m+1, \\ \mathbf{b}_{j-1}^* & \text{for } m+2 \le j \le n+1, \end{cases}$$

where for $1 \le \ell \le m$ we set

$$D_\ell = \sum_{i=\ell}^{m} v_i^2 \|\mathbf{b}_i^*\|^2. \tag{4.1}$$

In particular, we have $\mathbf{c}_{m+1}^* = \mathbf{0}$. Moreover, we have $\|\mathbf{c}_k^*\|^2 = D_k$, and for $k+1 \le j \le m$,

$$\|\mathbf{c}_j^*\|^2 = \frac{D_j}{D_{j-1}} \|\mathbf{b}_{j-1}^*\|^2.$$

Proof. By Lemma 4.1, we have $\mathbf{c}_k^* = \pi_k(\mathbf{v}) = \sum_{i=k}^{m} v_i \mathbf{b}_i^*$. For the case $m+2 \le j \le n+1$, since

$$\langle \mathbf{c}_k^*, \mathbf{b}_k^*, \dots, \mathbf{b}_m^* \rangle_{\mathbb{R}} = \langle \mathbf{b}_k^*, \dots, \mathbf{b}_m^* \rangle_{\mathbb{R}},$$

the $\mathbb{R}$-vector space $\langle \mathbf{b}_{m+1}^*, \dots, \mathbf{b}_n^* \rangle_{\mathbb{R}}$ is orthogonal to $\langle \mathbf{b}_k^*, \dots, \mathbf{b}_m^* \rangle_{\mathbb{R}}$. Therefore the vectors $\mathbf{b}_{m+1}^*, \dots, \mathbf{b}_n^*$ are unchanged after the insertion of $\mathbf{v}$. For the case $k+1 \le j \le m+1$, let us begin with the simple case $j = k+1$.

By Lemma 4.1, we have

$$\begin{aligned}\mathbf{c}_{k+1}^* &= \mathbf{b}_k^* - \frac{\langle \mathbf{b}_k^*, \mathbf{c}_k^* \rangle}{\|\mathbf{c}_k^*\|} \mathbf{c}_k^* \\ &= \mathbf{b}_k^* - \frac{\nu_k \|\mathbf{b}_k^*\|^2}{D_k} \sum_{i=k}^{m} \nu_i \mathbf{b}_i^* \\ &= \Big(1 - \frac{\nu_k^2 \|\mathbf{b}_k^*\|^2}{D_k}\Big) \mathbf{b}_k^* - \sum_{i=k+1}^{m} \frac{\nu_k \nu_i \|\mathbf{b}_k^*\|^2}{D_k} \mathbf{b}_i^* \\ &= \frac{D_{k+1}}{D_k} \mathbf{b}_k^* - \sum_{i=k+1}^{m} \frac{\nu_k \nu_i \|\mathbf{b}_k^*\|^2}{D_k} \mathbf{b}_i^*.\end{aligned}$$

This completes the proof of the case $j = k + 1$. For each $k + 1 \le j \le m + 1$, set

$$\mathbf{a}_j = \frac{D_j}{D_{j-1}} \mathbf{b}_{j-1}^* - \sum_{i=j}^{m} \frac{\nu_i \nu_{j-1} \|\mathbf{b}_{j-1}^2\|^2}{D_{j-1}} \mathbf{b}_i^*$$

and we shall show $\mathbf{c}_j^* = \mathbf{a}_j$. For $k + 1 \le \ell \le m$, we assume $\mathbf{c}_j^* = \mathbf{a}_j$ for all $k + 1 \le j \le \ell$. Now let us consider the case $j = \ell + 1$. Set $W_\ell = \langle \mathbf{c}_k^*, \dots, \mathbf{c}_\ell^* \rangle_{\mathbb{R}}$ as in the proof of Lemma 4.1, and we begin to show that $\mathbf{a}_{\ell+1}$ is orthogonal to W_ℓ. Actually, we have

$$\begin{aligned}\langle \mathbf{a}_{\ell+1}, \mathbf{c}_k^* \rangle &= \Big\langle \frac{D_{\ell+1}}{D_\ell} \mathbf{b}_\ell^* - \sum_{i=\ell+1}^{m} \frac{\nu_i \nu_\ell \|\mathbf{b}_\ell^*\|^2}{D_\ell} \mathbf{b}_i^*, \sum_{j=k}^{m} \nu_j \mathbf{b}_j^* \Big\rangle \\ &= \frac{\nu_\ell D_{\ell+1} \|\mathbf{b}_\ell^*\|^2}{D_\ell} - \sum_{i=\ell+1}^{m} \frac{\nu_\ell \nu_i^2 \|\mathbf{b}_\ell^*\|^2 \|\mathbf{b}_i^*\|^2}{D_\ell} \\ &= \frac{\nu_\ell \|\mathbf{b}_\ell^*\|^2}{D_\ell} \times \Big(D_{\ell+1} - \sum_{i=\ell+1}^{m} \nu_i^2 \|\mathbf{b}_i^*\|^2 \Big) = 0.\end{aligned}$$

Furthermore, for any $k + 1 \le j \le \ell$, we have

$$\begin{aligned}\langle \mathbf{a}_{\ell+1}, \mathbf{c}_j^* \rangle &= \Big\langle \frac{D_{\ell+1}}{D_\ell} \mathbf{b}_\ell^* - \sum_{i=\ell+1}^{m} \frac{\nu_i \nu_\ell \|\mathbf{b}_\ell^*\|^2}{D_\ell} \mathbf{b}_i^*, \frac{D_j}{D_{j-1}} \mathbf{b}_{j-1}^* - \sum_{i=j}^{m} \frac{\nu_i \nu_{j-1} \|\mathbf{b}_{j-1}^*\|^2}{D_{j-1}} \mathbf{b}_i^* \Big\rangle \\ &= -\frac{D_{\ell+1} \nu_\ell \nu_{j-1} \|\mathbf{b}_{j-1}^*\|^2 \|\mathbf{b}_\ell^*\|^2}{D_\ell D_{\ell-1}} + \sum_{i=\ell+1}^{m} \frac{\nu_i^2 \nu_\ell \nu_{j-1} \|\mathbf{b}_\ell^*\|^2 \|\mathbf{b}_{j-1}^*\|^2 \|\mathbf{b}_i^*\|^2}{D_\ell D_{\ell-1}} \\ &= \frac{\nu_\ell \nu_{j-1} \|\mathbf{b}_{j-1}^*\|^2 \|\mathbf{b}_\ell\|^2}{D_\ell D_{\ell-1}} \Big(\sum_{i=\ell+1}^{m} \nu_i^2 \|\mathbf{b}_i^*\|^2 - D_{\ell+1} \Big) = 0.\end{aligned}$$

Therefore $\mathbf{a}_{\ell+1}$ is orthogonal to W_ℓ. On the other hand, since

$$\mathbf{c}_{\ell+1}^* = \mathbf{b}_\ell^* - \sum_{i=k}^{\ell} \frac{\langle \mathbf{b}_\ell^*, \mathbf{c}_i^* \rangle}{\|\mathbf{c}_i^*\|^2} \mathbf{c}_i^*$$

by Lemma 4.1, we have

$$\begin{aligned}\mathbf{c}_{\ell+1}^* - \mathbf{a}_{\ell+1} &= \Big(\mathbf{b}_\ell^* - \sum_{i=k}^{\ell} \frac{\langle \mathbf{b}_\ell^*, \mathbf{c}_i^* \rangle}{\|\mathbf{c}_i^*\|^2} \mathbf{c}_i^* \Big) - \Big(\frac{D_{\ell+1}}{D_\ell} \mathbf{b}_\ell^* - \sum_{i=\ell+1}^{m} \frac{\nu_i \nu_\ell \|\mathbf{b}_\ell^*\|^2}{D_\ell} \mathbf{b}_i^* \Big) \\ &= \frac{D_\ell - D_{\ell+1}}{D_\ell} \mathbf{b}_\ell^* + \sum_{i=\ell+1}^{m} \frac{\nu_i \nu_\ell \|\mathbf{b}_\ell^*\|^2}{D_\ell} \mathbf{b}_i^* - \sum_{i=k}^{\ell} \frac{\langle \mathbf{b}_\ell^*, \mathbf{c}_i^* \rangle}{\|\mathbf{c}_i^*\|^2} \mathbf{c}_i^* \\ &= \frac{\nu_\ell \|\mathbf{b}_\ell^*\|^2}{D_\ell} \sum_{i=\ell}^{m} \nu_i \mathbf{b}_i^* - \sum_{i=k}^{\ell} \frac{\langle \mathbf{b}_\ell^*, \mathbf{c}_i^* \rangle}{\|\mathbf{c}_i^*\|^2} \mathbf{c}_i^* \\ &= \frac{\nu_\ell \|\mathbf{b}_\ell^*\|^2}{D_\ell} \Big(\mathbf{c}_k^* - \sum_{i=k}^{\ell-1} \nu_i \mathbf{b}_i^* \Big) - \sum_{i=k}^{\ell} \frac{\langle \mathbf{b}_\ell^*, \mathbf{c}_i^* \rangle}{\|\mathbf{c}_i^*\|^2} \mathbf{c}_i^*.\end{aligned}$$

Since $\mathbf{b}_i^* \in W_\ell$ for any $k \le i \le \ell - 1$ by the proof of Lemma 4.1, we have $\mathbf{c}_{\ell+1}^* - \mathbf{a}_{\ell+1} \in W_\ell$. By the above arguments, we have $\mathbf{c}_{\ell+1}^* - \mathbf{a}_{\ell+1} \in W_\ell^\perp \cap W_\ell = \{\mathbf{0}\}$ and hence $\mathbf{c}_{\ell+1}^* = \mathbf{a}_{\ell+1}$. This shows $\mathbf{c}_j^* = \mathbf{a}_j$ for $k + 1 \le j \le m + 1$

by induction. Finally, for $k+1 \le j \le m$, we have

$$\begin{aligned}\|\mathbf{c}_j^*\|^2 &= \frac{D_j^2}{D_{j-1}^2}\|\mathbf{b}_{j-1}^*\|^2 + \sum_{i=j}^{m}\frac{\nu_i^2\nu_{j-1}^2\|\mathbf{b}_{j-1}\|^4}{D_{j-1}^2}\|\mathbf{b}_i^*\|^2\\ &= \frac{\|\mathbf{b}_{j-1}^*\|^2}{D_{j-1}^2}(D_j^2 + \nu_{j-1}^2\|\mathbf{b}_{j-1}^*\|^2 D_j)\\ &= \frac{D_j\|\mathbf{b}_{j-1}^*\|^2}{D_{j-1}^2}D_{j-1}\\ &= \frac{D_j}{D_{j-1}}\|\mathbf{b}_{j-1}^*\|^2,\end{aligned}$$

where the second equation is by $D_j = \sum_{i=j}^{m}\nu_i^2\|\mathbf{b}_i^*\|^2$, and the third by $D_{j-1} = D_j + \nu_{j-1}^2\|\mathbf{b}_{j-1}^*\|^2$. This completes the proof of Proposition 4.2. □

By inserting $\mathbf{v} = \sum_{i=1}^{n}\nu_i\mathbf{b}_i^*$ into $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ at the k-th position, the n vectors

$$[\mathbf{b}_1, \dots, \mathbf{b}_{k-1}, \mathbf{v}, \mathbf{b}_k, \dots, \mathbf{b}_{m-1}, \mathbf{b}_{m+1}, \dots, \mathbf{b}_n] \tag{4.2}$$

give a basis of a *sub-lattice* of L (recall $m = \max\{k \le i \le n \mid \nu_i \ne 0\}$). In contrast, we have the following result on a basis of the whole lattice L.

Proposition 4.3. *If $\nu_m = 1$, the vectors* (4.2) *give a basis of the whole lattice L.*

Proof. Let L' denote the sub-lattice with basis (4.2). By Proposition 4.2, we have

$$\begin{aligned}\frac{\mathrm{vol}(L')^2}{\mathrm{vol}(L)^2} &= \prod_{i=k}^{m}\frac{\|\mathbf{c}_i^*\|^2}{\|\mathbf{b}_i^*\|^2}\\ &= D_k \times \frac{D_{k+1}}{D_k} \times \cdots \times \frac{D_m}{D_{m-1}} \times \frac{1}{\|\mathbf{b}_m^*\|^2}\\ &= \frac{D_m}{\|\mathbf{b}_m^*\|^2}.\end{aligned}$$

If $\nu_m = 1$, we have $D_m = \|\mathbf{b}_m^*\|^2$ by definition. In this case, we have $\mathrm{vol}(L') = \mathrm{vol}(L)$, and hence $L = L'$. □

Henceforth, as in (1.2), we always take $\mathbf{v} = \sum_{i=1}^{n}\nu_i\mathbf{b}_i^*$ with $\nu_n = 1$. By Proposition 4.3, the n vectors $[\mathbf{b}_1, \dots, \mathbf{b}_{k-1}, \mathbf{v}, \mathbf{b}_k, \dots, \mathbf{b}_{n-1}]$ give a basis of L. Then we take the basis $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_n]$ as in (1.2), namely, $\mathbf{c}_i = \mathbf{b}_i$ for $1 \le i \le k-1$, $\mathbf{c}_k = \mathbf{v}$, and $\mathbf{c}_j = \mathbf{b}_{j-1}$ for $k+1 \le j \le n$. By Proposition 4.2, we obtain the explicit gap between two squared-sums $\mathrm{SS}(\mathbf{B})$ and $\mathrm{SS}(\mathbf{C})$ as follows.

Theorem 4.4. *The explicit gap between* $\mathrm{SS}(\mathbf{B})$ *and* $\mathrm{SS}(\mathbf{C})$ *is given by*

$$E(\mathbf{v}, k) := \mathrm{SS}(\mathbf{B}) - \mathrm{SS}(\mathbf{C}) = \sum_{j=k}^{n-1}\nu_j^2\|\mathbf{b}_j^*\|^2\Big(\frac{\|\mathbf{b}_j^*\|^2}{D_j} - 1\Big). \tag{4.3}$$

Proof. By Proposition 4.2, we have

$$\begin{aligned}E(\mathbf{v}, k) &= \sum_{i=k}^{n}\|\mathbf{b}_i^*\|^2 - \Big(D_k + \sum_{j=k+1}^{n}\frac{D_j}{D_{j-1}}\|\mathbf{b}_{j-1}^*\|^2\Big)\\ &= -D_k + \sum_{j=k+1}^{n}\Big(1 - \frac{D_j}{D_{j-1}}\Big)\|\mathbf{b}_{j-1}^*\|^2 + \|\mathbf{b}_n^*\|^2\\ &= -\sum_{j=k}^{n-1}\nu_j^2\|\mathbf{b}_j^*\|^2 + \sum_{j=k+1}^{n}\frac{\nu_{j-1}^2\|\mathbf{b}_{j-1}^*\|^2}{D_{j-1}}\|\mathbf{b}_{j-1}^*\|^2\\ &= \sum_{j=k}^{n-1}\nu_j^2\|\mathbf{b}_j^*\|^2\Big(\frac{\|\mathbf{b}_j^*\|^2}{D_j} - 1\Big)\end{aligned}$$

since $D_n = \|\mathbf{b}_n^*\|^2$ by setting $\nu_n = 1$. □

Remark 4.5. To strictly decrease the squared-sum $\mathrm{SS}(\mathbf{B})$, we may take $\mathbf{v} \in L$ satisfying $E(\mathbf{v}, k) > 0$. However, in most cases, the value $E(\mathbf{v}, k)$ is negative when the inserted vector $\mathbf{v}$ is generated by Schnorr's SA. Below, we shall consider only the case $E(\mathbf{v}, k) \le 0$.

For the original basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$, let

$$\mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\|\mathbf{b}_j^*\|^2}$$

be as in (2.1). For $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_n]$ with $\mathbf{c}_k = \mathbf{v} = \sum_{i=1}^n v_i \mathbf{b}_i^* \in L$, we set

$$\xi_{i,j} = \frac{\langle \mathbf{c}_i, \mathbf{c}_j^* \rangle}{\|\mathbf{c}_j^*\|^2} \quad \text{for } i > j. \tag{4.4}$$

For simplicity, we set $\xi_{i,i} = 1$ for all $1 \le i \le n$.

Proposition 4.6. *We have the following on $\xi_{i,j}$:*
(i) *For $1 \le i \le k-1$, we have $\xi_{i,j} = \mu_{i,j}$ for $i \ge j$.*
(ii) *For $i = k$, we have $\xi_{k,j} = v_j$ for $1 \le j \le k-1$.*
(iii) *For $i \ge k+1$, we have*

$$\xi_{i,j} = \begin{cases} \mu_{i-1,j} & \text{for } j \le k-1, \\ \dfrac{\sum_{\ell=k}^{i-1} \mu_{i-1,\ell} v_\ell \|\mathbf{b}_\ell^*\|^2}{D_k} & \text{for } j = k, \\ \mu_{i-1,j-1} - \dfrac{\sum_{\ell=j}^{i-1} \mu_{i-1,\ell} v_\ell v_{j-1} \|\mathbf{b}_\ell^*\|^2}{D_j} & \text{for } k < j < i. \end{cases}$$

Proof. Cases (i) and (ii) are trivial. Case (iii) is trivial for $j \le k$. For $k < j < i$, by Proposition 4.2, we have

$$\begin{aligned} \xi_{i,j} &= \frac{D_{j-1}}{D_j \|\mathbf{b}_{j-1}^*\|^2} \left\langle \mathbf{b}_{i-1}, \frac{D_j}{D_{j-1}} \mathbf{b}_{j-1}^* - \sum_{\ell=j}^{n} \frac{v_\ell v_{j-1} \|\mathbf{b}_{j-1}^*\|^2}{D_{j-1}} \mathbf{b}_\ell^* \right\rangle \\ &= \mu_{i-1,j-1} - \frac{v_{j-1}}{D_j} \left\langle \mathbf{b}_{i-1}, \sum_{\ell=j}^{n} v_\ell \mathbf{b}_\ell^* \right\rangle \\ &= \mu_{i-1,j-1} - \frac{v_{j-1}}{D_j} \sum_{\ell=j}^{i-1} \mu_{i-1,\ell} v_\ell \|\mathbf{b}_\ell^*\|^2. \end{aligned}$$

This completes the proof of Proposition 4.6. □

5 LLL-reduction for general bases

Before analyzing the LLL-reduction for the basis $\mathbf{C}$, we study the behavior of the LLL-reduction for general bases and give some basic properties in this section. Let $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_n]$ be a basis of a lattice L. The LLL algorithm for $\mathbf{S}$ consists of the following two steps (see, e.g., [15, Chapter 2, Algorithm 6]):

From $i = 2$ to n, do:

- *Step 1*. Size-reduce $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_n]$ (see, e.g., [15, Chapter 2, Algorithm 3]). Note that this procedure does not change the lengths of the Gram–Schmidt vectors $[\mathbf{s}_1^*, \dots, \mathbf{s}_n^*]$ by [6, Lemma 17.4.1].
- *Step 2*. Swap $\mathbf{s}_i$ with $\mathbf{s}_{i-1}$ if the Lovász condition

$$\|\mathbf{s}_i^*\|^2 \ge (\delta - \eta_{i,i-1}^2) \|\mathbf{s}_{i-1}^*\|^2 \tag{5.1}$$

is not satisfied, where δ is the reduction parameter of LLL satisfying $\frac{1}{4} < \delta < 1$ (we used $\delta = 0.99$ in our experiments) and

$$\eta_{i,j} = \frac{\langle \mathbf{s}_i, \mathbf{s}_j^* \rangle}{\|\mathbf{s}_j^*\|^2} \quad \text{for } i > j.$$

In this case, set $i \leftarrow \max\{2, i-1\}$. Otherwise set $i \leftarrow i+1$. Then go back to Step 1.

In the following, we give a key lemma [6, Lemma 17.4.3] on our analysis of the behavior of the LLL algorithm.

Lemma 5.1. *Given a basis* $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_n]$ *and an integer* $1 \le \ell \le n-1$, *assume that the pair* $(\mathbf{s}_\ell, \mathbf{s}_{\ell+1})$ *does not satisfy the Lovász condition. Let* $\mathbf{T} = [\mathbf{t}_1, \dots, \mathbf{t}_n]$ *be the new basis obtained by swapping* $\mathbf{s}_\ell$ *and* $\mathbf{s}_{\ell+1}$, *namely,* $\mathbf{t}_\ell = \mathbf{s}_{\ell+1}$ *and* $\mathbf{t}_{\ell+1} = \mathbf{s}_\ell$. *Then the Gram–Schmidt vectors* $[\mathbf{t}_1^*, \dots, \mathbf{t}_n^*]$ *of* $\mathbf{T}$ *are as follows:*

(i) *For* $1 \le i < \ell$ *and* $\ell + 1 < i \le n$, *the vector* $\mathbf{s}_i^*$ *is unchanged (i.e.* $\mathbf{t}_i^* = \mathbf{s}_i^*$*).*

(ii) *The Gram–Schmidt vector* $\mathbf{t}_\ell^*$ *and its squared-length are given, respectively, by*

$$\mathbf{t}_\ell^* = \mathbf{s}_{\ell+1}^* + \eta_{\ell+1,\ell}\mathbf{s}_\ell^* \quad \text{and} \quad T_\ell = S_{\ell+1} + \eta_{\ell+1,\ell}^2 S_\ell,$$

where we set $S_i = \|\mathbf{s}_i^*\|^2$ *and* $T_i = \|\mathbf{t}_i^*\|^2$ *for* $1 \le i \le n$.

(iii) *The Gram–Schmidt vector* $\mathbf{t}_{\ell+1}^*$ *and its squared-length are given, respectively, by*

$$\mathbf{t}_{\ell+1}^* = \frac{S_{\ell+1}}{T_\ell}\mathbf{s}_\ell^* - \frac{\eta_{\ell+1,\ell}S_\ell}{T_\ell}\mathbf{s}_{\ell+1}^* \quad \text{and} \quad T_{\ell+1} = \frac{S_\ell S_{\ell+1}}{T_\ell}.$$

Moreover, if we set

$$\delta_\ell := \frac{T_\ell}{S_\ell} = \frac{S_{\ell+1}}{S_\ell} + \eta_{\ell+1,\ell}^2, \tag{5.2}$$

then $\delta_\ell < \delta$ *since we assume that the pair* $(\mathbf{s}_\ell, \mathbf{s}_{\ell+1})$ *does not satisfy* (5.1).

Now we consider the LLL-reduction for $\mathbf{S}$. Recall that the size-reduce procedure does not change the lengths of the Gram–Schmidt vectors. By Lemma 5.1, we see that each swap in the LLL algorithm can strictly decrease the sum $\mathrm{SS}(\mathbf{S})$. Specifically, the decreasing value of $\mathrm{SS}(\mathbf{S}) = \sum_{i=1}^n S_i$ by one time swap at the ℓ-th index is estimated as follows:

Lemma 5.2. *Let* $\mathbf{S}$ *and* $\mathbf{T}$ *be as in Lemma 5.1. Then we have*

$$\mathrm{SS}(\mathbf{S}) - \mathrm{SS}(\mathbf{T}) = \frac{\eta_{\ell+1,\ell}^2(1-\delta_\ell)}{\delta_\ell}S_\ell > \frac{\eta_{\ell+1,\ell}^2(1-\delta)}{\delta}S_\ell > 0,$$

by $\delta_\ell < \delta < 1$. *Namely, the squared-sum* $\mathrm{SS}(\mathbf{S})$ *strictly decreases by each swap in the LLL algorithm.*

Proof. By Lemma 5.1, we have

$$\begin{aligned}
\mathrm{SS}(\mathbf{S}) - \mathrm{SS}(\mathbf{T}) &= (1-\eta_{\ell+1,\ell}^2)S_\ell - \frac{S_\ell S_{\ell+1}}{T_\ell} \\
&= \frac{S_\ell}{T_\ell}\{(1-\eta_{\ell+1,\ell}^2)T_\ell - S_{\ell+1}\} \\
&= \frac{S_\ell}{T_\ell}\{(1-\eta_{\ell+1,\ell}^2)(S_{\ell+1} + \eta_{\ell+1,\ell}^2 S_\ell) - S_{\ell+1}\} \\
&= \frac{\eta_{\ell+1,\ell}^2 S_\ell}{T_\ell}\{(1-\eta_{\ell+1,\ell}^2)S_\ell - S_{\ell+1}\} \\
&= \frac{\eta_{\ell+1,\ell}^2 S_\ell}{\delta_\ell}\left\{1 - \left(\eta_{\ell+1,\ell}^2 + \frac{S_{\ell+1}}{S_\ell}\right)\right\} \\
&= \frac{\eta_{\ell+1,\ell}^2(1-\delta_\ell)}{\delta_\ell}S_\ell > \frac{\eta_{\ell+1,\ell}^2(1-\delta)}{\delta}S_\ell > 0.
\end{aligned}$$

This completes the proof of Lemma 5.2. □

In the following, let us give the definition of the loop invariant of a lattice basis [3, Definition 4.15].

Definition 5.3 (Loop invariant). The loop invariant of a basis $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_n]$ is defined as the quantity

$$\mathcal{LI}(\mathbf{S}) = \prod_{i=1}^{n-1}\left(\prod_{\ell=1}^{i}\|\mathbf{s}_\ell^*\|^2\right) = \prod_{i=1}^{n-1}\|\mathbf{s}_i^*\|^{2n-2i}.$$

The loop invariant plays an important role in estimating the number of swaps in the LLL algorithm. As in Lemma 5.1, for $1 \le \ell \le n-1$, we consider a case of swapping $(\mathbf{s}_\ell, \mathbf{s}_{\ell+1})$ to obtain a new basis $\mathbf{T} = [\mathbf{t}_1, \dots, \mathbf{t}_n]$ with $\mathbf{t}_\ell = \mathbf{s}_{\ell+1}$ and $\mathbf{t}_{\ell+1} = \mathbf{s}_\ell$. In this case, we clearly have

$$\|\mathbf{t}_\ell^*\|^2 \cdot \|\mathbf{t}_{\ell+1}^*\|^2 = \|\mathbf{s}_\ell^*\|^2 \cdot \|\mathbf{s}_{\ell+1}^*\|^2.$$

Then we have

$$\mathcal{LI}(\mathbf{T}) = \mathcal{LI}(\mathbf{S}) \times \frac{\|\mathbf{t}_\ell^*\|^2}{\|\mathbf{s}_\ell^*\|^2} = \mathcal{LI}(\mathbf{S}) \times \delta_\ell. \tag{5.3}$$

Thus the loop invariant $\mathcal{LI}(\mathbf{S})$ is reduced by the factor of δ_ℓ by each swap in the LLL algorithm.

5.1 Whole LLL procedure

In this subsection, we consider the whole LLL procedure for a basis $\mathbf{A}$. Now we assume the following on the whole LLL procedure for $\mathbf{A}$, where we denote the output by $\mathbf{A}' \leftarrow \mathrm{LLL}(\mathbf{A})$:

- In total, N swaps occur in the LLL procedure.
- For $0 \le s \le N$, by $\mathbf{A}^{(s)} = [\mathbf{a}_1^{(s)}, \dots, \mathbf{a}_n^{(s)}]$ we denote the basis obtained by the s-th swap and size-reduced, and set $\mathbf{A}^{(0)} = \mathbf{A}$. Let $A_i^{(s)} = \|\mathbf{a}_i^{(s)*}\|^2$ for $1 \le i \le n$, where $\mathbf{a}_1^{(s)*}, \dots, \mathbf{a}_n^{(s)*}$ denote the Gram–Schmidt vectors of $\mathbf{A}^{(s)}$. Then $\mathbf{A} = \mathbf{A}^{(0)}$ and $\mathbf{A}' = \mathbf{A}^{(N)} \leftarrow \mathrm{LLL}(\mathbf{A})$.
- By $\ell(s)$ we denote the index where the s-th swap occurs, that is, the $\ell(s)$-th vector $\mathbf{a}_{\ell(s)}^{(s-1)}$ and $(\ell(s)+1)$-st vector $\mathbf{a}_{\ell(s)+1}^{(s-1)}$ is swapped. We call $\ell(s)$ the *s-th swap index*.
- For $1 \le s \le N$, we let
$$\xi_{\ell(s)+1,\ell(s)}^{(s)} = \frac{\langle \mathbf{a}_{\ell(s)+1}^{(s-1)}, \mathbf{a}_{\ell(s)}^{(s-1)*} \rangle}{\|\mathbf{a}_{\ell(s)}^{(s-1)*}\|^2} \quad \text{and} \quad \delta_{\ell(s)}^{(s)} = \frac{A_{\ell(s)}^{(s)}}{A_{\ell(s)}^{(s-1)}}.$$
We call $\xi_{\ell(s)+1,\ell(s)}^{(s)}$ the *normalized inner product of the swap vectors* and $\delta_{\ell(s)}^{(s)}$ *the swap ratio* at the s-th swap (see equation (5.2) for the swap ratio), and write $\xi(s)$ and $\delta(s)$ for simplicity.

Lemma 5.4. *For each $1 \le s \le N$ and $1 \le i \le n$, there exist indices $m(s,i)$ and $M(s,i)$ in the set $\{1, \dots, n\}$ such that*

$$A_{M(s,i)} \ge A_i^{(s)} \ge A_{m(s,i)}.$$

We have $M(s,\ell(s)) = M(s,\ell(s)+1) = M(s-1,\ell(s))$ and $m(s,\ell(s)) = m(s,\ell(s)+1) = m(s-1,\ell(s)+1)$.

Proof. We use an induction argument on s. By setting $m(0,i) = M(0,i) = i$, the case $s = 0$ is clear. We assume that Lemma 5.4 holds for the case s and now consider the case $s+1$. For $i \ne \ell(s+1), \ell(s+1)+1$, we have $A_i^{(s+1)} = A_i^{(s)}$ and hence $m(s+1,i) = m(s,i)$ and $M(s+1,i) = M(s,i)$. By Lemma 5.1, we clearly have

$$\begin{aligned}
A_{\ell(s+1)}^{(s+1)} &= \delta(s+1) A_{\ell(s+1)}^{(s)} < A_{\ell(s+1)}^{(s)}, \\
A_{\ell(s+1)+1}^{(s+1)} &= \frac{A_{\ell(s+1)}^{(s)} A_{\ell(s+1)+1}^{(s)}}{A_{\ell(s+1)}^{(s+1)}} = \frac{1}{\delta(s+1)} A_{\ell(s+1)+1}^{(s)} > A_{\ell(s+1)+1}^{(s)}, \\
A_{\ell(s+1)}^{(s+1)} &= A_{\ell(s+1)+1}^{(s)} + \xi(s+1)^2 A_{\ell(s+1)}^{(s)} \ge A_{\ell(s+1)+1}^{(s)}, \\
A_{\ell(s+1)+1}^{(s+1)} &= \frac{A_{\ell(s+1)}^{(s)} A_{\ell(s+1)+1}^{(s)}}{A_{\ell(s+1)}^{(s+1)}} \le \frac{A_{\ell(s+1)}^{(s)} A_{\ell(s+1)+1}^{(s)}}{A_{\ell(s+1)+1}^{(s)}} = A_{\ell(s+1)}^{(s)}.
\end{aligned}$$

Thus, we may set

$$\begin{aligned}
m(s+1,\ell(s+1)) &= m(s+1,\ell(s+1)+1) = m(s,\ell(s+1)+1), \\
M(s+1,\ell(s+1)) &= M(s+1,\ell(s+1)+1) = M(s,\ell(s+1)).
\end{aligned}$$

This completes the proof of Lemma 5.4 by induction. □

Recall that the basis $\mathbf{A}^{(s+1)}$ is obtained by swapping the $\ell(s+1)$-th and $(\ell(s+1)+1)$-st vectors of $\mathbf{A}^{(s)}$. We obtain the following result on the gap of squared-sums of Gram–Schmidt lengths of $\mathbf{A}^{(s)}$ and $\mathbf{A}^{(s+1)}$.

Lemma 5.5. *The gap* $\mathrm{SS}(\mathbf{A}^{(s)}) - \mathrm{SS}(\mathbf{A}^{(s+1)})$ *is greater than*

$$\frac{\xi(s+1)^2(1-\delta(s+1))}{\delta(s+1)} A_{m(s,\ell(s+1))} \quad \text{or} \quad \frac{\xi(s+1)^2(1-\delta)}{\delta} A_{m(s,\ell(s+1))},$$

where $A_m = \|\mathbf{a}_m^*\|^2$ *with index* $m = m(s, \ell(s+1))$.

Proof. By Lemmas 5.2 and 5.4, the gap $\mathrm{SS}(\mathbf{A}^{(s)}) - \mathrm{SS}(\mathbf{A}^{(s+1)})$ is given by

$$\begin{aligned}
\mathrm{SS}(\mathbf{A}^{(s)}) - \mathrm{SS}(\mathbf{A}^{(s+1)}) &= (A^{(s)}_{\ell(s+1)} + A^{(s)}_{\ell(s+1)+1}) - (A^{(s+1)}_{\ell(s+1)} + A^{(s+1)}_{\ell(s+1)+1}) \\
&= \frac{\xi(s+1)^2(1-\delta(s+1))}{\delta(s+1)} A^{(s)}_{\ell(s+1)} \\
&> \frac{\xi(s+1)^2(1-\delta(s+1))}{\delta(s+1)} A_{m(s,\ell(s+1))} \\
&> \frac{\xi(s+1)^2(1-\delta)}{\delta} A_{m(s,\ell(s+1))}
\end{aligned}$$

since $\delta(s+1) < \delta$ by Lemma 5.1. □

Recall that each $\xi(s)$ is reduced to the range $[-\frac{1}{2}, \frac{1}{2}]$ in every time of the size-reducing procedure in the LLL algorithm. We let the symbol $E[x]$ denote the expected value of x in the distribution. Let $E[\xi(s)^2]$ denote the expected value of $\xi(s)^2$ for $1 \le s \le N$.

Assumption 5.6. *We assume that the value* $\xi(s)$ *for* $1 \le s \le N$ *is uniformly distributed over the range* $[-\frac{1}{2}, \frac{1}{2}]$.

Under Assumption 5.6, we have

$$E[\xi(s)^2] = \frac{1}{12} \tag{5.4}$$

by [17, Lemma 1]. Although this may not hold strictly as pointed out in [4], we assume it for simple discussion. Under Assumption 5.6, we may estimate the average gap between $\mathrm{SS}(\mathbf{A}^{(s)})$ and $\mathrm{SS}(\mathbf{A}^{(s+1)})$ as follows: Let A_0 be the minimum value among $\{A_1, \dots, A_n\}$. By Lemma 5.5, we have

$$\mathrm{SS}(\mathbf{A}^{(s)}) - \mathrm{SS}(\mathbf{A}^{(s+1)}) > \frac{\xi(s+1)^2(1-\delta)}{\delta} A_{m(s,\ell(s+1))} \ge \frac{\xi(s+1)^2(1-\delta)}{\delta} A_0,$$

and hence, for its average, we have

$$E[\mathrm{SS}(\mathbf{A}^{(s)}) - \mathrm{SS}(\mathbf{A}^{(s+1)})] > \frac{(1-\delta)}{12\delta} A_0.$$

Note that A_0 is very small compared to $A_{m(s,\ell(s+1))}$. To get more precise estimation, we will introduce other assumptions on distributions on $\ell(s)$ and $m(s, \ell(s+1))$ when we analyze the LLL-reduction for the basis $\mathbf{C}$ in Section 6 below.

5.2 Estimation of bounds for *N*

Recall that by equation (5.3), the loop invariant $\mathcal{LI}(\mathbf{A})$ is reduced by each swap ratio $\delta(s)$ by the s-th swap in the LLL algorithm for $\mathbf{A}$. Since $\delta(s) < \delta < 1$ by Lemma 5.1, an upper bound of the total number N of swaps is given by $-\log_\delta(\mathcal{LI}(\mathbf{A}))$ (see, e.g., [3, Theorem 4.19] for details). In contrast, we study a lower bound of N in this subsection.

Since by Lemma 5.1 each swap ratio $\delta(s)$ is defined as

$$\delta(s) = \xi(s)^2 + \frac{A^{(s-1)}_{\ell(s)+1}}{A^{(s-1)}_{\ell(s)}}, \tag{5.5}$$

we have to deal with the second term of (5.5) to estimate N in more detail. Since $A^{(s-1)}_{\ell(s)+1} < A_{M(s-1,\ell(s)+1)}$ and $A^{(s-1)}_{\ell(s)} > A_{m(s-1,\ell(s))}$ by Lemma 5.4, we have

$$\frac{A^{(s-1)}_{\ell(s)+1}}{A^{(s-1)}_{\ell(s)}} < \frac{A_{M(s-1,\ell(s)+1)}}{A_{m(s-1,\ell(s))}} \tag{5.6}$$

and it can be bounded by the *minimal ratio*

$$T = \min_{1\le i\ne j\le n}\left\{\frac{A_i}{A_j}\right\}$$

among the squared lengths $A_i = \|\mathbf{a}_i^*\|^2$ of Gram–Schmidt vectors of $\mathbf{A}$. Then we have $\delta(s) \ge \xi(s)^2 + T$, and we may consider $\xi(s)^2 + T$ instead of $\delta(s)$ for an *approximate lower bound* for N. On the other hand, by equation (5.3), we have $\mathcal{LI}(\mathbf{A}^{(s)}) = \mathcal{LI}(\mathbf{A}^{(s-1)}) \times \delta(s)$, and thus

$$\mathcal{LI}(\mathbf{A}') = \mathcal{LI}(\mathbf{A}^{(N)}) = \mathcal{LI}(\mathbf{A}) \times \prod_{s=1}^{N} \delta(s).$$

Therefore

$$\sum_{s=1}^{N} \log(\delta(s)) = \log(\mathcal{LI}(\mathbf{A}')) - \log(\mathcal{LI}(\mathbf{A})). \tag{5.7}$$

Lemma 5.7. *The total number N of swaps is not smaller than*

$$\frac{\log(\mathcal{LI}(\mathbf{A}')) - \log(\mathcal{LI}(\mathbf{A}))}{\log(T)}.$$

Proof. By $\delta(s) \ge T$, we have $N\log(T) \le \sum_{s=1}^{N} \log(\delta(s))$. Then Lemma 5.7 follows by equation (5.7). □

By Lemma 5.7, if we calculated the difference $\log(\mathcal{LI}(\mathbf{A}')) - \log(\mathcal{LI}(\mathbf{A}))$ and the value $\log(T)$, we could obtain an lower bound of N. Now we give a more precise estimate on the total number N of swaps. Under Assumption 5.6, the left-hand side of equation (5.7) can be expressed as $N \times E[\log(\delta(s))]$. Thus, we estimate the expected value $E[\log(\delta(s))] = E[\log(\xi(s)^2 + T)]$ as

$$E\left[\log(\eta^2 + T) : |\eta| \le \frac{1}{2}\right] \approx \int_{-\frac{1}{2}}^{\frac{1}{2}} \log(x^2 + T)dx = \log\left(\frac{1}{4} + T\right) - 2 + 4\sqrt{T}\arctan\left(\frac{1}{2\sqrt{T}}\right).$$

Now we set

$$\beta = \exp\left(\log\left(\frac{1}{4} + T\right) - 2 + 4\sqrt{T}\arctan\left(\frac{1}{2\sqrt{T}}\right)\right). \tag{5.8}$$

Then

$$\log(\beta) \approx E[\log(\xi(s)^2 + T)] < 0 \quad \text{and} \quad \sum_{s=1}^{N} \log(\delta(s)) \approx N \cdot \log(\beta).$$

Under Assumption 5.6, the total number N of swaps in the LLL algorithm for $\mathbf{A}$ is roughly estimated as

$$N \gtrsim \frac{\log(\mathcal{LI}(\mathbf{A}')) - \log(\mathcal{LI}(\mathbf{A}))}{\log(\beta)},$$

where $A \gtrsim B$ means $A \approx B$ or $A > B$.

Remark 5.8. The values of β can be calculated from T as in Table 1. In this remark, we assume that $\mathbf{A}$ is LLL-reduced. As to an estimation of T, it follows from GSA that T could be very small for larger n, since T is expected as

$$\frac{1}{q^{2(n-1)}} \approx 0.96^{2(n-1)} \quad \text{when } q = 1.04.$$

T	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7
β	0.034	0.170	0.274	0.376	0.478	0.579	0.680	0.780

Table 1. Values of T and β.

For a more precise estimation of (5.6), the difference between $m(s-1,\ell(s))$ and $M(s-1,\ell(s)+1)$ could not be so large. Thus, we may introduce a heuristic bound on the difference such as

$$|m(s-1,\ell(s)) - M(s-1,\ell(s)+1)| \le a,$$

for a small number a. In this case, the minimal ration T_a with parameter a could be

$$T_a = \min_{\substack{1\le i\ne j\le n\\ |i-j|\le a}} \left\{\frac{A_i}{A_j}\right\}$$

and it is expected as $\frac{1}{q^{2a}} \approx 0.96^{2a}$.

6 Analysis of LLL-reduction for C

In the previous section, we analyzed the behavior of the LLL-reduction for a general basis. In contrast, in this section, we analyze the LLL-reduction for the basis $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_n]$ defined in (1.2), by using the results in the previous section. We assume that $\mathbf{B}$ is LLL-reduced. As in (1.2), let $\mathbf{B}' = [\mathbf{b}'_1, \dots, \mathbf{b}'_n] \leftarrow \mathrm{LLL}(\mathbf{C})$ denote the output basis of the LLL algorithm for $\mathbf{C}$.

6.1 Initial swap of the LLL algorithm for C

In this subsection, we consider the initial swap in the LLL algorithm for $\mathbf{C}$. We assume that the inserted vector $\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{b}_i^* \in L$ satisfies $|v_i| \le \frac{1}{2}$ for $i = 1, \dots, n-u$ as in (3.1), and for the insertion index k of $\mathbf{v}$, it satisfies

$$\begin{cases} \|\pi_i(\mathbf{v})\|^2 \ge \|\mathbf{b}_i^*\|^2 & \text{for } i = 1, \dots, k-1, \\ \|\pi_k(\mathbf{v})\|^2 < \|\mathbf{b}_k^*\|^2, \end{cases} \tag{6.1}$$

where we set $\alpha = 1$ in Definition 3.2 for simple and practical discussion (see Remark 6.2 below). We assume $k < n-u$. Then we obtain the following:

Lemma 6.1. *The first $k-1$ vectors of $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_n]$ are unchanged at the beginning of the LLL algorithm.*

Proof. By the assumption $k < n-u$, the first k vectors of $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_n]$ are unchanged by the size-reducing procedure since $|\xi_{i,i-1}| = |\mu_{i,i-1}| \le \frac{1}{2}$ for $i = 2, \dots, k-1$ and $|\xi_{k,k-1}| = |v_k| < \frac{1}{2}$ by Proposition 4.6 (recall that $\xi_{i,j}$ is defined by (4.4)). Since the original basis $\mathbf{B}$ is assumed to be LLL-reduced, the first $k-2$ vectors of $\mathbf{C}$ are also unchanged at the beginning of the LLL algorithm. For the pair $(\mathbf{c}_{k-1}, \mathbf{c}_k) = (\mathbf{b}_{k-1}, \mathbf{v})$, we have

$$\|\mathbf{c}_k^*\| = D_{k-1} - v_{k-1}^2 \|\mathbf{b}_{k-1}^*\|^2 \ge (1 - \xi_{k,k-1}^2)\|\mathbf{b}_{k-1}^*\|^2$$

since $D_{k-1} = \|\pi_{k-1}(\mathbf{v})\|^2 \ge \|\mathbf{b}_{k-1}^*\|^2$ by condition (6.1) and $\xi_{k,k-1} = v_{k-1}$ by Proposition 4.6 (recall that D_ℓ is defined by (4.1)). Hence the pair $(\mathbf{c}_{k-1}, \mathbf{c}_k)$ satisfies the Lovász condition (5.1), and hence it cannot be swapped at the beginning of the LLL algorithm. □

Remark 6.2. While we set $\alpha = 1$ for our analysis, Fukase and Kashiwabara [5] set $\alpha = 0.99$ for their experiments. Note that it is harder to find a candidate vector for insertion as α is smaller than 1. Hence $\alpha \approx 1$ seems to be useful in practice. In [5, Section 8], they also consider up to $\alpha = 1.4$ for the second candidate vectors for insertion (they call such vectors *stock vectors*, see [5, Algorithm 3] for details).

Proposition 6.3. *The first k vectors of $\mathbf{C}$ cannot be swapped at the beginning of the LLL algorithm.*

Proof. By Lemma 6.1, it is sufficient to consider the pair $(\mathbf{c}_k, \mathbf{c}_{k+1})$. If the pair $(\mathbf{c}_k, \mathbf{c}_{k+1}) = (\mathbf{v}, \mathbf{b}_k)$ is swapped, then we obtain a new basis $\mathbf{C}' = [\mathbf{c}'_1, \dots, \mathbf{c}'_n]$ with $(\mathbf{c}'_k, \mathbf{c}'_{k+1}) = (\mathbf{b}_k, \mathbf{v})$. Set $C_i = \|\mathbf{c}_i^*\|^2$ and $C'_i = \|\mathbf{c}_i'^*\|^2$ for $1 \le i \le n$. Then $C'_k = \|\mathbf{b}_k^*\|^2$ and $C'_{k+1} = D_{k+1}$. The gap $\mathrm{SS}(\mathbf{C}) - \mathrm{SS}(\mathbf{C}')$ is given by

$$\begin{aligned}(C_k + C_{k+1}) - (C'_k + C'_{k+1}) &= \Big(D_k + \frac{D_{k+1}}{D_k}\|\mathbf{b}_k^*\|^2\Big) - (\|\mathbf{b}_k^*\|^2 + D_{k+1}) \\ &= (D_k - D_{k+1}) + \Big(\frac{D_{k+1} - D_k}{D_k}\Big)\|\mathbf{b}_k^*\|^2 \\ &= \Big(\frac{D_k - D_{k+1}}{D_k}\Big)(D_k - \|\mathbf{b}_k^*\|^2) < 0\end{aligned}$$

since $D_k < \|\mathbf{b}_k^*\|^2$ by condition (6.1) and $D_k - D_{k+1} = v_k^2\|\mathbf{b}_k^*\|^2 > 0$. This is a contradiction to Lemma 5.2. □

6.2 Swaps in the LLL algorithm for C

In this subsection, we consider all swaps in the LLL algorithm for the basis $\mathbf{C}$. As in Section 5.1, we fix the following notations on the LLL-reduction for $\mathbf{C}$:

- Let N be the total number of swaps.
- For $0 \le s \le N$, denote by $\mathbf{C}^{(s)} = [\mathbf{c}_1^{(s)}, \dots, \mathbf{c}_n^{(s)}]$ the basis obtained by s times swaps and size-reduced, and set $\mathbf{C}^{(0)} = \mathbf{C}$. Let $C_i^{(s)} = \|\mathbf{c}_i^{(s)*}\|^2$ for $1 \le i \le n$, where $\mathbf{c}_1^{(s)*}, \dots, \mathbf{c}_n^{(s)*}$ denote the Gram–Schmidt vectors of $\mathbf{C}^{(s)}$. Then $\mathbf{B}' = \mathbf{B}^{(N)} \leftarrow \mathrm{LLL}(\mathbf{C})$.
- By $\ell(s)$ we denote the index where the s-th swap occurs, that is, the $\ell(s)$-th vector $\mathbf{c}_{\ell(s)}^{(s-1)}$ and $(\ell(s)+1)$-st vector $\mathbf{c}_{\ell(s)+1}^{(s-1)}$ are swapped.
- For $1 \le s \le N$, we let

$$\xi_{\ell(s)+1,\ell(s)}^{(s)} = \frac{\langle \mathbf{c}_{\ell(s)+1}^{(s-1)}, \mathbf{c}_{\ell(s)}^{(s-1)*}\rangle}{\|\mathbf{c}_{\ell(s)}^{(s-1)*}\|^2} \quad \text{and} \quad \delta_{\ell(s)}^{(s)} = \frac{C_{\ell(s)}^{(s)}}{C_{\ell(s)}^{(s-1)}}.$$

  As in Section 5.1, we write $\xi(s)$ and $\delta(s)$ for simplicity.

  By Proposition 6.3, the first swap index $\ell(1)$ should be larger than $k+1$. We assume the following for simple analysis (see Figures 1 and 2 for examples).

Assumption 6.4. *For any $1 \le s \le N$, the s-th swap index $\ell(s)$ is not less than the insertion index k in the LLL algorithm for $\mathbf{C}$ (see also Assumption 6.6 below).*

By Lemma 5.5, we obtain the following result on the decreasing value of the squared-sum $\mathrm{SS}(\mathbf{C})$ by one time swap in the LLL algorithm for $\mathbf{C}$.

Proposition 6.5. *If a swap occurs at the ℓ-th index, then $\mathrm{SS}(\mathbf{C}) = \sum_{i=1}^n C_i$ is reduced by at least*

$$\frac{\xi_{\ell+1,\ell}^2(1-\delta_\ell)}{\delta_\ell}C_m \quad \text{or} \quad \frac{\xi_{\ell+1,\ell}^2(1-\delta)}{\delta}C_m,$$

for $m = m(s, \ell(s+1))$, where $\xi_{\ell+1,\ell} = \xi(s)$, $\ell = \ell(s)$, and $\delta_\ell = \delta(s)$ for some $1 \le s \le N$. Furthermore, by the proof of Lemma 5.4, we expect $m = \ell$ in most cases (sometimes $m = \ell+1$, $\ell+2$ or so on).

6.2.1 Average of decreasing values by swaps

Here we estimate the average of decreasing values of the squared-sum $\mathrm{SS}(\mathbf{C})$ by one time swap in the LLL algorithm for $\mathbf{C}$. It follows by Proposition 6.5 that decreasing values of $\mathrm{SS}(\mathbf{C})$ depend mainly on swap indices $\ell(s)$. In Figures 1 and 2, we show two examples of the number of swaps and the swap indices $\ell(s)$ in the LLL algorithm for $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_n]$ with insertion index $k = 5$ and 10 (the inserted vector $\mathbf{c}_k = \mathbf{v}$ is generated by

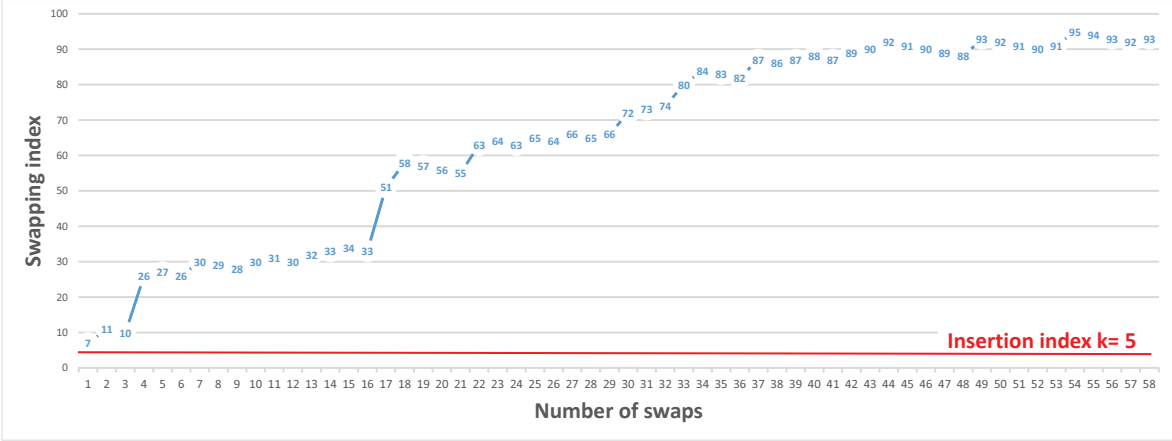


Figure 1. Example of swap indices in the LLL algorithm for **C** with insertion index $k = 5$ in a lattice of dimension 100.

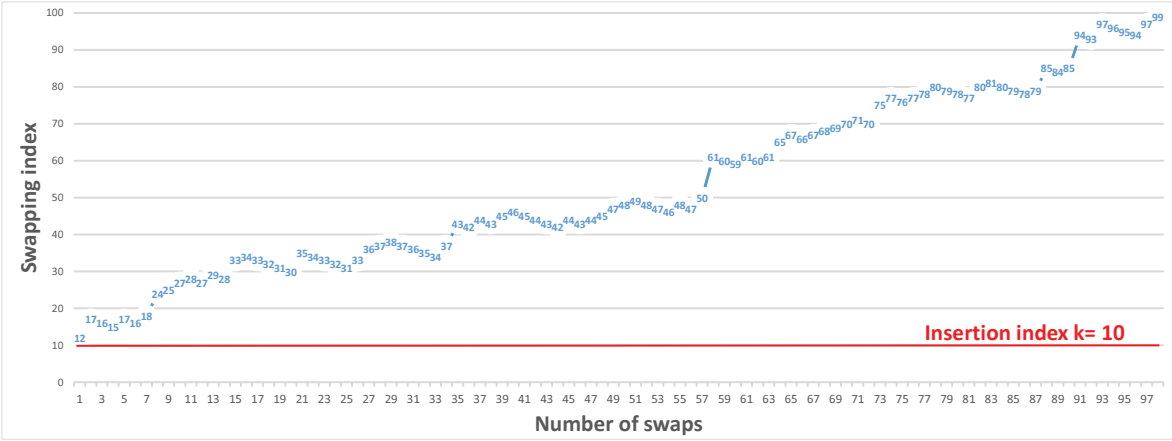


Figure 2. Same as Figure 1, but $k = 10$.

Schnorr's SA). The lattice used for Figures 1 and 2 is a lattice of dimension 100, chosen from Darmstadt SVP challenge problems. It is hard to grasp all the swap indices $\ell(s)$ accurately. However, we see from Figures 1 and 2 that swap indices $\ell(s)$ are roughly distributed over the range from k to $n - 1$ *evenly*. Then we assume the following for simple analysis.

Assumption 6.6. *We assume that swap indices $\ell(s)$ are distributed over the range from k to $n - 1$ evenly (uniformly) and independently to the distribution of $\xi(s)$.*

Under Assumption 6.6, we obtain the following estimate on the average of decreasing values of $\mathrm{SS}(\mathbf{C})$ by the LLL algorithm for **C**.

Proposition 6.7. *Under Assumptions 5.6 and 6.6, the average of decreasing values of $\mathrm{SS}(\mathbf{C})$ by one time swap in the LLL algorithm for* **C** *is estimated to be greater than*

$$\frac{1-\delta}{12\delta} \cdot \mathrm{vol}(\pi_k(L))^{2/(n-k)}.$$

Proof. Due to the independence of $\xi(s)$ in Assumption 6.6, it follows from (5.4) and Proposition 6.5 that $\mathrm{SS}(\mathbf{C}) = \sum_{i=1}^{n} C_i$ is reduced at least by

$$E\left[\frac{\xi_{\ell+1,\ell}^2(1-\delta)}{\delta} C_m\right] \approx \frac{(1-\delta)}{\delta} \cdot E[\xi_{\ell+1,\ell}^2] \cdot E[C_m] \approx \frac{1-\delta}{12\delta} E[C_m]$$

by a swap at the ℓ-th index for some m with $k \le \ell \le m \le n-1$. Moreover, under Assumption 6.6, the value $E[C_m]$ is estimated as

$$\begin{aligned}\frac{1}{(n-k)}\sum_{m=k}^{n-1} C_m &\ge \Big(\prod_{m=k}^{n-1} C_m\Big)^{1/(n-k)} \\ &= \Big(D_k \cdot \Big(\frac{D_{k+1}}{D_k}\|\mathbf{b}_k^*\|^2\Big)\cdots\Big(\frac{D_n}{D_{n-1}}\|\mathbf{b}_{n-1}^*\|^2\Big)\Big)^{1/(n-k)} \\ &= \mathrm{vol}(\pi_k(L))^{2/(n-k)} \end{aligned} \tag{6.2}$$

by the inequality of arithmetic and geometric means and Proposition 4.2. Note that $D_n = \|\mathbf{b}_n^*\|^2$ by setting $\nu_n = 1$ and

$$\mathrm{vol}(\pi_k(L))^2 = \prod_{i=k}^{n} \|\mathbf{b}_i^*\|^2,$$

where $\pi_k(L)$ is the lattice of dimension $n-k+1$ with basis $[\pi_k(\mathbf{b}_k), \dots, \pi_k(\mathbf{b}_n)]$. □

6.2.2 Expected number of swaps

The total number N of swaps in the LLL algorithm for $\mathbf{C}$ is the most important to analyze the gap between two squared-sums $\mathrm{SS}(\mathbf{C})$ and $\mathrm{SS}(\mathbf{B}')$. Here we give an estimate of the number N. For the two bases $\mathbf{B}$ and $\mathbf{C}$ defined in (1.2), let $B_i = \|\mathbf{b}_i^*\|^2$ as $C_i = \|\mathbf{c}_i^*\|^2$ for $1 \le i \le n$. A relation between two loop invariants $\mathcal{LI}(\mathbf{B})$ and $\mathcal{LI}(\mathbf{C})$ is given as follows (recall that D_ℓ is defined by (4.1)).

Lemma 6.8. *We have*

$$\mathcal{LI}(\mathbf{C}) = \mathcal{LI}(\mathbf{B}) \times \frac{D_k \cdots D_{n-1}}{B_k \cdots B_{n-1}}.$$

Proof. By Proposition 4.2, we have $C_k = D_k$ and $C_j = \frac{D_j}{D_{j-1}} B_{j-1}$ for $k+1 \le j \le n$. By definition, the loop invariant is given by

$$\mathcal{LI}(\mathbf{C}) = \prod_{i=1}^{k-1} B_i^{n-i} \times D_k^{n-k} \times \prod_{i=k+1}^{n-1} \Big(\frac{D_i}{D_{i-1}} B_{i-1}\Big)^{n-i}.$$

Then we obtain

$$\begin{aligned}\frac{\mathcal{LI}(\mathbf{C})}{\mathcal{LI}(\mathbf{B})} &= \frac{D_k^{n-k} \times \prod_{i=k+1}^{n-1} \big(\frac{D_i}{D_{i-1}} B_{i-1}\big)^{n-i}}{\prod_{i=k}^{n-1} B_i^{n-i}} \\ &= \frac{D_k^{n-k} \times \big(\frac{D_{k+1}}{D_k}\big)^{n-k-1} \times \big(\frac{D_{k+2}}{D_{k+1}}\big)^{n-k-2} \times \cdots \times \big(\frac{D_{n-1}}{D_{n-2}}\big)}{B_k \cdots B_{n-1}} \\ &= \frac{D_k \cdots D_{n-1}}{B_k \cdots B_{n-1}}. \end{aligned}$$

This completes the proof of Lemma 6.8. □

Recall that the loop invariant $\mathcal{LI}(\mathbf{C})$ is reduced by the factor of the swap ratio $\delta(s)$ at the s-th swap for $1 \le s \le N$. For $\mathbf{B}' \leftarrow \mathrm{LLL}(\mathbf{C})$, we have

$$\mathcal{LI}(\mathbf{B}') = \prod_{s=1}^{N} \delta(s) \times \mathcal{LI}(\mathbf{C}).$$

By combining this with Lemma 6.8, we obtain

$$\mathcal{LI}(\mathbf{B}') = \prod_{s=1}^{N} \delta(s) \times \prod_{j=k}^{n-1} \frac{D_j}{B_j} \times \mathcal{LI}(\mathbf{B}). \tag{6.3}$$

For simple analysis, we assume the following.

Assumption 6.9. *Set* $R := \frac{\mathcal{LI}(\mathbf{B}')}{\mathcal{LI}(\mathbf{B})}$. *We assume* $1 \succeq R$, *that is,* $\mathcal{LI}(\mathbf{B}) \succeq \mathcal{LI}(\mathbf{B}')$.

This assumption is based on GSA for $\mathbf{B}$ and $\mathbf{B}'$. Since both $\mathbf{B}$ and $\mathbf{B}'$ are LLL-reduced, we roughly expect

$$\frac{B_i}{B_{i+1}} \approx \frac{B'_i}{B'_{i+1}} \approx q^2 \quad \text{for all } 1 \le i \le n-1,$$

where we set $B'_i = \|\mathbf{b}'^*_i\|^2$ for the Gram–Schmidt vectors $[\mathbf{b}'^*_1, \dots, \mathbf{b}'^*_n]$ of $\mathbf{B}'$ (see Section 2.2 for the q-value). Hence we can roughly expect that R would be approximately equal to 1 under GSA for $\mathbf{B}$ and $\mathbf{B}'$. As Example 6.11 below implies, we expect that R would be much smaller than 1 in practice when we take a shorter lattice vector $\mathbf{v}$ as the inserted vector into $\mathbf{B}$.

By equation (6.3), we have

$$\sum_{s=1}^{N} \log(\delta(s)) = \sum_{j=k}^{n-1} \log\Big(\frac{B_j}{D_j}\Big) + \log(R). \tag{6.4}$$

By Lemma 5.1, we have $\delta(s) < \delta$ for any $1 \le s \le N$. However, the parameter δ gives just an upper bound of each factor $\delta(s)$. Since each $\delta(s)$ is defined as

$$\delta(s) = \xi(s)^2 + \frac{C^{(s-1)}_{\ell(s)+1}}{C^{(s-1)}_{\ell(s)}},$$

we estimate that $E[\delta(s)] \ge E[\xi(s)^2] = \frac{1}{12}$ for $1 \le s \le N$ by (5.4). We take a constant $0 < \epsilon < 1$ satisfying $E[\log(\delta(s))] \ge \log(\epsilon)$. Then by equation (6.4), we obtain the following estimate on the number N.

Proposition 6.10. *Under Assumption 6.9, the total number N of swaps in the LLL algorithm for the basis $\mathbf{C}$ is roughly estimated as*

$$N \ge \sum_{j=k}^{n-1} \log_\epsilon\Big(\frac{B_j}{D_j}\Big) + \log_\epsilon(R) \ge \sum_{j=k}^{n-1} \log_\epsilon\Big(\frac{B_j}{D_j}\Big). \tag{6.5}$$

As discussed in Section 5.2, we may take ϵ as β defined in (5.8), and it satisfies $\beta \ge 0.034$ from Table 1. However, for simple analysis, we may take

$$\epsilon = E[\xi(s)^2] = \frac{1}{12}$$

to obtain a lower bound of N under Assumption 5.6 (this ϵ is experimentally chosen, see Example 6.11 below for details).

Example 6.11. As an example, we take a lattice L of dimension $n = 100$, chosen from Darmstadt SVP challenge problems (using seed 0). Let $\mathbf{B}$ be an LLL-reduced basis of L with reduction parameter $\delta = 0.99$. We also take a lattice vector $\mathbf{v} \in L$ with insertion index $k = 4$, which is generated by Schnorr's SA. Let $\mathbf{C}$, $\mathbf{B}'$ be two lattice bases constructed by (1.2) (note $\mathbf{B}' \leftarrow \mathrm{LLL}(\mathbf{C})$). In Figure 3, we give the GSA behavior of three lattices $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{B}'$. More specifically, for three lattice bases $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$, $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_n]$ and $\mathbf{B}' = [\mathbf{b}'_1, \dots, \mathbf{b}'_n]$, the values $\log_2(\|\mathbf{b}_1\|^2/\|\mathbf{b}^*_i\|^2)$, $\log_2(\|\mathbf{c}_1\|^2/\|\mathbf{c}^*_i\|^2)$ and $\log_2(\|\mathbf{b}'_1\|^2/\|\mathbf{b}'^*_i\|^2)$ for $1 \le i \le n = 100$ are plotted. We have

$$\mathrm{SS}(\mathbf{B}') < \mathrm{SS}(\mathbf{B}) < \mathrm{SS}(\mathbf{C}).$$

We give some numerical data related with our assumptions. In this example, we have $\mathcal{LI}(\mathbf{B}) \approx 4.99 \times 10^{32973}$ and $\mathcal{LI}(\mathbf{B}') \approx 2.43 \times 10^{32899}$ (cf. $\mathrm{vol}(L)^2 \approx 5.0 \times 10^{601}$), and

$$R = \frac{\mathcal{LI}(\mathbf{B}')}{\mathcal{LI}(\mathbf{B})} \approx 4.86 \times 10^{-75} \ll 1.$$

Note that we have $R \le 1$ in most examples, which implies that Assumption 6.9 holds in practice. For our estimation (6.5), we have

$$\sum_{j=k}^{n-1} \log_\epsilon\Big(\frac{B_j}{D_j}\Big) \approx 54.96 \quad \text{and} \quad \log_\epsilon(R) \approx 68.86.$$

Then our estimation (6.5) gives $N \approx 54.96 + 68.86 = 123.82$. In contrast, the actual total number of swaps in the LLL algorithm for $\mathbf{C}$ is equal to 132, close to our estimation. On the other hand, if we take $\delta = 0.99$ as

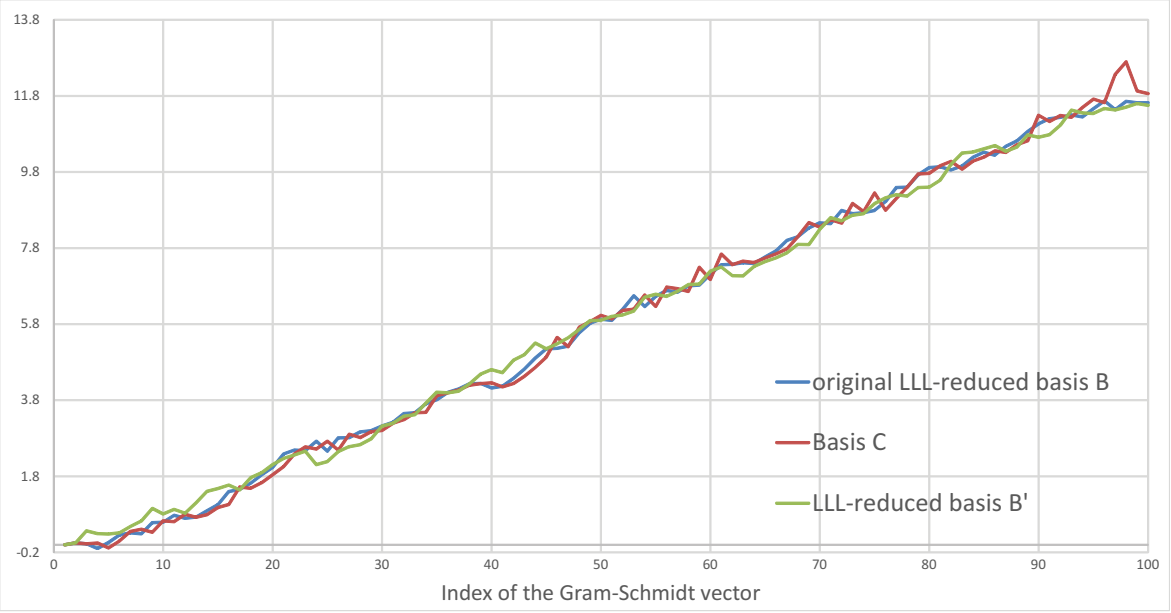


Figure 3. The GSA behavior of three lattice bases $\mathbf{B}$, $\mathbf{C}$, $\mathbf{B}'$ with insertion index $k = 4$ (the original basis $\mathbf{B}$ is given by LLL-reducing a lattice basis of dimension 100, chosen from Darmstadt SVP challenge problems).

the base of logarithm, then

$$\sum_{j=k}^{n-1} \log_\delta\left(\frac{B_j}{D_j}\right) \approx 13588 \quad \text{and} \quad \log_\delta(R) \approx 17026,$$

which are much larger than the actual total number of swaps. If we take the minimum value $\beta = 0.034$ as the base of logarithm, then

$$\sum_{j=k}^{n-1} \log_\beta\left(\frac{B_j}{D_j}\right) \approx 40.39 \quad \text{and} \quad \log_\beta(R) \approx 50.60,$$

which are about 1.36 times smaller than our estimation with $\epsilon = \frac{1}{12}$.

6.3 Estimate of gap between SS(C) and SS(B′)

In this subsection, we give an estimate of the gap between two squared-sums $\mathrm{SS}(\mathbf{C})$ and $\mathrm{SS}(\mathbf{B}')$. By Proposition 6.5, the gap is estimated as

$$\mathrm{SS}(\mathbf{C}) - \mathrm{SS}(\mathbf{B}') > \sum_{s=1}^{N} \frac{\xi(s)^2(1-\delta(s))}{\delta(s)} C_m \tag{6.6}$$

$$> \sum_{s=1}^{N} \frac{\xi(s)^2(1-\delta)}{\delta} C_m, \tag{6.7}$$

for $m = m(s, \ell(s+1))$. Gap (6.7) is approximately determined by both the average of decreasing values of $\mathrm{SS}(\mathbf{C})$ and the total number N of swaps in the LLL algorithm for the basis $\mathbf{C}$. By Propositions 6.7 and 6.10, we obtain the following estimate on gap (6.7).

Theorem 6.12. *Under Assumptions 5.6, 6.6 and 6.9, the gap* $\mathrm{SS}(\mathbf{C}) - \mathrm{SS}(\mathbf{B}')$ *is estimated to be greater than*

$$\underbrace{\left(\frac{1-\delta}{12\delta} \cdot \mathrm{vol}(\pi_k(L))^{2/(n-k)}\right)}_{\text{Average of decreasing values}} \cdot \underbrace{\sum_{j=k}^{n-1} \log_\epsilon\left(\frac{B_j}{D_j}\right)}_{\text{Estimation of } N}. \tag{6.8}$$

6.4 Alternative estimate for gap between SS(C) and SS(B′)

In this subsection, we give an alternative heuristic estimation for the gap between two squared-sums SS($\mathbf{C}$) and SS($\mathbf{B}'$). In Theorem 6.12, although the total number N of swaps plays an important role, it is rather difficult to give its precise estimation. Moreover, the term $\frac{1-\delta}{\delta}$ in gap (6.7) seems much smaller than $\frac{1-\delta(s)}{\delta(s)}$ in gap (6.6), that might cause our estimation of the gap very small. In order to overcome such defects, we use an approximation of $\sum_{s=1}^{N}\log(\delta(s))$ directly. Specifically, we give a certain estimation of gap (6.6) under the following settings, where we assume that three distributions of $\xi(s)$, C_m and $\delta(s)$ are independent:

- For estimation of each $\xi(s)^2$, we use its *average* $\frac{1}{12}$ given in equation (5.4).
- For estimation of each C_m, we use its *average* $\mathrm{vol}(\pi_k(L))^{2/(n-k)}$ given in equation (6.2).
- For estimation of each $\frac{1-\delta(s)}{\delta(s)}$, we use its *average* $\frac{1}{N}\cdot\sum_{s=1}^{N}\frac{1-\delta(s)}{\delta(s)}$.

Proposition 6.13 (Alternative estimate). *Under Assumptions 5.6, 6.6 and 6.9, the gap* SS($\mathbf{C}$) − SS($\mathbf{B}'$) *is estimated to be greater than (cf. equation* (6.8)*)*

$$-\Big(\frac{1}{12}\cdot\mathrm{vol}(\pi_k(L))^{2/(n-k)}\Big)\cdot\sum_{j=k}^{n-1}\log\Big(\frac{B_j}{D_j}\Big).$$

Proof. Under the above settings, gap (6.6) can be estimated as

$$\sum_{s=1}^{N}\frac{\xi(s)^2(1-\delta(s))}{\delta(s)}C_m \approx \frac{1}{12}\cdot\mathrm{vol}(\pi_k(L))^{2/(n-k)}\cdot\sum_{s=1}^{N}\frac{1-\delta(s)}{\delta(s)}.$$

Since $\frac{(1-x)}{x}\geq -\log(x)$ for $0<x<1$, we have $\frac{1-\delta(s)}{\delta(s)}\geq -\log(\delta(s))$ with $0<\delta(s)<\delta<1$. Under Assumption 6.9, we have

$$\begin{aligned}\sum_{s=1}^{N}\frac{1-\delta(s)}{\delta(s)} &\geq \sum_{s=1}^{N}-\log(\delta(s))\\ &= -\sum_{j=k}^{n-1}\log\Big(\frac{B_j}{D_j}\Big)-\log(R)\\ &\geq -\sum_{j=k}^{n-1}\log\Big(\frac{B_j}{D_j}\Big)\end{aligned}$$

by equation (6.4). This completes the proof of Proposition 6.13. □

7 Estimated gap between SS(B) and SS(B′) and mutant vectors

Let $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{B}'\leftarrow\mathrm{LLL}(\mathbf{C})$ as in (1.2). In this section, we estimate the gap between two squared-sums SS($\mathbf{B}$) and SS($\mathbf{B}'$), and define mutant vectors in order to definitely decrease the squared-sum SS($\mathbf{B}$) of the original basis $\mathbf{B}$. By combining Theorems 4.4 and 6.12, we obtain the following estimate.

Theorem 7.1. *Under Assumptions 5.6, 6.6 and 6.9, the gap* SS($\mathbf{B}$) − SS($\mathbf{B}'$) *is estimated to be greater than*

$$E(\mathbf{v},k)+\frac{1-\delta}{12\delta\log(\epsilon)}\cdot\mathrm{vol}(\pi_k(L))^{2/(n-k)}\cdot\sum_{j=k}^{n-1}\log\Big(\frac{B_j}{D_j}\Big). \tag{7.1}$$

If the total value of (7.1) is positive, then the squared-sum SS($\mathbf{B}$) can be strictly decreased (i.e. condition (1.1) is satisfied). However, we cannot find $\mathbf{v}\in L$ such that the total value of (7.1) is positive, due to that the constant term $\frac{1-\delta}{12\delta\log(\epsilon)}<0$ seems too small (see Section 8 below for our experimental results). Then we give the following definition of candidate lattice vectors $\mathbf{v}\in L$ which enable us to strictly decrease SS($\mathbf{B}$) in practice.

Definition 7.2 (Mutant vectors). Given an LLL-reduced basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ of a lattice L, let $\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{b}_i^* \in L$ be a lattice vector sampled by Schnorr's SA. Given a constant $c > 0$, we call $\mathbf{v}$ a *mutant vector with factor c* if

(i) the insertion index k of $\mathbf{v}$ is smaller than $n - u$ (where u is the constant of search space bound for SA), and

(ii) the following condition is satisfied:

$$E(\mathbf{v}, k) > c \cdot \mathrm{vol}(\pi_k(L))^{2/(n-k)} \cdot \sum_{j=k}^{n-1} \log\Big(\frac{B_j}{D_j}\Big). \tag{7.2}$$

Remark 7.3. The alternative estimation in Proposition 6.13 may require us to set $c = \frac{1}{12} \approx 0.083$ in Definition 7.2. However, compared to the experimental constants in (8.1) below, the value $c = \frac{1}{12}$ is still 3 to 4 times smaller for practical use. This seems due to the ignorance of the value $\log(R)$. In fact, as Example 6.11 implies, the value $\log_\epsilon(R)$ gives a valuable information about the number N of swaps. However, the value can be obtained after computing $\mathbf{B}' \leftarrow \mathrm{LLL}(\mathbf{C})$. In this paper, we set $\log_\epsilon(R) = 0$. We leave the analysis of $\log_\epsilon(R)$ as our future work.

As mentioned in Remark 4.5, we consider only the case $E(\mathbf{v}, k) \le 0$. In this case, we expect from equation (4.3) that $B_j \le D_j$ for most $j = k, \dots, n-1$, and hence

$$\sum_{j=k}^{n-1} \log\Big(\frac{D_j}{B_j}\Big) > 0$$

with high probability. Then, in Definition 7.2, we estimate the gap between two squared-sums $\mathrm{SS}(\mathbf{C})$ and $\mathrm{SS}(\mathbf{B}')$ as

$$O\Big(\mathrm{vol}(\pi_k(L))^{2/(n-k)} \cdot \sum_{j=k}^{n-1} \log\Big(\frac{D_j}{B_j}\Big)\Big).$$

If we set a suitable constant $c > 0$, a mutant vector $\mathbf{v}$ with factor c can definitely decrease $\mathrm{SS}(\mathbf{B})$ (with high probability). In Section 8 below, we give a suitable constant c in practice. In this paper, we focus on Schnorr's SA to generate a number of short lattice vectors $\mathbf{v}$. Note that Definition 7.2 can be applied to more general short lattice vectors. Given a lattice vector $\mathbf{v}$ sampled by Schnorr's SA, we expect $B_j \approx D_j$ for $j = k, \dots, n-1$ (it is verified by our experiments). Then we approximately have

$$\log\Big(\frac{B_j}{D_j}\Big) = \log\Big(\Big(\frac{B_j}{D_j} - 1\Big) + 1\Big) \approx \Big(\frac{B_j}{D_j} - 1\Big)$$

since $\log(1 + x) \approx x$ if $x \in \mathbb{R}$ is sufficiently close to 0. Therefore the right-hand side of (7.2) can be replaced with (cf. equation (4.3) for $E(\mathbf{v}, k)$)

$$c \cdot \mathrm{vol}(\pi_k(L))^{2/(n-k)} \cdot \sum_{j=k}^{n-1} \Big(\frac{B_j}{D_j} - 1\Big).$$

Remark 7.4. Given a basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ of a lattice L, we have

$$\frac{\sum_{i=1}^{n} \|\mathbf{b}_i^*\|^2}{n} \ge \Big(\prod_{i=1}^{n} \|\mathbf{b}_i^*\|^2\Big)^{1/n} = \mathrm{vol}(L)^{2/n}.$$

Hence a theoretical lower bound of $\mathrm{SS}(\mathbf{B}) = \sum_{i=1}^{n} \|\mathbf{b}_i^*\|^2$ is given by $n \cdot \mathrm{vol}(L)^{2/n}$. However, this lower bound seems considerably small in practice. On the other hand, if $\mathbf{B}$ is reduced by LLL or BKZ, we approximately have

$$\mathrm{SS}(\mathbf{B}) = \|\mathbf{b}_1\|^2 \cdot \sum_{i=1}^{n} (q^2)^{1-i} = \|\mathbf{b}_1\|^2 \cdot \frac{1 - q^{-2n}}{1 - q^{-2}}$$

under GSA, where the q-value depends on the lattice reduction algorithm. In case of the LLL algorithm, we can estimate from Section 2.2 that $q \approx 1.04$ and $\|\mathbf{b}_1\|^2 \approx 1.022^{2n} \cdot \mathrm{vol}(L)^{2/n}$ in average for high dimension $n \ge 100$.

8 Experimental verification

In this section, we verify our analysis by experiments. Specifically, we verify that a mutant vector $\mathbf{v}$ with certain factor $c > 0$ (see Definition 7.2) can decrease the squared-sum $\mathrm{SS}(\mathbf{B})$ for an LLL-reduced basis $\mathbf{B}$. In our experiments, we used Schnorr's SA with search space bound $u = 30$ to generate a number of short lattice vectors (i.e. the search space size $\#S_{u,\mathbf{B}} = 2^{30}$). We also used the PARI library (http://pari.math.u-bordeaux.fr/) for the LLL algorithm with reduction parameter $\delta = 0.99$. We took three lattices of dimensions $n = 100, 110$ and 120 from Darmstadt SVP challenge problems. In particular, we set

$$c = \begin{cases} 0.25 & \text{for } n = 100, \\ 0.35 & \text{for } n = 110, 120. \end{cases} \tag{8.1}$$

Note that these constants are determined by our experiments. Given an LLL-reduced basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ of a lattice L of dimensions $n = 100, 110$ and 120, we repeatedly performed the following procedures and computed $\sum_{i=1}^{n} \|\mathbf{b}_i^*\|^2$:

- *Step 1*. Randomly generate a vector $\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{b}_i^* \in L$ by Schnorr's SA with $u = 30$ (see Section 3.1), and compute the insertion index k of $\mathbf{v}$ by checking $\|\pi_i(\mathbf{v})\|^2 < \|\mathbf{b}_i^*\|^2$ for $i = 1, \dots, n$ (see Definition 3.2). If $\|\pi_i(\mathbf{v})\|^2 \geq \|\mathbf{b}_i^*\|^2$ for all $1 \leq i \leq n$, generate another lattice vector $\mathbf{v}$.
- *Step 2*. Compute $E(\mathbf{v}, k)$ from $\mathbf{B}$ and $\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{b}_i^*$ (see equation (4.3) for $E(\mathbf{v}, k)$). If condition (7.2) is satisfied, insert $\mathbf{v}$ into $\mathbf{B}$ at the k-th position to obtain a basis $\mathbf{C}$ as in (1.2). Otherwise, go back to Step 1.
- *Step 3*. Set $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n] \leftarrow \mathrm{LLL}(\mathbf{C})$, and compute the squared-sum $\mathrm{SS}(\mathbf{B})$.

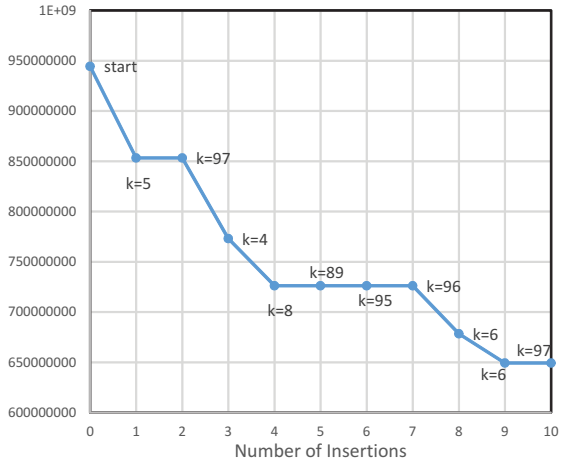
8.1 Experimental results

In Figures 4–6, we give our experimental results on the transition of $\mathrm{SS}(\mathbf{B})$ and the value $\log_2(\|\mathbf{b}_1\|^2/\|\mathbf{b}_i^*\|^2)$ for $1 \leq i \leq n$, which represents the GSA behavior of $\mathbf{B}$. Note that the value k in the left side figure denotes the insertion index of each mutant vector. From Figures 4–6, we see the following:

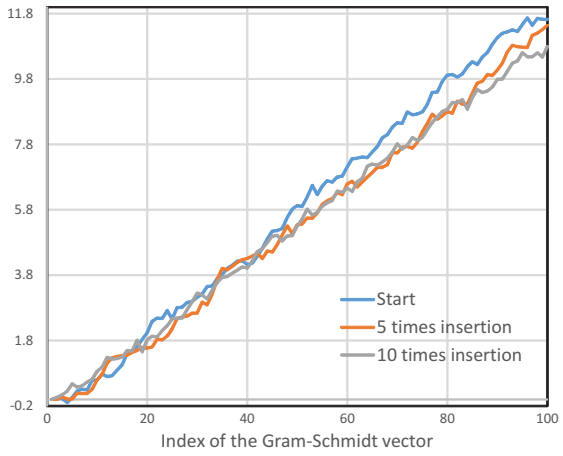
- Mutant vectors with factor c given by equation (8.1) can decrease $\mathrm{SS}(\mathbf{B})$ in most cases. Since it requires small factor c, condition (7.2) gives a good criterion of choosing candidate lattice vectors $\mathbf{v}$ to decrease $\mathrm{SS}(\mathbf{B})$. In particular, mutant vectors with smaller factor c can decrease $\mathrm{SS}(\mathbf{B})$ more greatly (but it is harder to generate mutant vectors with smaller factor c).
- By repeatedly inserting a mutant vector, we can obtain better GSA behavior of $\mathbf{B}$. Namely, the values $\log_2(\|\mathbf{b}_1\|^2/\|\mathbf{b}_i^*\|^2)$ for $1 \leq i \leq n$ become to lie on a straight line gradually.
- Moreover, there is a trade-off between the size of $\mathrm{SS}(\mathbf{B})$ and the required number of sampling short lattice vectors $\mathbf{v}$ to find a mutant vector. More specifically, as the squared-sum $\mathrm{SS}(\mathbf{B})$ becomes smaller, it requires to sample more short lattice vectors by Schnorr's SA to find a mutant vector. Note that we can easily find mutant vectors $\mathbf{v}$ with small insertion index k after the latter half of vectors of $\mathbf{B}$ are shortened (see [14] for such phenomenon).

8.2 Comparison to the RR algorithm by Fukase and Kashiwabara

Fukase and Kashiwabara [5] propose the RR algorithm in order to *steadily* decrease the squared-sum $\mathrm{SS}(\mathbf{B})$. The strategy of the RR algorithm is similar to the strategy of the BKZ algorithm, and it restricts insertion positions. More specifically, given a restriction index $1 \leq r \leq n$, the RR algorithm does not insert a short lattice vector $\mathbf{v}$ with insertion index $k \leq r$. Different from the RR algorithm, our criterion (7.2) for mutant vectors does not enforce us to restrict insertion positions. Therefore we expect that our criterion (7.2) would enable us to search mutant vectors more flexibly and hence to decrease $\mathrm{SS}(\mathbf{B})$ more efficiently.

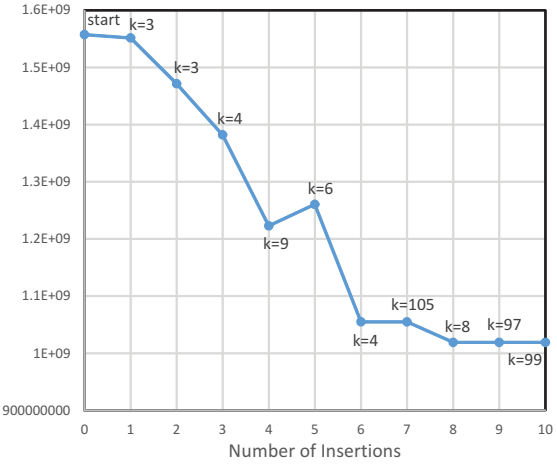


(a) Transition of the squared-sum SS(**B**).

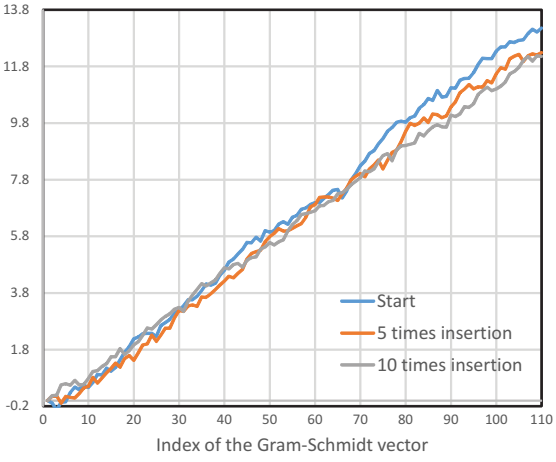


(b) The value $\log_2(\|\mathbf{b}_1\|^2/\|\mathbf{b}_i^*\|^2)$ for GSA.

Figure 4. Transition of the squared-sum SS(**B**) and the GSA behavior by insertion of mutant vectors in a lattice of dimension $n = 100$.

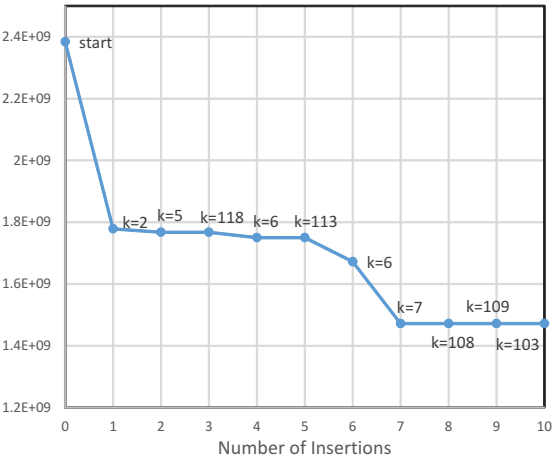


(a)

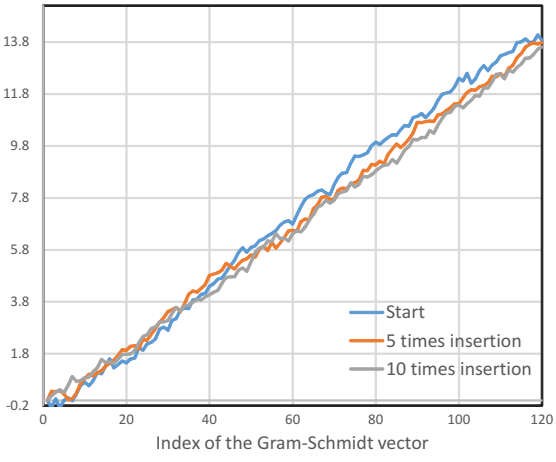


(b)

Figure 5. Same as Figure 4, but for lattice dimension $n = 110$.



(a)



(b)

Figure 6. Same as Figure 4, but for lattice dimension $n = 120$.

9 Conclusion and future work

Given an LLL-reduced basis $\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_n]$ of a lattice L, we gave an attempt to estimate a lower bound of the total number of swaps in the LLL algorithm. In Definition 7.2, we also gave a condition of mutant vectors $\mathbf{v} = \sum_{i=1}^{n} v_i \mathbf{b}_i^* \in L$. Our experiments showed that although the constant c should be determined experimentally, our condition of mutant vectors gives a good criterion to decrease the sum $\mathrm{SS}(\mathbf{B}) = \sum_{i=1}^{n} \|\mathbf{b}_i^*\|^2$ of the squared lengths of the Gram–Schmidt vectors $[\mathbf{b}_1^*, \dots, \mathbf{b}_n^*]$ of $\mathbf{B}$. Compared to the RR algorithm of [5], our condition enables us to search mutant vectors more flexibly, and hence we expect that we could decrease $\mathrm{SS}(\mathbf{B})$ more efficiently.

Our future work is to study how to search and sample mutant vectors efficiently. Specifically, while in this paper we focused on Schnorr's SA for sampling short lattice vectors, we would like to improve the method in [5] to efficiently sample a number of mutant vectors. With our condition, we also would like to try to solve SVP in lattices of high dimensions.

Acknowledgment: The authors would like to thank Phong Nguyen and the anonymous reviewers for their helpful comments.

Funding: This work was supported by CREST, JST. A part of this work was also supported by JSPS KAKENHI grant number 16H02830.

References

[1] M. Ajtai, The shortest vector problem in L_2 is NP-hard for randomized reductions, in: *Proceedings of the 30th Annual ACM Symposium on Theory of Computing – STOC 1998*, ACM, New York (1998), 10–19.

[2] M. Ajtai, R. Kumar and D. Sivakumar, A sieve algorithm for the shortest lattice vector problem, in: *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing – STOC 2001*, ACM, New York (2001), 601–610.

[3] M. R. Bremner, *Lattice Basis Reduction: An Introduction to the LLL Algorithm and its Applications*, CRC Press, Boca Raton, 2011.

[4] J. Buchmann and C. Ludwig, Practical lattice basis sampling reduction, in: *Algorithmic Number Theory – ANTS 2006*, Lecture Notes in Comput. Sci. 4076, Springer, Berlin (2006), 222–237.

[5] M. Fukase and K. Kashiwabara, An accelerated algorithm for solving SVP based on statistical analysis, *J. Inform. Process.* **23** (2015), no. 1, 1–15.

[6] S. D. Galbraith, *Mathematics of Public Key Cryptography*, Cambridge University Press, Cambridge, 2012.

[7] N. Gama and P. Q. Nguyen, Predicting lattice reduction, in: *Advances in Cryptology – EUROCRYPT 2008*, Lecture Notes in Computer Sci. 4965, Springer, Berlin (2008), 31–51.

[8] N. Gama, P. Q. Nguyen and O. Regev, Lattice enumeration using extreme pruning, in: *Advances in Cryptology – EUROCRYPT 2010*, Lecture Notes in Computer Sci. 6110, Springer, Berlin (2010), 257–278.

[9] O. Goldreich, S. Goldwasser and S. Halevi, Public-key cryptosystems from lattice reduction problems, in: *Advances in Cryptology – CRYPTO 1997*, Lecture Notes in Computer Sci. 1294, Springer, Berlin (1997), 112–131.

[10] J. Hoffstein, J. Pipher and J. H. Silverman, NTRU: A ring-based public key cryptosystem, in: *Algorithmic Number Theory – ANTS III*, Lecture Notes in Computer Sci. 1423, Springer, Berlin (1998), 267–288.

[11] A. K. Lenstra, H. W. Lenstra and L. Lovász, Factoring polynomials with rational coefficients, *Math. Ann.* **261** (1982), no. 4, 515–534.

[12] C. Ludwig, *Practical lattice basis sampling reduction*, Ph.D thesis, Technische Universität Darmstadt, 2005.

[13] D. Micciancio, The shortest vector in a lattice is hard to approximate to within some constant, *SIAM J. Comput.* **30** (2001), no. 6, 2008–2035.

[14] D. Micciancio and W. Michael, Fast lattice point enumeration with minimal overhead, in: *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms – SODA 2015*, SIAM, Philadelphia (2015), 276–294.

[15] P. Q. Nguyen and B. Vallée, *The LLL algorithm*, Inf. Secur. Cryptography, Springer, Berlin, 2010.

[16] M. Schneider and N. Göttert, Random sampling for short lattice vectors on graphics cards, in: *Cryptographic Hardware and Embedded Systems – CHES 2011*, Lecture Notes in Computer Sci. 6917, Springer, Berlin (2011), 160–175.

[17] C. P. Schnorr, Lattice reduction by random sampling and birthday methods, in: *20th Annual Symposium of Theoretical Aspects on Computer Science – STACS 2003*, Lecture Notes in Computer Sci. 2606, Springer, Berlin (2003), 145–156.

[18] C. P. Schnorr and M. Euchner, Lattice basis reduction: Improved practical algorithms and solving subset sum problems, *Math. Program.* **66** (1994), 181–199.