GfcLLL: A Greedy Selection Based Approach for Fixed-Complexity LLL Reduction

Jinming Wen and Xiao-Wen Chang

Abstract—A greedy selection based approach GfcLLL for fixed-complexity LLL reductions (FCLLL) is proposed. Two greedy selection strategies are presented. Simulations show that each of the two strategies gives Babai points with lower bit error rate in a more or less equal or much shorter CPU time than existing FCLLL algorithms.

Index Terms—Integer least squares problem, fixed-complexity LLL reduction, success probability, GfcLLL.

I. INTRODUCTION

In communications and some other applications, we encounter the following linear model:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{v}, \quad \mathbf{v} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

(1)

where $\mathbf{y} \in \mathbb{R}^m$ is an observation vector, $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a deterministic model matrix with full column rank, $\hat{\mathbf{x}}$ is an unknown integer parameter vector, $\mathbf{v} \in \mathbb{R}^m$ is a noise vector following the Gaussian distribution $\mathcal{N}(0, \sigma^2 \mathbf{I})$.

A common method to estimate $\hat{\mathbf{x}}$ from (1) is to solve the following ordinary integer least squares (ILS) problem:

$$\min_{\mathbf{x} \in \mathbb{Z}^n} \| \mathbf{y} - \mathbf{A}\mathbf{x} \|_2$$

(2)

whose solution is the maximum likelihood estimator of $\hat{\mathbf{x}}$. A typical method for solving (2) is a sphere decoder, which searches for the optimal solution in a hyper-ellipsoid. Before the search process starts, a lattice reduction strategy, typically the LLL reduction [1], is applied, which transforms $\mathbf{A}$ to an upper triangular matrix. Since the ILS problem (2) is NP-hard [2], for some real-time applications, an approximate solution, which can be produced quickly, is computed instead. One often used approximate solution is the ordinary Babai point $\mathbf{x}^\text{mb}$, produced by Babai’s nearest plane algorithm [3]. In communications, Babai point is referred to as the successive interference cancelation decoder. It is shown in [4] that applying the LLL algorithm [1] in the reduction process can always increase the success probability of $\mathbf{x}^\text{mb}$, which is the probability of $\mathbf{x}^\text{mb}$ being equal to $\hat{\mathbf{x}}$.

In communications, often the parameter vector $\hat{\mathbf{x}}$ is subject to a box constraint (after some transformations), i.e.,

$$\hat{\mathbf{x}} \in \mathcal{B} := \{ \mathbf{x} : l \leq \mathbf{x} \leq \mathbf{u}, \quad \mathbf{x} \in \mathbb{Z}^n \}.$$  

(3)

In this situation, unfortunately, it is difficult to use the LLL reduction in solving (2) by a sphere decoding method, because after size reductions (a main component of the LLL reduction), the box constraint would become too complicated to handle in the search process. However, one can first use the LLL reduction to get the LLL-aided ordinary Babai point $\mathbf{x}^\text{mb}$ and then round it into the constraint box $\mathcal{B}$ to get an estimator of $\hat{\mathbf{x}}$ (see, e.g., [5]), to be denoted by $\mathbf{x}^\text{mb\text{-lll}}$. Another method is to use a column permutation strategy instead of the LLL reduction for preprocessing and then take the box constraint into account to get a box-constrained Babai point, to be denoted by $\mathbf{x}^\text{bc}$. Simulations show that usually the bit error rate (BER) of the LLL-aided $\mathbf{x}^\text{mb\text{-lll}}$ is smaller than that of $\mathbf{x}^\text{bc}$, see, e.g., [6].

The LLL reduction is useful to improve the accuracy of the Babai points for both unconstrained and box-constrained cases. However, the running time of the LLL reduction algorithm varies much from matrices to matrices even for a fixed dimension. And it was shown in [7] that in the MIMO context, the worst-case computational cost of the LLL reduction is not even bounded by a function of $n$. This may cause problems for the real-time communications system from the implementation point of view. To address this issue, some so-called fixed-complexity LLL reduction (FCLLL) algorithms have been proposed [8], [9], [10]. An FCLLL reduction is to make the reduction to be close to the LLL reduction in a more or less fixed computational cost for the same dimensions no matter what the matrix $\mathbf{A}$ is.

In this paper we will propose a new approach for FCLLL reduction. Unlike the existing approaches, which use predefined traversal order for selecting two consecutive columns for size reduction and permutation, our new approach uses a traversal order based on greedy selection strategies. It is motivated by increasing success probability of the Babai point. We will propose two greedy selection strategies. Simulations show that our greedy selection based FCLLL algorithms can produce Babai points with lower BER than the FCLLL algorithms proposed in [8], [9], [10], while the former costs more or less equal or much less CPU time than the latter.

The rest of this paper is organized as follows. In section II, we introduce the LLL and FCLLL reductions. In section III, we present our new algorithms. In section IV, we do some simulations to show the effectiveness and efficiency of the new algorithms. Finally we summarize this paper in section V.

For $\mathbf{x} \in \mathbb{R}^n$, we use $\lfloor \mathbf{x} \rfloor$ to denote its nearest integer vector, i.e., each entry of $\mathbf{x}$ is rounded to its nearest integer (if there is a tie, the one with smaller magnitude is chosen).
II. BACKGROUND

In this section, we first introduce the LLL reduction, and the success probability of the Babai point which is the motivation for our new algorithm, and then we briefly review some recent FCLLL algorithms, which will be used for comparisons later.

A. The LLL reduction and success probability of the Babai point

Assume that $A$ in (1) has the QR factorization

$$A = [Q, \tilde{Q}] \begin{bmatrix} R \\ 0 \end{bmatrix}$$  \hspace{1cm} (4)

where $[Q, \tilde{Q}] \in \mathbb{R}^{m \times m}$ is orthonormal and $R \in \mathbb{R}^{n \times n}$ is upper triangular. Define $\bar{y} = Q^T y$ and $\bar{v} = Q^T v$, then (1) can be transformed to

$$\bar{y} = R\bar{x} + \bar{v}, \hspace{0.5cm} \bar{v} \sim \mathcal{N}(0, \sigma^2 I).$$  \hspace{1cm} (5)

The ordinary Babai (integer) point $x^b \in \mathbb{Z}^n$ found by the Babai nearest plane algorithm [3] is defined as

$$c_n = \frac{\bar{y}_n}{r_{nn}}, \hspace{0.5cm} x^b_n = [c_n],$$

$$c_i = (\bar{y}_i - \sum_{j=i+1}^{n} r_{ij} x^j) / r_{ii}, \hspace{0.5cm} x^b_i = [c_i], \hspace{0.5cm} i = n-1, \ldots, 1.$$  \hspace{1cm} (6)

The ordinary Babai point $x^b \in \mathbb{Z}^n$ can be used as an estimator of the true parameter vector $\hat{x}$, and its success probability can be obtained by using the following formula (see [4]).

$$P(R) := \Pr(x^b = \hat{x}) = \prod_{i=1}^{n} \Phi(ri_i),$$  \hspace{1cm} (7)

$$\Phi(ri_i) = \sqrt{\frac{2}{\pi}} \exp(-\frac{1}{2} r^2)dt.$$  \hspace{1cm} (8)

With the QR factorization (4), the LLL reduction algorithm [1] reduces $R$ to $\hat{R}$:

$$\hat{Q}^TRZ = \hat{R},$$  \hspace{1cm} (9)

where $\hat{Q} \in \mathbb{R}^{n \times n}$ is an orthonormal matrix, $Z \in \mathbb{Z}^{n \times n}$ is a unimodular matrix (i.e., $\det(Z) = \pm 1$) and $\hat{R} \in \mathbb{R}^{n \times n}$ is an upper triangular matrix satisfying the following conditions:

$$|r_{ik}| \leq \frac{1}{2}|r_{ii}|, \hspace{0.5cm} i = 1, 2, \ldots, k-1$$  \hspace{1cm} (10)

$$\delta r_{k-1,k}^2 \leq r_{k-1,k}^2 + r_{kk}^2, \hspace{0.5cm} k = 2, 3, \ldots, n,$$  \hspace{1cm} (11)

where $\delta$ is a constant satisfying $1/4 < \delta \leq 1$. The matrix $\hat{R}$ is said to be $\delta$-LLL reduced or simply LLL reduced. Equations (10) and (11) are referred to as the size reduced condition and the Lovász condition, respectively.

Define $y = \hat{Q}^T \bar{y}$, $v = \hat{Q}^T \bar{v}$, $z = Z^{-1} \bar{x}$, then (5) can be transformed to

$$\bar{y} = R\bar{z} + \bar{v}, \hspace{0.5cm} \bar{v} \sim \mathcal{N}(0, \sigma^2 I).$$  \hspace{1cm} (12)

Using (6), we can obtain the Babai point $z^b$ corresponding to the model (12). Then we get the LLL-aided ordinary Babai point $\hat{x}^{bb}$, which can be used as an estimator of $\hat{x}$:

$$\hat{x}^{bb} = Zz^b.$$  \hspace{1cm} (13)

It is shown in [4] that each permutation of two consecutive columns in the LLL reduction process increases the success probability of the Babai point and any size reduction on the super-diagonal entry of $R$ which is followed by a permutation of the two corresponding columns also increases the success probability of the Babai point.

If $\hat{x} \in B$ (see (3)), after we obtain $x^{bb}$ in (13), we round it to the nearest point in $B$, leading to the box-constrained LLL-aided Babai point $x^{b\text{bb}}$:

$$x^{b\text{bb}}_i = \begin{cases} l_i, & \text{if } x^{bb}_i \leq l_i \\ x^{bb}_i, & \text{if } l_i < x^{bb}_i < u_i, i = n, n-1, \ldots, 1. \end{cases}$$  \hspace{1cm} (14)

B. FCLLL Reductions

In the original LLL algorithm [1] or its close variants, the reduction process starts with column 2 of $R$ and ends with column $n$ of $R$. When the reduction processes at column $k$, it first performs the size reduction on the element $r_{ik}$ for $i = k - 1 : -1 : 1$, and then checks whether the Lovász condition (11) is satisfied or not. If it is, the algorithm moves to column $k + 1$, i.e., the column index increases by 1; otherwise it permutes columns $k - 1$ and $k$ of $R$ and moves back to column $k - 1$, i.e., the column index decreases by 1. One does not know exactly how many iterations are required to finish the reduction process (here the number of iteration means the number of tests on the Lovász condition [9]). The FCLLL algorithm proposed in [8], to be referred to as fcLLL, is a modification of the LLL algorithm. It always goes from column 2 to column n and never comes back in the process. But it repeats the process for $J$ times, where $J$ is a fixed positive integer, resulting a number of $L = J(n-1)$ iterations. For the sake of readability, we describe the fcLLL reduction algorithm in Algorithm 1.

**Algorithm 1 fcLLL** [8]

1: compute the QR factorization (4), set $Z = I_n$
2: for $j = 1 : J$
3: set $k = 2$
4: while $k \leq n$
5: for $i = k - 1 : -1 : 1$
6: perform size reduction on $r_{i,k}$ and update $Z$
7: end for
8: if (11) does not hold then
9: permute columns $k - 1$ and $k$ of $R$ and triangularize $R$, and update $Z$ and $Q$
10: end if
11: $k = k + 1$
12: end while
13: end for

A modification of fcLLL was given in [9], which does only size reductions on the super-diagonal entries of $R$, because
size reductions on other off-diagonal entries have no effect on the Babai point. This results in a new algorithm, “effective” FCLLL, to be referred to as EfcLLL for convenience.

Like LLL, both fcLLL and EfcLLL start the iterations from the second column and finish at the last column of \( R \), i.e., \( k \) starts from 2 and ends at \( n \). Different from this traversal strategy, a so-called “enhanced” FCLLL algorithm was proposed in [10], to be referred to as EnfcLLL for convenience. Simulation results indicated that EnfcLLL can give much better performance than fcLLL and EfcLLL with similar computational cost. Due to the complexity of describing the algorithm and the limitation of space, we omit its details. Interesting readers are referred to [10].

III. AN EFFICIENT FIXED-COMPLEXITY LLL REDUCTION ALGORITHM

As explained in Section II, both fcLLL and EfcLLL start the iterations from the second column and finish at the last column, which may not be an effective way of increasing the success probability of the Babai point in a fixed time. The algorithm EnfcLLL, which uses a different traversal strategy, can improve the performance significantly. But its traversal strategy is not simple and is not easy to be understood. Also, like the previous ones, the traversal order is still fixed in advance. Let us use an extreme case to explain why a fixed order may not work well sometimes. Suppose we are allowed to do only one size reduction and column permutation. Then it is obvious that the order-fixed algorithms are unlikely to produce good results.

The idea of our approach is that at each step, we choose two consecutive columns which can get highest improvements of the success probability of the Babai point to do size reduction and column permutation. Thus our approach is a greedy approach. Generally speaking, the higher the success probability of the Babai point, the lower its BER.

Given an upper triangular matrix \( R \), suppose that for any specific \( k \), the Lovász condition (11) is not satisfied after \( r_{k-1,k} \) is size reduced, i.e., the following inequality holds:

\[
\delta r_{k-1,k-1}^2 > \left( r_{k-1,k} - \frac{r_{k-1,k}}{\bar{r}_{k-1,k-1}} \right)^2 + r_{kk}^2. \tag{15}
\]

After the size reduction, permutation of the two columns and triangularization, we obtain \( R \) which satisfies

\[
\bar{r}_{k-1,k} = r_{k-1,k} - \frac{r_{k-1,k}}{\bar{r}_{k-1,k-1}} \bar{r}_{k-1,k-1},
\]

\[
\bar{r}_{k-1,k-1} = \sqrt{r_{k-1,k-1}^2 + r_{kk}^2}, \tag{16}
\]

\[
|\bar{r}_{kk}| = |r_{k-1,k-1}/\bar{r}_{k-1,k-1}|.
\]

Note that the above operation decreases \(|r_{k-1,k-1}|\) and increases \(|r_{kk}|\). Then by (7)

\[
P(\bar{R}) = \phi(\bar{r}_{k-1,k-1})\phi(\bar{r}_{kk})/\phi(\bar{r}_{k-1,k-1})\phi(\bar{r}_{kk}) := T_k
\]

Instead we will look at other possible greedy strategies, which cost less computationally.

In the following, we propose two different greedy selection strategies to choose two columns of \( R \) to do reduction at each step.

The first greedy selection strategy is to find

\[
 j = \arg\max_k \{T_k^{(1)}, T_k^{(2)} = \frac{|r_{k-1,k-1}|}{|r_{kk}|}, \tag{15} \}
\]

If the above \( j \) does not exist, \( R \) is essentially LLL reduced as it can become fully LLL reduced after performing size reductions, which will not change the Babai point or the sphere decoding process for finding the optimal solution of the ILS problem [9] [11]. Otherwise, we perform size reduction on \( r_{j-1,j} \), permute columns \( j - 1 \) and \( j \) of \( R \), and triangularize \( R \) by a Givens rotation. After that, we update \( T_j^{(1)} \), \( T_{j-1}^{(1)} \) (if \( j > 1 \) and \( T_{j-1}^{(1)} \) (if \( j < n \)) (note that other \( T_j^{(1)} \)’s are not changed), and start next iteration. This greedy selection strategy is to find a pair of columns which can reduce the larger one of the two diagonal elements most significantly and it was first proposed in [12] for computing the full LLL reduction in solving integer least squares problem for GPS applications. Later the same strategy was proposed in [13] for the same purpose. One problem with this strategy is the ratio \(|r_{k-1,k-1}|/|r_{kk}| \) is invariant with respect to scaling of \( R(:,k-1:k) \). But the success probability of the Babai point will change by scaling.

The second greedy selection strategy is based on the following criterion:

\[
 j = \arg\max_k \{T_k^{(2)}, T_k^{(2)} = \frac{1}{|r_{kk}|} - \frac{1}{|r_{kk}|}, (15) \}
\]

Note that in the LLL reduction, after columns \( k - 1 \) and \( k \) are permuted, \(|r_{kk}|\) will increase, thus \( 1/|r_{kk}| \) will decrease. This strategy is to find columns \( k - 1 \) and \( k \), which leads to the biggest decrease in \( 1/|r_{kk}| \). We can rewrite \( T_k^{(2)} \) as

\[
T_k^{(2)} = \frac{|r_{k-1,k-1}| - |\bar{r}_{k-1,k-1}|}{|r_{k-1,k-1}|r_{kk}}.
\]

Note that the numerator is the gap between \(|r_{k-1,k-1}|\) and \(|\bar{r}_{k-1,k-1}|\) and the denominator is \( |\det(R(k-1:k,k-1:k))|\), which is defined in the lattice theory as the determinant of the lattice \( \{R(k-1:k,k-1:k)x : x \in \mathbb{Z}^2 \} \) and can be regarded as a weight of the gap.

For the sake of convenience, when (15) does not hold, we set \( T_k^{(i)} = 0 \) for \( i = 1,2 \).

In our algorithms, we suppose the maximum number of column permutations (to be denoted by \( N \)) is given. The description of our algorithm is given in Algorithm 2, where the final reduced upper triangular matrix is still denoted by \( R \).

Here we make some remarks about possible improvement of Algorithm 2. Like all previous FCLLL algorithms, Algorithm 2 forms \( Q \). But our goal is to get the LLL-aided Babai point and it is not necessary to form \( Q \). When we apply an orthogonal transformation to \( A \) (or \( R \)), we apply it to \( y \) (or \( \tilde{y} \)) at the same time. This will save the computational cost. To make \( R \) obtained by the QR factorization of \( A \) more like LLL reduced, we can incorporate the minimum
column pivoting strategy, which is equivalent to the minimum symmetric pivoting strategy used in [12] for computing the LLL reduction. This minimum column pivoting strategy is mathematically equivalent to the column pivoting strategy proposed in [14], but it can be implemented more efficiently and reliably (see [15]). Due to the limitation of space, we omit the details.

IV. SIMULATIONS

In this section, we give MATLAB simulation results to compare the efficiency and effectiveness of GfcLLL(i) with fcLLL, EfcLLL, EnfcLLL, and LLL. In the tests, we took $\delta = 1$. All of the tests were performed in MATLAB R2016a on iMac with processor 3.2 GHz Intel Core i5 and memory 16 GB 1867 MHz DDR3.

We used the MIMO model to do the tests. For a fixed dimension, a fixed type of QAM and a fixed $E_b/N_0$, we randomly generated 500 complex channel matrices whose entries independently and identically follow the standard complex normal distribution, and for each generated channel matrix, we randomly generated 200 complex signal vectors and 200 complex Gaussian noise vectors, resulting in 10000 instances of complex linear models. Each complex instance was then transformed to an instance of the real linear model (1).

To compare GfcLLL(i) with the other FLLL algorithms, ideally we would like to compare the BER of the box-constrained Babai points aided by these algorithms with more or less the same computational cost. However, the parameters used by different algorithms for termination are different, thus it is difficult to set the parameters so that these algorithms have similar computational cost.

What we did in our tests is basically to control the number of column permutations each algorithm performs. We first fix the number of sweep $J$ in fcLLL and EfcLLL. For any channel matrix, we can record the number of column permutations performed by fcLLL and EfcLLL, which is denoted by $K$ (note that $K \leq J(n-1)$). Then for the same channel matrix, we set the number of column permutations for EnfcLLL and GfcLLL(i) as $\lceil 0.35K \rceil$ and $\lceil 0.7K \rceil$, respectively. Our simulations indicate that the about choices usually make the CPU time taken by our GfcLLL(i) less than the CPU time taken by other algorithms.

Figures 1 and 2 show the average BER (over 10000 instances) versus $E_b/N_0 = 10 : 5 : 30$ for the $8 \times 8$ complex MIMO systems with 4-QAM for $J = 1$ and $J = 2$, respectively. Similarly, Figures 3 and 4 show the corresponding results for the $16 \times 16$ complex MIMO systems with 16-QAM for $J = 1$ and $J = 2$, respectively. To see how these lattice reductions improve the performance of the box-constrained Babai points, we also give the results obtained by using the QR factorization only in these figures.
In the above tests EnfcLLL performed about half number of permutations performed by GfclLLL(i). In our simulations we found that if we set the same number of permutations for both EnfcLLL and GfclLLL(i), the former still gives a worse performance than the latter in terms of the BER of the box-constrained Babai points (note that in this case, the CPU time used by EnfcLLL is much higher than that by GfclLLL(i)).

V. SUMMARY

We have proposed a greedy selection based approach for FCLLL reduction and presented two greedy selection strategies. Simulations showed that the two new FCLLL algorithms give Babai points with lower BER in shorter CPU time than existing FCLLL algorithms.

REFERENCES

[1] A. Lenstra, H. Lenstra, and L. Lovász, “Factoring polynomials with rational coefficients,” Mathematische Annalen, vol. 261, no. 4, pp. 515–534, 1982.
[2] D. Micciancio, “The hardness of the closest vector problem with preprocessing,” IEEE Transactions on Information Theory, vol. 47, no. 3, pp. 1212–1215, 2001.
[3] L. Babai, “On lovazs lattice reduction and the nearest lattice point problem,” Combinatorica, vol. 6, no. 1, pp. 1–13, 1986.
[4] X.-W. Chang, J. Wen, and X. Xie, “Effects of the LLL reduction on the success probability of the babai point and on the complexity of sphere decoding,” IEEE Transactions on Information Theory, vol. 59, no. 8, pp. 4915–4926, 2013.
[5] M. Taherzadeh and A. K. Khandani, “On the limitations of the naive lattice decoding,” IEEE Transactions on Information Theory, vol. 56, no. 10, pp. 4820–4826, 2010.
[6] Y. H. Gan and W. H. Mow, “Novel joint sorting and reduction technique for delay-constrained LLL-aided MIMO detection,” IEEE Signal Processing Letters, vol. 15, pp. 194–197, 2008.
[7] J. Jaldén, D. Seethaler, and G. Matz, “Worst-and average-case complexity of LLL lattice reduction in MIMO wireless systems,” in Proceedings of the 2008 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 2685–2688.
[8] H. Vetter, V. Ponnampalam, and P. A. Sandell, M.and Hoeher, “Fixed complexity LLL algorithm,” IEEE Transactions on Signal Processing, vol. 57, no. 4, pp. 1634–1637, 2009.
[9] C. Ling, W. Mow, and N. Howgrave-Graham, “Reduced and fixed-complexity variants of the LLL algorithm for communications,” IEEE Transactions on Communications, vol. 61, no. 3, pp. 1040–1050, 2013.
[10] Q. Wen, Q. Zhou, and X. Ma, “An enhanced fixed-complexity LLL algorithm for MIMO detection,” in the Proceedings of IEEE GLOBECOM 2014, pp. 3231–3236.
[11] X. Xie, X. Chang, and M. Al Borno, “Partial LLL reduction,” in the Proceedings of IEEE GLOBECOM 2011, 5 pages.
[12] X.-W. Chang, X. Yang, and T. Zhou, “MLAMBDA: A modified LAMBDA method for integer least-squares estimation,” Journal of Geodesy, vol. 79, no. 9, pp. 552–565, 2005.
[13] W. Zhang, S. Qiao, and Y. Wei, “A diagonal lattice reduction algorithm for mimo detection,” IEEE Signal Processing Letters, vol. 19, no. 5, pp. 311–314, 2012.
[14] D. Wübhen, R. Bohnke, J. Rinas, V. Kuhn, and K. Kammeyer, “Efficient algorithm for decoding layered space-time codes,” Electronics Letters, vol. 43, no. 22, pp. 1348–1350, 2001.
[15] X.-W. Chang, X. Xie, and T. Zhou, “MILES: MATLAB package for solving mixed integer least squares problems version 2.0, http://www.cs.mcgill.ca/~chang/software.php.”