Some Theory and Algorithms for Integer Least Squares Estimation

Jinming Wen

Doctor of Philosophy

Department of Mathematics and Statistics

McGill University

Montreal,Quebec

October 2014

A thesis submitted to McGill University
in partial fulfillment of the requirements for the degree of Doctor
of Philosophy in Mathematics and Statistics.

DEDICATION

To my beloved family, and especially to my parents Dongsheng Wen and Luoxiu Huang

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my thesis advisor Professor Xiao-Wen Chang for his numerous help, support and guidance on both academic and personnel level over the past several years. I can never forget the nights that we exchanged emails, discussed academic problems through the skype later than 1.00 am. The rigorous attitude and enthusiasm on research I learned from him is an invaluable fortune of my life. All I can say is that I can not thank him enough.

I gratefully thank Professor Wai Ho Mow who invited me to visit his group in Hong Kong University of Science and Technology, where part of this thesis was written. I would like to thank his constructive comments, financial supports and his group's hospitalities. I would also like to thank Professor Mohamed Oussama Damen for reading my draft and providing invaluable feedback on it.

I would like to thank my lovely wife Di Liu who gave me endless care. It would be much harder for me to do research without her help and supports. I would also like to thank my family for their love.

I would like to express my sincere thanks to Baojian, Zhou, Professors Ming Mei and Jian-Jun Xu, for their advice, help and friendships, my office mates Ben Smith and Takei Luiz, and my lab mates, Sevan Hanssian, Wen-Yang Ku, Wanru Lin and Xiaohu Xie for all the inspiring discussions and joyful time we shared together.

There are a number of friends who should be recognized as well. I am grateful to Jan Feys, Tao Lei, Dongfang Li, Alexandra Tcheng and Xiangwen Zhang for the fun times and friendly help.

ABSTRACT

Integer least squares (ILS) estimation problems arise from many applications. In this thesis, we investigate the effects of the celebrated LLL reduction and some well-known column permutation strategies on the success probability of the Babai estimators, the widely used suboptimal solutions in practical applications, for both ordinary ILS (OILS) and box-constrained ILS (BILS) estimation problems, and solve a shortest vector problem (SVP) arising in compute-and-forward protocol design. For the OILS estimation problems, we rigorously prove that the success probability P^{OB} of the ordinary Babai estimators can always be improved (not strictly) after the LLL algorithm or its permutation-only version LLL-P is applied and give examples to show that both the V-BLAST and SQRD may decrease P^{OB}. For the BILS estimation problems, on the one hand, we show that under a condition LLL-P always increases the success probability P^{BB} of the box-constrained Babai estimators and argue why both V-BLAST and SQRD often increase P^{BB} under the same condition; and on the other hand, we show that under an opposite condition LLL-P always decreases P^{BB} and argue why both V-BLAST and SQRD often decrease P^{BB} under the same condition. This surprising result shows that the corresponding conditions should be checked before applying these column permutation strategies. We also solve a conjecture related to P^{BB}. An efficient algorithm which is of polynomial time indicated by the simulation is proposed for solving the SVP arising in compute-and-forward protocol design. Simulations show that our algorithm is not only much more

efficient than the existing ones that give the optimal solution, but also faster than some of the suboptimal methods.

ABRÉGÉ

Les problèmes d'estimation des moindres carrés entiers (MCI) surgissent de plusieurs applications. Dans cette thèse, nous étudions les effets de la célèbre réduction de réseau LLL et quelques stratégies bien connues de permutation de colonnes sur la probabilité de succès des estimateurs de Babai, les solutions sous-optimales largement utilisées dans des applications pratiques, pour des problèmes d'estimation de MCI ordinaires (MCIO) et des MCI contraints par des boîtes (MCICB), ainsi qu'un problème du vecteur le plus court (PVC) qui survient dans la conception de protocole calcul-et-transfert. En ce qui concerne les problèmes d'estimation MCIO, nous prouvons rigoureusement que la probabilité de succès P^{OB} des estimateurs de Babai ordinaires peut toujours être améliorée (non strictement) après que l'algorithme LLL, ou sa version de permutation seule LLL-P, est appliqué et donnons des exemples pour montrer que les V-BLAST et SQRD peuvent diminuer P^{OB}. Quant aux problèmes d'estimation MCICB, nous montrons d'une part que, sous une contrainte, LLL-P augmente toujours la probabilité de succès P^{BB} des estimateurs de Babai contraints par des boîtes et affirmons pourquoi V-BLAST et SQRD souvent augmentent P^{BB} sous la même condition; d'autre part, nous montrons que sous une contrainte opposée, LLL-P diminue toujours P^{BB} et affirmons pourquoi V-BLAST et SQRD souvent diminuent P^{BB} sous la même contrainte. Ce résultat surprenant démontre que les contraintes correspondantes doivent être vérifiées avant d'appliquer V-BLAST et SQRD. Nous résolvons également une conjecture reliée à P^{BB}. Étant donné que le PVC survient dans la conception de protocole calcul-et-transfert, nous proposons

un algorithme polynomial pour le résoudre. Des simulations démontrent que notre algorithme est non seulement plus efficace que ceux qui existent présentement pour produire la solution optimale, mais il est aussi plus rapide que quelques méthodes qui produisent des solutions sous-optimales.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
Introduction

In this chapter, first we introduce the integer least squares problems, their applications and some numerical methods to solve them. Then, we briefly introduce the background of three different problems which will be investigated in this thesis. Finally, we summarize the contributions of this thesis and present the notation which will be used in this thesis.

1.1 Integer least squares estimation

Suppose that we have the following linear model:

$$\boldsymbol{y} = \boldsymbol{A}\hat{\boldsymbol{x}} + \boldsymbol{v}, \boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}), \tag{1.1}$$

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

A common method to estimate $\hat{\boldsymbol{x}}$ from (1.1) is to solve the following ordinary integer least squares (OILS) problem:

$$\min_{\boldsymbol{x}\in\mathbb{Z}^n} \| \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x} \|_2^2, \tag{1.2}$$

whose solution $\boldsymbol{x}^{\mathrm{OL}}$, to be referred to as the OILS estimator, is the maximum likelihood estimator of $\hat{\boldsymbol{x}}$.

In some applications, the integer parameter vector $\hat{\boldsymbol{x}}$ in (1.1) may be subject to some constraints, such as the box constraints, i.e., we have the following box-constrained linear model:

$$\boldsymbol{y} = \boldsymbol{A}\hat{\boldsymbol{x}} + \boldsymbol{v}, \quad \boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}), \tag{1.3a}$$

$$\hat{\boldsymbol{x}} \in \mathcal{B} \equiv \{\boldsymbol{x} \in \mathbb{Z}^n : \boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u},\ \boldsymbol{l}, \boldsymbol{u} \in \mathbb{Z}^n\}. \tag{1.3b}$$

Then instead of solving the OILS (1.2), one solves the following box-constrained integer least squares (BILS) problem:

$$\min_{\boldsymbol{x} \in \mathcal{B}} \| \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x} \|_2^2, \tag{1.4}$$

whose solution $\boldsymbol{x}^{\mathrm{BL}}$, to be referred to as the BILS estimator, is the maximum likelihood estimator of $\hat{\boldsymbol{x}} \in \mathcal{B}$.

The OILS problem (1.2) is also referred to as the closest vector problem (CVP) in information theory (see, e.g., [36, 62]) as it is equivalent to find a point in the lattice $\{\boldsymbol{A}\boldsymbol{x} : \boldsymbol{x} \in \mathbb{Z}^n\}$ which is closest to $\boldsymbol{y}$. If $\boldsymbol{y} = \boldsymbol{0}$, then (1.2), where $\boldsymbol{x} \neq \boldsymbol{0}$, degenerates to the shortest vector problem (SVP).

ILS problems (OILS and BILS) arise from many applications, such as, combinatorial optimization (see, e.g., [23]), GPS (see, e.g., [82, 37]), cryptography (see, e.g., [61, 35, 36]), communications (see, e.g., [65, 2, 21]), number theory (see, e.g., [28, 33]), lattice design (see, e.g., [1]), Monte Carlo second moment estimation (see, e.g., [19]), noncryptographic random number generation [46, pp. 89–113] and transportation science (see, e.g., [48]), etc.

Van Emde Boas [86] showed that the general CVP is an NP-hard problem and Micciancio [59] gave a simpler proof. Despite the similarities between CVP and SVP, the analysis for the complexity of SVP has progressed much slower, and whether the SVP is also NP-hard is still an open problem. The first breakthrough result is due to Ajtai [3] who proved that solving the problem exactly is NP-hard for randomized reductions. Later, Micciancio [60] showed that the SVP is hard to approximate within any factor less than $\sqrt{2}$ under reverse unfaithful random reductions. For more details, see e.g., [45].

1.2 Numerical methods

In communications, the commonly used approach to solving the OILS problem (1.2) or the BILS problem (1.4) is called sphere decoding, which usually has two stages: reduction and search. The most widely used reduction strategy for the OILS problem (1.2) is the LLL reduction, invented by Arjen Lenstra, Hendrik Lenstra and László Lovász in 1982 [51], which consists of size reductions and column permutations. But it is difficult to use it to solve the BILS problem (1.4) because in general after size reductions which are involved in the LLL reduction, the box constraint would become too complicated to handle in the search process, although very recently it was shown in [97] that some size reductions could be performed on some columns of $\boldsymbol{A}$. However, one can use its permutation strategy, which was referred to as LLL-P in [90] (in [18] we referred to it as LLL-permute). Two well-known reduction strategies for the BILS problem (1.4) are V-BLAST [25, 21] and SQRD [95, 16]. The LLL-P, SQRD and V-BLAST strategies use only the information of $\boldsymbol{A}$ to do the column permutations. Some column reordering strategies which use not

only the information of $\boldsymbol{A}$, but also the information of $\boldsymbol{y}$ and the box constraint have also been proposed, see, e.g., [80], [16] and [15].

The commonly used search strategy for solving an OILS problem (1.2) or a BILS problem (1.4) is the Schnorr-Euchner search strategy [74], a variation of the Fincke-Pohst search strategy [24], or its variants, see, e.g., [2, 21, 16], which enumerate the lattice points in a hypersphere (this is the reason why the discrete enumeration algorithms following the Fincke-Pohst search strategy or its variants are referred to as the sphere decoding algorithms in communications), or equivalently, integer points in a hyper-ellipsoid. Probably the earliest search strategy is Kannan's approach [43, 44], which enumerates the lattice points in a parallelotope. The complexity analysis of Kannan's approach can be found in [35]. In general, Fincke-Pohst search strategy is more often used in practice and Kannan's approach serves more as a theoretical tool since Kannan's approach has better theoretical results on the complexity and simulations in [68] showed that, in general, Fincke-Pohst search strategy is usually faster.

There are some other approaches to solving (1.2), such as, the Monte Carlo probabilistic algorithms, see, e.g., [4, 5, 12, 36], which find lattice vectors that are no more than $1+\epsilon$ times further away from the target than the optimal solution, for an arbitrary $\epsilon > 0$. Their best known upper bounds on the time and space complexities are higher than that of the Voronoi cell based approach [79, 62], which is based on computing the Voronoi cell of the lattice and can solve (1.2) in time $\mathcal{O}(2^{2n+\mathcal{O}(n)})$ and space $\mathcal{O}(2^{n+\mathcal{O}(n)})$. For more details, see the survey paper [36] and references therein. Very recently, a real relaxation based branch and bound approach has been proposed

in [7], which may be more efficient than the sphere decoding approach for some types of problem data. We will only focus on the Schnorr-Euchner search strategy and its variants since the Schnorr-Euchner search is usually the fastest one.

1.3 Babai estimators

The OILS problem (1.2) is NP-hard [86, 59] and solving (1.4) may become time-prohibitive when σ is large, the dimension of $\boldsymbol{A}$ is large, or $\boldsymbol{A}$ is ill conditioned [41]. Therefore, for some real-time applications, an approximate solution, which can be produced quickly, is computed instead. For the OILS problem, the Babai integer point $\boldsymbol{x}^{\mathrm{OB}}$, to be referred to as the ordinary Babai estimator, which can be obtained by the Babai nearest plane algorithm [9], is an often used approximate solution. Taking the box constraint into account, one can easily modify the Babai nearest plane algorithm to get an approximate solution $\boldsymbol{x}^{\mathrm{BB}}$ to the BILS problem (1.4), which will be referred to as the box-constrained Babai estimator. The Babai points $\boldsymbol{x}^{\mathrm{OB}}$ and $\boldsymbol{x}^{\mathrm{BB}}$ are also the first integer points found by the Schnorr-Euchner algorithm for the OILS problem (1.2) ([2]) and BILS problem (1.4) ([21, 16]), respectively. In communications, algorithms for finding the Babai estimators are often referred to as successive interference cancelation detectors.

There are many other suboptimal algorithms for the BILS problems. Probability or statistic based pruning strategies have been used in, e.g., [30, 78, 94, 20], to reduce the complexity of search. A so-called sphere-projection algorithm which is able to achieve near-maximal likelihood performance and significantly increase the diversity gains has been proposed in [8]. K-best decoding algorithm which only keeps the K-best candidates at each level of the sublattice in the breadth-first search step

has been proposed in [32] to approach near-maximum-likelihood performance for MIMO detection. There are some variants of K-best algorithms too, see, e.g., [34]. Some algorithms based on semidefinite programme techniques have been proposed, see, e.g., [57, 88]. The fixed-complexity sphere decoder (FCD) which performs a fixed number of operations during the detection process was developed in [11]. The error probability of FCD has been analyzed in [40] and many variants of FCD have also been proposed, see, e.g., [50]. In this thesis, we will only focus on the Babai estimators as their cost are lowest among these algorithms and they are commonly used in practice.

1.4 Success probabilities of Babai estimators

In order to verify whether an estimator is good enough for a practical use, one finds the probability of the estimator being equal to the true integer parameter vector, which is referred to as success probability [37]. The probability of wrong estimation is referred to as error probability, see, e.g., [21, 40].

Let P^{OB} and P^{OL} respectively denote the success probabilities of the ordinary Babai estimator $\boldsymbol{x}^{\text{OB}}$ and the OILS estimator $\boldsymbol{x}^{\text{OL}}$ of $\hat{\boldsymbol{x}}$ in (1.1). Similarly, the success probabilities of the box-constrained Babai estimator $\boldsymbol{x}^{\text{BB}}$ and the BILS estimator $\boldsymbol{x}^{\text{BL}}$ of $\hat{\boldsymbol{x}}$ in (1.3) are respectively denoted by P^{BB} and P^{BL}.

The success probability P^{OB} of the ordinary Babai estimator $\boldsymbol{x}^{\text{OB}}$ is very useful due to a few reasons. If one just uses $\boldsymbol{x}^{\text{OB}}$ as an estimator of the true parameter vector $\hat{\boldsymbol{x}}$ in (1.1) as often done in communications, certainly one is interested in its success probability. Even if one computes the optimal solution $\boldsymbol{x}^{\text{OL}}$ of the OILS problem (1.2), P^{OB} is still useful, as computing the success probability P^{OL} of the

OILS estimator $\boldsymbol{x}^{\mathrm{OL}}$ is time-consuming and one often uses P^{OB}, a lower bound on P^{OL} [84], to approximate P^{OL}. Furthermore, if P^{OB} is close to 1, one does not need to spend extra computational time to find $\boldsymbol{x}^{\mathrm{OL}}$.

It is well-known that after the LLL reduction, P^{OB} usually increases, but there has been no rigorous theoretical justification and it is not known whether the LLL reduction can decrease P^{OB}. We will solve this interesting and important problem in Chapter 3.

Similarly, for the BILS problem (1.4), it is also very important to compute P^{BB} if $\boldsymbol{x}^{\mathrm{BB}}$ is used to estimate the integer parameter vector $\hat{\boldsymbol{x}}$ in (1.3). Although the formula for P^{OB} was given in [83]. There has been no formula for computing P^{BB} in the literature and whether the LLL-P, V-BLAST and SQRD can always improve P^{BB} is still not clear from the literature. We will study this interesting and important problem in Chapter 4.

In [58], the authors made a conjecture, based on which a stopping criterion for the search process was proposed to reduce the computational cost of solving the BILS problem (1.4). The conjecture is related to the success probability P^{OB} of the ordinary Babai estimator $\boldsymbol{x}^{\mathrm{OB}}$. We will solve the conjecture in Chapter 4.

1.5 An SVP in compute-and-forward protocol design

In relay networks, compute-and-forward (CF) [67] is a promising relaying strategy that can offer higher achievable rates than traditional ones (e.g., amplify-and-forward, decode-and-forward), especially at the moderate signal-to-noise ratio (SNR) regime. The main difficulty in the computer-and-forward scheme is to find the optimal coefficient vector that maximizes the computation rate, which is to solve the

following optimization problem [67, 89, 73, 101, 72, 102] to get the optimal coefficient vector $\boldsymbol{a}^\star$:

$$\boldsymbol{a}^\star = \arg \min_{\boldsymbol{a} \in \mathbb{Z}^n \setminus \{\boldsymbol{0}\}} \boldsymbol{a}^T \boldsymbol{G} \boldsymbol{a}, \text{ where } \boldsymbol{G} = \boldsymbol{I} - \frac{P}{1 + P \parallel \boldsymbol{h} \parallel_2^2} \boldsymbol{h}\boldsymbol{h}^T, \tag{1.5}$$

where P is the transmission power and $\boldsymbol{h} \in \mathbb{R}^n$ is the channel vector.

Various methods have been proposed to solve (1.5). But the complexity of the existing optimal methods are high and the existing suboptimal algorithms may fail to offer satisfactory performance. We will propose an efficient algorithm to solve (1.5) in Chapter 5.

1.6 Organization and contributions

The rest part of this thesis is organized as follows. In Chapter 2, we introduce various reduction algorithms to transform the OILS problem (1.2) and the BILS problem (1.4) to simpler problems, which can then be solved by search algorithms.

In Chapter 3, we investigate the effects of some typical reductions on the success probability P^{OB} of the ordinary Babai estimator $\boldsymbol{x}^{\text{OB}}$ for the ordinary linear model (1.1). In particular, we rigorously prove that the LLL reduction can always improve P^{OB}. We first show that P^{OB} as a lower bound on the success probability P^{OL} of the OILS estimator $\boldsymbol{x}^{\text{OL}}$ is sharper than the lower bound given in [37]. Then, we show that any column permutation and any size reduction on the super diagonal entries of the R-factor $\boldsymbol{R}$ of the QR factorization of $\boldsymbol{A}$ in (1.1) that is immediately followed by a column permutation in the LLL reduction algorithm [51] increase (not strictly) P^{OB} and all other size reductions have no effect on P^{OB}. We also investigate how the parameter δ in the LLL reduction affects P^{OB}. After that, we give examples

to show that unlike the LLL and LLL-P, the permutation strategies SQRD and V-BLAST may decrease P^{OB}. Finally, we give some upper bounds on P^{OB} after the LLL reduction algorithm is applied by using the entries of the R-factor $\boldsymbol{R}$ of the QR factorization of $\boldsymbol{A}$ in (1.1). These contributions were published in [18], which also includes other results that are not covered by this thesis.

In Chapter 4, we investigate the effects of some column permutation strategies on the success probability P^{BB} of the box-constrained Babai estimator $\boldsymbol{x}^{\text{BB}}$ for the box-constrained linear model (1.3) and solve a conjecture proposed in [58]. In particular, we show that under some conditions, all of the LLL-P, V-BLAST and SQRD may decrease P^{BB}. This surprising result indicates that we need to check whether the corresponding conditions hold before applying them. First, we derive a formula for the success probability P^{BB}. Then, we investigate the effects of the LLL-P, SQRD and V-BLAST on P^{BB}. On the one hand, we show that the LLL-P always increases P^{BB} and argue why both V-BLAST and SQRD often increase P^{BB} under a condition; and on the other hand, we show that the LLL-P always decreases P^{BB} and argue why both V-BLAST and SQRD often decrease P^{BB} under an opposite condition. After this, we derive a column permutation invariant bound on P^{BB}, which is an upper bound and a lower bound under the two opposite conditions, respectively. We also present some numerical results to illustrate our findings. Finally, we consider a conjecture concerning the success probability of the ordinary Babai integer point proposed in [58]. We first construct an example to show that the conjecture does not always hold, and then propose some conditions to guarantee it holds. These contributions were contained in [90].

In Chapter 5, we propose an efficient algorithm for finding the optimal coefficient vector that maximizes the computation rate at a relay in the computer-and-forward scheme, i.e., solving (1.5). First, we derive an algorithm with only $\mathcal{O}(n)$ flops to compute the Cholesky factorization of $\boldsymbol{G}$ in (1.5) (we do not form the whole Cholesky factor explicitly), to transform (1.5) to a standard SVP. Then, we propose some conditions that can be checked by $\mathcal{O}(n)$ flops, under which the optimal coefficient vector $\boldsymbol{a}^\star$ can be obtained immediately without using any search algorithm. After that, by taking into account some resultant useful properties of $\boldsymbol{a}^\star$, we modify the Schnorr-Euchner search algorithm to solve the SVP. Simulation results show that the average cost of our new algorithm is $\mathcal{O}(n^{1.5})$ flops for i.i.d. Gaussian channel entries, and our algorithm is not only much more efficient than the existing ones that give the optimal solution, but also faster than some of the suboptimal methods. These contributions were contained in [92].

Finally, we summarize theconclusions and discuss the directions for future research in Chapter 6.

1.7 Notation

Let $\mathbb{R}^n$ and $\mathbb{Z}^n$ be the spaces of $n-$dimensional column real and integer vectors, respectively. Let $\mathbb{R}^{m\times n}$ and $\mathbb{Z}^{m\times n}$ respectively be the spaces of $m\times n$ real and integer matrices. Boldface lowercase letters denote column vectors and boldface uppercase letters denote matrices, e.g., $\boldsymbol{x}\in\mathbb{R}^n$ and $\boldsymbol{A}\in\mathbb{R}^{m\times n}$. For $\boldsymbol{x}\in\mathbb{R}^n$, we use $\lfloor\boldsymbol{x}\rceil$ to denote its nearest integer vector, i.e., each entry of $\boldsymbol{x}$ is rounded to its nearest integer (if there is a tie, the one with smaller magnitude is chosen). Let $\boldsymbol{x}^T$, $\boldsymbol{A}^T$ respectively denote the transpose of $\boldsymbol{x}$ and $\boldsymbol{A}$. For a vector $\boldsymbol{x}$, let x_i be the element with index i

and $\boldsymbol{x}_{i:j}$ be the vector composed of elements with indices from i to j. For a matrix $\boldsymbol{A}$, let a_{ij} be the element at row i and column j, $\boldsymbol{A}_{i:j,k:\ell}$ be the submatrix containing elements with row indices from i to j and column indices from k to ℓ, and $\boldsymbol{A}_{i:j,k}$ be the vector containing elements with row indices from i to j and column index k. Let $\mathbf{0}^n$ and $\mathbf{0}^{m\times n}$ respectively denote the $n-$dimensional zero column vector and $m \times n$ zero matrix, and let $\boldsymbol{e}_k^n$ and $\mathbf{1}^n$ denote the k-th column of an $n \times n$ identity matrix $\boldsymbol{I}$ and an $n-$dimensional vector with all of it entries being 1, respectively (sometimes the superscripts are omitted if the dimensions are obvious).

CHAPTER 2
Reductions of Integer Least Squares Problems

In this chapter, we first introduce the QR, QRZ and QRP reductions, each of which reduces a full column rank matrix to an upper triangular matrix. All of these reductions can be used to transform the OILS problem (1.2) to a new OILS problem. The QR and QRP reductions can also be used to transform the BILS problem (1.4) to a new BILS problem. The transformed OILS problem and BILS problem can then be solved by a search process, typically by sphere decoding methods in applications.

Both the QRZ and QRP reductions are frameworks. The QRZ reduction is actually the lattice reduction and the most often used one is the LLL reduction [51]. The QRP reduction involves column reordering strategies and two well-known QRP reductions are V-BLAST [25] and SQRD [95].

We also introduce the Babai estimators, the suboptimal solutions to the OILS and BILS problems, which are often used in practice.

The contents of this chapter will be used later.

2.1 QR reduction and Babai estimators

Assume that $\boldsymbol{A}$ in the linear model (1.1) or in the box constrained linear model (1.3) has the QR factorization

$$\boldsymbol{A} = [\boldsymbol{Q}_1, \boldsymbol{Q}_2] \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{bmatrix}, \tag{2.1}$$

where $[\underset{n}{\boldsymbol{Q}_1}, \underset{m-n}{\boldsymbol{Q}_2}] \in \mathbb{R}^{m\times m}$ is orthogonal and $\boldsymbol{R} \in \mathbb{R}^{n\times n}$ is upper triangular. There are several methods to get (2.1), for more details, see, e.g., [29]. Without loss of generality, we assume that the diagonal entries of $\boldsymbol{R}$ are positive throughout the thesis.

Define

$$\tilde{\boldsymbol{y}} = \boldsymbol{Q}_1^T \boldsymbol{y}, \quad \tilde{\boldsymbol{v}} = \boldsymbol{Q}_1^T \boldsymbol{v}. \tag{2.2}$$

Then from (1.1), we have

$$\tilde{\boldsymbol{y}} = \boldsymbol{R}\hat{\boldsymbol{x}} + \tilde{\boldsymbol{v}}, \quad \tilde{\boldsymbol{v}} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}). \tag{2.3}$$

And the OILS problem (1.2) is reduced to

$$\min_{\boldsymbol{x}\in\mathbb{Z}^n} \|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{x}\|_2^2. \tag{2.4}$$

Similarly, (1.3) and (1.4) can be respectively reduced to:

$$\tilde{\boldsymbol{y}} = \boldsymbol{R}\hat{\boldsymbol{x}} + \tilde{\boldsymbol{v}}, \quad \tilde{\boldsymbol{v}} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}), \tag{2.5a}$$

$$\hat{\boldsymbol{x}} \in \mathcal{B} \equiv \{\boldsymbol{x} \in \mathbb{Z}^n : \boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u},\ \boldsymbol{l}, \boldsymbol{u} \in \mathbb{Z}^n\}, \tag{2.5b}$$

and

$$\min_{\boldsymbol{x}\in\mathcal{B}} \|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{x}\|_2^2. \tag{2.6}$$

One can then apply a discrete search algorithm such as the Schnorr-Euchner search algorithm [74] and its extension [14, 21, 16] to solve (2.4) and (2.6), respectively.

In some applications, the Babai estimators, which are suboptimal solutions and can be computed quickly, are usually computed instead of solving (2.4) or (2.6). The Babai estimator $\boldsymbol{x}^{\mathrm{OB}}$ of OILS (2.4) found by the Babai nearest plane algorithm [9] is defined as follows:

$$c_i^{\mathrm{OB}} = (\tilde{y}_i - \sum_{j=i+1}^{n} r_{ij} x_j^{\mathrm{OB}})/r_{ii}, \quad x_i^{\mathrm{OB}} = \lfloor c_i^{\mathrm{OB}} \rceil \tag{2.7}$$

for $i = n, n-1, \ldots, 1$, where $\sum_{n+1}^{n} = 0$.

Similarly, by taking the box into consideration, the box-constrained Babai estimator $\boldsymbol{x}^{\mathrm{BB}}$ of BILS (2.6) is defined as follows:

$$c_i^{\mathrm{BB}} = (\tilde{y}_i - \sum_{j=i+1}^{n} r_{ij} x_j^{\mathrm{BB}})/r_{ii}, \quad x_i^{\mathrm{BB}} = \begin{cases} l_i, & \text{if } \lfloor c_i^{\mathrm{BB}} \rceil \leq l_i \\ \lfloor c_i^{\mathrm{BB}} \rceil, & \text{if } l_i < \lfloor c_i^{\mathrm{BB}} \rceil < u_i \\ u_i, & \text{if } \lfloor c_i^{\mathrm{BB}} \rceil \geq u_i \end{cases} \tag{2.8}$$

for $i = n, n-1, \ldots, 1$, where $\sum_{n+1}^{n} = 0$.

For search efficiency and improving the success probability of the Babai estimators, one typically adopts some lattice reduction for the OILS (2.4) and some column reordering strategies for the BILS (2.6) to obtain better $\boldsymbol{R}'s$. See the following sections for more details.

2.2 QRZ reduction

For any full column rank matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, the lattice generated by $\boldsymbol{A}$ is defined by:

$$\mathcal{L}(\boldsymbol{A}) = \{\boldsymbol{A}\boldsymbol{z} | \boldsymbol{z} \in \mathbb{Z}^n\}. \tag{2.9}$$

The columns of $\boldsymbol{A}$ form a basis of $\mathcal{L}(\boldsymbol{A})$. For any $n \geq 2$, $\mathcal{L}(\boldsymbol{A})$ has infinity many bases and any of two are connected by a unimodular matrix, which is defined as follows:

Definition 2.2.1 *A matrix $\boldsymbol{Z} \in \mathbb{Z}^{n\times n}$ is said to be unimodular if its determinant is 1 or -1.*

For any given lattice basis matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{AZ}$ is also a basis matrix of $\mathcal{L}(\boldsymbol{A})$ if and only if $\boldsymbol{Z} \in \mathbb{Z}^{n\times n}$ is unimodular [2]. The process of selecting a good basis for a given lattice, given some criterion, is called lattice reduction and in many applications, it is advantageous if the basis vectors are short and nearly orthogonal to each other [2].

For more than a century, lattice reductions have been investigated by many people and several types of reduction have been proposed, which includes the HKZ reduction [47, 39], the Minowski reduction [64], the LLL reduction [51] and Seysen's reduction [77], etc. Lattice reduction plays an important role in many research areas, for more details, see the survey paper [96] and references therein.

Often a lattice reduction can be described as a QRZ reduction:

$$\boldsymbol{Q}^T\boldsymbol{AZ} = \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{bmatrix}, \tag{2.10}$$

where $\boldsymbol{R}$ satisfies the conditions of the corresponding reductions.

We introduce two commonly used QRZ reductions: the LLL reduction [51] and the HKZ reduction [47, 39] in the following two subsections.

2.2.1 LLL reduction

After the QR factorization (2.1) of $\boldsymbol{A}$, the LLL reduction [51] reduces the matrix $\boldsymbol{R}$ in (2.1) to $\bar{\boldsymbol{R}}$:

$$\bar{\boldsymbol{Q}}^T \boldsymbol{R} \boldsymbol{Z} = \bar{\boldsymbol{R}}, \tag{2.11}$$

where $\bar{\boldsymbol{Q}} \in \mathbb{R}^{n\times n}$ is an orthogonal matrix, $\boldsymbol{Z} \in \mathbb{Z}^{n\times n}$ is a unimodular matrix and $\bar{\boldsymbol{R}} \in \mathbb{R}^{n\times n}$ is an upper triangular matrix satisfies the following conditions:

$$|\bar{r}_{ik}| \leq \frac{1}{2}\bar{r}_{ii}, \quad i = 1, 2, \ldots, k-1, \tag{2.12}$$

$$\delta \bar{r}_{k-1,k-1}^2 \leq \bar{r}_{k-1,k}^2 + \bar{r}_{kk}^2, \quad k = 2, 3, \ldots, n, \tag{2.13}$$

where δ is a constant satisfying $1/4 < \delta \leq 1$. The matrix $\bar{\boldsymbol{R}}$ is said to be δ-LLL reduced or simply LLL reduced. Equations (2.12) and (2.13) are respectively referred to as the size-reduced condition and Lovász condition.

Notice that combining (2.1) and (2.11), the LLL reduction result in the following QRZ factorization (see (2.10)) of $\boldsymbol{A}$:

$$\tilde{\boldsymbol{Q}}^T \boldsymbol{A} \boldsymbol{Z} = \begin{bmatrix} \bar{\boldsymbol{R}} \\ \boldsymbol{0} \end{bmatrix}, \quad \tilde{\boldsymbol{Q}} \equiv \boldsymbol{Q} \begin{bmatrix} \bar{\boldsymbol{Q}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{m-n} \end{bmatrix}.$$

The matrix language can be used to describe the original LLL algorithm [51]. Two types of basic unimodular matrices, i.e., the integer Gauss transformations (IGT) and the permutation matrices, are implicitly used to update $\boldsymbol{R}$ so that it satisfies both (2.12) and (2.13).

To meet the first condition in (2.12), we can apply an IGT, which has the following form:

$$\boldsymbol{Z}_{ik} = \boldsymbol{I} - \zeta \boldsymbol{e}_i \boldsymbol{e}_k^T.$$

Applying $\boldsymbol{Z}_{ik}$ $(i < k)$ to $\boldsymbol{R}$ from the right gives

$$\bar{\boldsymbol{R}} = \boldsymbol{R}\boldsymbol{Z}_{ik} = \boldsymbol{R} - \zeta \boldsymbol{R}\boldsymbol{e}_i \boldsymbol{e}_k^T.$$

Thus $\bar{\boldsymbol{R}}$ is the same as $\boldsymbol{R}$, except that $\bar{r}_{jk} = r_{jk} - \zeta r_{ji}$ for $j = 1, \ldots, i$. By setting $\zeta = \lfloor r_{ik}/r_{ii} \rceil$, we ensure $|\bar{r}_{ik}| \leq \bar{r}_{ii}/2$. This process is called size reduction.

To meet the second condition in (2.13), permutations are needed in the reduction process. Suppose that $\delta\, r_{k-1,k-1}^2 > r_{k-1,k}^2 + r_{k,k}^2$ for some k. Then we interchange columns $k-1$ and k of $\boldsymbol{R}$. After the permutation the upper triangular structure of $\boldsymbol{R}$ is no longer maintained. But we can bring $\boldsymbol{R}$ back to an upper triangular matrix by using the Gram-Schmidt orthogonalization technique (see [51]) or by a Givens rotation:

$$\bar{\boldsymbol{R}} = \boldsymbol{G}_{k-1,k}^T \boldsymbol{R}\boldsymbol{P}_{k-1,k}, \tag{2.14}$$

where $\boldsymbol{G}_{k-1,k}$ is an orthonormal matrix and $\boldsymbol{P}_{k-1,k}$ is a permutation matrix, and

$$\bar{r}_{k-1,k-1}^2 = r_{k-1,k}^2 + r_{k,k}^2, \quad \bar{r}_{k-1,k}^2 + \bar{r}_{k,k}^2 = r_{k-1,k-1}^2, \quad \bar{r}_{k-1,k-1}\bar{r}_{kk} = r_{k-1,k-1} r_{kk}. \tag{2.15}$$

Note that the above operations guarantee $\delta\, \bar{r}_{k-1,k-1}^2 < \bar{r}_{k-1,k}^2 + \bar{r}_{k,k}^2$ since $\delta \leq 1$.

The LLL reduction algorithm is described in Algorithm 2.2.1, where the final reduced upper triangular matrix is still denoted by $\boldsymbol{R}$.

It is well known that the cost of the LLL algorithm is a polynomial of the dimension n for any lattice with integer basis [51], [6]. Although the average complexity

Algorithm 2.2.1 LLL reduction

compute the QR factorization: $\boldsymbol{A} = \boldsymbol{Q}\begin{bmatrix}\boldsymbol{R}\\ \boldsymbol{0}\end{bmatrix}$;
set $\boldsymbol{Z} = \boldsymbol{I}_n$, $k = 2$;
while $k \leq n$ **do**
  apply IGT $\boldsymbol{Z}_{k-1,k}$ to reduce $r_{k-1,k}$:
    $\boldsymbol{R} = \boldsymbol{R}\boldsymbol{Z}_{k-1,k}$;
  update $\boldsymbol{Z}$: $\boldsymbol{Z} = \boldsymbol{Z}\boldsymbol{Z}_{k-1,k}$;
  if $\delta\, r_{k-1,k-1}^2 > r_{k-1,k}^2 + r_{kk}^2$ **then**
    permute and triangularize $\boldsymbol{R}$:
      $\boldsymbol{R}{=}\boldsymbol{G}_{k-1,k}^T\boldsymbol{R}\boldsymbol{P}_{k-1,k}$;
    update $\boldsymbol{Z}$: $\boldsymbol{Z} = \boldsymbol{Z}\boldsymbol{P}_{k-1,k}$;
    $k = k - 1$, when $k > 2$;
  else
    for $i = k-2, \ldots, 1$ **do**
      apply IGT $\boldsymbol{Z}_{ik}$ to reduce r_{ik}: $\boldsymbol{R} = \boldsymbol{R}\boldsymbol{Z}_{ik}$;
      update $\boldsymbol{Z}$: $\boldsymbol{Z} = \boldsymbol{Z}\boldsymbol{Z}_{ik}$;
    end for
    $k = k + 1$;
  end if
end while

of the LLL algorithm is polynomial if the basis vectors are independently uniformly distributed inside the unit ball of $\mathbb{R}^n$ [22] or the entries of $\boldsymbol{A}$ independently follow the normal distribution $\mathcal{N}(0,1)$ [53, 42, 55], it was proved in [42] that in the MIMO context, the worst-case complexity is not even finite.

The LLL reduction is a powerful preprocessing tool that reduces the cost of searching the OILS estimator for (2.4) [37, 2]. And it has many variants, see, e.g., [75, 76]. To reduce the cost of the arithmetic operations, the so-called effective LLL and partial LLL have respectively been proposed in [53] and [98]. To reduce the cost of the bit operations, a LLL with quadratic complexity and quasi-linear complexity have been proposed in [69] and [70], respectively. For reducing the complex lattices,

a complex LLL algorithm which reduces the computational cost roughly by half compared to the traditional LLL has been proposed in [27].

2.2.2 HKZ reduction

A lattice basis matrix $\boldsymbol{A}$ is called HKZ reduced if its QR factor $\boldsymbol{R}$ in (2.10) satisfies (2.12), and for each $1 \leq i \leq n$, r_{ii} is the 2-norm of a shortest vector in $\mathcal{L}(\boldsymbol{R}_{i:n,i:n})$. An efficient HKZ reduction algorithm can be found in [100] and the properties of the HKZ reduced basis can be found in [49].

HKZ reduction is more powerful than the LLL reduction in the sentence that of the columns the resulting matrix $\boldsymbol{R}$ are shorter and closer to orthogonal, but it has high complexity since $n-1$ SVP's need to be solved. Therefore, it is usually used to solve a sequence of OILS problems (1.2) where $\boldsymbol{A}$ keeps the same for these problems. In this thesis, we only focus on the LLL reduction since we only consider its applications in solving the OILS problems with different $\boldsymbol{A}$ and $\boldsymbol{y}$.

2.2.3 Reduced OILS

Let

$$\bar{\boldsymbol{y}} = \bar{\boldsymbol{Q}}^T\tilde{\boldsymbol{y}},\ \ \bar{\boldsymbol{v}} = \bar{\boldsymbol{Q}}^T\tilde{\boldsymbol{v}},\ \ \boldsymbol{z} = \boldsymbol{Z}^{-1}\boldsymbol{x},\ \ \hat{\boldsymbol{z}} = \boldsymbol{Z}^{-1}\hat{\boldsymbol{x}}. \tag{2.16}$$

Then by (2.11), the linear model (2.3) and the OILS problem (2.4) can be respectively transformed to

$$\bar{\boldsymbol{y}} = \bar{\boldsymbol{R}}\hat{\boldsymbol{z}} + \bar{\boldsymbol{v}},\ \ \ \bar{\boldsymbol{v}} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}), \tag{2.17}$$

and

$$\min_{\boldsymbol{z}\in\mathbb{Z}^n} \|\bar{\boldsymbol{y}} - \bar{\boldsymbol{R}}\boldsymbol{z}\|_2^2. \tag{2.18}$$

The solution $\boldsymbol{z}^{\text{OL}}$ of (2.18) is the OILS estimator of $\hat{\boldsymbol{z}}$. One can also define the ordinary Babai point $\boldsymbol{z}^{\text{OB}}$ for (2.18) (see (2.4)).

2.3 QRP reduction

It is difficult to use a QRZ reduction to reduce the BILS problem (1.4) because in general after size reductions, the box constraint would become too complicated to handle in the search process, although very recently it was showed in [97] that some size reductions could be performed on some columns of $\boldsymbol{A}$ under some conditions. Therefore, one uses a QRP reduction to reduce the BILS problem (1.4). The QRP reduction has the following form:

$$\boldsymbol{Q}^T\boldsymbol{A}\boldsymbol{P} = \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{bmatrix}, \tag{2.19}$$

where $\boldsymbol{P}$ is a permutation matrix and $\boldsymbol{R}$ satisfies the conditions of the corresponding reductions.

We introduce three types of QRP reductions: LLL-P, V-BLAST and SQRD in the following three subsections.

2.3.1 LLL-P

The LLL-P strategy [18] does the column permutations of the LLL reduction algorithm. After the QR factorization (2.1) of $\boldsymbol{A}$, it performs on $\boldsymbol{R}$ in (2.1), and instead of (2.11), the following can be obtained:

$$\bar{\boldsymbol{Q}}^T\boldsymbol{R}\boldsymbol{P} = \bar{\boldsymbol{R}}, \tag{2.20}$$

where $\boldsymbol{P}$ is a permutation matrix and $\bar{\boldsymbol{R}} \in \mathbb{R}^{n\times n}$ is an upper triangular matrix satisfying the Lovász condition (2.13).

Notice that combining (2.1) and (2.20), the LLL-P reduction result in the following QRP factorization (see (2.19)) of $\boldsymbol{A}$:

$$\tilde{\boldsymbol{Q}}^T \boldsymbol{A}\boldsymbol{P} = \begin{bmatrix} \bar{\boldsymbol{R}} \\ \boldsymbol{0} \end{bmatrix}, \quad \tilde{\boldsymbol{Q}} \equiv \boldsymbol{Q} \begin{bmatrix} \bar{\boldsymbol{Q}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{m-n} \end{bmatrix}. \tag{2.21}$$

Like the LLL reduction, permutation matrices and Givens rotations can be used to update $\boldsymbol{R}$ so that $\bar{\boldsymbol{R}}$ satisfies (2.13), for more details, one can refer to [18]. The LLL-P algorithm can be described by Algorithm 2.3.1, where the final reduced upper triangular matrix is still denoted by $\boldsymbol{R}$.

Algorithm 2.3.1 LLL-P

compute the QR factorization: $\boldsymbol{A} = \boldsymbol{Q} \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{bmatrix}$;
set $\boldsymbol{P} = \boldsymbol{I}_n$, $k = 2$;
while $k \leq n$ **do**
  if $\delta\, r_{k-1,k-1}^2 > r_{k-1,k}^2 + r_{kk}^2$ **then**
    perform a column permutation: $\boldsymbol{R} = \boldsymbol{G}_{k-1,k}^T \boldsymbol{R}\boldsymbol{P}_{k-1,k}$;
    update $\boldsymbol{P}$: $\boldsymbol{P} = \boldsymbol{P}\boldsymbol{P}_{k-1,k}$;
    $k = k - 1$, when $k > 2$;
  else
    $k = k + 1$;
  end if
end while

2.3.2 V-BLAST

The so-called vertical Bell Labs layered space time (V-BLAST) optical detection ordering [25] was proposed to do the column reordering for the BILS problem (1.4) in [21].

After the QR factorization (2.1) of $\boldsymbol{A}$, the V-BLAST strategy is performed on $\boldsymbol{R}$ in (2.1), and (2.20) can be obtained. Combining (2.1) and (2.20), the V-BLAST strategy result in the QRP factorization of $\boldsymbol{A}$ in the form (2.21).

The V-BLAST strategy determines the columns of $\bar{\boldsymbol{R}}$ from the last to the first. Suppose columns $n, n-1, \ldots, k+1$ of $\bar{\boldsymbol{R}}$ have been determined, this strategy chooses a column from k remaining columns of $\boldsymbol{R}$ as the k-th column such that $|\bar{r}_{kk}|$ is maximum over all of the k choices. For more details, including efficient algorithms, see [25, 21, 38, 17, 54, 103] etc. For the performance analysis of V-BLAST, one may refer to [56].

2.3.3 SQRD

The sorted QR decomposition strategy (SQRD), which was used to improve the performance of the Babai point $\boldsymbol{x}^{\text{BB}}$ for the BILS (2.6) in [95], was proposed to do the column reordering for the BILS problem (1.4) in [16]. Although simulations in [95] show that the bit error rate of the SQRD aided $\boldsymbol{x}^{\text{BB}}$ is larger than that of the aided V-BLAST $\boldsymbol{x}^{\text{BB}}$, SQRD costs less than V-BLAST.

The original SQRD can perform directly on $\boldsymbol{A}$ in (1.1) and it is computationally more efficient than computing the QR factorization of $\boldsymbol{A}$ in (2.1) and updating $\boldsymbol{R}$. For the sake of description simplicity, we assume that the SQRD strategy is performed on $\boldsymbol{R}$ in (2.1), then like using the V-BLAST strategy, (2.20) can be obtained. Combining (2.1) and (2.20), the SQRD result in the QRP factorization of $\boldsymbol{A}$ in the form (2.21).

In contrast to the V-BLAST, the SQRD strategy determines the columns of the permuted $\bar{\boldsymbol{R}}$ from the first to the last by using the modified Gram-Schmidt algorithm or the Householder QR algorithm. In the k-th step of the algorithm, the k-th column

of the permuted $\bar{\boldsymbol{R}}$ we seek is chosen from the remaining $n-k+1$ columns of $\boldsymbol{R}$ such that $|\bar{r}_{kk}|$ is smallest. For more details and an efficient algorithm, see [95] and [16].

2.3.4 Reduced BILS

With (2.20), let

$$\begin{aligned} &\bar{\boldsymbol{y}} = \bar{\boldsymbol{Q}}^T\tilde{\boldsymbol{y}}, \quad \hat{\boldsymbol{z}} = \boldsymbol{P}^T\hat{\boldsymbol{x}}, \quad \bar{\boldsymbol{v}} = \bar{\boldsymbol{Q}}^T\tilde{\boldsymbol{v}}, \\ &\boldsymbol{z} = \boldsymbol{P}^T\boldsymbol{x}, \quad \bar{\boldsymbol{l}} = \boldsymbol{P}^T\boldsymbol{l}, \quad \bar{\boldsymbol{u}} = \boldsymbol{P}^T\boldsymbol{u}. \end{aligned} \tag{2.22}$$

Then by (2.20), the linear model (2.5) is transformed to

$$\bar{\boldsymbol{y}} = \bar{\boldsymbol{R}}\hat{\boldsymbol{z}} + \bar{\boldsymbol{v}}, \quad \bar{\boldsymbol{v}} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}), \tag{2.23a}$$

$$\hat{\boldsymbol{z}} \in \bar{\mathcal{B}} = \{\boldsymbol{z} \in \mathbb{Z}^n : \bar{\boldsymbol{l}} \le \boldsymbol{z} \le \bar{\boldsymbol{u}},\ \bar{\boldsymbol{l}},\ \bar{\boldsymbol{u}} \in \mathbb{Z}^n\}, \tag{2.23b}$$

and the BILS problem (2.6) is transformed to

$$\min_{\boldsymbol{z}\in\bar{\mathcal{B}}} \|\bar{\boldsymbol{y}} - \bar{\boldsymbol{R}}\boldsymbol{z}\|_2^2 \tag{2.24}$$

whose solution $\boldsymbol{z}^{\text{BL}}$ is the BILS estimator of $\hat{\boldsymbol{z}}$. One can also define the box-constrained Babai point $\boldsymbol{z}^{\text{BB}}$ for (2.24) (see (2.8)).

CHAPTER 3
Effects of the LLL and Some QRP Reductions on the Success Probability of the Ordinary Babai Estimator

In this chapter, we investigate the effects of the LLL and some QRP reductions on the success probability P^{OB} of the Babai point $\boldsymbol{x}^{\text{OB}}$ (see (2.7)) for the linear model (1.1). The success probability P^{OB} is very important for several reasons. If $\boldsymbol{x}^{\text{OB}}$ is used as an estimator of the integer parameter vector $\hat{\boldsymbol{x}}$ in (1.1), certainly it is important to find its success probability P^{OB}, which can easily be computed. Even if one intends to solve the OILS problem (1.2) to get the OILS estimator $\boldsymbol{x}^{\text{OL}}$, it is still important to find P^{OB} since it is very difficult to compute the success probability P^{OL} of $\boldsymbol{x}^{\text{OL}}$, and P^{OB}, which is a lower bound on P^{OL}, is often used as an approximation to P^{OL}. In general, the higher the P^{OB}, the lower the computational cost of solving (1.2) by the discrete search approach. In practice, if P^{OB} is high, say close to 1, then one does not need to spend extra computational time to find $\boldsymbol{x}^{\text{OL}}$. It is well-known that the LLL reduction can usually improve P^{OB}. But, there was no rigorous theoretical proof and it was not known whether the LLL reduction can always improve P^{OB}.

In this chapter, we first prove that P^{OB} as a lower bound on P^{OL} is sharper than the lower bound given in [37]; Then, we show rigorously that both the LLL reduction given by Algorithm 2.2.1 and the LLL-P given by Algorithm 2.3.1 increase (not strictly) P^{OB} and investigate how the parameter δ in the LLL reduction affect the enhancement of P^{OB}. After that, we give examples to show that unlike the LLL

and the LLL-P, the permutation strategies SQRD and V-BLAST may decrease P^{OB}. Finally, we give some upper bounds on P^{OB} after the LLL reduction is used by using the entries of $\boldsymbol{R}$ in (2.4). The main results of this chapter were published in [18].

3.1 P^{OB} and P^{OL} and their relationships

In this section, we first give formulas for P^{OB} and P^{OL} and then show that P^{OB} as a lower bound on P^{OL} is sharper than the lower bound given in [37].

In the following, we give a formula to compute P^{OB}. This formula was original given in [83], which considered a variant form of the OILS problem (2.4). But our formula is more concise and our proof is easier to follow than that given in [83].

Theorem 3.1.1 *Let P^{OB} denote the success probability of the Babai estimator $\boldsymbol{x}^{OB}$ given in* (2.7)*, then*

$$P^{OB} \equiv \Pr(\boldsymbol{x}^{OB} = \hat{\boldsymbol{x}}) = \prod_{i=1}^{n} \phi_\sigma(r_{ii}),$$

where

$$\phi_\sigma(\zeta) = \sqrt{\frac{2}{\pi}} \int_0^{\zeta/(2\sigma)} \exp(-\frac{t^2}{2}) dt. \tag{3.1}$$

Proof. By the chain rule of conditional probabilities:

$$\begin{aligned} P^{\text{OB}} = \Pr(\boldsymbol{x}^{\text{OB}} = \hat{\boldsymbol{x}}) = P\big(\bigcap_{i=1}^{n}(x_i^{\text{OB}} = \hat{x}_i)\big) = \Pr(x_n^{\text{OB}} = \hat{x}_n) \\ \times \prod_{i=1}^{n-1} \Pr(x_i^{\text{OB}} = \hat{x}_i | x_{i+1}^{\text{OB}} = \hat{x}_{i+1}, \cdots, x_n^{\text{OB}} = \hat{x}_n). \end{aligned} \tag{3.2}$$

By (2.3), $\tilde{\boldsymbol{y}} \sim \mathcal{N}(\boldsymbol{R}\hat{\boldsymbol{x}}, \sigma^2 \boldsymbol{I})$ and

$$\tilde{y}_n \sim \mathcal{N}(r_{nn}\hat{x}_n, \sigma^2), \;\; \tilde{y}_i \sim \mathcal{N}(r_{ii}\hat{x}_i + \sum_{j=i+1}^{n} r_{ij}\hat{x}_j, \sigma^2), \quad i = n-1, \ldots, 1.$$

Thus, from (2.7) we have

$$c_n^{\mathrm{OB}} \sim \mathcal{N}(\hat{x}_n, \sigma^2/r_{nn}^2),$$

and if $x_{i+1}^{\mathrm{OB}} = \hat{x}_{i+1}, \cdots, x_n^{\mathrm{OB}} = \hat{x}_n$,

$$c_i^{\mathrm{OB}} \sim \mathcal{N}(\hat{x}_i, \sigma^2/r_{ii}^2).$$

Then it follows from (2.7) that

$$\begin{aligned}\Pr(x_n^{\mathrm{OB}} = \hat{x}_n) = \Pr(|c_n - \hat{x}_n| \leq 1/2) &= \frac{1}{\sqrt{2\pi}\frac{\sigma}{r_{nn}}} \int_{-0.5}^{0.5} \exp(-\frac{t^2}{2(\frac{\sigma}{r_{nn}})^2})dt \\ &= \frac{2}{\sqrt{2\pi}} \int_0^{r_{nn}/(2\sigma)} \exp(-\frac{t^2}{2})dt = \phi_\sigma(r_{nn}).\end{aligned}$$

Similarly, we can obtain

$$\Pr(x_i^{\mathrm{OB}} = \hat{x}_i | x_{i+1}^{\mathrm{OB}} = \hat{x}_{i+1}, \cdots, x_n^{\mathrm{OB}} = \hat{x}_n) = \phi_\sigma(r_{ii}).$$

Then from (3.2) we can conclude that the theorem holds. □

In the following, we give a formula, which was original given in [37] without proof, for P^{OL}. For the sake of readability, we prove it.

Theorem 3.1.2 *Let P^{OL} denote the success probability of the OILS estimator $\boldsymbol{x}^{OL}$ for the OILS* (2.4)*, then*

$$P^{OL} \equiv \Pr(\boldsymbol{x}^{OL} = \hat{\boldsymbol{x}}) = \frac{1}{(2\pi\sigma^2)^{n/2}} \int_{V_{OL}} \exp\big(-\frac{1}{2\sigma^2}\|\boldsymbol{\xi}\|_2^2\big)d\boldsymbol{\xi}, \tag{3.3}$$

where

$$V_{OL} = \{\boldsymbol{\xi} \mid 2\boldsymbol{x}^T\boldsymbol{R}^T\boldsymbol{\xi} \leq \|\boldsymbol{R}\boldsymbol{x}\|_2^2 \text{ for } \forall \boldsymbol{x} \in \mathbb{Z}^n\}. \tag{3.4}$$

Proof. Define sets E_1, E_2, E_3 as follows:

$$E_1 = \{\tilde{\boldsymbol{v}} \,|\, \tilde{\boldsymbol{v}} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}), \tilde{\boldsymbol{y}} = \boldsymbol{R}\hat{\boldsymbol{x}} + \tilde{\boldsymbol{v}}, \boldsymbol{x}^{\text{OL}} = \hat{\boldsymbol{x}}\},$$

$$E_2 = \{\tilde{\boldsymbol{v}} \,|\, \tilde{\boldsymbol{v}} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}), \tilde{\boldsymbol{y}} = \boldsymbol{R}\hat{\boldsymbol{x}} + \tilde{\boldsymbol{v}}, \|\tilde{\boldsymbol{v}}\|_2^2 \leq \|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{x}\|_2^2 \text{ for } \forall\, \boldsymbol{x} \in \mathbb{Z}^n\},$$

$$E_3 = \{\tilde{\boldsymbol{v}} \,|\, \tilde{\boldsymbol{v}} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}), \tilde{\boldsymbol{y}} = \boldsymbol{R}\hat{\boldsymbol{x}} + \tilde{\boldsymbol{v}}, \|\tilde{\boldsymbol{v}}\|_2^2 = \|\tilde{\boldsymbol{y}} - \boldsymbol{R}(\boldsymbol{x} + \hat{\boldsymbol{x}})\|_2^2 \text{ for some } \boldsymbol{x} \in \mathbb{Z}^n, \boldsymbol{x} \neq \boldsymbol{0}\}.$$

Then it is easily to show that

$$E_2 \backslash E_3 \subseteq E_1 \subseteq E_2.$$

Therefore, the following equalities hold:

$$\Pr(\tilde{\boldsymbol{v}} \in E_2) - \Pr(\tilde{\boldsymbol{v}} \in E_3) \leq \Pr(\tilde{\boldsymbol{v}} \in E_1) \leq \Pr(\tilde{\boldsymbol{v}} \in E_2). \tag{3.5}$$

We firstly show $\Pr(\tilde{\boldsymbol{v}} \in E_3) = 0$. Let $\tilde{\boldsymbol{v}} \in E_3$, so there exists $\bar{\boldsymbol{x}} \in \mathbb{Z}^n$ such that $\bar{\boldsymbol{x}} \neq \boldsymbol{0}$ and $\|\tilde{\boldsymbol{v}}\|_2^2 = \|\tilde{\boldsymbol{y}} - \boldsymbol{R}(\bar{\boldsymbol{x}} + \hat{\boldsymbol{x}})\|_2^2$. Then, by the following equality,

$$\|\tilde{\boldsymbol{v}}\|_2^2 = \|\tilde{\boldsymbol{y}} - \boldsymbol{R}(\bar{\boldsymbol{x}} + \hat{\boldsymbol{x}})\|_2^2 = \|\tilde{\boldsymbol{v}} - \boldsymbol{R}\bar{\boldsymbol{x}}\|_2^2 = \|\tilde{\boldsymbol{v}}\|_2^2 + \|\boldsymbol{R}\bar{\boldsymbol{x}}\|_2^2 - 2\bar{\boldsymbol{x}}^T \boldsymbol{R}^T \tilde{\boldsymbol{v}}$$

we have $2\bar{\boldsymbol{x}}^T \boldsymbol{R}^T \tilde{\boldsymbol{v}} = \|\boldsymbol{R}\bar{\boldsymbol{x}}\|_2^2$. This indicates that $\tilde{\boldsymbol{v}}$ lies on an $(n-1)-$dimensional plane, but $\tilde{\boldsymbol{v}}$ is an $n-$dimensional random variable, so $\Pr(\tilde{\boldsymbol{v}} \in E_3) = 0$. Therefore, by (3.5), we have

$$\Pr(\tilde{\boldsymbol{v}} \in E_1) = \Pr(\tilde{\boldsymbol{v}} \in E_2). \tag{3.6}$$

Note that $\|\tilde{\boldsymbol{v}}\|_2^2 \leq \|\tilde{\boldsymbol{y}} - \boldsymbol{R}\boldsymbol{x}\|_2^2$ is equivalent to $2\boldsymbol{x}^T \boldsymbol{R}^T \tilde{\boldsymbol{v}} \leq \|\boldsymbol{R}\boldsymbol{x}\|_2^2$. Then, using the density function of $\tilde{\boldsymbol{v}}$, we obtain the formula for P^{OL}. $\square$

From Theorems 3.1.1 and 3.1.2, P^{OB} and P^{OL} depend on $\boldsymbol{R}$, so sometimes we also write them as $P^{\text{OB}}(\boldsymbol{R})$ and $P^{\text{OL}}(\boldsymbol{R})$, respectively.

By Theorem 3.1.2, P^{OL} depends on its Voronoi cell V_{OL} (see (3.4)) and it is difficult to compute it because the shape of V_{OL} is complicated [37]. In [37], a lower bound $F(d^2_{\min}/(4\sigma^2), n)$ is proposed to approximate it, where $d_{\min}$ is the length of the shortest lattice vector, i.e., $d_{\min} = \min_{\mathbf{0}\neq \boldsymbol{x}\in\mathbb{Z}^n} \|\boldsymbol{R}\boldsymbol{x}\|_2$, and $F(x, n)$ denotes the cumulative distribution function of the chi-square distribution with degree n. However, no polynomial-time algorithm has been found to compute $d_{\min}$. To overcome this problem, [37] proposed a more practical lower bound $F(r^2_{\min}/(4\sigma^2), n)$, where $r_{\min} \equiv \min_i r_{ii}$. It is well-known that P^{OB} is also a lower bound on P^{OL}, for more details, see [84]. The following result shows that P^{OB} is a sharper lower bound than $F(r^2_{\min}/(4\sigma^2), n)$.

Theorem 3.1.3 $F\Big(\frac{r^2_{\min}}{4\sigma^2}, n\Big) \leq P^{OB}$.

Proof. Let $\boldsymbol{u} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_n)$. Thus $u_1, u_2, \ldots, u_n$ are i.i.d. and $\sum_{i=1}^n u_i^2$ follows the chi-squared distribution with degree n. Let events $E = \{\sum_{i=1}^n u_i^2 \leq r^2_{\min}/(4\sigma^2)\}$ and $E_i = \{u_i^2 \leq r^2_{ii}/(4\sigma^2)\}$ for $i = 1, 2, \ldots, n$. Since $r_{\min} \leq r_{ii}$, $E \subseteq \bigcap_{i=1}^n E_i$. Thus,

$$
\begin{aligned}
F\Big(\frac{r^2_{\min}}{4\sigma^2}, n\Big) &= \Pr(E) \leq \Pr(\bigcap_{i=1}^n E_i) = \prod_{i=1}^n \Pr(E_i) \\
&= \prod_{i=1}^n \frac{1}{\sqrt{2\pi}} \int_{-r_{ii}/(2\sigma)}^{r_{ii}/(2\sigma)} \exp\big(-\frac{t^2}{2}\big) dt = \prod_{i=1}^n \phi_\sigma(r_{ii}) = P^{\text{OB}}. \quad \square
\end{aligned}
$$

In the following, we give an example to show that $F(r^2_{\min}/(4\sigma^2), n)$ can be much smaller than P^{OB}.

Example 3.1.1 *Let* $\boldsymbol{R} = \begin{bmatrix} 0.001 & 0 \\ 0 & 10 \end{bmatrix}$ *and* $\sigma = 0.5$. *By simple calculations, we obtain* $F(r^2_{\min}/(4\sigma^2), n)/P^{OB} = 1/1596$.

Although this is a contrived example, where the signal-to-noise ratio is small, it shows that P^{OB} can be much sharper than $F(r_{\min}^2/(4\sigma^2), n)$ as a lower bound on P^{OL}.

3.2 Effects of the reductions on P^{OB}

By (2.7), we can define the Babai estimator $\boldsymbol{z}^{\text{OB}}$ for $\hat{\boldsymbol{z}}$ in (2.17) and use it to estimate $\hat{\boldsymbol{z}}$ in (2.17), or equivalently, we use $\boldsymbol{Z}\boldsymbol{z}^{\text{OB}}$ to estimate $\hat{\boldsymbol{x}}$ in (2.4).

In this section, first, we will rigorously prove that column permutations and size reductions on the superdiagonal entries of $\boldsymbol{R}$ that immediately follows a permutation in the process of the LLL reduction given by Algorithm 2.2.1 enhance (not strictly) the success probability P^{OB} of the Babai point $\boldsymbol{x}^{\text{OB}}$, as a result, the LLL-P can improve P^{OB}; Then, we will give simulations to show that unlike LLL and LLL-P, both V-BLAST and SQRD may decrease P^{OB}; After this, we will discuss how the parameter δ affects the enhancement of P^{OB}. At last, we will give some upper bounds on P^{OB}, after the LLL reduction, by using the entries of $\boldsymbol{R}$ in (2.1).

3.2.1 Effects of the LLL reduction and the LLL-P on P^{OB}

In the subsection we look at how the success probability P^{OB} of the Babai point $\boldsymbol{x}^{\text{OB}}$ changes after LLL reduction or LL-P is performed to $\boldsymbol{R}$.

The following result shows that if the Lovász condition (2.13) is not satisfied, after a column permutation and triangularization, P^{OB} increases.

Lemma 3.2.1 *Suppose that $\delta\, r_{k-1,k-1}^2 > r_{k-1,k}^2 + r_{kk}^2$ for some k for the $\boldsymbol{R}$ matrix in the OILS problem* (2.4)*. After the permutation of columns $k-1$ and k and triangularization, $\boldsymbol{R}$ becomes $\bar{\boldsymbol{R}}$, i.e., $\bar{\boldsymbol{R}} = \boldsymbol{G}_{k-1,k}^T \boldsymbol{R}\boldsymbol{P}_{k-1,k}$* (see (2.14))*. With $\bar{\boldsymbol{y}} = \boldsymbol{G}_{k-1,k}^T \tilde{\boldsymbol{y}}$ and $\boldsymbol{z} = \boldsymbol{P}_{k-1,k}^{-1}\boldsymbol{x}$,* (2.4) *can be transformed to* (2.18)*. Denote $\hat{\boldsymbol{z}} \equiv \boldsymbol{P}_{k-1,k}^{-1}\hat{\boldsymbol{x}}$. Then the Babai point $\boldsymbol{z}^{OB}$ has a success probability greater than or equal to the Babai*

point $\boldsymbol{x}^{OB}$*, i.e.,*

$$\Pr(\boldsymbol{x}^{OB} = \hat{\boldsymbol{x}}) \leq \Pr(\boldsymbol{z}^{OB} = \hat{\boldsymbol{z}}), \tag{3.7}$$

where the equality holds if and only if $r_{k-1,k} = 0$*.*

Proof. By Theorem 3.1.1, what we need to show is the following inequality:

$$\prod_{i=1}^{n} \phi_\sigma(r_{ii}) \leq \prod_{i=1}^{n} \phi_\sigma(\bar{r}_{ii}). \tag{3.8}$$

Since $\bar{r}_{ii} = r_{ii}$ for $i \neq k-1, k$, we only need to show

$$\phi_\sigma(r_{k-1,k-1})\phi_\sigma(r_{kk}) \leq \phi_\sigma(\bar{r}_{k-1,k-1})\phi_\sigma(\bar{r}_{kk}),$$

which is equivalent to

$$\int_0^{\frac{r_{k-1,k-1}}{2\sigma}} \exp(-\frac{t^2}{2})dt \int_0^{\frac{r_{kk}}{2\sigma}} \exp(-\frac{t^2}{2})dt \leq \int_0^{\frac{\bar{r}_{k-1,k-1}}{2\sigma}} \exp(-\frac{t^2}{2})dt \int_0^{\frac{\bar{r}_{kk}}{2\sigma}} \exp(-\frac{t^2}{2})dt. \tag{3.9}$$

By the last equality in (2.15), we can let

$$a = \frac{r_{k-1,k-1}}{2\sigma}\frac{r_{kk}}{2\sigma} = \frac{\bar{r}_{k-1,k-1}}{2\sigma}\frac{\bar{r}_{kk}}{2\sigma}, \tag{3.10}$$

$$f(\zeta) = \ln \int_0^{\zeta} \exp(-\frac{t^2}{2})dt + \ln \int_0^{a/\zeta} \exp(-\frac{t^2}{2})dt. \tag{3.11}$$

Note that $f(\zeta) = f(a/\zeta) = f(\max\{\zeta, a/\zeta\})$. Then (3.9) is equivalent to

$$f\Big(\frac{\max\{r_{k-1,k-1}, r_{kk}\}}{2\sigma}\Big) \leq f\Big(\frac{\max\{\bar{r}_{k-1,k-1}, \bar{r}_{kk}\}}{2\sigma}\Big). \tag{3.12}$$

Obviously, if $r_{k-1,k} = 0$, then the equality in (3.12) holds since in this case

$$\frac{\max\{r_{k-1,k-1}, r_{kk}\}}{2\sigma} = \frac{\max\{\bar{r}_{k-1,k-1}, \bar{r}_{kk}\}}{2\sigma}.$$

So we only need to show if $r_{k-1,k} \neq 0$, then the strict inequality in (3.12) holds. In the following, we assume $r_{k-1,k} \neq 0$.

From $\delta r_{k-1,k-1}^2 > r_{k-1,k}^2 + r_{kk}^2$ and (2.15) we can conclude that

$$r_{kk}, \bar{r}_{k-1,k-1}, \bar{r}_{kk} < r_{k-1,k-1}.$$

Then, with (3.10) it follows that

$$\frac{\max\{r_{k-1,k-1}, r_{kk}\}}{2\sigma} = \frac{r_{k-1,k-1}}{2\sigma} > \frac{\max\{\bar{r}_{k-1,k-1}, \bar{r}_{kk}\}}{2\sigma} \geq \sqrt{a}.$$

Thus, to show the strict inequality in (3.12) holds, it suffices to show that when $\zeta > \sqrt{a}$, $f(\zeta)$ is a strict monotonically decreasing function or equivalently $f'(\zeta) < 0$.

From (3.11),

$$f'(\zeta) = \frac{\exp(-\frac{1}{2}\zeta^2)}{\int_0^\zeta \exp(-\frac{t^2}{2})dt} - \frac{\frac{a}{\zeta^2}\exp(-\frac{(a/\zeta)^2}{2})}{\int_0^{a/\zeta} \exp(-\frac{t^2}{2})dt} = \frac{1}{\zeta}\left(g(\zeta) - g\Big(\frac{a}{\zeta}\Big)\right),$$

where $g(\zeta) = \frac{\zeta \exp(-\frac{1}{2}\zeta^2)}{\int_0^\zeta \exp(-\frac{t^2}{2})dt}$. Note that $\zeta > \sqrt{a}$, $\zeta > a/\zeta$. Thus, in order to show $f'(\zeta) < 0$ for $\zeta > \sqrt{a}$, we need only to show that $g(\zeta)$ is a strict monotonically decreasing function or equivalently $g'(\zeta) < 0$ when $\zeta > 0$.

Simple calculations give

$$g'(\zeta) = \frac{\exp(-\frac{1}{2}\zeta^2)}{(\int_0^\zeta \exp(-\frac{t^2}{2})dt)^2} \times \left[(1-\zeta^2)\int_0^\zeta \exp(-\frac{t^2}{2})dt - \zeta\exp(-\frac{1}{2}\zeta^2)\right].$$

If $1 - \zeta^2 \leq 0$ and $\zeta > 0$, then obviously $g'(\zeta) < 0$. If $1 - \zeta^2 > 0$ and $\zeta > 0$, since $\exp(-\frac{t^2}{2}) \leq 1$,

$$(1-\zeta^2)\int_0^\zeta \exp(-\frac{t^2}{2})dt \leq \zeta(1-\zeta^2) < \zeta\exp(-\frac{1}{2}\zeta^2),$$

where the second inequality can easily be verified. Thus again $g'(\zeta) < 0$ when $\zeta > 0$, completing the proof. $\square$

Now we make some remarks. The above proof shows that $f(\zeta)$ for $\zeta \geq \sqrt{a}$ reaches its maximum when $\zeta = \sqrt{a}$. Thus if $\bar{r}_{k-1,k-1} = \bar{r}_{kk}$, or equivalently,

$$r_{k-1,k}^2 + r_{kk}^2 = r_{k-1,k-1} r_{kk},$$

P^{OB} will increase most. For a more general result, see Lemma 3.3.2 and the remark after it.

In Lemma 3.2.1 there is no requirement that $r_{k-1,k}$ should be size-reduced. The question we would like to ask here is do size reductions in the LLL reduction algorithm affect P^{OB}? From Theorem 3.1.1 we observe that P^{OB} only depends on the diagonal entries of $\boldsymbol{R}$. Thus size reductions *alone* will not change P^{OB}. However, if a size reduction can bring changes to the diagonal entries of $\boldsymbol{R}$ after a permutation, then it will likely affect P^{OB}. Therefore, all the size reductions on the off-diagonal entries above the superdiagonal have no effect on P^{OB}. But the size reductions on the superdiagonal entries may affect P^{OB}. There are a few different situations, which we will discuss below.

Suppose that the Lovász condition (2.13) holds for a specific k. If (2.13) does not hold any more after the size reduction on $r_{k-1,k}$, then columns $k-1$ and k of $\boldsymbol{R}$ are permuted by the LLL reduction algorithm and according to Lemma 3.2.1 P^{OB} strictly increases or keeps unchanged if and only if the size reduction makes $r_{k-1,k}$ zero (this occurs if $r_{k-1,k}$ is a multiple of $r_{k-1,k-1}$ before the reduction). If (2.13) still holds after the size reduction on $r_{k-1,k}$, then this size reduction does not affect P^{OB}.

Suppose that the Lovász condition (2.13) does not hold for a specific k. Then by Lemma 3.2.1 P^{OB} increases after a permutation and triangularization. If the size reduction on $r_{k-1,k}$ is performed before the permutation, we show in the next lemma that P^{OB} increases further.

Lemma 3.2.2 *Suppose that in the OILS problem* (2.4) $\boldsymbol{R}$ *satisfies* $\delta\, r_{k-1,k-1}^2 > r_{k-1,k}^2 + r_{kk}^2$ *and* $|r_{k-1,k}| > r_{k-1,k-1}/2$ *for some* k. *Let* $\bar{\boldsymbol{R}}$, $\bar{\boldsymbol{y}}$, $\boldsymbol{z}$ *and* $\hat{\boldsymbol{z}}$ *be defined as in Lemma 3.2.1. Suppose a size reduction on* $r_{k-1,k}$ *is performed first and then after the permutation of columns* $k-1$ *and* k *and triangularization,* $\boldsymbol{R}$ *becomes* $\hat{\boldsymbol{R}}$, *i.e.,* $\hat{\boldsymbol{R}} = \hat{\boldsymbol{G}}_{k-1,k}^T \boldsymbol{R} \boldsymbol{Z}_{k-1,k} \boldsymbol{P}_{k-1,k}$. *Let* $\hat{\boldsymbol{y}} = \hat{\boldsymbol{G}}_{k-1,k}^T \tilde{\boldsymbol{y}}$ *and* $\boldsymbol{w} = \boldsymbol{P}_{k-1,k}^{-1} \boldsymbol{Z}_{k-1,k}^{-1} \boldsymbol{x}$, *then* (2.4) *is transformed to* $\min_{\boldsymbol{w}\in\mathbb{Z}^n} \|\hat{\boldsymbol{y}} - \hat{\boldsymbol{R}}\boldsymbol{w}\|_2$. *Denote* $\hat{\boldsymbol{w}} = \boldsymbol{P}_{k-1,k}^{-1} \boldsymbol{Z}_{k-1,k}^{-1} \hat{\boldsymbol{x}}$. *Then the Babai point* $\boldsymbol{w}^{OB}$ *corresponding to the new transformed ILS problem has a success probability greater than or equal to the Babai point* $\boldsymbol{z}^{OB}$, *i.e.,*

$$\Pr(\boldsymbol{z}^{OB} = \hat{\boldsymbol{z}}) \leq \Pr(\boldsymbol{w}^{OB} = \hat{\boldsymbol{w}}), \tag{3.13}$$

where the equality holds if and only if

$$|r_{k-1,k-1} r_{k-1,k}| = r_{k-1,k}^2 + r_{kk}^2. \tag{3.14}$$

Proof. Obviously (3.13) is equivalent to

$$\phi_\sigma(\bar{r}_{k-1,k-1})\phi_\sigma(\bar{r}_{kk}) \leq \phi_\sigma(\hat{r}_{k-1,k-1})\phi_\sigma(\hat{r}_{kk}),$$

which, by the proof of Lemma 3.2.1, is also equivalent to

$$f\Big(\frac{\max\{\bar{r}_{k-1,k-1}, \bar{r}_{kk}\}}{2\sigma}\Big) \leq f\Big(\frac{\max\{\hat{r}_{k-1,k-1}, \hat{r}_{kk}\}}{2\sigma}\Big),$$

where f is defined in (3.11). Since $f(\zeta)$ has been showed to be strict monotonically decreasing when $\zeta > \sqrt{a}$, what we need to show is that

$$\max\{\bar{r}_{k-1,k-1}, \bar{r}_{kk}\} \geq \max\{\hat{r}_{k-1,k-1}, \hat{r}_{kk}\}, \tag{3.15}$$

where the equality holds if and only if (3.14) holds.

Since $|r_{k-1,k}| > r_{k-1,k-1}/2$,

$$\bar{r}_{k-1,k-1} = \sqrt{r_{k-1,k}^2 + r_{kk}^2} > \sqrt{r_{k-1,k-1}^2/4 + r_{kk}^2},$$
$$\bar{r}_{kk} = \frac{r_{k-1,k-1} r_{kk}}{\sqrt{r_{k-1,k}^2 + r_{kk}^2}} < \frac{r_{k-1,k-1} r_{kk}}{\sqrt{r_{k-1,k-1}^2/4 + r_{kk}^2}}.$$

But $\sqrt{r_{k-1,k-1}^2/4 + r_{kk}^2} \geq \frac{r_{k-1,k-1} r_{kk}}{\sqrt{r_{k-1,k-1}^2/4 + r_{kk}^2}}$, thus

$$\max\{\bar{r}_{k-1,k-1}, \bar{r}_{kk}\} = \bar{r}_{k-1,k-1}.$$

Suppose that after the size reduction, $r_{k-1,k}$ becomes $\tilde{r}_{k-1,k}$. Note that

$$\hat{r}_{k-1,k-1} = \sqrt{\tilde{r}_{k-1,k}^2 + r_{kk}^2} < \sqrt{r_{k-1,k}^2 + r_{kk}^2} = \bar{r}_{k-1,k-1}.$$

Thus, it follows from (3.15) what we need to prove is that $\hat{r}_{kk} \leq \bar{r}_{k-1,k-1}$ or equivalently

$$\hat{r}_{kk} \leq \sqrt{r_{k-1,k}^2 + r_{kk}^2}, \tag{3.16}$$

and the equality holds if and only if (3.14) holds.

By the conditions given in the lemma,

$$|r_{k-1,k}| < r_{k-1,k-1} < 2|r_{k-1,k}|.$$

Thus

$$\tilde{r}_{k-1,k} = r_{k-1,k} - \lfloor r_{k-1,k}/r_{k-1,k-1} \rceil r_{k-1,k-1} = r_{k-1,k} - \text{sign}(r_{k-1,k}) r_{k-1,k-1}.$$

Now we consider two cases $r_{k-1,k} > 0$ and $r_{k-1,k} < 0$ separately. If $r_{k-1,k} > 0$, then

$$\hat{r}_{kk} = \frac{r_{k-1,k-1} r_{kk}}{\hat{r}_{k-1,k-1}} = \frac{r_{k-1,k-1} r_{kk}}{\sqrt{\tilde{r}_{k-1,k}^2 + r_{kk}^2}} = \frac{r_{k-1,k-1} r_{kk}}{\sqrt{(r_{k-1,k} - r_{k-1,k-1})^2 + r_{kk}^2}}.$$

Thus, to show (3.16) it suffices to show that

$$\frac{r_{k-1,k-1} r_{kk}}{\sqrt{(r_{k-1,k} - r_{k-1,k-1})^2 + r_{kk}^2}} \leq \sqrt{r_{k-1,k}^2 + r_{kk}^2}.$$

Simple algebraic manipulations shows that the above inequality is equivalent to

$$(r_{k-1,k-1} r_{k-1,k} - r_{k-1,k}^2 - r_{kk}^2)^2 \geq 0,$$

which certainly holds. And obviously, the equality in (3.16) holds if and only if

$$r_{k-1,k-1} r_{k-1,k} = r_{k-1,k}^2 + r_{kk}^2.$$

If $r_{k-1,k} < 0$, we can similarly prove that (3.16) holds and the equality holds if and only if

$$-r_{k-1,k-1} r_{k-1,k} = r_{k-1,k}^2 + r_{kk}^2,$$

completing the proof. $\square$

Here we make a remark about the equality (3.14). From the proof of Lemma 3.2.2, we see that if (3.14) holds, then the equality in (3.16) holds, thus $\hat{r}_{kk} = \bar{r}_{k-1,k-1}$. But the absolute value of the determinant of the submatrix $\boldsymbol{R}_{k-1:k,k-1:k}$ is unchanged

by the size reduction, we must have $\hat{r}_{k-1,k-1} = \bar{r}_{kk}$. Thus if (3.14) holds, the effect of the size reduction on $r_{k-1,k}$ is to make $\bar{r}_{k-1,k-1}$ and $\bar{r}_{kk}$ permuted; therefore the success probability P^{OB} is not changed by the size reduction. Here we give an example.

Example 3.2.1 *Let* $\boldsymbol{R} = \begin{bmatrix} 5 & 4 \\ 0 & 2 \end{bmatrix}$. *Then it is easy to verify that* $\bar{\boldsymbol{R}} = \begin{bmatrix} 2\sqrt{5} & 2\sqrt{5} \\ 0 & \sqrt{5} \end{bmatrix}$ *and* $\hat{\boldsymbol{R}} = \begin{bmatrix} \sqrt{5} & -\sqrt{5} \\ 0 & 2\sqrt{5} \end{bmatrix}$. *From the diagonal entries of* $\bar{\boldsymbol{R}}$ *and* $\hat{\boldsymbol{R}}$ *we can conclude that the success probabilities of the two Babai points corresponding to* $\bar{\boldsymbol{R}}$ *and* $\hat{\boldsymbol{R}}$ *are equal.*

From Lemmas 3.2.1 and 3.2.2 we immediately obtain the following results.

Theorem 3.2.1 *Suppose that the OILS problem* (2.4) *is transformed to the OILS problem* (2.18)*, where* $\bar{\boldsymbol{R}}$ *is obtained by the LLL reduction given by Algorithm 2.2.1. Then*

$$\Pr(\boldsymbol{x}^{OB} = \hat{\boldsymbol{x}}) \leq \Pr(\boldsymbol{z}^{OB} = \hat{\boldsymbol{z}}),$$

where the equality holds if and only if no column permutation occurs during the LLL reduction process or whenever two consecutive columns, say $k-1$ *and* k*, are permuted,* $r_{k-1,k}$ *is a multiple of* $r_{k-1,k-1}$ *(before the size reduction on* $r_{k-1,k}$ *is performed). Any size reductions on the superdiagonal entries of* $\boldsymbol{R}$ *which are immediately followed by a column permutation during the LLL reduction process will enhance the success probability of the Babai point. All other size reductions have no effect on the success probability of the Babai point.*

Now we make some remarks. Note that the LLL reduction is not unique. Two different LLL reduction algorithms may produce different $\boldsymbol{R}$'s. In Algorithm 2.2.1,

when the Lovász condition for two consecutive columns is not satisfied, then a column permutation takes places to ensure the Lovász condition to be satisfied. If an algorithm which computes the LLL reduction does not do permutations as Algorithm 2.2.1 does, e.g., the algorithm permutes two columns which are not consecutive or permutes two consecutive columns but the corresponding Lovász condition is not satisfied after the permutation, then we cannot guarantee this specific LLL reduction will increase P^{OB}.

It is showed in [81] that the LLL-aided zero-forcing can achieve the maximum receive diversity in MIMO systems. Here, we show the effects of LLL reduction from another point of view.

It is interesting to note that [53] showed that all the size reductions on the off-diagonal entries above the superdiagonal of $\boldsymbol{R}$ have no effect on the residual of the Babai point. Here we see that those size reductions are not useful from another perspective.

From Lemmas 3.2.1 we can also obtain the following results.

Theorem 3.2.2 *Suppose that the OILS problem* (2.4) *is transformed to the OILS problem* (2.18)*, where* $\bar{\boldsymbol{R}}$ *is obtained by the LLL-P which is given by Algorithm 2.3.1. Then*

$$\Pr(\boldsymbol{x}^{OB} = \hat{\boldsymbol{x}}) \leq \Pr(\boldsymbol{z}^{OB} = \hat{\boldsymbol{z}}),$$

where the equality holds if and only if no column permutation occurs during the process of the LLL-P or whenever two consecutive columns, say $k-1$ *and* k*, are permuted,* $r_{k-1,k} = 0$*.*

3.2.2 Effects of the V-BLAST and SQRD on P^{OB}

Theorem 3.2.1 and 3.2.2 respectively show that the LLL reduction and the LLL-P can always improve (not strictly) P^{OB}. In this subsection, we will show that both V-BLAST and SQRD may decrease P^{OB}.

In the following, we give simple numerical test results to see how SQRD, V-BLAST, LLL-P with $\delta = 1$ and LLL with $\delta = 1$ affect P^{OB}.

We performed our MATLAB simulations for the following two cases:

- Case 1: $\boldsymbol{A} = \text{randn}(n, n)$, where $\text{randn}(n, n)$ is a MATLAB built-in function to generate a random $n \times n$ matrix, whose entries follow the normal distribution $\mathcal{N}(0, 1)$.
- Case 2: $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T$, $\boldsymbol{U}, \boldsymbol{V}$ are random orthogonal matrices obtained by the QR factorization of random matrices generated by $\text{randn}(n, n)$ and $\boldsymbol{D}$ is a $n \times n$ diagonal matrix with $d_{ii} = 10^{3(n/2-i)/(n-1)}$.

In the tests for each case for a fixed n we gave 200 runs to generate 200 different $\boldsymbol{A}$'s. For $n = 20$, Figures 3–1 and 3–2 display the average success probabilities of the Babai points corresponding to various reduction or permutation strategies over 200 runs versus $\sigma = 0.05 : 0.05 : 0.4$, for Cases 1 and 2, respectively. In both figures, "QR" means the QR factorization is used, giving $\Pr(\boldsymbol{x}^{\text{OB}} = \hat{\boldsymbol{x}})$.

From Figures 3–1 and 3–2, we can see that on average the LLL reduction improves P^{OB} much more significantly than the other three, V-BLAST performs better than LLL-P and SQRD, and LLL-P and SQRD have similar performance. Note that LLL-P is the permutation strategy of the LLL reduction and the size reductions on the super diagonal entries can further increase P^{OB} under some conditions (see

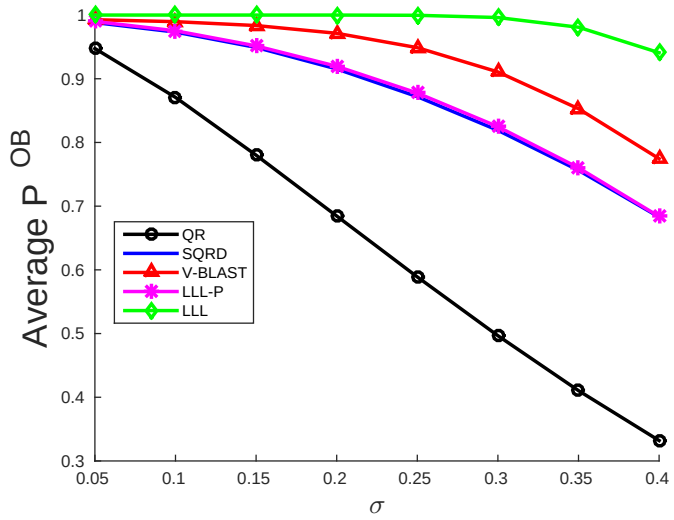

Figure 3–1: Average P^{OB} over 200 runs versus σ for Case 1, $n = 20$

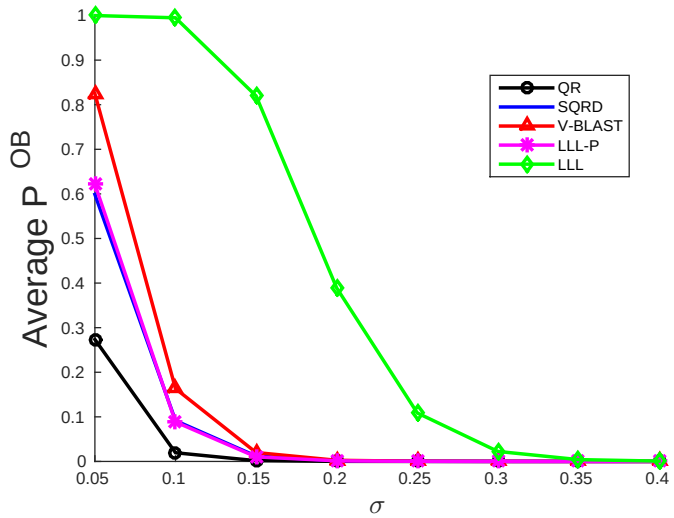

Figure 3–2: Average P^{OB} over 200 runs versus σ for Case 2, $n = 20$

Lemma 3.2.2) and may bring more permutations, so it is easy to understand that the LLL reduction can usually bring higher improvement. We observed the same phenomenon when we changed the dimensions of $\boldsymbol{A}$.

Figures 3–1 and 3–2 indicate that on average SQRD and V-BLAST increase P^{OB}. However, unlike LLL-P, both SQRD and V-BLAST may decrease P^{OB} sometimes. Table 3–1 gives the number of runs out of 200 in which SQRD and V-BLAST decrease P^{OB} for various σ and n. From the table we can see that for both Cases 1 and 2, the chance that SQRD decreases P^{OB} is much larger than V-BLAST and when σ increases, the chance that SQRD decreases P^{OB} tends to decrease. For Case 2, when n increases, the chance that SQRD decreases P^{OB} tends to decrease, but this phenomenon is not seen for Case 1.

Table 3–1: Number of runs out of 200 in which P^{OB} decreases

		Case 1			Case 2		
Methods	n \ σ	0.1	0.2	0.3	0.1	0.2	0.3
SQRD	10	9	10	6	13	8	5
	20	12	11	7	6	2	1
	30	16	14	11	0	1	1
	40	15	9	5	0	0	0
V-BLAST	10	0	0	0	2	6	7
	20	0	0	0	0	0	0
	30	0	0	0	0	0	0
	40	0	0	0	0	0	0

From Table 3–1, one can easily find concrete examples to show that neither SQRD nor V-BLAST can always improve P^{OB}.

3.2.3 Effects of δ in the LLL reduction on the enhancement of P^{OB}

Suppose that $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ are obtained by applying Algorithm 2.2.1 to $\boldsymbol{A}$ with $\delta = \delta_1$ and $\delta = \delta_2$, respectively and $\delta_1 < \delta_2$. A natural question is what is the relation between $P^{\text{OB}}(\boldsymbol{R}_1)$ and $P^{\text{OB}}(\boldsymbol{R}_2)$? We try to address this question in this subsection.

First we give a result for $n = 2$.

Theorem 3.2.3 *Suppose that $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ are obtained by applying Algorithm 2.2.1 to $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ with $\delta = \delta_1$ and $\delta = \delta_2$, respectively and $\delta_1 < \delta_2$. If $n = 2$, then*

$$P^{OB}(\boldsymbol{R}_1) \leq P^{OB}(\boldsymbol{R}_2). \tag{3.17}$$

Proof. Note that only two columns are involved in the reduction process and the value of δ only determines when the process should terminate. In the reduction process, the upper triangular matrix $\boldsymbol{R}$ either first becomes δ_1-LLL reduced and then becomes δ_2-LLL reduced after some more permutations or becomes δ_1-LLL reduced and δ_2-LLL reduced at the same time. Therefore, by Lemma 3.2.1 the conclusion holds. □

However, the inequality (3.17) in Theorem 3.2.3 may not hold when $n \geq 3$. In fact, for any given $n \geq 3$, we can give an example to illustrate this.

Example 3.2.2 *Let δ_1 and δ_2 satisfy $1/4 < \delta_1 < \delta_2 \leq 1$ and $\delta_2 < \delta_1^2 + 1/4$. Let η and θ satisfy $\delta_1 < \eta < \delta_2$ and $0 < \theta < \frac{1}{2}\sqrt{\delta_1(\eta - \delta_1)}$. Let*

$$\boldsymbol{R} = \begin{bmatrix} 1 & 0 & 1/2 \\ 0 & \sqrt{\eta} & \theta \\ 0 & 0 & \delta_1 \end{bmatrix}. \tag{3.18}$$

Note that $\boldsymbol{R}$ is size reduced already.

Suppose that we apply Algorithm 2.2.1 with $\delta = \delta_1$ to $\boldsymbol{R}$, leading to $\boldsymbol{R}_1$. The first two columns of $\boldsymbol{R}$ do not permute as the Lovász condition holds. However, the Lovász condition does not hold for the last two columns and a permutation is needed. Then by Lemma 3.2.1 we must have $P^{OB}(\boldsymbol{R}_1) > P^{OB}(\boldsymbol{R})$.

Applying Algorithm 2.2.1 with $\delta = \delta_2$ to $\boldsymbol{R}$, we obtain

$$\boldsymbol{R}_2 = \begin{bmatrix} \sqrt{\eta} & 0 & \theta \\ 0 & 1 & 1/2 \\ 0 & 0 & \delta_1 \end{bmatrix},$$

whose diagonal entries are the same as those of $\boldsymbol{R}$ with a different order. Then we have $P^{OB}(\boldsymbol{R}_2) = P^{OB}(\boldsymbol{R})$. Therefore, $P^{OB}(\boldsymbol{R}_1) > P^{OB}(\boldsymbol{R}_2)$.

With $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$ given in (3.18), we define $\boldsymbol{A}$ as $\boldsymbol{A} = \left[\begin{smallmatrix} \boldsymbol{R} & 0 \\ 0 & \boldsymbol{I}_{n-3} \end{smallmatrix}\right] \in \mathbb{R}^{n\times n}$, it is easy to show that we still have $P^{OB}(\boldsymbol{R}_1) > P^{OB}(\boldsymbol{R}_2)$, where $\boldsymbol{R}_1$ and $\boldsymbol{R}_2$ were obtained by applying Algorithm 2.2.1 to $\boldsymbol{A}$ with $\delta = \delta_1$ and $\delta = \delta_2$, respectively.

Although the above example shows that larger δ may not guarantee to produce higher P^{OB} when $n \geq 3$, we can expect that the chance that $P^{\mathrm{OB}}(\boldsymbol{R}_1) \leq P^{\mathrm{OB}}(\boldsymbol{R}_2)$ is much higher than the chance that $P^{\mathrm{OB}}(\boldsymbol{R}_1) > P^{\mathrm{OB}}(\boldsymbol{R}_2)$. Here we give an explanation. If $\boldsymbol{R}_1$ is not δ_2-LLL reduced, applying Algorithm 2.2.1 with $\delta = \delta_2$ to $\boldsymbol{R}_1$ produces $\bar{\boldsymbol{R}}_1$ with $P^{\mathrm{OB}}(\bar{\boldsymbol{R}}_1) \geq P^{\mathrm{OB}}(\boldsymbol{R}_1)$. Although $\bar{\boldsymbol{R}}_1$ may not be equal to $\boldsymbol{R}_2$, we can expect that the difference between these two δ_2-LLL reduced matrices is small. Thus it is likely that $P^{\mathrm{OB}}(\boldsymbol{R}_2) \approx P^{\mathrm{OB}}(\bar{\boldsymbol{R}}_1) \geq P^{\mathrm{OB}}(\boldsymbol{R}_1)$.

Here we give numerical results to show how δ affects P^{OB} (i.e., $\Pr(\boldsymbol{z}^{\mathrm{OB}} = \hat{\boldsymbol{z}})$). We used the matrices defined in Cases 1 and 2 of Section 3.2.2. As before, in the

tests for each case we gave 200 runs to generate 200 different $\boldsymbol{A}$'s for a fixed n. For $n = 20$, Figures 3–3 and 3–4 display the average $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$ over 200 runs versus $\delta = 0.3 : 0.1 : 1.0$ for Cases 1 and 2, respectively. The three curves in both figures correspond to $\sigma = 0.1, 0.2, 0.3$. For comparisons, we give the corresponding $\Pr(\boldsymbol{x}^{\text{OB}} = \hat{\boldsymbol{x}})$ in the following table.

Table 3–2: Success probability $\Pr(\boldsymbol{x}^{\text{OB}} = \hat{\boldsymbol{x}})$

	$\sigma = 0.1$	$\sigma = 0.2$	$\sigma = 0.3$
Case 1	0.864	0.679	0.490
Case 2	1.92×10^{-2}	1.96×10^{-4}	5.54×10^{-6}

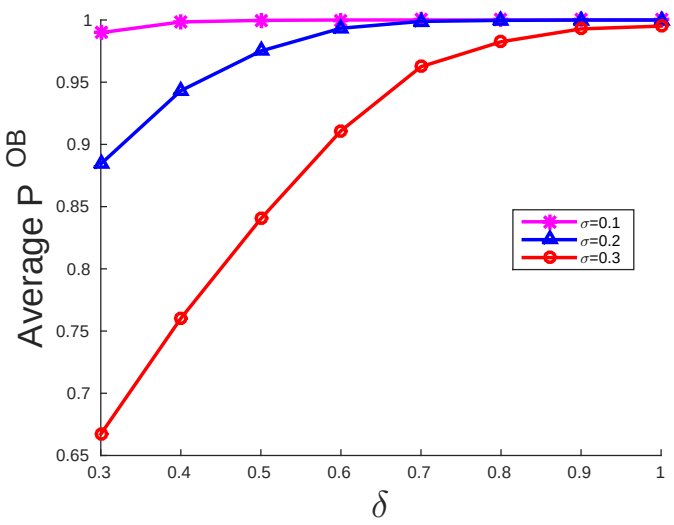

Figure 3–3: Average P^{OB} over 200 runs after the LLL reduction for Case 1, $n = 20$

From Table 3–2, Figures 3–3 and 3–4, we can see that the LLL reduction has a significant effect on improving P^{OB}. Figures 3–3 and 3–4 show that as δ increases, on average P^{OB} increases too, in particular for large σ. But we want to point out that we also noticed that sometimes a larger δ resulted in a smaller P^{OB} in the tests. Table

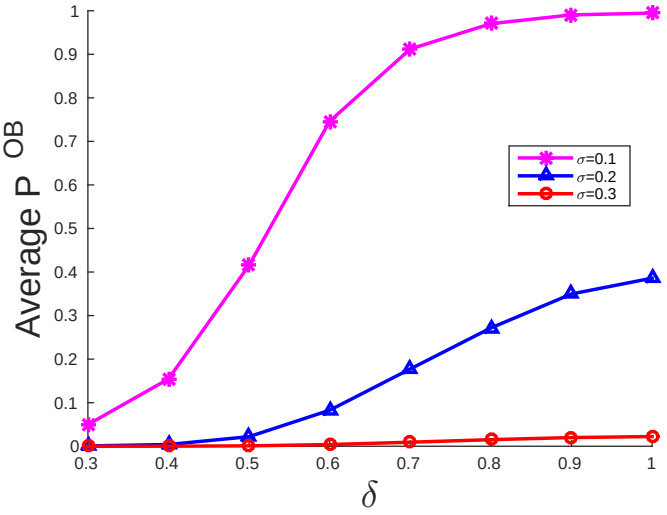

Figure 3–4: Average P^{OB} over 200 runs after the LLL reduction for Case 2, $n = 20$

3–3 gives the exact number of runs out of those 200 runs in which P^{OB} decreases when δ increases from t to $t + 0.1$ for $t = 0.3 : 0.1 : 0.9$. From Table 3–3 we can see that most of the time P^{OB} does not decrease when δ increases. We would like to point out that in our numerical tests we tried various dimension size n for the two test cases and observed the same phenomena.

Table 3–3: Number of runs out of 200 in which P^{OB} decreases when δ increases

	Case 1			Case 2		
δ \ σ	0.1	0.2	0.3	0.1	0.2	0.3
0.3—0.4	6	4	5	10	8	5
0.4—0.5	9	8	7	15	16	16
0.5—0.6	17	18	17	19	18	20
0.6—0.7	9	9	7	16	20	20
0.7—0.8	5	8	9	20	16	15
0.8—0.9	1	10	8	7	10	12
0.9—1.0	0	13	14	15	11	11

3.3 Some upper bounds on P^{OB} after the LLL reduction

We have shown that the LLL reduction by Algorithm 2.2.1 can enhance the success probability of the Babai point. A natural question is how much is the enhancement? We will answer this question in this section.

If the LLL reduction has been computed by Algorithm 2.2.1, then we can easily obtain the ratio $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})/\Pr(\boldsymbol{x}^{\text{OB}} = \hat{\boldsymbol{x}})$ by using Theorem 3.1.1. If we only know the R-factor of the QR factorization of $\boldsymbol{A}$, usually it is impossible to know the ratio exactly. However, we will derive some bounds on $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$, which involve only the R-factor of the QR factorization of $\boldsymbol{A}$. From these bounds one can immediately obtain bounds on the ratio.

Before giving an upper bound on $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$, we give the following result, see, e.g., [69, Thm 6].

Lemma 3.3.1 *Let $\boldsymbol{R}$ be the R-factor of the QR factorization of $\boldsymbol{A}$ and let $\boldsymbol{R}^{(p)}$ be the upper triangular matrix after the p-th column permutation and triangularization in the LLL reduction process by Algorithm 2.2.1, then for $i = 1, 2, \ldots, n$*

$$\min\{r_{ii}, r_{i+1,i+1}, \ldots, r_{nn}\} \leq r_{ii}^{(p)} \leq \max\{r_{11}, r_{22}, \ldots, r_{ii}\}. \tag{3.19}$$

When the LLL reduction process finishes, the diagonal entries of the upper triangular matrix certainly satisfy (3.19). Then using the second inequality in (3.19) we obtain the following result from Theorem 3.1.1.

Theorem 3.3.1 *Suppose that the OILS problem* (2.4) *is transformed to the OILS problem* (2.18) *after the LLL reduction by Algorithm 2.2.1. The success probability*

of the Babai point for the OILS problem (2.18) *satisfies:*

$$\Pr(\boldsymbol{z}^{OB} = \hat{\boldsymbol{z}}) \leq \Pi_{i=1}^{n} \phi_\sigma(\gamma_i), \tag{3.20}$$

where $\gamma_i = \max\{r_{11}, r_{22}, \cdots, r_{ii}\}$.

In the following we give another upper bound on the success probability of the Babai point, which is invariant to the unimodular transformation to $\boldsymbol{R}$. The result was essentially obtained in [85], but our proof is much simpler.

Lemma 3.3.2 *Let* $\boldsymbol{R} \in \mathbb{R}^{n\times n}$ *be an upper triangular matrix with positive diagonal entries, then*

$$\prod_{i=1}^{n} \phi_\sigma(r_{ii}) \leq \phi_\sigma^n\Big(\Big(\prod_{i=1}^{n} r_{ii}\Big)^{1/n}\Big), \tag{3.21}$$

where the equality holds if and only if all the diagonal entries of $\boldsymbol{R}$ *are equal.*

Proof. Let $h(\xi) = \ln(\phi_\sigma(\exp(\xi))$ and $v_i = \ln r_{ii}$ for $i = 1, \ldots, n$. Define $v = \frac{1}{n}\sum_{i=1}^{n} v_i = \frac{1}{n}\ln(\prod_{i=1}^{n} r_{ii})$. To prove (3.21), it suffices to show that

$$\frac{1}{n}\sum_{i=1}^{n} h(v_i) \leq h(v). \tag{3.22}$$

It is easy to verify that

$$h''(\xi) = \frac{1}{2\sigma}\exp(\xi) g'\Big(\frac{1}{2\sigma}\exp(\xi)\Big),$$

where $g(\cdot)$ was defined in the proof of Lemma 3.2.1. According to the proof of Lemma 3.2.1, $g'(\zeta) < 0$ for $\zeta > 0$. Thus $h''(\xi) < 0$, i.e., $h(\xi)$ is a strictly concave function. Therefore, (3.22) must hold and the equality holds if and only if all v_i are equal, or equivalently all r_{ii} are equal. $\square$

Suppose that the OILS problem (2.4) is transformed to the OILS problem (2.18) after the LLL reduction by Algorithm 2.2.1. Then $\det(\bar{\boldsymbol{R}}) = \det(\boldsymbol{R}) = \prod_{i=1}^{n} r_{ii}$. Thus by Lemma 3.3.2 we have

$$\Pr(\boldsymbol{z}^{\mathrm{OB}} = \hat{\boldsymbol{z}}) = \prod_{i=1}^{n} \phi_\sigma(\bar{r}_{ii}) \leq \phi_\sigma^n\Big(\Big(\prod_{i=1}^{n} r_{ii}\Big)^{1/n}\Big). \tag{3.23}$$

The upper bound is reachable if and only if all the diagonal entries of $\bar{\boldsymbol{R}}$ are equal to $\det^{1/n}(\boldsymbol{R})$. If the gap between the largest diagonal entry and the smallest diagonal entry of $\bar{\boldsymbol{R}}$ is large, the upper bound in (3.23) will not be tight. In the following, we give an improved upper bound.

Theorem 3.3.2 *Under the same assumption as in Theorem 3.3.1, if there exist indices $i_1, i_2, \ldots, i_l$ such that*

$$M_k \leq m_{k+1}, \quad k = 1, \ldots, l, \tag{3.24}$$

where

$$M_k = \max\{r_{i_{k-1}+1,i_{k-1}+1}, r_{i_{k-1}+2,i_{k-1}+2}, \ldots, r_{i_k,i_k}\},$$
$$m_{k+1} = \min\{r_{i_k+1,i_k+1}, r_{i_k+2,i_k+2}, \ldots, r_{i_{k+1},i_{k+1}}\},$$

with $i_0 = 0$ and $i_{l+1} = n$, then

$$\Pr(\boldsymbol{z}^{OB} = \hat{\boldsymbol{z}}) \leq \prod_{k=1}^{l+1} \phi_\sigma^{i_k - i_{k-1}}(\nu_k) \leq \phi_\sigma^n(\nu), \tag{3.25}$$

where

$$\nu_k = \Big(\prod_{j=i_{k-1}+1}^{i_k} r_{jj}\Big)^{1/(i_k - i_{k-1})}, \quad \nu = \Big(\prod_{j=1}^{n} r_{jj}\Big)^{1/n}.$$

Proof. Partition $\boldsymbol{R}$ as follows:

$$\boldsymbol{R} = [\boldsymbol{R}_1, \boldsymbol{R}_2, \cdots, \boldsymbol{R}_{l+1}],$$

where the diagonal entries of $\boldsymbol{R}$ which are in block $\boldsymbol{R}_k \in \mathbb{R}^{n\times(i_k - i_{k-1})}$ are $r_{i_{k-1}+1,i_{k-1}+1}$, $r_{i_{k-1}+2,i_{k-1}+2}$, ..., r_{i_k,i_k} for $k = 1, \ldots, l+1$. The condition (3.24) is to ensure that in the LLL reduction process by Algorithm 2.2.1 there are no column permutations between $\boldsymbol{R}_k$s. Now we prove this claim. Suppose that Algorithm 2.2.1 has just finished the operations on $\boldsymbol{R}_2$ and is going to work on $\boldsymbol{R}_3$. At this moment, $[\boldsymbol{R}_1, \boldsymbol{R}_2]$ is LLL reduced. In the LLL reduction of $[\boldsymbol{R}_1, \boldsymbol{R}_2]$, no column permutation between the last column of $\boldsymbol{R}_1$ and and the first column of $\boldsymbol{R}_2$ occurred. In fact, by (3.19) in Lemma 3.3.1 and the inequality $M_1 \le m_2$ from (3.24), after a permutation, say the p-th permutation, in the LLL reduction of $[\boldsymbol{R}_1, \boldsymbol{R}_2]$ by Algorithm 2.2.1,

$$r^{(p)}_{i_1,i_1} \le \max\{r_{11}, \ldots, r_{i_1,i_1}\} \le \min\{r_{i_1+1,i_1+1}, \cdots, r_{i_2,i_2}\} \le r^{(p)}_{i_1+1,r_1+1}.$$

Thus for any δ satisfying $1/4 < \delta \le 1$, the Lovász condition (2.13) is satisfied for columns i_1 and $i_1 + 1$ and no permutation between these two columns would occur. Now the algorithm goes to work on the first column of $\boldsymbol{R}_3$. Again we can similarly show that no column permutation between the last column of $\boldsymbol{R}_2$ and and the first column of $\boldsymbol{R}_3$ will occur, so the algorithm will not go back to $\boldsymbol{R}_2$. The algorithm continues and whenever the current block is LLL reduced it goes to next block and will not come back to the previous block. Then by applying the result given in (3.23) for each block $\boldsymbol{R}_k$ we obtain the first inequality in (3.25). The second inequality in (3.25) is obtained immediately by applying Lemma 3.3.2. $\square$

If indices i_k for $k = 1, \ldots, l$ defined in Theorem 3.3.2 do not exist, we assume $l = 0$, then the first inequality in (3.25) still holds as its right hand side is just $\phi_\sigma^n(\nu)$.

We now show how to find these indices if they exist. It is easy to verify that (3.24) is equivalent to

$$\max\{M_1, \ldots, M_k\} \leq \min\{m_{k+1}, \ldots, m_{l+1}\} \tag{3.26}$$

for $k = 1, \ldots, l$. Define two vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^{n-1}$ as follows:

$$u_1 = r_{11}, u_i = \max\{r_{11}, \ldots, r_{ii}\} = \max\{u_{i-1}, r_{ii}\},$$

for $i = 2, \ldots, n-1$.

$$v_{n-1} = r_{nn}, v_i = \min\{r_{i+1,i+1}, \ldots, r_{nn}\} = \min\{r_{i+1,i+1}, v_{i+1}\},$$

Then (3.26) is equivalent to

$$u_{i_k} \leq v_{i_k}, \quad k = 1, \ldots, l.$$

Thus we can compare the entries of $\boldsymbol{u}$ and $\boldsymbol{v}$ from the first to the last to obtain all indices i_k. It is easy to observe that the total cost is $O(n)$.

Let β_1, β_2 and β_3 denote the three upper bounds on $\Pr(\boldsymbol{z}^{\mathrm{OB}} = \hat{\boldsymbol{z}})$ given in (3.20) and (3.25), respectively, i.e.,

$$\beta_1 = \Pi_{i=1}^n \phi_\sigma(\gamma_i), \quad \beta_2 = \prod_{k=1}^{l+1} \phi_\sigma^{i_k - i_{k-1}}(\nu_k), \quad \beta_3 = \phi_\sigma^n(\nu).$$

In the following, we first give some special examples to compare β_1, β_2 and β_3.

Example 3.3.1 *Let* $\boldsymbol{R} = \begin{bmatrix} 1/\eta & \times \\ 0 & \eta^2 \end{bmatrix}$, *where* $0 < \eta < 1$ *and* $\times$ *is any real number. Then*

$$\beta_1 = \phi_\sigma^2(1/\eta), \quad \beta_2 = \beta_3 = \phi_\sigma^2(\sqrt{\eta}).$$

By the definition of $\phi_\sigma(\zeta)$ *given in* (3.1), $\phi_\sigma(1/\eta) \to 1$ *and* $\phi_\sigma(\sqrt{\eta}) \to 0$ *when* $\eta \to 0$. *Thus, when* η *is very small,* β_2 *and* β_3 *are much sharper than* β_1.

Example 3.3.2 *Let*

$$\boldsymbol{R} = \begin{bmatrix} \eta/3 & \times & \times & \times \\ 0 & \eta & \times & \times \\ 0 & 0 & 1/\eta^3 & \times \\ 0 & 0 & 0 & \eta/2 \end{bmatrix}, \quad 0 < \eta < 1,$$

where $\times$ *is any real number. Then*

$$\beta_1 = \phi_\sigma(\eta/3)\phi_\sigma(\eta)\phi_\sigma^2(1/\eta^3),$$
$$\beta_2 = \phi_\sigma(\eta/3)\phi_\sigma^3\big(\sqrt[3]{1/(2\eta)}\big), \quad \beta_3 = \phi_\sigma^4(\sqrt[4]{1/6}).$$

From the definition of $\phi_\sigma(\zeta)$, *we see that when* $\eta \to 0$,

$$\beta_1 \to 0, \quad \beta_2 \to 0, \quad \beta_1/\beta_2 \to 0, \quad \beta_2/\beta_3 \to 0.$$

Therefore, when η *is very small,* β_1 *is much sharper than* β_2, *which is also much sharper than* β_3.

Now we use more general examples to compare the three upper bounds and also compare them with $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$. In additional to Cases 1 and 2 given in Section 3.2.2, we also tested the following case:

Case 3: $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$, where $\boldsymbol{Q}$ is a random orthogonal matrix obtained by the QR factorization of a random matrix generated by randn(n, n) and $\boldsymbol{R}$ is an $n \times n$ upper triangular matrix with r_{ii}^2 following the χ^2 distribution with freedom degree i and with r_{ij} $(j > i)$ following the normal distribution $\mathcal{N}(0, 1)$.

Case 3 is motivated by Case 1. In Case 1, the entries of the R-factor of the QR factorization of $\boldsymbol{A}$ have the same distributions as the entries of $\boldsymbol{R}$ in Case 3, except that the freedom degree for r_{ii}^2 is $n - i + 1$, see [66, p99].

In the numerical experiments, for a given n and for each case, we gave 200 runs to generate 200 different $\boldsymbol{A}$'s.

All the six tables given below display the average values of $\Pr(\boldsymbol{x}^{\text{OB}} = \hat{\boldsymbol{x}})$ (corresponding to QR), $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$ (corresponding to LLL with $\delta = 1$), β_1, β_2 and β_3. For each case, we give two tables. In the first table, n is fixed and σ varies, and in the second table, n varies and σ is fixed. In Tables 3–5 and 3–9 σ was fixed to be 0.4, while in Table 3–7 σ was fixed to be 0.1. We used different values of σ for these three tables so that $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$ is neither close to 0 nor close to 1, otherwise the bounds would not be much interesting.

For Case 1, from Tables 3–4 and 3–5 we observe that the upper bounds β_2 and β_3 are sharper than the upper bound β_1, especially when n is small, and the former are good approximations to $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$.

For Case 2, from Table 3–6 we observe that the upper bound β_1 is extremely loose when σ is large, and β_2 and β_3 are much sharper for all those σ. From Table 3–7 we see that when n becomes larger, the upper bounds β_2 and β_3 become worse, although they are still sharper than β_1. Tables 3–6-3–7 show that β_2 is equal to β_3. Actually it is indeed true.

For Case 3, from Tables 3–8 and 3–9 we observe that the success probability of the Babai point improves after the LLL reduction, but not as much as Cases 1 and 2. We also observe that β_2 is sharper than β_1, both are much sharper than β_3, and β_2 is a reasonable approximation to $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$.

Based on the numerical experiments and Theorem 3.3.2 we suggest taking $\min\{\beta_1, \beta_2\}$ as an upper bound on $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$ in practice.

Although the upper bound $\min\{\beta_1, \beta_2\}$ is a good approximation to $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})$ in the above numerical tests, we want to point out that this upper bound can be very loose. Here is a contrived example: Suppose all the off-diagonal entries of $\boldsymbol{R}$ in Example 3.3.2 are zero. Then

$$\Pr(\boldsymbol{x}^{\text{OB}} = \hat{\boldsymbol{x}}) = \Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}}) = \phi_\sigma(\eta/3)\phi_\sigma(\eta)\phi_\sigma(1/\eta^3)\phi_\sigma(\eta/2).$$

Thus, when $\eta \to 0$, $\Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}})/\min\{\beta_1, \beta_2\} \to 0$.

Table 3–4: Average P^{OB} and upper bounds over 200 runs for Case 1, $n = 20$

σ	QR	LLL	β_1	β_2	β_3
0.05	0.93242	1.00000	1.00000	1.00000	1.00000
0.10	0.84706	1.00000	1.00000	1.00000	1.00000
0.15	0.75362	0.99999	1.00000	1.00000	1.00000
0.20	0.66027	0.99966	1.00000	0.99984	0.99984
0.25	0.56905	0.99815	1.00000	0.99891	0.99891
0.30	0.48130	0.99289	1.00000	0.99645	0.99645
0.35	0.39864	0.97589	0.99999	0.98849	0.98849
0.40	0.32279	0.93432	0.99997	0.96319	0.96319

Table 3–5: Average P^{OB} and upper bounds over 200 runs for Case 1, $\sigma = 0.4$

n	QR	LLL	β_1	β_2	β_3
5	0.37181	0.52120	0.92083	0.55777	0.56437
10	0.33269	0.73310	0.99634	0.75146	0.75146
15	0.30324	0.87116	0.99967	0.89076	0.89076
20	0.32896	0.94211	0.99999	0.97004	0.97004
25	0.31439	0.95364	1.00000	0.98993	0.98993
30	0.32649	0.96961	1.00000	0.99752	0.99752
35	0.34107	0.97361	1.00000	0.99939	0.99939
40	0.32538	0.97579	1.00000	0.99980	0.99980

Table 3–6: Average P^{OB} and upper bounds over 200 runs for Case 2, $n = 20$

σ	QR	LLL	β_1	β_2	β_3
0.05	0.27379	1.00000	1.00000	1.00000	1.00000
0.10	0.01864	0.99490	1.00000	0.99939	0.99939
0.15	0.00161	0.82023	1.00000	0.89650	0.89650
0.20	0.00019	0.38963	1.00000	0.46930	0.46930
0.25	0.00003	0.10896	1.00000	0.13462	0.13462
0.30	0.00001	0.02248	1.00000	0.02738	0.02738
0.35	0.00000	0.00411	1.00000	0.00489	0.00489
0.40	0.00000	0.00074	1.00000	0.00086	0.00086

Table 3–7: Average P^{OB} and upper bounds over 200 runs for Case 2, $\sigma = 0.1$

n	QR	LLL	β_1	β_2	β_3
5	0.06157	0.75079	0.99984	0.83688	0.83688
10	0.05522	0.98875	1.00000	0.99344	0.99344
15	0.03069	0.99670	1.00000	0.99860	0.99860
20	0.01865	0.99486	1.00000	0.99939	0.99939
25	0.01149	0.97374	1.00000	0.99963	0.99963
30	0.00562	0.88945	1.00000	0.99973	0.99973
35	0.00324	0.76654	1.00000	0.99978	0.99978
40	0.00175	0.68623	1.00000	0.99981	0.99981

Table 3–8: Average P^{OB} and upper bounds over 200 runs for Case 3, $n = 20$

σ	QR	LLL	β_1	β_2	β_3
0.05	0.91780	0.92401	0.92450	0.92471	1.00000
0.10	0.85132	0.86372	0.87017	0.86856	1.00000
0.15	0.77339	0.79087	0.80902	0.79945	1.00000
0.20	0.68615	0.70836	0.74366	0.72379	1.00000
0.25	0.59499	0.62040	0.67610	0.64530	0.99986
0.30	0.50466	0.53153	0.60831	0.56704	0.99837
0.35	0.41858	0.44528	0.54164	0.49161	0.99038
0.40	0.33919	0.36432	0.47679	0.42031	0.96432

Table 3–9: Average P^{OB} and upper bounds over 200 runs for Case 3, $\sigma = 0.4$

n	QR	LLL	β_1	β_2	β_3
5	0.35057	0.37086	0.47342	0.38878	0.53300
10	0.35801	0.38542	0.49866	0.42252	0.75949
15	0.32379	0.35068	0.47865	0.40583	0.90613
20	0.34612	0.37149	0.49066	0.44551	0.96841
25	0.35252	0.37865	0.48907	0.44248	0.99232
30	0.32538	0.35542	0.46208	0.43224	0.99708
35	0.33183	0.35421	0.46524	0.42288	0.99933
40	0.32196	0.34759	0.45264	0.41220	0.99975

CHAPTER 4
Effects of Some QRP Reductions on the Success Probability of the Box-Constrained Babai Estimator and Solving a Conjecture

In this chapter, we study the box-constrained linear model (1.3), which arise from some applications, including wireless communications, for more details, see, e.g., [65, 2, 21]. Throughout this chapter we assume that $\hat{\boldsymbol{x}}$ is random and uniformly distributed over the box $\mathcal{B}$. This assumption is often made for MIMO applications, see, e.g., [41].

We showed in Chapter 3 that the success probability P^{OB} of the ordinary Babai estimator increases after applying the LLL reduction or the LLL-P column permutation strategy, but it may decrease after applying the SQRD or the V-BLAST permutation strategies. Therefore, a natural question is to ask whether this can be extended to the box-constrained case.

In this chapter, we will present a formula for the success probability P^{BB} of the box-constrained Babai estimator $\boldsymbol{x}^{\text{BB}}$ (2.8) and a formula for the success probability P^{OB} of the ordinary Babai estimator $\boldsymbol{x}^{\text{OB}}$ (2.7) which is obtained via ignoring the box constraints. Some properties of P^{BB} and P^{OB} and the relationships between them will also be given. Then we will investigate the effect of the LLL-P column permutation strategy on P^{BB}. We will show that P^{BB} increases under a condition. Surprisingly, we will also show that it decreases after LLL-P is applied under an opposite condition. Roughly speaking the two opposite conditions are that the noise

standard deviation σ in (1.3a) are relatively small and large, respectively. This is different from the ordinary case, where P^{OB} always increases after the LLL-P strategy is applied. Although our theoretical results for LLL-P cannot be extended to SQRD and V-BLAST, our numerical tests indicate that under the two conditions, often (not always) P^{BB} increases and decreases, respectively, after applying SQRD or V-BLAST. Explanations will be given for these phenomenons. These suggest that before we apply LLL-P, SQRD or V-BLAST we should check the corresponding conditions. Moreover, we will give a bound on P^{BB}, which is column permutation invariant. It is interesting that the bound is an upper bound under the small noise condition we just mentioned and becomes a lower bound under the opposite condition. These results were presented in [90].

In [58], the authors made a conjecture, based on which a stopping criterion for the search process was proposed to reduce the computational cost of solving the BILS problem (1.4). The conjecture is related to P^{OB}. We will first give a counter example to show that the conjecture does not always hold and then show it holds under some conditions. Based on some new theoretical results, we will also propose a modified stopping criterion, which is more reliable, for solving the BILS problem (1.4). These results were presented in [90].

4.1 Success probabilities of the Babai estimators

If we do not take the box-constraint into account, we can get the ordinary Babai point $\boldsymbol{x}^{\text{OB}}$ (see (2.7)). In this section, we will respectively give formulas for

the success probability $P^{\rm BB}$ of $\boldsymbol{x}^{\rm BB}$ (see (2.8)) and $P^{\rm OB}$ of $\boldsymbol{x}^{\rm OB}$ for the reduced box-constrained liner model (2.5). Then, we will investigate the relationships between $P^{\rm BB}$ and $P^{\rm OB}$ and their properties.

Theorem 4.1.1 *Suppose that in the linear model* (1.3) $\hat{\boldsymbol{x}}$ *is uniformly distributed over the constraint box* $\mathcal{B}$ *and* $\hat{\boldsymbol{x}}$ *and* $\boldsymbol{v}$ *are independent. Suppose that the linear model* (1.3) *is transformed to the linear model* (2.5) *through the QR factorization* (2.1). *Then*

$$P^{BB} \equiv \Pr(\boldsymbol{x}^{BB} = \hat{\boldsymbol{x}}) = \prod_{i=1}^{n} \Big[\frac{1}{u_i - l_i + 1} + \frac{u_i - l_i}{u_i - l_i + 1}\phi_\sigma(r_{ii})\Big], \tag{4.1}$$

$$P^{OB} \equiv \Pr(\boldsymbol{x}^{OB} = \hat{\boldsymbol{x}}) = \prod_{i=1}^{n} \phi_\sigma(r_{ii}), \tag{4.2}$$

where $\phi_\sigma(\zeta)$ *is defined as in* (3.1).

Proof. Since the random vectors $\hat{\boldsymbol{x}}$ and $\boldsymbol{v}$ in (1.3) are independent, $\hat{\boldsymbol{x}}$ and $\tilde{\boldsymbol{v}}$ in (2.5) are also independent. From (2.5a),

$$\tilde{y}_i = r_{ii}\hat{x}_i + \sum_{j=i+1}^{n} r_{ij}\hat{x}_j + \tilde{v}_i.$$

Then from (2.8), we obtain

$$c_i^{\rm BB} = \hat{x}_i + \sum_{j=i+1}^{n} \frac{r_{ij}}{r_{ii}}(\hat{x}_j - x_j^{\rm BB}) + \frac{\tilde{v}_i}{r_{ii}}, \quad i = n, n-1, \ldots, 1. \tag{4.3}$$

Therefore, if $x_{i+1}^{\rm BB} = \hat{x}_{i+1}, \cdots, x_n^{\rm BB} = \hat{x}_n$ and $\hat{x}_i$ is fixed,

$$c_i^{\rm BB} \sim \mathcal{N}(\hat{x}_i, \sigma^2/r_{ii}^2). \tag{4.4}$$

To simplify notation, denote event

$$E_i = (x_i^{\mathrm{BB}} = \hat{x}_i, \ldots, x_n^{\mathrm{BB}} = \hat{x}_n), \quad i = 1, \ldots, n.$$

Then by the chain rule of conditional probabilities,

$$P^{\mathrm{BB}} = \Pr(E_1) = \prod_{i=1}^{n} \Pr(x_i^{\mathrm{BB}} = \hat{x}_i | E_{i+1}), \tag{4.5}$$

where E_{n+1} is the sample space Ω, so $\Pr(x_n^{\mathrm{BB}} = \hat{x}_n | E_{n+1}) = \Pr(x_n^{\mathrm{BB}} = \hat{x}_n)$.

In the following we will use this fact: if A, B and C are three events and A and C are independent, then

$$\Pr(A, B|C) = \Pr(A) \Pr(B|A, C). \tag{4.6}$$

This can easily be verified.

Then, using (4.6), we obtain

$$\begin{aligned}
\Pr(x_i^{\mathrm{BB}} = \hat{x}_i \,|\, E_{i+1}) &= \Pr(\hat{x}_i = l_i, c_i^{\mathrm{BB}} \le l_i + 1/2 \,|\, E_{i+1}) \qquad (4.7)\\
&\quad + \Pr(l_i < \hat{x}_i < u_i, \hat{x}_i - 1/2 \le c_i^{\mathrm{BB}} < \hat{x}_i + 1/2 \,|\, E_{i+1})\\
&\quad + \Pr(\hat{x}_i = u_i, c_i^{\mathrm{BB}} \ge u_i - 1/2 \,|\, E_{i+1})\\
&= \Pr(\hat{x}_i = l_i) \Pr(c_i^{\mathrm{BB}} \le l_i + 1/2 \,|\, \hat{x}_i = l_i, E_{i+1}) \qquad (4.8)\\
&\quad + \Pr(l_i < \hat{x}_i < u_i) \Pr(\hat{x}_i - 1/2 \le c_i^{\mathrm{BB}} \le \hat{x}_i + 1/2 \,|\, l_i < \hat{x}_i < u_i, E_{i+1})\\
&\quad + \Pr(\hat{x}_i = u_i) \Pr(c_i^{\mathrm{BB}} \ge u_i - 1/2 \,|\, \hat{x}_i = u_i, E_{i+1})
\end{aligned}$$

where in deriving the second equality we used the independency of relevant events, which can be observed from (4.3) and (2.8).

Since $\hat{\boldsymbol{x}}$ is uniformly distributed over the box $\mathcal{B}$, for the first factors of the three terms on the right-hand side of (4.8), we have

$$\Pr(\hat{x}_i = l_i) = \frac{1}{u_i - l_i + 1}, \quad \Pr(l_i < \hat{x}_i < u_i) = \frac{u_i - l_i - 1}{u_i - l_i + 1}, \quad \Pr(\hat{x}_i = u_i) = \frac{1}{u_i - l_i + 1}.$$

By (4.4), for the second factors of these three terms, we have

$$\begin{aligned}\Pr(c_i^{\mathrm{BB}} \le l_i + 1/2 \,|\, \hat{x}_i = l_i, E_{i+1}) &= \frac{1}{\sqrt{2\pi(\frac{\sigma}{r_{ii}})^2}} \int_{-\infty}^{l_i+\frac{1}{2}} \exp\Big(-\frac{(t-l_i)^2}{2(\frac{\sigma}{r_{ii}})^2}\Big) dt \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\frac{r_{ii}}{2\sigma}} \exp\big(-\frac{t^2}{2}\big) dt = \frac{1}{2}[1 + \phi_\sigma(r_{ii})],\end{aligned}$$

$$\begin{aligned}\Pr(\hat{x}_i - 1/2 \le c_i^{\mathrm{BB}} \le \hat{x}_i + 1/2 \,|\, l_i < \hat{x}_i < u_i, E_{i+1}) &= \frac{1}{\sqrt{2\pi(\frac{\sigma}{r_{ii}})^2}} \int_{\hat{x}_i-\frac{1}{2}}^{\hat{x}_i+\frac{1}{2}} \exp\Big(-\frac{(t-\hat{x}_i)^2}{2(\frac{\sigma}{r_{ii}})^2}\Big) dt \\ &= \phi_\sigma(r_{ii}),\end{aligned}$$

$$\begin{aligned}\Pr(c_i^{\mathrm{BB}} \ge u_i - 1/2 \,|\, \hat{x}_i = u_i, E_{i+1}) &= \frac{1}{\sqrt{2\pi(\frac{\sigma}{r_{ii}})^2}} \int_{u_i-\frac{1}{2}}^{\infty} \exp\Big(-\frac{(t-u_i)^2}{2(\frac{\sigma}{r_{ii}})^2}\Big) dt \\ &= \frac{1}{2}[1 + \phi_\sigma(r_{ii})].\end{aligned}$$

Combining the equalities above, from (4.8) we obtain

$$\begin{aligned}\Pr(x_i^{\mathrm{BB}} = \hat{x}_i \,|\, E_{i+1}) = {} & \frac{1}{2(u_i - l_i + 1)}[1 + \phi_\sigma(r_{ii})] + \frac{u_i - l_i - 1}{u_i - l_i + 1}\phi_\sigma(r_{ii}) \\ & + \frac{1}{2(u_i - l_i + 1)}[1 + \phi_\sigma(r_{ii})] = \frac{1}{u_i - l_i + 1} + \frac{u_i - l_i}{u_i - l_i + 1}\phi_\sigma(r_{ii})\end{aligned}$$

which, with (4.5), gives (4.1).

Now we consider the success probability of the ordinary Babai estimator $\boldsymbol{x}^{\mathrm{OB}}$. Everything in the first three paragraphs of this proof still holds if we replace each

superscript BB by OB. But we need to make more significant changes to the last two paragraphs. We change (4.7) and (4.8) as follows:

$$\begin{aligned}\Pr(x_i^{\mathrm{OB}} = \hat{x}_i \,|\, E_{i+1}) &= \Pr(l_i \le \hat{x}_i \le u_i, \hat{x}_i - 1/2 \le c_i^{\mathrm{OB}} \le \hat{x}_i + 1/2 \,|\, E_{i+1}) \\ &= \Pr(l_i \le \hat{x}_i \le u_i) \Pr(\hat{x}_i - 1/2 \le c_i^{\mathrm{OB}} \le \hat{x}_i + 1/2 \,|\, l_i \le \hat{x}_i \le u_i, E_{i+1}).\end{aligned}$$

Here

$$\begin{aligned}&\Pr(l_i \le \hat{x}_i \le u_i) = 1, \\ &\Pr(\hat{x}_i - 1/2 \le c_i^{\mathrm{OB}} < \hat{x}_i + 1/2 \,|\, l_i \le \hat{x}_i \le u_i, E_{i+1}) = \phi_\sigma(r_{ii}).\end{aligned}$$

Thus

$$\Pr(x_i^{\mathrm{OB}} = \hat{x}_i \,|\, E_{i+1}) = \phi_\sigma(r_{ii}).$$

Then (4.2) follows from (4.5) with each superscript BB replaced by OB. □

From the proof for (4.2), we observe that the formula holds no matter what distribution of $\hat{\boldsymbol{x}}$ is over the box $\mathcal{B}$. Furthermore, the formula is identical to the one for the success probability of the ordinary Babai estimator $\boldsymbol{x}^{\mathrm{OB}}$ when $\hat{\boldsymbol{x}}$ in the linear model (1.1) is deterministic and is not subject to any box constraint, see Theorem 3.1.1.

The following result shows the relation between P^{BB} and P^{OB}.

Corollary 4.1.1 *Under the same assumption as in Theorem 4.1.1,*

$$P^{OB} \le P^{BB}, \tag{4.9}$$

$$\lim_{\forall i, u_i - l_i \to \infty} P^{BB} = P^{OB}. \tag{4.10}$$

Proof. Note that $\phi_\sigma(r_{ii}) \leq 1$. Thus

$$\phi_\sigma(r_{ii}) = \frac{1}{u_i - l_i + 1}\phi_\sigma(r_{ii}) + \frac{u_i - l_i}{u_i - l_i + 1}\phi_\sigma(r_{ii}) \leq \frac{1}{u_i - l_i + 1} + \frac{u_i - l_i}{u_i - l_i + 1}\phi_\sigma(r_{ii}).$$

Then (4.9) follows from Theorem 4.1.1. Obviously the equality (4.10) holds. □

Corollary 4.1.2 *Under the same assumption as in Theorem 4.1.1, P^{BB} and P^{OB} increase when σ decreases and*

$$\lim_{\sigma\to 0} P^{BB} = \lim_{\sigma\to 0} P^{OB} = 1.$$

Proof. For a given ζ, when σ decreases $\phi_\sigma(\zeta)$ increases and $\lim_{\sigma\to 0}\phi_\sigma(\zeta) = 1$. Then from (4.1) and (4.2), we immediately see that the corollary holds.

4.2 Effects of LLL-P, V-BLAST and SQRD on P^{BB}

Similar to (2.8), for the reduced box-constrained linear model (2.23), we can define its corresponding Babai point $\boldsymbol{z}^{\text{BB}}$ and use it as an estimator of $\hat{\boldsymbol{z}}$, which is equal to $\boldsymbol{P}^T\hat{\boldsymbol{x}}$, or equivalently we use $\boldsymbol{P}\boldsymbol{z}^{\text{BB}}$ to estimate $\hat{\boldsymbol{x}}$.

In this section, we will investigate how the LLL-P, SQRD and V-BLAST column permutation strategies affect the success probability P^{BB} of the box-constrained Babai estimator $\boldsymbol{x}^{\text{BB}}$.

4.2.1 Effect of LLL-P on P^{BB}

The LLL-P strategy involves a sequence of permutations of two consecutive columns of $\boldsymbol{R}$. To investigate how LLL-P affects P^{BB}, we look at one column permutation first. Suppose that $\delta\, r_{k-1,k-1}^2 > r_{k-1,k}^2 + r_{kk}^2$ for some k for the $\boldsymbol{R}$ matrix in the linear model (2.5). After the permutation of columns $k-1$ and k, $\boldsymbol{R}$ becomes $\bar{\boldsymbol{R}} = \boldsymbol{G}_{k-1,k}^T\boldsymbol{R}\boldsymbol{P}_{k-1,k}$ (see (2.14)). Then with the transformations given in

(2.22), where $\bar{\boldsymbol{Q}} = \boldsymbol{G}_{k-1,k}$ and $\boldsymbol{P} = \boldsymbol{P}_{k-1,k}$, the linear model (2.5) is transformed to the linear model (2.23). We will compare $\Pr(\boldsymbol{x}^{\text{BB}} = \hat{\boldsymbol{x}})$ and $\Pr(\boldsymbol{z}^{\text{BB}} = \hat{\boldsymbol{z}})$ in this subsection.

To prove our results, we need the following lemmas.

Lemma 4.2.1 *Given $\alpha \geq 0$, define*

$$f(\zeta, \alpha) = (1 - \zeta^2)\Big(\alpha + \int_0^{\zeta} \exp\big(-\frac{t^2}{2}\big)dt\Big) - \zeta \exp\big(-\frac{\zeta^2}{2}\big), \quad \zeta \geq 0. \tag{4.11}$$

Then, $f(\zeta, \alpha)$ is a strictly decreasing function of ζ and has a unique zero $r(\alpha)$, i.e.,

$$f(r(\alpha), \alpha) = 0. \tag{4.12}$$

When $\zeta > r(\alpha)$, $f(\zeta, \alpha) < 0$ and when $\zeta < r(\alpha) \neq 0$, $f(\zeta, \alpha) > 0$. Furthermore, $0 \leq r(\alpha) < 1$, where the first inequality becomes an equality if and only if $\alpha = 0$, and $r(\alpha)$ is a strictly increasing function of α.

Proof. By a simple calculation, we obtain

$$\frac{\partial f(\zeta, \alpha)}{\partial \zeta} = -2\zeta\Big(\alpha + \int_0^{\zeta} \exp(-\frac{t^2}{2})dt\Big).$$

Thus, for any $\zeta \geq 0$ and $\alpha \geq 0$, $\partial f(\zeta, \alpha)/\partial \zeta \leq 0$, where the equality holds if and only $\zeta = 0$. Therefore, $f(\zeta, \alpha)$ is a strictly decreasing function of ζ.

Since $f(0, \alpha) = \alpha \geq 0$ and $f(1, \alpha) < 0$, there exists a unique $r(\alpha)$ such that (4.12) holds and $0 \leq r(\alpha) < 1$. Obviously $r(0) = 0$. Since $f(\zeta, \alpha)$ is strictly decreasing with respect to ζ, when $\zeta > r(\alpha)$, $f(\zeta, \alpha) < 0$ and when $\zeta < r(\alpha)$, $f(\zeta, \alpha) > 0$.

From (4.12), we obtain that for $\alpha > 0$,

$$r'(\alpha) = \frac{(1-(r(\alpha))^2)^2}{2(r(\alpha))^2 \exp\left(-(r(\alpha))^2/2\right)} > 0.$$

Thus, $r(\alpha)$ is a strictly increasing function of α. $\square$

Given α, we can easily solve (4.12) by a numerical method, e.g., the Newton method, to find $r(\alpha)$.

Lemma 4.2.2 *Given $\alpha \geq 0$ and $\beta > 0$, define*

$$\begin{aligned} g(\zeta,\alpha,\beta) = \alpha\Big(\int_0^{\zeta} \exp(-\frac{t^2}{2})dt + \int_0^{\beta/\zeta} \exp(-\frac{t^2}{2})dt\Big) \\ + \int_0^{\zeta} \exp(-\frac{t^2}{2})dt \int_0^{\beta/\zeta} \exp(-\frac{t^2}{2})dt, \quad \zeta > 0. \end{aligned} \tag{4.13}$$

Then, when

$$\min\{\sqrt{\beta}, \beta/r(\alpha)\} \leq \zeta < \max\{\sqrt{\beta}, \beta/r(\alpha)\}, \tag{4.14}$$

where $r(\alpha)$ is defined in Lemma 4.2.1 and $\beta/r(\alpha) \equiv \infty$ if $\alpha = 0$, $g(\zeta,\alpha,\beta)$ is a strictly decreasing function of ζ.

Proof. From the definition of g, we obtain

$$\begin{aligned} \frac{\partial g(\zeta,\alpha,\beta)}{\partial \zeta} &= \alpha\Big(\exp(-\frac{1}{2}\zeta^2) - \frac{\beta}{\zeta^2}\exp\left(-\frac{1}{2}\frac{\beta^2}{\zeta^2}\right)\Big) + \exp(-\frac{1}{2}\zeta^2)\int_0^{\beta/\zeta}\exp(-\frac{t^2}{2})dt \\ &\quad - \frac{\beta}{\zeta^2}\exp\left(-\frac{1}{2}\frac{\beta^2}{\zeta^2}\right)\int_0^{\zeta}\exp(-\frac{t^2}{2})dt \\ &= \frac{1}{\zeta}\Big[\zeta\exp(-\frac{1}{2}\zeta^2)\Big(\alpha + \int_0^{\beta/\zeta}\exp(-\frac{t^2}{2})dt\Big) \\ &\quad - \frac{\beta}{\zeta}\exp(-\frac{1}{2}\frac{\beta^2}{\zeta^2})\Big(\alpha + \int_0^{\zeta}\exp(-\frac{t^2}{2})dt\Big)\Big] \\ &= \frac{1}{\zeta}\Big(\alpha + \int_0^{\beta/\zeta}\exp(-\frac{t^2}{2})dt\Big)\Big(\alpha + \int_0^{\zeta}\exp(-\frac{t^2}{2})dt\Big)\left[h(\zeta,\alpha) - h(\beta/\zeta,\alpha)\right], \end{aligned}$$

where

$$h(\zeta,\alpha)=\frac{\zeta\exp(-\frac{\zeta^2}{2})}{\alpha+\int_0^{\zeta}\exp(-\frac{t^2}{2})dt}.$$

It is easy to see that in order to show the result, we need only to show $h(\zeta,\alpha)-h(\beta/\zeta,\alpha)<0$ under the condition (4.14) with $\zeta\neq\beta/\zeta$.

By some simple calculations and (4.11), we have

$$\frac{\partial h(\zeta,\alpha)}{\partial\zeta}=\frac{\exp(-\frac{\zeta^2}{2})}{\left(\alpha+\int_0^{\zeta}\exp(-\frac{t^2}{2})dt\right)^2}\times f(\zeta,\alpha). \tag{4.15}$$

Now we assume that ζ satisfies (4.14) with $\zeta\neq\beta/\zeta$. If $\sqrt{\beta}<\beta/r(\alpha)$, then $\zeta>\beta/\zeta>r(\alpha)$ and by Lemma 4.2.1, $\partial h(\zeta,\alpha)/\partial\zeta<0$, i.e., $h(\zeta,\alpha)$ is a strictly deceasing function of ζ, thus $h(\zeta,\alpha)-h(\beta/\zeta,\alpha)<0$. If $\sqrt{\beta}>\beta/r(\alpha)$, then $\zeta<\beta/\zeta<r(\alpha)$ and by Lemma 4.2.1, $\partial h(\zeta,\alpha)/\partial\zeta>0$, i.e., $h(\zeta,\alpha)$ is a strictly increasing function of ζ, thus again $h(\zeta,\alpha)-h(\beta/\zeta,\alpha)<0$. $\square$

With the above lemmas, we can show how the success probability of the box-constrained Babai estimator changes after two consecutive columns are swapped when the LLL-P strategy is applied.

Theorem 4.2.1 *Suppose that in the linear model* (1.3) *the box* $\mathcal{B}$ *is a cube with edge length of* d, $\hat{\boldsymbol{x}}$ *is uniformly distributed over* $\mathcal{B}$, *and* $\hat{\boldsymbol{x}}$ *and* $\boldsymbol{v}$ *are independent. Suppose that the linear model* (1.3) *is transformed to the linear model* (2.5) *through the QR factorization* (2.1) *and* $\delta\, r_{k-1,k-1}^2>r_{k-1,k}^2+r_{kk}^2$. *After the permutation of columns* $k-1$ *and* k *of* $\boldsymbol{R}$ *(see* (2.20)*), the linear model* (2.5) *is transformed to the linear model* (2.23).

1. *If* $r_{kk}/2\sigma \geq r(\sqrt{2\pi}/(2d))$*, where* $r(\cdot)$ *is defined in Lemma 4.2.1, then after the permutation, the success probability of the box-constrained Babai estimator increases, i.e.,*

$$\Pr(\boldsymbol{x}^{BB} = \hat{\boldsymbol{x}}) \leq \Pr(\boldsymbol{z}^{BB} = \hat{\boldsymbol{z}}). \tag{4.16}$$

2. *If* $r_{k-1,k-1}/2\sigma \leq r(\sqrt{2\pi}/(2d))$*, then after the permutation, the success probability of the box-constrained Babai estimator decreases, i.e.,*

$$\Pr(\boldsymbol{x}^{BB} = \hat{\boldsymbol{x}}) \geq \Pr(\boldsymbol{z}^{BB} = \hat{\boldsymbol{z}}). \tag{4.17}$$

Furthermore, the equality in each of (4.16) *and* (4.17) *holds if and only if* $r_{k-1,k} = 0$*.*

Proof. When $r_{k-1,k} = 0$, by Theorem 4.1.1, we see the equalities in (4.16) and (4.17) hold. In the following we assume $r_{k-1,k} \neq 0$ and show the strict inequalities in (4.16) and (4.17) hold.

Define

$$\beta \equiv \frac{r_{k-1,k-1}}{2\sigma}\frac{r_{kk}}{2\sigma} = \frac{\bar{r}_{k-1,k-1}}{2\sigma}\frac{\bar{r}_{kk}}{2\sigma}, \tag{4.18}$$

where for the second equality, see (2.15). Using $\delta\, r_{k-1,k-1}^2 > r_{k-1,k}^2 + r_{kk}^2$ and the equalities in (2.15), we can easily verify that

$$\sqrt{\beta} \leq \max\left\{\frac{\bar{r}_{k-1,k-1}}{2\sigma}, \frac{\bar{r}_{kk}}{2\sigma}\right\} < \max\left\{\frac{r_{k-1,k-1}}{2\sigma}, \frac{r_{kk}}{2\sigma}\right\} = \frac{r_{k-1,k-1}}{2\sigma} = \frac{\beta}{r_{kk}/(2\sigma)}, \tag{4.19}$$

$$\frac{\beta}{r_{k-1,k-1}/(2\sigma)} = \frac{r_{kk}}{2\sigma} = \min\left\{\frac{r_{k-1,k-1}}{2\sigma}, \frac{r_{kk}}{2\sigma}\right\} < \min\left\{\frac{\bar{r}_{k-1,k-1}}{2\sigma}, \frac{\bar{r}_{kk}}{2\sigma}\right\} \leq \sqrt{\beta}. \tag{4.20}$$

Now we prove part 1. Note that after the permutation, $r_{k-1,k-1}$ and r_{kk} change, but other diagonal entries of $\boldsymbol{R}$ do not change. Then by Theorem 4.1.1, we can easily

observe that (4.16) is equivalent to

$$\begin{aligned}&\Big[\frac{1}{d+1}+\frac{d}{d+1}\phi_\sigma(r_{k-1,k-1})\Big]\Big[\frac{1}{d+1}+\frac{d}{d+1}\phi_\sigma(r_{kk})\Big]\\ \le&\Big[\frac{1}{d+1}+\frac{d}{d+1}\phi_\sigma(\bar{r}_{k-1,k-1})\Big]\Big[\frac{1}{d+1}+\frac{d}{d+1}\phi_\sigma(\bar{r}_{kk})\Big].\end{aligned}\tag{4.21}$$

By the definition of ϕ_σ in (3.1) and the definition of g in (4.13), we can easily verify that (4.21) is equivalent to

$$g\Big(\max\Big\{\frac{r_{k-1,k-1}}{2\sigma},\frac{r_{kk}}{2\sigma}\Big\},\frac{\sqrt{2\pi}}{2d},\beta\Big)\le g\Big(\max\Big\{\frac{\bar{r}_{k-1,k-1}}{2\sigma},\frac{\bar{r}_{kk}}{2\sigma}\Big\},\frac{\sqrt{2\pi}}{2d},\beta\Big).\tag{4.22}$$

If $r_{kk}/2\sigma \ge r(\sqrt{2\pi}/(2d))$, then the right-hand side of the last equality in (4.19) satisfies

$$\frac{\beta}{r_{kk}/(2\sigma)}\le\frac{\beta}{r(\sqrt{2\pi}/(2d))}.\tag{4.23}$$

Then by combining (4.19) and (4.23) and applying Lemma 4.2.2 we can conclude that the strict inequality in (4.22) holds.

The proof for part 2 is similar. The inequality (4.17) is equivalent to

$$g\Big(\min\Big\{\frac{r_{k-1,k-1}}{2\sigma},\frac{r_{kk}}{2\sigma}\Big\},\frac{\sqrt{2\pi}}{2d},\beta\Big)\ge g\Big(\min\Big\{\frac{\bar{r}_{k-1,k-1}}{2\sigma},\frac{\bar{r}_{kk}}{2\sigma}\Big\},\frac{\sqrt{2\pi}}{2d},\beta\Big).\tag{4.24}$$

If $r_{k-1,k-1}/2\sigma \le r(\sqrt{2\pi}/(2d))$, then the left-hand side of the first equality in (4.20) satisfies

$$\frac{\beta}{r(\sqrt{2\pi}/(2d))}\le\frac{\beta}{r_{k-1,k-1}/(2\sigma)}.\tag{4.25}$$

Then by combining (4.20) and (4.25) and applying Lemma 4.2.2 we can conclude that the strict inequality in (4.24) holds. □

We make a few remarks about Theorem 4.2.1.

Remark 4.2.1 *In the theorem, $\mathcal{B}$ is assumed to be a cube, not a more general box. This restriction simplified the theoretical analysis. In practical applications, such as in communications, indeed $\mathcal{B}$ is often a cube.*

Remark 4.2.2 *After the permutation, the larger one of $r_{k-1,k-1}$ and r_{kk} becomes smaller (see* (4.19)*) and the smaller one becomes larger (see* (4.20)*), so the gap between $r_{k-1,k-1}$ and r_{kk} becomes smaller. This makes P^{BB} increase under the condition $r_{kk}/2\sigma \geq r(\sqrt{2\pi}/(2d))$ or decrease under the condition $r_{k-1,k-1}/2\sigma \leq r(\sqrt{2\pi}/(2d))$. It is natural to ask for fixed $r_{k-1,k-1}$ and r_{kk} when will P^{BB} increase most or decrease most after the permutation under the the corresponding conditions? From the proof we observe that P^{BB} will become maximal when the first inequality in* (4.19) *becomes an equality or minimal when the last inequality in* (4.20) *becomes an equality under the corresponding conditions. Either of the two equalities holds if and only if $\bar{r}_{k-1,k-1} = \bar{r}_{kk}$, which is equivalent to $r_{k-1,k}^2 + r_{kk}^2 = r_{k-1,k-1}r_{kk}$ by* (2.15).

Remark 4.2.3 *The case where $r_{kk}/2\sigma < r(\sqrt{2\pi}/(2d)) < r_{k-1,k-1}/2\sigma$ is not covered by the theorem. For this case, P^{BB} may increase or decrease after the permutation, for more details, see the simulations in Sec. 4.2.4.*

Based on Theorem 4.2.1, we can establish the following general result for the LLL-P strategy.

Theorem 4.2.2 *Suppose that in the linear model* (1.3) *the box $\mathcal{B}$ is a cube with edge length of d, $\hat{\boldsymbol{x}}$ is uniformly distributed over $\mathcal{B}$, and $\hat{\boldsymbol{x}}$ and $\boldsymbol{v}$ are independent. Suppose that the linear model* (1.3) *is first transformed to the linear model* (2.5) *through the QR factorization* (2.1) *and then to the new linear model* (2.23) *through the QR factorization* (2.20) *where the LLL-P strategy is used for column permutations.*

1. *If the diagonal entries of* $\boldsymbol{R}$ *in* (2.3) *satisfies*

$$\min_i r_{ii}/(2\sigma) \geq r(\sqrt{2\pi}/(2d)), \tag{4.26}$$

where $r(\cdot)$ *is defined in Lemma 4.2.1, then*

$$\Pr(\boldsymbol{x}^{BB} = \hat{\boldsymbol{x}}) \leq \Pr(\boldsymbol{z}^{BB} = \hat{\boldsymbol{z}}). \tag{4.27}$$

2. *If the diagonal entries of* $\boldsymbol{R}$ *in* (2.3) *satisfies*

$$\max_i r_{ii}/(2\sigma) \leq r(\sqrt{2\pi}/(2d)), \tag{4.28}$$

then

$$\Pr(\boldsymbol{x}^{BB} = \hat{\boldsymbol{x}}) \geq \Pr(\boldsymbol{z}^{BB} = \hat{\boldsymbol{z}}). \tag{4.29}$$

And the equalities in (4.27) *and* (4.29) *hold if and only if no column permutation occurs in the process or whenever two consecutive columns, say* $k-1$ *and* k*, are permuted,* $r_{k-1,k} = 0$*.*

Proof. It is easy to show that after each column permutation, the smaller one of the two diagonal entries of $\boldsymbol{R}$ involved in the permutation either keeps unchanged (the involved super-diagonal entry is 0 in this case) or strictly increases, while the larger one either keeps unchanged or strictly decreases (see (4.19) and (4.20)). Thus, after each column permutation, the minimum of the diagonal entries of $\boldsymbol{R}$ either keeps unchanged or strictly increases and the maximum either keeps unchanged or strictly decreases, so the diagonal entries of any upper triangular $\bar{\boldsymbol{R}}$ produced after a column permutation satisfies $\min_i r_{ii} \leq \bar{r}_{kk} \leq \max_i r_{ii}$ for all $k = 1, \ldots, n$. Then the conclusions follows from Theorem 4.2.1. $\square$

We make some remarks about Theorem 4.2.2.

Remark 4.2.4 *The quantity* $r(\sqrt{2\pi}/(2d))$ *is involved in the conditions. To get some idea about how large it is, we compute it for a few different* $d = 2^k - 1$. *For* $k = 1, 2, 3, 4, 5$, *the corresponding values of* r *are 0.83992, 0.69666, 0.57157, 0.46475, 0.37525. They are decreasing with* k *as proved in Lemma 4.2.1. As* $d \to \infty$, $r(\sqrt{2\pi}/(2d)) \to r(0) = 0$. *Thus, when* d *is large enough, the condition* (4.26) *will be satisfied. By Corollary 4.1.1, taking the limit as* $d \to \infty$ *on both sides of* (4.27), *we obtain the following result given in Theorem 4.2.2:*

$$\Pr(\boldsymbol{x}^{OB} = \hat{\boldsymbol{x}}) \leq \Pr(\boldsymbol{z}^{OB} = \hat{\boldsymbol{z}}),$$

i.e., LLL-P always increases the success probability of the ordinary Babai estimator.

Remark 4.2.5 *The two conditions* (4.26) *and* (4.28) *also involve the noise standard deviation* σ. *When* σ *is small,* (4.26) *is likely to hold, so applying LLL-P is likely to increase* P^{BB}, *and when* σ *is large,* (4.28) *is likely to hold, so applying LLL-P is likely to decrease* P^{BB}. *It is quite surprising that when* σ *is large enough applying LLL-P will decrease* P^{BB}. *Thus, before applying LLL-P, one needs to check the conditions* (4.26) *and* (4.28). *If* (4.26) *holds, one has confidence to apply LLL-P. If* (4.28) *holds, one should not apply it. If both do not hold, i.e.,* $\min_i r_{ii}/(2\sigma) < r(\sqrt{2\pi}/(2d)) < \max_i r_{ii}/(2\sigma)$, *applying LLL-P may increase or decrease* P^{BB}.

4.2.2 Effects of SQRD and V-BLAST on P^{BB}

SQRD [95] and V-BLAST [25] have been used to find better box-constrained Babai estimators in the literature. It has been demonstrated in Chapter 3 that unlike LLL-P, both SQRD and V-BLAST may decrease the success probability P^{OB}

of the ordinary Babai estimator when the parameter vector $\hat{\boldsymbol{x}}$ is deterministic and not subject to any constraint.

We would like to know how SQRD and V-BLAST affect P^{BB}. Unlike LLL-P, both SQRD and V-BLAST usually involve two non-consecutive columns permutations, resulting in the changes of all diagonal entries between and including the two columns. This makes it very difficult to analyze under which conditions P^{BB} increases or decreases. We will use numerical test results to show the effects of SQRD and V-BLAST on P^{BB} with explanations.

In Theorem 4.2.2 we showed that if the condition (4.26) holds, then applying LLL-P will increase P^{BB}, and if (4.28) holds, then applying LLL-P will decrease P^{BB}. The following example shows they are not true for SQRD and V-BLAST.

Example 4.2.1 *Let $d = 1$ and consider two matrices:*

$$\boldsymbol{R}^{(1)} = \begin{bmatrix} 3.5 & 3 & 0 \\ 0 & 1 & -1.5 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{R}^{(2)} = \begin{bmatrix} 1 & -1.5 & 1.5 \\ 0 & 0.8 & -1 \\ 0 & 0 & 0.42 \end{bmatrix}.$$

Applying SQRD, V-BLAST and LLL-P to $\boldsymbol{R}^{(1)}$ and $\boldsymbol{R}^{(2)}$, we obtain

$$\boldsymbol{R}_S^{(1)} = \begin{bmatrix} 1.8028 & -0.8321 & 0 \\ 0 & 3.0509 & 3.4417 \\ 0 & 0 & 0.6364 \end{bmatrix}, \quad \boldsymbol{R}_V^{(1)} = \boldsymbol{R}_L^{(1)} = \begin{bmatrix} 3.1623 & 3.3204 & -0.4743 \\ 0 & 1.1068 & 1.4230 \\ 0 & 0 & 1 \end{bmatrix},$$

$$\boldsymbol{R}_V^{(2)} = \begin{bmatrix} 1.7 & -1.7941 & -0.8824 \\ 0 & 0.4556 & -0.1823 \\ 0 & 0 & 0.4338 \end{bmatrix}, \quad \boldsymbol{R}_S^{(2)} = \boldsymbol{R}_L^{(2)} = \boldsymbol{R}^{(2)}.$$

If $\sigma = 0.2$, then it is easy to verify that for both $\boldsymbol{R}^{(1)}$ and $\boldsymbol{R}^{(2)}$, the condition (4.26) *holds. Simple calculations by using* (4.1) *give*

$$P^{BB}(\boldsymbol{R}^{(1)}) = 0.9876, \quad P^{BB}(\boldsymbol{R}^{(2)}) = 0.8286.$$

Then

$$P^{BB}(\boldsymbol{R}_S^{(1)}) = 0.9442, \quad P^{BB}(\boldsymbol{R}_V^{(1)}) = P^{BB}(\boldsymbol{R}_L^{(1)}) = 0.9910,$$

$$P^{BB}(\boldsymbol{R}_V^{(2)}) = 0.7513, \quad P^{BB}(\boldsymbol{R}_S^{(2)}) = P^{BB}(\boldsymbol{R}_L^{(2)}) = 0.8286.$$

Thus for $\boldsymbol{R}^{(1)}$ SQRD decreases P^{BB}, while V-BLAST and LLL-P increase P^{BB} and for $\boldsymbol{R}^{(2)}$ V-BLAST decreases P^{BB}, while SQRD and LLL-P keep P^{BB} unchanged.

If $\sigma = 2.2$, then it is easy to verify that for both $\boldsymbol{R}^{(1)}$ and $\boldsymbol{R}^{(2)}$, the condition (4.28) *holds. Simple calculations by using* (4.1) *give*

$$P^{BB}(\boldsymbol{R}^{(1)}) = 0.2738, \quad P^{BB}(\boldsymbol{R}^{(2)}) = 0.1816.$$

Then

$$P^{BB}(\boldsymbol{R}_S^{(1)}) = 0.2777, \quad P^{BB}(\boldsymbol{R}_V^{(1)}) = P^{BB}(\boldsymbol{R}_L^{(1)}) = 0.2700,$$

$$P^{BB}(\boldsymbol{R}_V^{(2)}) = 0.1898, \quad P^{BB}(\boldsymbol{R}_S^{(2)}) = P^{BB}(\boldsymbol{R}_L^{(2)}) = 0.1816.$$

Thus for $\boldsymbol{R}^{(1)}$ SQRD increases P^{BB}, while V-BLAST and LLL-P decrease P^{BB} and for $\boldsymbol{R}^{(2)}$ V-BLAST increases P^{BB}, while SQRD and LLL-P keep P^{BB} unchanged.

Although Example 4.2.1 indicates that, unlike LLL-P, under the condition (4.26), both SQRD and V-BLAST may decrease P^{BB}, but they often increase P^{BB}. This is the reason why SQRD and V-BLAST (especially the latter) have often been used

to increase the accuracy of the Babai estimator in practice. Example 4.2.1 also indicates that, unlike LLL-P, under the condition (4.28), both SQRD and V-BLAST may increase P^{BB}, but they often decrease P^{BB}. This is the opposite of what we commonly believe. Later we give numerical test results to show both phenomenons. In the following we give some explanations.

It is easy to show that like LLL-P, V-BLAST increases $\min_i r_{ii}$ after each permutation and like LLL-P, SQRD decreases $\max_i r_{ii}$ after each permutation, see [25], [54]. For the relation between V-BLAST and SQRD, see, e.g., [17] and [54]. Thus if the condition (4.26) holds before applying V-BLAST, it will also hold after applying it; and if the condition (4.28) holds before applying SQRD, it will also hold after applying it. Often applying V-BLAST decreases $\max_i r_{ii}$ and applying SQRD increases $\min_i r_{ii}$ (both may not be true sometimes, see Example 4.2.1). Thus often the gaps between the large diagonal entries and the small ones of $\boldsymbol{R}$ decrease after applying SQRD or V-BLAST. From the proof of Theorem 4.2.1 we see reducing the gaps will likely increase P^{BB} under the condition (4.26) and decrease P^{BB} under the condition (4.28). Thus it is likely both SQRD and V-BLAST will increase P^{BB} under (4.26) and decrease it under (4.28). We will give further explanations in the next subsection.

4.2.3 A bound on P^{BB}

In this subsection we give a bound on P^{BB}, which is an upper bound under one condition and becomes a lower bound under an opposite condition. This bound can help us to understand what a column permutation strategy should try to achieve.

Theorem 4.2.3 *Suppose the assumptions in Theorem 4.1.1 hold. Let the box $\mathcal{B}$ in* (1.3b) *be a cube with edge length of d and denote $\gamma = (\det(\boldsymbol{R}))^{1/n}$.*

1. If the condition (4.26) *holds, then*

$$\Pr(\boldsymbol{x}^{BB} = \hat{\boldsymbol{x}}) \leq \Big[\frac{1}{d+1} + \frac{d}{d+1}\phi_\sigma(\gamma)\Big]^n. \tag{4.30}$$

2. If the condition (4.28) *holds, then*

$$\Pr(\boldsymbol{x}^{BB} = \hat{\boldsymbol{x}}) \geq \Big[\frac{1}{d+1} + \frac{d}{d+1}\phi_\sigma(\gamma)\Big]^n. \tag{4.31}$$

The equality in either (4.30)*or* (4.31) *holds if and only if $r_{ii} = \gamma$ for $i = 1, \ldots, n$.*

Proof. We prove only part 1. Part 2 can be proved similarly. Note that $\gamma^n = \Pi_{i=1}^n r_{ii}$. Obviously, if $r_{ii} = \gamma$ for $i = 1, \ldots, n$, then by (4.1) the equality in (4.30) holds. In the following we assume there exist j and k such that $r_{jj} \neq r_{kk}$, we only need to show that the strict inequality (4.30) holds.

Denote $F(\zeta) = \ln(1 + d\phi_\sigma(\exp(\zeta)))$, $\eta_i = \ln(r_{ii})$ for $i = 1, 2, \ldots, n$ and $\eta = \frac{1}{n}\sum_{i=1}^n \eta_i$. It is easy to see that (4.30) is equivalent to

$$\frac{1}{n}\sum_{i=1}^n F(\eta_i) < F(\eta).$$

Since $\min_i r_{ii} \geq 2\sigma r(\sqrt{2\pi}/(2d))$ and $r_{jj} \neq r_{kk}$, it suffices to show that $F(\zeta)$ is a strict concave function on $(\ln(2\sigma r(\sqrt{2\pi}/(2d))), +\infty)$. Therefore, we only need to show that $F''(\zeta) < 0$ when $\zeta > \ln(2\sigma r(\sqrt{2\pi}/(2d)))$.

To simplify notation, denote $\xi = \exp(\zeta)/(2\sigma)$. Simple calculations give

$$F'(\zeta) = \frac{\xi \exp\big(-\frac{1}{2}\xi^2\big)}{\frac{\sqrt{2\pi}}{2d} + \int_0^\xi \exp\big(-\frac{1}{2}t^2\big)dt}.$$

Using (4.11), we obtain

$$F''(\zeta) = \frac{\xi \exp\big(-\frac{1}{2}\xi^2\big)}{(\frac{\sqrt{2\pi}}{2d} + \int_0^{\xi} \exp\big(-\frac{1}{2}t^2\big)dt)^2} f\Big(\xi, \frac{\sqrt{2\pi}}{2d}\Big).$$

When $\zeta > \ln(2\sigma r(\sqrt{2\pi}/(2d)))$, $\xi > r(\sqrt{2\pi}/(2d))$. Thus, by Lemma 4.2.1, $f(\xi, \sqrt{2\pi}/(2d)) < 0$. Then we can conclude that $F''(\zeta) < 0$ when $\zeta > \ln(2\sigma r(\sqrt{2\pi}/(2d)))$, completing the proof. $\square$

Now we make some remarks about Theorem 4.2.3.

Remark 4.2.6 *The quantity γ is invariant with respect to column permutations, i.e., for $\boldsymbol{R}$ and $\bar{\boldsymbol{R}}$ in* (2.20)*, we have the same γ no matter what the permutation matrix $\boldsymbol{P}$ is. Thus the bounds in* (4.30) *and* (4.31)*, which are actually the same quantity, are invariant with respect to column permutations. Although the condition* (4.26) *is variant with respect to column permutations, if it holds before applying LLL-P or V-BLAST, it will hold afterwards, since the minimum of the diagonal entries of $\bar{\boldsymbol{R}}$ will not be smaller than that of $\boldsymbol{R}$ after applying LLL-P or V-BLAST. Similarly, the condition* (4.28) *is also variant with respect to column permutations. But if it holds before applying LLL-P or SQRD, it will hold afterwards, since the maximum of the diagonal entries of $\bar{\boldsymbol{R}}$ will not be larger than that of $\boldsymbol{R}$ after applying LLL-P or SQRD.*

Remark 4.2.7 *The bounds* (4.30) *and* (4.31) *are reached if all the diagonal entries of $\boldsymbol{R}$ are identical. This suggests that if the gaps between the larger entries and small entries become smaller after permutations, it is likely that P^{BB} increases under the condition* (4.26) *or decreases under the condition* (4.28)*. As we know, the gap*

between the largest one and the smallest one decreases after applying LLL-P. Numerical tests indicate usually this is also true for both V-BLAST and SQRD. Thus both V-BLAST and SQRD will likely bring P^{BB} closer to the bound under the two opposite conditions, respectively.

Remark 4.2.8 *When $d \to \infty$, by Lemma 4.2.1, $r(\sqrt{2\pi}/(2d)) \to 0$, thus the condition in part 1 of Theorem 4.2.3 becomes $\max_i r_{ii} \geq 0$, which certainly holds always. Taking the limit as $d \to \infty$ on both sides of* (4.30) *and using Corollary 4.1.1, we obtain*

$$\Pr(\boldsymbol{x}^{OB} = \hat{\boldsymbol{x}}) \leq (\phi_\sigma(\gamma))^n. \tag{4.32}$$

The above result was obtained in [85] and a simple proof was provided by Lemma 3.3.2, for more details, see [18].

4.2.4 Numerical tests

We have shown in Theorem 4.2.2 that if (4.26) holds, then the LLL-P increases P^{BB} and (4.30) is an upper bound on P^{BB}; and if (4.28) holds, then the LLL-P decreases P^{BB} and (4.31) is a lower bound on P^{BB}. Example 4.2.1 shows that this conclusion does not always hold for SQRD and V-BLAST. To further understand the effects of the LLL-P, SQRD and V-BLAST on P^{BB} and to see how close they bring their corresponding P^{BB} to the bounds given by (4.30) and (4.31), we performed some numerical tests. For comparisons, we also performed tests for P^{OB}.

We performed MATLAB tests for the following two cases.

- Case 1: $\boldsymbol{A} = \frac{\sqrt{2}}{2}\text{randn}(n, n)$, where $\text{randn}(n, n)$ is a MATLAB built-in function to generate a random $n \times n$ matrix, whose entries follow the i.i.d normal distribution $\mathcal{N}(0, 1)$. So the elements of $\boldsymbol{A}$ follow the i.i.d normal distribution $\mathcal{N}(0, 1/2)$.
- Case 2: $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T$, $\boldsymbol{U}, \boldsymbol{V}$ are random orthogonal matrices obtained by the QR factorization of random matrices generated by $\text{randn}(n, n)$ and $\boldsymbol{D}$ is a $n \times n$ diagonal matrix with $d_{ii} = 10^{3(n/2-i)/(n-1)}$. The condition number of $\boldsymbol{A}$ is 1000.

In the tests for each case, we first chose $n = 4$ and $\mathcal{B} = [0, 1]^4$ and took different noise standard deviation σ to test different situations according to the conditions (4.26) and (4.28) imposed in Theorems 4.2.2 and 4.2.3. The edge length d of $\mathcal{B}$ is equal to 1. So in (4.26) and (4.28), $r(\sqrt{2\pi}/2d) = r(\sqrt{2\pi}/2) = 0.83992$. Details about choosing σ will be given later.

We use P^{BB}, P_L^{BB}, P_S^{BB} and P_V^{BB} respectively denote the success probability of the box-constrained Babai estimator corresponding to QR factorization (i.e., no permutations are involved), LLL-P, SQRD and V-BLAST. Let μ^{BB} denote the right-hand side of (4.30) or (4.31), so it is an upper bound if (4.26) holds and a lower bound if (4.28) holds. Similarly, P^{OB}, P_L^{OB}, P_S^{OB} and P_V^{OB} respectively denote the success probability of the ordinary Babai estimator corresponding to QR factorization, LLL-P, SQRD and V-BLAST. We use μ^{OB} to denote the right-hand side of (4.32), which is an upper bound on P^{OB}, P_L^{OB}, P_S^{OB} and P_V^{OB}. For each case, we performed 10 runs (notice that for each run we have different $\boldsymbol{A}$, $\hat{\boldsymbol{x}}$ and $\boldsymbol{v}$ due to randomness) and the results are displayed in Table 4–1–Table 4–6.

In Tables 4–1 and 4–2, $\sigma = \sigma_1 \equiv \min(r_{ii})/1.8$. It is easy to verify that the condition (4.26) holds. This means that in theory $P^{\text{BB}} \leq P_L^{\text{BB}}$ by Theorem 4.2.2 and $P^{\text{BB}}, P_L^{\text{BB}}, P_V^{\text{BB}} \leq \mu^{\text{BB}}$ by Theorem 4.2.3 and Remark 4.2.6. The numerical results given in Tables 4–1 and 4–2 are consistent with the theoretical results. The numerical results also indicate that SQRD and V-BLAST (nonstrictly) increase P^{BB}, although there is one exceptional case for SQRD in Table 4–2. We observe that the permutation strategies increase P^{BB} more significantly for Case 2 than for Case 1. The reason is that $\boldsymbol{A}$ is more ill-conditioned for Case 2, resulting in bigger gaps between the diagonal entries of $\boldsymbol{R}$, which can usually be reduced more effectively by the permutation strategies. We also observe that $P_S^{\text{BB}} \leq \mu^{\text{BB}}$ in both tables. Although in theory the inequality may not hold as we cannot guarantee the condition (4.26) holds after applying SQRD, usually SQRD can make $\min_i r_{ii}$ larger. Thus if (4.26) holds before applying SQRD, it is likely the condition still holds after applying it. Thus it is likely $P_S^{\text{BB}} \leq \mu^{\text{BB}}$ holds.

Tables 4–3 and 4–4 are opposite to Tables 4–1 and 4–2. In both tables, $\sigma = \sigma_2 \equiv \max(r_{ii})/1.6$, then the condition (4.28) holds. This means that in theory $P^{\text{BB}} \geq P_L^{\text{BB}}$ by Theorem 4.2.2 and $P^{\text{BB}}, P_L^{\text{BB}}, P_S^{\text{BB}} \geq \mu^{\text{BB}}$ by Theorem 4.2.3 and Remark 4.2.6. The numerical results given in the two tables are consistent with the theoretical results. The results in the two tables also indicate that both SQRD and V-BLAST (nonstrictly) decrease P^{BB}, although Example 4.2.1 shows that neither is always true under the condition (4.28). We also observe that $P_V^{\text{BB}} \geq \mu^{\text{BB}}$ in both tables. Although in theory the inequality may not hold as we cannot guarantee the condition (4.28) holds after applying V-BLAST, usually V-BLAST can make $\max_i r_{ii}$ smaller. Thus

if (4.28) holds before applying V-BLAST, it is likely the condition still holds after applying it. Thus it is likely $P_V^{\text{BB}} \geq \mu^{\text{BB}}$ holds.

In Tables 4–5 and 4–6, $\sigma = \sigma_3 \equiv (0.3\max(r_{ii}) + 0.7\min(r_{ii}))/1.68$. It is easy to verify that neither (4.26) (almost) nor (4.28) holds in theory. In theory we do not have results cover this situation. The numerical results in the two tables indicate all of the three permutation strategies can either increase or decrease P^{BB} and μ^{BB} can be larger or smaller than P^{BB}, P_L^{BB}, P_S^{BB} and P_V^{BB}. The reason we chose 0.3 and 0.7 rather than a more natural choice of 0.5 and 0.5 in defining σ here is that we may not be able to observe both increasing and decreasing phenomenons due to limited runs.

Now we make comments on the success probability of ordinary Babai points. From Table 4–1–Table 4–6, we observe that LLL-P always (nonstrictly) increases P^{OB}, and SQRD and V-BLAST almost always increases P^{OB} (there is one exceptional case for SQRD in Table 4–2 and two exceptional cases for V-BLAST in Table 4–6). Thus the ordinary case is different from the box-constrained case. We also observe $P^{\text{OB}} \leq P^{\text{BB}}$ for the same permutation strategies. Sometimes the difference between the two is large (see Tables 4–4 and 4–6).

Each of Table 4–1—Table 4–6 displays the results for only 10 runs due to space limitation. To make up for this shortcoming, we give Tables 4–7 and 4–8, which display some statistics for 1000 runs on the data generated exactly the same way as the data for the 10 runs. Specifically these two tables display the number of runs, in which P^{BB} (P^{OB}) increases, keeps unchanged and decreases after each of the three

permutation strategies is applied for Case 1 and Case 2, respectively. In the two tables, σ_1, σ_2 and σ_3 are defined in the same as those used in Tables 4–1—4–6.

Table 4–1: Success probabilities of Babai points and bounds for Case 1, $\sigma = \min(r_{ii})/1.8$

σ	P^{BB}	P_L^{BB}	P_S^{BB}	P_V^{BB}	μ^{BB}	P^{OB}	P_L^{OB}	P_S^{OB}	P_V^{OB}	μ^{OB}
0.0738	0.8159	1.0000	1.0000	1.0000	1.0000	0.6319	1.0000	1.0000	1.0000	1.0000
0.1537	0.7632	0.8423	0.8423	0.8423	0.9083	0.5503	0.6988	0.6988	0.6988	0.8231
0.1575	0.7938	0.9491	0.9491	0.9491	0.9698	0.5977	0.8998	0.8998	0.8998	0.9403
0.2170	0.7235	0.8577	0.8577	0.8577	0.8670	0.4893	0.7300	0.7300	0.7300	0.7477
0.1285	0.8133	0.8534	0.8534	0.8521	0.9882	0.6278	0.7070	0.7070	0.7049	0.9766
0.1676	0.6809	0.7529	0.7529	0.7529	0.8896	0.4255	0.5375	0.5375	0.5375	0.7885
0.3665	0.7039	0.7273	0.7273	0.7273	0.8004	0.4629	0.5093	0.5093	0.5093	0.6324
0.1968	0.6892	0.7320	0.7320	0.7385	0.8073	0.4420	0.5103	0.5103	0.5270	0.6439
0.3322	0.7087	0.7317	0.7317	0.7317	0.7665	0.4718	0.5156	0.5156	0.5156	0.5765
0.5221	0.4754	0.4754	0.4754	0.4754	0.4758	0.1899	0.1899	0.1899	0.1899	0.1910

Table 4–2: Success probabilities of Babai points and bounds for Case 2, $\sigma = \min(r_{ii})/1.8$

σ	P^{BB}	P_L^{BB}	P_S^{BB}	P_V^{BB}	μ^{BB}	P^{OB}	P_L^{OB}	P_S^{OB}	P_V^{OB}	μ^{OB}
0.0101	0.8155	0.9452	0.9354	0.9452	1.0000	0.6312	0.8905	0.8708	0.8905	1.0000
0.0130	0.7983	0.9839	0.9839	0.9839	1.0000	0.6045	0.9679	0.9679	0.9679	1.0000
0.0173	0.8159	0.9793	0.9793	0.9793	1.0000	0.6319	0.9586	0.9586	0.9586	1.0000
0.0066	0.8159	0.9913	0.9913	0.9967	1.0000	0.6319	0.9826	0.9826	0.9933	1.0000
0.0177	0.8106	0.9998	0.9998	0.9998	1.0000	0.6236	0.9997	0.9997	0.9997	1.0000
0.0060	0.8159	0.9841	0.9841	0.9998	1.0000	0.6319	0.9681	0.9681	0.9996	1.0000
0.0168	0.7833	0.8098	0.7625	0.8159	1.0000	0.5813	0.6224	0.5250	0.6319	1.0000
0.0150	0.8159	0.9999	0.9999	0.9999	1.0000	0.6319	0.9998	0.9998	0.9998	1.0000
0.0231	0.8159	0.9999	0.9999	0.9999	1.0000	0.6319	0.9999	0.9999	0.9999	1.0000
0.0211	0.7912	0.9696	0.9696	0.9892	1.0000	0.5935	0.9393	0.9393	0.9784	1.0000

Table 4–3: Success probabilities of Babai points and bounds for Case 1, $\sigma = \max(r_{ii})/1.6$

σ	P^{BB}	P_L^{BB}	P_S^{BB}	P_V^{BB}	μ^{BB}	P^{OB}	P_L^{OB}	P_S^{OB}	P_V^{OB}	μ^{OB}
1.1726	0.1557	0.1310	0.1310	0.1380	0.1121	0.0005	0.0006	0.0006	0.0006	0.0006
0.6432	0.2756	0.2756	0.2756	0.2756	0.2731	0.0387	0.0387	0.0387	0.0387	0.0395
0.5962	0.2915	0.2912	0.2912	0.2909	0.2900	0.0472	0.0473	0.0473	0.0475	0.0478
1.2435	0.1875	0.1632	0.1673	0.1632	0.1571	0.0040	0.0044	0.0044	0.0044	0.0045
0.8332	0.1873	0.1769	0.1769	0.1769	0.1750	0.0070	0.0074	0.0074	0.0074	0.0074
0.4875	0.2709	0.2709	0.2709	0.2709	0.2667	0.0356	0.0356	0.0356	0.0356	0.0366
0.9684	0.2769	0.2709	0.2709	0.2709	0.2688	0.0358	0.0369	0.0369	0.0369	0.0375
0.9971	0.1846	0.1665	0.1665	0.1665	0.1588	0.0043	0.0046	0.0046	0.0046	0.0047
1.2791	0.1501	0.1308	0.1308	0.1308	0.1294	0.0015	0.0016	0.0016	0.0016	0.0016
0.6327	0.2641	0.2564	0.2564	0.2564	0.2556	0.0301	0.0316	0.0316	0.0316	0.0318

Table 4–4: Success probabilities of Babai points and bounds for Case 2, $\sigma = \max(r_{ii})/1.6$

σ	P^{BB}	P_L^{BB}	P_S^{BB}	P_V^{BB}	μ^{BB}	P^{OB}	P_L^{OB}	P_S^{OB}	P_V^{OB}	μ^{OB}
3.9438	0.1064	0.0947	0.0987	0.0947	0.0709	0.0000	0.0000	0.0000	0.0000	0.0000
2.4510	0.1173	0.1173	0.1173	0.1173	0.0764	0.0000	0.0000	0.0000	0.0000	0.0000
0.5790	0.1788	0.1640	0.1640	0.1640	0.1363	0.0019	0.0019	0.0019	0.0019	0.0021
5.3809	0.1011	0.0701	0.0701	0.0701	0.0686	0.0000	0.0000	0.0000	0.0000	0.0000
2.2574	0.1140	0.1023	0.0954	0.0954	0.0777	0.0000	0.0000	0.0000	0.0000	0.0000
3.7623	0.1099	0.0801	0.0801	0.0757	0.0713	0.0000	0.0000	0.0000	0.0000	0.0000
3.9225	0.1063	0.0834	0.0834	0.0834	0.0709	0.0000	0.0000	0.0000	0.0000	0.0000
1.3198	0.1153	0.1153	0.1153	0.1153	0.0900	0.0001	0.0001	0.0001	0.0001	0.0001
1.2416	0.1394	0.1108	0.1108	0.1108	0.0920	0.0001	0.0001	0.0001	0.0001	0.0001
0.8411	0.1719	0.1532	0.1532	0.1532	0.1090	0.0004	0.0004	0.0004	0.0004	0.0005

Table 4–5: Success probabilities of Babai points and bounds for Case 1, $\sigma = (0.3\max(r_{ii}) + 0.7\min(r_{ii}))/1.68$

σ	P^{BB}	P_L^{BB}	P_S^{BB}	P_V^{BB}	μ^{BB}	P^{OB}	P_L^{OB}	P_S^{OB}	P_V^{OB}	μ^{OB}
0.2848	0.4208	0.4336	0.4336	0.3184	0.2846	0.0154	0.0165	0.0165	0.0252	0.0451
0.6313	0.4720	0.4829	0.4829	0.4829	0.4863	0.1630	0.1932	0.1932	0.1932	0.2017
0.4328	0.4540	0.4599	0.4599	0.4599	0.4623	0.1517	0.1673	0.1673	0.1673	0.1776
0.6105	0.5054	0.5061	0.5061	0.5061	0.5092	0.2123	0.2161	0.2161	0.2161	0.2259
0.3306	0.4268	0.3807	0.3807	0.3805	0.3484	0.0321	0.0539	0.0539	0.0539	0.0829
0.2600	0.5055	0.5103	0.5103	0.5103	0.5252	0.1544	0.1868	0.1868	0.1868	0.2437
0.4743	0.4235	0.4283	0.4283	0.4283	0.4259	0.0631	0.1225	0.1225	0.1225	0.1437
0.5878	0.4104	0.4161	0.4161	0.4161	0.4170	0.1159	0.1304	0.1304	0.1304	0.1359
0.3977	0.4429	0.4431	0.4431	0.4431	0.4477	0.1477	0.1479	0.1479	0.1479	0.1636
0.6273	0.4684	0.4684	0.4684	0.4684	0.4696	0.1792	0.1792	0.1792	0.1792	0.1848

Table 4–6: Success probabilities of Babai points and bounds for Case 2, $\sigma = (0.3\max(r_{ii}) + 0.7\min(r_{ii}))/1.68$

σ	P^{BB}	P_L^{BB}	P_S^{BB}	P_V^{BB}	μ^{BB}	P^{OB}	P_L^{OB}	P_S^{OB}	P_V^{OB}	μ^{OB}
1.0377	0.1608	0.1324	0.1324	0.1625	0.0987	0.0001	0.0002	0.0002	0.0002	0.0002
0.3648	0.2774	0.2774	0.2774	0.2405	0.1987	0.0034	0.0034	0.0034	0.0025	0.0126
0.7603	0.1681	0.1758	0.1758	0.1758	0.1150	0.0003	0.0005	0.0005	0.0005	0.0007
0.8769	0.1835	0.2062	0.1713	0.2062	0.1067	0.0002	0.0003	0.0004	0.0003	0.0004
0.4708	0.2794	0.2352	0.2352	0.2352	0.1590	0.0010	0.0030	0.0030	0.0030	0.0048
1.1983	0.1572	0.1319	0.1319	0.1319	0.0932	0.0001	0.0001	0.0001	0.0001	0.0001
1.0001	0.1758	0.1596	0.1596	0.1464	0.1003	0.0001	0.0002	0.0002	0.0001	0.0002
0.8523	0.1671	0.1733	0.1733	0.1715	0.1082	0.0002	0.0003	0.0003	0.0003	0.0005
0.2128	0.3478	0.3478	0.3728	0.3478	0.3539	0.0599	0.0599	0.0711	0.0599	0.0866
0.3956	0.2188	0.2117	0.2117	0.1973	0.1844	0.0047	0.0047	0.0047	0.0034	0.0093

Table 4–7: Number of runs out of 1000 in which P^{BB} and P^{OB} changes for Case 1

		P^{BB}			P^{OB}		
σ	Result \ Strategy	LLL-P	SQRD	V-BLAST	LLL-P	SQRD	V-BLAST
σ_1	Increase	933	928	951	933	922	953
	No change	67	47	42	67	47	42
	Decrease	0	25	7	0	31	5
σ_2	Increase	0	25	6	942	947	950
	No change	58	40	37	58	40	37
	Decrease	942	935	957	0	13	13
σ_3	Increase	781	797	740	942	945	952
	No change	58	40	37	58	40	37
	Decrease	161	163	223	0	15	11

Table 4–8: Number of runs out of 1000 in which P^{BB} and P^{OB} changes for Case 2

		P^{BB}			P^{OB}		
σ	Result \ Strategy	LLL-P	SQRD	V-BLAST	LLL-P	SQRD	V-BLAST
σ_1	Increase	858	803	938	858	800	938
	No change	142	76	56	142	76	56
	Decrease	0	121	6	0	124	6
σ_2	Increase	0	23	69	906	944	831
	No change	94	46	48	94	46	48
	Decrease	906	931	883	0	10	121
σ_3	Increase	134	189	97	906	943	840
	No change	94	46	48	94	46	48
	Decrease	772	765	855	0	11	112

From Tables 4–7 and 4–8, we can see that often these permutation strategies increase or decrease P^{BB} for the same data. The numerical results given in all the tables suggest that if the condition (4.26) holds, we should have confidence to use any of these permutation strategies; and if the condition (4.28) holds we should not use any of them.

Tables 4–7 and 4–8 do not show which permutation strategy can increase P^{BB} most. The information on this given in Tables 4–1 and 4–6 are limited. In the following we give more test results to investigate this.

We still consider Cases 1 and 2, but we take $\mathcal{B} = [0, 15]^n$ and choose different n and σ from before. In Figures 4–1 and 4–2 for $n = 20$, we take $\sigma = 0.1 : 0.1 : 0.8$ and $\sigma = 0.01 : 0.01 : 0.08$ for Cases 1 and 2, respectively. For each σ, we give

200 runs to generate 200 different $\boldsymbol{A}$'s. These two figures display the average P^{BB} corresponding to QR, LLL-P, SQRD and V-BLAST over 200 runs for Cases 1 and 2, respectively. Figures 4–3 and 4–4 display the average P^{BB} corresponding to the various permutation strategies over 200 runs versus $n = 5 : 5 : 40$ with $\sigma = 0.4$ and $\sigma = 0.04$ for Cases 1 and 2, respectively. The reason we choose different σ for the two cases is to ensure P^{BB} is neither close to 0 nor close to 1; otherwise, it is not much interesting to investigate the effects of the column permutations on P^{BB}.

From Figure 4–1—Figure 4–4, we can see that on average all of the three column permutation strategies improve P^{BB}. The effect of V-BLAST is much more significantly than that of LLL-P and SQRD, which have more or less the same performance. This phenomenon is similar to that for P^{OB}, as shown in [18].

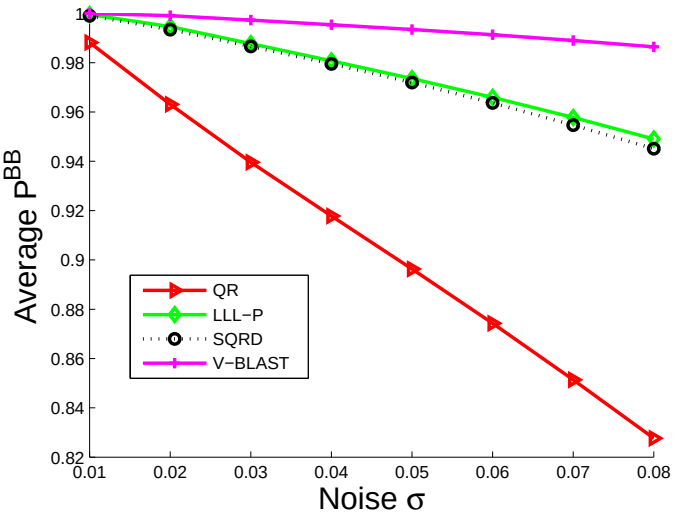

Figure 4–1: Average P^{BB} over 200 runs versus σ for Case 1, $n = 20$

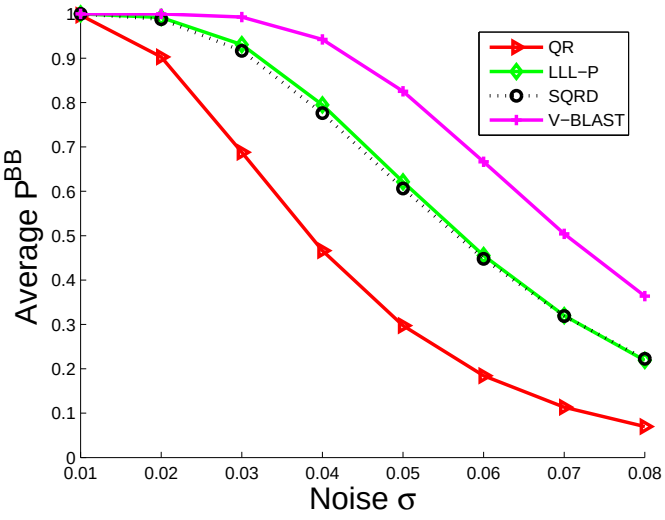

Figure 4–2: Average P^{BB} over 200 runs versus σ for Case 2, $n = 20$

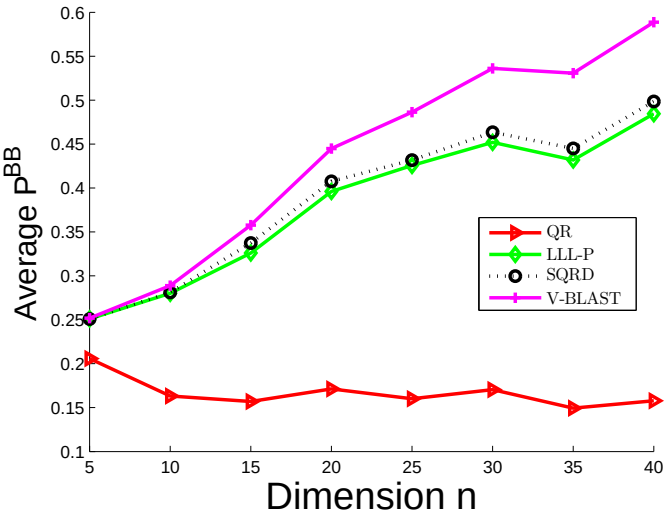

Figure 4–3: Average P^{BB} over 200 runs versus n for Case 1, $\sigma = 0.4$

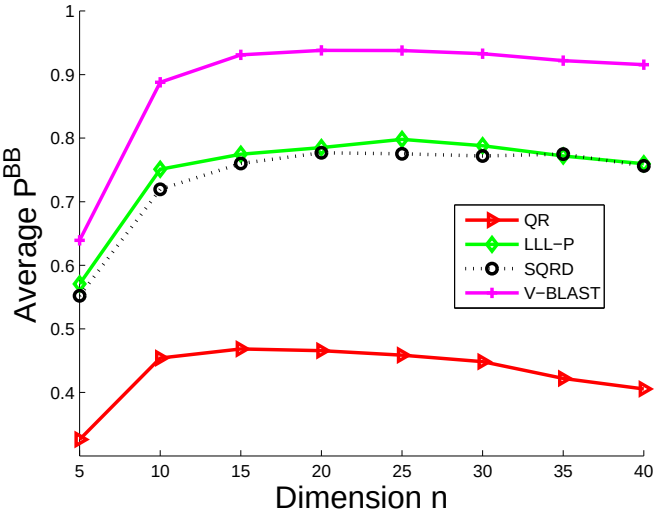

Figure 4–4: Average P^{BB} over 200 runs versus n for Case 2, $\sigma = 0.04$

4.3 Solving a conjecture

In [58], a conjecture was made on the ordinary Babai estimator, based on which a stopping criterion was then proposed for the sphere decoding search process for solving the BILS problem (1.4). In this section, first, we give an example to show that this conjecture may not hold in general. Then, we show that the conjecture holds under some conditions. And finally, we propose a new stopping criterion, which is more reliable, for solving the BILS problem (1.4).

The problem considered in [58] is to estimate the integer parameter vector $\hat{\boldsymbol{x}}$ in the box-constrained linear model (1.3). The method proposed in [58] first ignores the box constraint (1.3b). Instead of using the column permutations in (2.20), it performs the LLL reduction (2.11), then the linear model (2.3) can be transformed to (2.17). For the reduced model (2.17), one can find its ordinary Babai estimator $\boldsymbol{z}^{\mathrm{OB}}$. Define $\bar{\boldsymbol{x}} = \boldsymbol{Z}\boldsymbol{z}^{\mathrm{OB}}$. In [58], $\bar{\boldsymbol{x}}$ is used as an estimator of the true parameter

vector $\hat{\boldsymbol{x}}$. If $\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}}$, then a vector error (VE) is said to have occurred. Note that $\bar{\boldsymbol{x}}$ may be outside the constraint box $\mathcal{B}$ in (1.3b). If $\bar{\boldsymbol{x}} \in \mathcal{B}$, then $\bar{\boldsymbol{x}}$ is called a valid vector, otherwise, i.e., $\bar{\boldsymbol{x}} \notin \mathcal{B}$, $\bar{\boldsymbol{x}}$ is called an invalid vector. The conjecture proposed in [58] is: a VE is most likely to occur if $\bar{\boldsymbol{x}}$ is invalid; conversely, if $\bar{\boldsymbol{x}}$ is valid, there is little chance that the vector is in error.

From the definition of VE, if $\bar{\boldsymbol{x}}$ is invalid, then VE must occur. So in the following, we will only consider the second part of the conjecture, i.e., $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) \approx 0$.

4.3.1 The conjecture does not always hold

In this subsection, we first show that $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B})$ can be very close to 1, then give a specific example to show $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) \geq 0.9275$, and finally perform some Matlab simulations to illustrate this example.

Theorem 4.3.1 *For any given $\epsilon > 0$, any fixed dimension $n \geq 2$, any box $\mathcal{B}$ and any standard deviation σ of the noise vector $\boldsymbol{v}$, there always exists a box-constrained linear model in the form of (2.5), where $\hat{\boldsymbol{x}}$ is uniformly distributed over the box $\mathcal{B}$, such that*

$$\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) \geq 1 - \frac{1}{u_1 - l_1 + 1} - \epsilon. \tag{4.33}$$

Proof. Note that

$$\begin{aligned} \Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) &= \frac{\Pr(\bar{\boldsymbol{x}} \in \mathcal{B}) - \Pr((\bar{\boldsymbol{x}} = \hat{\boldsymbol{x}}) \cap (\bar{\boldsymbol{x}} \in \mathcal{B}))}{\Pr(\bar{\boldsymbol{x}} \in \mathcal{B})} \\ &= 1 - \frac{\Pr(\bar{\boldsymbol{x}} = \hat{\boldsymbol{x}})}{\Pr(\bar{\boldsymbol{x}} \in \mathcal{B})}. \end{aligned} \tag{4.34}$$

Thus, to prove the theorem, it suffices to show that there exists a box-constrained linear model such that

$$\Pr(\bar{\boldsymbol{x}} = \hat{\boldsymbol{x}}) \leq \Big(\frac{1}{u_1 - l_1 + 1} + \epsilon\Big) \Pr(\bar{\boldsymbol{x}} \in \mathcal{B}). \tag{4.35}$$

For any fixed σ, box $\mathcal{B}$ and $\epsilon > 0$, there exists $r_{11} \in \mathbb{R}^+$ such that

$$\begin{aligned}\phi_\sigma(r_{11}) \leq & \Big(\frac{1}{u_1 - l_1 + 1} + \epsilon\Big) \\ & \times \Big(\frac{1}{2(u_1 - l_1 + 1)} \sum_{i=0}^{u_1 - l_1} [\phi_\sigma((2u_1 - 2l_1 - 2i + 1)r_{11}) + \phi_\sigma((2i+1)r_{11})]\Big)\end{aligned} \tag{4.36}$$

where $\phi_\sigma(\cdot)$ is defined in (3.1). In fact, for any fixed σ, it is easy to verify by L'Höpital's rule that

$$\lim_{r_{11} \to 0} \frac{\phi_\sigma(r_{11})}{\frac{1}{2(u_1 - l_1 + 1)} \sum_{i=0}^{u_1 - l_1} [\phi_\sigma((2u_1 - 2l_1 - 2i + 1)r_{11}) + \phi_\sigma((2i+1)r_{11})]} = \frac{1}{u_1 - l_1 + 1}.$$

Therefore, there exists $r_{11} \in \mathbb{R}^+$ such that (4.36) holds.

To construct the linear model, we need only to construct a matrix $\boldsymbol{R} \in \mathbb{R}^{m \times n}$. With r_{11} satisfying (4.36), take r_{22} such that $r_{22} \geq r_{11}$ and define

$$\boldsymbol{R} = \begin{bmatrix} r_{11} & 0.5 r_{11} \boldsymbol{e}^T \\ \boldsymbol{0} & r_{22} \boldsymbol{I}_{n-1, n-1} \end{bmatrix}$$

where $\boldsymbol{e} = [1, \ldots, 1]^T \in \mathbb{R}^{n-1}$. For the resulting linear model, in the following we show that (4.35) holds.

Note that $\boldsymbol{R}$ is already LLL reduced, thus, $\bar{\boldsymbol{x}} = \boldsymbol{z}^{\mathrm{OB}} = \boldsymbol{x}^{\mathrm{OB}}$ and $\hat{\boldsymbol{x}} = \hat{\boldsymbol{z}}$. Then, by (4.2), the left-hand side of (4.35) satisfies

$$\Pr(\bar{\boldsymbol{x}} = \hat{\boldsymbol{x}}) = \phi_\sigma(r_{11})\phi_\sigma^{n-1}(r_{22}). \tag{4.37}$$

Obviously,

$$\begin{aligned}\Pr(\bar{\boldsymbol{x}} \in \mathcal{B}) = \Pr(\boldsymbol{x}^{\mathrm{OB}} \in \mathcal{B}) &\geq \Pr\left(x_1^{\mathrm{OB}} \in [l_1, u_1] \bigcap (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))\right) \\ &= \Pr\big(x_1^{\mathrm{OB}} \in [l_1, u_1] | (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))\big) \cdot \Pr\big(\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i)\big) \\ &= \Pr\big(x_1^{\mathrm{OB}} \in [l_1, u_1] | (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))\big)\phi_\sigma^{n-1}(r_{22}) \end{aligned} \tag{4.38}$$

where the last equality follows from (4.2). Therefore, by (4.37) and (4.38), to show (4.35) it suffices to show

$$\phi_\sigma(r_{11}) \leq \left(\frac{1}{u_1 - l_1 + 1} + \epsilon\right) \Pr(x_1^{\mathrm{OB}} \in [l_1, u_1] | (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))).$$

Then, by (4.36), it suffices to show

$$\begin{aligned}&\Pr(x_1^{\mathrm{OB}} \in [l_1, u_1] | (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))) \\ =&\frac{1}{2(u_1 - l_1 + 1)} \times \sum_{i=0}^{u_1 - l_1} [\phi_\sigma((2u_1 - 2l_1 - 2i + 1)r_{11}) + \phi_\sigma((2i+1)r_{11})]. \end{aligned} \tag{4.39}$$

From the proof for Theorem 4.1.1, we see that if $x_i^{\mathrm{OB}} = \hat{x}_i$ for $i = n, n-1, \ldots, 2$ and $\hat{x}_1$ is fixed, then $c_1^{\mathrm{OB}} \sim \mathcal{N}(\hat{x}_1, \sigma^2/r_{11}^2)$. Since $x_1^{\mathrm{OB}} = \lfloor c_1^{\mathrm{OB}} \rceil$ and $\hat{\boldsymbol{x}}$ is uniformly

distributed over the box $\mathcal{B}$,

$$
\begin{aligned}
& \Pr\big(x_1^{\mathrm{OB}} \in [l_1, u_1] | (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))\big) \\
= & \sum_{i=0}^{u_1-l_1} \Pr\big(\hat{x}_1 = l_1 + i, x_1^{\mathrm{OB}} \in [l_1, u_1] | (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))\big) \\
= & \sum_{i=0}^{u_1-l_1} \Pr(\hat{x}_1 = l_1 + i) \Pr\big(x_1^{\mathrm{OB}} \in [l_1, u_1] | (\hat{x}_1 = l_1 + i) \cap (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))\big) \\
= & \frac{1}{u_1 - l_1 + 1} \sum_{i=0}^{u_1-l_1} \Pr\big(c_1^{\mathrm{OB}} \in [l_1 - 1/2, u_1 + 1/2] | (\hat{x}_1 = l_1 + i) \cap (\cap_{i=2}^n (x_i^{\mathrm{OB}} = \hat{x}_i))\big) \\
= & \frac{1}{u_1 - l_1 + 1} \sum_{i=0}^{u_1-l_1} \frac{1}{\sqrt{2\pi}\sigma/r_{11}} \int_{l_1-1/2}^{u_1+1/2} \exp\left(-\frac{(t - l_1 - i)^2}{2\sigma^2/r_{11}^2}\right) dt \\
= & \frac{1}{u_1 - l_1 + 1} \sum_{i=0}^{u_1-l_1} \frac{1}{\sqrt{2\pi}} \int_{-(2i+1)r_{11}/(2\sigma)}^{(2u_1-2l_1-2i+1)r_{11}/(2\sigma)} \exp(-t^2/2) dt \\
= & \frac{1}{2(u_1 - l_1 + 1)} \sum_{i=0}^{u_1-l_1} [\phi_\sigma((2u_1 - 2l_1 - 2i + 1)r_{11}) + \phi_\sigma((2i + 1)r_{11})],
\end{aligned}
$$

where the second equality follows from (4.6). Therefore, (4.39) holds, and this completes the proof. □

As $u_1 - l_1 + 1$ is at least 2, Theorem 4.3.1 shows that $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B})$ can be at least $1/2 - \epsilon$ and can be very close to 1. In the following, we give a specific example to show that $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) \geq 0.9275$ and give some simulation results.

Example 4.3.1 *For any fixed n and σ, let $\epsilon = 0.01$ and $\mathcal{B} = [0, 15]^n$, and define*

$$
\boldsymbol{R} = \begin{bmatrix} 0.04\sigma & 0.02\sigma \boldsymbol{e}^T \\ \mathbf{0} & 10\sigma \boldsymbol{I}_{n-1,n-1} \end{bmatrix}. \tag{4.40}
$$

It is easy to verify that this matrix $\boldsymbol{R}$ satisfies the requirements given in the proof of Theorem 4.3.1. Then by (4.33)*, we have* $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) \geq 0.9275$.

We use MATLAB *to do some simulations to illustrate the probability. In the simulations, for any fixed* n *and* σ*, we generated an* $n \times n$ *matrix* $\boldsymbol{R}$ *by using* (4.40)*. After fixing* $\boldsymbol{R}$*, we gave 10000 runs to generate 10000* $\tilde{\boldsymbol{v}}$*'s and 10000* $\hat{\boldsymbol{x}}$*'s according to their distributions, producing 10000* $\tilde{\boldsymbol{y}}$*'s according to* (2.3)*. For each* $\tilde{\boldsymbol{y}}$*, we found the Babai point* $\boldsymbol{x}^{OB}$ *by using* (2.7)*. For each pair of* $\boldsymbol{R}$ *and* σ*, we computed the theoretical probability* $\Pr(\bar{\boldsymbol{x}} = \hat{\boldsymbol{x}})$ *denoted by* P_{th} *by using* (4.37) *(notice that* $P_{th} = P^{OB}$ *since* $\bar{\boldsymbol{x}} = \boldsymbol{x}^{OB}$ *here) and the corresponding experimental probability* P_{ex} *(i.e., the ratio of the number of runs in which* $\bar{\boldsymbol{x}} = \hat{\boldsymbol{x}}$ *to 10000). We also computed the experimental probability* P_b *of* $\bar{\boldsymbol{x}} \in \mathcal{B}$ *(i.e., the ratio of the number of runs in which* $\bar{\boldsymbol{x}} \in \mathcal{B}$ *to 10000) and the experimental probability* P_e *corresponding to* $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B})$ *(i.e.,* $P_e = 1 - P_{ex}/P_b$*).*

Tables 4–9 and 4–10 respectively display those probabilities versus $n = 5 : 5 : 40$ *with* $\sigma = 0.1$ *and versus* $\sigma = 0.1 : 0.1 : 0.8$ *with* $n = 20$*. From these two tables, we can see that the values of* P_e *are larger than 0.9275 except the case that* $n = 40$ *in Table 4–9, in which* P_e *is smaller than 0.9275, but it is close to the latter. Thus the test results are consistent with the theoretical result. We also observe that* P_{th} *is very small and* P_{ex} *is a good approximation to* P_{th}*. In Tables 4–9 the values of* P_{th} *are actually different, but very close because* $\phi_\sigma(r_{22})$ *is very close to 1 (c.f.* (4.37)*) and in Tables 4–10 the values of* P_{th} *are exactly equal because in* (4.37) $\phi(r_{11})$ *and* $\phi_\sigma(r_{22})$ *are independent of* σ*. This experiment confirms that even if* $\bar{\boldsymbol{x}}$ *is valid, there may be a large chance that it is in error.*

4.3.2 The conjecture holds under some conditions

In this subsection, we will show that the conjecture holds under some conditions.

Table 4–9: Probabilities versus $n = 5 : 5 : 40$ with $\sigma = 0.1$

$\boldsymbol{n}$	P_{th}	P_{ex}	P_b	P_e
5	0.0160	0.0160	0.2484	0.9356
10	0.0160	0.0150	0.2485	0.9396
15	0.0160	0.0164	0.2469	0.9336
20	0.0160	0.0163	0.2499	0.9348
25	0.0160	0.0173	0.2564	0.9325
30	0.0160	0.0172	0.2500	0.9312
35	0.0160	0.0172	0.2473	0.9304
40	0.0160	0.0182	0.2434	0.9252

Table 4–10: Probabilities versus $\sigma = 0.1 : 0.1 : 0.8$ with $n = 20$

σ	P_{th}	P_{ex}	P_b	P_e
0.1	0.0160	0.0159	0.2503	0.9365
0.2	0.0160	0.0168	0.2522	0.9334
0.3	0.0160	0.0156	0.2434	0.9359
0.4	0.0160	0.0156	0.2399	0.9350
0.5	0.0160	0.0175	0.2435	0.9281
0.6	0.0160	0.0173	0.2475	0.9301
0.7	0.0160	0.0157	0.2541	0.9382
0.8	0.0160	0.0162	0.2517	0.9356

Recall $\bar{\boldsymbol{x}} = \boldsymbol{Z}\boldsymbol{z}^{\mathrm{OB}}$ and $\hat{\boldsymbol{x}} = \boldsymbol{Z}\hat{\boldsymbol{z}}$, thus $\Pr(\bar{\boldsymbol{x}} = \hat{\boldsymbol{x}}) = \Pr(\boldsymbol{z}^{\mathrm{OB}} = \hat{\boldsymbol{z}})$. Then, by (4.34), we have

$$\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) \leq 1 - \Pr(\boldsymbol{z}^{\mathrm{OB}} = \hat{\boldsymbol{z}}). \tag{4.41}$$

So, if $\Pr(\boldsymbol{z}^{\mathrm{OB}} = \hat{\boldsymbol{z}}) \approx 1$, then the conjecture holds. From Corollary 4.1.2 we see that when σ is small enough, we have $\Pr(\boldsymbol{z}^{\mathrm{OB}} = \hat{\boldsymbol{z}}) \approx 1$. But the upper bound given in (4.41) is not sharp because it was derived from (4.34) by using the inequality $\Pr(\bar{\boldsymbol{x}} \in \mathcal{B}) \leq 1$. We will give a sharper upper bound on $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B})$ based on a sharper upper bound on $\Pr(\bar{\boldsymbol{x}} \in \mathcal{B})$.

Since $\bar{\boldsymbol{x}} = \boldsymbol{Z}\boldsymbol{z}^{\rm OB}$, $\bar{\boldsymbol{x}} \in \mathcal{B}$ if and only if $\boldsymbol{z}^{\rm OB} \in \mathcal{E} \equiv \{\boldsymbol{Z}^{-1}\boldsymbol{s} \,|\, \forall \boldsymbol{s} \in \mathcal{B}\}$. Thus $\Pr(\bar{\boldsymbol{x}} \in \mathcal{B}) = \Pr(\boldsymbol{z}^{\rm OB} \in \mathcal{E})$. But the set $\mathcal{E}$ is a parallelotope and it is difficult to analyze $\Pr(\boldsymbol{z}^{\rm OB} \in \mathcal{E})$. Thus in the following we will give a box $\mathcal{F}$ to include $\mathcal{E}$, then we analyze $\Pr(\boldsymbol{z}^{\rm OB} \in \mathcal{F})$. Let $\boldsymbol{U} = (u_{ij}) = \boldsymbol{Z}^{-1}$ and define for $i, j = 1, 2, \ldots, n$,

$$\mu_{ij} = \begin{cases} l_j, & \text{if } \ u_{ij} \geq 0 \\ u_j, & \text{if } \ u_{ij} < 0 \end{cases}, \quad \nu_{ij} = \begin{cases} u_j, & \text{if } \ u_{ij} \geq 0 \\ l_j, & \text{if } \ u_{ij} < 0 \end{cases}.$$

Then define $\bar{\boldsymbol{l}} \in \mathbb{Z}^n$ and $\bar{\boldsymbol{u}} \in \mathbb{Z}^n$ as follows:

$$\bar{l}_i = \sum_{j=1}^{n} u_{ij}\mu_{ij}, \quad \bar{u}_i = \sum_{j=1}^{n} u_{ij}\mu_{ij}, \quad i = 1, 2, \ldots, n. \tag{4.42}$$

It is easy to observe that

$$\mathcal{E} \subseteq \mathcal{F} \equiv \{\boldsymbol{z} \in \mathbb{Z}^n : \bar{\boldsymbol{l}} \leq \boldsymbol{z} \leq \bar{\boldsymbol{u}}\}. \tag{4.43}$$

Actually it is easy to observe that $\mathcal{F}$ is the smallest box including $\mathcal{E}$.

With the above preparation, we now give the following result.

Theorem 4.3.2 *Suppose that the assumptions in Theorem 4.1.1 hold and the linear model* (2.5a) *is transformed to the linear model* (2.17) *through the LLL reduction* (2.11). *Then the estimator* $\bar{\boldsymbol{x}}$ *defined as* $\bar{\boldsymbol{x}} = \boldsymbol{Z}\boldsymbol{z}^{OB}$ *satisfies*

$$\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) \leq 1 - \prod_{i=1}^{n} \frac{\phi_\sigma(\bar{r}_{ii})}{\phi_\sigma((\bar{u}_i - \bar{l}_i + 1)\bar{r}_{ii})}, \tag{4.44}$$

where ϕ_σ *is defined in* (3.1) *and* $\bar{\boldsymbol{l}}$ *and* $\bar{\boldsymbol{u}}$ *are defined in* (4.42).

Proof. Since $\Pr(\bar{\boldsymbol{x}} \in \mathcal{B}) = \Pr(\boldsymbol{z}^{\text{OB}} \in \mathcal{E})$, it follows from (4.43) that

$$\Pr(\bar{\boldsymbol{x}} \in \mathcal{B}) \leq \Pr(\boldsymbol{z}^{\text{OB}} \in \mathcal{F}). \tag{4.45}$$

In the following, we will show

$$\Pr(\boldsymbol{z}^{\text{OB}} \in \mathcal{F}) \leq \prod_{i=1}^{n} \phi_\sigma((\bar{u}_i - \bar{l}_i + 1)\bar{r}_{ii}). \tag{4.46}$$

Then combining (4.45), (4.46) and the fact that $\Pr(\bar{\boldsymbol{x}} = \hat{\boldsymbol{x}}) = \Pr(\boldsymbol{z}^{\text{OB}} = \hat{\boldsymbol{z}}) = \prod_{i=1}^{n} \phi_\sigma(\bar{r}_{ii})$, we can conclude (4.44) holds from (4.34).

To show (4.46), instead of analyzing the probability of $\boldsymbol{z}^{\text{OB}}$ on its left-hand side, we will analyze an equivalent probability of $\bar{\boldsymbol{v}}$ as we know the distribution of $\bar{\boldsymbol{v}}$.

In our proof, we need to use the basic result: given $v \sim \mathcal{N}(0, \sigma^2)$ and $\eta > 0$, for any $\zeta \in \mathbb{R}$,

$$\Pr(v \in [\zeta, \zeta + \eta]) \leq \Pr(v \in [-\eta/2, \eta/2]) = \phi_\sigma(\eta). \tag{4.47}$$

By (3.1), the equality in (4.47) obviously holds. To prove the inequality, let

$$q(\zeta) = \int_{\zeta}^{\zeta+\eta} \exp\big(-\frac{t^2}{2\sigma^2}\big) dt.$$

Thus the inequality can be written as $q(\zeta) \leq q(-\eta/2)$. We can easily show that $q'(-\eta/2) = 0$, $q'(\zeta) > 0$ if $\zeta < -\eta/2$ and $q'(\zeta) < 0$ if $\zeta > -\eta/2$. Thus, the inequality holds.

From (2.17) and (2.7), we observe that $\boldsymbol{z}^{\text{OB}}$ is a function of $\bar{\boldsymbol{v}}$. To emphasize this, we write it as $\boldsymbol{z}^{\text{OB}}(\bar{\boldsymbol{v}})$. When $\bar{\boldsymbol{v}}$ changes, $\boldsymbol{z}^{\text{OB}}(\bar{\boldsymbol{v}})$ may change too. In the following analysis, we assume that $\hat{\boldsymbol{z}}$ is fixed and $\bar{\boldsymbol{v}}$ satisfies the model (2.23a).

For later uses, for $= n, n-1, \ldots, 1$, define

$$\mathcal{G}_i = \{\boldsymbol{w}_{i:n} |\, \bar{\boldsymbol{y}}_{i:n} = \bar{\boldsymbol{R}}_{i:n,i:n}\hat{\boldsymbol{z}}_{i:n} + \boldsymbol{w}_{i:n},\, z_k^{\mathrm{OB}}(\boldsymbol{w}_{i:n}) \in [\bar{l}_k, \bar{u}_k],\, k = i, i+1, \ldots, n\}. \quad (4.48)$$

From (4.48), it is easy to verify that $\boldsymbol{z}^{\mathrm{OB}}(\bar{\boldsymbol{v}}) \in \mathcal{F}$ if and only if $\bar{\boldsymbol{v}} \in \mathcal{G}_1$. Therefore, (4.46) is equivalent to

$$\Pr(\bar{\boldsymbol{v}}_{1:n} \in \mathcal{G}_1) \le \prod_{i=1}^{n} \phi_\sigma((\bar{u}_i - \bar{l}_i + 1)\bar{r}_{ii}). \quad (4.49)$$

We prove (4.49) by induction. First, we prove the base case:

$$\Pr(\bar{v}_n \in \mathcal{G}_n) \le \sigma((\bar{u}_n - \bar{l}_n + 1)\bar{r}_{nn}).$$

By (2.7) and (2.17), we have

$$c_n^{\mathrm{OB}} = \frac{\bar{y}_n}{\bar{r}_{nn}} = \frac{\bar{r}_{nn}\hat{z}_n + \bar{v}_n}{\bar{r}_{nn}} = \hat{z}_n + \frac{\bar{v}_n}{\bar{r}_{nn}}.$$

Since $z_n^{\mathrm{OB}}(\bar{v}_n) = \lfloor c_n^{\mathrm{OB}} \rceil$, by (4.48),

$$\begin{aligned}
\Pr(\bar{v}_n \in \mathcal{G}_n) &= \Pr\left(\hat{z}_n + \frac{\bar{v}_n}{\bar{r}_{nn}} \in [\bar{l}_n - 1/2, \bar{u}_n + 1/2]\right) \\
&= \Pr(\bar{v}_n \in [(\bar{l}_n - \hat{z}_n - 1/2)\bar{r}_{nn}, (\bar{u}_n - \hat{z}_n + 1/2)\bar{r}_{nn}]) \\
&\le \phi_\sigma((\bar{u}_n - \bar{l}_n + 1)\bar{r}_{nn})
\end{aligned}$$

where in deriving the inequality, we used (4.47).

Suppose for some $i > 1$, we have

$$\Pr(\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_i) \le \prod_{k=i}^{n} \phi_\sigma((\bar{u}_k - \bar{l}_k + 1)\bar{r}_{kk}). \quad (4.50)$$

Now we want to prove

$$\Pr(\bar{\boldsymbol{v}}_{i-1:n} \in \mathcal{G}_{i-1}) \leq \prod_{k=i-1}^{n} \phi_\sigma((\bar{u}_k - \bar{l}_k + 1)\bar{r}_{kk}). \tag{4.51}$$

For our proof, we partition the set $\mathcal{G}_i$ into a sequence of disjoint subsets. To do that, for $i = n, n-1, \ldots, 1$, we first define the discrete set

$$\mathcal{H}_i = \Big\{ \sum_{j=i}^{n} \frac{r_{i-1,j}}{r_{i-1,i-1}} (\hat{z}_j - z_j^{\mathrm{OB}}(\boldsymbol{w}_{j:n})) \Big| \boldsymbol{w}_{i:n} \in \mathcal{G}_i \Big\}.$$

Then, for any $t \in \mathcal{H}_i$, we define

$$\mathcal{G}_{i,t} = \Big\{ \boldsymbol{w}_{i:n} | \boldsymbol{w}_{i:n} \in \mathcal{G}_i \text{ such that } \sum_{j=i}^{n} \frac{r_{i-1,j}}{r_{i-1,i-1}} (\hat{z}_j - z_j^{\mathrm{OB}}(\boldsymbol{w}_{j:n})) = t \Big\}.$$

It is easy to verify that $\cup_{t \in \mathcal{H}_i} \mathcal{G}_{i,t} = \mathcal{G}_i$ and $\mathcal{G}_{i,t_1} \cap \mathcal{G}_{i,t_2} = \varnothing$ for $t_1, t_2 \in \mathcal{H}_i$ and $t_1 \neq t_2$. Therefore,

$$\Pr(\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_i) = \sum_{t \in \mathcal{H}_i} \Pr(\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_{i,t}) \tag{4.52}$$

and

$$\begin{aligned}
\Pr(\bar{\boldsymbol{v}}_{i-1:n} \in \mathcal{G}_{i-1}) &= \Pr(\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_i,\ z_{i-1}^{\mathrm{OB}}(\bar{\boldsymbol{v}}_{i-1:n}) \in [\bar{l}_{i-1}, \bar{u}_{i-1}]) \\
&= \sum_{t \in \mathcal{H}_i} \Pr(\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_{i,t},\ z_{i-1}^{\mathrm{OB}}(\bar{\boldsymbol{v}}_{i-1:n}) \in [\bar{l}_{i-1}, \bar{u}_{i-1}]) \\
&= \sum_{t \in \mathcal{H}_i} \Pr(\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_{i,t}) \Pr\left(z_{i-1}^{\mathrm{OB}}(\bar{\boldsymbol{v}}_{i-1:n}) \in [\bar{l}_{i-1}, \bar{u}_{i-1}] | \bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_{i,t}\right).
\end{aligned} \tag{4.53}$$

Now we derive a bound on the second probability of each term on the right-hand side of (4.53). By (2.7) and (2.17), we have

$$c_{i-1}^{\mathrm{OB}} = \frac{\bar{r}_{i-1,i-1}\hat{z}_{i-1} + \sum_{j=i}^{n} \bar{r}_{i-1,j}\hat{z}_j + \bar{v}_{i-1} - \sum_{j=i}^{n} \bar{r}_{i-1,j} z_j^{\mathrm{OB}}(\bar{\boldsymbol{v}}_{j:n})}{\bar{r}_{i-1,i-1}} = \hat{z}_{i-1} + t' + \frac{\bar{v}_{i-1}}{\bar{r}_{i-1,i-1}}$$

where

$$t' = \sum_{j=i}^{n} \frac{\bar{r}_{i-1,j}}{\bar{r}_{i-1,i-1}} (\hat{z}_j - z_j^{\mathrm{OB}}(\bar{\boldsymbol{v}}_{j:n})).$$

If $\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_{i,t}$ for some t, then $t' = t \in \mathcal{H}_i$. Since $z_{i-1}^{\mathrm{OB}}(\bar{\boldsymbol{v}}_{i-1:n}) = \lfloor c_{i-1}^{\mathrm{OB}} \rceil$,

$$\begin{aligned}
&\Pr\left(z_{i-1}^{\mathrm{OB}}(\bar{\boldsymbol{v}}_{i-1:n}) \in [\bar{l}_{i-1}, \bar{u}_{i-1}] | \bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_{i,t}\right) \\
&= \Pr\left(\hat{z}_{i-1} + t + \frac{\bar{v}_{i-1}}{\bar{r}_{i-1,i-1}} \in [\bar{l}_{i-1} - 1/2, \bar{u}_{i-1} + 1/2]\right) \\
&= \Pr\left(\bar{v}_{i-1} \in \left[(\bar{l}_{i-1} - \hat{z}_{i-1} - t - 1/2)\bar{r}_{i-1,i-1},\ (\bar{u}_{i-1} - \hat{z}_{i-1} - t + 1/2)\bar{r}_{i-1,i-1}\right]\right) \\
&\leq \phi_\sigma((\bar{u}_{i-1} - \bar{l}_{i-1} + 1)\bar{r}_{i-1,i-1}) \qquad (4.54)
\end{aligned}$$

where for the inequality we used (4.47). Thus, from (4.53) it follows that

$$\begin{aligned}
\Pr(\bar{\boldsymbol{v}}_{i-1:n} \in \mathcal{G}_{i-1}) &\leq \sum_{t \in \mathcal{H}_i} \Pr(\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_{i,t}) \phi_\sigma((\bar{u}_{i-1} - \bar{l}_{i-1} + 1)\bar{r}_{i-1,i-1}) \\
&= \Pr(\bar{\boldsymbol{v}}_{i:n} \in \mathcal{G}_i) \phi_\sigma((\bar{u}_{i-1} - \bar{l}_{i-1} + 1)\bar{r}_{i-1,i-1})
\end{aligned}$$

where the equality is due to (4.52). Then the inequality (4.51) follows by using the induction hypothesis (4.50). Therefore, the inequality (4.49), or the equivalent inequality (4.46), holds for any fixed $\hat{\boldsymbol{z}}$.

Since (4.46) holds for any fixed $\hat{\boldsymbol{z}}$, it is easy to argue that it holds no matter what distribution of $\hat{\boldsymbol{z}}$ is over the box $\mathcal{F}$, so the theorem is proved. $\square$

By Theorem 4.3.2, if $\prod_{i=1}^{n} \frac{\phi_\sigma(\bar{r}_{ii})}{\phi_\sigma((\bar{u}_i - \bar{l}_i + 1)\bar{r}_{ii})} \approx 1$, then $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B}) \approx 0$, i.e., the conjecture holds. The condition will be satisfied when the noise standard deviation σ is sufficiently small.

Here we make a comment on the upper bound in (4.44). The derivation of (4.44) was based on the two inequalities (4.45) and (4.46). The inequality (4.45) was established based on the fact that $\mathcal{E} \subseteq \mathcal{F}$ in (4.43). If the absolute values of the entries of the unimodular matrix $\boldsymbol{Z}^{-1}$ are big, then it is likely that $\mathcal{F}$ is much bigger than $\mathcal{E}$ although $\mathcal{F}$ is the smallest box to contain $\mathcal{E}$, making the inequality (4.45) loose. Otherwise it will be tight; in particular, when $\boldsymbol{Z} = \boldsymbol{I}$, then $\mathcal{E} = \mathcal{F}$ and the inequality (4.45) becomes an equality. In establishing the inequality (4.46) we used the inequality (4.47) (see (4.54)), which is simple but may not be tight if ζ is not close to $-\eta/2$. Thus the inequality (4.46) may not be tight. Overall, the upper bound in (4.44) may not be tight sometimes, but it is always tighter than than the upper bound in (4.41). The following example shows that the former can be significantly tighter than the latter and can be a sharp bound.

Example 4.3.2 *We use exactly the same data generated in Example 4.3.1 to compute the upper bounds in* (4.41) *and* (4.44)*, which are denoted by* μ_{eb1} *and* μ_{eb2}*, respectively. The results for* $n = 5 : 5 : 40$ *with* $\sigma = 0.1$ *are given in Table 4–11. To see how tight they are, the values of* P_e *given in Table 4–9 are displayed here again. Recall* P_e *is the experimental probability corresponding to the theoretical probability* $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B})$ *in* (4.41) *and* (4.44)*.*

From Table 4–11, we can see the upper bound μ_{eb2} *is obviously tighter than the upper bound* μ_{eb1} *and* μ_{eb2} *is close to* P_e*. When* $n = 10$*,* $P_e > \mu_{eb2}$*, this is because*

there are some deviations between the experimental values and the theoretical values. The values of μ_{eb1} are actually not exactly the same for different n, but they are very close. This is also true for μ_{eb2}.

Table 4–11: P_e and bounds versus $n = 5 : 5 : 40$ with $\sigma = 0.1$

$\boldsymbol{n}$	P_e	μ_{eb1}	μ_{eb2}
5	0.9356	0.9840	0.9364
10	0.9396	0.9840	0.9364
15	0.9336	0.9840	0.9364
20	0.9348	0.9840	0.9364
25	0.9325	0.9840	0.9364
30	0.9312	0.9840	0.9364
35	0.9304	0.9840	0.9364
40	0.9252	0.9840	0.9364

4.3.3 A modified stopping criterion

From Theorem 4.3.1 and Example 4.3.1, the stopping criterion developed in [58] may be too optimistic. However, we can use Theorem 4.3.2 to develop a more reliable stopping criterion, see Algorithm 4.3.1.

Algorithm 4.3.1 Modified Stopping Criterion

1: Use the LLL reduction to reduce (1.3a) to (2.17) and find the corresponding Babai point $\boldsymbol{z}^{\text{OB}}$ by using (2.7).

2: Define $\bar{\boldsymbol{x}} = \boldsymbol{Z}\boldsymbol{z}^{\text{OB}}$, if $\bar{\boldsymbol{x}} \in \mathcal{B}$ (see (1.3b)) and $\prod_{i=1}^{n} \frac{\phi_\sigma(\bar{r}_{ii})}{\phi_\sigma((\bar{u}_i - \bar{l}_i + 1)\bar{r}_{ii})}$ is close to 1 (see Theorem 4.3.2), then use $\bar{\boldsymbol{x}}$ to estimate $\hat{\boldsymbol{x}}$; otherwise, solve the BILS problem (1.4) to get the maximal likelihood estimator $\boldsymbol{x}^{\text{BL}}$ and use it to estimate $\hat{\boldsymbol{x}}$;

From Algorithm 4.3.1, if $\bar{\boldsymbol{x}} \notin \mathcal{B}$, but part entries of $\bar{\boldsymbol{x}}$ are in the corresponding constraint interval, then the BILS problem (1.4) should be solved to get $\boldsymbol{x}^{\text{BL}}$. Therefore, from this point of view, there is still room to improve the stopping criterion.

One method of dealing with this case is to use column permutation strategies instead of the LLL algorithm to reduce (1.4), establish a theorem which is similar to Theorem 4.3.2 to bound $\Pr(\bar{\boldsymbol{x}} \neq \hat{\boldsymbol{x}} | \bar{\boldsymbol{x}} \in \mathcal{B})$ and use partial validation techniques to do the estimation. We leave this part to the interesting readers.

From the definition of $\mathcal{E}$ (see the paragraph above (4.42)), if $\bar{\boldsymbol{x}} \in \mathcal{B}$, then $\boldsymbol{z}^{\text{OB}} \in \mathcal{E}$, by (4.43), $\boldsymbol{z}^{\text{OB}} \in \mathcal{F}$. Therefore, instead of finding the Babai point $\boldsymbol{z}^{\text{OB}}$ by using (2.7), we can find the box-constrained Babai point $\boldsymbol{z}^{\text{BB}}$ with the constraint box $\mathcal{F}$ via (2.8) and assume $\bar{\boldsymbol{x}} = \boldsymbol{Z}\boldsymbol{z}^{\text{BB}}$. Note that, in practice, this modification probably can not bring much improvement of the effectiveness of the stopping criterion if the box $\mathcal{B}$ is large, because in this case the box $\mathcal{F}$ may be very large.

CHAPTER 5
An Efficient Algorithm for an SVP Problem in Computer-and-Forward Protocol Design

In this chapter, we consider the problem of finding the optimal coefficient vector that maximizes the computation rate at a relay in the computer-and-forward scheme. Based on the idea of sphere decoding, we propose a very efficient method that gives the optimal coefficient vector, i.e., solves (1.5). First, we derive an algorithm with only $\mathcal{O}(n)$ flops, to compute the Cholesky factorization of $\boldsymbol{G}$ in (1.5) (we do not form the whole Cholesky factor $\boldsymbol{R}$ explicitly), to transform (1.5) to a SVP. Then, we propose some conditions that can be checked by $\mathcal{O}(n)$ flops, under which the optimal coefficient vector $\boldsymbol{a}^\star$ can be obtained immediately without using any search algorithm. After that, by taking into account some resultant useful properties of $\boldsymbol{a}^\star$, we modify the Schnorr-Euchner search algorithm to solve the SVP. Simulation results show that the average cost of our new algorithm is $\mathcal{O}(n^{1.5})$ flops for i.i.d. Gaussian channel entries, and our algorithm is not only much more efficient than the existing ones that give the optimal solution, but also faster than some of the suboptimal methods. Part of the contributions of this chapter appear in [92] and [93] is a more complete version which includes all of the main contributions.

5.1 Introduction of compute-and-forward

In relay networks, compute-and-forward (CF) [67] is a promising relaying strategy that can offer higher rates than traditional ones (e.g., amplify-and-forward,

decode-and-forward), especially in the moderate SNR regime. The crucial idea of CF is the application of linear/lattice codes [99] and physical layer network coding (PLNC) [52]. The received signal at a relay is the linear combination of a set of transmitted signals, where the linear combination coefficients form the channel vector from the involved sources to that relay. By multiplying the channel vector by an amplifying factor, the obtained new channel vector can be close to a coefficient vector with all integer-valued entries. This means that after applying an appropriate amplifying factor to the received signal at a relay, it will be approximately an integer linear combination of the transmitted signals. Since the same linear code is used at the sources, an integer linear combination of valid codewords is still a valid codeword, which means the aforementioned integer linear combination of the transmitted signals is possible to be successfully decoded as a linear combination of the messages corresponding to the transmitted signals. Under certain conditions, with enough such decoded linear combinations of the transmitted messages, the transmitted messages can be recovered.

Obviously, the amplifying factors and the integer-valued coefficient vectors need to be carefully designed. When Nazer and Gastpar [67] proposed the CF scheme, they defined the computation rate and set it as the metric for designing the amplifying factor and the integer-valued coefficient vector. Roughly speaking, computation rate refers to the maximum transmission rate at the involved sources of a relay such that the combined signals at the relay can be reliably decoded. It has been pointed out that setting the amplifying factor at a relay as the minimum-mean-square-error (MMSE) coefficient can maximize the computation rate at that relay. The difficulty

lies in the design of the coefficient vectors. To optimize the performance of the whole system, the coefficient vectors should be designed jointly. However, this requires each relay to know the channel state information (CSI) at other relays. Also, it could be far too complex. One suboptimal choice is to design the coefficient vector of each relay with the criterion being maximizing the computation rate of that relay. This is reasonable when only the local CSI at each relay is available. Unfortunately, the problem is not trivial even for this case, as it turns out to be a classical shortest vector problem (SVP) in a lattice. In this chapter, we will focus on this case, i.e., finding the optimal coefficient vector at a relay such that the computation rate at that relay is maximized.

Various methods have been proposed for designing the coefficient vectors. The Fincke-Phost method [24] was modified in [89] to solve a different but related problem, leading to the optimal coefficient vector and some other suboptimal vectors. A branch-and-bound algorithm, which uses part of the properties of the optimal vector, was used in [71]. But it appears that this algorithm is not efficient in this application. There are methods that give suboptimal solutions. Three methods were proposed in [73]: the one based on the complex LLL [27], the simple quantized search method and the iterative MMSE-based quantization method. Although the average complexity of the LLL algorithm [51] is polynomial if the entries of the basis vectors independently follow the normal distribution $\mathcal{N}(0,1)$ (see, e.g., [42], [55]), the cost of the first method maybe too high since it was proved in [42] that in the MIMO context, the worst-case complexity of the LLL algorithm is not even finite. The last two methods are of low complexity, but they may fail to offer satisfactory performance especially

when the dimension is large. Besides these, the quadratic programming relaxation method in [101] and its improvement in [102], are of relatively low complexity. Although their performance in terms of the computation rate are better than that of the last two methods proposed in [73], the differences between their performance and that of the optimal methods are still large when the dimension is large and the SNR is high.

In this chapter, we will propose an efficient algorithm for finding the optimal coefficient vector that maximizes the computation rate at a relay. First, we will derive an efficient algorithm with only $\mathcal{O}(n)$ flops to transform the problem to a standard SVP by fully using the structure of the matrix to compute its Cholesky factorization (we do not form the whole Cholesky factor $\boldsymbol{R}$ explicitly). We will also propose some conditions that can be checked with $\mathcal{O}(n)$ flops, under which the optimal coefficient vector can be obtained immediately without using any search algorithm. Then, we will propose a modified Schnorr-Euchner search algorithm to solve the SVP by taking advantage of the properties of the optimal solution. Simulation results indicate that the average cost of our new algorithm is $\mathcal{O}(n^{1.5})$ flops for i.i.d. Gaussian channel entries.

We have noticed that very recently, an algorithm with the complexity of $\mathcal{O}(n^{2.5})$ flops has been proposed in [72]. This algorithm finds the optimal solution by solving an optimization problem with one variable over a bounded region. Our proposed algorithm is totally different from this one and our simulations indicate that ours is much faster.

5.2 Problem statement

We consider the problem of finding the optimal coefficient vector that maximizes the *computation rate* (defined in [67]) at a relay in the CF scheme. The application scenario we focus on is the additive white Gaussian noise (AWGN) network, where sources, relays and destinations are linked with linear channels with AWGN. For the ease of explanation, we will focus on the real-valued channel model in the sequel.

Definition 5.2.1 (Channel Model) *Each relay (indexed by $i = 1, 2, \ldots, m$) observes a noisy linear combination of the transmitted signals through the* channel*,*

$$\boldsymbol{y}_i = \sum_{j=1}^{n} \boldsymbol{h}_i(j)\boldsymbol{x}_j + \boldsymbol{z}_i,$$

where $\boldsymbol{x}_j \in \mathbb{R}^N$ with the power constraint $\frac{1}{N}\|\boldsymbol{x}_j\|_2^2 \leq P$ is the transmitted codeword from source j ($j = 1, 2, \ldots, n$), $\boldsymbol{h}_i \in \mathbb{R}^n$ is the channel vector to relay i (here $\boldsymbol{h}_i(j)$ denotes the j-th entry of $\boldsymbol{h}_i$), $\boldsymbol{z}_i \in \mathbb{R}^N$ is the noise vector with entries being i.i.d. Gaussian, i.e., $\boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, and $\boldsymbol{y}_i$ is the signal received at relay i.

For relay i with the channel vector $\boldsymbol{h}_i$, let $\boldsymbol{a}_i$ be the chosen coefficient vector, the computation rate is calculated according to the following theorem [67].

Theorem 5.2.1 *The computation rate at relay i is uniquely maximized by choosing the amplifying factor as the MMSE coefficient, which results in a computation rate*

$$\mathcal{R}(\boldsymbol{h}_i, \boldsymbol{a}_i) = \frac{1}{2}\log^+\left(\frac{1}{\|\boldsymbol{a}_i\|^2 - \frac{P\left(\boldsymbol{h}_i^T\boldsymbol{a}_i\right)^2}{1+P\|\boldsymbol{h}_i\|_2^2}}\right), \tag{5.1}$$

where log *function is with respect to base 2 and* $\log^+(x) \triangleq \max(\log(x), 0)$.

Also, we define the optimal coefficient vector as below.

Definition 5.2.2 (The Optimal Coefficient Vector) *The* optimal coefficient vector $\boldsymbol{a}_i^\star$ *for a channel vector* $\boldsymbol{h}_i$ *is the one that maximizes the computation rate,*

$$\boldsymbol{a}_i^\star = \arg \max_{\boldsymbol{a}_i \in \mathbb{Z}^n \setminus \{\boldsymbol{0}\}} \mathcal{R}\left(\boldsymbol{h}_i, \boldsymbol{a}_i\right). \tag{5.2}$$

The optimization problem (5.2) can be further formulated as the following quadratic form [89]:

$$\boldsymbol{a}_i^\star = \arg \min_{\boldsymbol{a}_i \in \mathbb{Z}^n \setminus \{\boldsymbol{0}\}} \boldsymbol{a}_i^T \boldsymbol{G}_i \boldsymbol{a}_i, \tag{5.3a}$$

$$\boldsymbol{G}_i = \boldsymbol{I} - \frac{P}{1 + P\|\boldsymbol{h}_i\|_2^2} \boldsymbol{h}_i \boldsymbol{h}_i^T. \tag{5.3b}$$

Hereafter, we will ignore the subscript "i", e.g., $\boldsymbol{h}_i$ will be directly written as $\boldsymbol{h}$. In the next section, we will propose an efficient method based on sphere decoding to solve (5.3).

5.3 Proposed method

In this section, we will derive an efficient algorithm to solve (5.3).

Define the *scaled channel vector* $\boldsymbol{t}$ as

$$\boldsymbol{t} = \sqrt{\frac{P}{1 + P\|\boldsymbol{h}\|_2^2}} \boldsymbol{h}, \tag{5.4}$$

then, (5.3) is equivalent to the following problem:

$$\boldsymbol{a}^\star = \arg \min_{\boldsymbol{a} \in \mathbb{Z}^n \setminus \{\boldsymbol{0}\}} \boldsymbol{a}^T \boldsymbol{G} \boldsymbol{a}, \tag{5.5a}$$

$$\boldsymbol{G} \triangleq \boldsymbol{I} - \boldsymbol{t}\boldsymbol{t}^T. \tag{5.5b}$$

Obviously, $\|\boldsymbol{t}\|_2 < 1$ and $\boldsymbol{G}$ is symmetric positive definite. Throughout this chapter, we assume $\boldsymbol{h} \neq \boldsymbol{0}$; otherwise, it is trivial.

The problem in (5.5) can be solved via two steps:

- First, for a given $\boldsymbol{t}$, computes $\boldsymbol{G}$ and finds its Cholesky factorization, i.e., finds an upper triangular matrix $\boldsymbol{R}$ such that $\boldsymbol{G} = \boldsymbol{R}^T\boldsymbol{R}$. Then (5.5) is equivalent to the following shortest vector problem (SVP):

$$\boldsymbol{a}^\star = \arg \min_{\boldsymbol{a} \in \mathbb{Z}^n \setminus \{\boldsymbol{0}\}} \|\boldsymbol{R}\boldsymbol{a}\|_2. \tag{5.6}$$

- Second, use a search algorithm, such as the LLL-aided Schnorr-Euchner search strategy [74], to solve (5.6). We will explain the details later.

It is easy to see that for a given $\boldsymbol{t}$, computing $\boldsymbol{G}$ costs $\mathcal{O}(n^2)$ flops. Besides, it is well-known that computing the Cholesky factorization of a general $n \times n$ matrix costs $\mathcal{O}(n^3)$ flops. Moreover, the cost of the LLL-aided Schnorr-Euchner search strategy [74] for solving (5.6) maybe too high. Fortunately, we find out that it is possible to accelerate the two steps mentioned above as follows:

- First, take advantage of the special structure of $\boldsymbol{G}$ in (5.5b) to compute its Cholesky factorization and transform (5.5) to (5.6), but do not form $\boldsymbol{G}$, the whole $\boldsymbol{R}$ and the SVP explicitly.
- Second, investigate the properties of a solution $\boldsymbol{a}^*$ to (5.6) and take them into account to modify the Schnorr-Euchner search strategy [74] to find $\boldsymbol{a}^\star$.

If $\boldsymbol{a}^\star$ is a solution of (5.6), then $-\boldsymbol{a}^\star$ is also a solution. To reduce redundancy, we apply the following restriction.

Restriction 5.3.1 *Throughout this chapter, we restrict the solution* $\boldsymbol{a}^\star$ *to* (5.6) *such that* $\boldsymbol{t}^T\boldsymbol{a}^\star \geq 0$.

5.3.1 Transformation of the problem

To transform (5.5) to the SVP (5.6), we need to find the Cholesky factorization of $\boldsymbol{G}$ in (5.5b). Besides the regular method, one can use the algorithm proposed in [13], which costs $2n^2+\mathcal{O}(n)$ flops, to get the Cholesky factor $\boldsymbol{R}$. However, the cost can be further reduced. Also, to analyze the complexity of our proposed search algorithm in Section 5.4, we need to know the diagonal entries of $\boldsymbol{R}$. In this subsection, we will take into account the special structure of $\boldsymbol{G}$ to achieve this goal with only $\mathcal{O}(n)$ flops (we do not form the whole $\boldsymbol{R}$ explicitly. If the whole $\boldsymbol{R}$ is needed for other applications, it costs $n^2/2+\mathcal{O}(n)$ flops). Based on the diagonal entries of $\boldsymbol{R}$ and by investigating their properties, we will also propose some conditions that can be checked by $\mathcal{O}(n)$ flops, under which the optimal solution $\boldsymbol{a}^\star$ can be obtained immediately without using any search algorithm.

Our proposed algorithm to find the Cholesky factor $\boldsymbol{R}$ of $\boldsymbol{G}$ in (5.5b) is based on the following theorem:

Theorem 5.3.1 *The Cholesky factor $\boldsymbol{R}$ of $\boldsymbol{G}$ in* (5.5b) *is given by:*

$$r_{ij}=\begin{cases}\sqrt{\frac{1-\sum_{l=1}^{i}t_l^2}{1-\sum_{l=1}^{i-1}t_l^2}}, & j=i\\ \frac{-t_it_j}{\sqrt{1-\sum_{l=1}^{i-1}t_l^2}\sqrt{1-\sum_{l=1}^{i}t_l^2}}, & i<j\leq n\end{cases}, \tag{5.7}$$

where $1\leq i\leq n$ and denote $\sum_1^0\cdot=0$.

Proof. To prove the theorem, we show any element of $\boldsymbol{G}$ is equal to the corresponding element of $\boldsymbol{R}^T\boldsymbol{R}$ in the same position, i.e., by (5b), we would like to show

$$\sum_{i=1}^{k}r_{ik}^2=1-t_k^2,\ \ 1\leq k\leq n \tag{5.8}$$

and

$$\sum_{i=1}^{k} r_{ik} r_{ij} = -t_k t_j, \;\; 1 \le k < j \le n. \tag{5.9}$$

By (5.7), we have

$$\begin{aligned}
\sum_{i=1}^{k} r_{ik}^2 &= r_{kk}^2 + \sum_{i=1}^{k-1} r_{ik}^2 \\
&= \frac{1 - \sum_{l=1}^{k} t_l^2}{1 - \sum_{l=1}^{k-1} t_l^2} + \sum_{i=1}^{k-1} \frac{t_i^2 t_k^2}{(1 - \sum_{l=1}^{i-1} t_l^2)(1 - \sum_{l=1}^{i} t_l^2)} \\
&= \frac{1 - \sum_{l=1}^{k} t_l^2}{1 - \sum_{l=1}^{k-1} t_l^2} + \frac{t_k^2 \sum_{l=1}^{k-1} t_l^2}{1 - \sum_{l=1}^{k-1} t_l^2} = 1 - t_k^2,
\end{aligned}$$

and

$$\begin{aligned}
\sum_{i=1}^{k} r_{ik} r_{ij} &= r_{kk} r_{kj} + \sum_{i=1}^{k-1} r_{ik} r_{ij} \\
&= \frac{-t_k t_j}{1 - \sum_{i=1}^{k-1} t_i^2} + \frac{\sum_{i=1}^{k-1} t_i^2 t_k t_j}{(1 - \sum_{l=1}^{i-1} t_l^2)(1 - \sum_{l=1}^{i} t_l^2)} \\
&= \frac{-t_k t_j}{1 - \sum_{i=1}^{k-1} t_i^2} + \frac{t_k t_j \sum_{i=1}^{k-1} t_i^2}{1 - \sum_{i=1}^{k-1} t_i^2} = -t_k t_j.
\end{aligned}$$

Thus, both (5.8) and (5.9) hold, completing the proof. □

We can use Theorem 5.3.1 to design an efficient algorithm to find $\boldsymbol{R}$. To simplify notation, we introduce an $n-$dimensional vector variable $\boldsymbol{f}$. Let

$$f_0 = 1, \quad f_i = 1 - \sum_{l=1}^{i} t_l^2, \quad 1 \le i \le n. \tag{5.10}$$

Then by (5.7), we have

$$r_{ii} = \sqrt{f_i / f_{i-1}}, \quad 1 \le i \le n,$$

and

$$\boldsymbol{R}_{i,i+1:n} = (-t_i/\sqrt{f_i f_{i-1}})\boldsymbol{t}_{i+1:n}^T, \quad 1 \le i < n.$$

Note that $\boldsymbol{R}_{i,i+1:n}$ is a scaled $\boldsymbol{t}_{i+1:n}^T$.

After getting $\boldsymbol{R}$, we will modify the Schnorr-Euchner search algorithm to solve (5.6). Later we will see that it is not necessary to form $\boldsymbol{R}$ explicitly (we will give more details to explain this in the last subsection of this section), i.e, we do not need to compute the multiplication of $-t_i/\sqrt{f_i f_{i-1}}$ and $\boldsymbol{t}_{i+1:n}$. Thus, the cost is $\mathcal{O}(n)$ flops.

By (5.7), it is easy to see that the entries of $\boldsymbol{R}$ have the following properties:

Theorem 5.3.2 *For $1 \le k \le n$, the following inequalities hold:*

$$\sqrt{1 - \sum_{i=1}^{k} t_i^2} \le r_{kk} \le \sqrt{1 - t_k^2}, \tag{5.11}$$

$$\prod_{i=k}^{n} r_{ii} = \frac{\sqrt{1 - \|\boldsymbol{t}\|_2^2}}{\sqrt{1 - \sum_{i=1}^{k-1} t_i^2}} \ge \sqrt{1 - \|\boldsymbol{t}\|_2^2}. \tag{5.12}$$

Under Theorem 5.3.1, we have the following interesting result, which can be used to describe the geometry of the search space later.

Theorem 5.3.3 *For $1 \le i < j \le n$, the eigenvalues of $\boldsymbol{R}_{i:j,i:j}^T \boldsymbol{R}_{i:j,i:j}$ are f_j/f_{i-1} ($\boldsymbol{f}$ is defined in* (5.10)*) and 1 with algebraic multiplicity $j - i$.*

Proof. We first prove

$$\boldsymbol{R}_{i:n,i:n}^T \boldsymbol{R}_{i:n,i:n} = \boldsymbol{I}_{n-i+1} - \frac{\boldsymbol{t}_{i:n}}{\sqrt{f_{i-1}}} \frac{\boldsymbol{t}_{i:n}^T}{\sqrt{f_{i-1}}}. \tag{5.13}$$

If $i = 1$, then by Theorem 5.3.1 and (5.10), (5.13) holds. So we only need to prove it holds for $i > 1$.

By Theorem 5.3.1, we have

$$\boldsymbol{R}_{i:n,i:n}^T \boldsymbol{R}_{i:n,i:n} = \boldsymbol{I}_{n-i+1} - \boldsymbol{t}_{i:n}\boldsymbol{t}_{i:n}^T - \boldsymbol{R}_{1:i-1,i:n}^T \boldsymbol{R}_{1:i-1,i:n},$$
$$\boldsymbol{R}_{1:i-1,i:n} = (\boldsymbol{t}_{i:n} \begin{bmatrix} \frac{-t_1}{\sqrt{f_0 f_1}} & \frac{-t_2}{\sqrt{f_1 f_2}} & \dots & \frac{-t_{i-1}}{\sqrt{f_{i-2} f_{i-1}}} \end{bmatrix})^T.$$

Thus, we have

$$\boldsymbol{R}_{i:n,i:n}^T \boldsymbol{R}_{i:n,i:n} = \boldsymbol{I}_{n-i+1} - (1 + \sum_{k=1}^{i-1} \frac{t_k^2}{f_k f_{k-1}}) \boldsymbol{t}_{i:n}\boldsymbol{t}_{i:n}^T.$$

By (5.10),

$$\frac{t_k^2}{f_k f_{k-1}} = \frac{1}{f_k} - \frac{1}{f_{k-1}}.$$

Therefore,

$$\boldsymbol{R}_{i:n,i:n}^T \boldsymbol{R}_{i:n,i:n} = \boldsymbol{I}_{n-i+1} - \frac{\boldsymbol{t}_{i:n}}{\sqrt{f_{i-1}}} \frac{\boldsymbol{t}_{i:n}^T}{\sqrt{f_{i-1}}},$$

i.e., (5.13) holds.

From (5.13), we can immediately get

$$\boldsymbol{R}_{i:j,i:j}^T \boldsymbol{R}_{i:j,i:j} = \boldsymbol{I}_{j-i+1} - \frac{\boldsymbol{t}_{i:j}}{\sqrt{f_{i-1}}} \frac{\boldsymbol{t}_{i:j}^T}{\sqrt{f_{i-1}}}.$$

Thus, the eigenvalues of $\boldsymbol{R}_{i:j,i:j}^T \boldsymbol{R}_{i:j,i:j}$ are 1's and

$$1 - \frac{\sum_{k=i}^{j} t_k^2}{f_{i-1}} = \frac{f_j}{f_{i-1}}.$$

□

Generally speaking, after getting $\boldsymbol{R}$, a search algorithm should be used to find the solution $\boldsymbol{a}^\star$ to (5.6). Theorem 5.3.1 gives the closed-form expression of $\boldsymbol{R}$, so a natural question is whether there exist some easily-checked conditions, under which the optimal solution $\boldsymbol{a}^\star$ can be obtained without using any search algorithm? In the following, we will answer this question.

Theorem 5.3.4 *The optimal solution $\boldsymbol{a}^\star$ satisfies*

$$\|\boldsymbol{R}\boldsymbol{a}^\star\|_2 \geq \min_{1\leq i\leq n} \sqrt{\frac{1-\sum_{j=1}^{i} t_j^2}{1-\sum_{j=1}^{i-1} t_j^2}} \geq \sqrt{1-\|\boldsymbol{t}\|_2^2}. \tag{5.14}$$

Furthermore, if we have

$$t_i^2 \leq t_1^2(1-\sum_{l=1}^{i-1} t_l^2), \qquad i=2,3\ldots,n, \tag{5.15}$$

then $\boldsymbol{e}_1$ is a solution to (5.6).

Proof. The first inequality in (5.14) follows directly from (5.7) and

$$\|\boldsymbol{R}\boldsymbol{a}^\star\|_2 \geq \min_{1\leq i\leq n} r_{ii}, \tag{5.16}$$

which was given in [63, pp.99].

By the first inequality in (5.11),

$$\min_{1\leq i\leq n} r_{ii} \geq \sqrt{1-\|\boldsymbol{t}\|_2^2}.$$

Therefore, the second inequality in (5.14) follows.

If (5.15) holds, then by some simple calculations, we have:

$$\min_{1\leq i\leq n} \sqrt{\frac{1-\sum_{j=1}^{i} t_j^2}{1-\sum_{j=1}^{i-1} t_j^2}} = r_{11},$$

and the first inequality in (5.14) becomes an equality with $\boldsymbol{a}^\star = \boldsymbol{e}_1$. $\square$

It is easy to see that (5.15) can be checked by $\mathcal{O}(n)$ flops.

5.3.2 Reordering the entries of $\boldsymbol{t}$

After getting (5.6), a search algorithm, such as the Schnorr-Euchner search strategy [74] can be used to solve it. For efficiency, lattice reduction for $\boldsymbol{R}$ in (5.6) is usually used to strive for

$$r_{11} \leq r_{22} \leq \ldots \leq r_{nn} \tag{5.17}$$

to accelerate searching. Notice that (15) may not be achievable. For more details on why (5.17) should be strived, readers are referred to, e.g., [2] and [18].

The LLL reduction [51] is a commonly used reduction method to strive for (5.17). However, for this application, it has two main drawbacks. First, its complexity is high. In fact, it was shown in [42] that in the MIMO context, the worst-case cost is not even finite. For more details, see, e.g., [51], [22] and [55]. Also, from the simulation results in Section 5.5, we will see that the cost of the LLL reduction is even higher than that of our proposed algorithm. Second, it may destroy the structure of $\boldsymbol{R}$ and some properties of the optimal solution $\boldsymbol{a}^\star$ to the reduced problem (we will explain this in the latter part of this subsection). In this subsection, we will propose a method to strive for (5.17) without the above shortcomings.

From (5.7), to strive for (5.17), we permute the entries of $\boldsymbol{t}$. To make r_{11} as small as possible, we permute $\boldsymbol{t}$ such that $|t_1|$ is the largest. Suppose that $t_j, 1 \leq j \leq i$ have been fixed, then from (5.7), $r_{jj}, 1 \leq j \leq i$ are fixed. To make $r_{j+1,j+1}$ as small as possible, we permute the entries of $t_j, i+1 \leq j \leq n$ such that $|t_{j+1}|$ is the largest.

So after the permutations we have

$$|t_1| \geq |t_2| \geq \ldots \geq |t_n|. \tag{5.18}$$

Here we want to point out the above idea of reordering the entries of $\boldsymbol{t}$ is actually the same as that of SQRD [95], a column reordering strategy for a general matrix in the box-constrained integer least squares (BILS) problem [16], [90]. It is interesting to note that if we use the idea of V-BLAST [25], another column reordering strategy used in solving BILS problems [21], we will get the same ordering of $\boldsymbol{t}$. In fact, by (5.7),

$$r_{nn}^2 = \frac{1 - \|\boldsymbol{t}\|_2^2}{1 - \|\boldsymbol{t}\|_2^2 + t_n^2}.$$

Thus, to make r_{nn} as large as possible, we need to permute $\boldsymbol{t}$ such that $|t_n|$ is the smallest. Suppose that $t_j, i+1 \leq j \leq n$ have been fixed, then from (5.7), $r_{jj}, i+1 \leq j \leq n$ are fixed. By (5.7),

$$r_{ii}^2 = \frac{1 - \|\boldsymbol{t}\|_2^2 + \sum_{j=i+1}^n t_j^2}{1 - \|\boldsymbol{t}\|_2^2 + \sum_{j=i+1}^n t_j^2 + t_i^2}.$$

Thus, to make r_{jj} as large as possible, we permute the entries of $t_j, 1 \leq j \leq i$ such that $|t_i|$ is the smallest. So after the permutations we also have (5.18).

To make the late search process faster, we also want to make $t_i \geq 0$ for $1 \leq i \leq n$. This can easily be done. In fact, when we determine the i-th entry of $\boldsymbol{t}$ in the permutation process, we can use a sign permutation matrix so that the new i-th entry is nonnegative. Thus, eventually we have

$$t_1 \geq t_2 \geq \ldots \geq t_n \geq 0. \tag{5.19}$$

The above process can be described mathematically as follows. For a given $\boldsymbol{t}$, we can find a signed permutation matrix $\boldsymbol{Z} \in \mathbb{Z}^{n\times n}$ such that $\bar{\boldsymbol{t}} = \boldsymbol{Z}\boldsymbol{t}$ satisfying:

$$\bar{t}_1 \geq \bar{t}_2 \geq \ldots \geq \bar{t}_n \geq 0.$$

This transformation is a sorting process and the cost is $\mathcal{O}(n\log(n))$, see [101] for more details. Note that $\boldsymbol{Z}\boldsymbol{Z}^T = \boldsymbol{I}$. Then, with $\bar{\boldsymbol{a}} = \boldsymbol{Z}\boldsymbol{a}$, the optimization problem (5.5) can be transformed to

$$\begin{aligned}\bar{\boldsymbol{a}}^\star &= \min_{\bar{\boldsymbol{a}}\in\mathbb{Z}^n\setminus\{\boldsymbol{0}\}} \bar{\boldsymbol{a}}^T\bar{\boldsymbol{G}}\bar{\boldsymbol{a}},\\ \bar{\boldsymbol{G}} &\triangleq \boldsymbol{I} - \bar{\boldsymbol{t}}\bar{\boldsymbol{t}}^T.\end{aligned}$$

Obviously $\boldsymbol{a}^\star = \boldsymbol{Z}^T\bar{\boldsymbol{a}}^\star$.

Therefore, for a given $\boldsymbol{t}$, we use the above method to transform it such that the transformed $\boldsymbol{t}$ satisfies (5.19). For the sake of convenience, in our later analysis, we assume the entries of $\boldsymbol{t}$ satisfy (5.19).

Except speeding up the search, ordering the entries of $\boldsymbol{t}$ like in (5.19) has another important effect, i.e., by the results in [71] and [101], if (5.19) holds, we can find a solution $\boldsymbol{a}^\star$ to (5.6) such that

$$a_1^\star \geq a_2^\star \geq \ldots \geq a_n^\star \geq 0. \tag{5.20}$$

The order of the elements of the solution $\boldsymbol{a}^*$ in (5.20) is a key property of the solution we restricted for (5.6). It has been used in [101] to find a suboptimal solution to (5.6), but only the property that $a_i \geq 0, 1 \leq i \leq n$ has been used in [71] to solve (5.6). In this chapter, we will fully use it in designing the search algorithm. Note

that, if the LLL reduction is used for reducing $\boldsymbol{R}$ in (5.6), then (5.20) may not hold, which is the second drawback of using the LLL reduction in striving for (5.19). The motivation for reordering the entries of $\boldsymbol{t}$ in [71] and [101] is to obtain the property (5.20), which was (partially) used in their methods. Here we gave another motivation from the search point of view.

Under (5.19) and Theorem 5.3.1, we have the following interesting results:

Theorem 5.3.5 *If* (5.19) *holds, then for* $1 \leq i \leq n-1$

$$r_{ii} \leq \|\boldsymbol{R}_{i:i+1,i+1}\|_2 \leq \|\boldsymbol{R}_{i:i+2,i+2}\|_2 \leq \ldots \leq \|\boldsymbol{R}_{i:n,n}\|_2 \tag{5.21}$$

Proof. By (5.7), for $1 \leq i < j \leq n$,

$$\begin{aligned}
\|\boldsymbol{R}_{i:j,j}\|^2 =& \sum_{k=i}^{j-1} r_{kj}^2 + r_{jj}^2 \\
=& \sum_{k=i}^{j-1} \frac{t_k^2 t_j^2}{(1-\sum_{l=1}^{k-1} t_l^2)(1-\sum_{l=1}^{k} t_l^2)} + \frac{(1-\sum_{l=1}^{j} t_l^2)}{(1-\sum_{l=1}^{j-1} t_l^2)} \\
=& \frac{t_j^2}{(1-\sum_{l=1}^{j-1} t_l^2)} - \frac{t_j^2}{(1-\sum_{l=1}^{i-1} t_l^2)} + \frac{(1-\sum_{l=1}^{j} t_l^2)}{(1-\sum_{l=1}^{j-1} t_l^2)} \\
=& 1 - \frac{t_j^2}{(1-\sum_{l=1}^{i-1} t_l^2)}.
\end{aligned}$$

By the aforementioned equations, (5.7) and (5.19), it is easy to see that (5.21) holds. □

5.3.3 Schnorr-Euchner search algorithm

We first introduce a depth-first tree search algorithm: the Schnorr-Euchner search algorithm [74], [2], a variation of the Fincke-Pohst search strategy [24], to solve a general SVP, which has the form of (5.6). Note that, the Schnorr-Euchner

algorithm is generally more efficient than the Fincke-Phost, for their comparisons, see, e.g., [2]. Then we modify it by using the properties of $\boldsymbol{R}$ and the optimal solution $\boldsymbol{a}^\star$ to make the search process faster..

Let the optimal solution be within the following hyper-ellipsoid:

$$\|\boldsymbol{R}\boldsymbol{a}\|_2^2 < \beta^2, \tag{5.22}$$

where β is a constant. Define

$$d_n = 0,\ d_k = -\frac{1}{r_{kk}} \sum_{j=k+1}^{n} r_{kj} a_j, \quad k = n-1, \ldots, 1. \tag{5.23}$$

Then (5.22) can be written as:

$$\sum_{i=1}^{n} r_{ii}^2 (a_i - d_i)^2 < \beta^2$$

which is equivalent to

$$r_{kk}^2 (a_k - d_k)^2 < \beta^2 - \sum_{j=k+1}^{n} r_{jj}^2 (a_i - d_j)^2 \tag{5.24}$$

for $k = n, n-1, \ldots, 1$, where k is called the level index and $\sum_{j=n+1}^{n} \cdot = 0$.

Based on (5.24), the Schnorr-Euchner search algorithm can be described as follows. First we set the initial $\beta = \infty$, and for $k = n, n-1, \ldots, 1$, we compute d_k by (5.23) and set $a_k = \lfloor d_k \rceil$, leading to $a_k = 0$, for which (5.24) holds. So we obtain an integer vector $\boldsymbol{a} = \boldsymbol{0}$. Since the optimal solution $\boldsymbol{a}^\star$ is a nonzero vector, we need to update $\boldsymbol{a}$. Specifically, we set a_1 as the next closest integer to d_1. Note that (5.24) with $k = 1$ holds for the updated $\boldsymbol{a}$. Then, we store this updated $\boldsymbol{a}$ and set $\beta = \|\boldsymbol{R}\boldsymbol{a}\|_2$. After this, we try to find an integer vector within the new ellipsoid

by updating the latest found $\boldsymbol{a}$. Obviously, we cannot update only its first entry a_1, since we cannot find any new integer a_1 that satisfies (5.24) with $k = 1$, which is now an equality for the current $\boldsymbol{a}$. Thus we move up to level 2 to try to update a_2 by choosing it being the next nearest integer to d_2. If it satisfies (5.24) with $k = 2$, we move down to level 1 to update a_1 by computing d_1 ((5.23)) and setting $a_1 = \lfloor d_1 \rceil$ and then checking if (5.24) with $k = 1$ holds and so on; otherwise we move up to level 3 to try to update a_3, and so on. Finally, when we fail to find a new value for a_n to satisfy (5.24) with $k = n$, the search process stops and the latest found integer vector is the optimal solution $\boldsymbol{a}^\star$ we seek. This is a depth-first tree search. For more details, see, e.g., [2] and [16].

We summarize the search process in Algorithm 5.3.1, where

$$\operatorname{sgn}(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0 \end{cases}. \tag{5.25}$$

Algorithm 5.3.1 Schnorr-Euchner search algorithm

Given a nonsingular upper triangular matrix $\boldsymbol{R} \in \mathbb{R}^{n \times n}$, this algorithm finds a solution $\boldsymbol{a}^\star$ to the SVP (5.6).

1. (Initialization) Set $k = n, \beta = +\infty$.
2. Compute d_k by using (5.23), set $a_k = \lfloor d_k \rceil$ and $s_k = \operatorname{sgn}(d_k - a_k)$ (see (5.25)).
3. (Main Step) If the inequality in (5.24) does not hold, then go to Step 4. Else if $k > 1$, set $k = k - 1$ and go to Step 2. Else ($k = 1$), go to Step 5.
4. (Outside ellipsoid) If $k = n$, terminate. Else, set $k = k + 1$ and go to Step 6.
5. (A valid point is found) If $\boldsymbol{a}$ is a nonzero vector, then save $\boldsymbol{a}^\star = \boldsymbol{a}$, set $\beta = \|\boldsymbol{R}\boldsymbol{a}\|_2$ and $k = k + 1$.
6. (Enumeration at level k) Set $a_k = a_k + s_k$, $s_k = -s_k - \operatorname{sgn}(s_k)$ and go to Step 3.

5.3.4 Modified Schnorr-Euchner search algorithm

In the following we make some comments to Algorithm 5.3.1 and make some modifications. It is easy to see that, the first nonzero integer vector encountered by Algorithm 2 is $\boldsymbol{e}_1$ and the corresponding search radius is

$$\beta = |r_{11}| = \sqrt{1 - t_1^2}. \tag{5.26}$$

Note that reordering the entries of $\boldsymbol{t}$ that makes (5.19) hold gives the smallest β among any other orderings. This shows one of the benefits of the reordering leading to (5.19). Also from (5.21), the reordering gives:

$$\beta = |r_{11}| = \min_{1 \le i \le n} \|\boldsymbol{R}_{1:i,i}\|_2,$$

which implies $\boldsymbol{e}_1$ is better than any other $\boldsymbol{e}_i$ for $i = 2, \ldots, n$, as the former corresponds to the smallest residual. In the modified algorithm, we just start with β given by (5.26).

In Section 5.3.1, we mentioned that it is not necessary to form the entries of $\boldsymbol{R}$ explicitly; in the following, we show how to compute r_{kk} and d_k for $1 \le k \le n$, which are needed in (5.24). By (5.7) and (5.10), we have

$$r_{kk}^2 = f_k / f_{k-1}, \quad 1 \le k \le n. \tag{5.27}$$

In the modified algorithm, we will use a $n-$dimensional vector $\boldsymbol{q}$ to store r_{kk}^2, i.e., let $q_k = r_{kk}^2$.

By (5.7), (5.10) and (5.23),

$$d_k = \frac{t_k}{f_k} \sum_{j=k+1}^{n} t_j a_j.$$

Thus, for computational efficiency, we introduce an $(n+1)-$dimensional vector $\boldsymbol{p}$ whose last entry is 0 to store some computed quantities. Specifically, after a_k, $1 \leq k \leq n$ is chosen in the search process, we assume

$$p_k = p_{k+1} + t_k a_k, \quad 1 \leq k \leq n, \tag{5.28}$$

which explains why $p_{n+1} = 0$. Therefore, we have

$$d_k = \frac{t_k p_{k+1}}{f_k}, \quad 1 \leq k \leq n. \tag{5.29}$$

Now we make the main modification to Algorithm 2 by using the property of $\boldsymbol{a}^\star$ in (5.20). Note that in the search process of finding an integer point $\boldsymbol{a}$ in the hyper-ellipsoid, the entries of $\boldsymbol{a}$ are determined in the following order: a_n, a_{n-1}, ..., a_1. When we enumerate candidates for a_n at level n, we will only enumerate the non-negative integers. When we enumerate candidates for a_k at level k (note that at this point, a_n, a_{n-1}, ..., a_{k+1} have been chosen), we will only enumerate those greater than or equal to a_{k+1}. By doing these we can prune a lot of nodes from the search tree to make the search process much faster.

For the users to implement the algorithm easily and for our later complexity analysis, we provide the pseudo code of the modified algorithm in Algorithm 5.3.2.

Here we make a few comments to Algorithm 3. To unify the enumeration strategies for level n and for any lower level, we set $\boldsymbol{a}$ to be an $(n+1)-$dimensional vector

Algorithm 5.3.2 Finding the optimal coefficient vector based on sphere decoding

Given a $n-$dimensional vector $\boldsymbol{t}$ that satisfies $\|\boldsymbol{t}\| < 1$ (see (5.4)) and (5.19), this algorithm solves (5.5).

$\boldsymbol{f} = \mathbf{0}^n$, $\boldsymbol{q} = \mathbf{0}^n$; // $q_k = r_{kk}^2$
$f_1 = 1 - t_1^2, q_k = f_1$; // see (5.10) and (5.27)
for i=2:n **do**
    $f_i = f_{i-1} - t_i^2$;// see (5.10)
    $q_i = f_i / f_{i-1}$;// see (5.27)
end for
$\boldsymbol{d} = \mathbf{0}^n$, $\boldsymbol{p} = \mathbf{0}^{(n+1)}$//see (5.23) and (5.28)
dist $= \mathbf{0}^n$; // dist$(k) = \sum_{i=k+1}^{n} r_{ii}^2 (a_i - d_i)^2$ for $k < n$;
$\boldsymbol{a} = \boldsymbol{e}_1^{n+1}$; // intermediate solution
$\boldsymbol{a}^\star = \boldsymbol{e}_1^n$;
$\beta^2 = f_1$, $\gamma = f_1$, $k = 1$, $\boldsymbol{s} = \mathbf{1}^n$, flag $= \mathbf{1}^n$;
while $k \geq 1$ **do**
    newdist $=$ dist$(k) + \gamma$;
    if newdist $< \beta^2$ **then**
        if $k \neq 1$ **then**
            $p_k = p_{k+1} + t_k a_k$//see (5.28)
            $k = k - 1$;
            dist(k) = newdist;
            $d_k = t_k p_{k+1} / f_k$; // see (5.29)
            $a_k = \lfloor d_k \rceil$;
            flag$_k = 0$
            if $a_k \leq a_{k+1}$ **then**
                $a_k = a_{k+1}$;
                $s_k = 1$;
                flag$_k = 1$
            else
                $s_k = \mathrm{sgn}(d_k - a_k)$; // see (5.25)
            end if
            $\gamma = q_k((d_k - a_k))^2$
        else
            $\beta^2 =$ newdist;
            $\boldsymbol{a}^\star = \boldsymbol{a}_{1:n}$;
        end if
    else
        if $k = n$ **then**
            return;
        else
            $k = k + 1$;
            $a_k = a_k + s_k$;
            if $a_k = a_{k+1}$ **then**
                flag$_k = 1$;
                $s_k = -s_k - \mathrm{sgn}(s_k)$;
            else if flag$_k = 1$ **then**
                $s_k = 1$;
            else
                $s_k = -s_k - \mathrm{sgn}(s_k)$;
            end if
            $\gamma = q_k((d_k - a_k))^2$;
        end if
    end if
end while

with $a_{n+1} \equiv 0$, so that $a_k \geq a_{k+1}$ holds for $k = n$. To avoid enumerating any integer smaller than a_{k+1} at level k, we introduced a flag variable "flag" in the algorithm to indicate whether the enumeration has reached the lower bound a_{k+1} for $1 \leq k \leq n$. In the algorithm s_k is the difference between the next integer candidate for a_k and the current value of a_k and it is used to get the next integer candidate for a_k.

5.4 Complexity analysis

In this section, we will analyze the complexity, in terms of flops, of the proposed method, and compare it with two optimal methods proposed in [71] and [72], and two suboptimal methods, which are the LLL reduction approach [73] and the quadratic programming relaxation (QPR) approach [101] and its improvement in [102].

5.4.1 Complexity analysis for the modified Schnorr-Euchner search algorithm

In this subsection, we try to analyze the cost of Algorithm 5.3.2. The approach is to first count the number of nodes visited in the search tree and then to count the number of arithmetic operations for each node. Unfortunately, we cannot give a good bound on the number of nodes visited. Instead, we will give a conjecture about it, which will be supported by numerical results.

It is difficult, if not impossible, to analyze the cost of Algorithm 5.3.2 because the search radius β changes in the search process. Thus, we assume that the search radius β keeps unchanged in our following analysis.

To illustrate our discussion, in Figure 5–1 we display the search tree corresponding to Algorithm 5.3.2 with the assumption that β is a constant. Since there is not a true tree root, the dashed line is used for the root node in Figure 5–1. We will

analyze the cost of this search tree, which is an upper bound on the cost of Algorithm 5.3.2.

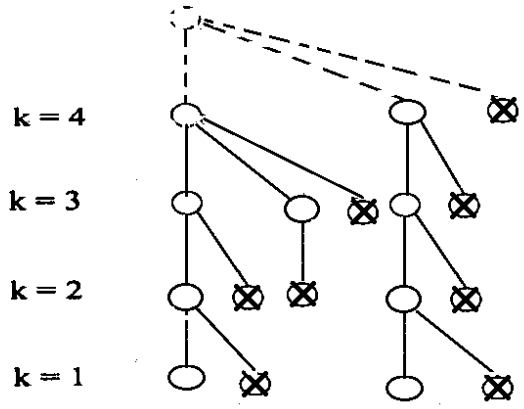

Figure 5–1: Search tree

To estimate the number of nodes at each level of the search tree, for $k = n, n-1, \ldots, 1$, we define sets:

$$E_k(\beta) = \{\boldsymbol{a}_{k:n} \in \mathbb{Z}^{n-k+1} : \ a_k \geq a_{k+1} \geq \ldots \geq a_n \geq 0, \|\boldsymbol{R}_{k:n,k:n}\boldsymbol{a}_{k:n}\|_2 < \beta\}. \quad (5.30)$$

Note that each non-leaf node at level k in the search tree corresponds to an $\boldsymbol{a}_{k:n} \in E_k(\beta)$, and each leaf node labeled by $\times$ at level k corresponds to an $\boldsymbol{a}_{k:n} \notin E_k(\beta)$ with $\boldsymbol{a}_{k+1:n} \in E_{k+1}(\beta)$ $(k < n)$.

Let $|E_k(\beta)|$ denote the number of elements belong to $E_k(\beta)$. Thus the number of non-leaf nodes at level k in the search tree is $|E_k(\beta)|$. It is easy to argue that the number of leaf nodes at level k in the search tree is exactly equal to $|E_{k+1}(\beta)|$. Thus

the total number of nodes (including both the non-leaf and leaf nodes) at level k is $|E_k(\beta)| + |E_{k+1}(\beta)|$.

From Algorithm 5.3.2, any node at level k ($k < n$) comes from two possibilities. One is that it is generated after its parent node at level $k+1$ is generated. This process corresponds lines 14-33 of Algorithm 5.3.2 and the cost is $\mathcal{O}(1)$ flops. The number of such nodes is $|E_{k+1}(\beta)|$. The other is that it is generated after a leaf node at level $k-1$ is generated. This process corresponds lines 38-48 and the cost is also $\mathcal{O}(1)$ flops. The number of such nodes is $|E_k(\beta)|$. Thus, the total cost for generating all nodes at level k is

$$c_k = (|E_k(\beta)| + |E_{k+1}(\beta)|)\mathcal{O}(1), \tag{5.31}$$

where we denote $|E_{n+1}(\beta)| = 0$. Let $C(n)$ denote the total cost of the search tree, then, by (5.31), we obtain

$$C(n) = \sum_{k=1}^{n} c_k = \mathcal{O}(1) \sum_{k=1}^{n} |E_k(\beta)|. \tag{5.32}$$

Obviously, $|E_n(\beta)| \leq \lceil \beta / r_{nn} \rceil$. However, it is hard to rigorously compute or estimate $|E_k(\beta)|$ since the inequalities are involved in (5.30), so for $k = 1, 2, \ldots, n$, we define supersets:

$$F_k(\beta) = \{\boldsymbol{a}_{k:n} \in \mathbb{Z}^{n-k+1} : \ \|\boldsymbol{R}_{k:n,k:n}\boldsymbol{a}_{k:n}\| < \beta\}, \tag{5.33}$$

where β is the initial search radius used in Algorithm 3 (see (5.26)).

Let $|F_k(\beta)|$ denote the number of elements belong to $F_k(\beta)$. Obviously, we have

$$|E_k(\beta)| \leq |F_k(\beta)|. \tag{5.34}$$

Then we would like to find $|F_k(\beta)|$. But it is difficult to give a good bound on it. Naturally we would like to try to use a common approach used in the complexity analysis of sphere decoding methods, which is approximating $|F_k(\beta)|$ by the volume of the hyper-ellipsoid $\|\boldsymbol{R}_{k:n,k:n}\boldsymbol{a}_{k:n}\|_2 < \beta$ (see, e.g., [31, 10, 2]), i.e.,

$$|F_k(\beta)| \approx \frac{\beta^{n-k+1}}{\prod_{i=k}^{n} r_{ii}} V_{n-k+1}, \tag{5.35}$$

where V_{n-k+1} denotes the volume of an $(n-k+1)-$dimensional unit Euclidean ball, i.e.,

$$V_{n-k+1} = \frac{\pi^{(n-k+1)/2}}{\Gamma((n-k+1)/2+1)} \tag{5.36}$$

with Γ being the Gamma function.

Unfortunately, the approximation (5.35) is not valid for this specific problem. In fact, by (5.12) and (5.26), from (5.35) we have

$$|F_k(\beta)| \approx \frac{(1-\sum_{i=1}^{k-1} t_i^2)^{1/2}(1-t_1^2)^{(n-k+1)/2}}{\sqrt{1-\|\boldsymbol{t}\|_2^2}} V_{n-k+1} \leq \frac{1}{\sqrt{1-\|\boldsymbol{t}\|_2^2}} V_{n-k+1}. \tag{5.37}$$

According to [26],

$$\sum_{k=1}^{\infty} V_k = e^{\pi}(1+\frac{2}{\sqrt{\pi}}\int_0^{\sqrt{\pi}} e^{-t^2}dt) \leq 2e^{\pi}.$$

Therefore, by (5.34)-(5.37), we have

$$\sum_{k=1}^{n} |E_k(\beta)| \leq \sum_{k=1}^{n} |F_k(\beta)| \lessapprox \frac{2e^{\pi}}{\sqrt{1-\|\boldsymbol{t}\|_2^2}}. \tag{5.38}$$

This indicates that if $\boldsymbol{t}$ is fixed, the number of nodes in the tree is approximately a constant. But the number of nodes in the search true certainly depend on n. Thus the approximation (5.37) is problematic. In fact, if we assume $\boldsymbol{t} = \|\boldsymbol{t}\|_2 \boldsymbol{e}_1$, then by (5.7), $\boldsymbol{R}$ is a diagonal matrix with

$$r_{11} = \sqrt{1-\|\boldsymbol{t}\|_2^2}, \;\; r_{kk} = 1, 2 \leq k \leq n.$$

Thus, by (5.30) and (5.26), $|E_k(\beta)| = 1$ for $2 \leq k \leq n$ since only the zero vector is contained in the corresponding sets. And it is not hard to see that

$$\frac{\beta}{\sqrt{1-\|\boldsymbol{t}\|_2^2}} \leq |E_1(\beta)| \leq \frac{\beta}{\sqrt{1-\|\boldsymbol{t}\|_2^2}} + 1.$$

Thus, for this special case,

$$n + \frac{\beta}{\sqrt{1-\|\boldsymbol{t}\|_2^2}} - 1 \leq \sum_{k=1}^{n} |E_k(\beta)| \leq n + \frac{\beta}{\sqrt{1-\|\boldsymbol{t}\|_2^2}}. \tag{5.39}$$

If we assume $\|\boldsymbol{t}\|_2 \approx 0$, then by (5.38), $\sum_{k=1}^{n} |E_k(\beta)| \approx 2e^{\pi}$, contradicting with (5.39) if n is very large. However, we think the following conjecture holds.

Conjecture 5.4.1

$$\sum_{k=1}^{n} |E_k(\beta)| \leq \frac{cn}{\sqrt{1-\|\boldsymbol{t}\|_2^2}}, \tag{5.40}$$

where c is a constant.

In the following, we do some simulations to support that (5.40) holds for general n and $\boldsymbol{t}$. Note that $\boldsymbol{t}$ is defined by P and $\boldsymbol{h}$ in (5.4). Thus, we consider the case that

Table 5–1: Average and largest ratios of $\sum_{k=1}^{n} |E_k(\beta)|$ to $n/\sqrt{1-\|\boldsymbol{t}\|_2^2}$ over 10000 realizations of $\boldsymbol{h}$

n \ P	$P = 0\,\mathrm{dB}$		$P = 20\,\mathrm{dB}$		$P = 40\,\mathrm{dB}$	
	AR	LR	AR	LR	AR	LR
2	0.4241	1.4747	0.3032	1.3399	0.3178	0.9292
4	0.5259	1.7803	0.4273	1.3123	0.4401	1.4314
8	0.4408	1.2875	0.4040	1.2632	0.4253	1.5001
16	0.3204	0.9109	0.1813	0.5917	0.1887	0.6826
32	0.2205	0.6348	0.0610	0.1770	0.0509	0.2033
64	0.1471	0.3783	0.0265	0.0602	0.0127	0.0394
10^2	0.1143	0.2937	0.0183	0.0338	0.0054	0.0154
10^3	0.0381	0.0567	0.0048	0.0063	0.0005	0.0006
10^4	0.0129	0.0151	0.0014	0.0016	0.0001	0.0002
10^5	0.0040	0.0043	0.0004	0.0004	0.0000	0.0000

the channel vector $\boldsymbol{h} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ (see Definition 5.2.1). For each n and each P, we randomly generate 10000 realizations of $\boldsymbol{h}$.

Table 5–1 displays the average and largest ratios of $\sum_{k=1}^{n} |E_k(\beta)|$ to $n/\sqrt{1-\|\boldsymbol{t}\|_2^2}$ over 10000 samples. "AR" and "LR" in Table 5–1 respectively denote average and largest ratio. From the largest ratios in Table 5–1, we can see that (5.40) holds with a small constant $c = 2$ for all the test cases. The average ratios in Table 5–1 show that averagely our algorithm searched far less number of nodes than what the conjectured bound with $c = 1$ especially when n is large. Note that the number of nodes searched by Algorithm 5.3.2 can not be larger than $\sum_{k=1}^{n} |E_k(\beta)|$ because the radius β reduces whenever a valid integer vector is found in the search process.

By (5.32), we get the following result.

Theorem 5.4.1 *If* (5.40) *holds, then we have*

$$C(n) \leq \frac{\mathcal{O}(n)}{\sqrt{1 - \|\boldsymbol{t}\|_2^2}}. \tag{5.41}$$

In the following, we would like to investigate the expected value of $C(n)$ when the entries of $\boldsymbol{h}$ are independent and identically follow normal distribution $\mathcal{N}(0, 1)$. It is easy to see that $\|\boldsymbol{h}\|_2^2$ follows the chi-squared distribution $\chi^2(n)$. Therefore, $\mathbb{E}[\|\boldsymbol{h}\|_2^2] = n$. Since $\sqrt{1 + Px}$ is a concave function of x, by Jensen's Inequality,

$$\mathbb{E}\left[\sqrt{1 + P\|\boldsymbol{h}\|_2^2}\right] \leq \sqrt{1 + P\mathbb{E}\left[\|\boldsymbol{h}\|_2^2\right]} = \sqrt{1 + nP}. \tag{5.42}$$

Therefore, by (5.4), (5.41) and (5.42), it is easy to see that the cost of Algorithm 5.3.2 is $\mathcal{O}(n^{1.5})$ flops if (5.40) holds.

5.4.2 Comparison of the complexity of the proposed method with other methods

It is easy to see that, for any given $\boldsymbol{h}$, computing $\boldsymbol{t}$ by (5.4) costs $\mathcal{O}(n)$ flops. And for any fixed $\boldsymbol{t}$, transform it such that (5.19) holds costs $\mathcal{O}(n \log(n))$ comparisons. Since the total cost of Algorithm 5.3.2 is $\mathcal{O}(n^{1.5})$ flops if (5.40) holds. Simulation results in the above subsection indicates (5.40) holds for the test cases, thus, the total cost of the whole method is $\mathcal{O}(n^{1.5})$ flops for the test cases.

The complexity of the QPR in [101] and [102] is $\mathcal{O}(n^3)$ and $\mathcal{O}(n^{1.5})$ flops, respectively. The method based on LLL lattice reduction [73] uses the regular method, costing $\mathcal{O}(n^3)$, to obtain the Cholesky factor $\boldsymbol{R}$. The optimal method proposed in [71] needs to find the inverse of n matrices and solving n linear equations with the dimensions from 1 to n, so its cost is higher than $\mathcal{O}(n^3)$. The complexity of the optimal method proposed in [72] is $\mathcal{O}(n^{2.5})$ flops. Therefore, it is expected that our

optimal algorithm is faster than the LLL reduction based method, the QPR in [101] and the two optimal methods proposed in [73] and [72], and faster than or has more or less the same speed as the QPR in [102] if the conjecture (5.40) holds.

5.5 Numerical simulations

In this section, we present the numerical results to demonstrate the effectiveness and efficiency of our new method. We consider the case that the entries of the channel vector $\boldsymbol{h} \in \mathbb{R}^n$ are i.i.d. Gaussian, i.e., $\boldsymbol{h} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. The dimension n of $\boldsymbol{h}$ ranges from 2 to 16. For a given n, we randomly generate 10000 realizations of $\boldsymbol{h}$ for each P from 0 dB to 20 dB with the step length 2 dB, and apply different methods to calculate the corresponding computation rates. To compare the effectiveness of different methods, we compute the average computation rates; To compare the efficiency, we record the running time.

The methods considered include our new method called the improved sphere decoding (ISD) method, the branch-and-bound (BnB) algorithm [71], the optimal method proposed in [72] (to be called SG named after the authors), the method based on LLL lattice reduction algorithm [73] (to be called LLL), the quadratic programming relaxation (QPR) approach [102] and the simple quantized search (QS) method [73]. The *quality-complexity tradeoff factor* δ in the LLL method is set as 0.75. A larger δ $(1/4 < \delta \leq 1)$ can give a higher rate, but the running time will increase drastically as δ increases. The *number of real-valued approximations*, K, in the QPR method is set according to the criterion proposed in [102], i.e., setting K to be the smallest K_0 such that the simulated average computation rate at 20dB using the QPR method with $K = K_0$ is greater than 99% of that with $K = K_0 + 1$. Exact

values of K are listed in Table 5–2. The QS method implemented here consists of two phases: 1) selecting an amplification factor $\alpha = \alpha_0 \in \{1, 2, \ldots, \lfloor P^{1/2} \} \rfloor$; 2) refining α by searching in $[\alpha_0 - 1, \alpha_0 + 1]$ with a step size 0.1.

Table 5–2: Number of real-valued approximations in QPR method

n	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
K	2	3	4	5	5	5	6	6	6	7	7	7	6	7	6

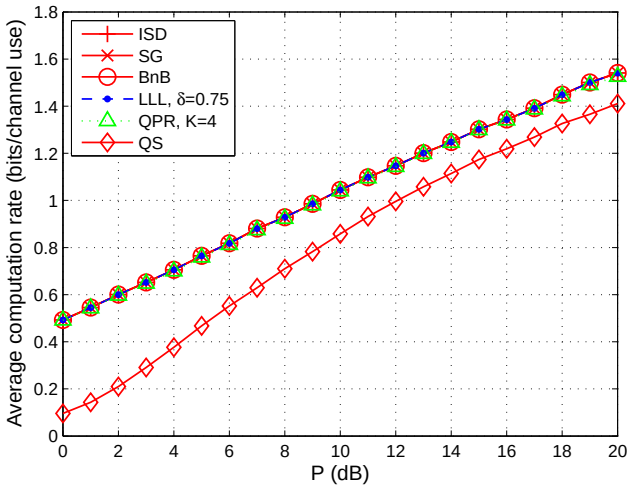

Figure 5–2: Average computation rates by different methods for $n = 4$

We first compare the average computation rates. Figures 5–2, 5–3, 5–4 show the average computation rates over 10000 samples with the dimension n being 4, 8, and 16, respectively. The ISD method, the SG method and the BnB method are optimal. As expected, numerical results show that they always provide the highest computation rate. The corresponding curves of these two methods in Figures 5–2, 5–3, 5–4 exactly overlap with each other. The QS method is of low complexity; but as shown in Figures 5–2, 5–3, 5–4, the corresponding rate is very low, especially when

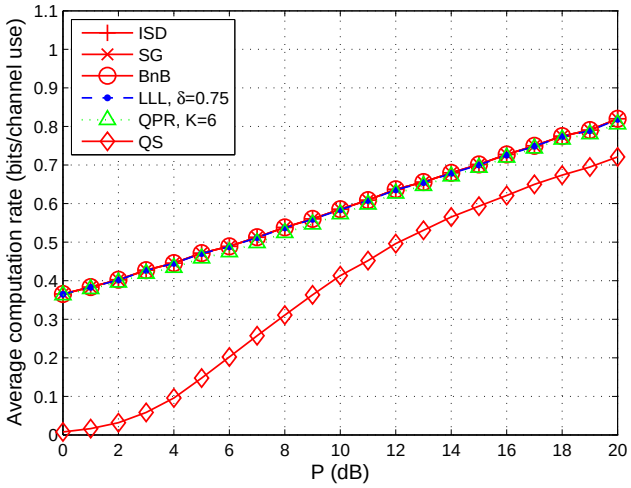

Figure 5–3: Average computation rates by different methods for $n = 8$

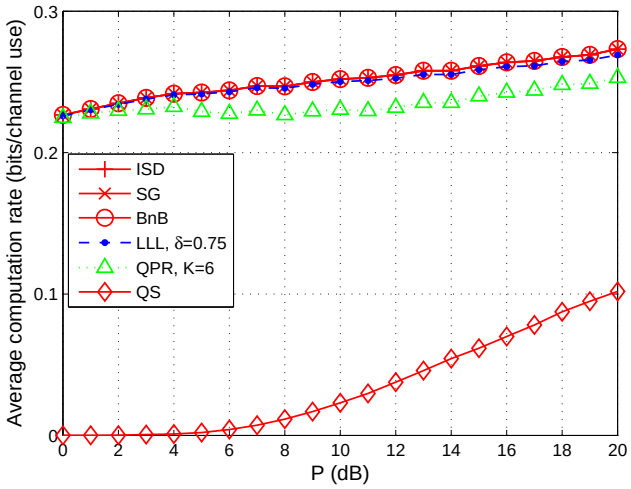

Figure 5–4: Average computation rates by different methods for $n = 16$

the dimension becomes large. The QPR method and the LLL based method provide rates close to that of the optimal methods. However, as the dimension increases, their performance degrade.

Now we compare the running time. We consider methods that offer optimal and close-to-optimal rates: ISD, BnB, SG, LLL, and QPR in [102]. Figures 5–5, 5–6, 5–7 show the running time of simulating 10000 samples with P being 0 dB, 10 dB, and 20 dB, respectively. For the optimal methods, it is obvious that our new ISD method is much more efficient than both the BnB method and SG method. It can also be observed that the ISD method is also faster than the LLL based method. Although the QPR method [102] is faster than our ISD method in Figure 5–7, it is a suboptimal solution and its performance degrades for high dimension (see Figure 5–4).

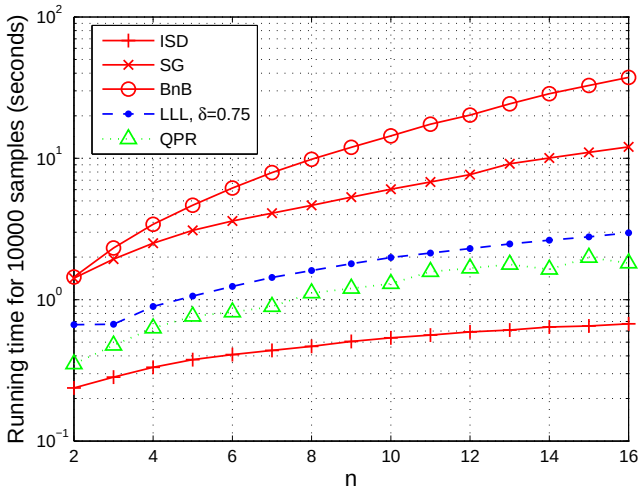

Figure 5–5: Running time for 10000 samples by different methods for $P = 0$ dB

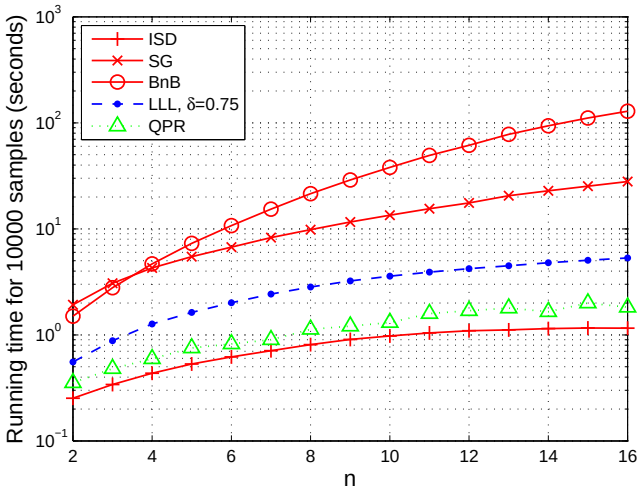

Figure 5–6: Running time for 10000 samples by different methods for $P = 10$ dB

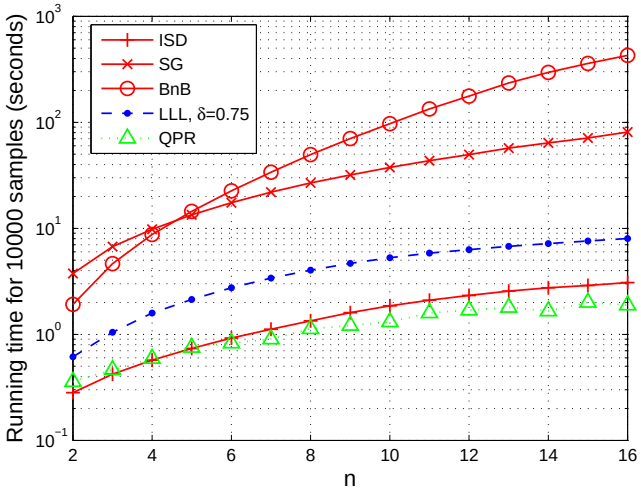

Figure 5–7: Running time for 10000 samples by different methods for $P = 20$ dB

CHAPTER 6
Summary and Future Work

In this thesis, we have investigated the effects of the LLL reduction, and the LLL-P, V-BLAST and SQRD column permutation strategies on the success probabilities of the Babai estimators for the ILS (OILS and BILS) estimation problems, and designed algorithms to solve a SVP arising in computer-and-forward protocol design.

In Chapter 2, we gave an extensive background introduction to the QRZ reduction, i.e., the lattice reduction, and the QRP reduction, i.e., the column reordering strategies, to transfer the ILS problems to new ILS problems which can then be solved by a search algorithm. We also introduced the Babai estimators, the commonly used suboptimal solutions to the ILS problems in practice.

In Chapter 3, we investigated the effects of the LLL reduction and some column permutation strategies on the success probability P^{OB} of the ordinary Babai estimator $\boldsymbol{x}^{\text{OB}}$ for the ordinary integer linear model (1.1). First, we showed that P^{OB} as a lower bound on the success probability P^{OL} of the OILS estimator $\boldsymbol{x}^{\text{OL}}$ is sharper than the lower bound given in [37]. Second, we gave a rigorous proof to show that the LLL reduction algorithm can always increase (not strictly) P^{OB}, and it keeps the P^{OB} unchanged if and only if no column permutation occurs during the LLL reduction process or whenever two consecutive columns, say $k-1$ and k, are permuted, $r_{k-1,k}$ is a multiple of $r_{k-1,k-1}$ (before the size reduction on $r_{k-1,k}$ is performed). We also showed that any size reductions on the super diagonal entries of $\boldsymbol{R}$ of the QR

factorization of the channel matrix which are immediately followed by a column permutation during the LLL reduction process will enhance P^{OB} and all other size reductions have no effect on P^{OB}. Furthermore, we show that the LLL-P column permutation strategy can also always increase (not strictly) P^{OB}, and it keeps P^{OB} unchanged if and only if no column permutation occurs during the LLL-P process or whenever two consecutive columns, say $k-1$ and k, are permuted, $r_{k-1,k} = 0$. Third, we gave an example to show that larger δ, which is a parameter in the LLL reduction, may not bring higher improvement of P^{OB}, and did some simulations to show that larger δ usually bring higher improvement of P^{OB}. Fourth, we gave examples to show that unlike LLL and LLL-P, the permutation strategies SQRD and V-BLAST may decrease P^{OB}. But simulations showed that both of them usually increase P^{OB}, and averagely the LLL reduction improves it much more significantly than the other three, V-BLAST performs better than LLL-P and SQRD, and LLL-P and SQRD have similar performances. Finally, we gave some upper bounds on P^{OB} after applying the LLL reduction algorithm by using the entries of the R-factor $\boldsymbol{R}$ of the QR factorization of the channel matrix.

We will investigate the following problems in the future. We will study the effects of the LLL reduction algorithm on P^{OB} if the model matrix is a random matrix following some distributions, instead of being deterministic assumed in this thesis, since in some communications applications, see, e.g. [42] and [55], the entries of the model matrix independently follow the normal distribution $\mathcal{N}(0,1)$. The HKZ reduction usually returns a basis matrix whose columns are shorter than that of the

LLL reduction. We will study if there is an algorithm for computing the HKZ reduction, which can always improve P^{OB} and guarantee to have higher improvement than the LLL reduction algorithm. Since the running time of the LLL reduction algorithm varies much from matrices to matrices even for matrices with fixed dimension, which may cause problems for the real-time communications system from the implementation point of view. To address this issue, fixed complexity LLL reduction algorithms have been proposed, see, e.g., [87]. We are designing some more effective fixed-complexity LLL reduction algorithms by using the theories developed in Chapter 3 in [91].

In Chapter 4, we investigated the effects of some typical column permutation strategies on the success probability P^{BB} of the box-constrained Babai estimator $\boldsymbol{x}^{\text{BB}}$ for the box-constrained integer linear model and solved a conjecture proposed in [58]. First, we derived a formula for P^{BB}. Then, on the one hand, we showed that LLL-P always increases P^{BB} and argued why both V-BLAST and SQRD often increase P^{BB} under a condition, which, roughly speaking, is the noise is relatively small; and on the other hand, we showed that LLL-P always decreases P^{BB} and argued why both V-BLAST and SQRD often decrease P^{BB} under an opposite condition, i.e., the noise is relatively large. These surprising results indicate that we need to check the corresponding conditions before applying these strategies. After this, we derived a column permutation invariant bound on P^{BB}, which is an upper bound and a lower bound under the two opposite conditions, respectively. We also gave some numerical results to illustrate the above results. Finally, we constructed an example to show

that the conjecture proposed in [58] does not always hold, and then provided some conditions to guarantee it hods.

In the future, we would like to consider the following problems. As shown in Chapter 4, the LLL-P has better theory than V-BLAST and SQRD in terms of their effects on P^{BB}, but numerical experiments indicate that often V-BLAST is more effective than LLL-P and SQRD. Therefore, we will try to combine their advantages to develop a more effective column permutation strategy for the BILS problems. Although it was shown in [84] that the success probability P^{OB} of the ordinary Babai estimator $\boldsymbol{x}^{\mathrm{OB}}$ can not be larger than the success probability P^{OL} of the OILS estimator $\boldsymbol{x}^{\mathrm{OL}}$, it can be shown that this conclusion can not be extended to the BILS problem if the true parameter vector $\hat{\boldsymbol{x}}$ is deterministic. Whether $P^{\mathrm{BB}} \leq P^{\mathrm{BL}}$ always holds for uniform distributed $\hat{\boldsymbol{x}}$ will be studied in the future.

In Chapter 5, we proposed an efficient algorithm for finding the optimal coefficient vector that maximizes the computation rate at a relay in the computer-and-forward scheme. First, we derived an $\mathcal{O}(n)$ algorithm to compute the Cholesky factorization of the $n \times n$ matrix by fully using its structure to transfer the optimization problem to a standard SVP (we did not form the whole Cholesky factor explicitly). Then, we proposed some conditions, which can be checked by $\mathcal{O}(n)$ flops and under which, the optimal coefficient vector can be obtained immediately without using any search algorithm. After this, we proposed a modified Schnorr-Euchner search algorithm to solve the SVP by taking into account some resultant useful properties of the optimal coefficient vector. Simulations showed that the average cost of our new algorithm is $\mathcal{O}(n^{1.5})$ if the entries of the channel vector independently follow

the normal distribution $\mathcal{N}(0,1)$, and our proposed algorithm is not only much more efficient than the existing ones that give the optimal solution, but also faster than some of the suboptimal methods.

In the future, first, we will try to show in theory that indeed the complexity of our algorithm is $\mathcal{O}(n^{1.5})$. Then, we intend to extend our algorithm to find the integer network coding coefficients over a compute-and-forward multi-source multi-relay system, i.e., solving the following optimization problem:

$$\min_{\boldsymbol{A}\in\mathbb{Z}^{n\times n}} \max_{1\leq m\leq n} \quad \boldsymbol{a}_m^T \boldsymbol{G}_m \boldsymbol{a}_m, \quad \boldsymbol{G}_m = \boldsymbol{I} - \frac{P}{1+P\|\boldsymbol{h}_m\|}, \tag{6.1}$$

$$\text{subject to:} \quad \boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n]^T, \tag{6.2}$$

$$\text{and:} \quad |\boldsymbol{A}| \neq 0, \tag{6.3}$$

where P is the transmission power and $\boldsymbol{h}_m \in \mathbb{R}^n$ is the channel vector. For more details, see, e.g., [89]. And we also intend to design efficient algorithms for finding the coefficient matrix with full rank for the integer forcing linear receiver design, i.e., solving the following optimization problem:

$$\min_{\boldsymbol{A}\in\mathbb{Z}^{n\times n}} \max_{1\leq m\leq n} \quad \boldsymbol{a}_m^T \boldsymbol{Q} \boldsymbol{a}_m, \quad \boldsymbol{Q} = \boldsymbol{I} - \boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{H}^T + \frac{1}{P}\boldsymbol{I})^{-1}\boldsymbol{H}, \tag{6.4}$$

$$\text{subject to:} \quad \boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n]^T, \tag{6.5}$$

$$\text{and:} \quad |\boldsymbol{A}| \neq 0, \tag{6.6}$$

where P is the transmission power and $\boldsymbol{H}$ is the channel matrix. For more details, see, e.g., [73].

References

[1] E. Agrell and T. Eriksson. Optimization of lattices for quantization. *IEEE Transactions on Information Theory*, 44(5):1814–1828, 1998.

[2] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. Closest point search in lattices. *IEEE Transactions on Information Theory*, 48(8):2201–2214, 2002.

[3] M. Ajtai. The shortest vector problem in $\boldsymbol{l}_2$ is NP-hard for randomized reductions (extended abstract). In *Proceedings of the thirtieth annual ACM symposium on Theory of Computing, Dallas, Texas, USA, May 23-26*, pages 10–19, 1998.

[4] 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*, pages 601–610, 2001.

[5] M. Ajtai, R. Kumar, and D. Sivakumar. Sampling short lattice vectors and the closest lattice vector problem. In *Proceedings of the 17th IEEE Annual Conference on Computational Complexity*, pages 41–45, 2002.

[6] A Akhavi. The optimal LLL algorithm is still polynomial in fixed dimension. *Theoretical Computer Sciences*, 297(1-3):3–23, 2003.

[7] M.F. Anjos, X.-W. Chang, and W.-Y. Ku. Lattice preconditioning for the real relaxation branch-and-bound approach for integer least squares problems. *Journal of Global Optimization.*, (59):227–242, 2014.

[8] H. Artés, D. Seethaler, and F. Hlawatsch. Efficient detection algorithms for MIMO channels: a geometrical approach to approximate ML detection. *IEEE Transactions on Signal Processing*, 51(11):2808–2820, 2003.

[9] L. Babai. On lovasz lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[10] A.H. Banihashemi and A. K. Khandani. On the complexity of decoding lattices using the korkin-zolotarev reduced basis. *IEEE Transactions on Information Theory*, 44(1):162–171, 1998.

[11] L. G. Barbero and J. S. Thompson. Fixing the complexity of the sphere decoder for mimo detection. *IEEE Transactions on Wireless Communication*, 7(6):2131–2142, 2008.

[12] J. Blömer and S. Naewe. Sampling methods for shortest vectors, closest vectors and successive minima. *Theoretical Computer Science*, 410(18):1648–1665, 2009.

[13] A. W. Bojanczyk, R.P. Brent, P. Van Dooren, and F. R. De Hoog. A note on downdating the cholesky factorization. *SIAM Journal on Scientific and Statistical Computing*, 8(3):210–221, 1987.

[14] J. Boutros, N. Gresset, L. Brunel, and M. Fossorier. Soft-input soft-output lattice sphere decoder for linear channels. In *Proceedings of IEEE 2003 GLOBECOM*, pages 213–217, December 2003.

[15] S. Breen and X.W. Chang. Column reording for box-bonstrained integer least squares problems. In *Proceedings of IEEE GLOBECOM 2011, 6 pages.*

[16] X.-W. Chang and Q. Han. Solving box-constrained integer least squares problems. *IEEE Transactions on Wireless Communications*, 7(1):277–287, 2008.

[17] X.-W. Chang and C.C. Paige. Euclidean distances and least squares problems for a given set of vectors. *Applied Numerical Mathematics*, 57(1):1240–1244, 2007.

[18] 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*, 59(8):4915–4926, 2013.

[19] J. H. Conway and N. J. A. Sloane. On the voronoi regions of certain lattices. *SIAM Journal on Algebraic and Discrete Methods*, 5(3):294–305, 1984.

[20] T. Cui, S. Han, and T. Tellambura. Probability-distribution-based node pruning for sphere decoding. *IEEE Transactions on Vehicular Technology*, 62(4):1586–1596, 2013.

[21] M. O. Damen, H. E. Gamal, and G. Caire. On maximum likelihood detection and the search for the closest lattice point. *IEEE Transactions on Information Theory*, 49(10):2389–2402, 2003.

[22] H. Daudé and B. Vallée. An upper bound on the average number of iterations of the lll algorithm. *Theoretical Computer Science*, 123(1):95–115, 1994.

[23] F Eisenbrand. *Integer Programming and Algorithmic Geometry of Numbers. In M. Jünger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, and L. A. Wolsey, editors, 50 Years of Integer Programming 1958-2008: From the Early Years to the State-of-the-Art, pp. 505-559.* Springer, Heidelberg, 2010.

[24] U. Fincke and M. Pohst. Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. *Mathematics of Computation*, 44(170):463–471, 1985.

[25] G. J. Foscini, G. D. Golden, R. A. Valenzuela, and P. W. Wolniansky. Simplified processing for high spectral efficiency wireless communication employing multi-element arrays. *IEEE Journal on Selected Areas in Communications*, 17(11):1841–1852, 1999.

[26] E Freden. Problem 10207: Summing a series of volumes. *Amer. Math. Monthly 100*, page 882, 1993.

[27] Y.H. Gan, C. Ling, and W. H. Mow. Complex lattice reduction algorithm for low-complexity full-diversity MIMO detection. *IEEE Transactions on Signal Processing*, 57(7):2701–2710, 2009.

[28] O. Goldreich, D. Ron, and M. Sudan. Chinese remaindering with errors. *IEEE Transactions on Information Theory*, 46(4):1330–1338, 2000.

[29] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 3 edition, 1996.

[30] R. Gowaikar and B. Hassibi. Statistical pruning for near-maximum likelihood decoding. *IEEE Transactions on Signal Processing*, 55(6):2661–2675, 2007.

[31] P. M. Gruber and J. M. Wills, editors. *Handbook of convex geometry*. North-Holland, Amsterdam, 1993.

[32] Z. Guo and P. Nilsson. Algorithm and implementation of the k-best sphere decoding for mimo detection. *IEEE Journal on Selected Areas in Communications*, 24(3):491–503, 2006.

[33] V. Guruswami, A. Sahai, and M. Sudan. soft-decision decoding of chinese remainder codes. In *Proceedings. 41st Annual Symposium on Foundations of Computer Science, Redondo Beach, CA, Nov. 2000*, pages 159–168.

[34] S. Han, T. Cui, and C. Tellambura. Improved k-best sphere detection for uncoded and coded MIMO systems. *IEEE Wireless Communications Letters*, 1(5):472–475, 2012.

[35] G. Hanrot and D. Stehlé. Improved analysis of kannan's shortest lattice vector algorithm. In *Proceedings of the 27th annual international cryptology conference on Advances in cryptology*, pages 170–186. Springer-Verlag, 2007.

[36] G. Hanrot, X. Xavier Pujol, and D. Stehlé. Algorithms for the shortest and closest lattice vector problems. In *IWCC'11 Proceedings of the Third international conference on Coding and cryptology*, pages 159–190, 2011.

[37] A. Hassibi and S. Boyd. Integer parameter estimation in linear models with applications to GPS. *IEEE Transactions on Singal Processing*, 46(11):2938–2952, 1998.

[38] B. Hassibi. An efficient square-root algorithm for BLAST. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 737–740, 2004.

[39] C. Hermite. Oeuvres. *Paris (Gauthier-Villars)*, 1917, 1905.

[40] J. Jaldén, L. G. Barbero, B. Ottersten, and J. S. Thompson. The error probability of the fixed-complexity sphere decoder. *IEEE Transactions on Singal Processing*, 57(7):2711–2720, 2009.

[41] J. Jaldén and B. Ottersten. On the complexity of sphere decoding in digital communications. *IEEE Transactions on Signal Processing*, 53(4):1474–1484, 2005.

[42] 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*, pages 2685–2688, 2008.

[43] R. Kannan. Improved algorithms for integer programming and related lattice problems. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pages 193–206, 1983.

[44] R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of operations research*, 12(3):415–440, 1987.

[45] S. Khot. Hardness of approximating the shortest vector problem in lattices. *Journal of the ACM*, 52(5):789–808, 2005.

[46] D. E. Knuth. *The Art of Computer Programming*. 2nd ed. Reading, MA: Addison-Wesley, 1981, vol. 2, 1981.

[47] A. Korkine and G. Zolotareff. Sur les formes quadratiques. *Mathematische Annalen*, 6(3):366–389, 1873.

[48] J.-K. Kuusinen, J. Sorsa, and M.-L. Siikonen. The elevator trip origin-destination matrix estimation problem. *to appear in Transportation Science*.

[49] J. C. Lagarias, H.W. Lenstra, and C. P. Schnorr. Korkin-zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica*, 10(4):333–348, 1990.

[50] K-C. Lai, C-C. Huang, and J-J Jia. Variation of the fixed-complexity sphere decoder. *IEEE Communications Letters*, 15(9):1001–1003, 2011.

[51] A.K. Lenstra, H.W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[52] Soung Chang Liew, Shengli Zhang, and Lu Lu. Physical-layer network coding: Tutorial, survey, and beyond. *Physical Communication*, 6:4–42, 2013.

[53] C. Ling and N. Howgrave-Graham. Effective LLL reduction for lattice decoding. In *IEEE International Symposium on Information Theory, 2007*, pages 196–200.

[54] C. Ling, W. Mow, and L. Gan. Dual-lattice ordering and partial lattice reduction for SIC-based MIMO detection. *IEEE Journal of Selected Topics in Signal Processing*, 3:975–985, 2009.

[55] C. Ling, W. Mow, and N. Howgrave-Graham. Reduced and fixed-complexity variants of the LLL algorithm for communications. *IEEE Transactions on Communications*, 61(3):1040–1050, 2013.

[56] S. Loyka and F. Gagnon. Performance analysis of the V-BLAST algorithm: an analytical approach. *IEEE Transactions on Wireless Communications*, 3(4):1326–1337, 2004.

[57] W-K. Ma, T. N. Davidson, K. M. Wong, Z-Q. Luo, and P-C. Ching. Quasi-maximum-likelihood multiuser detection using semi-definite relaxation with application to synchronous CDMA. *IEEE Transactions on Signal Processing*, 50(4):912–922, 2002.

[58] Zheng Ma, Bahram Honary, Pingzhi Fan, and Erik G. Larsson. Stopping criterion for complexity reduction of sphere decoding. *IEEE Communications Letters*, 13(6):402–404, 2009.

[59] D. Micciancio. The hardness of the closest vector problem with preprocessing. *IEEE Transactions on Information Theory*, 47(3):1212–1215, 2001.

[60] D. Micciancio. The shortest vector problem is NP-hard to approximate to within some constant. *SIAM Journal on Computing*, 30(6):2008–2035, 2001.

[61] D. Micciancio and O Regev. *Lattice-Based Cryptography*. Bernstein, D. J. and Buchmann, J. (eds.), Berlin: Springer Verlagem, 2008.

[62] D. Micciancio and P Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. *SIAM Journal on Computing*, 42(3):1364–1391, 2013.

[63] Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, March 2002.

[64] H. Minkowski. Geometrie der zahlen (2 vol.). *Teubner, Leipzig*, 1910, 1896.

[65] W. H. Mow. Maximum likelihood sequence estimation from the lattice viewpoint. *IEEE Transactions on Information Theory*, 40(5):1594–1600, 1994.

[66] Rubb I. Muirhead. *Aspects of Multivariate Statistical Theory*. New York: Wiley, 1982.

[67] B. Nazer and M. Gastpar. Compute-and-forward: Harnessing interference through structured codes. *IEEE Transactions on Information Theory*, 57(10):6463–6486, 2011.

[68] P. Q. Nguyen and T. Vidick. Sieve algorithms for the shortest vector problem are practical. *Journal of Mathematical Cryptology*, 2(2):181–207, 2008.

[69] P.Q. Nguyen and D. Stehlé. An LLL algorithm with quadratic complexity. *SIAM Journal on Computing*, 39(3):874–903, 2009.

[70] A. Novocin, D. Stehlé, and G. Villard. An lll-reduction algorithm with quasi-linear time complexity: extended abstract. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA*, pages 403–412. ACM, 2011.

[71] J. Richter, C. Scheunert, and E. Jorswieck. An efficient branch-and-bound algorithm for compute-and-forward. In *2012 IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, pages 77–82, Sept 2012.

[72] S. Sahraei and M. Gastpar. Compute-and-forward: Finding the best equation. In *to appear in 52nd Annual Allerton Conference on Communication, Control, and Computing, Champaign, Illinois, USA, October 1-3, 2014*.

[73] A. Sakzad, E. Viterbo, Yi Hong, and J. Boutros. On the ergodic rate for compute-and-forward. In *2012 International Symposium on Network Coding (NetCod)*, pages 131–136, 2012.

[74] C. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Mathematical Programming*, 66:181–191, 1994.

[75] C. P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53:201–224, 1987.

[76] C. P. Schnorr. A more efficient algorithm for lattice basis reduction. *Journal of Algorithms*, 9:47–62, 1988.

[77] M. Seysen. Simultaneous reduction of a lattice basis and its reciprocal basis. *Combinatorica*, 13(3):363–376, 1993.

[78] B. Shim and I. Kang. Sphere decoding with a probabilistic tree pruning. *IEEE Transactions on Signal Processing*, 56(15):4867–4878, 2008.

[79] N. Sommer, M. Feder, and O. Shalvi. Finding the closest lattice point by iterative slicing. *SIAM J. Discrete Math*, 23(2):715–731, 2009.

[80] K. Su and I. J. Wassell. A new ordering for efficient sphere decoding. In *IEEE International Conference on Communications,*, volume 3, pages 1906–1910, 2005.

[81] M. Taherzadeh, A. Mobasher, and A. K. Amir Khandani. Lll reduction achieves the receive diversity in mimo decoding. *IEEE Transactions on Information Theory*, 53(12):4801–4805, 2007.

[82] P. J. G Teunissen. *GPS carrier phase ambiguity fixing concepts. In Kleusberg A and Teunissen, P. J. G, editors, GPS for Geodesy, pp. 317-388.* Springer, Heidelberg.

[83] P. J. G. Teunissen. Success probability of integer GPS ambiguity rounding and bootstrapping. *Journal of Geodesy*, 72(10):606–612, 1998.

[84] P. J. G. Teunissen. An optimality property of integer least-squares estimator. *Journal of Geodesy*, 73(11):587–593, 1999.

[85] P. J. G. Teunissen. An invariant upper-bound for the GNSS bootstrappend ambiguity success rate. *Journal of Global Positioning Systems*, 2(1):13–17, 2003.

[86] P. van Emde Boas. Another NP-complete partition problem and the complexity of computing short vectors in a lattice. Technical report, Technical report 81-04,Mathematics Department, University of Amsterdam, 1981.

[87] H. Vetter, V. Ponnampalam, and P. A. Sandell, M.and Hoeher. Fixed complexity LLL algorithm. *IEEE Transactions on Signal Processing*, 57(4):1634–1637, 2009.

[88] H.-T. Wai, W.-K. Ma, and A.M.-C. So. Cheap semidefinite relaxation MIMO detection using row-by-row block coordinate descent. In *2011 IEEE Intl. Conf. Acoustic, Speech, and Signal Processing*, pages 3256–3259. IEEE.

[89] Lili Wei and Wen Chen. Compute-and-forward network coding design over multi-source multi-relay channels. *IEEE Transactions on Wireless Communications*, 11(9):3348–3357, 2012.

[90] J. Wen and X.-W. Chang. Success probability of the babai estimators for box-constrained integer linear models. *submitted to IEEE Trans. Inf. Theory.*

[91] J. Wen and X.-W. Chang. Two fixed-complexity LLL reduction algorithms. *in preparation.*

[92] J. Wen, B. Zhou, W. Mow, and X.-W. Chang. Compute-and-forward design based on improved sphere decoding. *to appear in ICC 2015.*

[93] J. Wen, B. Zhou, W. Mow, and X.-W. Chang. Compute-and-forward design based on improved sphere decoding. *to be submitted to IEEE Transactions on Signal Processing.*

[94] X. Wu and J.S. Thompson. Accelerated sphere decoding for multipleinput multiple-output systems using an adaptive statistical threshold. *IET Signal Processing*, 3(6):433–444, 2009.

[95] D. Wübben, R. Bohnke, J. Rinas, V. Kuhn, and K. Kammeyer. Efficient algorithm for decoding layered space-time codes. *Electronics Letters*, 37(22):1348–1350, 2001.

[96] D. Wübben, D. seethaler, J. Jaldén, and G Matz. Lattice reduction: A survey with applications in wireless communications. *IEEE Transactions on Magazine*, 28(3):79–91, 2011.

[97] Xie. X. *Theory and Algorithms for Some Integer Least Squares Problems.* PhD thesis, School of Computer Science, McGill University, 2014.

[98] X. Xie, X.W. Chang, and M. Al Borno. Partial LLL reduction. In *the Proceedings of IEEE GLOBECOM 2011, 5 pages.*

[99] R. Zamir. Lattices are everywhere. In *Information Theory and Applications Workshop*, pages 392–421, 2009.

[100] W. Zhang, S. Qiao, and Y. Wei. HKZ and Minkowski reduction algorithms for lattice-reduction-aided MIMO detection. *IEEE Transactions on Singal Processing*, 60(11):5963–5976, 2012.

[101] B. Zhou and W. Mow. A quadratic programming relaxation approach to compute-and-forward network coding design. In *The 2014 IEEE International Symposium on Information Theory (ISIT'2014)*, pages 2296–2300, 2014.

[102] B. Zhou, J. Wen, and W. Mow. A quadratic programming relaxation approach to compute-and-forward network coding design. *to be submitted to IEEE Transactions on Wireless Communications*.

[103] H. Zhu, W. Chen, B. Li, and F. Gao. An improved square-root algorithm for V-BLAST based on efficient inverse cholesky factorization. *IEEE Transactions on Wireless Communications*, 10(1):43–48, 2011.