J. KSIAM Vol.20, No.2, 107–121, 2016 http://dx.doi.org/10.12941/jksiam.2016.20.107

# ANALYSIS OF THE UPPER BOUND ON THE COMPLEXITY OF LLL ALGORITHM

YUNJU PARK[1] AND JAEHYUN PARK[2†]

[1]DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE, KOREA SCIENCE ACADEMY OF KOREA ADVANCED INSTITUTE OF SCIENCE AND TECHNOLOGY, KOREA
*E-mail address*: yunjupark@kaist.ac.kr

[2]DEPARTMENT OF ELECTRONIC ENGINEERING, PUKYONG NATIONAL UNIVERSITY, KOREA
*E-mail address*: jaehyun@pknu.ac.kr

ABSTRACT. We analyze the complexity of the *LLL* algorithm, invented by Lenstra, Lenstra, and Lovász as a a well-known lattice reduction (LR) algorithm which is previously known as having the complexity of $O(N^4 \log B)$ multiplications (or, $O(N^5(\log B)^2)$ bit operations) for a lattice basis matrix $\mathbf{H}(\in \mathbb{R}^{M\times N})$ where $B$ is the maximum value among the squared norm of columns of $\mathbf{H}$. This implies that the complexity of the lattice reduction algorithm depends only on the matrix size and the lattice basis norm. However, the matrix structures (i.e., the correlation among the columns) of a given lattice matrix, which is usually measured by its condition number or determinant, can affect the computational complexity of the LR algorithm. In this paper, to see how the matrix structures can affect the LLL algorithm's complexity, we derive a more tight upper bound on the complexity of LLL algorithm in terms of the condition number and determinant of a given lattice matrix. We also analyze the complexities of the LLL updating/downdating schemes using the proposed upper bound.

## 1. INTRODUCTION

Lattice reduction (LR) is a method to find the bases of the given lattice space close to the shortest vector, which has been successfully applied to cryptography, factoring the polynomials ( [1–3] and references therein), and multiple-input multiple output (MIMO) communication system [4–7]. One advantageous property of LR is to improve the conditioning of a given lattice basis matrix by multiplying a unimodular matrix which is a square integer matrix with determinant $\pm 1$.

One of famous LR algorithms due to the simplicity is the LLL algorithm invented by Lenstra, Lenstra, and Lovász [1] which has the complexity of $O(N^4 \log B)$ multiplications (or, $O(N^5(\log B)^2)$ bit operations) for the given lattice basis matrix $\mathbf{H}(\in \mathbb{R}^{M\times N})$ where $B$ is

Received by the editors February 16 2016; Revised May 25 2016; Accepted in revised form May 25 2016; Published online June 1 2016.

2000 *Mathematics Subject Classification.* 15B99.

*Key words and phrases.* Lattice Reduction, LLL algorithm, Complexity analysis, Matrix Updating/Downdating.

† Corresponding author.

the maximum value among the squared norm of columns of $\mathbf{H}$. However, it is a quite loose bound and depends only on the matrix size and the norm of each column which does not tell much about how the condition number or the determinant of the lattice basis matrix affect on the complexity of the LLL algorithm. For example, since the reduced lattice bases tend to be orthogonal to each other to become close to the shortest vector, we can intuitively expect that if the lattice basis matrix is well-conditioned it requires less computational complexities. Accordingly, the matrix structures (i.e., the correlation among the columns) of a given lattice matrix, which is usually measured by its condition number or determinant, can affect the computational complexity of the LR algorithm.

In this paper, to see how the matrix structures can affect the LLL algorithm's complexity, we derive a more tight upper bound on the complexity of LLL algorithm in terms of the condition number and determinant of a given lattice matrix. Accordingly, the proposed upper bound gives an insight into the effects of the lattice structure on the complexities of the LR algorithm. In addition, by using the proposed upper bound, we analyze the LLL updating/downdating algorithms in [8]. From the numerical results, by using the proposed upper bound, we can infer the behavior of the required computational complexities of LLL updating/downdating algorithms according to the structures of a given lattice basis matrix and the updated rows/columns.

The rest of this paper is organized as follows. In Section 2 the LLL algorithm is briefly reviewed and the basic notations and definitions related with the LLL algorithm are introduced. In Section 3 a new upper bound of the LLL algorithm's complexity is derived. In Section 4 row-wise LLL updating and downdating methods are introduced and they are analyzed by using the proposed upper bound. Several simulation results for various conditions are given in Section 5. Concluding remarks are made in Section 6.

The superscripts in $\mathbf{A}^T$ and $\mathbf{A}^{-1}$ denote, respectively, the transposition and the inverse of the matrix $\mathbf{A}$. $\|\mathbf{A}\|_2$ and $\det(\mathbf{A})$ denote 2-norm and determinant of $\mathbf{A}$, respectively. $\lceil a \rfloor$ denotes the nearest integer of a real number $a$. $\mathbf{I}_N$ and $\mathbf{0}_{M,N}$ denote an $N \times N$ identity matrix and a zero $M \times N$ matrix, respectively, and $\mathbf{e}_i$ denotes the $i$-th column of $\mathbf{I}_N$. Finally $\mathbf{A}(i:j, k:l)$ denotes the submatrix of $\mathbf{A}$ with elements from the $i$-th row to the $j$-th row and from the $k$-th column to the $l$-th column.

## 2. LATTICE REDUCTION: LLL ALGORITHM

Lattice $\mathcal{L}(\mathbf{H})$ is defined as $\sum_{n=1}^{N} \mathbf{h}_n s_n$, where $\mathbf{h}_n$, the $n$-th column vector of $\mathbf{H}$ $\left(\in \mathbb{R}^{M \times N}, M \geq N\right)$, is a basis vector of lattice and $s_n \in \mathbb{Z}$. A lattice basis matrix $\mathbf{H}$ is (LLL) lattice reduced if

$$|\mu_{i,j}| \leq 1/2 \text{ for } 1 \leq i < j \leq N \tag{2.1}$$

and

$$3/4\|\mathbf{h}_{i-1}^o\|^2 \leq \|\mathbf{h}_i^o + \mu_{i-1,i}\mathbf{h}_{i-1}^o\|^2 \text{ for } 1 < i \leq N, \tag{2.2}$$

where $\mathbf{h}_j^o = \mathbf{h}_j - \sum_{i=1}^{j-1} \mu_{i,j}\mathbf{h}_i^o$ with $\mu_{i,j} = \frac{<\mathbf{h}_i, \mathbf{h}_j^o>}{<\mathbf{h}_i^o, \mathbf{h}_i^o>}$, which can be obtained through the Gram-Schmidt orthogonalization process. Equivalently, a lattice basis matrix $\mathbf{H}$ with QR decomposition $\mathbf{H} = \mathbf{QR}$, where $\mathbf{Q}(\in \mathbb{R}^{M\times N})$ is orthogonal and $\mathbf{R}(\in \mathbb{R}^{N\times N})$ is upper triangular, is lattice reduced if

$$|r_{i,j}| \leq 1/2|r_{i,i}| \text{ for } 1 \leq i < j \leq N \tag{2.3}$$

and

$$3/4r_{i-1,i-1}^2 \leq r_{i,i}^2 + r_{i-1,i}^2 \text{ for } 1 < i \leq N, \tag{2.4}$$

where $r_{i,j}$ is the $(i,j)$-th element of $\mathbf{R}$. Note that $\mu_{i,j} = \frac{r_{i,j}}{r_{i,i}}$. If Eqn. (2.1) (equivalently, Eqn. (2.3)) is satisfied, $\mathbf{H}$ is called *size-reduced* and if Eqn. (2.2) (equivalently, Eqn. (2.4)) is satisfied, $\mathbf{H}$ is called *two-reduced*.[1] The constant $\frac{3}{4}$ in Eqn. (2.2) and Eqn. (2.4) could be arbitrarily replaced by any fixed real number within $(1/4, 1)$ [1].

LR algorithm is to reduce the norm of each basis of lattice by a linear combination of bases with integer coefficients which is equivalent to post-multiplying $\mathbf{H}$ with a unimodular matrix $\mathbf{T}$ such that $\mathbf{HT}$ is lattice reduced. In Table 1, LLL algorithm to compute $\mathbf{T}$ such that $\mathbf{HT}$ is lattice reduced is summarized [1].

TABLE 1. LLL algorithm

| | |
|---|---|
| 1 | $\mathbf{T} = \mathbf{I}_N$, $\mathbf{H} = \mathbf{QR}$ |
| 2 | $i = 2$ |
| 3 | while $i \leq N$ |
| 4 | for $l = i-1, ..., 1$ |
| 5 | [$\mathbf{R}$, $\mathbf{T}$] = Size-reduction($\mathbf{R}$, $\mathbf{T}$, $i$, $l$) |
| 6 | end |
| 7 | if $3/4r_{i-1,i-1}^2 > r_{i,i}^2 + r_{i-1,i}^2$ |
| 8 | [$\mathbf{R}$, $\mathbf{T}$] = two-reduction($\mathbf{R}$, $\mathbf{T}$, $i$ ) |
| 9 | $i = \max\{i-1,\ 2\}$ |
| 10 | else |
| 11 | $i = i + 1$ |
| 12 | end |
| 13 | end |

Note that the LLL algorithm is mainly composed of two subroutines – size-reduction and two-reduction routines in Table 2. In size-reduction step, by the modular operation, Eqn. (2.3) can be satisfied. In two-reduction step, if Eqn. (2.4) is not satisfied, the corresponding adjacent columns are swapped and the triangular form of matrix $\mathbf{R}$ is recovered by the Givens rotation. Note that since the Givens rotation preserves the norm of each column Eqn. (2.4) can be satisfied by simply swapping the columns. Then, $\mathbf{HT}$ with the unimodular matrix $\mathbf{T}$ obtained from LLL algorithm satisfies the LLL lattice reduction conditions (Eqns. (2.1) and (2.2)).

[1]Note that since $r_{i-1,i}^2 \leq 1/4r_{i-1,i-1}^2$ from Eqn. (2.3), Eqn. (2.4) becomes $r_{i-1,i-1}^2 \leq 2r_{i,i}^2$.

TABLE 2. Subroutines of LLL algorithm

| | [$\mathbf{R}$, $\mathbf{T}$]= Size-reduction($\mathbf{R}$, $\mathbf{T}$, $i$, $l$) |
|---|---|
| 1 | $\mu = \lceil r_{l,i}/r_{l,l} \rfloor$ |
| 2 | if $\mu \neq 0$ |
| 3 | $\mathbf{R}(1:l,i) = \mathbf{R}(1:l,i) - \mu\mathbf{R}(1:l,l)$ |
| 4 | $\mathbf{T}(1:N,i) = \mathbf{T}(1:N,i) - \mu\mathbf{T}(1:N,l)$ |
| 5 | end |
| | [$\mathbf{R}$, $\mathbf{T}$]= two-reduction($\mathbf{R}$, $\mathbf{T}$, $i$) |
| 1 | Swap columns $i-1$ and $i$ in $\mathbf{R}$ and $\mathbf{T}$ |
| 2 | Triangularize $\mathbf{R}$ using Givens rotation matrix $\mathbf{\Theta}$ |
| 3 | $\mathbf{R}(i-1:i, i-1:N) = \mathbf{\Theta}\mathbf{R}(i-1:i, i-1:N)$ |

Note that line 3 and 4 of Size-reduction subroutine in Table 2 can be represented in terms of matrices as

$$\mathbf{R} = \mathbf{R}\mathbf{S}_{li}, \quad \mathbf{T} = \mathbf{T}\mathbf{S}_{li}, \tag{2.5}$$

where $\mathbf{S}_{li} = \mathbf{I}_N - \mu\mathbf{e}_l\mathbf{e}_i$. Similarly, Line 1 of two-reduction subroutine in Table 2 can be rewritten as

$$\mathbf{R} = \mathbf{R}\mathbf{P}_i, \quad \mathbf{T} = \mathbf{T}\mathbf{P}_i, \tag{2.6}$$

where

$$\mathbf{P}_i = \begin{bmatrix} \mathbf{I}_{i-2} & & & \\ & 0 & 1 & \\ & 1 & 0 & \\ & & & \mathbf{I}_{N-i} \end{bmatrix}. \tag{2.7}$$

Therefore, $\mathbf{T}$ is a product of sequence of matrices $\mathbf{S}_{li}$ and $\mathbf{P}_i$.

To evaluate the computational complexity of LLL algorithm, we introduce the following lemma [9].

**Lemma 1.** *If we define $D \triangleq r_{11}^{2N} \cdot r_{22}^{2(N-1)} \cdot \ldots \cdot r_{NN}^{2}$, the size-reduction step does not affect $D$, while the two-reduction step decreases $D$ by a factor of at least 3/4.*

Accordingly, the number of $while$ loop iterations in the LLL algorithm of Table 1 is at most $O(\log D_0)$ [2], where $D_0 = |\mathbf{h}_1|^{2N} \cdot |\mathbf{h}_2|^{2(N-1)} \cdot \ldots \cdot |\mathbf{h}_N|^2$. If we set $B = \max\{|\mathbf{h}_1|^2, \ldots, |\mathbf{h}_N|^2\}$, $O(\log D_0) \simeq O(N^2 \log B)$. Since the size-reduction step in the lines $4-6$ of Table 1 requires only $O(N^2)$ multiplications, total $O(N^4 \log B)$ multiplications are required. Each real number used during the process of the algorithm is bounded by $O(N \log B)$ bits [1]. Hence, total $O(N^5(\log B)^2)$ bit operations are required in LLL algorithm. Note that because the LLL algorithm's complexity depends on the number of while loop iterations in the LLL algorithm,

[2]As for Lemma 1, in this paper log denotes the logarithm with a basis $4/3$

we would investigate the number of the two-reduction steps as a measure of the computational complexity.

**Remark 1.** *The LLL algorithm does not change the absolute value of the determinant of the lattice basis matrix since* $\det(\mathbf{T}) = \pm 1$. *Therefore,*

$$\det(\mathbf{H}^T\mathbf{H}) = \prod_{i=1}^{N} r_{ii}^2 = \prod_{i=1}^{N} \tilde{r}_{ii}^2, \tag{2.8}$$

*where* $\tilde{r}_{ii}^2$ *is the component of the upper triangular matrix* $\tilde{\mathbf{R}}(\in \mathbb{R}^{N\times N})$ *in the QR decomposition of the reduced lattice basis matrix* $\mathbf{G} = \mathbf{HT}$.

## 3. A NEW UPPER BOUND OF THE LLL ALGORITHM'S COMPLEXITY

The complexity derived from Lemma 1 in Section 2 indicates that it depends only on the matrix size and the lattice basis norm. In this section, we would derive a new upper bound which can be used to analyze the complexity of LLL algorithm as well as other variants of the LLL basis algorithms such as the LLL updating/downdating algorithms [8]. From Section 2, the number of two-reduction steps is critical in the complexity analysis and, from $D$ defined in Lemma 1, the upper bound of the number of two-reduction steps can be induced. Accordingly, in what follows, to get a more tight upper bound for the number of two reduction steps, we derive the lower bound of $D$, denoted as $D_l$.

Because the determinant of lattice basis matrix does not change as discussed in Remark 1, we can have the following lemma.

**Lemma 2.** *Let* $\mathbf{H}(\in \mathbb{R}^{M\times N}, M \geq N)$ *have the preconditioning matrix* $\mathbf{T}(\in \mathbb{Z}^{N\times N})$ *which is a unimodular matrix computed by LLL algorithm in Table 1. Then* $\mathbf{HT}$ *has a QR decomposition as*

$$\mathbf{G} = \mathbf{HT} = \tilde{\mathbf{Q}}\tilde{\mathbf{R}}, \tag{3.1}$$

*where* $\tilde{\mathbf{Q}}(\in \mathbb{R}^{M\times N})$ *is orthogonal and* $\tilde{\mathbf{R}}(\in \mathbb{R}^{N\times N})$ *is upper triangular satisfying Eqns. (2.3) and (2.4). Then,*

$$\max_{1\leq i,j\leq N} \frac{|\tilde{r}_{ii}|}{|\tilde{r}_{jj}|} \leq \max_{1\leq i,j\leq N} \frac{|r_{ii}|}{|r_{jj}|} \leq \kappa(\mathbf{H}), \tag{3.2}$$

*where* $\kappa(\mathbf{A})$ *is the condition number of a matrix* $\mathbf{A}$.

*Proof.* For the first inequality in Eqn. (3.2) ($\max_{1\leq i,j\leq N} \frac{|\tilde{r}_{ii}|}{|\tilde{r}_{jj}|} \leq \max_{1\leq i,j\leq N} \frac{|r_{ii}|}{|r_{jj}|}$), we let $(i_{mx}, j_{mn}) = \arg_{i,j} \max_{1\leq i,j\leq N} \frac{|r_{ii}|}{|r_{jj}|}$. Note that the two-reduction step can only change the diagonal elements of $\mathbf{R}$ during the LLL process. Accordingly, for the numerator $|r_{i_{mx}i_{mx}}|$, because $|r_{i_{mx}i_{mx}}| \geq |r_{ii}|$ for $1 \leq i \leq N$, two-reduction step is performed during the LLL process only when

$$3/4 r_{i_{mx}i_{mx}}^2 > r_{i_{mx}+1 i_{mx}+1}^2 + r_{i_{mx}i_{mx}+1}^2,$$

which alters $r_{i_{mx}i_{mx}}$ into $r'_{i_{mx}i_{mx}}$ with $|r'_{i_{mx}i_{mx}}| \leq |r_{i_{mx}i_{mx}}|$. Because the Givens rotation preserves the norm of each column, we have

$$|r'_{i_{mx}i_{mx}}| \geq |r_{i_{mx+1}i_{mx+1}}|. \tag{3.3}$$

In addition, because $r_{i_{mx}i_{mx}}r_{i_{mx+1}i_{mx+1}} = r'_{i_{mx}i_{mx}}r'_{i_{mx+1}i_{mx+1}}$ from Remark 1, together with Eqn. (3.3), we can get

$$|r'_{i_{mx+1}i_{mx+1}}| \leq |r_{i_{mx}i_{mx}}|.$$

Similarly, for the denominator $r_{j_{mn}j_{mn}}$ (which satisfies that $|r_{j_{mn}j_{mn}}| \leq |r_{ii}|$ for $1 \leq i \leq N$), the two-reduction step is performed during the LLL process when $3/4r^2_{j_{mn-1}j_{mn-1}} > r^2_{j_{mn}j_{mn}} + r^2_{j_{mn-1}j_{mn}}$, resulting

$$|r'_{j_{mn}j_{mn}}|,\ |r'_{j_{mn-1}j_{mn-1}}| \geq |r_{j_{mn}j_{mn}}|. \tag{3.4}$$

Therefore, during the LLL process, two-reduction step always reduces $\max\limits_{1\leq i,j\leq N} \dfrac{|r_{ii}|}{|r_{jj}|}$.

For the second inequality in Eqn. (3.2) ($\max_{1\leq i,j\leq N} \frac{|r_{ii}|}{|r_{jj}|} \leq \kappa(\mathbf{H})$), it can be easily derived from that $\kappa(\mathbf{H}) = \|\mathbf{R}\|_2\|\mathbf{R}^{-1}\|_2$, $\|\mathbf{R}\|_2 \geq \max|r_{ii}|$, and $\|\mathbf{R}^{-1}\|_2 \geq \max|r^{-1}_{jj}|$. □

From Lemma 2, we denote $(i'_{mx}, j'_{mn}) = \arg_{i,j}\max_{1\leq i,j\leq N}\frac{|\tilde{r}_{ii}|}{|\tilde{r}_{jj}|}$. Note that $\tilde{r}^2_{j'_{mn}j'_{mn}}$ has a lower bound as $\tilde{r}^2_{j'_{mn}j'_{mn}} = \kappa(\mathbf{H})^{-2}\tilde{r}^2_{i'_{mx}i'_{mx}}$. Let $\tilde{D} = \tilde{r}^{2N}_{11}\tilde{r}^{2(N-1)}_{22}\cdots\tilde{r}^2_{NN}$ for $\tilde{\mathbf{R}}$ as defined in Eqn. (3.1). Because $\tilde{r}^2_{ii} \geq \tilde{r}^2_{j'_{mn}j'_{mn}}$ for $1 \leq i \leq N$, $\tilde{D}$ has a lower bound when the set of diagonal elements is given as:

$$\begin{aligned}|\tilde{r}_{11}| = |\tilde{r}_{22}| = \cdots = |\tilde{r}_{N-1N-1}| = |\tilde{r}_{j_{mn}j_{mn}}| = \kappa(\mathbf{H})^{-1}|\tilde{r}_{i_{mx}i_{mx}}|,\\ |\tilde{r}_{NN}| = |\tilde{r}_{i_{mx}i_{mx}}|.\end{aligned} \tag{3.5}$$

Accordingly, we have the following corollary.

**Corollary 3.1.** *$\tilde{D}$ has a lower bound as:*

$$\tilde{D} \geq D_l = \frac{(\det(\mathbf{H}^T\mathbf{H}))^{(N+1)/2}}{(\kappa(\mathbf{H}))^{N-1}}. \tag{3.6}$$

*Proof.* Because $\det(\mathbf{H}^T\mathbf{H}) = \prod_{i=1}^N \tilde{r}^2_{ii}$, Eqn. (3.5) implies that

$$\det(\mathbf{H}^T\mathbf{H}) = \prod_{i=1}^N \tilde{r}^2_{ii} = \kappa(\mathbf{H})^{-2(N-1)}|\tilde{r}_{i_{mx}i_{mx}}|^{2N}. \tag{3.7}$$

That is, $|\tilde{r}_{i_{mx}i_{mx}}| = \det(\mathbf{H}^T\mathbf{H})^{1/2N}\kappa(\mathbf{H})^{2(N-1)/2N}$ and therefore, we can get

$$\begin{aligned}\tilde{r}^2_{11} = \tilde{r}^2_{22} = \cdots = \tilde{r}^2_{N-1N-1} = (\det(\mathbf{H}^T\mathbf{H}))^{1/N}(\kappa(\mathbf{H}))^{-2/N},\\ \tilde{r}^2_{NN} = (\det(\mathbf{H}^T\mathbf{H}))^{1/N}(\kappa(\mathbf{H}))^{2(N-1)/N}.\end{aligned} \tag{3.8}$$

Therefore $\tilde{r}_{ii}^2$ in Eqn. (3.8) induce the lower bound $D_l$ as:

$$D_l = \tilde{r}_{11}^{2N}\tilde{r}_{22}^{2(N-1)}\cdots\tilde{r}_{NN}^2 = \frac{(\det(\mathbf{H}^T\mathbf{H}))^{(N+1)/2}}{(\kappa(\mathbf{H}))^{N-1}} \tag{3.9}$$

□

Note that $D_l$ can be represented in terms of $\mathbf{R}$ as $\tilde{D}_l = \frac{(\det(\mathbf{R}))^{N+1}}{(\kappa(\mathbf{R}))^{N-1}}$. Therefore we can derive a new upper bound for the required number of two-reduction steps for LLL algorithm as $N_s = O(\log D_N)$, where

$$D_N = \frac{\prod_{i=1}^{N} r_{ii}^{2(N-i+1)}(\kappa(\mathbf{R}))^{N-1}}{(\det(\mathbf{R}))^{N+1}}. \tag{3.10}$$

**Remark 2.** *The new upper bound for the two-reduction steps indicating LLL complexity is dependent not only on the lattice basis matrix size and the lattice basis norm but also on its determinant and the condition number. That is, if the given lattice basis matrix is well conditioned or its determinant is large then the complexity can be diminished.*

**Remark 3.** *As a simple example of validity of the new upper bound, let us think about $\alpha\mathbf{I}_N$ where $\alpha \in \mathbb{R}$. While the previous upper bound for the number of the two-reduction steps is $O(\log D_0)$ where $D_0 = |\alpha|^{N(N+1)}$, the proposed upper bound is 0 because $D_N = 1$ which coincide with that $\alpha\mathbf{I}_N$ is already lattice-reduced.*

## 4. Complexity Analysis of LLL Updating and Downdating

In this section, we briefly introduce the row-wise updating/downdating methods [8] and analyze their complexities using the proposed upper bound in Section 3.

### 4.1. Row-wise Updating.

Let $\mathbf{H}(\in \mathbb{R}^{M\times N}, M \geq N)$ have a QR decomposition as

$$\mathbf{H} = \mathbf{Q}\mathbf{R}, \tag{4.1}$$

where $\mathbf{Q}(\in \mathbb{R}^{M\times N})$ is orthogonal and $\mathbf{R}(\in \mathbb{R}^{N\times N})$ is upper triangular and assume that given $\mathbf{H}$ we have the preconditioning matrix $\mathbf{T}(\in \mathbb{Z}^{N\times N})$ which is a unimodular matrix computed by LLL algorithm in Table 1. Here, the QR decomposition of $\mathbf{HT}(\in \mathbb{R}^{M\times N})$ is also known as:

$$\mathbf{G} = \mathbf{HT} = \tilde{\mathbf{Q}}\tilde{\mathbf{R}}, \quad \tilde{\mathbf{R}} \in \mathbb{R}^{N\times N} \tag{4.2}$$

where $\tilde{\mathbf{R}}$ satisfies Eqns. (2.3) and (2.4). We then want to find a new preconditioning matrix $\mathbf{T}_u$ after a new row $\mathbf{h}_r^T$ is added as:

$$\mathbf{G}_u = \mathbf{H}_u\mathbf{T}_u = \begin{bmatrix} \mathbf{h}_r^T \\ \mathbf{H} \end{bmatrix}\mathbf{T}_u, \quad \mathbf{h}_r^T \in \mathbb{R}^{1\times N} \tag{4.3}$$

where $\mathbf{G}_u$ is lattice-reduced. Because the QR decomposition of $\mathbf{HT}$ is known as Eqn. (4.2) we can update the new row $\mathbf{h}_r^T$ based on $\tilde{\mathbf{Q}}$ and $\tilde{\mathbf{R}}$ rather than using $\mathbf{Q}$ and $\mathbf{R}$ as follows:

$$\mathbf{G}' = \begin{bmatrix} \mathbf{h}_r^T\mathbf{T} \\ \mathbf{HT} \end{bmatrix} = \begin{bmatrix} \mathbf{h}_r^T\mathbf{T} \\ \tilde{\mathbf{Q}}\tilde{\mathbf{R}} \end{bmatrix} = \begin{bmatrix} 1 & \mathbf{0}_{1\times N} \\ \mathbf{0}_{N\times 1} & \tilde{\mathbf{Q}} \end{bmatrix}\begin{bmatrix} \mathbf{h}_r^T\mathbf{T} \\ \tilde{\mathbf{R}} \end{bmatrix}, \tag{4.4}$$

where $\begin{bmatrix} \mathbf{h}_r^T\mathbf{T} \\ \tilde{\mathbf{R}} \end{bmatrix}$ has a Hessenberg form. We can then recover the triangular form by using $N$ Givens rotations as:

$$\mathbf{J}_N^T \cdots \mathbf{J}_1^T \begin{bmatrix} \mathbf{h}_r^T\mathbf{T} \\ \tilde{\mathbf{R}} \end{bmatrix} = \tilde{\mathbf{R}}_u^{in}, \tag{4.5}$$

where $\mathbf{J}_i^T$ is the Givens rotation matrix forcing zero on the $(i+1,i)$-th entry of the Hessenberg matrix in Eqn. (4.5). We then compute the preconditioning matrix $\mathbf{T}_u$ by using $\tilde{\mathbf{R}}_u^{in}$ and $\mathbf{T}$ as initial parameters. Note that $\mathbf{Q}$ is not required during the LLL process.

Here, to analyze the number of two-reduction steps for the updating algorithm, we represent the squared norm of the diagonal elements of $\tilde{\mathbf{R}}_u^{in}$ in Eqn. (4.5) in terms of $\tilde{\mathbf{R}}$ and $\mathbf{h}_r^T\mathbf{T}$ ($\triangleq \mathbf{h}'^T$) as:

$$(\tilde{r}_{ii}^{in})^2 = \quad (k_i^{(i)})^2 = \tilde{r}_{ii}^2 + \textstyle\sum_{j=1}^{i-1} \tilde{r}_{ji}^2 \prod_{k=1}^{i-j} \sin^2\theta_i^{(k)} + h_i'^2 \prod_{j=1}^{i} \sin^2\theta_i^{(i-j)}, \tag{4.6}$$

where

$$\begin{aligned} k_i^{(j)} &= \sqrt{(k_i^{(j-1)}\sin\theta_i^{(j-1)})^2 + \tilde{r}_{ji}^2} \quad \text{for } 1 \le j \le i \le N, \\ \theta_i^{(j)} &= \cos^{-1}\frac{\tilde{r}_{jj}k_i^{(j-1)}\sin\theta_i^{(j-1)} + \tilde{r}_{ji}k_j^{(j-1)}\sin\theta_j^{(j-1)}}{k_j^{(j)}k_i^{(j)}} \quad \text{for } 1 \le j < i \le N, \\ \theta_i^{(i)} &= 0 \quad \text{for } 1 \le i \le N, \end{aligned} \tag{4.7}$$

with $(k_i^{(0)})^2 = h_i'^2$ and $\theta_i^{(0)} = 90^\circ$, and $h_i'$ is the $i$-th element of $\mathbf{h}'^T$. See also [8] for the details.

To get the upper bound of the number of two-reduction steps, we compute $D_N$ for $\tilde{\mathbf{R}}_u^{in}$ as defined in Eqn. (3.10):

$$\begin{aligned} D_N &= (k_1^{(1)})^{2N}(k_2^{(2)})^{2(N-2)}\cdots(k_N^{(N)})^2 \frac{(\kappa(\tilde{\mathbf{R}}_u^{in}))^{N-1}}{(\det(\tilde{\mathbf{R}}_u^{in}))^{N+1}} \\ &= (\tilde{r}_{11}^2 + h_1'^2)^N(\tilde{r}_{22}^2 + \tilde{r}_{12}^2\sin^2\theta_2^{(1)} + h_2'^2\sin^2\theta_2^{(1)}\sin^2\theta_2^{(0)})^{N-2}\cdots \\ &\quad (\tilde{r}_{NN}^2 + \sum_{j=1}^{N-1}\tilde{r}_{jN}^2\prod_{k=1}^{N-j}\sin^2\theta_N^{(k)} + h_N'^2\prod_{j=1}^{N}\sin^2\theta_N^{(N-j)})\frac{(\kappa(\tilde{\mathbf{R}}_u^{in}))^{N-1}}{(\det(\tilde{\mathbf{R}}_u^{in}))^{N+1}} \\ &= D_P D_S \prod_{i=1}^{N}\left(1 + \sum_{j=1}^{i-1}\frac{\tilde{r}_{ji}^2}{\tilde{r}_{ii}^2}\prod_{k=1}^{i-j}\sin^2\theta_i^{(k)} + \frac{h_i'^2}{\tilde{r}_{ii}^2}\prod_{j=1}^{i}\sin^2\theta_i^{(i-j)}\right)^{N-i+1}, \end{aligned} \tag{4.8}$$

where $D_P = \tilde{r}_{11}^{2N}\tilde{r}_{22}^{2(N-1)}\cdots\tilde{r}_{NN}^2\frac{(\kappa(\tilde{\mathbf{R}}))^{N-1}}{(\det(\tilde{\mathbf{R}}))^{N+1}}$ is associated with $\mathbf{HT}$, the already reduced lattice bases before adding a new row. Here, $D_S$ is given as:

$$D_S = \left(\frac{\kappa(\tilde{\mathbf{R}}_u^{in})}{\kappa(\tilde{\mathbf{R}})}\right)^{N-1}\left(\frac{\det(\tilde{\mathbf{R}})}{\det(\tilde{\mathbf{R}}_u^{in})}\right)^{N+1}. \tag{4.9}$$

Therefore, the number of two-reduction steps required in the updating algorithm is at most $O(\log D_u)$, where

$$D_u = D_S \prod_{i=1}^{N} \left( 1 + \sum_{j=1}^{i-1} \frac{\tilde{r}_{ji}^2}{\tilde{r}_{ii}^2} \prod_{k=1}^{i-j} \sin^2 \theta_i^{(k)} + \frac{h_i'^2}{\tilde{r}_{ii}^2} \prod_{j=1}^{i} \sin^2 \theta_i^{(i-j)} \right)^{N-i+1}. \tag{4.10}$$

Here, since $\tilde{r}_{ij}$ is from the QR decomposition of the already reduced lattice basis matrix in Eqn. (3.1), due to the two LR conditions of Eqns. (2.3) and (2.4), we can easily induce that $\tilde{r}_{ji}^2 \leq \tilde{r}_{ii}^2$ for $j < i$. Therefore, the second term in Eqn. (4.10) is always less than $i - 1$, which leads to the following inequality:

$$D_u \leq \tilde{D}_u, \tag{4.11}$$

where

$$\tilde{D}_u \triangleq D_S \prod_{i=1}^{N} \left( i + \frac{h_i'^2}{\tilde{r}_{ii}^2} \prod_{j=1}^{i} \sin^2 \theta_i^{(i-j)} \right)^{N-i+1}, \tag{4.12}$$

Accordingly, the required number of two-reduction steps for the LLL updating algorithm is upper bounded by $O(\log \tilde{D}_u)$.

**Remark 4.** *Whenever any new rows are added, due to the last two terms in Eqn. (4.6), the norm of each diagonal term always increases, from which it can be derived that $D_S \leq 1$. That is, from Eqn. (4.12), the upper bound of the number of two-reduction steps can be reduced. In other words, when a new row is added in the current lattice basis matrix, the correlation among the lattice bases tends to be decreased relatively (or the ratio of the norm of diagonal elements to that of off-diagonal element becomes larger). Accordingly, the increase of the row size induces less column subtractions in the size-reduction step, also resulting in less two-reduction steps in the LLL process.*

4.2. **Row-wise Downdating.** In this section, given $\mathbf{H}$ in Eqn. (4.1) with preconditioning matrix $\mathbf{T}$, we introduce the method in [8] to find the new preconditioning matrix $\mathbf{T}_d$ after the first row $\mathbf{h}_1^T$ is removed:

$$\mathbf{G}_d = \mathbf{H}_d \mathbf{T}_d, \quad \mathbf{H} = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{H}_d \end{bmatrix}, \quad \mathbf{H}_d \in \mathbb{R}^{M-1 \times N} \tag{4.13}$$

where $\mathbf{G}_d$ is lattice reduced. Since the LLL algorithm is independent on the orthogonal matrix of QR decomposition as discussed in Remark 1, we first start the following equation as:

$$\mathbf{H}_d^T \mathbf{H}_d = \mathbf{H}^T \mathbf{H} - \mathbf{h}_1 \mathbf{h}_1^T = \begin{bmatrix} \mathbf{H}^T & \mathbf{h}_1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_N & \mathbf{0} \\ \mathbf{0} & -1 \end{bmatrix} \begin{bmatrix} \mathbf{H} \\ \mathbf{h}_1^T \end{bmatrix}. \tag{4.14}$$

Since $\mathbf{H}^T \mathbf{H} = \mathbf{T}^{-T} \tilde{\mathbf{R}}^T \tilde{\mathbf{R}} \mathbf{T}^{-1}$, Eqn. (4.14) can be written as:

$$\mathbf{H}_d^T \mathbf{H}_d \;=\; \mathbf{T}^{-T} \tilde{\mathbf{R}}^T \tilde{\mathbf{R}} \mathbf{T}^{-1} - \mathbf{h}_1 \mathbf{h}_1^T$$

$$= \begin{bmatrix}\mathbf{T}^{-T}\tilde{\mathbf{R}}^T & \mathbf{h}_1\end{bmatrix}\begin{bmatrix}\mathbf{I}_N & \mathbf{0}\\ \mathbf{0} & -1\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{R}}\mathbf{T}^{-1}\\ \mathbf{h}_1^T\end{bmatrix}$$
$$= \mathbf{T}^{-T}\begin{bmatrix}\tilde{\mathbf{R}}^T & \mathbf{T}^T\mathbf{h}_1\end{bmatrix}\begin{bmatrix}\mathbf{I}_N & \mathbf{0}\\ \mathbf{0} & -1\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{R}}\\ \mathbf{h}_1^T\mathbf{T}\end{bmatrix}\mathbf{T}^{-1}. \quad (4.15)$$

Therefore,

$$\mathbf{T}^T\mathbf{H}_d^T\mathbf{H}_d\mathbf{T} = \begin{bmatrix}\tilde{\mathbf{R}}^T & \mathbf{T}^T\mathbf{h}_1\end{bmatrix}\begin{bmatrix}\mathbf{I}_N & \mathbf{0}\\ \mathbf{0} & -1\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{R}}\\ \mathbf{h}_1^T\mathbf{T}\end{bmatrix}. \quad (4.16)$$

Like Cholesky downdating, we can then recover the triangular form by applying the hyperbolic rotation as [10]

$$\mathbf{L}_N\cdots\mathbf{L}_1\begin{bmatrix}\tilde{\mathbf{R}}\\ \mathbf{h}_n^T\mathbf{T}\end{bmatrix} = \begin{bmatrix}\tilde{\mathbf{R}}_d^{in}\\ \mathbf{0}\end{bmatrix}, \quad (4.17)$$

where $\mathbf{L}_i$ is the hyperbolic rotation matrix forcing zero on the $i$-th entry of $\mathbf{h}_n^T\mathbf{T}$ ($\triangleq \mathbf{h}'^T$). Therefore, we can find the new preconditioning matrix $\mathbf{T}_d$ with initial parameters $\tilde{\mathbf{R}}_d^{in}$ and $\mathbf{T}$. The squared norm of the diagonal terms in $\tilde{\mathbf{R}}_d^{in}$ can be given as:

$$(\tilde{r}_{ii}^{in})^2 = \tilde{r}_{ii}^2 - (P(\tilde{\mathbf{r}}_{1:i-1,i}, h_i', \theta_{1:i-1}))^2 \geq 0, \quad (4.18)$$

where $P(\tilde{\mathbf{r}}_{1:i-1,i}, h_i', \theta_{1:i-1})$ is a polynomial of $\tilde{r}_{j,i}$, $h_i'$, $\sinh\theta_j$, and $\cosh\theta_j$ for $j = 1, ..., i-1$. For example, for $i = 1$, we can easily get $(\tilde{r}_{11}^i)^2 = \tilde{r}_{11}^2 - h_1'^2$. Accordingly, $D_N$ for $\tilde{\mathbf{R}}_d^{in}$ as defined in Eqn. (3.10) can be computed as:

$$D_N = (\tilde{r}_{11}^{in})^{2N}\cdots(\tilde{r}_{NN}^{in})^2\frac{(\kappa(\tilde{\mathbf{R}}_d^{in}))^{N-1}}{(\det(\tilde{\mathbf{R}}_d^{in}))^{N+1}}$$
$$= D_P D_S\prod_{i=1}^{N}\left(1 - \frac{1}{\tilde{r}_{ii}^2}(P(\tilde{\mathbf{r}}_{1:i-1,i}, h_i', \theta_{1:i-1}))^2\right)^{N-i+1}, \quad (4.19)$$

where $D_P = \tilde{r}_{11}^{2N}\tilde{r}_{22}^{2(N-1)}\cdots\tilde{r}_{NN}^2\frac{(\kappa(\tilde{\mathbf{R}}))^{N-1}}{(\det(\tilde{\mathbf{R}}))^{N+1}}$ is associated with $\mathbf{HT}$, the already reduced lattice bases before removing a row and $D_S$ is given as:

$$D_S = \left(\frac{\kappa(\tilde{\mathbf{R}}_d^{in})}{\kappa(\mathbf{R})}\right)^{N-1}\left(\frac{\det(\mathbf{R})}{\det(\tilde{\mathbf{R}}_d^{in})}\right)^{N+1}. \quad (4.20)$$

Because $\kappa(\tilde{\mathbf{R}}_d^{in}) \geq \kappa(\mathbf{R})$ and $\det(\mathbf{R}) \geq \det(\tilde{\mathbf{R}}_d^{in})$, $D_S \geq 1$, in general. Accordingly, the required number of two-reduction steps for the LLL downdating algorithm is upper bounded by $O(\log D_d)$, where

$$D_d = D_S\prod_{i=1}^{N}\left(1 - \frac{1}{\tilde{r}_{ii}^2}(P(\tilde{\mathbf{r}}_{1:i-1,i}, h_i', \theta_{1:i-1}))^2\right)^{N-i+1}. \quad (4.21)$$

## 5. NUMERICAL RESULTS

In this section, we discuss the numerical results obtained through Monte–Carlo simulation for various conditions. For each case, we generate 10000 lattice basis matrices randomly. Each entry of lattice basis matrix $\mathbf{H}$ is generated according to Gaussian distribution with zero-mean and unit-variance [3]. Newly updated row vector $\mathbf{h}_r^T$ also follows the Gaussian distribution with zero-mean and unit-variance. All the simulations are run by using MATLAB.

TABLE 3. Comparison of the average condition numbers and the average number of two-reduction steps.

| $M$ | $N$ | 8 | 10 | 12 | 14 | 16 | 18 | 20 |
|---|---|---|---|---|---|---|---|---|
| $N$ | $\kappa(\mathbf{H})$ | 154.30 | 228.70 | 208.58 | 603.65 | 266.06 | 293.43 | 375.31 |
| | $\kappa(\mathbf{HT})$ | 3.63 | 4.61 | 5.96 | 7.61 | 9.67 | 11.96 | 14.71 |
| | $N_s$ | 19.05 | 28.76 | 38.49 | 48.08 | 56.45 | 64.98 | 72.55 |
| | $\log(D_P)$ | 175.5 | 316.5 | 504.8 | 742.8 | 1030.1 | 1372.8 | 1766.8 |
| | $\log(D_N)$ | 142.37 | 208.80 | 283.01 | 365.60 | 454.50 | 554.22 | 666.02 |
| | $\log(D_P)/N_s$ | 9.21 | 11.01 | 13.11 | 15.45 | 18.25 | 21.13 | 24.35 |
| | $\log(D_N)/N_s$ | 7.47 | 7.26 | 7.35 | 7.60 | 8.05 | 8.53 | 9.18 |
| $N+2$ | $\kappa(\mathbf{H})$ | 10.66 | 13.16 | 16.01 | 18.65 | 21.62 | 24.46 | 26.59 |
| | $\kappa(\mathbf{HT})$ | 3.58 | 4.50 | 5.62 | 6.95 | 8.48 | 10.18 | 12.01 |
| | $N_s$ | 12.31 | 18.18 | 24.68 | 30.95 | 37.45 | 43.20 | 48.79 |
| | $\log(D_P)$ | 231.2 | 386.4 | 588.1 | 839.6 | 1142.0 | 1497.0 | 1907.5 |
| | $\log(D_N)$ | 83.77 | 126.20 | 178.01 | 235.22 | 302.88 | 375.42 | 454.41 |
| | $\log(D_P)/N_s$ | 18.78 | 21.25 | 23.83 | 27.13 | 30.50 | 34.65 | 39.10 |
| | $\log(D_N)/N_s$ | 6.81 | 6.94 | 7.21 | 7.60 | 8.09 | 8.69 | 9.31 |
| $N+4$ | $\kappa(\mathbf{H})$ | 6.52 | 8.00 | 9.47 | 11.02 | 12.46 | 14.02 | 15.56 |
| | $\kappa(\mathbf{HT})$ | 3.49 | 4.33 | 5.31 | 6.43 | 7.65 | 9.00 | 10.44 |
| | $N_s$ | 10.08 | 14.93 | 19.96 | 25.40 | 30.77 | 35.63 | 40.66 |
| | $\log(D_P)$ | 266.7 | 433.9 | 647.4 | 910.4 | 1225.3 | 1592.0 | 2015.0 |
| | $\log(D_N)$ | 64.62 | 99.42 | 140.45 | 188.72 | 242.87 | 303.91 | 373.26 |
| | $\log(D_P)/N_s$ | 26.47 | 29.06 | 32.43 | 35.85 | 39.82 | 44.68 | 49.56 |
| | $\log(D_N)/N_s$ | 6.41 | 6.66 | 7.04 | 7.43 | 7.89 | 8.53 | 9.18 |

Table 3 lists the data for various column sizes $N$ with $M = \{N,\ N+2,\ N+4\}$ – $\kappa(\mathbf{H})$, $\kappa(\mathbf{HT})$, $N_s$, $\log(D_P)$, and $\log(D_N)$ where $\mathbf{T}$ is obtained by using LLL algorithm and $N_s$ is the average numbers of two-reduction steps performed when the LLL algorithm is applied to $\mathbf{H}$. Here, $\log(D_P)$ is the upper bound of the number of two-reduction steps as defined in Lemma 1 and $\log(D_N)$ is a newly derived upper bound in Section 3. From the condition number, we

[3]Note that the alternative of using techniques to generate random unimodular matrices in [11–13] may be utilized, but in this paper, to consider the application of LLL to the engineering field (especially, wireless communications), Gaussian random distribution is adopted

can find that $\kappa(\mathbf{HT})$ is lower than $\kappa(\mathbf{H})$, regardless of matrix size. That is, through the LLL algorithm, we can find the better-conditioned or shorter lattice basis set (or, matrix) than the original lattice basis set. In comparison of the average numbers of the two-reduction steps ($N_s$), the case for $M = N$ requires more two-reduction steps than those for $M = \{N+2, N+4\}$. However, the conventional upper bound ($\log(D_P)$) for $N = M$ has the smallest values compared to the other cases ($M = \{N+2, N+4\}$). That is, it does not catch that the number of two-reduction steps decreases when $M-N$ increases. Interestingly, the proposed new upper bound ($\log(D_N)$) decreases as $M-N$ increases, which coincides with the observation that the average numbers of the two-reduction steps also decreases for increasing $M-N$. Accordingly, from the proposed upper bound, we can infer the behavior of the computational complexity of lattice reduction according to the change of the matrix structure.

Table 4 shows the data for various column sizes $N$ for $\mathbf{H}_a$, where $\mathbf{H}_a = [\mathbf{h}_r^T;\ \mathbf{H}]$ ($\mathbf{H} \in \mathbb{R}^{M\times N},\ M = N$, and $\mathbf{h}_r^T \in \mathbb{R}^{1\times N}$). We evaluate $\kappa(\mathbf{H}_a)$, $\kappa(\mathbf{H}_a\mathbf{T}_1)$, $\kappa(\mathbf{H}_a\mathbf{T}_2)$ where $\mathbf{T}_1$ and $\mathbf{T}_2$ are respectively obtained by using the updating algorithm and by using the conventional LLL algorithm. We also compare the average numbers of two-reduction steps – $N_u$ indicates that the updating algorithm in Section 4.1 is applied to $\mathbf{H}_a$ and $N_t$ indicates that the conventional LLL algorithm is applied to $\mathbf{H}_a$. From the condition numbers in Table 4, both LLL algorithm and the updating algorithm exhibit similar condition numbers. In comparison of the average numbers of the two-reduction steps, the updating algorithm performs much less two-reduction steps compared to the case when the LLL algorithm is applied to $\mathbf{H}_a$ without updating algorithm. That is, the proposed updating scheme removes the redundant computational complexities.

TABLE 4. Comparison of the average condition numbers and the average number of two-reduction steps for the row-wise updating algorithm when $M = N$.

| $N$ | $\kappa(\mathbf{H}_a)$ | $\kappa(\mathbf{H}_a\mathbf{T}_1)$ | $\kappa(\mathbf{H}_a\mathbf{T}_2)$ | $N_u$ | $N_t$ |
|---|---|---|---|---|---|
| 8 | 17.96 | 3.59 | 3.60 | 4.75 | 14.46 |
| 10 | 23.01 | 4.50 | 4.56 | 6.37 | 21.53 |
| 12 | 28.26 | 5.70 | 5.76 | 8.18 | 29.22 |
| 14 | 33.46 | 7.13 | 7.26 | 9.55 | 36.60 |
| 16 | 37.16 | 8.84 | 9.01 | 10.97 | 43.27 |
| 18 | 41.10 | 10.81 | 10.97 | 12.32 | 50.12 |
| 20 | 47.76 | 12.94 | 13.10 | 13.39 | 56.03 |

Table 5 shows the data for the same simulation environments as in Table 4 except $M = N+3$. We can find the similar observation that the updating algorithm requires less two-reduction steps indicating the less computational complexities. Note that the case for $M = N+3$ requires less two-reduction steps than that for $M = N$, which is consistent with the discussion in Remark 4.

TABLE 5. Comparison of the average condition numbers and the average number of two-reduction steps for the row-wise updating algorithm when $M = N + 3$.

| $N$ | $\kappa(\mathbf{H}_a)$ | $\kappa(\mathbf{H}_a\mathbf{T}_1)$ | $\kappa(\mathbf{H}_a\mathbf{T}_2)$ | $N_u$ | $N_t$ |
|---|---|---|---|---|---|
| 8 | 6.55 | 3.46 | 3.48 | 1.46 | 10.10 |
| 10 | 7.99 | 4.28 | 4.32 | 2.15 | 14.98 |
| 12 | 9.48 | 5.26 | 5.31 | 2.87 | 20.08 |
| 14 | 10.99 | 6.36 | 6.41 | 3.63 | 25.40 |
| 16 | 12.51 | 7.59 | 7.64 | 4.53 | 30.81 |
| 18 | 13.95 | 8.93 | 8.98 | 5.24 | 35.53 |
| 20 | 15.52 | 10.38 | 10.44 | 6.12 | 40.28 |

Table 6 and Table 7 show the condition numbers and the numbers of two-reduction steps for the outputs with and without the row-wise downdating method for the submatrix $\mathbf{H}_b$ of $\mathbf{H}$, where $\mathbf{H} = [\mathbf{h}_1^T;\ \mathbf{H}_b]$, $\mathbf{H} \in \mathbb{R}^{M\times N}$, and $\mathbf{h}_1^T \in \mathbb{R}^{1\times N}$. Again, we compare $\kappa(\mathbf{H}_b)$, $\kappa(\mathbf{H}_b\mathbf{T}_1)$, $\kappa(\mathbf{H}_b\mathbf{T}_2)$ where $\mathbf{T}_1$ and $\mathbf{T}_2$ are respectively obtained by using the downdating algorithm and by using the conventional LLL algorithm. In Table 6, we set $M = N + 1$ and can also find that both $\kappa(\mathbf{H}_b\mathbf{T}_1)$ and $\kappa(\mathbf{H}_b\mathbf{T}_2)$ have lower values than $\kappa(\mathbf{H}_b)$. Moreover, from the results for $M = N + 3$ in Table 7, as the difference $M - N$ becomes higher, the less computational complexities are required.

TABLE 6. Comparison of the average condition numbers and the average number of two-reduction steps for the row-wise downdating algorithm when $M = N + 1$.

| $N$ | $\kappa(\mathbf{H}_b)$ | $\kappa(\mathbf{H}_b\mathbf{T}_1)$ | $\kappa(\mathbf{H}_b\mathbf{T}_2)$ | $N_d$ | $N_t$ |
|---|---|---|---|---|---|
| 8 | 136.59 | 3.60 | 3.61 | 7.13 | 19.25 |
| 10 | 299.28 | 4.60 | 4.63 | 10.41 | 28.80 |
| 12 | 183.53 | 5.90 | 5.94 | 13.55 | 38.57 |
| 14 | 2205.13 | 7.55 | 7.62 | 16.59 | 48.22 |
| 16 | 379.73 | 9.53 | 9.63 | 19.21 | 57.12 |
| 18 | 327.57 | 11.89 | 12.00 | 21.47 | 65.01 |
| 20 | 518.40 | 14.49 | 14.64 | 23.48 | 72.16 |

## 6. CONCLUSION

In this paper, we derive a new upper bound for the number of two-reduction steps in LLL algorithm, which is a critical parameter in evaluating the computational complexity of LLL algorithm. Because the newly derived upper bound is expressed by the determinant and the condition number of the given lattice basis matrix relating with its matrix structure, the proposed

TABLE 7. Comparison of the average condition numbers and the average number of two-reduction steps for the row-wise downdating algorithm when $M = N + 4$.

| $N$ | $\kappa(\mathbf{H}_b)$ | $\kappa(\mathbf{H}_b\mathbf{T}_1)$ | $\kappa(\mathbf{H}_b\mathbf{T}_2)$ | $N_d$ | $N_t$ |
|---|---|---|---|---|---|
| 8 | 7.98 | 3.51 | 3.53 | 2.20 | 10.98 |
| 10 | 9.84 | 4.37 | 4.41 | 3.20 | 16.36 |
| 12 | 11.81 | 5.43 | 5.46 | 4.38 | 22.01 |
| 14 | 13.63 | 6.62 | 6.68 | 5.60 | 27.70 |
| 16 | 15.61 | 7.98 | 8.02 | 6.59 | 33.56 |
| 18 | 17.65 | 9.47 | 9.57 | 7.84 | 38.79 |
| 20 | 19.28 | 11.08 | 11.14 | 8.89 | 43.93 |

upper bound gives an insight into the effects of the lattice structure on the complexities of the LR algorithm. From the numerical results, while the conventional upper bound does not catch that the average number of two-reduction steps decreases when the number of rows increases, the proposed new upper bound decreases as the number of rows increases, which agrees with the changes in the average number of two-reduction steps. That is, from the proposed upper bound, we can infer the behavior of the computational complexity of lattice reduction according to the change of the matrix structure. We have also analyzed the row-wise LLL updating and downdating algorithms when a new basis row is added or when an existing basis row is removed in a given lattice basis matrix. Through the updating and downdating schemes we can eliminate the redundant complexities in finding newly updated preconditioning (unimodular) matrix. Especially, by the new upper bound, we can expect that the required computational complexities would be diminished when the row size becomes large, which is demonstrated by the simulation results.

## ACKNOWLEDGMENTS

This work was supported by a Research Grant of Pukyong National University (2015 year).

## REFERENCES

[1] A.K. Lenstra, J.H.W. Lenstra, and L. Lovász, *Factoring polynomials with rational coefficients*, Math. Annal., **261**(4) (1982), 515–534.

[2] P. Nguyen and J. Stern, *Lattice reduction in cryptology: An update*, Algorithm. Numb. Theor.: 4th Intern. Symp. ANTS-IV, Lecture Notes in Comput. Sci., **1838** (2000), Springer, 85–112.

[3] C.P. Schnorr, *A hierarchy of polynomial time lattice basis reduction algorithms*, Theor. Comput. Sci., **53** (1987), 201–224.

[4] K. Lee, J. Chun, and L. Hanzo, *Optimal lattice-reduction aided successive interference cancellation for mimo systems*, IEEE Trans. Wir. Comm., **6**(7) (2007), 2438–2443.

[5] C. Windpassinger and R.F.H. Fischer, *Low-complexity near-maximum-likelihood detection and precoding for MIMO systems using lattice reduction*, Proc. IEEE Inform. Theor. Workshop (ITW), (2003), 345–348.

[6] H.Yao and G.W. Wornell, *Lattice-reduction-aided detectors for MIMO communication systems*, Proc. IEEE Glob. Telec. Conf., Taipei, Taiwan, **1** (2002), 424–428.

[7] D.Wübben, R.Böhnke, V.Kühn, and K.D. Kammeyer, *Near-maximum-likelihood detection of MIMO systems using MMSE-based lattice-reduction*, Proc. IEEE Int. Conf. Comm., **2** (2004), 798–802.
[8] J. Park and Y. Park, *Efficient lattice reduction updating and downdating methods and analysis*, J. KSIAM, **19**(2) (2015), 171–188.
[9] M.C. Cary, *Lattice basis reduction: Algorithms and applications*, Unpublished draft available at https://www.researchgate.net/publication/265107426_Lattice_Basis_Reduction_Algorithms_and_Applications, (2002).
[10] G.H. Golub and C.F.V. Loan, *Matrix Computations*, 3rd ed. Baltimore: Johns Hopkins Univ. Press, (1996).
[11] W. Backes and S. Wetzel, *An efficient LLL gram using buffered transformations*, Proc. Comput. Alg. Sci. Comp., LNCS2005, **4770** (2007), 31–44.
[12] W. Backes and S. Wetzel, *Heuristics on lattice basis reduction in practice*, ACM J. Exp. Algor., **7** (2002).
[13] D.G. Papachristoudis, S.T. Halkidis, and G. Stephanides, *An experimental comparison of some LLL-type lattice basis reduction algorithms*, Int. J. Appl. Comput. Math., **1**(3) (2015), 327–342.